THÈSE

présentée à l'École Normale Supérieure de Cachan

pour obtenir le grade de

Docteur de l'École Normale Supérieure de Cachan

par : Vincent Cheval

Spécialité : Informatique

Automatic verification of cryptographic protocols : privacy-type properties

Soutenance prévue le 03 décembre 2012

Composition du Jury :

— Bruno Blanchet	examinateur
— Hubert Comon-Lundh	directeur de thèse
— Stéphanie Delaune	directrice de thèse
— Ralf Küsters	rapporteur
— Jean-Yves Marion	examinateur
— Alwen Tiu	rapporteur

Thèse préparée au sein du Laboratoire Spécification et Vérification (LSV).

Résumé

Plusieurs outils ont été développé pour vérifier automatiquement les propriétés de sécurité sur des protocoles cryptographiques. Jusqu'à maintenant, la plupart de ces outils permettent de vérifier des propriétés de trace (ou propriétés d'accessibilité) tel que le secret simple ou l'authentification. Néanmoins, plusieurs propriétés de sécurité ne peuvent pas être exprimées en tant que propriétés de trace, mais peuvent l'être en tant que propriétés d'équivalence. L'anonymat, la non-traçabilité ou le secret fort sont des exemples classiques de propriétés d'équivalence.

Typiquement, deux protocoles P et Q sont équivalent si les actions d'un adversaire (intrus) ne lui permettent pas de distinguer P de Q. Dans la littérature, plusieurs notions d'équivalence ont été étudié, par exemple l'équivalence de trace ou l'équivalence observationnelle. Néanmoins, ces équivalences se révèlent être très difficiles à démontrer, d'où l'importance de développer des outils de vérification automatique efficaces de ces équivalences.

Au sein de cette thèse, nous avons dans un premier temps travaillé sur une approche reposant sur des techniques de résolution de contraintes et nous avons créé un nouvel algorithme pour décider l'équivalence de trace entre deux protocoles pouvant contenir des conditionnelles avec branches "else", et pouvant également être non-déterministe. Cet algorithme a donné naissance à l'outil APTE (Algorithm for Proving Trace Equivalence) et a été appliqué sur des exemples concrets comme le "Private authentification protocol" ainsi que le "E-passport protocol".

Cette thèse propose également des résultats de composition pour l'équivalence de trace. En particulier, nous nous sommes intéressés à la composition parallèle de protocoles partageant certains secrets. Ainsi dans cette thèse, nous avons démontré que, sous certaines conditions, la composition parallèle de protocoles préserve les propriétés d'équivalence. Ce résultat fut appliqué au "E-passport protocol".

Enfin, cette thèse présente une extension à l'outil de vérification automatique ProVerif afin de démontrer automatiquement plus de propriétés d'équivalence. Cette extension a été implémenté au sein de ProVerif ce qui a permis de démontrer la propriété d'anonymat pour le "Private authentification protocol".

Abstract

Many tools have been developed to automatically verify security properties on cryptographic protocols. But until recently, most tools focused on trace properties (or reachability properties) such as authentication and secrecy. However, many security properties cannot be expressed as trace properties, but can be written as equivalence properties. Privacy, unlinkability, and strong secrecy are typical examples of equivalence properties.

Intuitively, two protocols P, Q are equivalent if an adversary cannot distinguish P from Q by interacting with these processes. In the literature, several notions of equivalence were studied, e.g. trace equivalence or a stronger one, observational equivalence. However, it is often very difficult to prove by hand any of these equivalences, hence the need for efficient and automatic tools.

We first worked on an approach that rely on constraint solving techniques and that is well suited for bounded number of sessions. We provided a new algorithm for deciding the trace equivalence between processes that may contain negative tests and non-determinism. A new tool called APTE (Algorithm for Proving Trace Equivalence) was born from this algorithm and we applied our results on concrete examples such as anonymity of the Private Authentication protocol and the E-passport protocol.

We also investigated composition results. More precisely, we focused on parallel composition under shared secrets. We showed that under certain conditions on the protocols, the privacy type properties are preserved under parallel composition and under shared secrets. We applied our result on the e-passport protocol.

At last this work presents an extension of the automatic protocol verifier ProVerif in order to prove more observational equivalences. This extension have been implemented in ProVerif and allows us to automatically prove anonymity in the private authentication protocol.

Contents

Chapter 1

Introduction

1.1 Security challenge

Successfully exchanging secret information has always been a difficult and important task. Several issues raise when secretly communicating with someone from a distance. Can the sender be sure that the expected receiver of a message actually receives it? Was someone able to intercept the message, and if there is one, was he able to understand the message? Ensuring that such communication successfully holds can have a huge impact. For example, during the second world war, the german army used a machine, called Enigma, for the generals to encrypt and decrypt the orders they convey to their army. The allied forces spent huge amount of resources to gather information, codenamed *Ultra* by the British army, by breaking many messages that had been enciphered using the Enigma. Winston Churchill told King George VI after the war: "It was thanks to Ultra that we won the war".

With the emergence of personal computers in the eighties and the internet in the nineties, privately and secretly conveying informations has become a problem in our daily life. All the international economy now goes through internet, most of the transactions of the stock market are done online and all banks provide services that allow a client to access his bank account, make transaction,... over the internet.

This last decade also witnesses the raise of wireless technology. The number of electronic devices that communicate wirelessly in a single house increased drastically (*e.g.* smartphone, electronic tablet, computer, printer, internet box). Even though users profit from the multiple interactions between all the different devices, they also present serious security risks that are regularly reported by the media. For instance, in 2009, during the creation of the french governmental organisation Hadopi, concerns were raised in the public opinion about the capability of one to protect and prove the security of his wireless internet connection.

The emergence of small communication devices (*e.g.* RFID tags, mobile phone) that we carry almost all the time also raises the issue of traceability. Although the traceability of mobile phones is already part of the cinematographic folklore, recent work reveals that the current communication protocols between a mobile phone and its operator present major security flaws [AMR+12].

Even simple electronic devices might present a security risk. For example, the current paper passports also contain a RFID chip that stores the critical information printed on the passport. Such a passport is called an *electronic passport* or *e-passport*. It facilitates the checks at the airport border and also ensures the authenticity of the passport. Although there exists an international specification for such an electronic passport, parts of the implementation only depend on the country that delivers the passport. Hence, recent studies have highlighted that the first versions of the French electronic passports could be easily traced [ACRR10] whereas the English electronic passports were not subject to such a flaw.

The previous examples are not of course an exhaustive list of applications whose security are critical for the privacy of their users. In fact, new applications are still emerging in order to face

the society needs and present new security challenges. It is therefore crucial to ensure that the applications we use are indeed secured.

1.2 Cryptographic protocols

1.2.1 Cryptographic primitives

Until few decades ago, the transmission of secret information consisted of modifying a message using small algorithms so that it becomes unreadable to anyone but the sender and the intended receiver of this message by relying on a secret key that they agreed on before hand. A *symmetric encryption* algorithm allows the sender to encrypt his message using the shared secret key, whereas a *symmetric decryption* algorithm allows the reader to decrypt the cipher using the shared secret key (see Figure 1.1). Ideally, a third party cannot understand (or decrypt) the cipher if he does not know the secret key shared between the sender and the receiver. Several symmetric encryption schemes were invented over the years. One of the oldest is the *Caesar cipher* which was supposed to be used by Julius Caesar. Typically, this encryption scheme consists of replacing the letters in the text by shifting letters in the alphabet, the length of the shift being the encryption key. Hence, "hello word" would be encrypted into "khoor zrug" with the key 3, *i.e.* $A \to D$, $B \to E$, ..., $Z \to C$. To decrypt this message, it suffices to reverse the shift of letters, *i.e.* $D \to A$, $E \to B$, ..., $C \to Z$. This encryption scheme and the slightly more sophisticated methods were shown to be easily breakable by using statistical analysis.

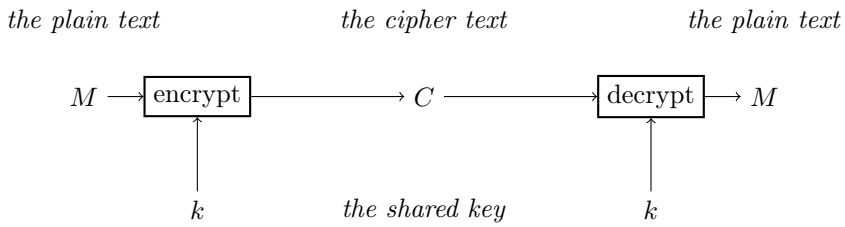


Figure 1.1: Symmetric encryption scheme

The security of a symmetric encryption scheme relies on the length of the key. Indeed, with the advent of powerful computers, an intruder might guess the shared secret key by trying every possibility, called a *brute-force attack*. In 1977, the United States Federal Information Processing Standard adopted the Data Encryption Standard (DES). While this encryption scheme was internationally used, this encryption method was shown to be vulnerable to brute-force attack due to the length of the keys. This encryption scheme was later on replaced by the AES (Advanced Encryption Standard) but DES is still used nowadays in several applications (*e.g.* ATM, email) usually in combination with an other encryption schemes.

One of the downsides of the symmetric encryption schemes is the key-management. Indeed, each pair of network members has to share a secret key, hence the number of required keys increases quadratically with the number of network members. Moreover, when two new members want to communicate, they first have to agree on the shared key, which can only be done if they already share some secret key or have access to a *secure channel* (*e.g.* by meeting face-to-face). Hence, this leads to the problem of "a dog chasing its tail". In 1976, W. Diffie and M. Hellman invented the notion of *asymmetric encryption schemes*. In such encryption schemes, each member of the network possesses a pair of keys, one private only known by the owner and used to decrypt ciphers, and the other one public shared to anyone in the network and used to encrypt messages (see Figure 1.2). Even though the keys are mathematically related, an intruder cannot deduce the private key of a member from its public key. For example, in the Diffie-Hellman asymmetric

encryption scheme, this property is ensured by the fact that the discrete logarithm problem is considered as difficult.

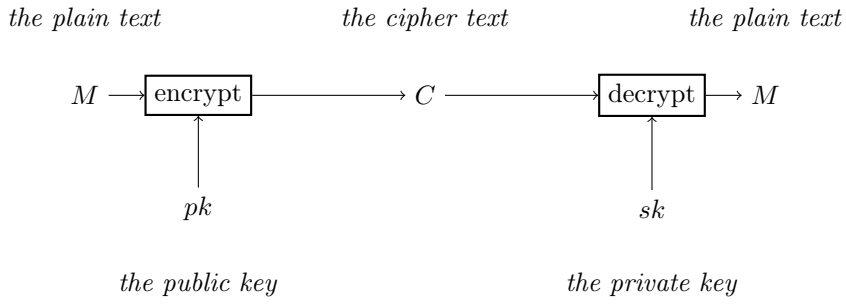


Figure 1.2: Asymmetric encryption scheme

This encryption scheme has the advantage to solve the problem of key-management since on the one hand, the number of required keys increases linearly in the number of network members and, on the other hand, two members do not have to agree on a shared key beforehand (they only have to send each other their own public key). Over the time, new asymmetric encryption schemes have been proposed such as RSA, invented by R. Rivest, A. Shamir, and L. Adleman in 1978.

However, the computational cost of most of the existing asymmetric encryption schemes is too high to be used in all applications (*e.g.* nobody would want to wait several minutes to get a 20 euros bill at the ATM). Thus, *key-establishment protocols* have been developed such as the Diffie-Hellman key exchange protocol [DH76]. In such protocols, two members of the network first communicate using an asymmetric encryption scheme in order to establish a fresh secret key for a symmetric encryption scheme *e.g.* AES. Once the shared key is established, the two members can communicate using the symmetric encryption scheme. These fresh secret keys are called *symmetric session keys*.

While the encryption schemes are designed to prevent anyone to understand a message unless the decryption key is known, another cryptographic primitive called *digital signature* was designed to demonstrate the authenticity of message. A valid signature gives some reason to believe that the message was created by a known sender, and was not altered during the communication. In 1985, T. Elgamal proposed a signature scheme based on discrete logarithms [Gam85], which is also based on a pair of public and private keys. Typically, a participant can sign a document using his private signing key and anyone can verify the signed document using the public key of the participant (see Figure 1.3). In contrast to the asymmetric encryption scheme, a signed document does not necessarily hide the content of the message.

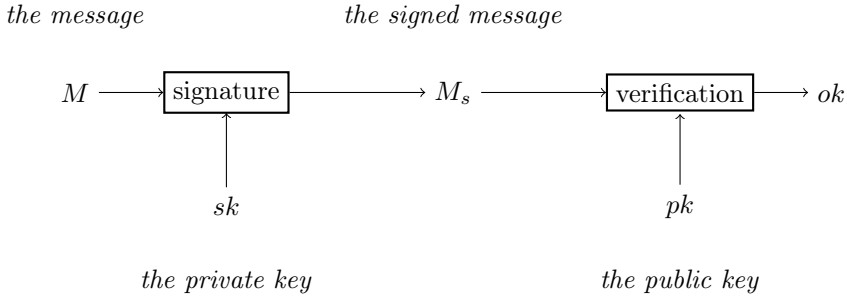


Figure 1.3: Digital signature scheme

Another cryptographic primitive called *cryptographic hash function* was designed to demon-

strate the integrity of a message. Typically, a hash function is a one-way function that takes an arbitrary message and returns a fixed-size *bitstring*, *i.e.* a sequence of 0 and 1. Ideally, one cannot recover the content of a message from its hash value. Moreover, an hash function should be *collision resistant*, *i.e.* it should be difficult to find two messages with the same hash value. Of course, this last property depends on the length of the bitstring returned by the hash function. Typical examples of hash functions used nowadays are the MD5 and SHA-1. Similar cryptographic primitives exists, called *message authentication codes* (or MAC), that allow one to authenticate the hash value thanks to a secret key.

The cryptographic primitives previously described are probably the most common primitives used in practice. In fact, these primitives are also the ones that will be used the most in this thesis. There exist several other primitives that are specific to an application such as blind signatures and trap door commitments for the electronic voting.

1.2.2 Protocols

Although it is essential to develop cryptographic primitives that are unbreakable, it is not sufficient to ensure the security of the whole communications between members of the networks. Even if an intruder is not able to break the cryptographic primitives, he can still gain some crucial knowledge or mislead a participant by intercepting, modifying and comparing the messages that are sent over the network. Hence it is crucial to analyse and verify not only the cryptographic primitives but also the complete communications between members of the network.

To that extend, we rely on *cryptographic protocols* that describe the computations and communications between the different participants. For illustration purposes, we consider the communication protocols specified by the International Civil Aviation Organisation (ICAO) standard [ICA04] and used for the electronic passports. In particular, the ICAO standard indicates that before anything else, the reader (*e.g.* a customs officer) has to recover some keys ke and km that are printed on the passport at the same page as the personal data and picture (usually, the customs officer recovers them by optically scanning the passport). Then the reader and the e-passport must execute the Basic Access Control (BAC) protocol in order to establish session keys to prevent skimming and eavesdropping on the subsequent communications with the e-passport. The *Alice & Bob* representation of this protocol is given in Figure 1.4.

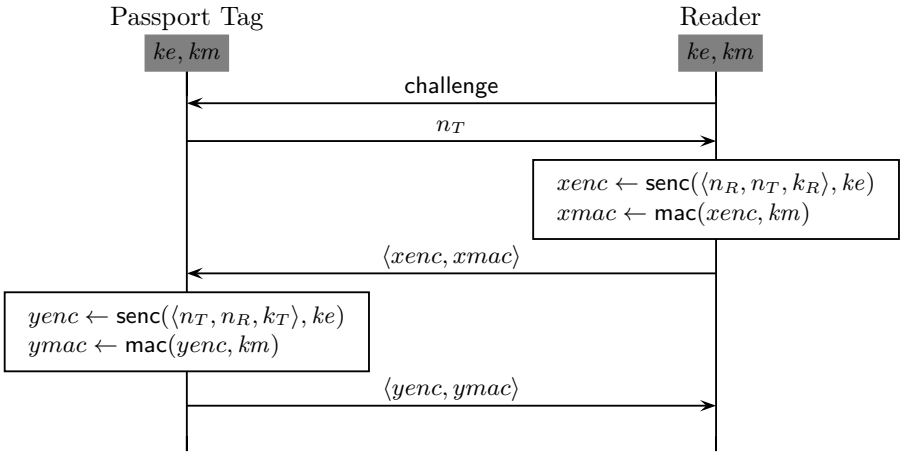


Figure 1.4: Basic Access Control protocol

$\mathsf{senc}(m, k)$ represents the application of a symmetric encryption scheme on the message m with the secret key k. The choice of the symmetric encryption scheme is not relevant, as long

as it satisfies the desired properties. In fact, in the ICAO standard, the choice of the symmetric encryption scheme is up to the state implementing the electronic passport. $\mathsf{mac}(m, k)$ represents a message authentication codes of the message m with the secret key k. $\langle m_1, \ldots, m_n \rangle$ represents the pairing of messages.

In this protocol, the participants proceed as follows: The reader starts by sending a challenge to the passport. Upon receipt of such a challenge, the passport creates a fresh nonce, *i.e.* a random bitstring, denoted n_T, and sends it to the reader. Once the reader receives n_T, he also creates two fresh nonces, denoted n_R and k_R, and encrypts the pairing of the three nonces with the secret key ke. The reader then creates a MAC of the cipher using the secret key km and sends both MAC and cipher to the passport. The passport first verifies the integrity of the message he received by computing himself a MAC of the cipher he received with the secret km and comparing with the received MAC. Then it decrypts the cipher using the secret key ke and retrieves the three nonces from the result of the decryption. After checking that the first nonce indeed corresponds to the nonce n_T that it generated at the beginning, the passport generates a fresh nonce k_T and sends back to the reader the encryption of the three nonces n_T, n_R and k_T by the secret key ke with the MAC of such an encryption using the secret key km. After checking the integrity of the MAC, the reader checks the nonce n_R and retrieves the nonce k_T by decrypting the cipher with the secret key ke.

Hence at the end of such a protocol, both participants, *i.e.* the reader and the passport, know the values of the nonces k_T and k_R that will be used later on to extract key sessions. The security of such a protocol should be ensured by the checks of the MACs and the nonces n_T and n_R respectively by the passport and the reader.

1.2.3 Security properties

Verifying the security of a cryptographic protocol depends on the goal of such a protocol. For example, the ICAO standard indicates that the data in the e-passport cannot be accessed except by an authorised person or organisation. Hence, when describing a cryptographic protocol, it is crucial to properly describe the *security properties* that such a protocol is supposed to achieve. Although each cryptographic protocol might have its own security properties, there exist classical security properties that are grouped into two different families: the *trace properties* and the *equivalence properties*. Intuitively, the trace properties, such as authentication, specify that the protocols cannot reach a bad state, whereas the equivalence properties specify the indistinguishability of some instances of the protocols. In the literature, most of the work and tools focused on the trace properties. Fewer works focus on equivalence properties and even fewer tools can automatically verify equivalence properties. In this thesis, we will focus on equivalence properties and more specifically on the following ones:

Anonymity In most of protocols, there is no guarantee concerning the privacy of the participants. Typically, a participant disclosing his identity to an intruder is not considered as a flaw in the protocol. However, for some applications such as electronic voting, requiring that the identity of the voters cannot be disclosed is a necessity. Informally defined by the ISO/IEC standard 15408 [ISO09], anonymity is the property ensuring that *a user may use a service or a resource without disclosing the his(her) identity.* The *private authentication protocol* [AF04], was specifically designed to ensure mutual authentication between participants but also to ensure the anonymity of the participants.

Unlinkability A protocol satisfying the anonymity property may still disclose informations on a user by allowing an intruder to track several uses of this protocol by the same user. Such informations might at the end allow the intruder to deduce or at least restrict the possible identities of a user. Unlinkability is the property ensuring that *a user may make multiple uses of a service or a resource without others being able to link these uses together.* For example, being able to track the different purchases of a user on an online-shop may reveal the habits of a user. In an

other area, the french subway company of Paris provides cards that contain RFID chip to their clients in place of the underground tickets. However, if the protocol used to communicate with the RFID chip does not satisfy the unlinkability property then an intruder might be able to track the daily movement of a user thus deducing for example his place of residence or workplace. The Basic Access Control protocol on e-Passports was designed, according to the ICAO standard, to make the passports unlinkable.

1.3 Difficulties of security verification

Verifying the security of a cryptographic protocol is a difficult task. Compared to trace properties, verifying equivalence properties is even more difficult.

1.3.1 General difficulties

The degree of security that we verify on a protocol is directly correlated to the capabilities (or power) that we give to the intruder. As such, verifying a security property for a *passive intruder* (*i.e.* when the intruder can only eavesdrop the communications between the participants) provides less guarantees on the security than the case where we consider an *active intruder* (*i.e.* when the intruder can intercept and modify the communications between the participants). However, the more capabilities we give to the intruder, the more difficult it is to verify some security property on a protocol. In fact, in the general case, deciding whether a protocol satisfies some security property is undecidable ([DLMS99, AC02]) which is mainly due to the many sources of unboundedness in the modelling of the capabilities of the intruder. We describe below some of these sources.

Number of sessions When verifying the security of a protocol, it is reasonable to consider that the attacker can interact with any number of sessions of the protocol. In the case of the e-passport protocol, the number of sessions would correspond to the number of people going through customs which is usually a lot in any airport. However, since considering unbounded number of sessions usually leads to undecidability, many works present decidability result for verification of security properties for a bounded number of sessions of the protocol [MS01, Bau05, CD09a]. Note that even when considering a bounded number of sessions, verifying the security of a protocol is still difficult since an intruder can intercept messages and replace them by any message that he can compute himself, which represent an infinite number of messages.

Cryptographic primitives As previously mentioned, there exists many different cryptographic primitives, all of them having different behaviours, and some being more difficult to model than others. For example, it is more difficult to verify some security property on a protocol that relies on blind signatures rather than a protocol relying on symmetric encryption only. As such, decision procedures usually fix or at least limit the the number of cryptographic primitives they can handle.

Environment Most of the time, security properties are ensured when considering the cryptographic protocol in isolation. Indeed, no one can predict all the protocols that are running on a network at the same time. Furthermore, real life protocols are usually complex and composed of several sub-protocols that rely on the same cryptographic material. For example, the UMTS standard [3GP11, 3GP10b, 3GP10a] specifies tens of sub-protocols running in parallel in 3G mobile phone systems. And, while one may hope to automatically verify each of these sub-protocols in isolation, it is unrealistic to expect that the whole suite of protocols can be automatically checked.

On the other hand, we still want some guarantees on the security of the protocols when placed in an environment with other protocols running at the same time, hence the need for secure composition of cryptographic protocols. Whereas securely composing protocols when they do not share any secret is an easy task, it is unrealistic to assume that a user always uses a different password for each protocol. Even if it was the case, some applications rely, as previously mentioned, on several sub-protocols that share passwords by construction.

1.3.2 Specificities of equivalence properties

As previously mentioned, equivalence properties intuitively indicate the indistinguishability of some instances of the protocol whereas trace properties indicate that the protocol cannot reach a bad state. More formally, when verifying a trace property, we have to verify that any execution of the protocol cannot reach a bad state. On the other hand, when verifying an equivalence property, we have to verify that for any execution of one of the instances of the protocol there exists an indistinguishable execution of the other instance of the protocol. Hence, the alternation of quantifiers makes even more difficult the verification of equivalence properties.

Even the modelling of indistinguishability is a difficult task and requires an extensive knowledge of the capabilities of the intruder. Indeed, depending on what a real intruder can observe (*e.g.* messages sent over the network, internal computation of honest participants), two executions might not always be considered as indistinguishable. Therefore in the literature, several notions of indistinguishability have been proposed (*e.g.* trace equivalence, may-equivalence, observational equivalence [MNP02, AG99, AF01]) and the question of knowing which notion is best-suited to model an intruder is disputable.

1.4 Automatic verification using symbolic models

One of the first task when verifying the security of a cryptographic protocol is to create a model that represents the behaviour of the protocol and also the behaviour of an intruder.

1.4.1 Existing models

Early in the eighties, two different kinds of models emerge. This first ones, called *computational models*, consider the messages sent over the network as bitstrings. Moreover, the behaviour of the intruder is modelled as any probabilistic polynomial-time Turing machine. The cryptographic primitives are also represented as polynomial algorithms. These models, since they are close to the reality, offer strong guarantees on the security of cryptographic protocols. However, the proofs are usually difficult, error prone and are almost impossible to automate.

In the second kind of models, called *symbolic models*, the behaviour of the protocols and intruders are abstracted. Typically, the messages are abstracted by terms and the cryptographic primitives, assumed to be perfect (*i.e.* unbreakable), are abstracted by function symbols, hence symbolic models are much simpler than computational models. As such, the verification of security properties on a protocol using a symbolic model also becomes simpler and can be automated. In the literature, there exists dozen of symbolic models, (*e.g.* spi-calculus [AG99], strand space [FHG99]) that usually correspond to the need of a specific application or a theoretical result. For example, symbolic models based on *constraint systems* (*e.g.* [MS01, CLS03]) were shown to be well-adapted to deal with a bounded number of sessions whereas symbolic models based on *Horn clauses* (*e.g.* [Wei99, Bla01, CLC03]) are more adapted to an unbounded number of sessions.

However, because of these abstractions, the guarantees provided by the verification of a security property using symbolic models is usually weaker than the guarantees provided by the verification of the same security property using computational models. Typically, whereas an attack in symbolic models implies an attack in computational models, the conserve is not necessary true. Hence when creating a cryptographic protocol, one could first rely on the automatic tools offered by symbolic models to quickly detect possible attacks and so modify accordingly the protocol. Then when no more attack is detected by symbolic models, one can rely on a computational model to conclude. Recently, several works have attempted to derive conditions under which the symbolic security implies the computational security, starting with M.Abadi and Ph. Rogaway in 2000 [AR00]. Although the first results mainly targeted trace properties, H. Comon-Lundh and V. Cortier [CC08] showed that computational indistinguishability in presence of an active intruder is implied by the *observational equivalence* of the corresponding symbolic protocol. This research area is crucial since it allows one to use the automatic tools developed in the symbolic models to obtain the strong securities of the computational models.

1.4.2 Existing results

As previously mentioned, verifying the security of a cryptographic protocol is a difficult task even in symbolic models. Thus, all existing decidability results always consider restrictions of some sort. The most common one consists of bounding the number of sessions of the protocol. Other restrictions consist of fixing the cryptographic primitives that are considered or even studying stronger notions of indistinguishability than the one modelling the capabilities of the intruder (*e.g.* by giving too much power to the intruder) in the case of equivalence properties.

Automating the decision of equivalence properties. The automated verification of equivalence properties for security protocols was first considered in [Hut02] (within the spi-calculus). [Bau05, CD09a] gives a decision procedure for the trace equivalence of bounded deterministic processes. In particular, the processes are restricted to be determinate and do not contain (non trivial) conditional branching. Furthermore, the procedure seems not be well-suited for an implementation. [CR12] gives a decision procedure for the trace equivalence of bounded processes for a class of primitives (that are defined by a subterm convergent rewrite system). The procedure is probably not well-suited for an implementation. Furthermore, again only determinate processes and trivial conditional branching are allowed.

Tools. Several techniques and tools have been designed for the formal verification of cryptographic protocols. For instance CSP/FDR [RSG+00], PROVERIF [Bla01], SCYTHER [Cre08], AVISPA [Vig06] and others. However, most results and tools only consider trace properties. To our knowledge, there are only three tools that can handle equivalence properties: PROVERIF [Bla01], SPEC [TD10] and AKISS [Cio11]. Note that the last two tools were built in parallel of this thesis thus ProVerif was the only existing tool at the beginning of this thesis.

The tool PROVERIF [Bla01] originally was designed to prove trace properties but it can also check some equivalence properties (so-called diff-equivalence) [BAF08] that are usually too strong to model a real intruder. Note that this is the only tool that can handle an unbounded number of sessions of a protocol with a large class of cryptographic primitives in practice. However, PROVERIF is not a decision procedure in the sense that it may not terminate, especially if the equational theory modelling the cryptographic primitives is complex such as exclusive-or. Nevertheless, this tool was used by several researchers to prove the security properties of protocols (e.g. proving the absence of guessing attacks in EKE or proving the core security of JFK [BAF08], privacy properties for electronic voting protocol [DKR09]).

The tool SPEC [TD10] gives a decision procedure for open-bisimulation for bounded processes in the spi calculus. This procedure has been implemented. The scope is however limited: open-bisimulation coincides with trace equivalence for determinate processes only and the procedure also assumes a fixed set of primitives (symmetric encryption and pairing), and a pattern based message passing, hence, in particular, no non-trivial conditional branching.

More recently, the tool AKISS [Cio11] was developed in order to decide the *trace-equivalence* of bounded processes that do not contain non-trivial conditional branching. Although this tool was proved to be sound and complete, the algorithm was only conjectured to terminate for subterm convergent equational theory.

Composition results. As previously mentioned, verifying some security property when considering several protocols running at the same time is usually too difficult, even when relying on the automatic tools previously presented. Indeed, each tool requires a lot of computation time to prove some equivalence properties, without taking into consideration the cases of non-termination. Hence, there are a number of papers studying the secure composition of security protocols in symbolic models (*e.g.* [GT00, CC10]) and in computational models (*e.g.* [Can01, KT11]). Actually, a lot of results have been established for trace-based security properties, *e.g.* [GT00, ACS+08, MV09, CC10].

Regarding equivalence-based properties, it has been shown that composition works for resistance against guessing attacks in the passive case without any additional hypothesis [DKR08], and in the active case when the protocols are tagged [DKR08, CDK11].

Other works study *universal composability of protocols* [Can01]. This approach consists of defining for each sub-protocol an *ideal functionality* and then showing that a certain implementation securely emulates the ideal functionality. Since this initial work, the universal composability framework has been improved in several ways, *e.g.* with joint states [BCNP04], without pre-established session identifiers [KT11].

1.5 Limitations of existing results

Despite all the existing results, none of them can decide equivalence properties such as unlinkability on the Basic Access Control protocol described in Section 1.2.2. Moreover, the ICAO standard indicates that after a successful execution of the BAC protocol, several other sub-protocols can be executed in any order. However, no existing composition result can handle unlinkability for such sub-protocols. Indeed, the previous composition results in symbolic models [DKR08, CDK11] assume that passwords are the only shared secrets and are not well-suited to analyse privacy-type properties such as unlinkability. Hence the need to study the secure composition of protocols for privacy-type properties.

Furthermore, from all the decision procedures (implemented or not) we described in the previous section, only ProVerif is able to handle non-trivial conditional branching. However, coming back to the Basic Access Control (BAC) protocol, the ICAO standard indicates that if the passport fails to check the MAC or the value of the nonce n_T then it must output an error message. On the other hand, it does not specify what the error message should be. Since the output of an error message implies some non-trivial conditional branching in the protocol, only ProVerif can take as input the Basic Access Control protocol.

One could argue that such error messages could be dismissed of the BAC protocol when verifying some security properties but [ACRR10] showed that the early versions of French passports did not satisfied the unlinkability property due to these error messages.

Attack on the French e-passport. [ACRR10] experimented on French, English, German, Russian and Irish e-passports and discovered that for all of these passports but the French ones, an error code "6300" is sent over the network when the check of the MAC or the nonce n_T fails. However, in the case of the French passports, [ACRR10] indicates that an error code "6300" is sent when the check of the MAC fails and a different error code "6A80" is sent when the check of the value of n_T fails. The difference between these two error codes yields an attack on the unlinkability property. The intruder proceeds as follows:

First the intruder eavesdrops an execution of the BAC protocol between a reader and the passport he wants to track. Thus, from this execution, he obtains the message $\langle m, \mathsf{mac}(m, km_p)\rangle$ that was send by the reader to the passport where $m = \mathsf{senc}(\langle n_R, n_T, k_R\rangle, ke_p)$, ke_p, km_p being the keys of the targeted passport and n_T is the random nonce generated by this passport. Then, when a new passport comes along, the intruder plays the role of a reader by sending a challenge to this passport. After discarding the freshly generated nonce that the passport sent and denoted n'_T, the intruder replays the message $\langle m, \mathsf{mac}(m, km_p)\rangle$. When receiving this message, the new passport will necessary send an error message since either this passport is not the targeted passport (*i.e* the check of the MAC will fail) or the check of the nonce n'_T will fail (n_T being different from n'_T). However, in the case of French passports, if the error code "6A80" is outputted then it means that the check of the MAC succeeds and so the key used to compute the MAC is the key km_p of the targeted passport. Thus by checking if the error code is either "6A80" or "6300", the intruder can determine when the targeted passport executes the BAC protocol.

[ACRR10] showed that English passports satisfy the unlinkability property. It is therefore crucial to include the error message in the description of the protocol thus to consider non-trivial

conditional branching, which discards all the existing procedures other than ProVerif. However, the notion of equivalence decided by ProVerif is stronger than the usual behavioural equivalence used to model the unlinkability property. Thus, ProVerif will fail to prove unlinkability on English passports and yield a false attack. Even in the case of French passports, ProVerif also yields a false attack.

In fact, these limitations are not specific to the e-passport protocol and can be found in several case studies, *e.g.* the *private authentication* protocol [AF04] and the *mobile telecommunication* protocols [AMR+12].

1.6 Contributions

This thesis is organised in three parts. In Part I, we study the privacy-type properties in the applied pi calculus which are modelled by an equivalence between processes and we show that the equivalence between processes compose well, provided some conditions on the protocols. In Part II, we show that deciding the equivalence between processes is possible for a class of bounded protocols with standard primitives. At last, in Part III, we propose an extension to the existing tool ProVerif that removes some specific false attacks. We detail our contributions below.

1.6.1 Privacy-type properties in the applied pi calculus

In Chapter 3, we show how equivalence properties of cryptographic protocols can be expressed in the *applied pi calculus* [AF01] by means of a behavioural equivalence on processes. More specifically, we formalise the resistance to guessing attack, strong secrecy, anonymity and unlinkability properties, and we provide with concrete examples of cryptographic protocols to illustrate the equivalence properties. In particular, we describe in detail the different protocols that compose the *e-passport protocol* and show how anonymity and unlinkabiltity are expressed in this protocol.

We also show the relations between classical behavioural equivalences, *i.e.* trace equivalence, observational equivalence and may-testing equivalence. More specifically, we show that although may-testing equivalence and trace equivalence are two very close notions, they are not equivalent. We prove that trace equivalence indeed implies may-testing equivalence but we provide with an example of processes that are may-testing equivalent but not trace equivalent. However, we show that trace equivalence and may-testing equivalence coincide for *image-finite processes* (*e.g.* processes without replication).

1.6.2 Composing trace equivalence in a modular way

The last chapter of Part I is devoted to the study of the preservation of trace equivalence under parallel composition of processes. We identify sufficient conditions of *disjointness*, under which protocols can "safely" be executed in parallel. We first state a composition result that also allows the protocols to share the usual cryptographic primitives of symmetric and asymmetric encryption, hashing, and signing, provided that these primitives are tagged and that public and verification shared keys are not derivable. In this setting, we are able to establish a strong result that basically says that the disjoint scenario, *i.e.* where protocols do not share any secret, is equivalent to the shared one, *i.e.* where protocols share some secrets. This allows us to go back to the disjoint case for which composition works unsurprisingly well.

Then, we further relax this condition. A second theorem shows that it is possible to compose protocols that share public and verification keys even if those are known by the attacker, provided that they are given to him from the beginning. However, in our setting such a sequence has to be finite, and thus our result can only be applied in presence of a bounded number of public shared keys. This is not a real limitation for the analysis of the e-passport application, but this could lead us to an unrealistic situation for some other applications.

1.6.3 A decision procedure for trace equivalence

Part II is dedicated to a decision procedure for the trace equivalence between bounded processes. We allow processes with non-trivial else branches and non-deterministic choice. Our procedure is sound, complete and always terminate. In contrast to the tools ProVerif and AKiSs, our algorithm only accepts a fixed set of primitives other than one-way functions, namely symmetric and asymmetric encryption, digital signature and pairing. In addition, any one-way function is accepted such as hash function. The behaviour of our cryptographic primitives are expressed using rewriting systems and we assume that our destructors may fail.

In Chapter 6, we reduce the problem of trace equivalence to the problem of symbolic equivalence between sets of constraint systems. A similar reduction is done in Chapter 4 for the *applied pi calculus*. Due to the restricted sets of cryptographic primitives we consider, we show, in Chapter 6, that it is possible to refine the symbolic equivalence by only considering canonical actions of the intruder. In Chapter 7, we detail the procedure for deciding the symbolic equivalence between sets of constraint systems. The general idea of our algorithm consists of simplifying the constraint systems until we reach sets of constraint systems that are simple enough to easily decide the symbolic equivalence. Chapter 8 is dedicated to the proofs of soundness, completeness and termination of our decision procedure of symbolic equivalence between sets of constraint systems.

Thanks to our algorithm, we are now able to automatically prove the anonymity and unlinkability properties on the *e-passport* protocol [ACRR10], or the anonymity property on the *private authentication* protocol [AF04] for a bounded number of sessions. An Ocaml implementation of an early version of the procedure described in Part II has already been completed and concludes within a few minutes for the private authentication protocol. The final implementation of the decision procedure is still ongoing.

1.6.4 Proving more observational equivalences with ProVerif

As previously mentioned, due to the presence of non trivial conditional branchings in the *BAC* protocol and the private authentication protocol, ProVerif yields false attacks when verifying the unlinkability and anonymity properties respectively on such protocols. However, since ProVerif is currently the only tool that can handle an unbounded number of sessions, we address the issue of these particular false attacks and propose an extension of the ProVerif tool in Chapter 9.

The general idea of our extension is to transform before hand the input processes by simplifying their control structure. To do so, we extend the behaviour of destructor function symbols so that the tests previously performed by conditional branchings will now be performed directly inside terms. We show in Chapter 9 how the original algorithm of ProVerif [BAF08] is adapted to our new framework. Moreover, we propose an algorithm to automatically simplify the input processes when such a simplification is possible.

Relying on an ongoing implementation of our extension, we can now automatically prove that the *private authentication* protocol satisfies its desired anonymity property for an unbounded number of sessions. However, even with our extension, ProVerif still cannot prove the unlinkability property for the *BAC* protocol and yields a new false attack.

1.7 Research Publications

Almost all the results obtained in this thesis have been published or submitted for publication. The results on ProVerif (Chapter 9) should be submitted this fall.

Journals

— *(Submitted)* V. Cheval, V. Cortier, S. Delaune. Deciding equivalence-base properties using constraint solving. Theoretical Computer Science, 2012.

Conferences

— M. Arapinis, V. Cheval and S. Delaune. Verifying privacy-type properties in a modular way. In the Proceedings of the 25th IEEE Computer Security Foundations Symposium (CSF'12), Cambridge, Massachusetts, USA, June 2012, IEEE Computer Society Press.

— V. Cheval, H. Comon-Lundh and S. Delaune. Trace Equivalence Decision: Negative Tests and Non-determinism. In Proceeding of the 18th ACM Conference on Computer and Communications Security (CCS'11), Pages 321-330, Chicago, Illinois, USA, October 2011, ACM Press.

— V. Cheval, H. Comon-Lundh and S. Delaune. Automating security analysis: symbolic equivalence of constraint systems. In the Proceedings of the 5th International Joint Conference on Automated Reasoning (IJCAR'10), Pages 412-426, Edinburgh, Scotland, UK, July 2010, Springer-Verlag.

Other

— V. Cheval. H. Comon-Lundh and S. Delaune. A decision procedure for proving observational equivalence. In the Preliminary Proceedings of the 7th International Workshop on Security Issues in Coordination Models, Languages and Systems (SecCo'09), Bologna, Italy, October 2009.

Tools

— V. Cheval. ADECS, a tool for deciding symbolic equivalence between two constraint systems for a fixed set of cryptographic primitives
`http://www.lsv.ens-cachan.fr/~cheval/program/adecs/`

The implementations of the extension of ProVerif (Chapter 9) and the decision procedure for trace equivalence described in Part II are still under development.

Chapter 2

Preliminaries

Contents

In this chapter, we review several standard definitions and concepts that we will use in this thesis.

2.1 Term Algebra

One starts with an infinite set of *names*, denoted $\mathcal{N} = \{a, b, \ldots, sk, n, m \ldots\}$, which are used to model atomic data, e.g. random numbers. We also define an infinite set of *variables*, denoted $\mathcal{X} = \{x, y, \ldots, X, Y, \ldots\}$.

Let $\mathcal{F}$ be a signature, i.e. a finite set of *function symbol*. Let $\mathsf{ar} : \mathcal{F} \to \mathbb{N}$ be the function that associates to a function symbol a natural number, called arity. We call *constant* any function symbol of arity 0.

We define a set of *types*, sometimes called *sorts*, denoted $\mathsf{Type} = \{base, channel, key, \ldots\}$. We define a type system that associates to each variable and each name a type, and that associates to each function symbol the types of its arguments and return value, denoted $n : s$, $x : s$ and $\mathsf{f} : s_1 \times \ldots \times s_k \to s$, where $n \in \mathcal{N}$, $x \in \mathcal{X}$, $\mathsf{f} \in \mathcal{F}$ with $\mathsf{ar}(\mathsf{f}) = k$, and $s, s_1, \ldots, s_k \in \mathsf{Type}$.

Terms are defined as names, variables, and function symbols applied to other terms. Let $\mathsf{N} \subseteq \mathcal{N}$, $\mathsf{X} \subseteq \mathcal{X}$ and $\mathsf{F} \subseteq \mathcal{F}$ the set of terms built from N and X by applying the function symbols in F is denoted by $\mathcal{T}(\mathsf{F}, \mathsf{N} \cup \mathsf{X})$. Formally, $\mathcal{T}(\mathsf{F}, \mathsf{N} \cup \mathsf{X})$ is the smallest set such that:

— $\mathsf{X} \cup \mathsf{N} \subseteq \mathcal{T}(\mathsf{F}, \mathsf{N} \cup \mathsf{X})$

— for all $\mathsf{f} \in \mathsf{F}$, if $\mathsf{ar}(f) = k$ and $\mathsf{f} : s_1 \times \ldots \times s_k \to s$ then for all $t_1 : s_1, \ldots, t_1 : s_k \in \mathcal{T}(\mathsf{F}, \mathsf{N} \cup \mathsf{X})$, we have that $\mathsf{f}(t_1, \ldots, t_k) : s \in \mathcal{T}(\mathsf{F}, \mathsf{N} \cup \mathsf{X})$.

We say that the set $\mathcal{T}(\mathsf{F}, \mathsf{N} \cup \mathsf{X})$ is *untyped* when each term in the set have the same type. In such a case, the types are omitted for the sake of simplicity.

The *size* of a term t, denoted $|t|$ is defined recursively as follows:

— if $t \in \mathcal{N} \cup \mathcal{X}$ then $|t| = 1$

— if $t = \mathsf{f}(t_1, \ldots, t_k)$ with $\mathsf{f} \in \mathcal{F}$ and $\mathsf{ar}(\mathsf{f}) = k$ then $|t| = 1 + \sum_{i=1}^{k} |t_i|$

Example 2.1. *Consider the following untyped signature*

$$\mathcal{F} = \{\mathsf{senc}/2,\ \mathsf{sdec}/2,\ \mathsf{pk}/1,\ \langle\ \rangle/2,\ \pi_1/1,\ \pi_2/1,\ \mathsf{h}/1\}$$

that contains function symbols for asymmetric encryption, decryption and pairing, each of arity 2*, as well as projection symbols and the function symbol* pk*, each of arity* 1*. The ground term* $\mathsf{pk}(sk)$ *represents the public counterpart of the private key* sk*.*

Let $a, b, ska \in \mathcal{N}$*, we have that* $\langle \mathsf{senc}(a, \mathsf{pk}(ska)), b)$ *is a ground term of* $\mathcal{T}(\mathcal{F}, \mathcal{N})$*.*

The set of *positions* of t, denoted $\mathcal{P}os(t)$, is a set of word over the alphabet of positive integers, defined recursively such that:

— if $t \in \mathcal{N} \cup \mathcal{X}$ then $\mathcal{P}os(t) = \{\varepsilon\}$

— if $t = \mathsf{f}(t_1, \ldots, t_k)$ with $\mathsf{f} \in \mathcal{F}$, then $\mathcal{P}os(t) = \{\varepsilon\} \cup \bigcup_{i=1}^{k} \{i \cdot p \mid p \in \mathcal{P}os(t_i)\}$

where $\cdot$ is the concatenation operator. For all $p \in \mathcal{P}os(t)$, we denote by $t|_p$ the subterm defined by induction on the length of p such that:

— if $p = \varepsilon$ then $t|_p = t$

— if $p = i \cdot q$ and $t = \mathsf{f}(t_1, \ldots, t_k)$ then $t|_p = t_i|_q$

We denote by root the function that associates to each term $t \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X})$ the function symbol at position ϵ (root position) in t. For $t \in \mathcal{N} \cup \mathcal{X}$, we define $\mathsf{root}(t) = \perp$, where $\perp$ is a new symbol.

The set of *subterms* of a term t, denoted $st(t)$, is the set such that

$$st(t) = \{u \mid \text{there exists } p \in \mathcal{P}os(t) \text{ such that } t|_p = u\}$$

The set of variables of a term t, denoted $vars(t)$, is the set such that

$$vars(t) = \{x \in \mathcal{X} \mid \text{there exists } p \in \mathcal{P}os(t) \text{ such that } t|_p = x\}$$

We say that t is a *ground term* when $vars(t) = \emptyset$. The set of names of a term t, denoted $names(t)$, is the set such that

$$names(t) = \{n \in \mathcal{N} \mid \text{there exists } p \in \mathcal{P}os(t) \text{ such that } t|_p = n\}$$

These sets can be extended to any structure that contain terms.

A (k-)context C is a term in $\mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X} \cup \{\__1, \ldots, \__k\})$ such that for all $i \in \{1, \ldots, k\}$, $\__i$ appears at most once in C. Given a k-context C, and $t_1, \ldots, t_k$ terms, we denote by $C[t_1, \ldots, t_k]$ the terms C where we replace $\__i$ by t_i for each $i \in \{1, \ldots, k\}$.

2.2 Unification

A *substitution* is a function $\sigma : \mathcal{X} \to \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X})$. Given a variable x and a substitution σ, we denote by $x\sigma$ the application of σ on x. We extend the application of a substitution to term or any structure that contains variables in the expected manner. Given two substitutions σ_1 and σ_2, we denote by $\sigma_1\sigma_2$ the composition of σ_1 and σ_2, *i.e.* for all $x \in \mathcal{X}$, $x\sigma_1\sigma_2 = (x\sigma_1)\sigma_2$.

The *domain* of a substitution σ is the set $\{x \in \mathcal{X} \mid x\sigma \neq \sigma\}$, denoted $\mathrm{dom}(\sigma)$. The *image* of a substitution σ is the set $\{x\sigma \mid x \in \mathrm{dom}(\sigma)\}$. The identity substitution, denoted id, is the only substitution with empty domain.

Let $\mathsf{X} \subset \mathcal{X}$ and a substitution σ. The *restriction of* σ *to* X is the substitution denoted $\sigma|_{\mathsf{X}}$ (or $\sigma_{|\mathsf{X}}$) such that $\mathrm{dom}(\sigma|_{\mathsf{X}}) = \mathsf{X} \cap \mathrm{dom}(\sigma)$ and for all $x \in \mathrm{dom}(\sigma|_{\mathsf{X}})$, $x\sigma = x\sigma|_{\mathsf{X}}$.

We adopt the notations $\{x_1 \mapsto t_1; \ldots; x_n \mapsto t_n\}$ and $\{^{t_1}/_{x_1}; \ldots; ^{t_n}/_{x_n}\}$ to represent a substitution σ of domain $\{x_1, \ldots, x_n\}$ and such that $x_i\sigma = t_i$, for $i = 1 \ldots n$.

A *variable renaming* is a bijection ρ from $\mathcal{X}$ to $\mathcal{X}$. Note that a variable renaming is a substitution such that $\mathrm{img}(\rho) = \mathrm{dom}(\rho)$.

A *name renaming* is a bijection ρ from $\mathcal{N}$ to $\mathcal{N}$. Similarly to substitutions, the application of ρ on a name n is denoted $n\rho$. We extend the application of a renaming to any structure that contains names in the expected manner. The domain of a name renaming ρ, denoted $\mathrm{dom}(\rho)$, is the set

$\{n \in \mathcal{N} \mid n\rho \neq \rho\}$. The image of a name renaming ρ, denoted $\mathrm{img}(\rho)$, is the set $\{x\rho \mid x \in \mathrm{dom}(\rho)\}$. Note that $\mathrm{img}(\rho) = \mathrm{dom}(\rho)$.

Two terms $t_1, t_2 \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X})$ are *unifiable* if there exists a substitution σ such that $t_1\sigma = t_2\sigma$. In such a case, the substitution σ is a *unifier* of t_1 and t_2.

Given a finite set of equation between terms $S = \{t_1 = s_1; \ldots; t_n = s_n\}$, we say that a substitution σ is a unifier of S if $t_i\sigma = s_i\sigma$ for $i = 1 \ldots n$. Moreover, a substitution σ is the most general unifier of S, denoted $\mathsf{mgu}(S)$, if

— $vars(\mathrm{img}(\sigma)) \subseteq vars(S)$; and

— σ is a unifier of S; and

— for all substitution σ', σ' is a unifier of S implies that there exists a substitution τ such that $\sigma' = \sigma'\tau$.

In case S contains only one equation, *i.e.* $S = \{t = s\}$, we might denote the most general unifier of S by $\mathsf{mgu}(t = s)$ or $\mathsf{mgu}(t, s)$. Moreover, we might sometimes use the symbol $\stackrel{?}{=}$ instead of $=$.

2.3 Equational theory

An *equational theory* is an equivalence relation on terms of $\mathcal{T}(\mathcal{F}, \mathcal{X} \cup \mathcal{N})$ that is closed under application of contexts, closed under substitutions of terms for variables and closed under name renaming. In most cases, an equational theory is derived from a finite set E of equations between terms of $\mathcal{T}(\mathcal{F}, \mathcal{X})$. In such a case, we denote by $=_{\mathsf{E}}$ the smallest equational theory containing E. A slight abuse of language is usually done and we say that E is an equational theory. Given two terms s and t, we say s and t are equal modulo the equation theory E when $t_1 =_{\mathsf{E}} t_2$.

Example 2.2. *Consider the signature $\mathcal{F}$ of Example 2.1. We define the equational theory $\mathsf{E}_{\mathsf{aenc}}$ by the following equations:*

$$\mathsf{adec}(\mathsf{aenc}(x, \mathsf{pk}(y)), y) = x \qquad \pi_i(\langle x_1, x_2 \rangle) = x_i \quad \textit{for } i \in \{1, 2\}.$$

The first equation represents asymmetric decryption whereas the second one represents the first and second projections of the pair.
For example, we have that $\pi_1(\mathsf{adec}(\mathsf{aenc}(\langle n_1, n_2 \rangle, \mathsf{pk}(sk)), sk)) =_{\mathsf{E}_{\mathsf{aenc}}} n_1$.

In this thesis, we always consider that an equation theory is E is consistent, i.e. there exist terms s and t such that $s \neq_E t$.

2.4 Rewriting systems

A *rewriting system* $\mathcal{R}$ is a set of *rewrite rules* $\ell \to r$ such that $\ell \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{X}$ and $r \in \mathcal{T}(\mathcal{F}, vars(\ell))$. A term s is rewritten into t by a rewriting system $\mathcal{R}$, denoted $s \to_{\mathcal{R}} t$ if the there exists a rewrite rule $\ell \to r \in \mathcal{R}$, a position $p \in \mathcal{P}os(s)$ and a substitution σ such that $s|_p = \ell\sigma$ and $t = s[r\sigma]_p$. The reflexive transitive closure of the relation $\to_{\mathcal{R}}$ is denoted by $\to^*_{\mathcal{R}}$.

A rewriting system $\mathcal{R}$ is *subterm* if for all $\ell \to r \in \mathcal{R}$, r is a subterm of ℓ, *i.e.* $r \in st(\ell)$.

A rewriting system $\mathcal{R}$ is *confluent* if for all terms s, u, v such that $s \to^*_{\mathcal{R}} u$ and $s \to^*_{\mathcal{R}} v$, there exists a term t such that $u \to^*_{\mathcal{R}} t$ and $v \to^*_{\mathcal{R}} t$. Moreover, we say that $\mathcal{R}$ is *convergent* if $\mathcal{R}$ is confluent and terminate.

A term t is in *normal formal* (w.r.t. to a rewrite system $\mathcal{R}$) if there is no term s such that $t \to_{\mathcal{R}} s$. Moreover, if $t \to^*_{\mathcal{R}} s$ and s is in normal form then we say that s is a normal form of t. When the rewriting system $\mathcal{R}$ is convergent, the normal form of a term t is unique and is denoted $t{\downarrow}$.

Part I

Equivalence properties in the applied pi calculus

Chapter 3

Modelling of cryptographic protocols

Contents

When verifying any security properties on a cryptographic protocol in a symbolic model, one first have to properly define the model in which the cryptographic protocol and the security property will be expressed. As mentioned in the introduction, there exist dozen of symbolic models in the literature, but we focus our attention on the applied pi calculus [AF01], an expressive model in which several results on trace and equivalence properties can be found in the literature e.g. [BMU08, DKR07, ABF04]. As any symbolic model, the applied pi calculus abstracts the messages by terms and the cryptographic primitives by function symbols. To model the properties of such cryptographic primitives, the applied pi calculus relies on equational theories which are most of the time represented by a set of equations between terms.

Since equivalence properties are expressed by the means of behavioural equivalence, we focus in Section 3.2 on four main definitions of equivalence (observational equivalence, labeled bisimilarity, may-testing equivalence, trace equivalence) and present the relations between these equivalences. In particular, we show that may-testing does not imply trace equivalence. The difference between the two notions is subtle. We exhibit a counter-example that relies on the fact that may-testing equivalence requires the attacker to commit in advance on part of its behaviour, yielding a slightly weaker attacker. We further show that may-testing equivalence does imply trace equivalence in case the processes have finitely many successors (*e.g* for processes without replication): trace equivalence and may-testing equivalence coincide for image-finite processes.

At last, we detail in this chapter the *private authentication* protocol [AF04] and some sub-protocols that are described in the ICAO standard [ICA04] for the e-passport protocol, namely

Basic Access Control protocol, *Passive Authentication* protocol and *Active Authentication* protocol. The last two protocols will be used in Chapter 5 to illustrate our result about composition of trace equivalence. Moreover, all these protocols will be part of our benchmark for the implementation of the decision algorithm of trace equivalence presented in Part II. At last, the *private authentication* along with the *BAC* protocol will also serve as guidelines for our work on the tool ProVerif [Bla01] (see Chapter 9).

3.1 The applied pi calculus

The *applied pi calculus* [AF01] is a derivative of the pi calculus that is specialised for modelling cryptographic protocols. Participants in a protocol are modelled as processes, and the communication between them is modelled by means of message passing.

3.1.1 Syntax

To describe processes in the applied pi calculus, we rely on a sort system for terms defined in Section 2.1. More specifically, we rely on two subsets of types: *base types* and *channel types*. The details of the sort system are unimportant, as long as *base types* differ from *channel types*. As such, we will denote $\mathcal{N}_b \subset \mathcal{N}$ and $\mathcal{X}_b \subset \mathcal{X}$ the infinite sets of names and variables of base type. Similarly, we denote by $\mathcal{N}_{ch} \subset \mathcal{N}$ and $\mathcal{X}_{ch} \subset \mathcal{X}$ the infinite sets of names and variables of channel type.

In the applied pi calculus, the function symbols only operate on and return terms of base type. It implies that any term of channel type is either a variable in $\mathcal{X}_{ch}$ or a name in $\mathcal{N}_{ch}$.

$P, Q, R :=$	plain processes	$A, B, C :=$	extended processes
0		P	
$P \mid Q$		$A \mid B$	
$!P$		$\nu n.A$	
$\nu n.P$		$\nu x.A$	
$\mathsf{if}\ u = v\ \mathsf{then}\ P\ \mathsf{else}\ Q$		$\{^u/_x\}$	
$\mathsf{in}(c, x).P$			
$\mathsf{out}(c, u).P$			

where u and v are terms of any type, n is a name, x is a variable and c is a term of channel type, *i.e.* a name or a variable.

Figure 3.1: Syntax of processes

The applied pi calculus defines *plain processes*, denoted by P, Q, R, and *extended processes*, denoted by A, B, C. Plain processes are built up in a similar way to processes in pi calculus except that messages can contain terms rather than just names. Extended processes add *active substitutions* and restriction on variables (see Figure 3.1). The substitution $\{^u/_x\}$ is an active substitution that replaces the variable x with the term u. Active substitutions generalize the "let" construct: $\nu x.(\{^u/_x\} \mid P)$ corresponds exactly to

$$\text{"let } x = u \text{ in } P\text{"}.$$

Example 3.1. *Let* $\mathcal{F}_{senc} = \{\mathsf{senc}/2,\ \mathsf{sdec}/2,\ \mathsf{f}/1\}$ *be the signature composed of the symmetric encryption/decryption and a unary function* f. *Consider the Handshake protocol [GML+93, DJ06] describe in the Alice & Bob representation as follows:*

$$\begin{array}{lll} 0. & A \to B : & \mathsf{senc}(N, k_{AB}) \\ 1. & B \to A : & \mathsf{senc}(\mathsf{f}(N), k_{AB}) \end{array}$$

In this protocol, we consider that the two participants A *and* B *share a secret key* k_{AB}.

1. *A starts by creating a fresh* nonce N, *i.e. a random number, and then sends to* B *this nonce encrypted with the secret key* k_{AB}.

2. *B recovers N by decrypting the message received, then applies a certain function* f *on N and sends the result to A encrypted with the same secret key* k_{AB}.
3. *At last, A also applies* f *on N, compares the result with the decryption of the message received and then executes a certain protocol P if the test is successful.*

We model this protocol using the applied pi calculus as follows:

$$\nu k_{AB}.\quad (\ \nu N.\ \mathsf{out}(c, \mathsf{senc}(N, k_{AB}))).\ \mathsf{in}(c, x).\ \mathsf{if}\ \mathsf{sdec}(x, k_{AB}) = \mathsf{f}(N)\ \mathsf{then}\ P\ \mathsf{else}\ 0.\\ \quad |\ \mathsf{in}(c, y).\ \mathsf{out}(c, \mathsf{senc}(\mathsf{f}(\mathsf{sdec}(y, k_{AB})), k_{AB})))$$

where c represents the public channel on which A and B communicate.

As usual, names and variables have scopes, which are delimited by restrictions and by inputs. We write $fvars(A)$, $bvars(A)$, $fnames(A)$ and $bnames(A)$ for the sets of *free* and *bound variables* and *free* and *bound names* of A, respectively. We say that an extended process is *closed* if all its variables are either bound or defined by an active substitution. An *evaluation context* $C[\_]$ is an extended process with a hole instead of an extended process.

Active substitutions are useful because they allow us to map an extended process A to its *frame*, denoted $\Phi(A)$, by replacing every plain process in A with 0. Hence, a frame is an extended process built up from 0 and active substitutions by parallel composition and restriction. The frame $\Phi(A)$ accounts for the set of terms statically possessed by the intruder (but does not take into account for A's dynamic behavior). The domain of a frame Φ, denoted by $\mathrm{dom}(\Phi)$, is the set of variables for which Φ defines a substitution (those variables x for which Φ contains a substitution $\{^{M}/_{x}\}$ not under a restriction on x).

Example 3.2. *Consider the signature* $\mathcal{F}_{senc}$ *of Example 3.1 and the following process A made up of three components in parallel:*

$$\nu s.\nu sk.\nu x_1.(\mathsf{out}(c_1, x_1) \mid \mathsf{in}(c_1, y).\mathsf{out}(c_2, \mathsf{sdec}(y, sk)) \mid \{^{\mathsf{senc}(s,sk)}/_{x_1}\}).$$

Its first component publishes the message $\mathsf{senc}(s, sk)$ *stored in* x_1 *by sending it on* c_1*. The second receives a message on* c_1*, uses the secret key sk to decrypt it, and forwards the result on* c_2*. We have* $\Phi(A) = \nu s, sk, x_1.\{^{\mathsf{senc}(s,sk)}/_{x_1}\}$ *and* $\mathrm{dom}(\Phi(A)) = \emptyset$ *(since* x_1 *is under a restriction).*

3.1.2 Semantics

We briefly recall the operational semantics of the applied pi calculus (see [AF01] for details). First, we associate an equational theory E to the signature $\mathcal{F}$. The purpose of this equational theory is to represent the behaviour of the cryptographic primitives. We require the equational theory to be closed under one-to-one renaming.

Structural equivalence, noted $\equiv$, is the smallest equivalence relation on extended processes that is closed under α-conversion of names and variables, by application of evaluation contexts, and satisfying some further basic structural rules such as $A \mid 0 \equiv A$ and $!A \equiv !A \mid A$, associativity and commutativity of $|$, binding-operator-like behavior of ν, and when $u =_{\mathsf{E}} v$ the equivalences:

$$\nu x.\{^{u}/_{x}\} \equiv 0 \qquad \{^{u}/_{x}\} \equiv \{^{v}/_{x}\} \qquad \{^{u}/_{x}\} \mid A \equiv \{^{v}/_{x}\} \mid A\{^{v}/_{x}\}$$

Example 3.3. *Let P be the following process:*

$$\nu s.\nu sk.\ \big(\mathsf{out}(c_1, \mathsf{senc}(s, sk)) \mid \mathsf{in}(c_1, y).\mathsf{out}(c_2, \mathsf{sdec}(y, sk))\big).$$

The process P is structurally equivalent to the process A given in Example 3.2. We have that $\Phi(P) = 0 \equiv \Phi(A)$.

The operational semantics of processes in the applied pi calculus is defined by structural rules defining two relations: *structural equivalence* (described above) and *internal reduction*, noted $\xrightarrow{\tau}$.

Internal reduction is the smallest relation on extended processes closed under structural equivalence and application of evaluation contexts such that:

$$\begin{array}{rcll} \mathsf{out}(a,u).P \mid \mathsf{in}(a,x).Q & \xrightarrow{\tau} & P \mid Q\{^{u}/_{x}\} & \\ \mathtt{if}\ u = v\ \mathtt{then}\ P\ \mathtt{else}\ Q & \xrightarrow{\tau} & P & \text{if } u =_{\mathsf{E}} v \\ \mathtt{if}\ u = v\ \mathtt{then}\ P\ \mathtt{else}\ Q & \xrightarrow{\tau} & Q & \text{if } u, v \text{ ground terms such that } u \neq_{\mathsf{E}} v \end{array}$$

The operational semantics is extended by a *labeled* operational semantics enabling us to reason about processes that interact with their environment. Labeled operational semantics defines the relation $\xrightarrow{\ell}$ where ℓ is a label $in(c,u)$ where u is a term that may contain names and variables and c is a term of channel type; a label $out(c,u)$ or $\nu u.out(c,u)$, where u is a variable of base type or a name of channel type. The scopes of names and variables are only delimited by restrictions, i.e. the input label $in(c,u)$ does not restrict variables contrary to the plain process $\mathsf{in}(c,x)$.

We adopt the following rules in addition to the internal reduction rules. Below, the names a and c are channel names whereas x is a variable of base type and y is a variable of any type.

$$\textsc{In}\quad \mathsf{in}(a,y).P \xrightarrow{in(a,u)} P\{^{u}/_{y}\}$$

$$\textsc{Out-Ch}\quad \mathsf{out}(a,c).P \xrightarrow{out(a,c)} P$$

$$\textsc{Open-Ch}\quad \frac{A \xrightarrow{out(a,c)} A' \qquad c \neq a}{\nu c.A \xrightarrow{\nu c.out(a,c)} A'}$$

$$\textsc{Out-T}\quad \mathsf{out}(a,M).P \xrightarrow{\nu x.out(a,x)} P \mid \{^{M}/_{x}\} \quad x \notin fvars(P) \cup fvars(M)$$

$$\textsc{Scope}\quad \frac{A \xrightarrow{\ell} A' \quad u \text{ does not occur in } \ell}{\nu u.A \xrightarrow{\ell} \nu u.A'}$$

$$\textsc{Par}\quad \frac{bnames(\ell) \cap fnames(B) = \emptyset \quad bvars(\ell) \cap fvars(B) = \emptyset \quad A \xrightarrow{\ell} A'}{A \mid B \xrightarrow{\ell} A' \mid B}$$

$$\textsc{Struct}\quad \frac{A \equiv B \quad B \xrightarrow{\ell} B' \quad B' \equiv A'}{A \xrightarrow{\ell} A'}$$

Let $\mathcal{A}$ be the alphabet of actions (in our case this alphabet is infinite) where the special symbol $\tau \in \mathcal{A}$ represents an unobservable action. For every $w \in \mathcal{A}^*$ the relation $\xrightarrow{w}$ on extended processes is defined in the usual way. By convention $A \xrightarrow{\epsilon} A$ where ϵ denotes the empty word.

For every $s \in (\mathcal{A} \setminus \{\tau\})^*$, the relation $\overset{s}{\Rightarrow}$ on extended processes is defined by: $A \overset{s}{\Rightarrow} B$ if, and only if, there exists $w \in \mathcal{A}^*$ such that $A \xrightarrow{w} B$ and s is obtained from w by erasing all occurrences of τ. Intuitively, $A \overset{s}{\Rightarrow} B$ means that A transforms into B by experiment s. We also consider the relation $A \overset{w}{\mapsto} B$ and $A \overset{s}{\Mapsto} B$ that are the restriction of the relations $\xrightarrow{w}$ and $\overset{s}{\Rightarrow}$ on closed extended processes.

Note that the labeled transition is not closed under application of evaluation contexts. Moreover the output of a term u needs to be made "by reference" using a restricted variable and an active substitution. The rules differ slightly from those described in [AF01] but it has been shown in [DKR07] that the two underlying notions of observational equivalence coincide.

Example 3.4. *Consider the protocol P in Example 3.3. One possible sequence of transitions for P is the following:*

$$\begin{array}{rcl} P & \xrightarrow{\nu x.\mathsf{out}(c_1,x)} & \nu s.\nu sk.(\mathsf{in}(c_1,y).\mathsf{out}(c_2,\mathsf{adec}(y,sk)) \mid \{^{\mathsf{aenc}(s,\mathsf{pk}(sk))}/_{x}\}) \\ & \xrightarrow{\mathsf{in}(c_1,x)} & \nu s.\nu sk.(\mathsf{out}(c_2,\mathsf{adec}(\mathsf{senc}(s,\mathsf{pk}(sk)),sk)) \mid \{^{\mathsf{aenc}(s,\mathsf{pk}(sk))}/_{x}\}) \\ & \xrightarrow{\nu y.\mathsf{out}(c_2,y)} & \nu s.\nu sk.(\{^{s}/_{y}\} \mid \{^{\mathsf{aenc}(s,\mathsf{pk}(sk))}/_{x}\}) \end{array}$$

Note that all transitions in this sequence are visible actions. However, the two first transitions only result on passing through a message. Hence, it is possible to apply an internal reduction:

$$\begin{array}{rcl} P & \xrightarrow{\tau} & \nu s.\nu sk.\mathsf{out}(c_2,\mathsf{adec}(\mathsf{senc}(s,\mathsf{pk}(sk)),sk)) \\ & \xrightarrow{\nu y.\mathsf{out}(c_2,y)} & \nu s.\nu sk.\{^{s}/_{y}\} \end{array}$$

Note that the frame is "smaller" than the frame in the previous sequence of transitions. It was predictable by the fact that we replaced some labeled actions (visible for the attacker) by an internal reduction (invisible action for the attacker).

3.2 Behavioural equivalences and their relations

Behavioural equivalences intuitively define the fact that no observer can see the difference between two processes. They can be used to formalise many interesting security properties, in particular privacy related properties, such as those studied in [AF04, DKR09, ACRR10, BCdH10]. Slightly more specifically, two processes can be said equivalent if an observer, whatever how he behaves, observes the same (or equivalent) outputs from the two processes. This can be formally defined as *trace equivalence* as introduced in [MNP02]. An alternative definition is *testing equivalence* as defined for example by M. Abadi and A. Gordon [AG99]. In the context of the pi calculus, may-testing and trace equivalences are known to be difficult to prove. Therefore, a stronger notion has been proposed: *observational equivalence*, which requires in addition the two processes to be (weakly) bisimilar.

This section is devoted to the definition of these three notions and the study of their relations in the context of the applied pi calculus. We first introduce the notion of *static equivalence* that compares sequences of messages, a notion of intruder's knowledge that has been extensively studied, for example in [AC06].

Definition 3.1 (static equivalence $\sim$)**.** *Two terms u and n are* equal in the frame Φ*, written $(u =_{\mathsf{E}} v)\Phi$, if there exists $\tilde{n}$ and a substitution σ such that $\Phi \equiv \nu\tilde{n}.\sigma$, $\tilde{n} \cap (fnames(u) \cup fnames(v)) = \emptyset$, and $u\sigma =_{\mathsf{E}} v\sigma$.*

Two closed frames Φ_1 and Φ_2 are statically equivalent*, written $\Phi_1 \sim \Phi_2$, when:*

- $\mathrm{dom}(\Phi_1) = \mathrm{dom}(\Phi_2)$*, and*
- *for all terms u, v, $(u =_{\mathsf{E}} v)\Phi_1$ if and only if $(u =_{\mathsf{E}} v)\Phi_2$.*

Example 3.5. *Consider the theory $\mathsf{E}_{\mathsf{aenc}}$ (see Example 2.2), $\Phi_a = \{^{\mathsf{aenc}(a,\mathsf{pk}(sk))}/_{x_1}\}$, and $\Phi_b = \{^{\mathsf{aenc}(b,\mathsf{pk}(sk))}/_{x_1}\}$. We have that $(\mathsf{adec}(x_1, sk) =_{\mathsf{E}_{\mathsf{aenc}}} a)\Phi_a$ whereas $(\mathsf{adec}(x_1, sk) \neq_{\mathsf{E}_{\mathsf{aenc}}} a)\Phi_b$, thus $\Phi_a \not\sim \Phi_b$. Note that Φ_a and Φ_b do not restrict sk, a nor b hence they are accessible to the attacker.*

However, $\nu sk.\varphi \sim \nu sk.\varphi'$. This is a non trivial equivalence. Intuitively, since sk is now restricted, it can not appear in the tests which also implies that $\mathsf{pk}(sk)$, the encryption key, can not appear directly in the tests. Thus, there is no test that allows one to distinguish the two frames.

3.2.1 Trace equivalence

For every closed extended process A, we define its set of traces, each trace consisting in a sequence of actions together with the sequence of sent messages:

$$\mathsf{trace}(A) = \{(s, \phi(B)) \mid A \overset{s}{\Rightarrow} B \text{ for some } B\}.$$

Note that, in the applied pi calculus, the sent messages of base type are exclusively stored in the frame and not in the sequence s (the outputs are made by "reference").

Two processes are trace equivalent if, whatever the messages they received (built upon previously sent messages), the resulting frames are in static equivalence.

Definition 3.2 (trace equivalence $\approx_t$)**.** *Let A and B be two closed extended processes, $A \sqsubseteq_t B$ if for every $(s, \Phi) \in \mathsf{trace}(A)$ such that $bnames(s) \cap fnames(B) = \emptyset$, there exists $(s', \Phi') \in \mathsf{trace}(B)$ such that $s = s'$ and $\Phi \sim \Phi'$.*

Two closed extended processes A and B are trace equivalent*, denoted by $A \approx_t B$, if $A \sqsubseteq_t B$ and $B \sqsubseteq_t A$.*

As previously explained, proving trace equivalence of two protocols is very difficult. Indeed the set of possible traces of a protocol is usually infinite hence one has to compute and compare two sets of infinite traces. Furthermore, given two traces, one still need to prove the static equivalence which also consist of evaluating and comparing an infinite set pair of terms.

Example 3.6. *Consider A and B the two following protocols:*

$$\begin{array}{rcl} A & \stackrel{def}{=} & \mathsf{out}(c_1, \mathsf{aenc}(a, \mathsf{pk}(sk))).\mathsf{out}(c_2, b) \\ B & \stackrel{def}{=} & \mathsf{out}(c_1, \mathsf{aenc}(b, \mathsf{pk}(sk))) \mid \mathsf{out}(c_2, b) \end{array}$$

We have that $A \not\approx_t B$. Indeed, we can find several traces that can't be matched in both protocols. For example, B can be reduced as follows:

$$B \xrightarrow{\nu x.out(c_2,x)} \mathsf{out}(c_1, \mathsf{aenc}(b, \mathsf{pk}(sk))) \mid \{^{b}/_{x}\}$$

However, to output on the public channel c_2, the protocol A must first output on the public channel c_1 and so such trace can't be matched in A, hence the non-equivalence.

This trace is not the only witness of the non-equivalence. Indeed, one can reduce A and B as follows:

$$\begin{array}{rcl} A & \xrightarrow{\nu x.out(c_1,x)} & \mathsf{out}(c_2, b) \mid \{^{\mathsf{aenc}(a,\mathsf{pk}(sk))}/_{x}\} \stackrel{def}{=} A' \\ B & \xrightarrow{\nu x.out(c_1,x)} & \mathsf{out}(c_2, b) \mid \{^{\mathsf{aenc}(b,\mathsf{pk}(sk))}/_{x}\} \stackrel{def}{=} B' \end{array}$$

In such a case, we have that $\Phi(A') = \{^{\mathsf{aenc}(a,\mathsf{pk}(sk))}/_{x}\}$ and $\Phi(B') = \{^{\mathsf{aenc}(b,\mathsf{pk}(sk))}/_{x}\}$. These frames correspond to the frame in Example 3.5 and we already showed that $\Phi(A') \not\sim \Phi(B')$, hence the non-equivalence.

3.2.2 May-testing equivalence

We write $A{\Downarrow}c$ when A can send a message on c, that is, when $A \stackrel{\epsilon}{\Rightarrow} C[\mathsf{out}(c, M).P]$ for some evaluation context C that does not bind c. A *test* consists of any evaluation context C and any channel name c. A closed extended process A *passes* the test if and only if $C[A]{\Downarrow}c$. The notion of testing gives rise to a *may-testing preorder* $\sqsubseteq_m$ and to a *may-testing equivalence* $\approx_m$ on the set of closed extended processes.

Definition 3.3 (may-testing equivalence)**.** *Let A and B be two closed extended processes such that* $\mathrm{dom}(A) = \mathrm{dom}(B)$*, $A \sqsubseteq_m B$ if for any test (C, c) such that C is a closing evaluation context for A (and B) we have that $C[A]{\Downarrow}c$ implies that $C[B]{\Downarrow}c$. Two closed extended processes A and B are in* may-testing equivalence*, denoted by $A \approx_m B$, if $A \sqsubseteq_m B$ and $B \sqsubseteq_m A$.*

The idea of may-testing equivalence comes from the work of R. De Nicola and M. Hennessy [NH84]. Our definition is similar to their notion of may-testing equivalence. Since then, this notion of may-testing equivalence has been used in several cryptographic calculi, *e.g.* spi-calculus [AG99]. This notion is usually shown to be equivalent to a notion of trace equivalence (as the one given in Definition 3.2) that is easier to manipulate since the universal quantification over all contexts has been removed.

Example 3.7. *Coming back to the protocols A and B in Example 3.6, we can find back the two witnesses of the non-equivalence brought to light using the notion of may-testing equivalence. Indeed, we have that $B{\Downarrow}c_2$ while $A \not\Downarrow c_2$. Furthermore, let C the following evaluation context:*

$$C \stackrel{def}{=} \_ \mid \mathsf{in}(c_1, x).\mathsf{if}\ \mathsf{adec}(x, sk) = a\ \mathsf{then}\ \mathsf{out}(bad, a)\ \mathsf{else}\ 0$$

where bad is a public channel. C is closing for A and B. Furthermore, we have that $C[A]{\Downarrow}bad$ and $C[B] \not\Downarrow bad$. Note that the test performed in the if then else *part of C corresponds exactly to the test used to show that $\Phi(A')$ and $\Phi(B')$ are not statically equivalent in Example 3.5.*

3.2.3 Relations between may-testing and trace equivalence

People usually consider that may-testing equivalence coincide with trace equivalence. Actually, this result has been proved in two variants of the spi-calculus [MNP02, DSV03] and is in the spirit of the main theorem stated in [AF01]. We tried to adapt their proof, but it happens that the two notions do not coincide in the applied pi calculus setting. We indeed exhibit two processes that are may-testing equivalent but not trace equivalent. However, these two notions are similar and try to capture the same concept. We show that trace equivalence implies may-testing equivalence (without any additional restriction) and the other implication holds as soon as we consider processes without replication. More precisely, this last implication holds for processes that are image-finite (up to static equivalence).

3.2.3.1 Trace equivalence implies may-testing equivalence

Theorem 3.1. *For all A and B two closed extended processes,*

$$A \approx_t B \text{ implies that } A \approx_m B.$$

The proof of Theorem 3.1 requires to characterize any evolution of the contextualized process A, that is, any evolution of $C[A]$.

Given a closed extended process A and an evaluation context $C[\_]$, the proposition below allows us to map a derivation issued from $C[A]$ to a labeled trace issued from A. Note that we only consider evaluation contexts having a special shape (there is no restriction in front of the hole). This additional assumption is actually fulfilled when proving Theorem 3.1.

Proposition 3.1. *Let A and B be two closed extended process with $\mathrm{dom}(A) = \mathrm{dom}(B)$, and $C[\_] = \nu\tilde{n}.(D \mid \_)$ be an evaluation context closing for A. If $C[A] \overset{\epsilon}{\Rightarrow} A''$ for some process A'', then there exist a closed extended process A', an evaluation context $C' = \nu\tilde{n}'.(D' \mid \_)$ closing for A', and a trace $\mathsf{tr} \in (\mathcal{A} \setminus \{\tau\})^*$ such that $A'' \equiv C'[A']$, $A \overset{\mathsf{tr}}{\Rightarrow} A'$, and for all closed extended process B',*

$$B \overset{\mathsf{tr}}{\Rightarrow} B' \text{ and } \Phi(B') \sim \Phi(A') \text{ imply that } C[B] \overset{\epsilon}{\Rightarrow} C'[B'].$$

*Proof (sketch).*Proving Proposition 3.1 is quite technical thus the full proof can be found in Appendix A.1.

The proof of Proposition 3.1 works by induction on the length ℓ of the derivation of $C[A] \overset{\epsilon}{\Rightarrow} A''$. The initial case (*i.e.* $\ell = 0$) is almost trivial by defining $C' = C$ thus we focus on the inductive case (*i.e.* $\ell > 0$):

The length of the derivation being strictly bigger than 0, we know that $C[A] \overset{\epsilon}{\Rightarrow} A''$ implies $C[A] \overset{\epsilon}{\Rightarrow} A_1 \overset{\tau}{\mapsto} A''$ for some closed extended process A_1. Since the length of the derivation $C[A] \overset{\epsilon}{\Rightarrow} A_1$ is strictly smaller than ℓ, we apply our inductive hypothesis: There exist a closed extended process A'_1, an evaluation content $C'_1[\_] = \nu\tilde{n}'_1.(D'_1 \mid \_)$ closing for A'_1, and a labeled sequence $\mathsf{tr}_1 \in (\mathcal{A} \setminus \{\tau\})^*$ such that $A_1 \equiv C'_1[A'_1]$, $A \overset{\mathsf{tr}_1}{\Rightarrow} A'_1$, and for all closed extended processes B'_1, we have that $B \overset{\mathsf{tr}_1}{\Rightarrow} B'_1$ and $\Phi(B'_1) \sim \Phi(A'_1)$ imply that $C[B] \overset{\tau}{\Rightarrow} C'_1[B'_1]$.

Then, the proof consists of matching the internal reduction $\nu\tilde{n}'_1.(D'_1 \mid A'_1) \overset{\tau}{\mapsto} A''$ either to a labeled transition applied on A'_1 or to a modification of the evaluation context C.

Typically, if the internal derivation is applied solely on A'_1, then we have $C' = C'_1$, $A'_1 \overset{\tau}{\mapsto} A'$ for some A'. On the other hand, if the internal derivation is applied solely on D'_1 then only the evaluation context is modified, i.e. $C'[\_] = \nu\tilde{n}'_1.(D'_2 \mid \_)$, $A' = A'_1$ and $D'_1 \overset{\tau}{\mapsto} D'_2$ for some D'_2. The last possible case of internal reduction is a communication between D'_1 and A'_1. But then, two cases must be distinguished: either the output was part of D'_1 and the input part of A'_1; or the output was part of A'_1 and the input part of D'_1. Intuitively, if the input was part of A'_1 then it will be matched by the labeled transition $A'_1 \xrightarrow{in(c,M)} A'$ where M is determined by the output

in D'_1. Else the internal reduction will be matched by the labeled transition $A'_1 \xmapsto{\nu x.out(c,x)} A'$ for some c.

The proof of Theorem 3.1 also relies on the fact that trace equivalence is closed by one-to-one renamings of free names. This is formally stated in the lemma below:

Lemma 3.1. *Let A and B be two closed extended processes such that $A \approx_t B$ and u be a name (resp. variable) that occurs in $fnames(A) \cup fvars(A) \cup fnames(B) \cup fvars(B)$ and not in $bnames(A) \cup bvars(A) \cup bnames(B) \cup bvars(B)$, and u' be a fresh name (resp. variable). We have that $A\{^{u'}/_{u}\} \approx_t B\{^{u'}/_{u}\}$.*

We are now able to prove Theorem 3.1.

Proof of Theorem 3.1. Let A, B be two closed extended processes such that $A \approx_t B$. Let $C[\_]$ be an evaluation context closing for A (and B), and c be a channel name. We assume w.l.o.g. that $C[\_] = \nu\tilde{n}.(D \mid \nu\tilde{m}.\_)$ for some extended process D and for some sequences of names and variables $\tilde{n}$ and $\tilde{m}$. We assume w.l.o.g. that $\tilde{m} \cap (bnames(A) \cup bvars(A)) = \emptyset$ and $\tilde{m} \cap (bnames(B) \cup bvars(B)) = \emptyset$. Let $A_2 = A\{^{\tilde{m}'}/_{\tilde{m}}\}$ and $B_2 = B\{^{\tilde{m}'}/_{\tilde{m}}\}$ where $\tilde{m}'$ is a sequence of fresh names and variables. Thanks to Lemma 3.1, we have that $A_2 \approx_t B_2$. Let $C_2[\_] = \nu\tilde{n}.\nu\tilde{m}'.(D \mid \_)$. We have that $C[A] \equiv C_2[A_2]$ and $C[B] \equiv C_2[B_2]$.

Assume now that $C[A]\Downarrow c$. This means that there exist a closing evaluation context C_1 that does not bind c, a term M, and a plain process P such that $C[A] \equiv C_2[A_2] \overset{\epsilon}{\Rightarrow} C_1[\mathsf{out}(c, M).P]$. Applying Proposition 3.1 on A_2, B_2 and $C_2[\_]$, we know that there exist a closed extended process A'_2, a closing evaluation context $C'_2[\_] = \nu\tilde{r}.(E \mid \_)$ for A'_2 and $\mathsf{tr} \in (\mathcal{A} \setminus \{\tau\})^*$ such that $C_1[\mathsf{out}(c, M).P] \equiv C'_2[A'_2]$, and $A_2 \overset{\mathsf{tr}}{\Rightarrow} A'_2$, and for all closed extended process B'_2 such that $B_2 \overset{\mathsf{tr}}{\Rightarrow} B'_2$ and $\Phi(B'_2) \sim \Phi(A'_2)$, we have that $C_2[B_2] \overset{\epsilon}{\Rightarrow} C'_2[B'_2]$. Moreover, we assume w.l.o.g. that $bnames(\mathsf{tr}) \cap fnames(B_2) = \emptyset$.

Since $C'_2 = \nu\tilde{r}.(E|\_)$, we can deduce from $C_1[\mathsf{out}(c, M).P] \equiv C'_2[A'_2]$ that the output $\mathsf{out}(c, M)$ comes either from the process E or from A'_2. We distinguish these two cases:

— *The output $\mathsf{out}(c, M)$ comes from E.* Since, we have that $A_2 \approx_t B_2$, we know that that there exists B'_2 such that $B_2 \overset{\mathsf{tr}}{\Rightarrow} B'_2$ and $\Phi(A'_2) \sim \Phi(B'_2)$. Therefore, we have that $C_2[B_2] \overset{\epsilon}{\Rightarrow} C'_2[B'_2] \equiv \nu\tilde{r}.(E \mid B'_2)$. But by hypothesis, we know that the output $\mathsf{out}(c, M)$ comes from E and $c \notin \tilde{r}$. Hence we have that $C_2[B_2]\Downarrow c$, and since $C[B] \equiv C_2[B_2]$, we conclude that $C[B]\Downarrow c$.

— *The output $\mathsf{out}(c, M)$ comes from A'_2.* Thus, we have that $A'_2 \equiv \nu\tilde{v}.(\mathsf{out}(c, M).P \mid A_3)$ with $c \notin \tilde{v}, \tilde{r}$. Thus, we have that $A'_2 \xmapsto{\nu z.\mathsf{out}(c,z)} \nu\tilde{v}.(P \mid A_3 \mid \{^{M}/_{z}\})$ (if M is a term of channel type, the transition is different but the proof can be done in a similar way.) Let $A'' = \nu\tilde{v}.(P \mid A_3 \mid \{^{M}/_{z}\})$ and $\mathsf{tr}' = \mathsf{tr} \cdot \nu z.\mathsf{out}(c, z)$, we have that $A_2 \overset{\mathsf{tr}'}{\Rightarrow} A''$. Since we have that $A_2 \approx_t B_2$, we have that there exists B'_2 such that $B_2 \overset{\mathsf{tr}'}{\Rightarrow} B'_2$ and $\Phi(A'') \sim \Phi(B'_2)$. Thus, we can deduce that there exists B' such that $B_2 \overset{\mathsf{tr}}{\Rightarrow} B' \xmapsto{\nu z.\mathsf{out}(c,z)} B'_2$. Therefore, we have that there exist a term N, an evaluation context C_3 and a process Q such that $B' \equiv C_3[\mathsf{out}(c, N).Q]$ and c is not bound by C_3. Furthermore, we have that $\Phi(A'_2) \sim \Phi(B')$ which means that $C_2[B_2] \overset{\epsilon}{\Rightarrow} C'_2[B']$, and thus $C_2[B_2] \overset{\epsilon}{\Rightarrow} C'_2[C_3[\mathsf{out}(c, N).Q]]$. Hence, we have that $C_2[B_2]\Downarrow c$, and since $C[B] \equiv C_2[B_2]$, we conclude that $C[B]\Downarrow c$. □

3.2.3.2 May-testing equivalence does not imply trace equivalence

While may-testing and trace equivalences seem to be two very similar notions, may-testing equivalence actually does not imply trace equivalence. Roughly, in may-testing equivalence, the attacker is allowed to perform a sequence of inputs/outputs followed by a finite number of tests, but he has to commit on these tests before knowing how the communication actions will be used. In trace equivalence, all the tests are considered through static equivalence. The attacker does

not have to commit on the tests he wants to use in advance. So, the attacker is more adaptive in trace equivalence and this gives him more power.

We describe two processes A and B such that $A \not\approx_t B$ whereas $A \approx_m B$. Let A and B be the two following processes:

— $A = \nu b.\nu c_1.\big(\mathsf{out}(c_1, \mathit{token}) \mid \mathsf{in}(c_1, x).\mathsf{out}(c, b) \mid \mathsf{in}(c_1, x).B\big)$; and

— $B = \nu c_2.\big(\mathsf{out}(c_2, \mathsf{h}(a)) \mid \mathsf{in}(c_2, x).\mathsf{out}(c, x) \mid !\mathsf{in}(c_2, x).\mathsf{out}(c_2, \mathsf{h}(x))\big)$.

We consider the empty equational theory and a signature which only contains the symbol h of arity 1 (with no equation). The private channel c_1 in A is used to model the choice operator. The behavior of A consists of outputting a fresh name b or executing the process B. The processes B relies also on a private channel, namely c_2, whose purpose is to model the choice operator. An execution of B will output a message of the form $\mathsf{h}(\mathsf{h}(\dots \mathsf{h}(a) \dots))$ on the public channel c, for an arbitrary number of h. In the remaining, we denote by $\mathsf{h}^n(a)$ the term obtained by applying n times h on top of a.

Regarding trace equivalence, we have that:

— $\mathsf{trace}(A) = \{(\epsilon, 0);\ (\mathsf{tr}, \nu b.\{^b/_x\});\ (\mathsf{tr}, \{^{\mathsf{h}(a)}/_x\});\ \dots;\ (\mathsf{tr}, \{^{\mathsf{h}^n(a)}/_x\});\ \dots\}$

— $\mathsf{trace}(B) = \{(\epsilon, 0);\ (\mathsf{tr}, \{^{\mathsf{h}(a)}/_x\});\ \dots;\ (\mathsf{tr}, \{^{\mathsf{h}^n(a)}/_x\});\ \dots\}$

with $\mathsf{tr} = \nu x.\mathsf{out}(c, x)$. We can easily see that $A \not\approx_t B$: the frame $\nu b.\{^b/_x\}$ is not statically equivalent to any other frame in $\mathsf{trace}(B)$.

On the other hand, we have that $A \approx_m B$. This is a non trivial equivalence that is not easy to prove. Below, we only give an intuition. First, it seems clear that $B \sqsubseteq_m A$ since A can easily mimic the process B by performing first an internal communication and giving the token to the last part of the process. Regarding the other inclusion, consider a test (C, c) such that C is a closing evaluation context for A. The most interesting case is when $C[A]$ outputs the fresh name b on the channel c and then performs some tests on this message. However, since the execution trace that leads to the output on c is fixed, we know the tests that have been performed in such a situation, and we can compute n_0 such that the same tests will be satisfied if the outputted message was $\mathsf{h}^{n_0}(a)$ instead of b (for instance, we can choose for n_0 the number of occurrences of h in the tests plus one).

3.2.3.3 Case of image-finite processes

As illustrated in Section 3.2.3.2, may-testing equivalence does not imply trace equivalence in general. However, the implication holds as soon as we consider processes without replication. More precisely, this implication holds for processes that are image-finite (up to static equivalence).

Definition 3.4 (image-finite)**.** *An extended process A is* image-finite *if for each sequence of actions tr, the set of equivalence classes $\{\Phi(A') \mid A \stackrel{\mathsf{tr}}{\Rightarrow} A'\}/\sim$ is finite.*

Any process without replication is image-finite. Indeed, if a process A does not contain a replication then we have that $\{\Phi(A') \mid A \stackrel{\mathsf{tr}}{\Rightarrow} A'\}$ is a finite set up to structural equivalence, thus so is $\{\Phi(A') \mid A \stackrel{\mathsf{tr}}{\Rightarrow} A'\}/\sim$.

Note that the processes defined in Subsection 3.2.3.2 are not image-finite.

Theorem 3.2. *Let A and B two closed extended processes with $\mathrm{dom}(A) = \mathrm{dom}(B)$ and such that A and B are image-finite. We have that:*

$$A \approx_m B \text{ implies that } A \approx_t B$$

Proof. Assume that $A \not\approx_t B$. We assume w.l.o.g. that $A \not\sqsubseteq_t B$. In such a case, there exists a witness for the non equivalence. This means that there exists $(\mathsf{tr}, \phi) \in \mathsf{trace}(A)$ such that $\mathit{bnames}(\mathsf{tr}) \cap \mathit{fnames}(B) = \emptyset$, and

1. either there does not exist Φ' such that $(\mathsf{tr}, \Phi') \in \mathsf{trace}(B)$;
2. or for all $(\mathsf{tr}, \Phi') \in \mathsf{trace}(B)$, we have that $\Phi' \not\sim \Phi$.

Moreover, we assume that no name in tr is bounded twice (*i.e.* $\nu a.$ can not occur twice in tr) and bounded names in tr are distinct from free names that occur in A, B, and tr.

We build an evaluation context C according to the trace tr and also the tests that witness the fact that static equivalence does not hold. Let $S_{\mathsf{tr}} = \{\Phi' \mid (\mathsf{tr}, \Phi') \in \mathsf{trace}(B)\}$. Since B is image-finite, we know that $S_{\mathsf{tr}}/\sim$ is finite. Let $\{\Phi'_1, \ldots, \Phi'_m\} = S_{\mathsf{tr}}/\sim$. If we are in the first case, *i.e.* $S_{\mathsf{tr}} = \emptyset$, we have that $m = 0$.

We know that $\{1, \ldots, m\} = T^+ \uplus T^-$ with:

— for each $i \in T^+$, there exist two terms M_i and N_i such that $\mathit{fvars}(M_i) \cup \mathit{fvars}(N_i) \subseteq \mathrm{dom}(\Phi)$, $(M_i =_{\mathsf{E}} N_i)\Phi$, and $(M_i \neq_{\mathsf{E}} N_i)\Phi'_i$; and

— for each $i \in T^-$, there exist two terms M_i and N_i such that $\mathit{fvars}(M_i) \cup \mathit{fvars}(N_i) \subseteq \mathrm{dom}(\Phi)$, $(M_i \neq_{\mathsf{E}} N_i)\Phi$, and $(M_i =_{\mathsf{E}} N_i)\Phi'_i$.

Let bad be a fresh channel name that does not occur in A and B. Let $P_1, \ldots, P_m, P_{m+1}$ be the plain processes defined as follows:

— $P_{m+1} \stackrel{def}{=} \mathsf{out}(\mathsf{bad}, \mathsf{bad}).0$

— for $1 \leq i \leq m$, we define P_i as follows:

$$P_i \stackrel{def}{=} \mathtt{if}\ M_i = N_i\ \mathtt{then}\ P_{i+1}\ \mathtt{else}\ 0 \qquad \text{when } i \in T^+$$
$$P_i \stackrel{def}{=} \mathtt{if}\ M_i = N_i\ \mathtt{then}\ 0\ \mathtt{else}\ P_{i+1} \qquad \text{when } i \in T^-$$

Let $\{a_1, \ldots, a_k\}$ be channel names that occur free in A, B, and tr. Let $\mathcal{X}^0_{ch} = \{x_{a_1}, \ldots, x_{a_k}\}$ be a set of variables of channel type, and $\sigma = \{x_{a_1} \mapsto a_1, \ldots, x_{a_k} \mapsto a_k\}$. We define C such that $C = Q(\mathsf{tr}, \mathcal{X}^0_{ch}) \mid \_$ where $Q(\mathsf{tr}, \mathcal{X}_{ch})$ is defined by recurrence on tr as follows:

— if $\mathsf{tr} = \epsilon$ then $Q(\mathsf{tr}, \mathcal{X}_{ch}) = P_1$;

— if $\mathsf{tr} = \mathit{in}(a, M).\mathsf{tr}'$ then $Q(\mathsf{tr}, \mathcal{X}_{ch}) = \mathsf{out}(x_a\sigma, M).Q(\mathsf{tr}', \mathcal{X}_{ch})$;

— it $\mathsf{tr} = \mathit{out}(a, c).\mathsf{tr}'$ then $Q(\mathsf{tr}, \mathcal{X}_{ch}) = \mathsf{in}(x_a\sigma, y).$ if $y = x_c\sigma$ then $Q(\mathsf{tr}', \mathcal{X}_{ch})$ else 0 where y is fresh variable of channel type; and

— if $\mathsf{tr} = \nu c.\mathit{out}(a, c)$ then $Q(\mathsf{tr}, \mathcal{X}_{ch}) = \mathsf{in}(x_a\sigma, x_c).$ if $x_c \in \mathcal{X}_{ch}\sigma$ then 0 else $Q(\mathsf{tr}', \mathcal{X}'_{ch})$ where $\mathcal{X}'_{ch} = \mathcal{X}_{ch} \uplus \{x_c\}$.

We use the conditional if $u \in \{u_1, \ldots, u_k\}$ then 0 else P as a shortcut for

$$\mathsf{if}\ u = u_1\ \mathsf{then}\ 0\ \mathsf{else}\ (\mathsf{if}\ u = u_2\ \mathsf{then}\ 0\ \mathsf{else}\ (\ldots.\ (\mathsf{if}\ u = u_k\ \mathsf{then}\ 0\ \mathsf{else}\ P))).$$

We can see that $C[A]{\Downarrow}\mathsf{bad}$ since $(\mathsf{tr}, \Phi) \in \mathsf{trace}(A)$ and Φ satisfies by definition all the tests that are tested in P_m. However, by construction of C, we have that $C[B]$ can not emit on bad. □

The class of image-finite processes is not the only class of processes where the may-testing equivalence implies the trace equivalence. In [MNP02], they proved that the may-testing equivalence implies the trace equivalence using a property of their equational theory. They provide an equational theory E such that any frame could be characterized by a finite formula on terms modulo E, *i.e.* for all frames Φ, a formula ϕ is the characteristic formula of Φ if for all frames Φ', $\Phi' \sim \Phi$ is equivalent to Φ' (as substitution) satisfies ϕ. With such property on the equational theory, one can show that may-testing equivalence implies trace equivalence using a proof similar to the proof of Theorem 3.2. Typically, given a closed extended process A and a trace $(\mathsf{tr}, \Phi) \in \mathsf{trace}(A)$, we build the same context C as in the proof of Theorem 3.2, except that the processes P_k will be defined following the characteristic formula of Φ.

3.2.4 Observational equivalence

Observational equivalence has been initially introduced as a mean for proving may-testing or trace equivalence. Comparing to may-testing and trace equivalence, the observation equivalence also checks the bisimilarity of the two processes.

Definition 3.5 (observational equivalence)**.** Observational equivalence *is the largest symmetric relation $\mathcal{R}$ between closed extended processes with the same domain such that $A\mathcal{R}B$ implies:*

1. *if $A{\Downarrow}c$, then $B{\Downarrow}c$;*
2. *if $A \overset{\epsilon}{\Mapsto} A'$, then $B \overset{\epsilon}{\Mapsto} B'$ and $A'\mathcal{R}B'$ for some B';*
3. *$C[A]\mathcal{R}C[B]$ for all closing evaluation contexts C.*

However, proofs of observational equivalences are difficult because of the universal quantification over all contexts. Therefore, an alternative definition has been proposed, considering labeled transitions for the processes.

Definition 3.6 (labeled bisimilarity $\approx$)**.** Labeled bisimilarity *is the largest symmetric relation $\mathcal{R}$ on closed extended processes such that $A\ \mathcal{R}\ B$ implies*

1. *$\phi(A) \sim \phi(B)$,*
2. *if $A \overset{\tau}{\mapsto} A'$, then $B \overset{\epsilon}{\Mapsto} B'$ and $A'\ \mathcal{R}\ B'$ for some B',*
3. *if $A \overset{\ell}{\mapsto} A'$ and $bnames(\ell) \cap fnames(B) = \emptyset$ then $B \overset{\ell}{\Mapsto} B'$ and $A'\ \mathcal{R}\ B'$ for some B'.*

Example 3.8. *Consider the theory $\mathsf{E}_{\mathsf{aenc}}$ and the two processes $P_a = \mathsf{out}(c, \mathsf{aenc}(a, \mathsf{pk}(sk)))$ and $P_b = \mathsf{out}(c, \mathsf{aenc}(b, \mathsf{pk}(sk)))$. We have that $\nu sk.P_a \approx \nu sk.P_b$ whereas $P_a \not\approx P_b$. These results are direct consequences of the static (in)equivalence relations stated and discussed in Example 3.5.*

It has been shown that observational equivalence coincides with labeled bisimilarity [AF01, Liu11]. In the literature, one can also find several results about the relation between observational and trace equivalence. As expected, observational equivalence is strictly stronger than trace equivalence. J. Engelfriet has shown that observational equivalence and trace equivalence actually coincide in a general model of parallel computation with atomic actions, when processes are *determinate* [Eng85]. In [CD09a], the authors generalise this result to the applied pi calculus and show that a large class of processes, named *simple processes*, enjoys the determinacy property.

3.3 Some security properties

We presented different behavioural equivalences in Section 3.2 which typically differ from the power we give to the attacker. When describing the security properties we will use the trace equivalence, *i.e.* $\approx_t$, however all these security properties are still valid when replacing the trace equivalence by an other behavioural equivalence. Of course, since some behavioural equivalences are stronger than others (see Section 3.2), one protocol might satisfy a security property for one behavioural equivalence and not for an other equivalence. We present in this section some security properties that has been studied in the literature and we provide small examples to illustrate them.

3.3.1 Guessing attacks

In our model, we assume that all cryptographic primitives are perfect. For example, an intruder cannot obtain a from $\mathsf{senc}(a, k)$ without knowing the secret key k. Typically, this assumption relies on the length of the key to ensure that an intruder cannot try all possible passwords, *i.e.* break the cipher using a brute force attack. However, in many applications such as mail, online bank account, etc, a human user will usually use a password whose length is at most 20 characters. Thanks to the power of nowadays computers, an intruder may be able deduce these passwords using a brute force attack. These short passwords are called *weak secret*.

However, a weak secret is not necessary a flaw in the security of a protocol. Indeed, during a brute force attack, an intruder still has to distinguish the cases where he guessed the correct key or not. Hence the notion of *guessing attack* was introduced to represent the capability of an intruder to deduce weak secret and obtain relevant information from it. Typically, a guessing attack works as follows: The intruder first interacts with several sessions of a protocol (usually

small number of sessions since most of online protocols allow only a small number of fail attempts on your password) then in a next phase the intruder may try *offline* all possible passwords on the message he previously received, *i.e. guessing a key is done without interaction with the protocols.*

Although guessing attacks are considered as equivalence properties, they are not modelled by equivalence of processes but equivalence of frames instead. The is formalise by the following definition [DKR08, Bau05].

Definition 3.7. *Let $\nu k.\Phi$ be a closed frame. Assume that k is a weak secret. We say that $\nu k.\Phi$ is resistant to guessing attacks on k if, and only if, $\nu k.(\Phi \mid \{^{k}/_{x}\}) \sim \nu n.\nu k.(\Phi \mid \{^{n}/_{x}\})$ where $x \notin \mathrm{dom}(\Phi)$ and n is a fresh name.*

An extended process A is resistant to guessing attacks on $k \in bnames(A)$ if for every $(\mathsf{tr}, B) \in \mathsf{trace}(A)$, $\Phi(B)$ is resistant to guessing attacks on k.

Example 3.9. *Consider the Handshake protocol given in Example 3.1. We show that the handshake protocol is not resistant to guessing attacks on k. Indeed, let's denote A the process from Example 3.1. Hence $(\mathsf{tr}, \nu k_{AB}.\Phi) \in \mathsf{trace}(A)$ where $\mathsf{tr} = \nu w_1.out(c, w_1).in(c, w_1).\nu w_2.out(c, w_2)$ and $\Phi = \nu N.\nu t.(\{^{\mathsf{senc}(N,k_{AB})}/_{w_1}\} \mid \{^{\mathsf{senc}(\mathsf{f}(N),k_{AB})}/_{w_2}\}$. However, one can note that $\nu k_{AB}.(\Phi \mid \{^{k_{AB}}/_{w_3}\}) \not\sim \nu k_{AB}.\nu k(\Phi \mid \{^{k}/_{w_3}\})$. Indeed, the test $\mathsf{sdec}(w_2, w_3) =_{\mathsf{E}} \mathsf{f}(\mathsf{sdec}(w_1, w_3))$ can distinguish the two frames. This attacks is well known and can be found in the literature [GML+93].*

3.3.2 Strong secrecy

When two honest agents exchange a secret during the execution of a protocol, we usually want that this secret may never be deduce by an intruder. This security property, called *simple secrecy*, is in fact a reachability property and do not require behavioural equivalence to be modelled.

On the other hand, there exists a stronger security property, called *strong secrecy* and introduced in [Bla04], where an intruder must not notice when the value of the secret changes. The strong secrecy can be model as follows:

Definition 3.8. *Let A be a closed extended process such that $A \equiv C[A' \mid \{^{M}/_{x}\}]$ for some evaluation context C, extended process A', term M. We say that A preserves the strong secrecy of M if we have:*

$$C[A' \mid \{^{M}/_{x}\}] \approx_t \nu k.C[A' \mid \{^{k}/_{x}\}]$$

The original definition of strong secrecy can be found in [Bla04] and is slightly stronger that the one we provide. Indeed, instead of checking the strong secrecy of a specific message, they check the strong secrecy of a free variable which can be replaced by any message. While these two notions do not coincide in general, they do coincide for many usual equational theories.

3.3.3 Anonymity

Anonymity is informally defined by the ISO/IEC standard 15408 [ISO09] as the property ensuring that *a user may use a service or a resource without disclosing the user's identity.* Formally, anonymity has been defined to hold [ACRR10] when an outside observer cannot tell the difference between a system in which the user with a publicly known identity id_0 executes the analysed protocol, from the system where id_0 is not present at all.

Definition 3.9. *Let P be a closed extend process. Let $\nu\tilde{k}$ stands for $\nu k_1 \ldots \nu k_n$ where $k_1, \ldots, k_n$ are long term secrets used in P. We say that P satisfies anonymity if the following equivalence holds:*

$$\nu\tilde{k}.((!\nu id.\ !P) \mid !P\{id_0/id\}) \approx_t \nu\tilde{k}.(!\nu id.\ !P)$$

where id might be contained in P and id_0 is a fresh free name.

Intuitively, anonymity is satisfied if an observer cannot tell if the user id_0 (known to the attacker) has been executing the protocol P or not. Note that if P does not contain id, the anonymity trivially holds.

If we look at, as it is often done in automatic protocol verification, a bounded number of sessions of the protocol P, the above formal property needs to be adapted. In the case of one session for example, we will say that P preserves anonymity if the following equivalence holds

$$\nu\tilde{k}.\ P\{id_1/id\} \approx \nu\tilde{k}.\ P\{id_2/id\}$$

Informally, this corresponds to the fact that an attacker cannot tell if a session of P is executed by id_1 or by some other user id_2.

Example 3.10. *We consider a simplified version of a protocol given in [AF04] designed for transmitting a secret without revealing its identity to other participants. In this protocol, A wishes to engage in communication with B whereas B is willing to talk to A. However, A does not want to compromise its privacy by revealing its identity or the identity of B more broadly. The participants A and B proceed as follows:*

$$\begin{array}{lcl} A \to B & : & \mathsf{aenc}(\langle N_a, \mathsf{pk}(sk_A)\rangle, \mathsf{pk}(sk_B)) \\ B \to A & : & \mathsf{aenc}(\langle N_a, \langle N_b, \mathsf{pk}(sk_B)\rangle\rangle, \mathsf{pk}(sk_A)) \end{array}$$

First A sends to B a nonce N_a and its public key encrypted with the public key of B. If the message is of the expected form then B sends to A the nonce N_a, a freshly generated nonce N_b and its public key, all of this being encrypted with the public key of A. Otherwise, B sends out a "decoy" message: $\mathsf{aenc}(N_b, \mathsf{pk}(sk_B))$. This message should basically look like B's other message from the point of view of an outsider. This is important since the protocol is supposed to protect the identity of the participants.

When A receives a message from B, A tries to decrypt it and checks that the second component of the decryption corresponds to the nonce N_a he first created. He also checks that the third component corresponds to the public key of B. If the checks fail then A does nothing. After a successful execution of this protocol, A and B will be able to use N_A and N_b as shared secrets. However for the sake of simplicity, we will consider here that they do nothing.

In this protocol, we assume that the association between principals and their public keys is known, even to the intruder. This might be model in the applied pi calculus by a server that outputs the public key of a principal associated with its identification.

A session of role B played by agent b with a can be modelled by the plain process $B(b,a)$ where $N = \mathsf{adec}(y, sk_b)$. Note that B is not given the value sk_a but is directly given the value $\mathsf{pk}(sk_a)$, that is the public key corresponding to A's private key. Similarly, a session of role A played by agent a with b can be modeled by the plain process $A(a,b)$ where $M = \mathsf{adec}(z, sk_a)$.

$$A(a,b) \stackrel{def}{=} \nu n_a.\mathsf{out}(c, \mathsf{aenc}(\langle n_a, \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b))).\mathsf{in}(c,z).0$$

$$\begin{array}{l} B(b,a) \stackrel{def}{=} \nu n_b.\mathsf{in}(c,y).\mathsf{if}\ \mathsf{proj}_2(N) = \mathsf{pk}(sk_a) \\ \qquad\qquad \mathsf{then}\ \mathsf{out}(c, \mathsf{aenc}(\langle \mathsf{proj}_1(N), \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))).0 \\ \qquad\qquad \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(sk_b))).0 \end{array}$$

Intuitively, this protocol preserves anonymity of the participant a if an attacker cannot distinguish whether b is willing to talk to a (represented by the process $B(b,a)$) or willing to talk to a' (represented by the process $B(b,a')$), provided that a, a' and b are honest participants. This can be modeled by the following equivalence:

$$\nu sk_a.\nu sk_{a'}.\nu sk_b.\big(S \mid B(b,a) \mid A(a,b)\big) \approx_t \nu sk_a.\nu sk_{a'}.\nu sk_b.\big(S \mid B(b,a') \mid A(a',b)\big)$$

where S represents the server that associates principals and their public keys, i.e.

$$S \stackrel{def}{=} \{^{\langle a, \mathsf{pk}(sk_a)\rangle}/_{z_1}\} \mid \{^{\langle a', \mathsf{pk}(sk_{a'})\rangle}/_{z_2}\} \mid \{^{\langle b, \mathsf{pk}(sk_b)\rangle}/_{z_3}\}$$

This is a non-trivial equivalence however the "decoy" message plays an important role. Indeed, considering now the process $B^-(b,a)$ where, instead of sending a decoy message when the test fail,

the process does nothing:

$$B^-(b,a) \stackrel{def}{=} \nu n_b.\mathsf{in}(c,y).\mathsf{if}\ \mathsf{proj}_2(N) = \mathsf{pk}(sk_a) \\ \mathsf{then}\ \mathsf{out}(c,\mathsf{aenc}(\langle \mathsf{proj}_1(N), \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))).0 \\ \mathsf{else}\ 0$$

In such a case, the trace equivalence does not hold anymore. This can be easily shown by considering the sequence of actions $s = in(c, \mathsf{aenc}(\langle n_i, \mathsf{proj}_2(z_1)\rangle, \mathsf{proj}_2(z_3))).\nu x.out(c,x)$. *We have that* $(s,\Phi) \in \mathsf{trace}\big(\nu sk_a.\nu sk_{a'}.\nu sk_b.(S \mid B^-(b,a) \mid A(a,b))\big)$ *for some* Φ. *However, this sequence* s *does not have a corresponding trace for the process* $\nu sk_a.\nu sk_{a'}.\nu sk_b.\big(S \mid B^-(b,a') \mid A(a',b)\big)$.

3.3.4 Unlinkability

Unlinkability is informally defined by the ISO/IEC standard 15408 [ISO09] as the property ensuring that *a user may make multiple uses of a service or a resource without others being able to link these uses together*. Formally, unlinkability has been defined to hold [ACRR10] when a system in which the analysed protocol can be executed by each user multiple times looks the same to an outside observer that the system in which the analysed protocol can be executed by each user at most once.

Definition 3.10. *Let* P *be a closed extend process. Let* $\nu\tilde{k}$ *stands for* $\nu k_1 \ldots \nu k_n$ *where* $k_1, \ldots, k_n$ *are long term secrets used in* P. *We say that* P *satisfies unlinkability if the following equivalence holds:*

$$\nu\tilde{k}.!\nu id.!P \approx_t \nu\tilde{k}.!\nu id.P$$

where id might be contained in P.

In other words, unlinkability is satisfied if an observer cannot tell if the users can execute multiple or at most once the protocol P. Note that if P does not contain id then the equivalence trivially holds.

Now one can also look at a bounded number of sessions of the protocol. In this case we need to consider at least two sessions. Consider two sessions of the protocol, we will say that P preserves unlinkability if the following equivalence holds

$$\nu\tilde{k}.\nu id.\ (P \mid P) \approx_t \nu\tilde{k}.(\nu id.P \mid \nu id.P)$$

Informally, this corresponds to the fact that an attacker cannot tell if the two sessions of P are executed by the same user, or by two different users.

Note that if we consider more than two sessions, several equivalences are possible to express unlinkability. For example, if you consider three sessions of the protocol, one might consider that the three sessions are executed by the same user and one might also consider that two out of three sessions are executed by the same user while the third session is executed by another user. Section 3.4 illustrates this security property on the e-passport protocol.

3.4 The e-passport protocol

One of the purpose of this thesis is to apply our results (practical and theoretical) on concrete protocols such as the *e-passport protocol*. Indeed, passports now contain a chip that stores additional information such as pictures and fingerprints of its holder. In order to ensure privacy, these chips include a mechanism that do not let the passport disclose private information to external users. Section 3.3 formally describes the *e-passport protocol* that was partially studied in [ACRR10] and the security properties we are interested in, *i.e.* anonymity and unlinkability.

An electronic passport (or e-passport) is a paper passport with an RFID chip that stores the critical information printed on the passport. The International Civil Aviation Organisation (ICAO) standard [ICA04] specifies the communication protocols that are used to access these information.

3.4.1 Protocols description

The information stored in the chip is organised in data groups (dg_1 to dg_{19}). For example, dg_5 contains a JPEG copy of the displayed picture, and dg_7 contains the displayed signature. The verification key $\mathsf{vk}(sk_P)$ of the passport, together with its certificate $\mathsf{sign}(\mathsf{vk}(sk_P), sk_{DS})$ issued by the Document Signer authority are stored in dg_{15}. The corresponding signing key sk_P is stored in a tamper resistant memory, and cannot be read or copied. For authentication purposes, a hash of all the dgs together with a signature on this hash value issued by the Document Signer authority are stored in a separate file, the Security Object Document:

$$sod \stackrel{def}{=} \langle \mathsf{sign}(\mathsf{h}(dg_1, \ldots, dg_{19}), sk_{DS}),\ \mathsf{h}(dg_1, \ldots, dg_{19}) \rangle.$$

The ICAO standard specifies several protocols through which these information can be accessed. First, the Basic Access Control (BAC) protocol establishes session keys $ksenc$ and $ksmac$ to prevent skimming and eavesdropping on the subsequent communication with the e-passport. Once the BAC protocol has been successfully executed, the reader gains access to the information stored in the RFID tag through the Passive Authentication and the Active Authentication protocols that can be executed in any order.

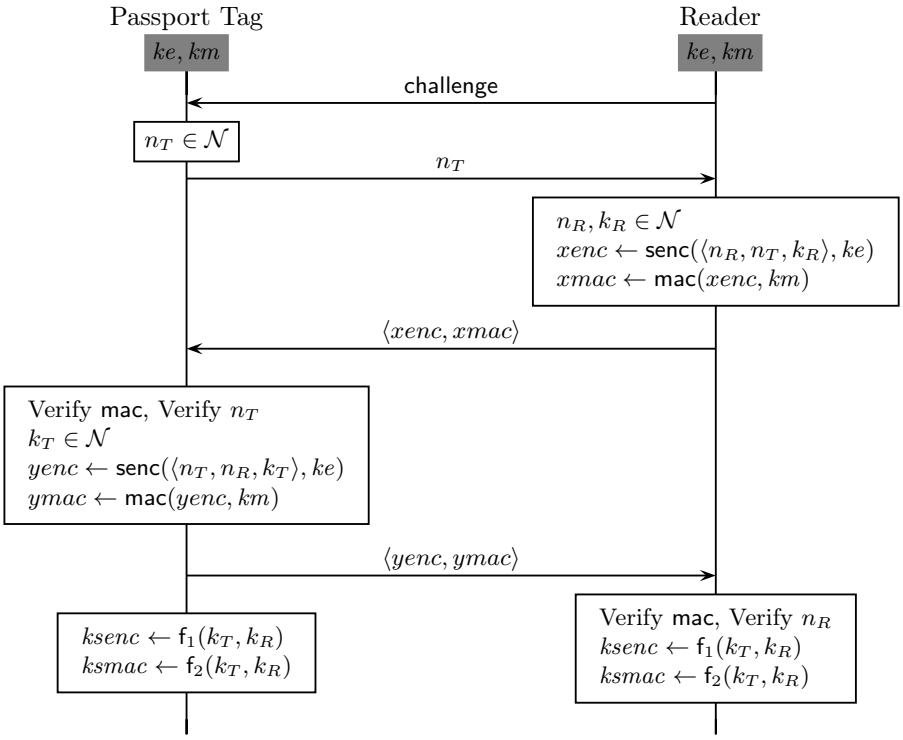


Figure 3.2: Basic Access Control protocol

The Basic Access Control (BAC) protocol is a key establishment protocol (see Figure 3.2). It relies on the keys ke and km that are printed on the passport at the same page as your personal data and your picture (usually, the customs officer get the key ke and km by optically scanning your passport). These keys are usually composed of some of your personal data and some small random number, hence these keys are easily broken if the intruder knows some basic data of its target or if he had once access to its passport.

When the verification of a mac or the verification of the nonces n_T and n_R fail then an error message is outputted. [ACRR10] showed that the unlinkability of the BAC protocol is broken if the error messages are different (case of the french passport) hence we will consider that the errors messages outputted are the same (case of the english passport).

The detail of the functions f_1 and f_2 are not important. We use them to illustrate the fact that the key sessions that will be used in the next two protocols are derived from k_T and k_R.

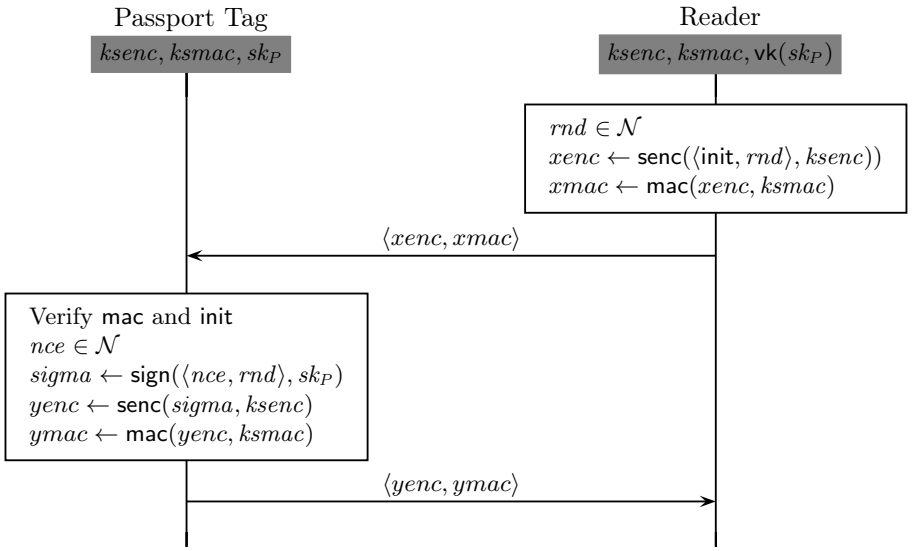


Figure 3.3: Active Authentication protocol

The Active Authentication (AA) protocol is an authentication mechanism that prevents cloning of the passport chip (see Figure 3.3). It relies on the fact that the secret key sk_P of the passport cannot be read or copied. The reader sends a random challenge to the passport, that has to return a signature on this challenge using its private signature key sk_P. The reader can then verify using the verification key $\mathsf{vk}(sk_P)$ that the signature was built using the expected passport key.

The Passive Authentication (PA) protocol is an authentication mechanism that proves that the content of the RFID chip is authentic (see Figure 3.4). Through PA the reader retrieves the information stored in the *dg*s and the *sod*. It then verifies that the hash value stored in the *sod* corresponds to the one signed by the Document Signer authority. It further checks that this hash value is consistent with the received *dg*s.

3.4.2 Security analysis

The protocols BAC, PA and AA rely on symmetric encryption, message authentication codes, signatures and the verification key generation function, to meet their security requirements.

According to the ICAO standard, once the keys *ksenc* and *ksmac* have been established using the BAC protocol, the reader can decide to execute PA and/or AA in any order. Formally, it corresponds to the parallel composition of PA and AA. Note that the execution of PA and AA occurring only when BAC succeed implies that the processes modeling BAC are in fact contexts on which the processes of PA and AA are plugged. However, instead of considering the three protocols at the same time, we might want to analyze first the privacy of BAC in isolation. Then, after considering that the keys *ksenc* and *ksmac* are "securely" pre-shared, we analyze the privacy

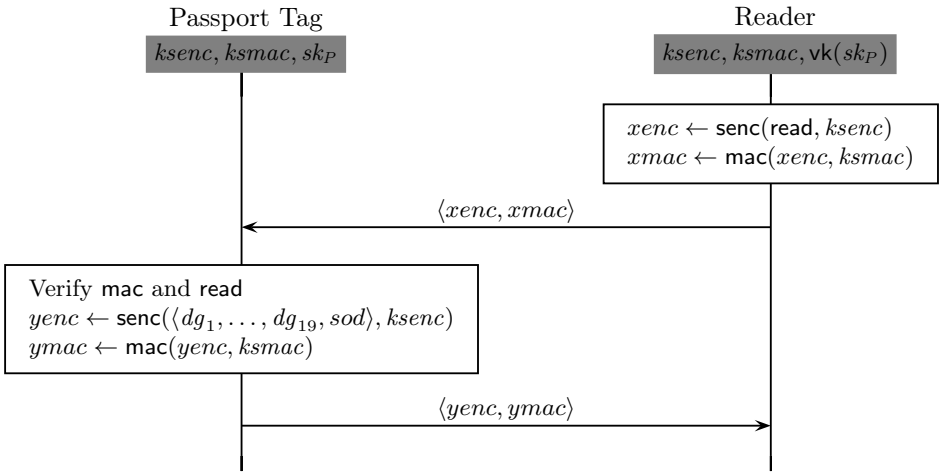


Figure 3.4: Passive Authentication protocol

of the parallel composition of PA and AA. Hence, in this section, we will provide the equivalences that model anonymity and unlinkability for BAC protocol in isolation, and also for the the parallel composition of PA and AA under securely pre-shared $ksenc$ and $ksmac$.

3.4.2.1 Security of *BAC*

We consider an arbitrary number of passports, each running an arbitrary number of times the BAC protocol. This situation can be modelled in our calculus as follows:

$$P \stackrel{def}{=} !\nu ke.\ \nu km.\ !BAC$$

where the subprocess BAC model one session of the BAC protocol respectively. Typically, each successful session of BAC create fresh session keys $ksenc$ and $ksmac$ that will be used in the PA and AA protocols.

To express unlinkability discussed in Section 3.3.4, we need on one hand to consider a system in which e-passports can execute the BAC protocol multiple times, and on the other hand a system in which e-passports can execute the BAC protocol at most once. This is modeled by the following equivalence:

$$!\nu ke.\ \nu km.\ !BAC \approx_t !\nu ke.\ \nu km.BAC$$

Normally, to express anonymity discussed in Section 3.3, we would need to consider a victim's e-passport but in the BAC protocol, the only data that can relevantly identify an e-passport are the keys ke and km. However, if we allow the intruder to know the value of ke and km of the victim's passport then it trivially breaks the anonymity of the protocol (he only has to try decrypting messages, if the decryption succeed then it is the victim's passport else it is not). On the other, if we do not allow the intruder to know the value of the keys ke and km of the victim's passport then the equivalence would be trivial, *i.e.* $!\nu ke.\ \nu km.\ !BAC \approx_t !\nu ke.\ \nu km.\ !BAC$, and so the anonymity is trivially preserved.

3.4.2.2 Security of the parallel composition of *PA* and *AA*

This case is very similar to the privacy of the BAC protocol. However, there is a lot more information (i.e. nonces) that is used in the PA and AA protocols, hence one should be be careful

about the names that depend on the passport itself and the one that depend on a session of the protocols. We consider an arbitrary number of passports, each running an arbitrary number of times the PA and the AA protocols. This situation can be modelled in our calculus as follows:

$$P \stackrel{def}{=} \nu sk_{DS}.\ !\nu sk_P.\ \nu id.\ \nu sig.\ \nu pic.\ \ldots\ !\nu ksenc.\ \nu ksmac.\ (PA \mid AA)$$

where id, sig, pic, ... represent the name, the signature, the displayed picture, *etc* of the e-passport owner, *i.e.* the data stored in the dgs (1-14) and (16-19). The subprocesses PA and AA model one session of the PA and AA protocol respectively. The name sk_{DS} models the signing key of the Document Signing authority used in all passports. Each passport (identified by its signing key sk_P, the owner's name, picture, signature, ...) can run multiple times and in any order the PA and AA protocols, but with different secret session keys $ksenc$ and $ksmac$, that should be established through execution of the BAC protocol (but that we have abstracted from).

Anonymity. To express anonymity discussed in Section 3.3.3, we will need to consider a victim's e-passport, whose name id_0, signature sig_0, picture pic_0, *etc.* are known to the attacker. The victim's e-passport follows like any other e-passport the PA and AA protocols which can be respectively modelled by the following processes:

$$PA_0 \stackrel{def}{=} PA\{id_0/id, sig_0/sig, pic_0/pic, \ldots, sk_{P0}\}$$
$$AA_0 \stackrel{def}{=} AA\{id_0/id, sig_0/sig, pic_0/pic, \ldots, sk_{P0}\}$$

To formally express anonymity, we will consider the following contexts:

$$\begin{aligned} C[\__1, \__2] \stackrel{def}{=}\ & !\ \nu sk_P.\ \nu id.\ \nu sig.\ \nu pic.\ \ldots \\ & \quad !\ \nu ksenc.\ \nu ksmac.\ \__1 \\ & \mid\ \nu sk_{P0}.\ !\nu ksenc.\ \nu ksmac.\ \__2 \end{aligned}$$

where the second hole will be filled with the process modelling the victim's e-passport, while the first hole will be filled with the processes modelling any other e-passport. Note that the signing key sk_{P0} of the victim's passport is not known to the attacker. The latter may know the verification key $\mathsf{vk}(sk_{P0})$ (accessible from dg_{15}) but not the signing key which is stored in a tamper resistant memory.

This system will be compared to the one where the victim's e-passport is not present at all. For this we consider the following context:

$$C'[\_] \stackrel{def}{=} !\nu sk_P.\ \nu id.\ \nu sig.\ \nu pic.\ \ldots\ !\ \nu ksenc.\ \nu ksmac.\ \_$$

whose unique hole will be filled with the processes modelling any e-passport but the victim's. Hence the anonymity of the PA and the AA protocols is modelled by the following equivalence:

$$\nu sk_{DS}.C[PA \mid AA, PA_0 \mid AA_0] \approx_t \nu sk_{DS}.C'[PA \mid AA]$$

Unlinkability. To express unlinkability discussed in Section 3.3.4, we need on one hand to consider a system in which e-passports can execute the PA and AA protocols multiple times, and on the other hand a system in which e-passports can execute the PA and AA protocols at most once. For this we consider the two following contexts:

$$\begin{aligned} C[\_] \stackrel{def}{=}\ & !\nu sk_P.\ \nu id.\ \nu sig.\ \nu pic.\ \ldots \\ & \quad !\nu ksenc.\ \nu ksmac. \\ C'[\_] \stackrel{def}{=}\ & !\nu sk_P.\ \nu id.\ \nu sig.\ \nu pic.\ \ldots \\ & \quad \nu ksenc.\ \nu ksmac.\ \_ \end{aligned}$$

These two contexts differ on the replication before the generation of the session keys $ksenc$ and $ksmac$, modelling in the first case an unbounded number of executions of the process that

will fill the unique hole, and in the second case a unique session of the filling process. Hence the unlinkability of the PA and the AA protocols is modelled by the following equivalence:

$$\nu sk_{DS}.C[PA \mid AA] \approx_t \nu sk_{DS}.C'[PA \mid AA]$$

Chapter 4

Towards deciding trace equivalence

Contents

In the previous chapter, we presented several definitions of behavioural equivalence that can be used to model the observational capabilities of an intruder. Hence, when verifying an equivalence property on a cryptographic protocol, the choice of the behavioural equivalence that we have to check is crucial. As mentioned in the introduction, the question of knowing which equivalence is best-suited to model an intruder is disputable. Although the may-testing equivalence is, in our opinion, the most appropriate equivalence to model the observational capabilities of an intruder, we prefer the trace equivalence that is easier to manipulate. This choice does not matter when considering bounded number of sessions since both equivalences coincide in such a case (see Section 3.2.3).

Moreover, the proof of [CC08] indicates that standard cryptographic definitions for security, based on indistinguishable games, are soundly abstracted by trace equivalence, with no need for observational equivalence. Conversely, two processes may not be observationally equivalent and yet be cryptographically indistinguishable.

The main contribution of this chapter is to adapt, from existing results, a proof technique for deciding trace equivalence. Since replication very quickly yields to undecidability even in the simpler case of accessibility properties [DLMS99, AC02], we focus here on processes without replication.

The applied pi calculus is elegant and convenient for expressing security protocols. However, its syntax and semantics (in particular name restriction and parallel composition) does not ease the verification task. In contrast, constraint systems are much simpler and capture exactly the core of protocol executions. They have shown their usefulness for analysing security protocols, for secrecy and authentication properties (*e.g.* [MS01, CLS03, CDM11, CKRT03]) as well as equivalence properties (*e.g.* [DKR07, CD09a]).

We show a reduction result for general processes without replication and for arbitrary equational theories. We reduce the decidability of trace equivalence (for bounded processes) to deciding symbolic equivalence between sets of constraint systems. To transfer executions from the applied pi

calculus to constraint systems, we introduce an intermediate calculus inspired by [DKR07, Liu11]. But contrary to their work, we allow replication in the process. This is useful when we want to prove some theoretical results such as our work on composition of trace equivalence (see Chapter 5).

4.1 Intermediate calculus

As mentioned in [DKR07], the semantics of the applied pi calculus is not well-suited for defining a symbolic semantics. One of the main reason is the presence of the structural equivalence in the semantics. It leads to too many traces for the same sequence of actions that are not necessary when studying behavioural equivalence and in particular the trace equivalence. On the other hand, the semantics of intermediate processes are more linear than the applied pi calculus semantics.

4.1.1 Syntax

We consider a new infinite subset of $\mathcal{X}$, called *parameters*, $\mathcal{AX} = \{ax_1, ax_2, \ldots, ax_n, \ldots\}$. Typically, the parameters will be used to express the domain of a frame and will avoid the possible confusion with the variables inside a process.

Definition 4.1 (intermediate process)**.** *An* intermediate process *is a triple* $(\mathcal{E}; \mathcal{P}; \Phi)$ *where:*

- $\mathcal{E}$ *is a set of names that represents the names restricted in* $\mathcal{P}$*;*
- $\Phi = \{ax_1 \rhd t_1, \ldots, ax_n \rhd t_n\}$ *where* $t_1, \ldots, t_n$ *are ground terms, and* $ax_1, \ldots, ax_n \in \mathcal{AX}$ *are parameters;*
- $\mathcal{P}$ *is a multiset of* plain processes *such that null processes are removed and* $fvars(\mathcal{P}) = \emptyset$*.*

Additionally, we require intermediate processes to be variable distinct*,* i.e. *any variable is at most bound once.*

Given a sequence $\Phi = \{ax_1 \rhd t_1, \ldots, ax_n \rhd t_n\}$ where $t_1, \ldots, t_n$ are terms, we also denote by Φ its associated frame, *i.e.* $\{^{t_1}/_{ax_1}\} \mid \ldots \mid \{^{t_n}/_{ax_n}\}$.

Given a closed extended process A of the original applied pi, we can easily transform it into an intermediate process $\tilde{A} = (\mathcal{E}; \mathcal{P}; \Phi)$ such that A is structurally equivalent to $\mathcal{E}.(\mathcal{P} \mid \Phi)$ up to a renaming of the variables in $\mathrm{dom}(\Phi)$. The idea is to rename names and variables to avoid clashes, to apply the active substitutions, to remove the restrictions on variables, and finally to push the restrictions on names, that define the scope of active substitutions with free variable, in front of the process.

Example 4.1. *Consider the extended process* A *described below (*M *and* N *are some terms that do not contain* n*):*

$$\nu sk.\nu x.(\mathsf{out}(c, \mathsf{aenc}(x, \mathsf{pk}(sk))).\nu n.\mathsf{out}(c, n).\mathsf{out}(c, y) \mid \{^{M}/_{x}\} \mid \{^{N}/_{y}\})).$$

An intermediate process A' *associated to* A *is:*

$$\begin{aligned} A' &= (\mathcal{E}; \mathcal{P}; \Phi) \\ &= (\{sk\}; \mathsf{out}(c, \mathsf{aenc}(M, \mathsf{pk}(sk))).\nu n.\mathsf{out}(c, n).\mathsf{out}(c, N); \{ax_1 \rhd N\}). \end{aligned}$$

We have that $A\{y \mapsto ax_1\} \equiv \nu\mathcal{E}.(\mathcal{P} \mid \Phi)$*.*

4.1.2 Semantics

The semantics for intermediate processes is given in Figure 4.1. The rules Open-Ch$_i$, Out-Ch$_i$, In$_i$, Out-T$_i$ and Par$_i$ correspond respectively to the rule Open-Ch, Out-Ch, In, Out-T and Par in the semantic of the applied pi calculus. The rules Comm$_i$, Then$_i$ and Else$_i$ correspond

to the internal reduction of the applied pi calculus. The rules NEW$_i$ and REPL$_i$ do not appear explicitly in the applied pi calculus but correspond in fact to the rule STRUCT.

For the rule OPEN-CH$_i$, we consider a new infinite subset of $\mathcal{N}_{ch}$, *i.e.* set of name of channel type, denoted $\mathcal{C}h$ such that $\mathcal{C}h = \{ch_1, \dots ch_n, \dots\}$. Hence, when in the rules in Figure 4.1, we indicate "ch_m is a fresh channel name", it implies that the last application of the rule OPEN-CH$_i$ in the derivation used the name ch_{m-1}. Intuitively, we use this set of channel names to dismiss the sequence of actions tr equivalent through renaming of the bounded names of channel type in tr. Note that one of the purpose of using of the parameters in $\mathcal{AX}$ is similar to the purpose of using the channel names in $\mathcal{C}h$.

Let $\mathcal{A}_i$ be the alphabet of actions for the intermediate semantics. For every $w \in \mathcal{A}_i^*$ the relation $\xrightarrow{w}_i$ on intermediate processes is defined in the usual way. For $s \in (\mathcal{A}_i \setminus \{\tau\})^*$, the relation $\stackrel{s}{\Rightarrow}_i$ on intermediate processes is defined by: $A \stackrel{s}{\Rightarrow}_i B$ if, and only if there exists $w \in \mathcal{A}_i^*$ such that $A \xrightarrow{w}_i B$ and s is obtained by erasing all occurrences of τ. Note that by definition, intermediate processes are closed.

$$(\mathcal{E}; \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_1\} \uplus \mathcal{P}; \Phi) \text{ if } u =_{\mathsf{E}} v \qquad (\text{THEN}_i)$$

$$(\mathcal{E}; \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_2\} \uplus \mathcal{P}; \Phi) \text{ if } u \neq_{\mathsf{E}} v \qquad (\text{ELSE}_i)$$

$$(\mathcal{E}; \{\mathsf{out}(p,u).Q_1; \mathsf{in}(p,x).Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_1; Q_2\{x \mapsto u\}\} \uplus \mathcal{P}; \Phi) \qquad (\text{COMM}_i)$$

$$(\mathcal{E}; \{\mathsf{in}(p,x).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{in(p,M)}_i (\mathcal{E}; \{Q\{x \mapsto u\}\} \uplus \mathcal{P}; \Phi) \qquad (\text{IN}_i)$$
if $p \notin \mathcal{E}$, $M\Phi = u$, $fvars(M) \subseteq \mathrm{dom}(\Phi)$ and $fnames(M) \cap \mathcal{E} = \emptyset$

$$(\mathcal{E}; \{\mathsf{out}(p,u).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{\nu ax_n.out(p,ax_n)}_i (\mathcal{E}; \{Q\} \uplus \mathcal{P}; \Phi \cup \{ax_n \rhd u\}) \qquad (\text{OUT-T}_i)$$
if $p \notin \mathcal{E}$, and ax_n is a variable such that $n = |\Phi| + 1$

$$(\mathcal{E}; \{\mathsf{out}(p,c).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{out(p,c)}_i (\mathcal{E}; \{Q\} \uplus \mathcal{P}; \Phi) \qquad (\text{OUT-CH}_i)$$
if $p, c \notin \mathcal{E}$

$$(\mathcal{E}; \{\mathsf{out}(p,c).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{\nu ch_m.out(p,ch_m)}_i (\mathcal{E}; (\{Q\} \uplus \mathcal{P})\{c \mapsto ch_m\}; \Phi) \qquad (\text{OPEN-CH}_i)$$
if $p \notin \mathcal{E}$, $c \in \mathcal{E}$, ch_m is a fresh channel name

$$(\mathcal{E}; \{\nu k.Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E} \cup \{n\}; Q\{k \mapsto n\} \uplus \mathcal{P}; \Phi) \qquad (\text{NEW})$$
if n is a fresh name with the same type as k

$$(\mathcal{E}; \{!Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{!Q; Q\} \uplus \mathcal{P}; \Phi) \qquad (\text{REPL})$$

$$(\mathcal{E}; \{P_1 \mid P_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{P_1; P_2\} \uplus \mathcal{P}; \Phi) \qquad (\text{PAR})$$

where p, c are channel names, u, v are ground terms, and x is a variable.

Figure 4.1: Intermediate semantics

Example 4.2. *Consider the P the following protocol:*

$$\nu s.\nu sk.\ \big(\mathsf{out}(c_1, \mathsf{senc}(s, sk)) \mid \mathsf{in}(c_1, y).\mathsf{out}(c_2, \mathsf{sdec}(y, sk))\big).$$

P is a plain process hence its associated intermediate process is $\tilde{P} = (\emptyset; \{P\}; \emptyset)$. Consider now the sequence of transitions for P described in Example 3.4. We had $P \stackrel{\mathsf{tr}}{\Rightarrow} P' = \nu s.\nu sk.(\{^s/_y\} \mid \{^{\mathsf{aenc}(s,\mathsf{pk}(sk))}/_x\})$ with $\mathsf{tr} = \nu ax_1.out(c_1, ax_1).\ in(c_1, ax_1).\ \nu ax_2.out(c_2, ax_2)$. The corresponding

trace of P is obtained as follows:

$$\begin{array}{rl}
\tilde{P} \xrightarrow{\tau}_i \xrightarrow{\tau}_i \xrightarrow{\tau}_i & (\{s', sk'\}; \{\mathsf{out}(c_1, \mathsf{aenc}(s', \mathsf{pk}(sk')))); \mathsf{in}(c_1, y).\mathsf{out}(c_2, \mathsf{adec}(y, sk')\}; \emptyset) \\
\xrightarrow{\nu ax_1.out(c_1, ax_1)}_i & (\{s', sk'\}; \{\mathsf{in}(c_1, y).\mathsf{out}(c_2, \mathsf{adec}(y, sk')\}; \{ax_1 \rhd \mathsf{aenc}(s', \mathsf{pk}(sk'))\}) \\
\xrightarrow{\mathsf{in}(c_1, ax_1)}_i & (\{s', sk'\}; \{\mathsf{out}(c_2, M)\}; \{ax_1 \rhd \mathsf{aenc}(s', \mathsf{pk}(sk'))\}) \\
\xrightarrow{\nu ax_2.\mathsf{out}(c_2, ax_2)}_i & (\{s', sk'\}; \emptyset; \{ax_1 \rhd \mathsf{aenc}(s', \mathsf{pk}(sk')); ax_2 \rhd M\})
\end{array}$$

with $M = \mathsf{adec}(\mathsf{senc}(s', \mathsf{pk}(sk')), sk')$. Note that $M =_{\mathsf{E}} s'$. If we denote by $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1)$ the last intermediate process in the derivation and $\mathsf{tr}' = \nu ax_1.out(c_1, ax_1).\ in(c_1, ax_1).\ \nu y.out(c_2, ax_2)$, we have that $\tilde{P} \overset{\mathsf{tr}'}{\Rightarrow}_i (\mathcal{E}_1; \mathcal{P}_1; \Phi_1)$. Furthermore, if we denote σ the substitution $\{x \mapsto ax_1; y \mapsto ax_2\}$, we also have $\nu\mathcal{E}_1.\Phi_1 \equiv \Phi(P')\sigma$ and $\mathsf{tr}' = \mathsf{tr}\sigma$.

In Example 3.4, we also consider a sequence of transitions with an internal reduction, i.e. *$P \overset{\mathsf{tr}}{\Rightarrow} P' = \nu s.\nu sk.(\{^s/_y\})$ with $\mathsf{tr} = \nu y.out(c_2, y)$. The corresponding trace of P' is obtained as follows:*

$$\begin{array}{rl}
\tilde{P} \xrightarrow{\tau} \xrightarrow{\tau} \xrightarrow{\tau} & (\{s', sk'\}; \{\mathsf{out}(c_1, \mathsf{aenc}(s', \mathsf{pk}(sk')))); \mathsf{in}(c_1, y).\mathsf{out}(c_2, \mathsf{adec}(y, sk')\}; \emptyset) \\
\xrightarrow{\tau}_i & (\{s', sk'\}; \{\mathsf{out}(c_2, M)\}; \emptyset) \\
\xrightarrow{\nu ax_1.\mathsf{out}(c_2, ax_1)}_i & (\{s', sk'\}; \emptyset; \{ax_1 \rhd M\})
\end{array}$$

with $M = \mathsf{adec}(\mathsf{senc}(s', \mathsf{pk}(sk')), sk')$. Once again, if we denote by $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1)$ the last intermediate process in the derivation and $\mathsf{tr}' = \nu ax_1.out(c_2, ax_1)$, then $\tilde{P} \overset{\mathsf{tr}'}{\Rightarrow}_i (\mathcal{E}_1; \mathcal{P}_1; \Phi_1)$. Furthermore, if we denote σ the substitution $\{y \mapsto ax_1\}$, we also have $\nu\mathcal{E}_1.\Phi_1 \equiv \Phi(P')\sigma$ and $\mathsf{tr}' = \mathsf{tr}\sigma$.

In both sequences of transitions, the rule NEW$_i$ *creates fresh names s' and sk' while we could have kept s and sk. It is important to create fresh names each time to avoid clashes between names, especially when the name restriction is under a replication.*

4.1.3 Equivalence

Let $A = (\mathcal{E}_1; \mathcal{P}_1; \Phi_1)$ be an intermediate process. We define the set of its traces as follows:

$$\mathsf{trace}_i(A) = \{(s, \nu\mathcal{E}_2.\Phi_2) \mid (\mathcal{E}_1; \mathcal{P}_1; \Phi_1) \overset{s}{\Rightarrow}_i (\mathcal{E}_2; \mathcal{P}_2; \Phi_2) \text{ for some } (\mathcal{E}_2; \mathcal{P}_2; \Phi_2)\}$$

Two intermediate processes are in (intermediate trace) equivalence if for any trace, the two corresponding frames are statically equivalent.

Definition 4.2 ($\approx_t$ for intermediate processes)**.** *Let A and B be two intermediate processes having the same set of restricted names, i.e. $A = (\mathcal{E}_1; \mathcal{P}_1; \Phi_1)$, $B = (\mathcal{E}_2; \mathcal{P}_2; \Phi_2)$ and $\mathcal{E}_1 = \mathcal{E}_2$.*

The processes A and B are intermediate trace equivalent*, denoted by $A \approx_t B$, if for every $(s, \nu\mathcal{E}.\Phi) \in \mathsf{trace}_i(A)$ such that $bnames(s) \cap fnames(B) = \emptyset$ there exists $(s', \nu\mathcal{E}'.\Phi') \in \mathsf{trace}_i(B)$ such that $s = s'$ and $\nu\mathcal{E}.\Phi \sim \nu\mathcal{E}'.\Phi'$ (and conversely).*

Despite the differences between the two semantics, it can be shown (see Proposition 4.1) that the two notions of trace equivalence coincide.

Proposition 4.1. *Let A and B be two processes without replication. Consider the two associated intermediate processes: $(\mathcal{E}; \mathcal{P}_A; \Phi_A)$ and $(\mathcal{E}; \mathcal{P}_B; \Phi_B)$.*

*The processes A and B are trace equivalent (*i.e. *$A \approx_t B$ in the original applied pi calculus semantics) if, and only if, $(\mathcal{E}; \mathcal{P}_A; \Phi_A) \approx_t (\mathcal{E}; \mathcal{P}_B; \Phi_B)$.*

Example 4.3. *Coming back to Example 3.10, the anonymity of the private authentication protocol is given by the following equivalence.*

$$\nu sk_a.\nu sk_{a'}.\nu sk_b.\big(S \mid B(b, a) \mid A(a, b)\big) \approx_t \nu sk_a.\nu sk_{a'}.\nu sk_b.\big(S \mid B(b, a') \mid A(a', b)\big)$$

where S represent the server that associates principals and their public keys, i.e.

$$S \stackrel{def}{=} \{^{\langle a,\mathsf{pk}(sk_a)\rangle}/_{z_1}\} \mid \{^{\langle a',\mathsf{pk}(sk_{a'})\rangle}/_{z_2}\} \mid \{^{\langle b,\mathsf{pk}(sk_b)\rangle}/_{z_3}\}$$

where the processes B and A are both plain processes. This equivalence can also be expressed with intermediate processes. The server S will be put directly in the frame while the names $sk_a, sk_{a'}$ and sk_b will be put into the set of restricted names:

$$(\mathcal{E}; \{B(b,a) \mid A(a,b)\}; \Phi_0) \approx_t (\mathcal{E}; \{B(b,a') \mid A(a',b)\}; \Phi_0)$$

where:

— $\mathcal{E} = \{ska, ska', skb\}$*; and*

— $\Phi_0 = \{ax_1 \rhd \langle a, \mathsf{pk}(sk_a)\rangle, ax_2 \rhd \langle a', \mathsf{pk}(sk_{a'})\rangle, ax_3 \rhd \langle b, \mathsf{pk}(sk_b)\rangle\}$.

4.1.4 Bounded intermediate processes

As mentioned earlier, our intermediate semantics was inspired by [DKR07]. However, in their semantics, they do not have a rule for replication (since they consider processes without replication) but they also do not have a rule for name restriction. Indeed, [DKR07] considers a class of protocol where the restricted names where all pushed inside $\mathcal{E}$.

Example 4.4. *Consider the extended process A and its associated intermediate process A' in Example 4.1:*

$$A' \;=\; (\{sk\}; \mathsf{out}(c, \mathsf{aenc}(M, \mathsf{pk}(sk))).\nu n.\mathsf{out}(c,n).\mathsf{out}(c,N); \{ax_1 \rhd N\}).$$

where M and N do not contain n. We can build a similar intermediate process A' where we push n inside the set of restricted names:

$$A'' \;=\; (\{sk, n\}; \mathsf{out}(c, \mathsf{aenc}(M, \mathsf{pk}(sk))).\mathsf{out}(c,n).\mathsf{out}(c,N); \{ax_1 \rhd N\}).$$

Note that A' and A'' are not in structural equivalence but we have $A' \approx_t A''$. Indeed, structural equivalence does not allow one to push all the restrictions in front of a process.

Given an intermediate process $A = (\mathcal{E}; \mathcal{P}; \Phi)$ without replication and a fixed sequence of actions tr, the presence of name restrictions in $\mathcal{P}$ yields necessary an infinite set of traces with this specific sequence of actions tr due to the infinite choice of name n in the rule New. However, most of the frames of these traces only differ by renaming of bound variables. Intuitively, pushing the name inside the set of restricted names is similar to only considering one representative frame of $\{\Phi \mid (\mathsf{tr}, \Phi) \in \mathsf{trace}_i(A)\}/\equiv$.

From now on, we will say that an intermediate process $(\mathcal{E}; \mathcal{P}; \Phi)$ is *bounded* if $\mathcal{P}$ do not contain replication nor name restriction. Given a process $A = (\mathcal{E}; \mathcal{P}; \Phi)$ without replication such that names are bound only once and there is no clash between free and bound variables, we will say that the process $(\mathcal{E}'; \mathcal{P}'; \Phi')$ is its associated bounded intermediate process if $\Phi = \Phi'$, $\mathcal{E}' = \mathcal{E} \cup bnames(\mathcal{P})$ and $\mathcal{P}'$ is the multiset $\mathcal{P}$ where any name restriction, *i.e.* νn, are removed.

Example 4.5. *Coming back to Example 4.3, the anonymity of the private authentication protocol can be expressed with bounded processes by the following equivalence:*

$$(\mathcal{E}'; \{B'(b,a) \mid A'(a,b)\}; \Phi_0) \approx_t (\mathcal{E}'; \{B'(b,a') \mid A'(a',b)\}; \Phi_0)$$

where $\mathcal{E}' = \mathcal{E} \cup \{n_a, n_b\}$ and $A'(a,b)$, $B'(b,a)$ are defined such that:

$$A'(a,b) \stackrel{def}{=} \mathsf{out}(c, \mathsf{aenc}(\langle n_a, \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b))).\mathsf{in}(c,z).0$$

$$\begin{aligned} B'(b,a) \stackrel{def}{=} \mathsf{in}(c,y).\mathsf{if}\ \mathsf{proj}_2(N) = \mathsf{pk}(sk_a) \\ \mathsf{then}\ \mathsf{out}(c, \mathsf{aenc}(\langle \mathsf{proj}_1(N), \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))).0 \\ \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(sk_b))).0 \end{aligned}$$

4.2 Symbolic calculus

In this section, we propose a symbolic semantics for our calculus similar to [Bau07, DKR07, RT01, CZ06]. By treating inputs symbolically, our symbolic semantics avoids potentially infinite branching of execution trees due to inputs from the environment. Correctness is maintained by associating with each process a set of constraints on terms. As mentioned in the previous section, we will only consider bounded intermediate processes in the rest of this chapter.

4.2.1 Constraint system

We consider a new set $\mathcal{X}^2$ of variables called *second order variables* $X, Y, \ldots$ In order to avoid conflict between the variables in $\mathcal{X}^2$ and the one we already used to described the protocol and the processes, we consider a new set $\mathcal{X}^1$ of variables, called *first order variables*, for the variables used in the processes. Note that $\mathcal{X}^1$, $\mathcal{X}^2$, $\mathcal{AX}$ are all disjoint subsets of $\mathcal{X}$. We call *recipe*, usually denoted ξ, the terms in $\mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X}^2 \cup \mathcal{AX})$. We say that a recipe ξ is *closed* (or *ground*) if $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{AX})$. Given a recipe $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X}^2 \cup \mathcal{AX})$, we denote $param(\xi)$ the set of parameter in ξ, *i.e.* $vars(\xi) \cap \mathcal{AX}$.

A constraint system represents the possible executions of a protocol once an interleaving has been fixed.

Definition 4.3 (constraint system)**.** *A* constraint system *is a tuple* $(\mathcal{E}; \Phi; D; Eq)$ *where*

- $\mathcal{E}$ *is a set of names (names that are initially unknown to the attacker);*
- Φ *is a sequence of the form* $\{ax_1 \rhd t_1, \ldots, ax_n \rhd t_n\}$ *where* t_i *are terms and* ax_i *are variables of* $\mathcal{AX}$*.*
- D *is a set of deducible constraints of the form* $X, i \overset{?}{\vdash} x$ *with* $i \leq n$*,* $X \in \mathcal{X}^2$*,* $x \in \mathcal{X}^1$*.*
- Eq *is a set of equations and inequations of the form* $s \overset{?}{=} s'$ *or* $s \overset{?}{\neq} s'$ *where* s, s' *are first-order terms of same type.*

Intuitively, the terms t_i represent the terms sent on the network, their variables represent messages sent by the attacker. Moreover, a constraint $X, i \overset{?}{\vdash} x$ is meant to ensure that x will be replaced by a deducible term. The *size* of Φ, denoted $|\Phi|$ is its length n. Given a set D of constraints, we denote by $vars^1(D)$ (resp. $vars^2(D)$) the first order (resp. second order) variables of D, that is $vars^1(D) = fvars(D) \cap \mathcal{X}^1$ (resp. $vars^2(D) = fvars(D) \cap \mathcal{X}^2$).

We also assume the following conditions are satisfied on a constraint system:

1. for every $x \in vars^1(D)$, there exists a unique X such that $(X, i \overset{?}{\vdash} x) \in D$, and each variable X occurs at most once in D.
2. for every $1 \leq k \leq n$, for every $x \in vars^1(t_k)$, there exists $(X, i \overset{?}{\vdash} x) \in D$ such that $i < k$.

The last property implies that any variables in Φ must be previously introduced by a deducible constraint in D. Given a recipe ξ with parameters included in $ax_1, \ldots, ax_k$ and $\Phi = \{ax_1 \rhd t_1, \ldots, ax_n \rhd t_n\}$, $n \geq k$, $\xi\Phi$ denotes the term ξ where each ax_i has been replaced by t_i. The *structure* of a constraint system $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq)$ is given by $\mathcal{E}$, $|\Phi|$ and $\{(X, i) \mid (X, i \overset{?}{\vdash} x) \in D\}$. A *positive constraint system* is a constraint system that does not contain any inequation.

Example 4.6. *The tuple* $\mathcal{C}_1 = (\mathcal{E}; \Phi_0 \cup \{ax_4 \rhd t\}; D_1; Eq_1)$ *where*

- $\mathcal{E} = \{sk_a, sk_{a'}, sk_b, n_b, n_a\}$,
- $\Phi_0 = \{ax_1 \rhd \langle a, \mathsf{pk}(sk_a)\rangle, ax_2 \rhd \langle a', \mathsf{pk}(sk_{a'})\rangle, ax_3 \rhd \langle b, \mathsf{pk}(sk_b)\rangle\}$,
- $t = \mathsf{aenc}(\langle \pi_1(\mathsf{adec}(y, sk_b)), \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))$,
- $D_1 = \{Z_1, 3 \overset{?}{\vdash} z_1 x;\ Y, 3 \overset{?}{\vdash} y;\ Z_2, 3 \overset{?}{\vdash} z_2\}$*; and*

— $Eq_1 = \{z_1 \stackrel{?}{=} c;\ \mathsf{proj}_2(\mathsf{adec}(y, sk_b)) \stackrel{?}{=} \mathsf{pk}(sk_a);\ z_2 \stackrel{?}{=} c\}$.

is a constraint system. We will see that it corresponds to an execution of the process $B'(b, a)$ presented in Example 4.5.

Definition 4.4 (solution)**.** *A* solution *of a constraint system $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq)$ is a pair of substitutions (σ, θ) such that σ is a mapping from $vars^1(\mathcal{C})$ to $\mathcal{T}(\mathcal{F}, \mathcal{N})$, θ is a mapping from $vars^2(\mathcal{C})$ to $\mathcal{T}(\mathcal{F}, \mathcal{N} \setminus \{\mathcal{E}\}, \mathcal{AX})$, and:*

1. *for every $(X, k \stackrel{?}{\vdash} x) \in D$, we have that $(X\theta)(\Phi\sigma) = x\sigma$ and $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$;*
2. *for every $(s \stackrel{?}{=} s') \in Eq$, we have that $s\sigma =_{\mathsf{E}} s'\sigma$;*
3. *for every $(s \stackrel{?}{\neq} s') \in Eq$, we have that $s\sigma \neq_{\mathsf{E}} s'\sigma$.*

The substitution θ is called second-order solution *of $\mathcal{C}$, and the substitution σ is called the* first-order solution *of $\mathcal{C}$ associated to θ. The set of solutions of a constraint system $\mathcal{C}$ is denoted $\mathsf{Sol}(\mathcal{C})$. A constraint system $\mathcal{C}$ is* satisfiable *if $\mathsf{Sol}(\mathcal{C}) \neq \emptyset$.*

Intuitively, in the preceding definition the substitution θ stores the computation done by the adversary in order to compute the messages he sends (stored in σ) during the execution.

Example 4.7. *Continuing Example 4.6, a solution to $\mathcal{C}_1 = (\mathcal{E}_1; \Phi_1; D_1; Eq_1)$ is (σ, θ) where $\mathrm{dom}(\theta) = \{Z_1, Z_2, Y\}$, $Z_1\theta = Z_2\theta = c$, and $Y\theta = \mathsf{aenc}(\langle n_i, \mathsf{proj}_2(ax_1)\rangle, \mathsf{proj}_2(ax_3))$ with n_i a public name of base type (*i.e. *$n_i \notin \mathcal{E}_1$). Furthermore, σ is the substitution whose domain is $\{z_1, z_2, y\}$ and such that $z_1\sigma = z_2\sigma = c$, and $y\sigma = \mathsf{aenc}(\langle n_i, \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b))$.*

Note that given a constraint system and a substitution θ, there exists at most one substitution σ such that (σ, θ) is a solution of this constraint system. On the other hand, given a substitution σ, there might exist several substitutions θ such that (σ, θ) is a solution. Intuitively, this due to the fact that there might exist several ways, *i.e.* recipes, to deduce a term. For example, we can deduce a from $\langle a, a\rangle$ by applying the first or the second projection. This is why one can find works in the literature (*e.g.* [Bau07]) where the solutions of a constraint system are only described trough the second-order solution. However, having both substitutions as representation of a solution is, in our opinion, more intuitive and facilitate manipulation of solutions in proofs.

As for processes in the applied-pi calculus, it is also possible to define equivalence of (sets of) constraint systems. Two sets of constraint systems Σ and Σ' are in equivalence if for any solution of a constraint system in Σ, there exists a constraint system in Σ' that has the same solution and such that the resulting frames are in static equivalence.

Definition 4.5 (symbolic equivalence)**.** *Let Σ and Σ' be two sets of constraint systems that contain constraint systems having the same structure. We say that Σ and Σ' are in symbolic equivalence, denoted by $\Sigma \approx_s \Sigma'$, if for all $\mathcal{C} \in \Sigma$, for all $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, there exist $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ and $\nu\mathcal{E}.\Phi\sigma \sim \nu\mathcal{E}.\Phi'\sigma'$ (and conversely) where $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq)$ and $\mathcal{C}' = (\mathcal{E}; \Phi'; D'; Eq')$.*

Example 4.8. *Consider the sets of constraint systems $\Sigma = \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3\}$ and $\Sigma' = \{\mathcal{C}'_1, \mathcal{C}'_2, \mathcal{C}'_3\}$ as defined in Example 4.9. These two sets are in symbolic equivalence whereas the sets $\{\mathcal{C}_1\}$ and $\{\mathcal{C}'_1\}$ are not in symbolic equivalence. Indeed, consider the substitutions (σ, θ) given in Example 4.6. We obtain that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$ whereas $\theta \notin \mathsf{Sol}(\mathcal{C}'_1)$. Indeed, we have that $\theta = \{Z_1 \mapsto c, Z_2 \mapsto c, Y \mapsto \mathsf{aenc}(\langle n_i, \mathsf{proj}_2(ax_1)\rangle, \mathsf{proj}_2(ax_3))\}$. Thus, in order to satisfy item 1 of Definition 4.4, the substitution σ' should be $\sigma' = \{z_1 \mapsto c, z_2 \mapsto c, y \mapsto \mathsf{aenc}(\langle n_i, \mathsf{pk}(ska)\rangle, \mathsf{pk}(skb))\}$. However, such a substitution σ' will not satisfy item 2 since the equality $\pi_2(\mathsf{adec}(y\sigma', skb)) =_{\mathsf{E}} \mathsf{pk}(ska')$ does not hold. This allows us to conclude that $\{\mathcal{C}_1\} \not\approx_s \{\mathcal{C}'_1\}$.*

4.2.2 Syntax and semantics

From a bounded intermediate process $(\mathcal{E};\mathcal{P};\Phi)$, we can compute the set of constraint systems capturing its possible executions, starting from the symbolic process $(\mathcal{E};\mathcal{P};\Phi;\emptyset;\emptyset)$ and applying the rules defined in Figure 4.2.

Definition 4.6 (symbolic process)**.** *A* symbolic process *is a tuple* $(\mathcal{E};\mathcal{P};\Phi;D;Eq)$ *where:*

— $\mathcal{E}$ *is a set of names;*

— $\mathcal{P}$ *is a multiset of plain processes that do not contain name restriction, replication, where null processes are removed and such that* $fvars(\mathcal{P}) \subseteq \{x \mid X \overset{?}{\vdash} x \in D\}$*;*

— $(\mathcal{E};\Phi;D;Eq)$ *is a constraint system.*

The rules of Figure 4.2 define the semantics of symbolic processes. This relation transforms a symbolic process into a symbolic process. The aim of this symbolic semantics is to avoid the infinite branching due to the inputs of the environment. This is achieved by keeping variables rather than the input terms. The constraint system gives a finite representation of the value these variables are allowed to take.

$$(\mathcal{E};\{\mathsf{if}\ u=v\ \mathsf{then}\ P_1\ \mathsf{else}\ P_2\}\uplus\mathcal{P};\Phi;D;Eq) \xrightarrow{\tau}_s (\mathcal{E};\{P_1\}\uplus\mathcal{P};\Phi;D;Eq\cup\{u\overset{?}{=}v\}) \quad (\textsc{Then}_s)$$

$$(\mathcal{E};\{\{\mathsf{if}\ u=v\ \mathsf{then}\ P_1\ \mathsf{else}\ P_2\}\uplus\mathcal{P};\Phi;D;Eq) \xrightarrow{\tau}_s (\mathcal{E};\{P_2\}\uplus\mathcal{P};\Phi;D;Eq\cup\{u\overset{?}{\neq}v\}) \quad (\textsc{Else}_s)$$

$$(\mathcal{E};\{\mathsf{out}(p,u).Q_1;\mathsf{in}(q,x).Q_2\}\uplus\mathcal{P};\Phi;D;Eq) \xrightarrow{\tau}_s (\mathcal{E};\{Q_1;Q_2\{x\mapsto u\}\}\uplus\mathcal{P};\Phi;D;Eq\cup\{p\overset{?}{=}q\}) \quad (\textsc{Comm}_s)$$

$$(\mathcal{E};\{P\mid Q\}\uplus\mathcal{P};\Phi;D;Eq) \xrightarrow{\tau}_s (\mathcal{E};\{P;Q\}\uplus\mathcal{P};\Phi;D;Eq) \quad (\textsc{Par}_s)$$

$$(\mathcal{E};\{\mathsf{in}(p,x).Q\}\uplus\mathcal{P};\Phi;D;Eq) \xrightarrow{in(Z,Y)}_s (\mathcal{E};\{Q\{x\mapsto y\}\}\uplus\mathcal{P};\Phi;D\cup\{Y,n\overset{?}{\vdash}y;Z,n\overset{?}{\vdash}z\};Eq\cup\{z\overset{?}{=}p\})$$
if Y,y,Z,z are fresh variables, $n=|\Phi|$ $(\textsc{In}_s)$

$$(\mathcal{E};\{\mathsf{out}(p,u).Q\}\uplus\mathcal{P};\Phi;D;Eq) \xrightarrow{\nu ax_n.out(Z,ax_n)}_s (\mathcal{E};\{Q\}\uplus\mathcal{P};\Phi\cup\{ax_{n+1}\rhd u\};D\cup\{Z,n\overset{?}{\vdash}z\};Eq\cup\{z\overset{?}{=}p\})$$
if $ax_n\in\mathcal{AX}$ such that $n=|\Phi|$, Z,z are fresh variables $(\textsc{Out-T}_s)$

$$(\mathcal{E};\{\mathsf{out}(p,c).Q\}\uplus\mathcal{P};\Phi;D;Eq) \xrightarrow{out(Z,Y)}_s (\mathcal{E};\{Q\}\uplus\mathcal{P};\Phi;D\cup\{Z,n\overset{?}{\vdash}z\ ;Y,n\overset{?}{\vdash}y\};Eq\cup\{z\overset{?}{=}p\ ;y\overset{?}{=}c\})$$
if $c\notin\mathcal{E}$ and Y,y,Z,z are fresh variables, $n=|\Phi|$ $(\textsc{Out-Ch}_s)$

$$(\mathcal{E};\{\mathsf{out}(p,c).Q\}\uplus\mathcal{P};\Phi;D;Eq) \xrightarrow{\nu ch_m.out(Z,ch_m)}_s (\mathcal{E};(\{Q\}\uplus\mathcal{P})\{c\mapsto ch_m\};\Phi;D\cup\{Z,n\overset{?}{\vdash}z\};Eq\cup\{z\overset{?}{=}p\})$$
if $c\in\mathcal{E}$, Z,z are fresh variables with $n=|\Phi|$, ch_m a fresh channel name $(\textsc{Open-Ch}_s)$

u, v are terms having the same type, x is a variable of any type, and p,q,c are terms of channel type, *i.e.* names or variables.

Figure 4.2: Symbolic semantics

The $\textsc{Then}_s$ (resp. $\textsc{Else}_s$) rule allows the process to pass a conditional. The corresponding constraint is added in the set of equations and inequations Eq. When a process is ready to input a term on a public channel p, a deducible constraint is added in the set D (rule $\textsc{In}_s$). When a process is ready to output a term u on a public channel p, the outputted term is added to the

frame Φ (rule Out-T$_s$), which means that this term is made available to the attacker. Note that when this term is actually a channel name, say c, the situation is slightly different. We distinguish two cases depending on whether c is restricted or not. In particular, in case of an output of a bound channel name, the rule Open-Ch$_s$ requires renaming the channel name (as this is done in the rule Out-T$_s$). This is needed for our symbolic trace equivalence relation since we require both the left- and right-hand processes to use the same label without allowing α-conversion.

The relations $\xrightarrow{w}_s$ and $\overset{s}{\Rightarrow}_s$ are defined as for our intermediate semantics.

Example 4.9. *Continuing Examples 4.5, the constraint system* $\mathcal{C}_1 = (\mathcal{E}; \Phi_0 \cup \{ax_4 \rhd t\}; D_1; Eq_1)$ *(see Example 4.6) is one of the constraint systems obtained by applying our symbolic rules on the process* $(\mathcal{E}; \{B'(b,a) \mid A'(a,b)\}; \Phi_0; \emptyset; \emptyset)$ *and considering the following symbolic trace* $\mathsf{tr}_s = in(Z_1, Y).\ \nu ax_4.\ out(Z_2, ax_4)$*. But there is two more possible constraint systems for the same sequence* tr_s*. The first one is* $\mathcal{C}_2 = (\mathcal{E}; \Phi_2; D_2; Eq_2)$ *where:*

- $\Phi_2 = \Phi_0 \cup \{ax_4 \rhd \mathsf{aenc}(n_b, \mathsf{pk}(sk_b))\}$*; and*
- $D_2 = \{Z_1, 3 \overset{?}{\vdash} z_1; Y, 3 \overset{?}{\vdash} y; Z_2, 3 \overset{?}{\vdash} z_2\}$*; and*
- $Eq_2 = \{z_1 \overset{?}{=} c; \mathsf{proj}_2(\mathsf{adec}(y, sk_b)) \overset{?}{\neq} \mathsf{pk}(sk_a); z_2 \overset{?}{=} c\}$.

This constraint system corresponds to the execution of $B'(b,a)$ *but in the* else *branch. The last one is* $\mathcal{C}_3 = (\mathcal{E}; \Phi_3; D_3; Eq_3)$ *where:*

- $\Phi_3 = \Phi_0 \cup \{ax_4 \rhd \mathsf{aenc}(\langle n_a, \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b))\}$*; and*
- $D_3 = \{Z_1, 3 \overset{?}{\vdash} z_1; Y, 3 \overset{?}{\vdash} y; Z_2, 3 \overset{?}{\vdash} z_2\}$*; and*
- $Eq_3 = \{z_1 \overset{?}{=} c; z_2 \overset{?}{=} c\}$.

This last constraint system corresponds to the first input of $B'(b,a)$ *but then the output of* $A'(a,b)$*. This is why there is no restriction on the variable* y *in* $\mathcal{C}_3$*.*

For the same sequence tr_s*, similar constraint systems, denoted* $\mathcal{C}'_1$*,* $\mathcal{C}'_2$ *and* $\mathcal{C}'_3$ *can be derived for the process* $(\mathcal{E}; \{B'(b,a') \mid A'(a',b)\}; \Phi_0)$*. The occurrences of* sk_a *will be replaced by* $sk_{a'}$ *(except the occurrence in* Φ_0*).*

We show that the set of symbolic processes obtained from a symbolic process $(\mathcal{E}; \mathcal{P}; \Phi; \emptyset; \emptyset)$ exactly captures the set of intermediate traces of the intermediate process $(\mathcal{E}; \mathcal{P}; \Phi)$. More specifically, we show that each solution of the symbolic processes corresponds to a trace of the intermediate process (Proposition 4.2) and reciprocally for each trace of the intermediate process there exists a corresponding solution of the symbolic process (Proposition 4.3). Figure 4.3 represents the relations between the intermediate and symbolic semantics.

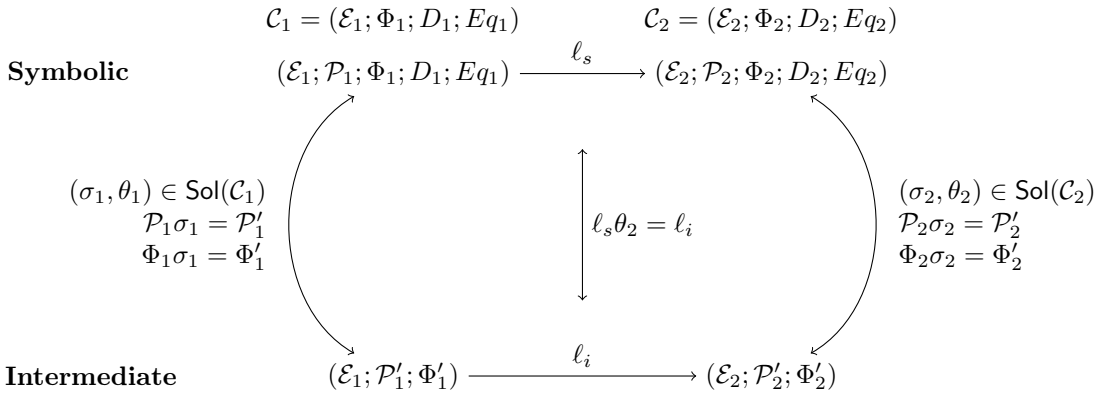


Figure 4.3: Relations between symbolic and intermediate semantics

The soundness of $\xrightarrow{\alpha_s}_s$ w.r.t. $\xrightarrow{\alpha}_i$ is given by the following proposition (proof in Appendix A.2).

Proposition 4.2 (soundness). *Let* $(\mathcal{E}_1;\mathcal{P}_1;\Phi_1;D_1;Eq_1)$*, and* $(\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2)$ *be two symbolic processes such that*

— $(\mathcal{E}_1;\mathcal{P}_1;\Phi_1;D_1;Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2)$*, and*

— $(\sigma_2,\theta_2) \in \mathsf{Sol}((\mathcal{E}_2;\Phi_2;D_2;Eq_2))$.

Let $\theta_1 = \theta_2|_{vars^2(D_1)}$ *and* $\sigma_1 = \sigma_2|_{vars^1(D_1)}$*. We have that:*

1. $(\sigma_1,\theta_1) \in \mathsf{Sol}((\mathcal{E}_1;\Phi_1;D_1;Eq_1))$*, and*
2. $(\mathcal{E}_1;\mathcal{P}_1\sigma_1;\Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2;\mathcal{P}_2\sigma_2;\Phi_2\sigma_2)$

The completeness of the symbolic semantics w.r.t. the intermediate one is given by the following proposition (proof in Appendix A.2).

Proposition 4.3 (completeness). *Let* $(\mathcal{E}_1;\mathcal{P}_1;\Phi_1;D_1;Eq_1)$ *be a symbolic process. Let* $(\sigma_1,\theta_1) \in \mathsf{Sol}((\mathcal{E}_1;\Phi_1;D_1;Eq_1))$*. Let* $(\mathcal{E};\mathcal{P};\Phi)$ *be an intermediate process such that* $(\mathcal{E}_1;\mathcal{P}_1\sigma_1;\Phi_1\sigma_1) \xrightarrow{\alpha}_i (\mathcal{E};\mathcal{P};\Phi)$*. There exist a symbolic process* $(\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2)$*, a pair of substitutions* (σ_2,θ_2)*, and a symbolic action* α_s *such that:*

1. $(\mathcal{E}_1;\mathcal{P}_1;\Phi_1;D_1;Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2)$*;*
2. $(\sigma_2,\theta_2) \in \mathsf{Sol}((\mathcal{E}_2;\Phi_2;D_2;Eq_2))$*;*
3. $(\mathcal{E};\mathcal{P};\Phi) = (\mathcal{E}_2;\mathcal{P}_2\sigma_2;\Phi_2\sigma_2)$*; and*
4. $\alpha_s\theta_2 = \alpha$.

Example 4.10. *Continuing Example 4.9 and 4.6, the constraint system* $\mathcal{C}_1 = (\mathcal{E};\Phi_0 \cup \{ax_4 \rhd t\};D_1;Eq_1)$ *was shown to be obtained by applying our symbolic rules on the process* $(\mathcal{E};\{B'(b,a) \mid A'(a,b)\};\Phi_0;\emptyset;\emptyset)$ *and considering the following symbolic trace* $\mathsf{tr}_s = in(Z_1,Y).\ \nu ax_4.\ out(Z_2,ax_4)$*:*

$$(\mathcal{E};\{B'(b,a) \mid A'(a,b)\};\Phi_0;\emptyset;\emptyset) \overset{\mathsf{tr}_s}{\Rightarrow}_s (\mathcal{E};\{A'(a,b)\};\Phi_0 \cup \{ax_4 \rhd t\};D_1;Eq_1)$$

Furthermore, in Example 4.7, we exhibited a solution of $\mathcal{C}_1$*:* (σ,θ) *where* $\mathrm{dom}(\theta) = \{Z_1,Z_2,Y\}$*,* $Z_1\theta = Z_2\theta = c$*,* $Y\theta = \mathsf{aenc}(\langle n_i, \mathsf{proj}_2(ax_1)\rangle, \mathsf{proj}_2(ax_3))$ *with* n_i *a public name of base type (*i.e. $n_i \notin \mathcal{E}_1$*),* σ *is the substitution whose domain is* $\{z_1,z_2,y\}$ *and such that* $z_1\sigma = z_2\sigma = c$*, and* $y\sigma = \mathsf{aenc}(\langle n_i, \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b))$*. Thus:*

$$(\mathcal{E};\{B'(b,a) \mid A'(a,b)\};\Phi_0) \overset{\mathsf{tr}}{\Rightarrow}_i (\mathcal{E};\{A'(a,b)\};\Phi_0 \cup \{ax_4 \rhd t'\})$$

where $\mathsf{tr} = \mathsf{tr}_s\theta$ *and* $t' = t\sigma$.

Given an intermediate bounded process, there exists only a finite number of symbolic traces (up to renaming of the second variables of the trace). In this section, we will use this property to show how to reduce the problem of deciding the trace equivalence to the problem of deciding the symbolic equivalence of sets of constraint systems.

4.2.3 Symbolic trace equivalence

Due to Proposition 4.1, it is sufficient to establish the link between symbolic equivalence and intermediate trace equivalence of intermediate processes.

Following the approach of [Bau07], we compute from an intermediate process $A = (\mathcal{E};\mathcal{P};\Phi)$ the set of constraint systems capturing the possible executions of A, starting from its associated symbolic process $A_s \overset{def}{=} (\mathcal{E};\mathcal{P};\Phi;\emptyset;\emptyset)$ and applying the rules defined in Figure 4.2. Formally, we define the set of traces of a symbolic process as follows:

$$\mathsf{trace}_s(A_s) = \{(\mathsf{tr}_s,(\mathcal{E}_2;\Phi_2;D_2;Eq_2)) \mid A_s \overset{\mathsf{tr}_s}{\Rightarrow}_s (\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2) \text{ for some } \mathcal{E}_2,\mathcal{P}_2,\Phi_2,D_2,Eq_2\}$$

When tr_s is fixed, we also write $(\mathsf{tr}_s,\Sigma) \in \mathsf{trace}_s(A_s)$ to define the set Σ as the set of constraint systems $\{\mathcal{C} \mid (\mathsf{tr}_s,\mathcal{C}) \in \mathsf{trace}_s(A_s)\}$.

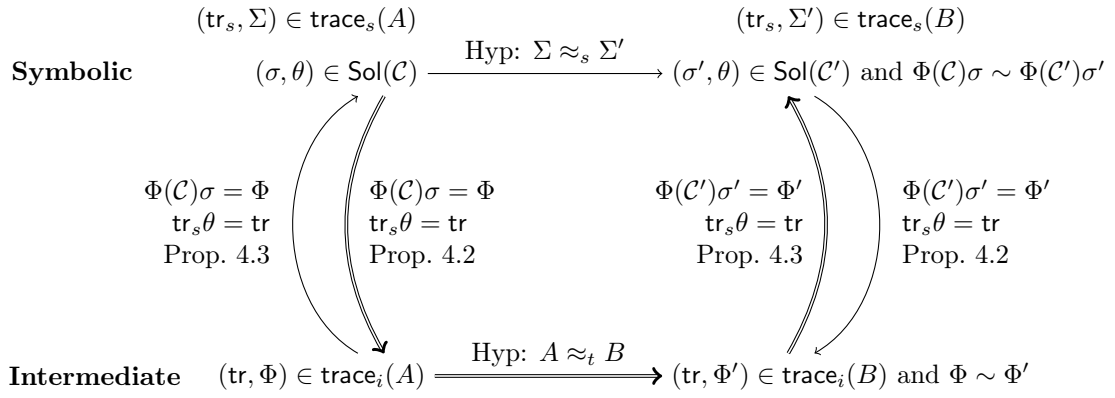
Two intermediate bounded processes are in intermediate trace equivalence if and only if, for any symbolic trace, their corresponding sets of constraints are in symbolic equivalence, which we call symbolic trace equivalence.

Definition 4.7 (symbolic trace equivalence)**.** *Let A and B be two intermediate bounded processes. They are in* symbolic trace equivalence *if for every sequence tr_s of symbolic actions, we have that:*

$$\{\mathcal{C} \mid (\mathsf{tr}_s, \mathcal{C}) \in \mathit{trace}_s(A_s)\} \approx_s \{\mathcal{C} \mid (\mathsf{tr}_s, \mathcal{C}) \in \mathit{trace}_s(B_s)\}$$

where A_s and B_s are the symbolic processes associated to A and B.

By relying of Propositions 4.2 and 4.3, we are now capable of establishing that trace equivalence coincide with symbolic trace equivalence. Figure 4.4 presents a sketch of proof of Proposition 4.4.



The steps of the proof that symbolic trace equivalence implies trace equivalence are represented by the simple arrows, whereas the steps of the proof that trace equivalence implies symbolic trace equivalence are represented by the double arrows.

Figure 4.4: Symbolic trace equivalence and trace equivalence coincide

Proposition 4.4. *Let A and B be two intermediate bounded processes: $A \approx_t B$ if, and only if A and B are in symbolic trace equivalence.*

Proof. We show the two directions separately. Let $A = (\mathcal{E}; \mathcal{P}_A; \Phi_A)$ and $B = (\mathcal{E}; \mathcal{P}_B; \Phi_B)$.

Case ($\Leftarrow$)**:** We have to show that for every $(\mathsf{tr}, \Phi'_A) \in \mathsf{trace}_i(A)$ there exists $(\mathsf{tr}, \Phi'_B) \in \mathsf{trace}_i(B)$ such that $\nu\mathcal{E}.\Phi'_A \sim \nu\mathcal{E}.\Phi'_B$ (and reciprocally). Let $(\mathsf{tr}, \Phi'_A) \in \mathsf{trace}_i(A)$. By definition of $\mathsf{trace}_i(A)$, this means that there exists $\mathcal{P}'_A$ such that $(\mathcal{E}; \mathcal{P}_A; \Phi_A) \overset{\mathsf{tr}}{\Rightarrow}_i (\mathcal{E}; \mathcal{P}'_A; \Phi'_A)$. Let $A_s = (\mathcal{E}; \mathcal{P}_A; \Phi_A; \emptyset; \emptyset)$ and σ, θ be the substitutions identity. We have that $(\sigma, \theta) \in \mathsf{Sol}((\mathcal{E}; \Phi_A; \emptyset; \emptyset))$. Thanks to Proposition 4.3, a simple induction on $|\mathsf{tr}|$ allows us to deduce that there exist a symbolic process $A'_s = (\mathcal{E}; \mathcal{P}'^s_A; \Phi'^s_A; D'_A; Eq'_A)$, a pair of substitution (σ'_A, θ'), and a sequence tr_s of symbolic actions such that:

1. $(\mathcal{E}; \mathcal{P}_A; \Phi_A; \emptyset; \emptyset) \overset{\mathsf{tr}_s}{\Rightarrow}_s (\mathcal{E}; \mathcal{P}'^s_A; \Phi'^s_A; D'_A; Eq'_A)$;
2. $(\sigma_A, \theta') \in \mathsf{Sol}((\mathcal{E}; \Phi'^s_A; D'_A; Eq'_A))$;
3. $(\mathcal{E}; \mathcal{P}'_A; \Phi'_A) = (\mathcal{E}; \mathcal{P}'^s_A\sigma'_A; \Phi'^s_A\sigma'_A)$; and
4. $\mathsf{tr}_s\theta' = \mathsf{tr}$.

Let $\mathcal{C}'_A = (\mathcal{E}; \Phi'^s_A; D'_A; Eq'_A)$. By definition of $\mathsf{trace}_s(A_s)$, we have that $(\mathsf{tr}_s, \mathcal{C}'_A) \in \mathsf{trace}_s(A_s)$. Since A and B are in symbolic trace equivalence, we deduce that there exist a substitution σ'_B and $\mathcal{C}'_B = (\mathcal{E}; \Phi'^s_B; D'_B; Eq'_B)$ such that $(\mathsf{tr}_s, \mathcal{C}'_B) \in \mathsf{trace}_s(B_s)$ with $(\sigma'_B, \theta') \in \mathsf{Sol}(\mathcal{C}'_B)$ and $\nu\mathcal{E}.\Phi'^s_A\sigma'_A \sim \nu\mathcal{E}.\Phi'^s_B\sigma'_B$. By definition of $\mathsf{trace}_s(B_s)$, we have that $(\mathcal{E}; \mathcal{P}_B; \Phi_B; \emptyset; \emptyset) \overset{\mathsf{tr}_s}{\Rightarrow}_s (\mathcal{E}; \mathcal{P}'^s_B; \Phi'^s_B; D'_B; Eq'_B)$ for some $\mathcal{P}'^s_B$. Now, thanks to Proposition 4.2, a simple induction on $|\mathsf{tr}_s|$ allows us to deduce that $(\mathcal{E}; \mathcal{P}_B; \Phi_B) \overset{\mathsf{tr}_s\theta'}{\Rightarrow}_i (\mathcal{E}; \mathcal{P}'^s_B\sigma'_B; \Phi'^s_B\sigma'_B)$. If we denote $\Phi'_B = \Phi'^s_B\sigma'_B$ and since $\mathsf{tr} = \mathsf{tr}_s\theta'$, we clearly have that $(\mathsf{tr}, \nu\mathcal{E}.\Phi'_B) \in \mathsf{trace}_i(B)$. At last, we already showed that $\nu\mathcal{E}.\Phi'^s_A\sigma'_A \sim \nu\mathcal{E}.\Phi'^s_B\sigma'_B$ and so $\nu\mathcal{E}.\Phi'_A \sim \nu\mathcal{E}.\Phi'_B$. Hence the result holds. The other inclusion can be shown in a similar way.

Case ($\Rightarrow$)**:** We have to show that A and B are in symbolic trace equivalence, *i.e.* for every sequence tr_s of symbolic actions,

$$\{\mathcal{C} \mid (\mathsf{tr}, \mathcal{C}) \in \mathsf{trace}_s(A_s)\} \approx_s \{\mathcal{C} \mid (\mathsf{tr}, \mathcal{C}) \in \mathsf{trace}_s(B_s)\}$$

Let $(\mathsf{tr}_s, \mathcal{C}'_A) \in \mathsf{trace}_s(A_s)$ and $(\sigma_A, \theta) \in \mathsf{Sol}(\mathcal{C}'_A)$. By definition of $\mathsf{trace}_s(A_s)$, we know that there exists $\mathcal{P}'^s_A$ such that $A_s \overset{w_s}{\Rightarrow}_s (\mathcal{E}; \mathcal{P}'^s_A; \Phi'^s_A; D'_A; Eq'_A)$ and $\mathcal{C}'_A = (\mathcal{E}; \Phi'^s_A; D'_A; Eq'_A)$.

Thanks to Proposition 4.2, we have that $A \overset{\mathsf{tr}_s\theta}{\Rightarrow}_i (\mathcal{E}; \mathcal{P}'^s_A\sigma_A; \Phi'^s_A\sigma_A)$. Since A and B are in trace equivalence, we deduce that there exists $(\mathcal{E}; \mathcal{P}'_B; \Phi'_B)$ such that $B \overset{\mathsf{tr}_s\theta}{\Rightarrow}_i (\mathcal{E}; \mathcal{P}'_B; \Phi'_B)$ and $\nu\mathcal{E}.\Phi'^s_A\sigma_A \sim \nu\mathcal{E}.\Phi'_B$. Thanks to Proposition 4.3, we deduce that there exist $(\mathcal{E}; \mathcal{P}'^s_B; \Phi'^s_B; D'_B; Eq'_B)$, a pair of substitutions (σ_B, θ'), and a sequence tr'_s of symbolic actions such that:

1. $B_s \overset{\mathsf{tr}'_s}{\Rightarrow}_s (\mathcal{E}; \mathcal{P}'^s_B; \Phi'^s_B; D'_B; Eq'_B)$;
2. $(\sigma_B, \theta') \in \mathsf{Sol}((\mathcal{E}; \Phi'^s_B; D'_B; Eq'_B))$;
3. $(\mathcal{E}; \mathcal{P}'_B; \Phi'_B) = (\mathcal{E}; \mathcal{P}'^s_B\sigma_B; \Phi'^s_B\sigma_B)$; and
4. $\mathsf{tr}'_s\theta' = \mathsf{tr}_s\theta$.

Let $\mathcal{C}'_B = (\mathcal{E}; \Phi'^s_B; D'_B; Eq'_B)$. Actually, we can assume that $w'_s = w_s$ (by renaming the second order variables that occur in w'_s). Moreover, since $w_s\theta' = w_s\theta$, we have that $\theta' = \theta$. Lastly, since $\nu\mathcal{E}.\Phi'^s_A\sigma_A \sim \nu\mathcal{E}.\Phi'_B$ and $\Phi'_B = \Phi'^s_B\sigma_B$, we easily deduce that $\nu\mathcal{E}.\Phi'^s_A\sigma_A \sim \nu\mathcal{E}.\Phi'^s_B\sigma_B$. □

4.3 Main result and conclusion

In this chapter, we have shown that deciding trace equivalence of processes in the applied pi calculus can be reduced to deciding symbolic equivalence of sets of constraint systems. Hence, we extract a decision procedure for trace equivalence.

Theorem 4.1. *Given an algorithm for deciding symbolic equivalence between sets of constraint systems, we can derive an algorithm for deciding trace equivalence between processes without replication.*

Proof. The algorithm for deciding trace equivalence follows from Proposition 4.4: Given a fixed sequence of symbolic labels tr_s, given Σ_A such that $(\mathsf{tr}_s, \Sigma_A) \in \mathsf{trace}_s(A_s)$, for any renaming α of the second order variables of tr_s, we have $(\mathsf{tr}_s\alpha, \Sigma\alpha) \in \mathsf{trace}_s(A_s)$. Hence, when computing the set of constraint systems capturing the possible executions of A, we only consider one sequence of symbolic labels tr_s to represent the set $\{\mathsf{tr}'_s \mid (\mathsf{tr}'_s, \Sigma'_A) \in \mathsf{trace}_s(A_s)$ and tr'_s is equal to tr_s through renaming of second order variables$\}$. Of course the same sequence of symbolic labels tr_s must be used to represent the set $\{\mathsf{tr}'_s \mid (\mathsf{tr}'_s, \Sigma'_B) \in \mathsf{trace}_s(B_s)$ and tr'_s is equal to tr_s through renaming of second order variables$\}$. In such a case, there is only a finite set of traces of a symbolic process to consider, and each symbolic trace leads to a finite number of constraint systems. □

Thanks to this reduction result, we can now focus on deciding the symbolic equivalence between sets of constraint systems. So far, the previous works in the literature focus on symbolic equivalence between two positive constraint systems [Bau07, TD10, CR12] and not between two sets of constraint systems. More details on these three previous works are given in the introduction of Chapter 6. Using these algorithms, [CD09a] shows that the trace equivalence of *simple processes* can be decided by reducing the problem of trace equivalence for simple processes to the symbolic equivalence between two positives constraint systems.

Even though the class of protocol that can be expressed as *simple process* is large, protocols that need non-trivial else branches, private channels or that are not deterministic, such as the *private authentication protocol* and the*e-passport* protocols, cannot be expressed as simple processes. This is why, in Part II, we provide a decision procedure for symbolic equivalence of sets of constraint systems with standard cryptographic primitives and inequalities.

Chapter 5

Composing trace equivalence

Contents

As previously mentioned, most of the existing techniques for analysing protocols with respect to privacy-type properties (*e.g.* [TD10, BAF08]), consider protocols to be executed in isolation. But in reality many applications run in parallel and the underlying protocols may interact in unexpected ways if cryptographic material is shared amongst them. This situation can arise if, for example, a user chooses the same password for two different network services, or a server uses the same key for different protocols. Furthermore, real life protocols are usually complex and composed of several sub-protocols that rely on the same cryptographic material. For example, the *e-passport protocol* (see Section 3.4) is composed of at least three sub-protocols (BAC, AA and PA) that share private keys.

Unfortunately, even if a protocol is secure for an unbounded number of sessions, there is no guarantee if the protocol is executed in an environment where other protocols sharing some common keys are executed. The interaction with the other protocols may dramatically damage the security of the former protocol. This is a well-known fact that has been already observed for trace-based security properties *e.g.* [GT00, CC10], and that remains true for privacy-type properties.

An attacker may take advantage of a protocol Q to break anonymity of another protocol P that has been proved secure in isolation. This can happen for instance if the security of P relies on the secrecy of a particular shared key that is revealed by the protocol Q.

In order to enable verification of complex real life systems, composition theorems for modular reasoning about security and privacy are therefore desirable. They may allow one to deduce security guarantees for a complex protocol, from the security guarantees of the individual sub-protocols. The goal of this chapter is to study the composition of protocols with respect to privacy-type properties.

As mentioned in the introduction, most of the existing results focus on trace properties and few focus on equivalence properties. However, we highlight a result closely related to ours that is the one of S. Ciobaca and V. Cortier [CC10]. Their result holds for any cryptographic primitives that can be modelled using equational theories, and their main result transforms any attack trace of the combined protocol into an attack trace of one of the individual protocols. This allows various ways of combining protocols such as sequentially or in parallel, possibly with inner replications. Although, the major difference with our result is that they consider trace-based security properties, and more precisely secrecy (encoded as a reachability property), the proof technique we present in this chapter is directly inspired from their work.

More specifically, we show that whenever processes P and Q (*resp.* P' and Q') satisfy some disjointness properties (*e.g.* tagged processes, no shared key revealed, ...), we can derive that P and Q running in parallel under the *composition context* $C[\_]$ are equivalent to P' and Q' running in parallel under the *composition context* $C'[\_]$, *i.e.*

$$C[P \mid Q] \;\approx\; C'[P' \mid Q']$$

from the equivalences $C[P] \approx C'[P']$ and $C[Q] \approx C'[Q']$. The composition context under which two processes are composed contains the shared keys possibly under some replications. We illustrate the application of our results on the *e-passport* protocol described in Section 3.4.

5.1 Some difficulties

The secure parallel composition of protocols is an easy task if the protocols do not share any secret key which is represented by the following lemma.

Lemma 5.1. *Let P, Q, P', Q' four processes. $P \approx_t Q$ and $P' \approx_t Q'$ imply $P \mid P' \approx_t Q \mid Q'$.*

This lemma comes directly from the fact the trace equivalence is closed under application of a composition context. This holds in many cryptographic calculus and more specifically in the applied pi calculus.

However, as soon as the protocols share some private keys, some difficulties arises. In particular, the security of the parallel composition can be compromised if one of the protocol revealed a shared key.

Example 5.1. *Consider the equational theory introduced in Example 2.2 and consider the two processes $P_i = \nu r.\mathsf{out}(c, \mathsf{aenc}(\langle r, id_i\rangle, \mathsf{pk}(sk_S)))$ with $i \in \{1, 2\}$. The first component generates a fresh random number r, publishes the message $\mathsf{aenc}(\langle r, id_i\rangle, \mathsf{pk}(sk_S))$ containing its identity id_i by sending it on the public channel c. Note that the encryption of a pairing where one of its component is a fresh nonce is a way to model randomized asymmetric encryption.*

We have that $\nu sk_S.P_1 \approx_t \nu sk_S.P_2$. This equivalence may express for instance the anonymity of the identity id_i.

Assume now that P is executed in an environment where another protocol $Q = \nu sk_S.\,\mathsf{out}(c, sk_S)$ is executed as well and Q uses the same asymmetric key sk_S as the protocol P. Clearly, the equivalence expressing the anonymity of P does not hold anymore:

$$\begin{array}{c} \nu sk_S.\big(\nu r.\mathsf{out}(c, \mathsf{aenc}(\langle r, id_1\rangle, \mathsf{pk}(sk_S))) \mid \mathsf{out}(c, sk_S)\big) \\ \not\approx_t \\ \nu sk_S.\big(\nu r.\mathsf{out}(c, \mathsf{aenc}(\langle r, id_2\rangle, \mathsf{pk}(sk_S))) \mid \mathsf{out}(c, sk_S)\big) \end{array}$$

Indeed, in this example, id_1 and id_2 are both public so can appear in the tests for the static equivalence. Hence a witness for the non-trace equivalence consists of the sequence of actions $\mathsf{tr} = \nu ax_1.out(c, ax_1).\nu ax_2.out(c, ax_2)$ *and the test* $\mathsf{proj}_2(\mathsf{adec}(ax_1, ax_2)) = id_1$

As shown by the example above, the equivalence $P_1 \approx_t P_2$ holds thanks to the secrecy of sk_S, and thus a protocol Q that uses this key sk_S should at least not reveal it. Revealing shared keys would clearly compromise the security of P.

Actually, even if shared keys are not revealed, the interaction of two protocols using common primitives may compromise their security (also true for trace properties such as secrecy).

Example 5.2. *Consider the processes P_i with $i \in \{1, 2\}$ as defined in Example 5.1 and consider the process Q defined as follows:*

$$Q \stackrel{def}{=} \mathsf{in}(c, x).\mathsf{out}(c, \mathsf{adec}(x, sk_S))$$

The equivalence expressing the anonymity of P (for one session) holds. We have that $\nu k.P_1 \approx_t \nu k.P_2$ whereas the equivalence expressing the anonymity of P in presence of Q does not hold anymore. We have that:

$$\nu sk_S.\big(P_1 \mid Q\big) \not\approx_t \nu k.\big(P_2 \mid Q\big)$$

The sequence of actions $\mathsf{tr} = \nu ax_1.out(c, ax_1).\nu ax_2.out(c, ax_2)$ *and the test $ax_2 = id_1$ are a witness of the non-equivalence. Intuitively, instead of revealing the secret key sk_S, Q can be used as an oracle to decrypt a ciphertext that comes from the process P, and thus Q can be used to reveal the identity hidden in the ciphertext.*

5.1.1 Sharing primitives with tagging

In this chapter, we will use the intermediate calculus with its associated semantics. However, given a plain process P, we will sometimes also denote by P the intermediate process $(\emptyset; \{P\}; \emptyset)$.

To avoid a ciphertext from a process to be decrypted by another one, we can consider processes that use disjoint primitives. However, this is an unnecessarily restrictive condition. So, we consider protocols that may share some cryptographic primitives provided they are *tagged*. Tagging is a syntactic transformation that consists in assigning to each protocol an identifier (*e.g.* the protocol's name) that should appear in any encrypted message. Many relevant equational theories are not so easy to tag (*e.g.* exclusive or). So, we only consider a fix common signature with its associated equational theory, denoted $(\mathcal{F}_0, \mathsf{E}_0)$, and we explain how to transform any process built on a signature $\mathcal{F}$ (possibly larger that $\mathcal{F}_0$) into a tagged process.

The common signature $\mathcal{F}_0$ is defined as follows:

$$\{\mathsf{sdec}/2, \mathsf{senc}/2, \mathsf{adec}/2, \mathsf{aenc}/2, \mathsf{pk}/1, \langle\,\rangle/2, \mathsf{proj}_1/1, \mathsf{proj}_2/1, \mathsf{sign}/2, \mathsf{check}/2, \mathsf{vk}/1, \mathsf{h}/1\}$$

The common equational theory E_0 is defined by the following equations ($i \in \{1, 2\}$):

$$\begin{array}{rr} \mathsf{sdec}(\mathsf{senc}(x, y), y) = x & \mathsf{adec}(\mathsf{aenc}(x, \mathsf{pk}(y)), y) = x \\ \mathsf{proj}_i(\langle x_1, x_2\rangle) = x_i & \mathsf{check}(\mathsf{sign}(x, y), \mathsf{vk}(y)) = x \end{array}$$

For our composition results, we will assume that the processes we want to compose are built on $(\mathcal{F}_a \cup \mathcal{F}_0, \mathsf{E}_a \cup \mathsf{E}_0)$ and $(\mathcal{F}_b \cup \mathcal{F}_0, \mathsf{E}_b \cup \mathsf{E}_0)$, where $(\mathcal{F}_a, \mathsf{E}_a)$, $(\mathcal{F}_b, \mathsf{E}_b)$ and $(\mathcal{F}_0, \mathsf{E}_0)$ are disjoint signatures.

5.1.1.1 Tagging a term

We define the signature $\mathcal{F}_{\mathsf{tag}_c} = \{\mathsf{tag}_c/1, \mathsf{untag}_c/1\}$ where tag_c and untag_c are two function symbols that we will use for tagging. The role of the tag_c function is to tag its argument with the tag c. The role of the untag_c function is to remove the tag. To model this interaction between tag_c and untag_c, we consider the equational theory:

$$\mathsf{E}_{\mathsf{tag}_c} = \{\mathsf{untag}_c(\mathsf{tag}_c(x)) = x\}.$$

Intuitively, tagging a protocol P_A, built on $(\mathcal{F}_a \cup \mathcal{F}_0, \mathsf{E}_a \cup \mathsf{E}_0)$, with the tag a will result a process built on $(\mathcal{F}_a \cup \mathcal{F}_{\mathsf{tag}_a} \cup \mathcal{F}_0, \mathsf{E}_a \cup \mathsf{E}_{\mathsf{tag}_a} \cup \mathsf{E}_0)$. We denote by $\mathcal{F}_c^+ = \mathcal{F}_c \cup \mathcal{F}_{\mathsf{tag}_c}$ and $\mathsf{E}_c^+ = \mathsf{E}_c \cup \mathsf{E}_{\mathsf{tag}_c}$ with $c \in \{a, b\}$ and a, b are the two tags that we will use in this chapter.

Definition 5.1. *Let u be a term built on $\mathcal{F}_c \cup \mathcal{F}_0$ ($c \in \{a, b\}$). The c-tagged version of u, denoted $[u]_c$ is defined as follows:*

$$\begin{array}{llll} [\mathsf{senc}(u,v)]_c \stackrel{def}{=} \mathsf{senc}(\mathsf{tag}_c([u]_c), [v]_c) & & [\mathsf{sdec}(u,v)]_c \stackrel{def}{=} \mathsf{untag}_c(\mathsf{sdec}([u]_c, [v]_c)) \\ [\mathsf{aenc}(u,v)]_c \stackrel{def}{=} \mathsf{aenc}(\mathsf{tag}_c([u]_c), [v]_c) & & [\mathsf{adec}(u,v)]_c \stackrel{def}{=} \mathsf{untag}_c(\mathsf{adec}([u]_c, [v]_c)) \\ [\mathsf{sign}(u,v)]_c \stackrel{def}{=} \mathsf{sign}(\mathsf{tag}_c([u]_c), [v]_c) & & [\mathsf{check}(u,v)]_c \stackrel{def}{=} \mathsf{untag}_c(\mathsf{check}([u]_c, [v]_c)) \end{array}$$

$$\begin{array}{rcl} [\mathsf{h}(u)]_c & \stackrel{def}{=} & \mathsf{h}(\mathsf{tag}_c([u]_c)) \\ [u]_c & \stackrel{def}{=} & u \quad \textit{when } u \in \mathcal{X} \cup \mathcal{N} \\ [\mathsf{f}(u_1, \ldots, u_n)]_c & \stackrel{def}{=} & \mathsf{f}([u_1]_c, \ldots, [u_n]_c) \quad \textit{otherwise.} \end{array}$$

Note that we do not tag the pairing function symbol (this is actually useless), and we do not tag the pk and vk function symbols. Actually, tagging pk and vk would greatly help us to establish our results and would also avoid us to introduce some additional assumptions, but this would lead us to consider an unrealistic modeling for asymmetric keys. Some of the difficulties encountered with asymmetric keys will be discussed in Section 5.4.

Example 5.3. *Consider $u_i = \mathsf{aenc}(\langle r, id_i\rangle, \mathsf{pk}(sk_S))$ with $i \in \{1, 2\}$ and $v = \mathsf{proj}_2(\mathsf{adec}(x, sk_S))$. We have that $[u_i]_a = \mathsf{aenc}(\mathsf{tag}_a(\langle r, id_i\rangle), \mathsf{pk}(sk_S))$, whereas $[v]_b = \mathsf{proj}_2(\mathsf{untag}_b(\mathsf{adec}(x, sk_S)))$.*

5.1.1.2 Tagging a process

Before extending the notion of tagging to processes, we have to express the tests that are performed by an agent when he receives a message that is supposed to be tagged. Indeed, the main purpose of tagging P_A and P_B with two different tags is to ensure that the intruder cannot reinject a message obtained from P_A (resp. P_B) into the process P_B (resp. P_A). Hence to do so, the tagged version of P_A and P_B will check that each message they receive are properly tagged with the tags a and b respectively. This is the purpose of $\mathsf{test}_c(u)$ that represents the tests which ensure that every projection and every untagging performed by an agent during the computation of u is successful.

Definition 5.2. *Let u be a term built on $\mathcal{F}_c^+ \cup \mathcal{F}_0$ with $c \in \{a, b\}$. We define $\mathsf{test}_c(u)$ as follows:*

- $\mathsf{test}_c(u) \stackrel{def}{=} \mathsf{test}_c(u_1) \wedge \mathsf{test}_c(u_2) \wedge \mathsf{tag}_c(\mathsf{untag}_c(u)) = u$, *if $u = \mathsf{g}(u_1, u_2)$ with $\mathsf{g} \in \{\mathsf{sdec}, \mathsf{adec}, \mathsf{check}\}$;*
- $\mathsf{test}_c(u) \stackrel{def}{=} \mathsf{test}_c(u_1) \wedge u_1 = \langle \mathsf{proj}_1(u_1), \mathsf{proj}_2(u_1)\rangle$, *if $u = \mathsf{proj}_i(u_1)$ with $i \in \{1, 2\}$;*
- $\mathsf{test}_c(u) \stackrel{def}{=} \mathsf{true}$, *if u is a name or a variable;*
- *Otherwise $u = \mathsf{f}(u_1, \ldots, u_n)$ for some n, and* $\mathsf{test}_c(u) \stackrel{def}{=} \mathsf{test}_c(u_1) \wedge \ldots \wedge \mathsf{test}_c(u_n)$.

Note that in the first case, *e.g.* $u = \mathsf{sdec}(u_1, u_2)$, if u is a ground term then we have $\mathsf{tag}_c(\mathsf{untag}_c(u)) =_{\mathsf{E}} u$ if and only if the decryption succeeds and the plain text is tagged with the tag c, *i.e.* $u_1 =_{\mathsf{E}} \mathsf{senc}(\mathsf{tag}_c(u_3), u_2)$ for some u_3.

For any term u, $\mathsf{test}_c(u)$ is a conjunction of elementary formulas (equalities between term). Hence, given a ground substitution α such that $\mathit{fvars}(u) \subseteq \mathrm{dom}(\alpha)$, we say that α satisfies $\mathsf{test}_c(u)$, denoted $\alpha \vDash \mathsf{test}_c(u)$ if α satisfies all elementary formulas of $\mathsf{test}_c(u)$ whereas α *satisfies* $t_1 = t_2$, denoted $\alpha \vDash t_1 = t_2$, if $t_1\alpha{\downarrow} = t_2\alpha{\downarrow}$.

Example 5.4. *Again, consider the terms $u_i = \mathsf{aenc}(\langle r, id_i\rangle, \mathsf{pk}(sk_S))$ with $i \in \{1, 2\}$ and $v = \mathsf{proj}_2(\mathsf{adec}(x, sk_S))$. We have that:*

— $[u_i]_a = \mathsf{aenc}(\mathsf{tag}_a(\langle r, id_i\rangle), \mathsf{pk}(sk_S))$ *and* $\mathsf{test}_a([u_i]_a) = \mathsf{true}$.

— $[v]_b = \mathsf{proj}_2(\mathsf{untag}_b(\mathsf{adec}(x, sk_S)))$ *and* $\mathsf{test}_b([v]_b)$ *is the following conjunction:*

$$\mathsf{tag}_b(\mathsf{untag}_b(\mathsf{adec}(x, sk_S))) = \mathsf{adec}(x, sk_S) \wedge \langle \mathsf{proj}_1(v'), \mathsf{proj}_2(v')\rangle = v'$$

where $v' = \mathsf{untag}_b(\mathsf{adec}(x, sk_S))$.

Let $\alpha_1 = \{x \mapsto n\}$, $\alpha_2 = \{x \mapsto \mathsf{aenc}(\mathsf{tag}_b(n), sk_S)\}$ *and* $\alpha_3 = \{x \mapsto \mathsf{aenc}(\mathsf{tag}_b(\langle n, n'\rangle), sk_S)\}$. *We have that* $\alpha_1 \not\vdash \mathsf{test}_b([v]_b)$ *and* $\alpha_2 \not\vdash \mathsf{test}_b([v]_b)$ *but* $\alpha_3 \vdash \mathsf{test}_b([v]_b)$.

We can now explain how to tag an intermediate process. Let $A = (\mathcal{E}; \mathcal{P}; \Phi)$ be an intermediate process built on $\mathcal{F}_c \cup \mathcal{F}_0$ with $c \in \{a, b\}$ such that $\mathcal{P} = \{P_1, \ldots, P_\ell\}$, and $\Phi = \{ax_1 \rhd u_1, \ldots, ax_n \rhd u_n\}$. The c-tagged version of the process A, denoted $[A]_c$, is the process $(\mathcal{E}; [\mathcal{P}]_c; [\Phi]_c)$ where $[\mathcal{P}]_c = \{[P_1]_c, \ldots, [P_\ell]_c\}$, and $[\Phi]_c = \{ax_1 \rhd [u_1]_c, \ldots, ax_n \rhd [u_n]_c\}$.

For plain processes, the transformation $[P]_c$ is defined as follows:

$$[0]_c \stackrel{def}{=} 0 \qquad [!P]_c \stackrel{def}{=} ![P]_c \qquad [P \mid Q]_c \stackrel{def}{=} [P]_c \mid [Q]_c$$
$$[\nu k.P]_c \stackrel{def}{=} \nu k.[P]_c \qquad [\mathsf{in}(u, x).P]_c \stackrel{def}{=} \mathsf{in}(u, x).[P]_c$$

$$[\mathsf{out}(u, v).Q]_c \stackrel{def}{=} \mathsf{if}\ \mathsf{test}_c([v]_c)\ \mathsf{then}\ \mathsf{out}(u, [v]_c).[Q]_c\ \mathsf{else}\ 0$$
$$[\mathsf{if}\ u_1 = u_2\ \mathsf{then}\ P\ \mathsf{else}\ Q]_c \stackrel{def}{=} \mathsf{if}\ \mathsf{test}_c([u_1]_c) \wedge \mathsf{test}_c([u_2]_c)\ \mathsf{then}\ \mathsf{if}\ [u_1]_c = [u_2]_c\ \mathsf{then}\ [P]_c\ \mathsf{else}\ [Q]_c\ \mathsf{else}\ 0$$

The logical connector $\wedge$ that are used in $\mathsf{test}_c(u)$ are syntactic sugar that can be easily encoded using nested conditionals.

Roughly, instead of simply outputting a term v, a process will first performed some tests to check that the term is correctly tagged, thanks to $\mathsf{test}_c([v]_c)$, and he will output its c-tagged version $[v]_c$. For a conditional, the process will first check that the terms u_1 and u_2 are correctly tagged before checking that the test is satisfied.

Note that $\mathsf{test}_c()$ is always applied on a term that is tagged thanks to $[\,]_c$. It may seems counter intuitive since one the purpose of $\mathsf{test}_c(u)$ is to verify that u is well tagged whereas the purpose of $[u]_c$ is to tag the term u. Indeed, for a ground term u that do not contain the symbol functions $\mathsf{sdec}, \mathsf{adec}, \mathsf{proj}_i$ or check, we have that $\mathsf{test}_c([u]_c) = \mathsf{true}$. However, since we tag an intermediate process, the term on which we apply $[\,]_c$ are not necessary ground. Furthermore, the value of their variable may differ in each derivation of the intermediate process. Hence, the tests $\mathsf{test}_c([u]_c)$ will ensure that for any derivation of the processes, the instantiation of $[u]_c$ is properly tagged.

Example 5.5. *We consider the processes P and Q defined in Example 5.2. Assume that P and Q are tagged with respectively the tags a and b. Hence we have:*

$$[Q]_b = \mathsf{in}(c, x).\mathsf{if}\ \mathsf{tag}_b(\mathsf{untag}_b(\mathsf{adec}(x, sk_S))) = \mathsf{adec}(x, sk_S)\ \mathsf{then}\ \mathsf{out}(c, \mathsf{untag}_b(\mathsf{adec}(x, sk_S)))\ \mathsf{else}\ 0$$

Note that $\mathsf{test}_b(\mathsf{untag}_b(\mathsf{adec}(x, sk_S))) \stackrel{def}{=} \mathsf{tag}_b(\mathsf{untag}_b(\mathsf{adec}(x, sk_S))) = \mathsf{adec}(x, sk_S)$.

Consider now the intermediate process $A = (\emptyset, [Q]_b, \emptyset)$ and $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathit{trace}_i(A)$ such that $\mathsf{tr} = in(c, M).\nu ax_1.out(c, ax_1)$ for some M. The presence of the conditional in $[Q]_b$ ensure that $M =_{\mathsf{E}} \mathsf{aenc}(\mathsf{tag}_b(u), \mathsf{pk}(sk_S))$ for some u. This is consistent with the goal of Q which is to reveal any plain text that is encrypted with $\mathsf{pk}(sk_S)$.

Note that the test only verifies the tags of the part of M on which the process $[Q]_b$ will apply some function symbols. In this example, even if u is tagged with the tag a or not tagged at all, M will satisfy the test. Intuitively, the intruder may have built u from scratch or even receive it from the process $[P]_a$ but from the point of view of Q, u is only some bitstring that is stored as-is.

5.1.2 Composition context

We saw in Example 5.1 that parallel composition of processes under name restriction may reveal itself to be problematic. However, for the purpose of this work, i.e. parallel composition of processes for security property such as anonymity and unlinkability, composition under name restriction is not enough.

Example 5.6. *Coming back to the* e-passport *protocol (Section 3.4), the anonymity of the parallel composition of the Active Authentication protocol (AA) and the Passive Authentication protocol (PA) can be expressed with the following equivalence:*

$$\nu sk_{DS}.C[PA \mid AA, PA_0 \mid AA_0] \approx_t \nu sk_{DS}.C'[PA \mid AA]$$

where C and C' are contexts defined as follows:

$$C[\__1, \__2] \stackrel{def}{=} \; !\, \nu sk_P.\, \nu id.\, \nu sig.\, \nu pic.\; \ldots \\ !\, \nu ksenc.\, \nu ksmac.\; \__1 \\ \mid\; \nu sk_{P0}.\, !\nu ksenc.\, \nu ksmac.\; \__2$$

$$C'[\_] \stackrel{def}{=} !\nu sk_P.\, \nu id.\, \nu sig.\, \nu pic.\; \ldots !\, \nu ksenc.\, \nu ksmac.\; \_$$

However, if we now look at the protocols AA and PA in isolation, the anonymity of theses protocols would be expressed with the two following equivalences:

$$\nu sk_{DS}.C[AA, AA_0] \approx_t \nu sk_{DS}.C'[AA]$$

$$\nu sk_{DS}.C[PA, PA_0] \approx_t \nu sk_{DS}.C'[PA]$$

One can note that in the previous example, the parallel composition is done under contexts which are different depending on the side of the equivalence. However, note that each time, the contexts we consider for expressing anonymity are only composed of replication, parallel composition and name restriction.

Furthermore, note that in Example 5.6, the restriction of some shared keys occurs under a replication, *e.g. ksenc*, *ksmac*. Hence, considering composition in a simpler setting where only a bounded number of keys $\tilde{k}$ are shared (as done in *e.g.* [CD09b]), would not allow us to establish unlinkability or anonymity in a modular way, but only some results of the form:

$$\nu\tilde{k}.P_1 \approx_t \nu\tilde{k}.P_2 \Rightarrow \nu\tilde{k}.\,(P_1 \mid Q) \approx_t \nu\tilde{k}.\,(P_2 \mid Q)$$

assuming that processes P_1, P_2, and Q satisfy some additional conditions.

Thus, we introduce the notion of *composition context* that will help us to describe under which keys the composition has to be done. Note that a composition context may contain several holes, parallel operators, and nested replications.

Definition 5.3. *A* composition context *C is defined by the following grammar where n is a name of base type.*

$$C, C_1, C_2 := \_ \;\mid\; \nu n.C \;\mid !C \;\mid\; C_1|C_2$$

We only allow names of base type (typically keys) to be shared between processes through the composition context. In particular, they are not allowed to share a private channel even if each process can use its own private channels to communicate internally. We also suppose w.l.o.g. that names occurring in C are distinct. A composition context may contain several holes. We can index them to avoid confusion. We write $C[P_1, \ldots, P_\ell]$ (or shortly $C[\overline{P}]$) the process obtained by filling the i^{th} hole with the process P_i (or the i^{th} process of the sequence $\overline{P}$). We will also use $\overline{P} \mid \overline{Q}$ to represent the sequence of processes obtained by putting in parallel the processes of the sequences of same size $\overline{P}$ and $\overline{Q}$ componentwise, *i.e.* the sequence $P_1 \mid Q_1, \ldots, P_\ell \mid Q_\ell$.

In Example 5.1, we saw that revealing a shared key is problematic for composition. Since some shared key may be restricted under a replication, we have to consider renaming and formalise the notion of revealing a shared key. Note that all names that occur in a composition context do not necessary represent shared keys.

Example 5.7. *Coming back to Example 5.6 and Figure 3.3 and 3.4, the names sk_{DS}, id, sig, pic, dg_1, ..., dg_{19} are only used in the Passive Authentication protocol and so are not shared keys. On the other hand, the secret key sk_P, associated to each passport, is used in both Active and Passive Authentication protocols ($\mathsf{vk}(sk_P)$ is stored in dg_{15}). Moreover, the session keys ksenc and ksmac are also used in both Active and Passive Authentication protocols and so ksenc, ksmac and sk_P are the only shared keys in the composition contexts $\nu sk_{DS}.C$ and $\nu sk_{DS}.C'$.*

Definition 5.4. *Let A be an intermediate process of the form $(\mathcal{E}; C[P_1, \ldots, P_\ell]; \Phi)$ where C is a composition context, and let key $\in \{n, \mathsf{pk}(n), \mathsf{vk}(n) \mid n \in \mathcal{E}$ or n occurs in $C\}$. We say that* the intermediate process A reveals the shared key *key when:*

Either $fnames(key) \in \mathcal{E}$, and

- *$A \stackrel{w}{\Rightarrow}_i (\mathcal{E}'; \mathcal{P}'; \Phi')$ for some $(\mathcal{E}'; \mathcal{P}'; \Phi')$; and*
- *$M\Phi' =_{\mathsf{E}} key$ for some M such that $fvars(M) \subseteq \mathrm{dom}(\Phi')$ and $fnames(M) \cap \mathcal{E}' = \emptyset$.*

Or, we have that $fnames(key)$ occurs in C, the $i_0{}^{th}$ hole is in the scope of $\nu fnames(key)$, and

- *$(\mathcal{E} \cup \{s\}; C[P_1^+, \ldots, P_\ell^+]; \Phi) \stackrel{w}{\Rightarrow}_i (\mathcal{E}'; \mathcal{P}'; \Phi')$ with $P_i^+ \stackrel{def}{=} P_i$ if $i \neq i_0$ and $P_{i_0}^+ \stackrel{def}{=} P_{i_0} \mid \mathsf{in}(c, x).\mathsf{if}\ x = key\ \mathsf{then}\ \mathsf{out}(c, s)\ \mathsf{else}\ 0$; and*
- *$M\Phi' =_{\mathsf{E}} s$ for some M such that $fvars(M) \subseteq \mathrm{dom}(\Phi')$ and $fnames(M) \cap \mathcal{E}' = \emptyset$.*

where s and c are fresh.

Intuitively, a key is revealed if there exists a trace of the extended process where this key can be deduced by the intruder. Hence, in the case where the shared key is in the set of private names $\mathcal{E}$, the definition is straight forward. However, in the case where the shared key appears in the composition context, due to the scope in which the key *key* is restricted, we say that *key* is revealed if the intruder is able to deduce the fresh private name s which is possible if and only if the intruder is able to output *key* on the fresh public channel c.

Example 5.8. *Consider the composition context $C[\_] = !\nu sk_S.\_$. Consider the process P_1 described in Example 5.1. The intermediate process $(\emptyset; C[P_1]; \emptyset)$ does not reveal the keys sk_S, $\mathsf{pk}(sk_S)$ and $\mathsf{vk}(sk_S)$. Indeed, let key $\in \{sk_S, \mathsf{pk}(sk_S), \mathsf{vk}(sk_S)\}$, the intermediate process*

$$(\{s\}; C[P_1 \mid \mathsf{in}(c, x).\mathsf{if}\ x = key\ \mathsf{then}\ \mathsf{out}(c, s)\ \mathsf{else}\ 0]; \emptyset)$$

can not reach a configuration from which s will be derivable by the attacker.

5.2 Preliminaries

As mentioned at the beginning of this chapter, our proof technique is similar to the one of [CC10] and consists of transforming any trace of the combined processes that share secret keys into a trace of a composition of the processes where no secret key is shared. In this section, we will introduce the main notions that will be useful to define and prove such transformation.

5.2.1 Material for composition

As mentioned in Section 5.1, we consider in this chapter several signatures with their associated equational theories. We have $(\mathcal{F}_a, \mathsf{E}_a)$ and $(\mathcal{F}_b, \mathsf{E}_b)$ that are not specified, $(\mathcal{F}_0, \mathsf{E}_0)$ the common signature, and the two signatures that we use for tagging processes: $(\mathcal{F}_{\mathsf{tag}_a}, \mathsf{E}_{\mathsf{tag}_a})$ and $(\mathcal{F}_{\mathsf{tag}_b}, \mathsf{E}_{\mathsf{tag}_b})$. Moreover, we know that all those signatures and equational theory are disjoint two at a time and consistent. Since a message sent over the network can be formed with a combination of all these

signatures, we introduce several notions that will help us describing a term. We define $\mathcal{F}$ and E the union of the previous signatures and equational theories, i.e. $\mathcal{F} = \mathcal{F}_a \cup \mathcal{F}_b \cup \mathcal{F}_0 \cup \mathcal{F}_{\mathsf{tag}_a} \cup \mathcal{F}_{\mathsf{tag}_b}$ and $\mathsf{E} = \mathsf{E}_a \cup \mathsf{E}_b \cup \mathsf{E}_0 \cup \mathsf{E}_{\mathsf{tag}_a} \cup \mathsf{E}_{\mathsf{tag}_b}$.

5.2.1.1 Factors

The term N is *alien* to M if $\mathsf{root}(N) \in \mathcal{F}_i$, $\mathsf{root}(M) \in \mathcal{F}_j$ and $i \neq j$. We now introduce the notion of *factors*. A similar notion is also used in [CR05].

Definition 5.5 (factors)**.** *Let $M \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X})$. The* factors *of M, denoted $Fct(M)$, are the maximal syntactic subterms of M that are alien to M*

The factors will help us determine which parts of a term might have been computed by the processes P_A and P_B that we want to compose. Indeed, if the root of a term comes from the signature $\mathcal{F}_a$ and the root of one of its factors comes from $\mathcal{F}_b$ then it's likely that this factor was computed by P_B and then injected into the process P_A. However, its is not necessary the case since P_A and P_B share the signature $\mathcal{F}_0$.

Example 5.9. *Assume that $\mathcal{F}_a$ is the signature made up of the constant symbol 0 and the binary function $+$ and that the associated equation theory E_a is the following set of equations:*

$$\begin{array}{rclcrcl} x + (y + z) & = & (x + y) + z & \qquad & x + 0 & = & x \\ x + y & = & y + x & & x + x & = & 0 \end{array}$$

Let u be a term built upon $(\mathcal{F}_a \cup \mathcal{F}_0, \mathsf{E}_a \cup \mathsf{E}_0)$ such that $u = \mathsf{sdec}(\langle n_1 + \langle n_2, n_3\rangle, \mathsf{proj}_1(n_1 + n_2)\rangle, n_3)$. The term $n_1 + \langle n_2, n_3\rangle$ is a syntactic subterm of u alien to u since $\mathsf{root}(n_1 + \langle n_2, n_3\rangle) \in \mathcal{F}_+$ and $\mathsf{root}(u) \in \mathcal{F}_0$. We have that

$$Fct(u) = \{n_1 + \langle n_2, n_3\rangle,\ n_1 + n_2,\ n_3\}$$

For our composition result, we work with tagged processes, i.e. $[P_A]_a$ and $[P_B]_b$. As described in Section 5.1, theses processes check the tags of a term before sending it over the network. While $\mathsf{senc}(\mathsf{tag}_a(n), k)$ might have been computed by the process P_A, the term $\mathsf{senc}(n, k)$ could not have been computed by P_A since it is not tagged with the tag a and was most likely computed by the intruder. Hence for terms whose root is in the common signature, we introduce a specific notion of factors that takes into account terms.

Definition 5.6 (Factor for $\mathcal{F}_0$)**.** *Let u be a term. We define $Fct_{\mathcal{F}_0}(u)$ the factors of a term u for $\mathcal{F}_0$ as the maximal syntactic subterms of u of the form $\mathsf{f}(\mathsf{tag}_i(u_1), u_2)$ with $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$ and $i \in \{a, b\}$; or of the form $\mathsf{h}(\mathsf{tag}_i(u_1))$ with $i \in \{a, b\}$; or whose root symbol is not in $\mathcal{F}_0$.*

Example 5.10. *Consider the theory $(\mathcal{F}_a, \mathcal{F}_a)$ given in Example 5.9. Let u be a term built on $(\mathcal{F}_a \cup \mathcal{F}_{\mathsf{tag}_a} \cup \mathcal{F}_0), \mathsf{E}_a \cup \mathsf{E}_{\mathsf{tag}_a} \cup \mathsf{E}_0)$ such that $u = \langle \mathsf{senc}(\mathsf{tag}_a(n_1), \langle n_1 + n_2, n_3\rangle), \mathsf{sign}(n_4, n_5)\rangle$. We have:*

- $Fct(u) = \{\mathsf{tag}_a(n_1);\ n_1 + n_2;\ n_3;\ n_4;\ n_5\}$
- $Fct_{\mathcal{F}_0}(u) = \{\mathsf{senc}(\mathsf{tag}_a(n_1), \langle n_1 + n_2, n_3\rangle);\ n_4;\ n_5\}$
- $Fct_{\mathcal{F}_0}(n_1 + n_2) = \{n_1 + n_2\}$

We can see in this example that since the root of u is in $\mathcal{F}_0$, $Fct_{\mathcal{F}_0}(u)$ gives more informations than $Fct(u)$. Indeed, $\mathsf{senc}(\mathsf{tag}_a(n_1), \langle n_1 + n_2, n_3\rangle)$ could have been computed by $[P_A]_a$ whereas $\mathsf{sign}(n_4, n_5)$ could not have been computed by $[P_A]_a$ or $[P_B]_b$. Hence, only n_4 and n_5 might have been computed by our processes.

5.2.1.2 Ordered rewriting

Most of the definitions and results in this subsection are borrowed from [CR05] and [CD12] since we use similar techniques. We consider the notion of *ordered rewriting* defined in [DJ90], which is a useful tool that has been used (*e.g.* [BS96]) for proving correctness of combination of unification algorithms. It will allows us to use rewriting systems rather than equational theories which are sometimes difficult to handle in proofs.

Let $\prec$ be a simplification ordering on ground terms , i.e. $\prec$ satisfies that for all ground terms u, v_1, v_2, and for any position $p \neq \epsilon$ in m, $v_1 \prec u[v_1]_p$ and $v_1 \prec v_2$ implies $u[v_1]_p \prec u[v_2]_p$. Assume furthermore that $\prec$ is total, the minimum for $\prec$ is a name n_{min} and the constants in $\mathcal{F}$ are smaller that any ground term that is neither a constant nor a name. We define $\mathcal{F}^+$ to be the set of the function symbols of $\mathcal{F}$ plus the name n_{min}, *i.e.* $\mathcal{F}^+ = \mathcal{F} \cup \{n_{min}\}$. In what follows, we furthermore assume that n_{min} is never used under restriction in frames.

Given a possibly infinite set of equations $\mathcal{O}$, we define the ordered rewriting relation $\to_{\mathcal{O}}$ by $t \to_{\mathcal{O}} t'$ if and only if there exist an equation $u_1 = u_2 \in \mathcal{O}$, a position p in M and a substitution τ such that:

$$t = t[u_1\tau]_p, \;\; t' = t[u_2\tau]_p \text{ and } u_2\tau \prec u_1\tau$$

It has been shown (see [DJ90]) that by applying the *unfailing completion procedure* to a set of equations E we can derive a (possibly infinite) set of equations $\mathcal{O}$ such that on ground terms:

1. the relations $=_{\mathcal{O}}$ and $=_{\mathsf{E}}$ are equal,
2. the rewriting system $\to_{\mathcal{O}}$ is convergent.

It was showed ([BS96]) that applying unfailing completion to two disjoints sets of equations, *e.g.* $\mathsf{E}_a \cup \mathsf{E}_b$, yields the set of generated equations $\mathcal{O}$ that is the disjoint union of the two systems $\mathcal{O}_a$ and $\mathcal{O}_b$ obtained by applying unfailing completion procedures to E_a and to E_b respectively. We can easily extend this result to $\mathsf{E} = \mathsf{E}_a \cup \mathsf{E}_b \cup \mathsf{E}_0 \cup \mathsf{E}_{\mathsf{tag}_a} \cup \mathsf{E}_{\mathsf{tag}_b}$ since they are all disjoint two at a time.

Thus, applying unfailing completion to E yields the set of generated equations $\mathcal{O}$ that is the disjoint union of $\mathcal{O}_a, \mathcal{O}_b, \ldots, \mathcal{O}_{\mathsf{tag}_b}$ obtained by applying unfailing completion procedures respectively to $\mathsf{E}_a, \mathsf{E}_b, \ldots, \mathsf{E}_{\mathsf{tag}_b}$. In fact, applying the unfailing procedure (see [DJ90]) on E_0 (resp. $\mathsf{E}_{\mathsf{tag}_a}$ and $\mathsf{E}_{\mathsf{tag}_b}$) yields $\mathcal{O}_0 = \mathsf{E}_0$ (resp. $\mathcal{O}_{\mathsf{tag}_a} = \mathsf{E}_{\mathsf{tag}_a}$ and $\mathcal{O}_{\mathsf{tag}_b} = \mathsf{E}_{\mathsf{tag}_b}$). Typically, it is due to the fact that each equation has a variable as right hand side, which is subterm of the left hand side of the equation. Therefore, the rewriting systems $\to_{\mathcal{O}_0}$, $\to_{\mathcal{O}_{\mathsf{tag}_a}}$ and $\to_{\mathcal{O}_{\mathsf{tag}_b}}$ are as follows:

$$\begin{array}{ll} \to_{\mathcal{O}_{\mathsf{tag}_a}}: & \mathsf{untag}_a(\mathsf{tag}_a(x)) \to x \\ \to_{\mathcal{O}_{\mathsf{tag}_b}}: & \mathsf{untag}_b(\mathsf{tag}_b(x)) \to x \end{array} \qquad \to_{\mathcal{O}_0}: \begin{cases} \mathsf{sdec}(\mathsf{senc}(x,y),y) \to x \\ \mathsf{adec}(\mathsf{aenc}(x,\mathsf{pk}(y)),y) \to x \\ \mathsf{proj}_1(\langle x,y\rangle) \to x \\ \mathsf{proj}_2(\langle x,y\rangle) \to y \\ \mathsf{check}(\mathsf{sign}(x,y),\mathsf{vk}(y)) \to x \end{cases}$$

Since the relation $\to_{\mathcal{O}}$ is convergent on ground terms, we define $t{\downarrow}$ as the unique normal form of the ground term t for $\to_{\mathcal{O}}$.

We can now state two lemmas that we will use in the next sections. Intuitively, the first lemma shows that the rewrite rules, applied during the normalisation of a term with factors in normal form, never modify these factors. The second lemma indicates that the normalisation and the replacement of factors commute, provided that the replacement preserved the equalities between factors.

Lemma 5.2 (Proved in [CD12])**.** *Let t be a ground term such that all its factors are in normal form and $\mathsf{root}(t) \in \mathcal{F}_i$ with $i \in \{a, b, \mathsf{tag}_a, \mathsf{tag}_b, 0\}$. Then*

— *either $t{\downarrow} \in Fct(t) \cup \{n_{min}\}$,*

— *or $\mathsf{root}(t{\downarrow}) \in \mathcal{F}_i$ and $Fct(t{\downarrow}) \subseteq Fct(t) \cup \{n_{min}\}$.*

Lemma 5.3. *Let t be a ground term such that $t = C[u_1, \ldots, u_n]$ where C is a context built on $\mathcal{F}_i$, $i \in \{a, b, \mathsf{tag}_a, \mathsf{tag}_b, 0\}$ and $u_1, \ldots, u_n$ are the factors of t in normal form. Furthermore, let D be the context built on $\mathcal{F}_i$ (possibly a hole) such that $t{\downarrow} = D[u_{j_1}, \ldots, u_{j_k}]$ with $j_1, \ldots, j_k \in \{0 \ldots n\}$ and $u_0 = n_{min}$ (the existence is given by Lemma 5.2). For all ground terms $v_1, \ldots, v_n$ in normal form and alien to t, if*

$$\forall (p, q) \in \{0 \ldots n\}, u_p = u_q \Leftrightarrow v_p = v_q$$

then $C[v_1, \ldots, v_n]{\downarrow} = D[v_{j_1}, \ldots, v_{j_k}]$ with $v_0 = n_{min}$

Proof. Thanks to Lemma 19 of [CD12], and the fact that $u_1, \ldots, u_n$ are factors in normal form, we know that the derivation of t is due to rewriting rules from $\mathcal{O}_i$. Hence, we do a proof by induction on the length L of the derivation

$$t \to_{\mathcal{O}_i} t_1 \to_{\mathcal{O}_i} t_2 \to_{\mathcal{O}_i} \ldots \to_{\mathcal{O}_i} t_n \to_{\mathcal{O}_i} t{\downarrow}$$

where each t_n are minimal among the terms N such that $t_{k-1} \to_{\mathcal{O}_i} N$.

Base case $L = 0$: In such a case, t is in normal form and so $C = D$, $n = k$ and $j_\ell = \ell$ for all $\ell \in \{1, \ldots, n\}$. Let $v_1, \ldots, v_n$ in normal form and alien to t such that for all $(p, q) \in \{0 \ldots n\}, u_p = u_q$ implies $v_p = v_q$. We show that $C[v_1, \ldots, v_n]$ is in normal form.

Assume that it is not the case. Let's denote $M = C[v_1, \ldots, v_n]$. Thanks to Lemma [CD12].19, we know that there exists a rule in $\mathcal{O}_i$ applied on M. Let M' be the minimal term for $\prec$ among the terms N such that $M \to_{\mathcal{O}_i} N$. Let $\ell = r \in \mathcal{O}_i$ applied on M at position p with substitution σ in order to obtain M'.

Since the factors $v_1, \ldots, v_n$ are in normal form, the position p is above or incomparable with any position corresponding to a factor of M. Thus, for all $x \in \mathit{fvars}(\ell)$, there exists a position p' of C such that $x\sigma = C[v_1, \ldots, v_n]|_{p'}$. Combined with the fact that t being in normal form implies that ℓ and $t|_p$ are not unifiable, we have that there exists two positions q, q' of C and a variable $x \in \mathit{fvars}(\ell)$ such that $x\sigma = C[v_1, \ldots, v_n]|_q = C[v_1, \ldots, v_n]|_{q'}$ and $C[u_1, \ldots, u_n]|_q \neq C[u_1, \ldots, u_n]|_{q'}$. Thus there exists $j, j' \in \{1, \ldots, n\}$ such that $v_j = v_{j'}$ and $u_j \neq u_{j'}$ which is in contradiction with our hypothesis.

Inductive step $L > 0$: We have $t \to_{\mathcal{O}_i} t_1 \to^*_{\mathcal{O}_i} t{\downarrow}$. Let $\ell = r \in \mathcal{O}_i$ applied on t at position p with substitution σ in order to obtain t'. Thanks to Lemma 5.2, we know that $t' = C'[u_{j_1}, \ldots, u_{j_k}]$, with $j_1, \ldots, j_k \in \{0, \ldots n\}$ and $u_0 = n_{min}$. With the same proof as in the base case, we can show that the same rule $\ell = r$ can be applied on $C[v_1, \ldots, v_n]$ at position p with a substitution σ'. By minimality of the term t_1 and monotonicity of $\prec$, we have that $\mathit{fvars}(r)\sigma' \subseteq \mathit{fvars}(\ell)\sigma' \cup \{n_{min}\}$ (same for σ). With p being a position of C and r is built upon $\mathcal{F}_i$, we have that $(C[u_1, \ldots, u_n])[r\sigma]_p = C'[u_{j_1}, \ldots, u_{j_k}]$ implies $(C[v_1, \ldots, v_n])[r\sigma']_p = C'[v_{j_1}, \ldots, v_{j_k}]$. Thus by application of our inductive hypothesis on t_1 and $C'[v_{j_1}, \ldots, v_{j_k}]$, the result holds. □

5.2.2 Derived well-tagged processes

We are now focus on the messages that are sent over the network during the execution of the processes. For a term u that does not contain any tag, we defined in Section 5.1, a way to construct a term that is properly tagged (*i.e.* $[u]_i$). Hence, for a term properly tagged, we would never have $\mathsf{senc}(n, k)$ where n and k are both nonces, for example. Instead, we would have $\mathsf{senc}(\mathsf{tag}_i(n), k)$. However, even if we can force the processes to properly tag their term, we do not have any control on what the intruder can build. Typically, if the intruder is able to deduce n and k, he is allowed to send to a process the term $\mathsf{senc}(n, k)$. Similarly, while we can restrict our processes to only apply vk and pk on a nonce, we can not restrict the intruder from using these cryptographic primitive with terms different from a nonce.

Hence, we need to describe what kind of frame and process we obtained in a trace of a tagged process. To do so, we assume from now on that processes and frames are coloured by a or b. Intuitively, colouring a process by a means that this process was derived from a process originally tagged by a. The same way, we say that a frame element $(ax \rhd u)$ of a frame is coloured by a

if u was output by a process derived from a process originally tagged by a. We denote $\mathsf{col}(ax)$ the colour of the frame element $(ax \rhd u)$, and $\mathsf{col}(P)$ the colour of the process P.

Definition 5.7. *Let $(\mathcal{E}; \mathcal{P}; \Phi)$ be an intermediate process. Let's denote $\Phi = \{ax_1 \rhd u_1, \ldots, ax_n \rhd u_n\}$. We say that $\nu\mathcal{E}.\Phi$ is a* derived well-tagged frame *if for all $i \in \{1, \ldots, n\}$, u_i is a derived well-tagged term up to ax_i,* i.e. *there exists a term v, a substitution α and $c \in \{a, b\}$ such that:*

- *for all $\mathsf{vk}(t), \mathsf{pk}(t') \in st(v)$, $t, t' \in \mathcal{N}$*
- *$u_i = [v]_c\alpha$; and*
- *$\alpha \vDash \mathsf{test}_c([v]_c)$; and*
- *for all $x \in \mathrm{dom}(\alpha)$, either v is not a variable and $x\alpha$ is a derived well-tagged term up to ax_i; or there exists M such that $fvars(M) \subseteq \{ax_1, \ldots, ax_{i-1}\}$, $fnames(M) \cap \mathcal{E} = \emptyset$ and $M\Phi = x\alpha$.*

The terms $[v]_c$ represent the term that was in the original process before instantiation, *e.g.* $\mathsf{out}(c, [v]_c)$. On the other hand α represents the value of the variables defined by the inputs, *e.g.* $\mathsf{in}(c, x)$. The third condition is due the fact that in a tagged process, there is always a condition that test $\mathsf{test}_c([v]_c)$ before outputting it. Hence, if u_i is in the frame, it necessary means that α satisfies the tests $\mathsf{test}_c([v]_c)$.

Note that the definition is recursive. Indeed, since α represents the instantiation of the variables defined by the inputs, given a variable x of $\mathrm{dom}(\alpha)$, either it was instantiated by the rule $\textsc{In}_i$ and so the intruder can deduce $x\alpha$; or it was instantiate by an internal communication, *i.e.* $\textsc{Comm}_i$, and so it means that $x\alpha$ is also well-tagged up to ax_i (since the term $x\alpha$ comes from another tagged process).

The condition *v is not a variable* allows us to avoid an infinite loop in the definition. Indeed, if this condition did not exist, then any term u would be well-tagged up to ax_i (take v a variable and α such that $v\alpha = u$). In our case, when v is a variable, it will necessary means that this variable was instantiate by the rule $\textsc{In}_i$ and so that the intruder could deduce it.

Example 5.11. *Coming back to Example 5.12, we have $\mathcal{E} = \{sk_S^a, sk_S^b, r'\}$ and the following frame:*

$$\Phi(A'_d) = \{ax_1 \rhd \mathsf{aenc}(\mathsf{tag}_a(\langle r', id_1\rangle), \mathsf{pk}(sk_S^a)); ax_2 \rhd \mathsf{aenc}(\mathsf{tag}_b(u), \mathsf{pk}(sk_S^b))\}$$

where $u = \mathsf{aenc}(\mathsf{tag}_a(\langle r', id_1\rangle), \mathsf{pk}(sk_S^a))$. This is a well-tagged frame.

Indeed, consider first $u_1 = \mathsf{aenc}(\mathsf{tag}_a(\langle r', id_1\rangle), \mathsf{pk}(sk_S^a))$. Let $v_1 = \mathsf{aenc}(\langle r', id_1\rangle, \mathsf{pk}(sk_S^a))$ and α_1 be the identity. We have:

- *$sk_S^a \in \mathcal{N}$ hence the first condition is satisfied*
- *$\mathsf{aenc}(\mathsf{tag}_a(\langle r', id_1\rangle), \mathsf{pk}(sk_S^a)) = [v_1]_a\alpha_1$*
- *$\mathsf{test}_a([v]_a) = \mathsf{true}$ and $\alpha_1 \vDash \mathsf{true}$.*
- *The last property is trivially true since $\mathrm{dom}(\alpha_1) = \emptyset$.*

Consider now $u_2 = \mathsf{aenc}(\mathsf{tag}_b(u), \mathsf{pk}(sk_S^b))$. Let $v_2 = \mathsf{aenc}(x, \mathsf{pk}(sk_S^b))$ and $\alpha_2 = \{x \mapsto u_1\}$. We have:

- *$sk_S^b \in \mathcal{N}$ hence the first condition is satisfies*
- *$[v_2]_b = \mathsf{aenc}(\mathsf{tag}_b(x), \mathsf{pk}(sk_S^b))$ hence $[v_2]_b\alpha_2 = u_2$*
- *Once again $\mathsf{test}_b([v_2]_b) = \mathsf{true}$ and $\alpha_2 \vDash \mathsf{true}$*
- *At last, we already show that $x\alpha_2 = u_1$ is well-tagged up to ax_1 hence it is also well-tagged up to ax_2. Note that the last property is also satisfied by the fact that $ax_1\Phi(A'_d) = x\alpha = u_1$.*

Similarly to the notion of a derived well-tagged frame, we introduce the notion of derived well-tagged process.

Definition 5.8. *Let P a coloured plain process and α be a ground substitution such that $fvars(P) \subseteq \mathrm{dom}(\alpha)$. We will says that (P, α) is a derived well-tagged process if*

— *either* $P = [Q]_i$;
— *or* $P = \mathsf{out}(u, [v]_i).[Q]_i$, $\alpha \vDash \mathsf{test}_i([v]_i)$;
— *or* $P = \texttt{if}\ [u]_i = [v]_i\ \texttt{then}\ [Q_1]_i\ \texttt{else}\ [Q_2]_i$ *with* $\alpha \vDash \mathsf{test}_i([u]_i) \wedge \mathsf{test}_i([v]_i)$
— *or* $P = \mathsf{if}\ \mathsf{test}_i([v]_i)\ \mathsf{then}\ \big(\mathsf{if}\ [u]_i = [v]_i\ \mathsf{then}\ [Q_1]_i\ \mathsf{else}\ [Q_2]_i\big)\ \mathsf{else}\ 0$ *with* $\alpha \vDash \mathsf{test}_i([u]_i)$

where $i = \mathsf{col}(P)$ *and* Q, Q_1, Q_2 *are processes built on* $\mathcal{F}_i \cup \mathcal{F}_0$*, and* u, v *are some terms.*

For a coloured multi-set of processes $\mathcal{P}$*, we say that* $(\mathcal{P}, \alpha)$ *is an original well-tagged multi-set of processes if for all* $P \in \mathcal{P}$*,* (P, α) *is an original well-tagged process.*

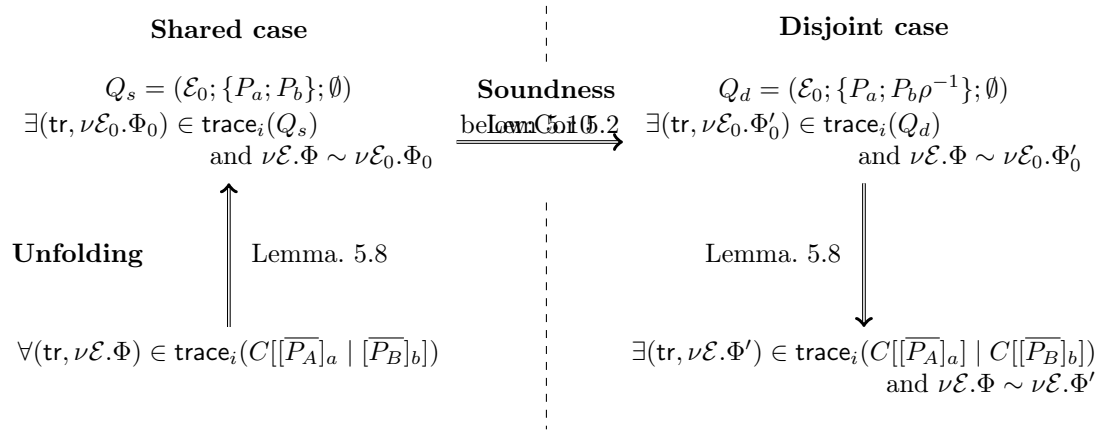
Intuitively, in a derived well-tagged process (P, α), P represents a subprocess of the original process P_a or P_b (depending on the colour of P) and α represents the values of the variables instantiated by the rule $\textsc{In}_i$. The definition of a derived well-tagged process expresses all the possible states of a plain process obtained from a derivation of $(\mathcal{E}_0; \{P_a, P_b\}; \emptyset)$ or $(\mathcal{E}_0; \{P_a, P_b\rho^{-1}\}; \emptyset)$.

5.3 Going back to the disjoint case

In this section, we will focus on our first composition result. As mentioned in previous sections, we show that the disjoint case (*i.e.* where the processes do not share any secret) and the shared case (*i.e.* when the processes share some secrets) are equivalent provided some conditions. More specifically, we will show that given two sequences of plain processes $\overline{P_A}$ and $\overline{P_B}$, built on $\mathcal{F}_a \cup \mathcal{F}_0$ and $\mathcal{F}_b \cup \mathcal{F}_0$ respectively, given a composition context C, we have that:

$$C[[\overline{P_A}]_a \mid [\overline{P_B}]_b] \approx_t C[[\overline{P_A}]_a] \mid C[[\overline{P_B}]_b].$$

if $C[[\overline{P_A}]_a]$ and $C[[\overline{P_B}]_b]$ do not reveal any shared key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \text{ occurs in } C\}$. A sketch of proof of right implication of this equivalence is given in Figure 5.1.



ρ is a fresh renaming of the shared private name and both Q_s and Q_d are bounded intermediate processes.

Figure 5.1: Sketch of proof of $C[[\overline{P_A}]_a \mid [\overline{P_B}]_b] \sqsubseteq_t C[[\overline{P_A}]_a] \mid C[[\overline{P_B}]_b]$

Intuitively, given a trace of the shared case (*i.e.* of the processes that share secret), we start by unfolding the replications in the process w.r.t. the trace so that we do not have any more replication or name restriction. Then, thanks to our soundness lemma, we show that this trace can be transform into a trace of the disjoint case (*i.e.* of the processes that do not share secret). We conclude by showing that a trace of a process that has been unfold can be transformed into a trace of this process. The sketch of proof for the left implication of this equivalence is similar although we use a completeness result, instead of a soundness result, in order to transform a trace on the disjoint case into a trace of the shared case.

This section is devoted to the complete proof of this result.

5.3.1 Name replacement

As mentioned in Figure 5.1, we have to transform a trace of the shared case into a trace of the disjoint case, and vice versa. In fact, we will see that the transformation will mainly consist of replacing some instances of the shared names by fresh ones.

Example 5.12. *Consider the process P_1 in Example 5.1, the composition context $C = \nu sk_S.\_$ and the process $Q = \mathsf{in}(c,x).\mathsf{out}(c,\mathsf{aenc}(x,\mathsf{pk}(sk_S)))$. We tag P_1 and Q respectively with the tags a and b. Hence the intermediate process corresponding to the shared case is the following:*

$$A_s = (\emptyset; C[\nu r.\mathsf{out}(c,\mathsf{aenc}(\mathsf{tag}_a(\langle r, id_1\rangle),\mathsf{pk}(sk_S))) \mid \mathsf{in}(c,x).\mathsf{out}(c,\mathsf{aenc}(\mathsf{tag}_b(x),\mathsf{pk}(sk_S)));\emptyset)$$

while the disjoint case corresponds to the following intermediate process:

$$A_d = (\emptyset; C[\nu r.\mathsf{out}(c,\mathsf{aenc}(\mathsf{tag}_a(\langle r, id_1\rangle),\mathsf{pk}(sk_S)))] \mid C[\mathsf{in}(c,x).\mathsf{out}(c,\mathsf{aenc}(\mathsf{tag}_b(x),\mathsf{pk}(sk_S)))];\emptyset)$$

Now, let tr be the sequence of labels $\nu ax_1.out(c,ax_1).in(c,ax_1).\nu ax_2.out(c,ax_2)$. We have that $A_s \overset{\mathsf{tr}}{\Rightarrow} A'_s$ and $A_d \overset{\mathsf{tr}}{\Rightarrow} A'_d$ where

$$A'_s = (\{sk_S^a, r'\};\emptyset;\{ax_1 \rhd \mathsf{aenc}(\mathsf{tag}_a(\langle r', id_1\rangle),\mathsf{pk}(sk_S^a));\ ax_2 \rhd \mathsf{aenc}(\mathsf{tag}_b(u),\mathsf{pk}(sk_S^a))\})$$

$$A'_d = (\{sk_S^a, sk_S^b, r'\};\emptyset;\{ax_1 \rhd \mathsf{aenc}(\mathsf{tag}_a(\langle r', id_1\rangle),\mathsf{pk}(sk_S^a));\ ax_2 \rhd \mathsf{aenc}(\mathsf{tag}_b(u),\mathsf{pk}(sk_S^b))\})$$

where $u = \mathsf{aenc}(\mathsf{tag}_a(\langle r', id_1\rangle),\mathsf{pk}(sk_S^a))$. In A'_d, sk_S^a corresponds to the renaming of the name sk_S restricted over P_1 while sk_S^b corresponds to the renaming of the name sk_S restricted over Q (application of the rule New_i). Furthermore, if we look at the second message of each frame, we can see that they are equal up to replacement of one instance of sk_S^a by sk_S^b. Typically, the key sk_S^a that is used to encrypt with the tag a stays sk_S^a in the disjoint case while the key sk_S^a that is used to encrypt with the tag b is replace by sk_S^b.

Even if Example 5.12 use very simple processes, we will see that all the messages in the processes and frames of two matched traces are equal up to a replacement of instances of "shared keys" with their corresponding "disjoint keys". We introduce a notion that formalise this replacement of names.

Let ρ be a bijective renaming of names of base type such that for all $k \in \mathrm{dom}(\rho)$, k is a "fresh name". Let δ_c^ρ ($c \in \{a,b\}$) be functions on terms that is defined as follows:

- $\delta_a^\rho(u) = u$ when u is a name or a variable;
- $\delta_b^\rho(u) = k$ when $u{\downarrow} = k\rho$ for some $k \in \mathrm{dom}(\rho)$ and $\mathsf{root}(u) \notin \mathcal{F}_b \cup \mathcal{F}_{\mathsf{tag}_b} \cup \mathcal{F}_0$; otherwise $\delta_b^\rho(u) = u$ when u is a name or a variable;
- $\delta_c^\rho(\mathsf{f}(t_1,\ldots,t_k)) = \mathsf{f}(\delta_d^\rho(t_1),\ldots,\delta_d^\rho(t_k))$ if $\mathsf{f} \in \mathcal{F}_d \cup \mathcal{F}_{\mathsf{tag}_d}$ with $d \in \{a,b\}$.
- $\delta_c^\rho(\mathsf{f}(\mathsf{tag}_d(t_1),t_2)) = \mathsf{f}(\mathsf{tag}_d(\delta_d^\rho(t_1)),\delta_d^\rho(t_2))$ if $\mathsf{f} \in \{\mathsf{senc},\mathsf{aenc},\mathsf{sign}\}$ and $d \in \{a,b\}$
- $\delta_c^\rho(\mathsf{h}(\mathsf{tag}_d(t_1))) = \mathsf{h}(\mathsf{tag}_d(\delta_d^\rho(t_1)))$ if $d \in \{a,b\}$
- $\delta_c^\rho(\mathsf{f}(t_1,\ldots,t_k)) = \mathsf{f}(\delta_c^\rho(t_1),\ldots,\delta_c^\rho(t_k))$ otherwise

Intuitively, $\mathrm{img}(\rho)$ correspond to the key generated in the shared case (same as the key generated by P_A in the disjoint case), *e.g.* sk_S^a, while $\mathrm{dom}(\rho)$ corresponds to keys generated by P_B in the disjoint case, *e.g.* sk_S^b. The purpose of δ_b^ρ is to replace the keys used by P_B in the shared case but created by P_A (*i.e.* $\mathrm{img}(\rho)$) with the keys generated by P_B (*i.e.* $\mathrm{dom}(\rho)$). We consider the names of $\mathrm{dom}(\rho)$ "fresh" in the sense that the names in $\mathrm{dom}(\rho)$ should never appear in the argument of δ_a^ρ and δ_b^ρ.

Example 5.13. *Coming back to Example 5.12, we have $\rho = \{sk_S^b \mapsto sk_S^a\}$, $\delta_a^\rho(ax_1\Phi(A'_s)) = ax_1\Phi(A'_d)$ and $\delta_b^\rho(ax_2\Phi(A'_s)) = ax_2\Phi(A'_d)$.*

Note that δ_a^ρ and δ_b^ρ do not behave the same way depending of the message being properly tagged or not.

Example 5.14. *Let* $\rho = \{sk_S^b \mapsto sk_S^a\}$ *and consider the following terms:*

— $u_1 = \mathsf{aenc}(n, \mathsf{pk}(sk_S^a))$ *and so* $\delta_a(u_1) = \mathsf{aenc}(n, \mathsf{pk}(sk_S^a))$ *and* $\delta_b(u_1) = \mathsf{aenc}(n, \mathsf{pk}(sk_S^b))$

— $u_2 = \mathsf{aenc}(\mathsf{tag}_a(n), \mathsf{pk}(sk_S^a))$ *and so* $\delta_a(u_2) = \delta_b(u_2) = \mathsf{aenc}(\mathsf{tag}_a(n), \mathsf{pk}(sk_S^a))$

— $u_3 = \mathsf{aenc}(\mathsf{tag}_b(n), \mathsf{pk}(sk_S^a))$ *and so* $\delta_a(u_3) = \delta_b(u_3) = \mathsf{aenc}(\mathsf{tag}_b(n), \mathsf{pk}(sk_S^b))$

Note that in both previous examples, the terms we considered are already in normal form. We prove three lemmas that describe some properties of δ_a^ρ and δ_b^ρ on terms in normal form (proofs in Appendix B.1).

Lemma 5.4. *If* t_1, t_2 *are terms (that do not use* $\mathrm{dom}(\rho)$*) in normal form then for all* $i \in \{a, b\}$*,* $t_1 = t_2$ *is equivalent to* $\delta_i^\rho(t_1) = \delta_i^\rho(t_2)$*.*

The previous lemma shows that δ_a^ρ and δ_b^ρ preserve equality between terms. It allows us to prove the following result:

Lemma 5.5. *Let* t_1, t_2 *two terms (that do not use* $\mathrm{dom}(\rho)$*) in normal form. If* $\delta_a^\rho(t_1) = \delta_b^\rho(t_2)$ *then* $t_1 = t_2$*.*

While the two previous lemmas mainly focus on the relations between equality and the name replacement. The following lemma indicates that the root and irreductibility of terms is also preserved by δ_a^ρ and δ_b^ρ.

Lemma 5.6. *Let* u *be a term in normal form that do no use* $\mathrm{dom}(\rho)$*. We have that for all* $i \in \{a, b\}$*,* $\delta_i^\rho(u)$ *is in normal form and* $\mathsf{root}(\delta_i^\rho(u)) = \mathsf{root}(u)$*.*

5.3.2 Unfolding the processes

The first difficulty is the presence of the replication in the composition context C and also in the sequences of processes $\overline{P_A}$ and $\overline{P_B}$. The main idea is to unfold the replication in advance so that replication is not needed anymore. Of course, it is impossible to unfold in advance a process P so that all traces of P are included in this unfolded process. However, given a trace P, it is possible to unfold P so that at least this specific trace is included in the unfolded process. The main idea of unfolding a process P given a trace of P is to replace any instance of $!Q$ in P by $Q \mid \ldots \mid Q$ where the number of parallel composition is higher than the number of application of $\textsc{Repl}_i$ in the derivation of the trace. At last, similarly to Chapter 4, we move forward the name restriction so that the unfolded process is an bounded intermediate process (*i.e.* without replication nor name restriction).

Example 5.15. *Consider the processes* P_1 *(renamed* P*) and* Q *in Example 5.12. Furthermore, consider the composition context* $C = \nu sk_S.!\_$ *and consider the intermediate process* $A_s = (\emptyset; C[P \mid Q]; \emptyset)$*. We have that for all* $n \in \mathbb{N}$*,* $(\mathsf{tr}_n, \nu\mathcal{E}_n.\Phi_n) \in \mathit{trace}(A_s)$ *where:*

— $\mathsf{tr}_n = \nu ax_1.out(c, ax_1).\ldots.\nu ax_n.out(c, ax_n)$

— $\mathcal{E}_n = \{sk_S, sk'_S, r_1, \ldots, r_n\}$

— $\Phi_n = \{ax_1 \rhd \mathsf{aenc}(\mathsf{tag}_a(\langle r_1, id_1\rangle), \mathsf{pk}(sk_S)); \ldots; ax_n \rhd \mathsf{aenc}(\mathsf{tag}_a(\langle r_n, id_1\rangle), \mathsf{pk}(sk_S))$

Now consider the intermediate process $A_s^f = (\emptyset; \{\nu sk_S.\big(P_1 \mid \ldots \mid P_n \mid Q_1 \mid \ldots \mid Q_n\big)\}; \emptyset)$ *where for all* $i \in \{1, \ldots, n\}$*,* $P_i = P$ *and* $Q_i = Q$*. The process* A_s^f *is an unfolded process of* A_s *given the trace* $(\mathsf{tr}_n, \nu\mathcal{E}_n.\Phi_n)$*. Furthermore, we have that* $(\mathsf{tr}_n, \nu\mathcal{E}_n.\Phi_n) \in \mathit{trace}(A_s^f)$*.*

We can announce an intuitive result on unfolded process:

Lemma 5.7. *Let* $n \in \mathbb{N}$*. Let* A *be an intermediate process and* A' *be the unfolded intermediate process where we replace every instance of* $!P$ *in* A *by* $P_1 \mid \ldots \mid P_n$ *with* $P_i = P$ *for all* $i \in \{1, \ldots, n\}$*. We have that*

— $\mathit{trace}(A') \subseteq \mathit{trace}(A)$

— *if* $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(A)$ *such that the derivation contains less than* n *application of* REPL_i *then* $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(A')$

Proof (sketch). Intuitively, given a trace of A' the idea is to replace each τ action corresponding to the application of PAR_i on an instance $P \mid \ldots \mid P$, by an application of REPL_i on $!P$.

Similarly, to show that $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(A')$, we replace each τ action corresponding to the application of REPL_i on an instance $!P$, by an application of PAR_i on $P \mid \ldots \mid P$. It is possible since we know that there is less than n application of REPL_i. □

Note that in the previous example, we only focused on the "shared case". But, since we have to link the shared case with the disjoint case, we show that we can unfold the same way the intermediate processes representing the shared and disjoint case. This can be expressed with the following lemma.

Lemma 5.8. *Let* C *be a composition context. Let* $\overline{P_A}$ *(resp.* $\overline{P_B}$*) be a sequences of plain processes built on* $\mathcal{F}_a \cup \mathcal{F}_0$ *(resp.* $\mathcal{F}_b \cup \mathcal{F}_0$*).*

Let $D = (\emptyset; C[[\overline{P_A}]_a] \mid C[[\overline{P_B}]_b]; \emptyset)$ *and* $S = (\emptyset; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]; \emptyset)$*, we have that for all* $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(D)$ *(resp.* $\mathsf{trace}(S)$*), there exists a renaming* ρ *and two bounded intermediate processes* $S' = (\mathcal{E}_0; \{P_a, P_b\}; \emptyset)$ *and* $D' = (\mathcal{E}_0; \{P_a, P_b\rho^{-1}\}; \emptyset)$ *such that*

— $\text{dom}(\rho) \cup \text{img}(\rho) \subseteq \mathcal{E}_0$ *and* $\text{dom}(\rho)$ *does not appear in* $\{P_a, P_b\}$

— *for all* $i \in \{a, b\}$*, there exists* P_i' *built on* $\mathcal{F}_i \cup \mathcal{F}_0$ *such that such that* $P_i = [P_i']_i$

— *there exists* Φ' *such that* $(\mathsf{tr}, \nu\mathcal{E}_0.\Phi') \in \mathsf{trace}(D')$ *(resp.* $\mathsf{trace}(S')$*) and* $\nu\mathcal{E}_0.\Phi' \sim \nu\mathcal{E}.\Phi$

— *for all* $(\mathsf{tr}', \nu\mathcal{E}_0.\Phi'') \in \mathsf{trace}(S')$ *(resp* $\mathsf{trace}(D')$*), there exists* $\nu\mathcal{E}'.\Phi'$ *such that* $(\nu\mathcal{E}'.\Phi') \in \mathsf{trace}(S)$ *(resp* $\mathsf{trace}(D)$*) and* $\nu\mathcal{E}'.\Phi' \sim \nu\mathcal{E}_0.\Phi''$

Note that $\delta_a^\rho(P_a) = P_a$ *and* $\delta_b^\rho(P_b) = P_b\rho^{-1}$

Proof. Assume w.l.o.g that $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(S)$. Hence, we have that $S \overset{\mathsf{tr}}{\Rightarrow} (\mathcal{E}; \mathcal{P}; \Phi)$ for some $\mathcal{P}$. Thus, there exists a word w such that $S \xrightarrow{w} (\mathcal{E}; \mathcal{P}; \Phi)$ where $\mathsf{tr} = w \setminus \tau$. Let N be the number of τ action corresponding to the application of the rule REPL_i.

Let C' be the composition context without replication, $\overline{P_A'}$ and $\overline{P_B'}$ the sequences of plain process without replication such that $C'[[\overline{P_A'}]_a \mid [\overline{P_B'}]_b]$ corresponds to $C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]$ where we replaced every instance $!P$ by $P_1 \mid \ldots \mid P_N$ for some where P_i is a renaming of P.

Let P_a be the plain process $C'[[\overline{P_A'}]_a]$ where we removed all name restriction and let P_b be the plain process $C'[[\overline{P_B'}]_b]$ where we removed all name restriction. At last, let ρ be a renaming such that $\text{dom}(\rho)$ are fresh names and $\text{img}(\rho) = \mathit{bnames}(C')$. At last, let $\mathcal{E}_0 = \mathit{bnames}(C') \cup \mathit{bnames}(\overline{P_A'}) \cup \mathit{bnames}(\overline{P_B'}) \cup \text{dom}(\rho)$.

Let $S' = (\mathcal{E}_0, \{P_a, P_b\}, \emptyset)$ and $D' = (\mathcal{E}_0\{P_a, P_b\rho^{-1}\}, \emptyset)$. We have that S' is a bounded intermediate process associated to $(\emptyset, \{C'[[\overline{P_A'}]_a \mid [\overline{P_B'}]_b]\}, \emptyset)$. Furthermore, since $\text{dom}(\rho)$ is only composed of fresh names, $P_b\rho^{-1}$ is a renaming of $C'[[\overline{P_B'}]_b]$ and so D' is a bounded intermediate process associated to $(\emptyset, \{C'[[\overline{P_A'}]_a] \mid C'[[\overline{P_B'}]_b]\}, \emptyset)$. We conclude by applying Lemma 5.7. □

Example 5.16. *Coming back to Example 5.15, consider the intermediate process* A_s *and the trace* $(\mathsf{tr}_n, \nu\mathcal{E}_n.\Phi_n) \in \mathsf{trace}(A_s)$*. Furthermore, consider the intermediate process* $A_d = (\emptyset; C[P] \mid C[Q]; \emptyset)$*. The process* D' *and* S' *obtained thanks to Lemma 5.8 from* A_d *and* A_s *are the processes* $(\mathcal{E}_0; \{P_a, P_b\}; \emptyset)$ *and* $(\mathcal{E}_0; \{P_a, P_b\rho^{-1}\}; \emptyset)$ *where:*

— $\mathcal{E}_0 = \{sk_S, sk_S', r_1, \ldots, r_n\}$

— $\rho = \{sk_S' \mapsto sk_S\}$

— $P_a = [P_1 \mid \ldots \mid P_n]_a$ *where* $P_i = \mathsf{out}(c, \mathsf{aenc}(\mathsf{tag}_a(\langle r_i, id_1\rangle), \mathsf{pk}(sk_S)))$*, for all* $i = 1 \ldots n$

— $P_b = [Q_1 \mid \ldots \mid Q_n]_b$ *where* $Q_i = \mathsf{in}(c, x).\mathsf{out}(c, \mathsf{aenc}(x, \mathsf{pk}(sk_S')))$

5.3.3 Soundness and completeness

In this subsection, we focus our attention on the transformation of a trace from the "shared case" (resp. "disjoint case") to a trace from the "disjoint case" (resp. "shared case").

We will assume that processes of different colours do not share private channels. We denote by $\delta^\rho(\cdot)$ the function that apply $\delta^\rho_a(\cdot)$ on frame elements or processes coloured by a, and that apply $\delta^\rho_b(\cdot)$ on frame elements or processes coloured by b. Moreover, given $i \in \{a, b\}$ and α a ground substitution, we denote by $\delta^\rho_i(\alpha)$ the substitution such that $\mathrm{dom}(\alpha) = \mathrm{dom}(\delta^\rho_i(\alpha))$ and for all $x \in \mathrm{dom}(\alpha)$, $x\delta^\rho_i(\alpha) = \delta^\rho_i(x\alpha)$.

5.3.3.1 Static equivalence

We first show that a frame Φ in the shared case and its corresponding frame in the disjoint case, *i.e.* $\delta^\rho(\Phi)$ are statically equivalent. This result is derived from the following lemma (proof in Appendix B.2):

Lemma 5.9. *Let $\mathcal{E}$ be a set of names and $\Phi = \{ax_1 \rhd u_1, \ldots, ax_n \rhd u_n\}$ such that $\nu\mathcal{E}.\Phi$ is a derived well-tagged frame in normal form. Let ρ be a renaming such that $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$ and $\mathrm{dom}(\rho) \cap fnames(\Phi) = \emptyset$. If one of the two following conditions is satisfied:*

(a) for all $k \in \mathrm{img}(\rho)$, $\nu\mathcal{E}.\Phi \not\vdash k$, $\nu\mathcal{E}.\Phi \not\vdash \mathsf{pk}(k)$ and $\nu\mathcal{E}.\Phi \not\vdash \mathsf{vk}(k)$

(b) for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash k$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash \mathsf{pk}(k)$ and $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash \mathsf{vk}(k)$;

then for all M such that $fvars(M) \subseteq \mathrm{dom}(\Phi)$ and $fnames(M) \cap \mathcal{E} = \emptyset$, for all $i \in \{a, b\}$, $\delta^\rho_i(M\Phi\!\downarrow) = M\delta^\rho(\Phi)\!\downarrow$.

Intuitively, Lemma 5.9 shows that the knowledge deduced by the intruder is the same (up to renaming δ^ρ) in Φ and $\delta^\rho(\Phi)$ as long as no shared key, public key and verification key are revealed (represented by $\mathrm{img}(\rho)$ and $\mathrm{dom}(\rho)$). From this lemma, we can derive two corollaries:

Corollary 5.1. *Let $\mathcal{E}$ be a set of names. Let Φ such that $\nu\mathcal{E}.\Phi$ is a derived well-tagged frame in normal form. Let ρ a renaming such that $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$ and $\mathrm{dom}(\rho) \cap fnames(\Phi) = \emptyset$. The two following properties are equivalent:*

- *for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash k$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash \mathsf{pk}(k)$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash \mathsf{vk}(k)$*
- *for all $k \in \mathrm{img}(\rho)$, $\nu\mathcal{E}.\Phi \not\vdash k$, $\nu\mathcal{E}.\Phi \not\vdash \mathsf{pk}(k)$, $\nu\mathcal{E}.\Phi \not\vdash \mathsf{vk}(k)$*

Proof. Assume first that for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash k$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash \mathsf{pk}(k)$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash \mathsf{vk}(k)$. Assume now that there exists $k \in \mathrm{img}(\rho)$ such that $\nu\mathcal{E}.\Phi \vdash k$. Hence there exists M such that $fvars(M) \subseteq \mathrm{dom}(\Phi)$, $fnames(M) \cap \mathcal{E} = \emptyset$ and $M\Phi\!\downarrow = k$. But by Lemma 5.9, we deduce that $\delta^\rho_a(M\Phi\!\downarrow) = M\delta^\rho(\Phi)\!\downarrow$ and so $M\delta^\rho(\Phi)\!\downarrow = \delta^\rho_a(k) = k$. Hence $\nu\mathcal{E}.\delta^\rho(\Phi) \vdash k$ which is a contradiction with our hypothesis. All the other cases are done in a similar way. □

Thanks to this result, we will be able to prove that if the shared keys are not revealed in the disjoint case then they will not be revealed in the shared case.

Corollary 5.2. *Let $\mathcal{E}$ be a set of names. Let Φ such that $\nu\mathcal{E}.\Phi$ is a derived well-tagged frame in normal form and let $\mathcal{E}$ be a set of names. Let ρ be a renaming such that $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$ and $\mathrm{dom}(\rho) \cap fnames(\Phi) = \emptyset$. If for all $k \in \mathrm{img}(\rho)$, $\nu\mathcal{E}.\Phi \not\vdash k$, $\nu\mathcal{E}.\Phi \not\vdash \mathsf{pk}(k)$ and $\nu\mathcal{E}.\Phi \not\vdash \mathsf{vk}(k)$, then we have $\nu\mathcal{E}.\Phi \sim \nu\mathcal{E}.\delta^\rho(\Phi)$.*

Proof. Let M_1, M_2 two term such that $fvars(M_1, M_2) \subseteq \mathrm{dom}(\Phi)$ and $fnames(M_1, M_2) \cap \mathcal{E} = \emptyset$. Thanks to Lemma 5.4, we have that $M_1\Phi\!\downarrow = M_2\Phi\!\downarrow$ is equivalent to $\delta^\rho_a(M_1\Phi\!\downarrow) = \delta^\rho_a(M_2\Phi\!\downarrow)$. But thanks to Lemma 5.9, this is equivalent to $M_1\delta^\rho(\Phi)\!\downarrow = M_2\delta^\rho(\Phi)\!\downarrow$. Thus the result holds. □

5.3.3.2 Soundness

We now give our soundness lemma which shows that any trace of the "shared case" can be matched by a similar trace in the "disjoint case".

Lemma 5.10 (Soundness)**.** *Let $S = (\mathcal{E}_S; \mathcal{P}_S; \Phi_S)$, $S' = (\mathcal{E}'_S; \mathcal{P}'_S; \Phi'_S)$ and $D = (\mathcal{E}_D; \mathcal{P}_D; \Phi_D)$ be three bounded intermediate processes. Assume that $S \xrightarrow{\ell} S'$, Φ_S is well-tagged and there exists a derived well-tagged multi-set of processes $(\mathcal{P}_0, \alpha)$ and a renaming ρ, such that*

- $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_S$, $\mathrm{dom}(\rho) \cap \mathit{fnames}(\mathcal{P}_S, \Phi_S) = \emptyset$*; and*
- $\mathcal{E}_S = \mathcal{E}_D$, $\Phi_D{\downarrow} = \delta^\rho(\Phi_S{\downarrow})$*; and*
- $\mathcal{P}_S = \mathcal{P}_0\alpha$ *and* $\mathcal{P}_D{\downarrow} = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha{\downarrow}){\downarrow}$*; and*
- *for all traces* (tr, Φ) *of D, for all* $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\Phi \not\vdash k$, $\Phi \not\vdash \mathsf{pk}(k)$ *and* $\Phi \not\vdash \mathsf{vk}(k)$.

There exists a bounded intermediate process $D' = (\mathcal{E}'_D; \mathcal{P}'_D; \Phi'_D)$, a derived well tagged multi-set of processes $(\mathcal{P}'_0, \alpha')$ such that Φ'_S is well-tagged and:

- $\mathcal{E}'_S = \mathcal{E}_S = \mathcal{E}'_D$ *and* $\Phi'_D{\downarrow} = \delta^\rho(\Phi'_S{\downarrow})$*; and*
- $\mathcal{P}'_S{\downarrow} = \mathcal{P}'_0\alpha'{\downarrow}$ *and* $\mathcal{P}'_D{\downarrow} = \delta^\rho(\mathcal{P}'_0)\delta^\rho(\alpha'{\downarrow}){\downarrow}$*; and*
- $D \xrightarrow{\ell} D'$.

5.3.3.3 Completeness

On the other hand, our completeness lemma shows that any trace of the "disjoint case" can be matched by a similar trace in the "shared case".

Lemma 5.11 (Completeness)**.** *Let $S = (\mathcal{E}_S; \mathcal{P}_S; \Phi_S)$, $D = (\mathcal{E}_D; \mathcal{P}_D; \Phi_D)$ and $D' = (\mathcal{E}'_D; \mathcal{P}'_D; \Phi'_D)$ and be three bounded intermediate processes. Assume that $D \xrightarrow{\ell} D'$, Φ_S is a derived well-tagged frame and there exists a derived well-tagged multi-set of processes $(\mathcal{P}_0, \alpha)$ and a renaming ρ, such that*

- $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_S$, $\mathrm{dom}(\rho) \cap \mathit{fnames}(\mathcal{P}_S, \Phi_S) = \emptyset$*; and*
- $\mathcal{E}_S = \mathcal{E}_D$, $\Phi_D{\downarrow} = \delta^\rho(\Phi_S{\downarrow})$*; and*
- $\mathcal{P}_S = \mathcal{P}_0\alpha$ *and* $\mathcal{P}_D{\downarrow} = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha{\downarrow}){\downarrow}$*; and*
- *for all traces* (tr, Φ) *of D, for all* $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\Phi \not\vdash k$, $\Phi \not\vdash \mathsf{pk}(k)$ *and* $\Phi \not\vdash \mathsf{vk}(k)$.

There exists a bounded intermediate process $S' = (\mathcal{E}'_S; \mathcal{P}'_S; \Phi'_S)$, a derived well tagged multi-set of processes $(\mathcal{P}'_0, \alpha')$ such that Φ'_S is a derived well-tagged frame and:

- $\mathcal{E}'_S = \mathcal{E}_S = \mathcal{E}'_D$ *and* $\Phi'_D{\downarrow} = \delta^\rho(\Phi'_S{\downarrow})$*; and*
- $\mathcal{P}'_S{\downarrow} = \mathcal{P}'_0\alpha'{\downarrow}$ *and* $\mathcal{P}'_D{\downarrow} = \delta^\rho(\mathcal{P}'_0)\delta^\rho(\alpha'{\downarrow}){\downarrow}$*; and*
- $S \xrightarrow{\ell} S'$.

The proofs of Lemmas 5.10 and 5.11 can be found in Appendix B.2. Note that in both lemmas, the invariants are the same. $\mathcal{P}_0$ represents subprocesses of the original process and α represents the value of the variables restricted by the inputs.

5.3.4 Main result

We conclude this section by the first main result of this chapter:

Theorem 5.1. *Let C a composition context. Let $\overline{P_A}$ and $\overline{P_B}$ two sequences of plain processes built on $\mathcal{F}_a \cup \mathcal{F}_0$ and $\mathcal{F}_b \cup \mathcal{F}_0$ respectively. If $C[[\overline{P_A}]_a]$ and $C[[\overline{P_B}]_b]$ do not reveal any shared key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k$ occurs in $C\}$ then we have that:*

$$C[[\overline{P_A}]_a \mid [\overline{P_B}]_b] \approx_t C[[\overline{P_A}]_a] \mid C[[\overline{P_B}]_b].$$

Proof. Let's denote $S = (\emptyset; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]; \emptyset)$ and $D = (\emptyset; C[[\overline{P_A}]_a] \mid C[[\overline{P_B}]_b]; \emptyset)$. We first show that for all $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(S)$, there exists $\nu\mathcal{E}'.\Phi'$ such that $(\mathsf{tr}, \nu\mathcal{E}'.\Phi') \in \mathsf{trace}(D)$ and $\nu\mathcal{E}.\Phi \sim \nu\mathcal{E}'.\Phi'$.

Let $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(S)$. Thanks to Lemma 5.8, we know that there exist a renaming ρ and two bounded intermediate processes $S' = (\mathcal{E}_0; \{P_a, P_b\}; \emptyset)$ and $D' = (\mathcal{E}_0; \{P_a, P_b\rho^{-1}\}; \emptyset)$ such that

- $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_0$ and $\mathrm{dom}(\rho)$ does not appear in $\{P_a, P_b\}$
- for all $i \in \{a, b\}$, there exists P'_i built on $\mathcal{F}_i \cup \mathcal{F}_0$ such that $P_i = [P'_i]_i$
- there exists Φ_0 such that $(\mathsf{tr}, \nu\mathcal{E}_0.\Phi_0) \in \mathsf{trace}(S')$ and $\nu\mathcal{E}_0.\Phi_0 \sim \nu\mathcal{E}.\Phi$
- for all $(\mathsf{tr}', \nu\mathcal{E}_0.\Phi'') \in \mathsf{trace}(D')$, there exists $\nu\mathcal{E}'.\Phi'$ such that $(\mathsf{tr}', \nu\mathcal{E}'.\Phi') \in \mathsf{trace}(D)$ and $\nu\mathcal{E}'.\Phi' \sim \nu\mathcal{E}_0.\Phi''$

$(\mathsf{tr}, \nu\mathcal{E}_0.\Phi_0) \in \mathsf{trace}(S')$ implies that there exists a bounded intermediate process $S'' = (\mathcal{E}_0; \mathcal{P}_S; \Phi_0)$ such that $S' \xrightarrow{w} S''$ where $\mathsf{tr} = w \setminus \tau$. Furthermore, we have $P_a = [P'_a]_a$ and $P_b = [P'_b]_b$ hence $(\{P_a, P_b\}, id)$ is a derived well-tagged multi-set of process. Moreover, since P_a and P_b are coloured with a and b respectively, and $\delta^\rho_a(P_a) = P_a$, $\delta^\rho_b(P_b) = P_b\rho^{-1}$, we deduce that $\{P_a, P_b\rho^{-1}\}\downarrow = \delta^\rho(\{P_a, P_b\})\downarrow$. At last, by hypothesis we know that $C[[\overline{P_A}]_a]$ and $C[[\overline{P_B}]_b]$ do not reveal any shared key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \text{ occurs in } C\}$ hence it implies that $C[[\overline{P_A}]_a] \mid C[[\overline{P_B}]_b]$ does not reveal any shared key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \text{ occurs in } C\}$. Thus, we conclude thanks to Lemma 5.8 that D does not reveal any shared key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$.

We have shown that S' and D' satisfy the conditions of Lemma 5.10, hence using a simple induction on the size of the derivation $S' \xrightarrow{w} S''$ and thanks to Lemma 5.10, we show that there exists a bounded intermediate process $D'' = (\mathsf{E}_0; \mathcal{P}_D; \Phi'_0)$ such that $D' \overset{\mathsf{tr}}{\Rightarrow} D''$, Φ'_0 is a derived well-tagged frame and $\Phi'_0\downarrow = \delta^\rho(\Phi_0\downarrow)$. But thanks to Corollaries 5.1 and 5.2, we deduce that $\nu\mathcal{E}_0.\Phi_0\downarrow \sim \nu\mathcal{E}_0.\delta^\rho(\Phi_0\downarrow)$. It implies that $\nu\mathcal{E}_0.\Phi_0\downarrow \sim \nu\mathcal{E}_0.\Phi'_0\downarrow$ and so $\nu\mathcal{E}_0.\Phi_0 \sim \nu\mathcal{E}'_0.\Phi'_0$.

Since $(\mathsf{tr}, \nu\mathcal{E}_0.\Phi'_0) \in \mathsf{trace}(D')$, then by Lemma 5.8, we also deduce that there exists $\nu\mathcal{E}'.\Phi'$ such that $(\nu\mathcal{E}'.\Phi') \in \mathsf{trace}(D)$ and $\nu\mathcal{E}'.\Phi' \sim \nu\mathcal{E}_0.\Phi'_0$. But we have $\nu\mathcal{E}_0.\Phi_0 \sim \nu\mathcal{E}.\Phi$ and $\nu\mathcal{E}_0.\Phi_0 \sim \nu\mathcal{E}_0.\Phi'_0$, hence we conclude that $\nu\mathcal{E}.\Phi \sim \nu\mathcal{E}'.\Phi'$.

Finally, we prove that for all $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(D)$, there exists $\nu\mathcal{E}'.\Phi'$ such that $(\mathsf{tr}, \nu\mathcal{E}'.\Phi') \in \mathsf{trace}(S)$ and $\nu\mathcal{E}.\Phi \sim \nu\mathcal{E}'.\Phi'$. The proof is symmetrical to the first case except that we apply Lemma 5.11 instead of Lemma 5.10. □

5.3.5 A first composition result

From the previous theorem, we can derive our first composition result.

Corollary 5.3. *Let C and C' be two composition contexts. Let $\overline{P_A}, \overline{P'_A}$ (resp. $\overline{P_B}, \overline{P'_B}$) be two sequences of plain processes built on the signature $\mathcal{F}_a \cup \mathcal{F}_0$ (resp. $\mathcal{F}_b \cup \mathcal{F}_0$). Assume that $C[[\overline{P_A}]_a]$ and $C[[\overline{P_B}]_b]$ (resp. $C'[[\overline{P'_A}]_a]$,$C'[[\overline{P'_B}]_b]$) do not reveal any shared key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k$ occurs in $C\}$ (resp. $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k$ occurs in $C'\}$). We have that:*

$$\frac{C[[\overline{P_A}]_a] \approx_t C'[[\overline{P'_A}]_a] \quad C[[\overline{P_B}]_b] \approx_t C'[[\overline{P'_B}]_b]}{C[[\overline{P_A}]_a \mid [\overline{P_B}]_b] \approx_t C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b]}$$

Proof. This composition result is proved in two main steps.

1. We have the equivalences $C[[\overline{P_A}]_a] \approx_t C'[[\overline{P'_A}]_a]$ and $C[[\overline{P_B}]_b] \approx_t C'[[\overline{P'_B}]_b]$. Thus, thanks to Lemma 5.1, we can show that:
$$C[[\overline{P_A}]_a] \mid C[[\overline{P_B}]_b] \approx_t C'[[\overline{P'_A}]_a] \mid C'[[\overline{P'_B}]_b].$$
2. Then, we apply Theorem 5.1 on both sides of the equivalence, and we obtain the expected result:
$$C[[\overline{P_A}]_a \mid [\overline{P_B}]_b] \approx_t C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b].$$

□

Note that in the hypothesis of Corollary 5.3, the equivalences $C[[\overline{P_A}]_a] \approx_t C'[[\overline{P'_A}]_a]$ and $C[[\overline{P_B}]_b] \approx_t C'[[\overline{P'_B}]_b]$ holds for the signatures $(\mathcal{F}_a^+ \cup \mathcal{F}_b^+ \cup \mathcal{F}_0, \mathsf{E}_a^+ \cup \mathsf{E}_b^+ \cup \mathsf{E}_0)$. In fact, one can show that $C[[\overline{P_A}]_a] \approx_t C'[[\overline{P'_A}]_a]$ only have to hold on the signature $(\mathcal{F}_a^+ \cup \mathcal{F}_0, \mathsf{E}_a^+ \cup \cup \mathsf{E}_0)$ which is more natural. Indeed, the process $\overline{P_A}$ and $\overline{P'_A}$ are only build on $\mathcal{F}_a^+ \cup \mathcal{F}_0$ and the signatures $(\mathcal{F}_a^+ \cup \mathcal{F}_0, \mathsf{E}_a^+ \cup \cup \mathsf{E}_0)$ and $(\mathcal{F}_b^+, \mathsf{E}_b^+)$ are disjoint. Even if [CD12] only shows a result on static equivalence and not trace equivalence, one can adapt their proof technique to show this result.

The main issue with this composition result is the hypothesis that no shared public or verification key can be deduced by the intruder. This lies in opposition to the purpose of the public and verifications that is to be known by anyone.

However, in some cases, it is not unreasonable to assume that some verification or public keys are not deducible by the intruder. For example, coming back to the *e-passport protocol* (see Section 3.4), we know that each passport has a private signing key sk_P stored in a tamper resistant memory and the associated verification key $\mathsf{vk}(sk_P)$ is stored in the data group dg_{15}. However, this data group is never send in clear over the network. Typically, in the Passive Authentication protocol, the data groups are encrypted under the symmetric session key *ksenc* obtained after execution of the Basic Access Control protocol. Hence, the verification of the e-passport will not be deducible by the intruder, provided of course that the intruder did not already know such verification key.

This could model for example an intruder that try to gain informations on some e-passports without any specific target in mind. Although one can assume that an intruder knows the verification keys of few targeted passports, it seems unrealistic to assume that an intruder knows the verification key of all the passports that have been hanged up so far. In such a case, one could apply our first composition result for the unlinkability of the parallel composition of the tagged protocols AA and PA.

Indeed, this security property is given by the following equivalence:

$$\nu sk_{DS}.C'[[PA]_a \mid [AA]_b] \approx_t \nu sk_{DS}.C''[[PA]_a \mid [AA]_b]$$

where $C''[\_] \stackrel{def}{=} !\nu sk_P.\, \nu id.\, \nu sig.\, \nu pic.\, \ldots \nu ksenc.\, \nu ksmac.\, \_$ and $C'[\_] \stackrel{def}{=} !\nu sk_P.\, \nu id.\, \nu sig.\, \nu pic.\, \ldots !\, \nu ksenc.\, \nu ksmac.\, \_$. Moreover, as mentioned in Example 5.7, the private signing key sk_{DS} is not a shared key since it is only used in the Passive Authentication protocol and so $\mathsf{vk}(sk_{DS})$ is allowed to be deducible by the intruder. Hence, it would remain to verify that the session keys *ksenc*, *ksmac* and the signing key sk_{DS} are not deducible; and of course that the tagged protocols satisfy the unlinkability property in isolation.

5.4 Main composition result

In the previous section, we presented a first composition result. However, this result does not hold as soon as some shared keys are revealed: such a key can be a symmetric shared key, the private part of an asymmetric key pair, but also the public part of an asymmetric key pair. In this section, we will see that we can relax this condition by allowing shared public or verification keys to be revealed from the beginning.

5.4.1 Some additional difficulties

First, as shown by the example below, we do not want public keys to be revealed (for the first time) during the execution of the protocol.

Example 5.17. *We consider a slightly different version of the process P_i introduced in Example 5.1. Basically, we remove the random r inside the encryption and we consider its well-tagged version. We consider the following processes:*

$$[P'_i]_a \stackrel{def}{=} \mathsf{out}(c, \mathsf{aenc}(\mathsf{tag}_a(id_i), \mathsf{pk}(sk_S))) \quad i \in \{1, 2\}$$

Consider the composition context $C[\_] = \nu sk_S.\ \_$. *Note that, the equivalence* $C[[P'_1]_a] \approx_t C[[P'_2]_a]$ *still holds in this setting. Assume now that* $[P'_i]_a$ *is executed in presence of the well-tagged process* $Q^{\mathsf{pk}} = \mathsf{out}(c, \mathsf{pk}(sk_S))$. *Clearly, the equivalence expressing the anonymity of* $[P'_i]_a$ *does not hold anymore. We have that:*

$$C[[P'_1]_a \mid Q^{\mathsf{pk}}] \not\approx_t C[[P'_2]_a \mid Q^{\mathsf{pk}}]$$

Actually, the knowledge of $\mathsf{pk}(sk_S)$ *will allow the attacker to distinguish the message emitted by* $[P'_1]_a$ *from the one emitted by* $[P'_2]_a$ *by rebuilding the message* $\mathsf{aenc}(\mathsf{tag}_a(id_1), \mathsf{pk}(sk_S))$ *since* id_1 *is public, and comparing it to the message outputted by* P'_1 *and* P'_2.

To avoid the problem mentioned above, we will assume that shared keys that are revealed have to be revealed from the very beginning. This hypothesis seems indeed reasonable since the purpose of a public key is in general to be disclosed at the beginning, or eventually never revealed to an outsider.

Note that the previous example is not a counter-example anymore if we analyse the equivalence expressing the anonymity of $[P'_i]_a$ assuming that $\mathsf{pk}(sk_S)$ is known by the attacker from the beginning. The fact that $\mathsf{pk}(sk_S)$ is revealed during the execution of Q^{pk} will not give any additional power to the attacker. However, the resulting equivalence still does not hold but essentially because the equivalence expressing the anonymity of P^{pk} (without the presence of Q) does not hold anymore in this setting.

Example 5.18. *We consider again the process* P_i *as presented in Example 5.1 with an additional output to reveal the public key* $\mathsf{pk}(sk_S)$ *at the very beginning. Basically, we consider the well-tagged process* $P''_i \stackrel{def}{=} \mathsf{out}(c, \mathsf{pk}(sk_S)).[P_i]_a$.

We have that $C[P''_1] \approx_t C[P''_2]$ *with* $C[\_] = \nu sk_S.\_$. *Now, the presence of* Q^{pk} *will not prevent this equivalence to hold. Indeed, we have that:*

$$C[P''_1 \mid Q^{\mathsf{pk}}] \approx C[P''_2 \mid Q^{\mathsf{pk}}].$$

This hypothesis that states that shared keys are either known from the beginning or never revealed during the execution of the protocol is reasonable, and seems to be sufficient to establish a composition result. However, this complicates a bit the setting. In particular, as illustrated in Example 5.19, there is no hope to obtain a result as the one stated in Theorem 5.1. The situation where the processes share some keys is not equivalent in this setting to the situation where the processes do not share any key.

Example 5.19. *Consider the processes* P''_i *and* Q^{pk} *used in Example 5.18. We have seen that composition works under the composition context* $C = \nu sk_S.\_$. *However, we have that (*$i \in \{1, 2\}$*):*

$$C[P''_i \mid Q^{\mathsf{pk}}] \not\approx_t C[P''_i] \mid C[Q^{\mathsf{pk}}].$$

Indeed, on the left-hand side, the same public-key will be output twice whereas the process on the right-hand side will emit two different public keys. The attacker will observe such a difference. The strong result stated in Theorem 5.1 allowing us to easily make the link between the joint state case and the disjoint case does not hold anymore.

The problems encountered for composing processes that reveal shared keys are due to the fact that we do not want to tag the function symbols pk and vk that are used to model asymmetric keys: such a tagging scheme would lead us to an unrealistic modelling of asymmetric keys.

5.4.2 Roadmap of the proof

We now consider public keys and verifications keys that can be made public from the beginning through an initial frame Φ_0 that will represent the initial knowledge of the attacker. As illustrated in Subsection 5.4.1, we cannot rely on Theorem 5.1 anymore to establish our composition result. We will still go back to the disjoint case but we have to explain how a trace corresponding to the situation where processes share some keys is transformed and mapped to a trace that models the disjoint case. We cannot simply consider the identity transformation as it was done to establish the previous result. The sets of traces issued from both situations are not the same anymore.

Consider the initial frame $\Phi_0 = \{ax_1 \rhd \mathsf{f}_1(k_1), \ldots, ax_n \rhd \mathsf{f}_n(k_n)\}$ and initial set of private names $\mathcal{K}_0$ such that for all $i \in \{1, \ldots, n\}$, $\mathsf{f}_i \in \{\mathsf{pk}, \mathsf{vk}\}$ and $k_i \in \mathcal{K}_0$. Consider $\mathcal{K}'_0$ the subset of $\mathcal{K}_0$ that regroups all shared keys of $\mathcal{K}_0$ between $\overline{P_A}, \overline{P_B}, \overline{P'_A}$ and $\overline{P'_B}$. The result we want to show in this section is similar to Corollary 5.3, *i.e.*

$$\frac{(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a]; \Phi_0) \quad (\mathcal{K}_0; C[[\overline{P_B}]_b]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_B}]_b]; \Phi_0)}{(\mathcal{K}_0; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]; \Phi_0) \;\approx\; (\mathcal{K}_0; C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b]; \Phi_0)}$$

Of course, as Corollary 5.3, we impose some additional conditions on our processes. The most important one is that $(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_0)$ and $(\mathcal{K}_0; C[[\overline{P_B}]_b]; \Phi_0)$ (resp. $(\mathcal{K}_0; C[[\overline{P'_A}]_a]; \Phi_0)$, and $(\mathcal{K}_0; C[[\overline{P'_B}]_b]; \Phi_0)$) do not reveal any key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \in \mathcal{K}'_0\}$ unless the key occurs explicitly in Φ_0; and do not reveal any shared key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k$ occurs in C (resp. $C')\}$. Hence, the only shared public or verification keys that can be revealed to the intruder must be in Φ_0.

The first step to prove the result is similar to the proof of Corollary 5.3, *i.e.* we will rely on the fact that the trace equivalence is closed under disjoint parallel composition. Moreover, we rename the restricted names in order to avoid clashes between the two sets of restricted names (possible since the trace equivalence is closed under renaming of restricted names). Hence if we denote ρ_0 a renaming on names such that $\mathrm{dom}(\rho_0)$ are fresh names and $\mathrm{img}(\rho_0) = \mathcal{K}'_0$, the "disjoint case" would be represented by the following equivalence:

$$(\mathcal{K}; \{C[[\overline{P_A}]_a] \mid C[[\overline{P_B}]_b]\rho_0^{-1}\}; \Phi'_0) \approx_t (\mathcal{K}; \{C[[\overline{P'_A}]_a] \mid C[[\overline{P'_B}]_b]\rho_0^{-1}\}; \Phi'_0)$$

where $\mathcal{K} = \mathcal{K}_0 \cup \mathcal{K}_0\rho_0^{-1}$ and $\Phi'_0 = \Phi_0 \uplus \Phi_0\rho_0^{-1}$.

Note that in $\Phi_0 \uplus \Phi_0\rho_0^{-1}$, the parameters of $\Phi_0\rho_0^{-1}$ should be renamed so that the parameters of Φ'_0 constitute a sequence. However, to facilitate the comprehension, we will denote $\Phi_a = \{ax_1^a \rhd \mathsf{f}_1(k_1), \ldots, ax_n^a \rhd \mathsf{f}_n(k_n)\}$ and $\Phi_b = \{ax_1^b \rhd \mathsf{f}_1(k_1), \ldots, ax_n^b \rhd \mathsf{f}_n(k_n)\}$ where Φ_a (resp. Φ_b) represents the initial frame of $C[\overline{P_A}]$ and $C[\overline{P'_A}]$ (resp. $C[\overline{P_B}]$ and $C[\overline{P'_B}]$). Hence, we have $\Phi'_0 = \Phi_a \uplus \Phi_b\rho_0^{-1}$

The main problem with this equivalence is that given a trace of $(\mathcal{K}; \{C[[\overline{P_A}]_a]; \Phi'_0)$, we have no guarantee that the actions in the trace which are due to the process $C[[\overline{P_A}]_a]$ (resp. $C[[\overline{P'_B}]_b]\rho_0^{-1}$), would be matched with some actions on the process $C[[\overline{P'_A}]_a]$ (resp. $C[[\overline{P_B}]_b]\rho_0^{-1}$). To solve this issue, we will rely on the fact that the trace equivalence is also closed under renaming of public names. Indeed, if we denote $\rho_{\mathcal{C}h_a}$ a renaming of channel names such that $\mathrm{dom}(\rho_{\mathcal{C}h_a}) = \mathit{fnames}(\overline{P_A}, \overline{P'_A}) \cap \mathcal{C}h$ and $\mathrm{img}(\rho_{\mathcal{C}h_a})$ are fresh channel names , we have that $(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_a) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a]; \Phi_a)$ implies $(\mathcal{K}_0; C[[\overline{P_A}]_a]\rho_{\mathcal{C}h_a}; \Phi_a) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a]\rho_{\mathcal{C}h_a}; \Phi_a)$.

If we denote $\rho_{\mathcal{C}h_b}$ a fresh renaming of public channel names of $\overline{P_B}$ and $\overline{P'_B}$, the "disjoint case" will in fact be represented by the following equivalence:

$$(\mathcal{K}; \{C[[\overline{P_A}]_a]\rho_{\mathcal{C}h_a} \mid C[[\overline{P_B}]_b]\rho_0^{-1}\rho_{\mathcal{C}h_b}\}; \Phi'_0) \approx_t (\mathcal{K}; \{C[[\overline{P'_A}]_a]\rho_{\mathcal{C}h_a} \mid C[[\overline{P'_B}]_b]\rho_0^{-1}\rho_{\mathcal{C}h_b}\}; \Phi'_0)$$

where $\mathcal{K} = \mathcal{K}_0 \cup \mathcal{K}_0\rho_0^{-1}$ and $\Phi'_0 = \Phi_a \uplus \Phi_b\rho_0^{-1}$. With such equivalence, an action labeled for example $in(c, M)$ with $c \in \mathrm{img}(\rho_{\mathcal{C}h_a})$ would automatically be match by some input in $\overline{P_A}$ or $\overline{P'_A}$.

From there, the proof of our main result will be similar to Section 5.3. Typically, given a trace (tr, Φ) of $(\mathcal{K}_0; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b; \Phi_0)$, we will start by unfolding the process according the given trace (see Lemma 5.12). Then using a soundness lemma (see Lemma 5.14), we will show that this trace can be matched to a similar trace (tr', Φ') in the disjoint case, *i.e.* $(\mathcal{K}; \{C[[\overline{P_A}]_a]\rho_{\mathcal{C}h_a} \mid C[[\overline{P_B}]_b]\rho_0^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_0')$. Thanks to the equivalence representing the "disjoint case", we will find a matching trace (tr', Φ'') in $(\mathcal{K}; \{C[[\overline{P'_A}]_a]\rho_{\mathcal{C}h_a} \mid C[[\overline{P'_B}]_b]\rho_0^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_0')$. At last, using our completeness lemma (see Lemma 5.15), we will show that we can derive from (tr', Φ'') a trace (tr, Φ''') of $(\mathcal{K}; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b; \Phi_0)$ such that the frames Φ''' and Φ are statically equivalent.

We first state the lemma that unfolds the processes according to a specific trace.

Lemma 5.12. *Let C be a composition context. Let $\overline{P_A}$ (resp. $\overline{P_B}$) be a sequences of plain processes built on $\mathcal{F}_a \cup \mathcal{F}_0$ (resp. $\mathcal{F}_b \cup \mathcal{F}_0$) such that $fnames(\overline{P_a}) \cap \mathcal{C}h \subseteq \mathrm{dom}(\rho_{\mathcal{C}h_a})$ and $fnames(\overline{P_b}) \cap \mathcal{C}h \subseteq \mathrm{dom}(\rho_{\mathcal{C}h_b})$. Furthermore, we assume that $\mathrm{dom}(\rho_0)$, $\mathrm{img}(\rho_{\mathcal{C}h_a})$ and $\mathrm{img}(\rho_{\mathcal{C}h_a})$ do not occur in C, $\overline{P_A}$ and $\overline{P_B}$. At last, we assume that the shared private names between $\overline{P_A}$ and $\overline{P_B}$ are included in* $\mathrm{img}(\rho_0)$, i.e. $\mathcal{K}_0 \cap names(\overline{P_A}) \cap names(\overline{P_B}) \subseteq \mathrm{img}(\rho_0)$.

Let $D = (\mathcal{K}_0 \cup \mathcal{K}_0\rho_0^{-1}; \{C[[\overline{P_A}]_a\rho_{\mathcal{C}h_a}] \mid C[[\overline{P_B}]_b\rho_0^{-1}\rho_{\mathcal{C}h_b}]\}; \Phi_a \uplus \Phi_b\rho_0^{-1})$ and $S = (\mathcal{K}_0; \{C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]\}; \Phi_a \uplus \Phi_b)$, we have that for all $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(D)$ (resp. $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(S)$), there exists a renaming ρ and two bounded intermediate processes $S' = (\mathcal{E}_0; \{P_a, P_b\}; \Phi_a \uplus \Phi_b)$ and $D' = (\mathcal{E}_0; \{P_a\rho_{\mathcal{C}h_a}, P_b\rho^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_a \uplus \Phi_b\rho^{-1})$ such that

- *$\rho_{|\mathrm{dom}(\rho_0)} = \rho_0$, $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_0$ and $\mathrm{dom}(\rho)$ does not appear in $\{P_a, P_b\}$*
- *for all $i \in \{a, b\}$, there exists P_i' built on $\mathcal{F}_i \cup \mathcal{F}_0$ such that such that $P_i = [P_i']_i$*
- *there exists Φ' such that $(\mathsf{tr}, \nu\mathcal{E}_0.\Phi') \in \mathsf{trace}(D')$ (resp. $\mathsf{trace}(S')$) and $\nu\mathcal{E}_0.\Phi' \sim \nu\mathcal{E}.\Phi$*
- *for all $(\mathsf{tr}', \nu\mathcal{E}_0.\Phi'') \in \mathsf{trace}(S')$ (resp $\mathsf{trace}(D')$), there exists $\nu\mathcal{E}'.\Phi'$ such that $(\nu\mathcal{E}'.\Phi') \in \mathsf{trace}(S)$ (resp $\mathsf{trace}(D)$) and $\nu\mathcal{E}'.\Phi' \sim \nu\mathcal{E}_0.\Phi''$*

Note that $\delta_a^\rho(P_a) = P_a$, $\delta_b^\rho(P_b) = P_b\rho^{-1}$, $\delta_a^\rho(\Phi_a) = \Phi_a$ and $\delta_b^\rho(\Phi_b) = \Phi_b\rho^{-1}$

Proof. The proof is very similar to the proof of Lemma 5.8. Assume w.l.o.g that $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(S)$. Hence, we have that $S \overset{\mathsf{tr}}{\Rightarrow} (\mathcal{E}; \mathcal{P}; \Phi)$ for some $\mathcal{P}$. Thus, there exists a word w such that $S \xrightarrow{w} (\mathcal{E}; \mathcal{P}; \Phi)$ where $\mathsf{tr} = w \setminus \tau$. Let N be the number of τ action corresponding to the application of the rule $\textsc{Repl}_i$.

Let C' be the composition context without replication, $\overline{P'_A}$ and $\overline{P'_B}$ the sequences of plain process without replication such that $C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b]$ corresponds to $C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]$ where we replaced every instance $!P$ by $P_1 \mid \ldots \mid P_N$ for some where P_i is a renaming of P.

Let P_a be the plain process $C'[[\overline{P'_A}]_a]$ where we removed all name restriction and let P_b be the plain process $C'[[\overline{P'_B}]_b]$ where we removed all name restriction. At last, let ρ be a renaming such that $\rho|_{\mathrm{dom}(\rho_0)} = \rho_0$, $\mathrm{dom}(\rho) \setminus \mathrm{dom}(\rho_0)$ are fresh names and $\mathrm{img}(\rho) = (bnames(C') \cap fnames(\overline{P'_A}) \cap fnames(\overline{P'_B})) \cup \mathrm{img}(\rho_0)$. At last, let $\mathcal{E}_0 = \mathcal{K}_0 \cup bnames(C') \cup bnames(\overline{P'_A}) \cup bnames(\overline{P'_B}) \cup \mathrm{dom}(\rho)$.

Let $S' = (\mathcal{E}_0; \{P_a, P_b\}; \Phi_a \uplus \Phi_b)$ and $D' = (\mathcal{E}_0, \{P_a\rho_{\mathcal{C}h_a}, P_b\rho^{-1}\rho_{\mathcal{C}h_b}\}, \Phi_a \uplus \Phi_b\rho_0^{-1})$. We have that S' is a bounded intermediate process associated to $(\mathcal{K}_0, \{C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b]\}, \Phi_a \uplus \Phi_b)$.

Furthermore, we know that $names(\Phi_b) \subseteq \mathcal{K}_0$. Hence we have that $\Phi_b\rho_0^{-1} = \Phi_b\rho^{-1}$. Moreover, since $\mathrm{dom}(\rho) \setminus \mathrm{dom}(\rho_0)$ is composed of fresh names and $\mathrm{dom}(\rho_0)$ do not occur in C, $\overline{P_A}$ and $\overline{P_B}$, then $\mathrm{dom}(\rho)$ do not occur in C', $\overline{P'_A}$ and $\overline{P'_B}$. Hence $P_b\rho^{-1}$ is a renaming of $C'[[\overline{P'_B}]_b]$ and so $(\mathcal{E}_0; \{P_a, P_b\rho^{-1}\}; \Phi_a \cup \Phi_b\rho^{-1})$ is a bounded intermediate process associated to $(\mathcal{K}_0 \cup \mathcal{K}_0\rho_0^{-1}, \{C'[[\overline{P'_A}]_a] \mid C'[[\overline{P'_B}]_b]\}, \Phi_a \cup \Phi_b\rho_0^{-1})$. But $\rho_{\mathcal{C}h_a}$ and $\rho_{\mathcal{C}h_b}$ are both renaming on public channel and $\mathrm{img}(\rho_{\mathcal{C}h_a})$, $\mathrm{img}(\rho_{\mathcal{C}h_b})$ do not occur in C', $\overline{P'_A}$ and $\overline{P'_B}$. Hence, we deduce that $D' = (\mathcal{E}_0; \{P_a\rho_{\mathcal{C}h_a}, P_b\rho^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_a \cup \Phi_b\rho^{-1})$ is a bounded intermediate process associated to D. We conclude by applying Lemma 5.7. □

5.4.3 Static equivalence

We will assume, as in Section 5.3, that processes and frames are coloured by a or b. Furthermore assume that Φ_a (resp. Φ_b) is coloured by a (resp. b).

In Section 5.3.3.1, we gave a lemma which stated that given a well-tagged frame Φ, and a recipe M of the intruder, $\delta_i^\rho(M\Phi\downarrow) = M\delta^\rho(\Phi)\downarrow$ for all $i \in \{a, b\}$, provided that the frame did not reveal any shared key including the public and verification key. This is not true anymore when some public or verification key are revealed.

Example 5.20. *Consider the initial frame $\Phi = \Phi_a \uplus \Phi_b$. We have $\delta^\rho(\Phi_a \uplus \Phi_b) = \Phi_a \uplus \Phi_b\rho^{-1}$. More specifically, $ax_1^a\Phi = ax_1^b\Phi = \mathsf{f}(k)$ for some $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ and $k \in \mathrm{img}(\rho)$. While we still have $\delta_a^\rho(ax_1^a\Phi\downarrow) = ax_1^a\delta^\rho(\Phi)\downarrow$ and $\delta_b^\rho(ax_1^b\Phi\downarrow) = ax_1^b\delta^\rho(\Phi)\downarrow$, we have on the other hand:*

- $\delta_b^\rho(ax_1^a\Phi\downarrow) = \delta_b^\rho(\mathsf{f}(k)) = \mathsf{f}(k)\rho^{-1}$ *and* $ax_1^a\delta^\rho(\Phi)\downarrow = \mathsf{f}(k)$.
- $\delta_a^\rho(ax_1^b\Phi\downarrow) = \delta_a^\rho(\mathsf{f}(k)) = \mathsf{f}(k)$ *and* $ax_a^b\delta^\rho(\Phi)\downarrow = \mathsf{f}(k)\rho^{-1}$.

This counter example given in Example 5.20 is very simple but the main idea is that the application of δ^ρ on a frame Φ depends on how the frame is coloured. Hence, if a public key or verification key is accessible on a frame element of Φ, the application of δ_a^ρ and δ_b^ρ on this frame element will not produce the same result since the public key and verification keys are not tagged.

Example 5.21. *Consider $\mathcal{E} = \{k_1, k_2, k_1', k_2'\}$ and the renaming $\rho = \{k_1' \mapsto k_1, k_2' \mapsto k_2\}$. Consider the frame $\Phi = \Phi_a \uplus \Phi_b \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle\}$ where $\Phi_a = \{ax_1^a \rhd \mathsf{pk}(k_1), ax_2^a \rhd \mathsf{pk}(k_2)\}$ and $\Phi_b = \{ax_1^b \rhd \mathsf{pk}(k_1), ax_2^b \rhd \mathsf{pk}(k_2)\}$. At last, assume that ax_5 is coloured by a and ax_6 is coloured by b. Let $M = \langle ax_5, ax_6\rangle$. We have that*

- $M\Phi\downarrow = \langle\langle \mathsf{pk}(k_1), id_1\rangle, \langle id_2, \mathsf{pk}(k_2)\rangle\rangle$;
- $\delta_a^\rho(M\Phi\downarrow) = M\Phi\downarrow$;
- $\delta_b^\rho(M\Phi\downarrow) = \langle\langle \mathsf{pk}(k_1'), id_1\rangle, \langle id_2, \mathsf{pk}(k_2')\rangle\rangle$;
- $\delta^\rho(\Phi) = \Phi_a \uplus \Phi_b\rho^{-1} \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k_2')\rangle\}$

While $M\delta^\rho(\Phi)\downarrow$ is not equal to both $\delta_a^\rho(M\Phi\downarrow)$ and $\delta_b^\rho(M\Phi\downarrow)$, we can still find new recipes that will link the results.

The idea is to extract from the recipe M the part that use the public key revealed (i.e. contained in Φ_a and Φ_b) and use the corresponding recipe of Φ_a or Φ_b depending on which δ_i^ρ we have to apply. Indeed, note that $ax_5\Phi\downarrow = \langle ax_1^b, \mathsf{proj}_2(ax_5)\rangle\Phi\downarrow$. Similarly, we have $ax_6\Phi\downarrow = \langle \mathsf{proj}_1(ax_6), ax_2^a\rangle\Phi\downarrow$.

Consider the recipes $M_a = \langle ax_5, \langle \mathsf{proj}_1(ax_6), ax_2^a\rangle\rangle$ and $M_b = \langle\langle ax_1^b, \mathsf{proj}_2(ax_5)\rangle, ax_6\rangle$:

- $M_a\delta^\rho(\Phi)\downarrow = \langle\langle \mathsf{pk}(k_1), id_1\rangle, \langle id_2, \mathsf{pk}(k_2)\rangle\rangle = \delta_a^\rho(M\Phi\downarrow)$
- $M_b\delta^\rho(\Phi)\downarrow = \langle\langle \mathsf{pk}(k_1'), id_1\rangle, \langle id_2, \mathsf{pk}(k_2')\rangle\rangle = \delta_b^\rho(M\Phi\downarrow)$

Following Example 5.21, we are now able to show the following result (Proof in Appendix B.3):

Lemma 5.13. *Let Φ and Φ' two frames in normal form such that $\mathrm{dom}(\Phi) = \mathrm{dom}(\Phi')$. Assume that Φ and Φ' have the same colors, i.e. for all $(ax \rhd u) \in \Phi$, for all $(ax' \rhd u') \in \Phi'$, $ax = ax'$ implies $\mathsf{col}(ax \rhd u) = \mathsf{col}(ax' \rhd u')$.*

Let $\mathcal{E}$ be a set of names and let ρ a renaming such that $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$ and $\mathrm{dom}(\rho) \cap fnames(\Phi, \Phi') = \emptyset$. Let's denote $\Phi_+ = \Phi_a \uplus \Phi_b \uplus \Phi$ and $\Phi'_+ = \Phi_a \uplus \Phi_b \uplus \Phi'$.

If the following properties are satisfied:

- *$\nu\mathcal{E}.\Phi_+$, $\nu\mathcal{E}.\Phi'_+$ are well-tagged*
- *$\nu\mathcal{E}.\delta^\rho(\Phi_+) \sim \nu\mathcal{E}.\delta^\rho(\Phi'_+)$*
- *for all $u \in \{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$, $\nu\mathcal{E}.\delta^\rho(\Phi_+) \vdash u$ or $\nu\mathcal{E}.\delta^\rho(\Phi'_+) \vdash u$ implies that $u \in \mathrm{img}(\delta^\rho(\Phi_a \uplus \Phi_b))$*

then for all M such that $fnames(M) \cap \mathcal{E} = \emptyset$ and $fvars(M) \subseteq \mathrm{dom}(\Phi_+)$, there exists M_a and M_b such that $fnames(M_a, M_b) \cap \mathcal{E} = \emptyset$, $fvars(M_a, M_b) \subseteq \mathrm{dom}(\Phi_+)$ and:

1. *$\delta_a^\rho(M\Phi_+\downarrow) = M_a\delta^\rho(\Phi_+)\downarrow$ and $\delta_a^\rho(M\Phi'_+\downarrow) = M_a\delta^\rho(\Phi'_+)\downarrow$*
2. *$\delta_b^\rho(M\Phi_+\downarrow) = M_b\delta^\rho(\Phi_+)\downarrow$ and $\delta_b^\rho(M\Phi'_+\downarrow) = M_b\delta^\rho(\Phi'_+)\downarrow$*

Similarly to Lemma 5.9, Lemma 5.13 induces a corollary that will help us prove the static equivalence between traces in the shared case.

Corollary 5.4. *Let Φ and Φ' two frames in normal form such that* $\mathrm{dom}(\Phi) = \mathrm{dom}(\Phi')$. *Assume that Φ and Φ' have the same colors. Let $\mathcal{E}$ be a set of names and let ρ a renaming such that* $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$, $\mathrm{dom}(\rho) \cap \mathit{fnames}(\Phi, \Phi') = \emptyset$ *and* $\rho_{|\,\mathrm{dom}(\rho_0)} = \rho_0$. *Let's denote* $\Phi_+ = \Phi_a \uplus \Phi_b \uplus \Phi$ *and* $\Phi'_+ = \Phi_a \uplus \Phi_b \uplus \Phi'$.

If the following properties are satisfied:

— $\nu\mathcal{E}.\Phi_+$, $\nu\mathcal{E}.\Phi'_+$ *are well-tagged,*

— $\nu\mathcal{E}.\delta^\rho(\Phi_+) \sim \nu\mathcal{E}.\delta^\rho(\Phi'_+)$,

— *for all* $u \in \{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$, $\nu\mathcal{E}.\delta^\rho(\Phi_+) \vdash u$ *or* $\nu\mathcal{E}.\delta^\rho(\Phi'_+) \vdash u$ *imply that* $u \in \mathit{fnames}(\delta^\rho(\Phi_a \uplus \Phi_b))$

then $\nu\mathcal{E}.\Phi_+ \sim \nu\mathcal{E}.\Phi'_+$.

Proof. Let M, N such that $\mathit{fnames}(M, N) \cap \mathcal{E} = \emptyset$ and $\mathit{fvars}(M, N) \subseteq \mathrm{dom}(\Phi_+)$. Assume that $M\Phi_+{\downarrow} = N\Phi_+{\downarrow}$. Thanks to Lemma 5.4, we know that $M\Phi_+{\downarrow} = N\Phi_+{\downarrow}$ is equivalent to $\delta^\rho_a(M\Phi_+{\downarrow}) = \delta^\rho_a(N\Phi_+{\downarrow})$. But thanks to Lemma 5.13, we know that there exists two recipe M_a and N_a such that :

— $\delta^\rho_a(M\Phi_+{\downarrow}) = M_a\delta^\rho(\Phi_+){\downarrow}$ and $\delta^\rho_a(M\Phi'_+{\downarrow}) = M_a\delta^\rho(\Phi'_+){\downarrow}$

— $\delta^\rho_a(N\Phi_+{\downarrow}) = N_a\delta^\rho(\Phi_+){\downarrow}$ and $\delta^\rho_a(N\Phi'_+{\downarrow}) = N_a\delta^\rho(\Phi'_+){\downarrow}$

Hence we have $\delta^\rho_a(M\Phi_+{\downarrow}) = \delta^\rho_a(N\Phi_+{\downarrow})$ is equivalent to $M_a\delta^\rho(\Phi_+){\downarrow} = N_a\delta^\rho(\Phi_+){\downarrow}$. By hypothesis, we know that $\nu\mathcal{E}.\delta^\rho(\Phi_+) \sim \nu\mathcal{E}.\delta^\rho(\Phi'_+)$, hence $M_a\delta^\rho(\Phi_+){\downarrow} = N_a\delta^\rho(\Phi_+){\downarrow}$ is equivalent to $M_a\delta^\rho(\Phi'_+){\downarrow} = N_a\delta^\rho(\Phi'_+){\downarrow}$. Therefore we deduce that it is also equivalent to $\delta^\rho_a(M\Phi'_+{\downarrow}) = \delta^\rho_a(N\Phi'_+{\downarrow})$. At last, once again thanks to Lemma 5.4, we have that $\delta^\rho_a(M\Phi'_+{\downarrow}) = \delta^\rho_a(N\Phi'_+{\downarrow})$ is equivalent to $M\Phi'_+{\downarrow} = N\Phi'_+{\downarrow}$ and so we conclude that $M\Phi_+{\downarrow} = N\Phi_+{\downarrow}$ is equivalent to $M\Phi'_+{\downarrow} = N\Phi'_+{\downarrow}$. □

5.4.4 Soundness and completeness

For this subsection, we consider $\mathcal{E}_0$ a set of private names and P_a, P_b two processes without name restriction or replication, and coloured respectively by a and b. At last consider ρ a renaming such that:

— $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_0$ and $\mathrm{dom}(\rho)$ does not appear in P_a, P_b, Φ_a or Φ_b.

— for all $i \in \{a, b\}$, there exists P'_i built on $\mathcal{F}_i \cup \mathcal{F}_0$ such that $P_i = [P'_i]_i$.

— for all $i \in \{a, b\}$, $\mathit{fnames}(P_i) \cap \mathcal{C}h \subseteq \mathrm{dom}(\rho_{\mathcal{C}h_i})$ and $\mathrm{img}(\rho_{\mathcal{C}h_i})$ does not appear in P_a or P_b.

We will denote $D = (\mathcal{E}_0; \{P_a\rho_{\mathcal{C}h_a}, P_b\rho^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_a \uplus \Phi_b\rho^{-1})$ and $S = (\mathcal{E}_0; \{P_a, P_b\}; \Phi_a \uplus \Phi_b)$. We assume that P_a and P_b do not share any private channel name, *i.e.* $\mathit{fnames}(P_a) \cap \mathit{fnames}(P_b) \cap \mathcal{C}h \cap \mathcal{E}_0 = \emptyset$, and do not use variable of channel type.

For our soundness and completeness, we want to preserve a similar invariant as Lemmas 5.10 and 5.11. Typically, for a derivation $D \overset{\mathsf{tr}}{\Rightarrow} (\mathcal{E}_0; \mathcal{P}_D; \Phi_D)$, we want to match it in the shared case to a derivation $S \to^* (\mathcal{E}_0; \mathcal{P}_S; \Phi_S)$ where $\Phi_D{\downarrow} = \delta^\rho(\Phi_S{\downarrow})$, $\mathcal{P}_S = \mathcal{P}_0\alpha$ and $\mathcal{P}_D{\downarrow} = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha{\downarrow}){\downarrow}$, for some original well-tagged multi-set of processes $(\mathcal{P}_0, \alpha)$.

However, contrary to Lemmas 5.10 and 5.11, we can't use the same sequence of actions tr. Typically, it is due to the fact that, as we saw in Subsection 5.4.3, given a recipe M and a frame Φ that may reveal some verification or public keys, we may have $\delta^\rho_a(M\Phi{\downarrow}) \neq \delta^\rho_b(M\Phi{\downarrow})$. Hence the main idea is to transform the sequence of labels tr using Lemma 5.13.

Example 5.22. *Coming back to Example 5.21, consider the process* $P_a = \mathsf{out}(c, \langle \mathsf{pk}(k_1), id_1\rangle)$ *and* $P_b = \mathsf{out}(c, \langle id_2, \mathsf{pk}(k_2)\rangle).\mathsf{in}(c, x).\mathsf{out}(c, \mathsf{aenc}(\mathsf{tag}_b(x), \mathsf{pk}(k_1)))$. *Furthermore, consider the set* $\mathcal{E}_0 = \{k_1, k_2, k'_1, k'_2\}$, $\rho = \{k'_1 \mapsto k_1, k'_2 \mapsto k_2\}$, $\rho_{\mathcal{C}h_a} = \{c \mapsto c_a\}$ *and* $\rho_{\mathcal{C}h_b} = \{c \mapsto c_b\}$. *At last consider the frames* $\Phi_a = \{ax^a_1 \rhd \mathsf{pk}(k_1), ax^a_2 \rhd \mathsf{pk}(k_2)\}$ *and* $\Phi_b = \{ax^b_1 \rhd \mathsf{pk}(k_1), ax^b_2 \rhd \mathsf{pk}(k_2)\}$ *coloured respectively with a and b. We denote* $\Phi_0 = \Phi_a \uplus \Phi_b$

Let's denote $S = (\mathcal{E}_0; \{P_a, P_b\}; \Phi_a \uplus \Phi_b)$ *and* $D = (\mathcal{E}_0; \{P_a\rho_{\mathcal{C}h_a}, P_b\rho^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_a \uplus \Phi_b\rho^{-1})$. *We have:*

$$\begin{array}{rcl} S & \xrightarrow{\nu ax_5.out(c,ax_5)} & (\mathcal{E}_0; \{P_b\}; \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle\}) \\ & \xrightarrow{\nu ax_6.out(c,ax_6)} & (\mathcal{E}_0; \{P'_b\}; \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle\}) \\ & \xrightarrow{in(c,M)} & (\mathcal{E}_0; \{P''_b\}; \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle\}) \\ & \xrightarrow{\nu ax_7.out(c,ax_7)} & (\mathcal{E}_0; \emptyset; \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle, ax_7 \rhd u\}) \end{array}$$

where $M = \langle ax_5, ax_6\rangle$, $u = \mathsf{aenc}(\mathsf{tag}_b(v), \mathsf{pk}(k_1))$ $P'_b = \mathsf{in}(c,x).\mathsf{out}(c, \mathsf{aenc}(\mathsf{tag}_b(x), \mathsf{pk}(k_1)))$, $P''_b = \mathsf{out}(c,u)$ *with* $v = \langle\langle \mathsf{pk}(k_1), id_1\rangle, \langle id_2, \mathsf{pk}(k_2)\rangle\rangle$.

In Example 5.21, we showed that $M_b\delta^\rho(\Phi)\downarrow = \delta^\rho_b(M\Phi\downarrow)$ *where* $M_b = \langle\langle ax^b_1, \mathsf{proj}_2(ax_5)\rangle, ax_6\rangle$ *and* $\Phi = \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle\}$. *Since the input corresponding to the label* $in(c,M)$ *corresponds to a process coloured by* b, *we will replace* M *by* M_b *in the sequence of action. Similarly, the output corresponding to the label* $\nu ax_5.out(c, ax_5)$ *corresponds to a process coloured by* a, *hence we will replace* c *by* $c\rho_{\mathcal{C}h_a} = c_a$. *Thus:*

$$\begin{array}{rcl} D & \xrightarrow{\nu ax_5.out(c_a,ax_5)} & (\mathcal{E}_0; \{\delta^\rho_b(P_b)\rho_{\mathcal{C}h_b}\}; \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle\}) \\ & \xrightarrow{\nu ax_6.out(c_b,ax_6)} & (\mathcal{E}_0; \{\delta^\rho_b(P'_b)\rho_{\mathcal{C}h_b}\}; \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k'_2)\rangle\}) \\ & \xrightarrow{in(c_b,M_b)} & (\mathcal{E}_0; \{\delta^\rho_b(P''_b)\rho_{\mathcal{C}h_b}\}; \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k'_2)\rangle\}) \\ & \xrightarrow{\nu ax_7.out(c_b,ax_7)} & (\mathcal{E}_0; \emptyset; \Phi_0 \uplus \{ax_5 \rhd \langle \mathsf{pk}(k_1), id_1\rangle, ax_6 \rhd \langle id_2, \mathsf{pk}(k'_2)\rangle, ax_7 \rhd u'\}) \end{array}$$

where $u' = \mathsf{aenc}(\mathsf{tag}_b(v'), \mathsf{pk}(k'_1))$ *with* $v' = \langle\langle \mathsf{pk}(k'_1), \mathsf{proj}_2(\langle \mathsf{pk}(k_1), id_1\rangle)\rangle, \langle id_2, \mathsf{pk}(k'_2)\rangle\rangle$. *Note that* $\delta^\rho_b(P'_b)\rho_{\mathcal{C}h_b} = \mathsf{in}(c,x).\mathsf{out}(c_b, \mathsf{aenc}(\mathsf{tag}_b(x), \mathsf{pk}(k'_1)))$ *and* $\delta^\rho_b(P''_b)\rho_{\mathcal{C}h_b} = \mathsf{out}(c_b, u')$.

Moreover, note that $v'\downarrow = \langle\langle \mathsf{pk}(k'_1), id_1\rangle, \langle id_2, \mathsf{pk}(k'_2)\rangle\rangle$ *and* $\delta^\rho_b(u\downarrow) = u'\downarrow$. *More generally, if we denote by* Φ_S *and* Φ_D *the last two frames of the derivations, we have* $\delta^\rho(\Phi_S\downarrow) = \Phi_D\downarrow$.

We formalise the transformation of sequence of labels (see Example 5.22) in the following lemmas (proofs in Appendix B.3):

Lemma 5.14 (Soundness)**.** *If for all* $(\mathsf{tr}, \Phi) \in \mathit{trace}(D)$, *for all* $u \in \{\mathsf{vk}(k), \mathsf{pk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$, $\nu\mathcal{E}_0.\Phi \vdash u$ *implies that* $u \in \mathrm{img}(\Phi_a \uplus \Phi_b\rho^{-1})$, *then for all* $S \xrightarrow{w} (\mathcal{E}_0; \mathcal{P}_S; \Phi_S)$ *such that* w *does not contain any* τ *action that corresponds to an internal communication between two processes of different colours, we have that* Φ_S *is well-tagged and there exists* $D \xrightarrow{w'} (\mathcal{E}; \mathcal{P}_D; \Phi_D)$ *such that* $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$.

Furthermore, if $w = \ell_1.\ldots.\ell_n$ *then* $w' = \ell'_1.\ldots.\ell'_n$ *such that for all* $k \in \{1, \ldots, n\}$,

- *if* $\ell_k = \nu ax_m.out(c, ax_m)$ *is an output coming from a process coloured by* $i \in \{a, b\}$, *then* $\ell'_k = \nu ax_m.\mathsf{out}(c\rho_{\mathcal{C}h_i}, ax_m)$
- *if* $\ell_k = in(c, M)$ *is an input coming from a process coloured by* $i \in \{a, b\}$, *then* $\ell'_k = in(c\rho_{\mathcal{C}h_i}, M_i)$ *with* $M_i\Phi_D\downarrow = \delta^\rho_i(M\Phi_S\downarrow)$.
- *if* $\ell_k = out(c, d)$ *is an output coming from a process coloured by* $i \in \{a, b\}$ *with* d *a channel name, then* $\ell'_k = \mathsf{out}(c\rho_{\mathcal{C}h_i}, d\rho_{\mathcal{C}h_i})$
- *if* $\ell_k = \nu ch_m.out(c, ch_m)$ *is an output coming from a process coloured by* $i \in \{a, b\}$ *with* ch_m *a channel name, then* $\ell'_k = \nu ch_m.out(c\rho_{\mathcal{C}h_i}, ch_m)$
- *if* $\ell_k = \tau$, *then* $\ell'_k = \tau$.

Note that in Lemma 5.14, we only consider traces that do not have internal communication between two processes of different colours. Indeed, for these internal communications, we will apply a specific transformation (see Example 5.23).

Lemma 5.15 (Completeness)**.** *Let* $D \xrightarrow{w} (\mathcal{E}_0; \mathcal{P}_D; \Phi_D)$. *Let* Φ_+ *be a ground well-tagged frame such that* $\Phi_+ = \Phi_a \uplus \Phi_b \uplus \Phi$ *for some* Φ, *and* Φ_+, Φ_D *have the same colours. If the following properties are satisfied:*

— $w = \ell_1.\ldots.\ell_n$

— $\nu\mathcal{E}_0.\Phi_D \sim \nu\mathcal{E}_0.\delta^\rho(\Phi_+)$

— *for all* $u \in \{k, \mathsf{vk}(k), \mathsf{pk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$, $\nu\mathcal{E}_0.\delta^\rho(\Phi_+) \vdash u$ *or* $\nu\mathcal{E}_0.\Phi_D \vdash u$ *implies that* $u \in \mathrm{img}(\Phi_a \uplus \Phi_b\rho^{-1})$

— *for all* $k \in \{1,\ldots,n\}$, *if* $\ell_k = \mathsf{in}(c, M_k)$ *with* $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a,b\}$ *then there exists* M_k^i *such that* $M_k\delta^\rho(\Phi_+\!\downarrow)\!\downarrow = \delta_i^\rho(M_k^i\Phi_+\!\downarrow)$.

then there exists a label $w' = \ell'_1.\ldots.\ell'_n$ *and a well-tagged frame* Φ_S *such that* $S \xrightarrow{w'} (\mathcal{E}; \mathcal{P}_S; \Phi_S)$, $\Phi_D\!\downarrow = \delta^\rho(\Phi_S\!\downarrow)$, *and for all* $k \in \{1,\ldots,n\}$,

— *if* $\ell_k = \nu ax.out(c, ax)$ *with* $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a,b\}$ *then* $\ell'_k = \nu ax.out(c\rho_{\mathcal{C}h_i}^{-1}, ax)$

— *if* $\ell_k = in(c, M_k)$ *with* $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a,b\}$, *then* $\ell'_k = in(c\rho_{\mathcal{C}h_i}^{-1}, M_k^i)$.

— *if* $\ell_k = out(c,d)$ *with* $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a,b\}$ *and* d *a channel name, then* $\ell'_k = out(c,d)\rho_{\mathcal{C}h_i}^{-1}$

— *if* $\ell_k = \nu ch_m.out(c, ch_m)$ *with* $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a,b\}$ *and* ch_m *a channel name, then* $\ell'_k = \nu ch_m.out(c\rho_{\mathcal{C}h_i}^{-1}, ch_m)$

— *if* $\ell_k = \tau$ *then* $\ell'_k = \tau$

5.4.5 Dealing with internal communication

As mentioned earlier, Lemma 5.14 focuses on traces of S that do not have internal communication between processes of different colours. More specifically, since in the intermediate process D, the processes coloured by a and b do not share any channel name, then an internal communication in a trace of S between processes of different colours could not be matched by an internal communication in a trace of D anymore. However, we know that the processes P_a and P_b do not share private channel name. Hence, if an internal communication occurs between two processes of different colours in a trace of S, then it implies that this internal communication was done over a public channel. This will allow us to modify the trace of S in order to remove such internal communication.

Example 5.23. *Coming back to Example 5.22, we have:*

$$\begin{array}{lll} S & \xrightarrow{\nu ax_5.out(c,ax_5)} & (\mathcal{E}_0; \{P_a, P'_b\}; \Phi_0 \uplus \{ax_5 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle\}) \\ & \xrightarrow{\tau} & (\mathcal{E}_0; \{\mathsf{out}(c, \mathsf{aenc}(\mathsf{tag}_b(u), \mathsf{pk}(k_1)))\}; \Phi_0 \uplus \{ax_5 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle\}) \end{array}$$

where $u = \langle \mathsf{pk}(k_1), id_1\rangle$. *To match this trace to a trace of* D, *we first have to remove the internal communication between* P_a *and* P'_b. *To do so, we replace the internal communication by the sequence of label* $\nu ax_6.out(c, ax_6).in(c, ax_6)$*:*

$$\begin{array}{lll} S & \xrightarrow{\nu ax_5.out(c,ax_5)} & (\mathcal{E}_0; \{P_a, P'_b\}; \Phi_0 \uplus \{ax_5 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle\}) \\ & \xrightarrow{\nu ax_6.out(c,ax_6)} & (\mathcal{E}_0; \{P'_b\}; \Phi_0 \uplus \{ax_5 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle, ax_6 \rhd u\}) \\ & \xrightarrow{in(c,ax_6)} & (\mathcal{E}_0; \{\mathsf{out}(c, \mathsf{aenc}(\mathsf{tag}_b(u), \mathsf{pk}(k_1)))\}; \Phi_0 \uplus \{ax_5 \rhd \langle id_2, \mathsf{pk}(k_2)\rangle, ax_6 \rhd u\}) \end{array}$$

Then we can apply the transformation described in Lemma 5.14 to obtain the corresponding trace of D*:*

$$\begin{array}{lll} D & \xrightarrow{\nu ax_5.out(c_b,ax_5)} & (\mathcal{E}_0; \{P_a, \delta_b^\rho(P'_b)\}; \Phi_0 \uplus \{ax_5 \rhd \langle id_2, \mathsf{pk}(k'_2)\rangle\}) \\ & \xrightarrow{\nu ax_6.out(c_a,ax_6)} & (\mathcal{E}_0; \{\delta_b^\rho(P'_b)\}; \Phi_0 \uplus \{ax_5 \rhd \langle id_2, \mathsf{pk}(k'_2)\rangle, ax_6 \rhd u\}) \\ & \xrightarrow{in(c_b,M)} & (\mathcal{E}_0; \{\mathsf{out}(c_b, \mathsf{aenc}(\mathsf{tag}_b(v), \mathsf{pk}(k'_1)))\}; \Phi_0 \uplus \{ax_5 \rhd \langle id_2, \mathsf{pk}(k'_2)\rangle, ax_6 \rhd u\}) \end{array}$$

where $M = \langle ax_1^b, \mathsf{proj}_2(ax_6)\rangle$ *and* $v = \langle \mathsf{pk}(k'_1), \mathsf{proj}_2(u)\rangle$. *Note that* $v\!\downarrow = \langle \mathsf{pk}(k'_1), id_1\rangle = \delta_b^\rho(u\!\downarrow)$.

However, it remains one final issue with the internal communication. As mentioned in Subection 5.4.2, the proof technique for our main result consists of taking a trace (tr, Φ) of $(\mathcal{K}_0; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b; \Phi_0)$ (the shared case), matching it with a trace (tr', Φ') of the disjoint case, *i.e.* the process $(\mathcal{K}'_0; \{C[[\overline{P_A}]_a]\rho_{\mathcal{C}h_a} \mid C[[\overline{P_B}]_b]\rho_0^{-1}\rho_{\mathcal{C}h_b}\}; \Phi'_0)$. Then, thanks to the equivalence representing the "disjoint case", we consider the trace (tr', Φ'') in $(\mathcal{K}'_0; \{C[[\overline{P'_A}]_a]\rho_{\mathcal{C}h_a} \mid C[[\overline{P'_B}]_b]\rho_0^{-1}\rho_{\mathcal{C}h_b}\}; \Phi'_0)$ such that $\Phi'' \sim \Phi'$, and at last we apply the completeness lemma in order to come back to a trace (tr, Φ''') of the shared case, *i.e.* $(\mathcal{K}_0; C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b; \Phi_0)$.

As we saw in this subsection, the trace internal communication between processes of different colours in (tr, Φ) are replaced by a sequence of visible actions of the form $\nu ax.out(c, ax).in(c, M)$ in the trace (tr', Φ'). However, when considering the trace (tr', Φ'') obtained thanks to the equivalence representing the disjoint case, it is possible that some τ-actions are applied between the actions labeled $\nu ax.out(c, ax)$ and $in(c, M)$ in the trace (tr', Φ''). But the two actions must be applied in sequence in order to transform back the sequence of actions $\nu ax.out(c, ax).in(c, M)$ into an internal communication in the trace (tr, Φ'''). Hence, we show that it is possible to find a derivation of the trace (tr', Φ'') where the two actions $\nu ax.out(c, ax)$ and $in(c, M)$ are applied consecutively.

Lemma 5.16. *Let D' be an intermediate bounded process and w a sequence of labels such that $D \xrightarrow{w} D'$. If $w = w_1.w_2.w_3$ with $w_2 \in \nu ax.out(c_1, ax).\tau^*.in(c_2, M)$, $c_1 \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $c_2 \in \mathrm{img}(\rho_{\mathcal{C}h_j})$ and $i \neq j$, then there exist two sequences of label w', w'_2 such that $w' = w_1.w'_2.w_3$, $w'_2 \in \tau^*.\nu ax.out(c_1, ax).in(c_2, M).\tau^*$ and $D \xrightarrow{w'} D'$.*

Proof. $D \xrightarrow{w} D'$ and $w = w_1.w_2.w_3$ with $w_2 \in \nu ax.out(c_1, ax).\tau^*.in(c_2, M)$ imply that there exist $D_1, \ldots, D_n$ such that $D \xrightarrow{w_1} D_1 \xrightarrow{\nu ax.out(c_1,ax)} D_2 \xrightarrow{\tau} \ldots \xrightarrow{\tau} D_{n-1} \xrightarrow{in(c_2,M)} D_n$. By hypothesis on D, we know that processes of different colors in D do not share public nor private channels. Thus, a τ action, i.e. Else, Then and Comm, can only be applied on processes of same colour. But, applying a τ action on a process of colour i does not modify the processes whose colours are $j \neq i$.

Let $S_1 = (\mathcal{E}; \mathcal{P}_a \uplus \mathcal{P}_b; \Phi)$ be an intermediate bounded process such that for all $P \in \mathcal{P}_a$ (resp. $\mathcal{P}_b$), $\mathsf{col}(P) = a$ (resp. b). Assume that we apply a τ action, denoted τ_a, on $\mathcal{P}_a$, i.e. there exists $\mathcal{P}'_a$ such that $S_1 \xrightarrow{\tau_a} (\mathcal{E}; \mathcal{P}'_a \uplus \mathcal{P}_b; \Phi) = S_2$. Assume now that we apply a τ action, denoted τ_b, on $\mathcal{P}_b$, i.e. there exists $\mathcal{P}'_b$ such that $S_2 \xrightarrow{\tau_b} (\mathcal{E}; \mathcal{P}'_a \uplus \mathcal{P}'_b; \Phi) = S_3$. Hence $S_1 \xrightarrow{\tau_a} S_2 \xrightarrow{\tau_b} S_3$. But we can see that $S_1 \xrightarrow{\tau_b} (\mathcal{E}; \mathcal{P}_a \uplus \mathcal{P}'_b; \Phi) \xrightarrow{\tau_a} S_3$. Thus it is possible to swap τ actions.

With a similar proof, we can show that if $S_1 \xrightarrow{\nu ax.out(c_1,ax)} S_2 \xrightarrow{\tau_b} S_3$, with τ_b a τ action initiated by a process of colour b, then we also have that there exists S'_2 such that $S_1 \xrightarrow{\tau_b} S'_2 \xrightarrow{\nu ax.out(c_1,ax)} S_3$. Furthermore, once again with a similar proof, we can show that if $S_1 \xrightarrow{\tau_a} S_2 \xrightarrow{in(c_2,M)} S_3$, with τ_a a τ action initiated by a process of colour a, then there exists S'_2 such that $S_1 \xrightarrow{in(c_2,M)} S'_2 \xrightarrow{\tau_a} S_3$.

Therefore, a simple induction on the number of τ actions in the derivation $D \xrightarrow{w_1.w_2} D'$ allows us to prove that there exists $D'_2, \ldots, D'_{n-1}$ and an index $k \in \{1, \ldots, n-2\}$ such that $D_1 \xrightarrow{\tau_b} D'_2 \xrightarrow{\tau_b} \ldots \xrightarrow{\tau_b} D'_k \xrightarrow{\nu ax.out(c_1,ax)} D'_{k+1} \xrightarrow{in(c_2,M)} D'_{k+1} \xrightarrow{\tau_a} \ldots \xrightarrow{\tau_a} D_n$. Hence the result holds. □

5.4.6 Main composition result

We can now state the main composition result of this chapter (proof in Appendix B.3)

Theorem 5.2. *Let $\overline{P_A}, \overline{P'_A}$ (resp. $\overline{P_B}, \overline{P'_B}$) be two sequences of plain processes built $\mathcal{F}_a \cup \mathcal{F}_0$ (resp. $\mathcal{F}_b \cup \mathcal{F}_0$). Let $\mathcal{K}_0$ be a finite set of names of base type, and C and C' be two composition contexts. Let $\mathcal{K}'_0$ be a subset of $\mathcal{K}_0$ such that $\mathcal{K}'_0 = names(P_A, P'_A) \cap names(P_B, P'_B) \cap \mathcal{K}_0$. Let $\Phi_0 = \{ax_1 \rhd \mathsf{f}_1(k_1), \ldots, ax_n \rhd \mathsf{f}_n(k_n)\}$ with $\mathsf{f}_i \in \{\mathsf{pk}, \mathsf{vk}\}$, and $k_i \in \mathcal{K}_0$ for any $i \in \{1, \ldots, n\}$. Assume that the processes $(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_0)$ and $(\mathcal{K}_0; C[[\overline{P_B}]_b]; \Phi_0)$ (resp. $(\mathcal{K}_0; C[[\overline{P'_A}]_a]; \Phi_0)$, and $(\mathcal{K}_0; C[[\overline{P'_B}]_b]; \Phi_0)$):*

— *do not reveal any key in* $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \in \mathcal{K}'_0\}$ *unless if the key occurs explicitly in* Φ_0*; and*

— *do not reveal any shared key in* C *(resp.* C'*);*

Lastly, we assume that plain processes $\overline{P_A}, \overline{P'_A}$ *and* $\overline{P_B}, \overline{P'_B}$ *do not use variable of channel type. In such a case,*

$$\frac{(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a]; \Phi_0) \quad (\mathcal{K}_0; C[[\overline{P_B}]_b]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_B}]_b]; \Phi_0)}{(\mathcal{K}_0; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b]; \Phi_0)}$$

5.5 Application

In this section, we present an example of application of Theorem 5.2 using the e-passport protocol. In Section 3.4, we described that the anonymity of the parallel composition of the protocols AA and PA can be expressed with the following equivalence:

$$\nu sk_{DS}.C[PA \mid AA, PA_0 \mid AA_0] \approx_t \nu sk_{DS}.C'[PA \mid AA]$$

where C and C' are contexts defined as follows:

$$C[\__1, \__2] \stackrel{def}{=} \; ! \; \nu sk_P.\ \nu id.\ \nu sig.\ \nu pic.\ \ldots \; ! \; \nu ksenc.\ \nu ksmac.\ \__1 \mid \nu sk_{P0}.\ !\nu ksenc.\ \nu ksmac.\ \__2$$

$$C'[\_] \stackrel{def}{=} !\nu sk_P.\ \nu id.\ \nu sig.\ \nu pic.\ \ldots ! \ \nu ksenc.\ \nu ksmac.\ \_$$

Using the ProVerif tool [BAF08], we tried to prove these two equivalences, but it failed to terminate. The problem comes from the PA protocol. Indeed, while ProVerif fails to prove that PA satisfies anonymity and unlinkability (ProVerif does not terminate), it can perfectly prove that AA satisfies these two properties.

In order to exploit the fact that ProVerif is able to prove anonymity of the AA protocol, we show how we can apply Theorem 5.2 on the e-passport protocol. More specifically, we use Theorem 5.2 to show that the anonymity (resp. unlinkability) of the protocol PA with tags implies the anonymity (resp. unlinkability) of the parallel composition of the protocols AA and PA with tags to the problem of anonymity of PA with tags.

First of all, note that both protocols PA and AA use the primitive $\mathsf{mac}/2$ that can be modelled using our hash function h and the pairing, *i.e.* $\mathsf{mac}(u, k)$ is modelled by $\mathsf{h}(\langle u, k\rangle)$. With this modelisation, we have that both protocols PA and AA are built over $\mathcal{F}_0$ which means that it is possible to tag these protocols. We will tag PA with a and AA with b. Note that the protocols PA and AA do not have private channel nor variable of channel type.

Secondly, the key sk_{DS} corresponds to the private signing key of a signing authority. Hence, we will assume that the intruder knows the verification key $\mathsf{vk}(sk_{DS})$. Moreover, for the victim's e-passport, we might consider that the intruder have access to the verification key $\mathsf{vk}(sk_{P0})$. Hence, using intermediate processes, the anonymity of the parallel composition of the protocols AA and PA with tags can be expressed with the following equivalence:

$$\begin{array}{c}(\{sk_{DS}, sk_{P0}\}; \{C'''[[PA]_a \mid [AA]_b, [PA_0]_a \mid [AA_0]_b]\}; \{ax_1 \rhd \mathsf{vk}(sk_{DS}), ax_2 \rhd \mathsf{vk}(sk_{P0})\}) \\ \approx_t \\ (\{sk_{DS}, sk_{P0}\}; \{C'[[PA]_a \mid [AA]_b]\}; \{ax_1 \rhd \mathsf{vk}(sk_{DS}), ax_2 \rhd \mathsf{vk}(sk_{P0})\})\end{array}$$

where C''' is defined as follows:

$$C'''[\__1, \__2] \stackrel{def}{=} \; ! \; \nu sk_P.\ \nu id.\ \nu sig.\ \nu pic.\ \ldots \; ! \; \nu ksenc.\ \nu ksmac.\ \__1 \mid !\nu ksenc.\ \nu ksmac.\ \__2$$

Note that both C' and C''' are composition context. While ProVerif can not prove this previous equivalence, it can prove that the following equivalence holds:

$$(\mathcal{K}_0; \{C'''[[AA]_b, [AA_0]_b]\}; \Phi_0) \approx_t (\mathcal{K}_0; \{C'[[AA]_b]\}; \Phi_0)$$

where $\mathcal{K}_0 = \{sk_{DS}, sk_{P0}\}$ and $\Phi_0 = \{ax_1 \rhd \mathsf{vk}(sk_{DS}), ax_2 \rhd \mathsf{vk}(sk_{P0})\}$.

It remains to prove that the shared key are not revealed. As already mentioned in Subsection 5.1.2, the only shared keys between the protocols PA and AA are sk_{P0} and the several instance of $ksenc$,$ksmac$ and sk_P. Hence we have to show that $(\mathcal{K}_0; \{C'''[[AA]_b, [AA_0]_b]\}; \Phi_0)$, $(\mathcal{K}_0; \{C'[[AA]_b]\}; \Phi_0)$, $(\mathcal{K}_0; \{C'''[[PA]_b, [PA_0]_b]\}; \Phi_0)$, and $(\mathcal{K}_0; \{C'[[PA]_b]\}; \Phi_0)$ do not reveal the keys sk_{P0} and the several instances of $ksenc$, $ksmac$, sk_P and $\mathsf{vk}(sk_P)$. Note that according to the hypotheses of Theorem 5.2, we should also check that the processes do not reveal $\mathsf{vk}(ksmac)$, $\mathsf{pk}(ksmac)$,... However, since the processes only use $ksmac, ksenc$ as symmetric key, and sk_P as signing key, one can note that these checks are not necessary.

According to Definition 5.4, the process $(\mathcal{K}_0; \{C'''[[PA]_b, [PA_0]_b]\}; \Phi_0)$ do not reveal the keys sk_{P0}, $ksenc$, $ksmac$, sk_P and $\mathsf{vk}(sk_P)$ if:

— $(\{sk_{DS}, sk_{P0}\}; \{C'''[[PA]_b, [PA_0]_b]\}; \Phi_0)$ preserves the secret of sk_{P0}; and

— $(\{sk_{DS}, sk_{P0}, s\}; \{C'''[[PA]_b | \mathsf{in}(c, x).\mathsf{if}\ x = key\ \mathsf{then}\ \mathsf{out}(c, s)\ \mathsf{else}\ 0, [PA_0]_b]\}; \Phi_0)$ preserves the secrecy of s, with $key \in \{ksenc, ksmac, sk_P, \mathsf{vk}(sk_P)\}$; and

— $(\{sk_{DS}, sk_{P0}, s\}; \{C'''[[PA]_b, [PA_0]_b | \mathsf{in}(c, x).\mathsf{if}\ x = key\ \mathsf{then}\ \mathsf{out}(c, s)\ \mathsf{else}\ 0]\}; \Phi_0)$ preserves the secrecy of s, with $key \in \{ksenc, ksmac, sk_P, \mathsf{vk}(sk_P)\}$.

Similar conditions exist for the three others processes. Using ProVerif, we are able to prove that all these conditions hold. Hence, thanks to Theorem 5.2, we have that:

$$(\mathcal{K}_0; \{C'''[[PA]_b, [PA_0]_b]\}; \Phi_0) \approx_t (\mathcal{K}_0; \{C'[[PA]_b]\}; \Phi_0)$$

implies

$$(\mathcal{K}_0; \{C'''[[AA]_a \mid [PA]_b, [AA_0]_a \mid [PA_0]_b]\}; \Phi_0) \approx_t (\mathcal{K}_0; \{C'[[AA]_a \mid [PA]_b]\}; \Phi_0)$$

5.6 Conclusion

In this chapter, we investigated composition results for privacy-type properties expressed using trace equivalence. We have shown that secure protocols can be safely composed. We consider arbitrary equational theories and we assume that protocols may share some usual primitives provided they are tagged. Moreover, we have to assume that the shared keys are not revealed and we allow public and verification shared keys to be known by the intruder, provided that they were given at the beginning (using the frame Φ_0). However, in our setting (and in many others) such a sequence has to be finite and thus we are only able to deal with a bounded number of public shared keys.

To relax this hypothesis without modifying our model, we need to abstract the public shared keys. For example, the public shared keys initially given to the intruder could be modelled by a unique private shared key (*e.g.* sk_0) and the successive applications of a new one-way function (*e.g.* $\mathsf{s}/1$). Hence, if $\{\mathsf{pk}(sk_1), \ldots, \mathsf{pk}(sk_n), \ldots\}$ represented the infinite set of public shared keys initially given to the intruder then this would be modelled by the set

$$\{\mathsf{pk}(sk_0), \mathsf{pk}(\mathsf{s}(sk_0)), \mathsf{pk}(\mathsf{s}(\mathsf{s}(sk_0))), \ldots\}$$

Then, relying on this representation, we could build a server that distributes the shared public keys to the intruder, *e.g.*

$$S = \nu d.(\mathsf{out}(c, \mathsf{pk}(sk_0)).\mathsf{out}(d, sk_0) \mid !\mathsf{in}(d, x).\mathsf{out}(c, \mathsf{pk}(\mathsf{s}(x))).\mathsf{out}(d, \mathsf{s}(x)))$$

where c is a public channel. However, we would probably need to add in the hypotheses of our composition result that any successive application of s on sk_0 cannot be deduced by the intruder.

We also showed in this chapter how to apply our composition result on the parallel composition of the *Active Authentication* and *Passive Authentication* protocols from the *e-passport* protocols. However, we had to abstract the symmetric session keys that were generated by the *Basic Access Control* protocol beforehand. Thus, to include the *Basic Access Control* protocol, we would have to establish a "sequential" composition result. The term sequential may be misleading since there is no general sequence operator in the applied pi calculus that would compose for example the process $(A \mid B)$ with the process $(P \mid Q)$. In fact, in the case of the e-passport protocols, the *Active Authentication* and *Passive Authentication*) protocols are plugged inside the *Basic Access Control* protocol, *i.e.*

$$BAC_P[PA_P \mid AA_P] \mid BAC_R[PA_R \mid AA_R]$$

where $BAC_P[\_], PA_P$ and AA_P (resp. $BAC_R[\_], PA_R$ and AA_R) are the processes represented the role of the passport (resp. the reader). Note that the processes of the *Basic Access Control* protocol are modelled by the means of contexts where the holes represented where the others processes will be plugged. This plugging is crucial to properly model the behaviour of the whole e-passport protocols. Indeed, since the protocols PA and AA should only be executed if the protocol BAC succeeded then they must be plugged in the proper conditional branch of the protocol BAC which model a successful execution.

Such a "sequential" composition result would probably subsume the results presented in this chapter.

Part II

A decision procedure for trace equivalence

Chapter 6

Model

Contents

We are focusing on a decision procedure of trace equivalence of bounded processes that allows a set of standard cryptographic primitives, in particular signatures, pairing, hash function, symmetric and asymmetric encryptions. Moreover, we will consider that the primitives such as decryption, projection, ... may fail. Hence instead of modelling the behaviour of these cryptographic primitives with an equational theory as in Part I, they will be modelled by a set of rewrite rules. Thus, we propose in this chapter a new model adapted to the behaviour of these primitives, and that is still close to the applied pi calculus.

The aim of this chapter is to provide with a number of simplifications and restrictions of the trace equivalence problem, without losing in generality. First, following the footsteps of Chapter 4, we reduce the decidability of trace equivalence to decidability of symbolic equivalence between sets of constraint systems. Then, we simplify the problem of symbolic equivalence by

1. getting rid of free names in the processes and the actions of the intruder; and
2. only considering normalised recipe
3. only considering constructor constraint system, *i.e.* constraint system which only contains constructor terms.

Chapter 7 will then be dedicated to a decision procedure of symbolic equivalence between sets of constructor constraint systems.

Amongst the existing tools, only SPEC [TD10] relies on constraint systems whereas the tools ProVerif [Bla01] and AKiSs [Cio11] relies on Horn clauses. However, since the aim of [TD10] is a decision procedure for open-bisimulation for bounded processes in the spi-calculus, they reduce this problem to the problem of symbolic equivalence between positive constraint systems. Baudet [Bau05, Bau07] also proves decidability of equivalence between positive constraint systems but for a much larger set of primitives that is for subterm convergent theories. In [CR12], a shorter proof of the result by Baudet is given. Although our decision procedure lacks of cryptographic

primitives in comparison of [Bau05, Bau07], their procedure along with [CD09a] is restricted the trace equivalence of determinate processes that do not contain (non trivial) conditional branching.

6.1 Syntax and semantics

6.1.1 Syntax

Similarly to the applied pi calculus, we consider an infinite set of names $\mathcal{N}$ and an infinite set of variable $\mathcal{X}$. However, we define $\mathcal{F}_c$ and $\mathcal{F}_d$ the finite sets of constructor function symbols and destructor function symbols respectively. More specifically, we consider:

$$\mathcal{F}_c \supseteq \{\mathsf{senc}/2,\ \mathsf{aenc}/2,\ \mathsf{pk}/1,\ \mathsf{sign}/2,\ \mathsf{vk}/1,\ \langle\,\rangle/2,\ \mathsf{h}/1\}$$
$$\mathcal{F}_d = \{\mathsf{sdec}/2,\ \mathsf{adec}/2,\ \mathsf{check}/2,\ \mathsf{proj}_1/1,\ \mathsf{proj}_2/1\}.$$

We allow $\mathcal{F}_c$ to contain more primitives than the one we specified. It allows the users to consider some constants or primitives such as mac. The *constructor terms*, resp. *ground constructor terms*, are those in $\mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$, resp. in $\mathcal{T}(\mathcal{F}_c, \mathcal{N})$.

We model the properties of our cryptographic primitives by the following set of rewrite rules:

$$\begin{array}{rcl} \mathsf{sdec}(\mathsf{senc}(x,y),y) & \to & x \\ \mathsf{adec}(\mathsf{aenc}(x,\mathsf{pk}(y)),y) & \to & x \\ \mathsf{check}(\mathsf{sign}(x,y),\mathsf{vk}(y)) & \to & x \end{array} \qquad \begin{array}{rcl} \mathsf{proj}_1(\langle x,y\rangle) & \to & x \\ \mathsf{proj}_2(\langle x,y\rangle) & \to & y \end{array}$$

This term rewriting system is convergent. Note that only the specified primitives intervene in the rewriting system.

A ground term u is a message if $v{\downarrow}$ is a constructor term for all $v \in st(u)$. We define the predicate $\mathsf{Message}(\cdot)$ on terms such that $\mathsf{Message}(u)$ is true if, and only if, u is a message. For instance, the terms $\mathsf{sdec}(a,b)$, $\mathsf{proj}_1(\langle a, \mathsf{sdec}(a,b)\rangle)$, and $\mathsf{proj}_1(a)$ are not messages. Intuitively, we view terms as modus operandi to compute bitstrings where we use the call-by-value evaluation strategy.

For our processes, since we do not consider replication, we define processes that are similar to the bounded intermediate processes defined in Chapter 4. The grammar of our *plain processes* is as follows:

$$\begin{array}{l} P, Q, R := 0 \\ \quad P \mid Q \\ \quad P + Q \\ \quad \mathsf{if}\ u_1 = u_2\ \mathsf{then}\ P\ \mathsf{else}\ Q \\ \quad \mathsf{in}(u,x).P \\ \quad \mathsf{out}(u,v).P \end{array}$$

where u_1, u_2, u, v are terms, and x is a variable.

This is almost identical to the definition of plain processes in the applied pi calculus. However, we added the choice operator $P + Q$. As mentioned in Chapter 3, this could have been modelled using private channels but we added it to enhance the efficiency of our algorithm. Moreover, we confuse channel with message. At last, similarly to the bounded intermediate processes (see Section 4.1.4), there is no name restriction in the plain processes.

As last, we define concrete processes from Definition 4.1 of intermediate processes with this definition of plain processes.

6.1.2 Semantics

The semantics of concrete processes is given in the Figure 6.1. Note also that, since we allow arbitrary terms for channels, we have to check whether the channel is known by the attacker or not (see rules In and Out). Moreover, we check that all terms that have to be evaluated during the execution are messages.

Let $\mathcal{A}_c$ be the alphabet of actions for the intermediate semantics. For every $w \in \mathcal{A}_c^*$ the relation $\xrightarrow{w}_c$ on intermediate processes is defined in the usual way. For $s \in (\mathcal{A}_c \setminus \{\tau\})^*$, the relation $\stackrel{s}{\Rightarrow}_c$ on intermediate processes is defined by: $A \stackrel{s}{\Rightarrow}_c B$ if, and only if there exists $w \in \mathcal{A}_c^*$ such that $A \xrightarrow{w}_c B$ and s is obtained by erasing all occurrences of τ. Note that by definition, intermediate processes are closed.

$$(\mathcal{E}; \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_1\} \uplus \mathcal{P}; \Phi) \quad \text{if } u{\downarrow} = v{\downarrow}, \mathsf{Message}(u) \text{ and } \mathsf{Message}(v) \qquad (\textsc{Then}_c)$$

$$(\mathcal{E}; \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_2\} \uplus \mathcal{P}; \Phi) \quad \text{if } u{\downarrow} \neq v{\downarrow} \text{ or } \neg\mathsf{Message}(u) \text{ or } \neg\mathsf{Message}(v) \qquad (\textsc{Else}_c)$$

$$(\mathcal{E}; \{\mathsf{out}(u,t).Q_1; \mathsf{in}(v,x).Q_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{Q_1; Q_2\{x \mapsto t\}\} \uplus \mathcal{P}; \Phi) \quad \text{if } \mathsf{Message}(u), \mathsf{Message}(v), \mathsf{Message}(t) \text{ and } u{\downarrow} = v{\downarrow} \qquad (\textsc{Comm}_c)$$

$$(\mathcal{E}; \{\mathsf{in}(u,x).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{in(N,M)}_i (\mathcal{E}; \{Q\{x \mapsto t\}\} \uplus \mathcal{P}; \Phi) \quad \text{if } M\Phi = t,\ \mathit{fvars}(M,N) \subseteq \mathrm{dom}(\Phi),\ \mathit{fnames}(M,N) \cap \mathcal{E} = \emptyset,\ N\Phi{\downarrow} = u{\downarrow}, \mathsf{Message}(M\Phi), \mathsf{Message}(N\Phi), \text{ and } \mathsf{Message}(u) \qquad (\textsc{In}_c)$$

$$(\mathcal{E}; \{\mathsf{out}(u,t).Q\} \uplus \mathcal{P}; \Phi) \xrightarrow{\nu ax_n.out(M,ax_n)}_i (\mathcal{E}; \{Q\} \uplus \mathcal{P}; \Phi \cup \{ax_n \rhd t\}) \quad \text{if } M\Phi{\downarrow} = u{\downarrow}, \mathsf{Message}(u),\ \mathit{fvars}(M) \subseteq \mathrm{dom}(\Phi),\ \mathit{fnames}(M) \cap \mathcal{E} = \emptyset,\ \mathsf{Message}(M\Phi), \mathsf{Message}(t) \text{ and } ax_n \in \mathcal{AX},\ n = |\Phi| + 1 \qquad (\textsc{Out}_c)$$

$$(\mathcal{E}; \{P_1 \mid P_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{P_1; P_2\} \uplus \mathcal{P}; \Phi) \qquad (\textsc{Par}_c)$$

$$(\mathcal{E}; \{P_1 + P_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{P_1\} \uplus \mathcal{P}; \Phi) \qquad (\textsc{Choice}_c\text{-}1)$$

$$(\mathcal{E}; \{P_1 + P_2\} \uplus \mathcal{P}; \Phi) \xrightarrow{\tau}_i (\mathcal{E}; \{P_2\} \uplus \mathcal{P}; \Phi) \qquad (\textsc{Choice}_c\text{-}2)$$

where u, v, t are ground terms, and x is a variable.

Figure 6.1: Concrete semantics

As we will see in Chapter 9, checking whether a term is a message is very similar the check of failure of a term evaluation in ProVerif. In ProVerif, the evaluation of terms is explicit in $\mathsf{let}\ x = D\ \mathsf{in}\ P$ whereas in our semantics, the evaluation of a term is performed *on the fly*. Typically, given a plain process P, one can transform P into a ProVerif process that has the same behaviour by replacing every $\mathsf{out}(u,t)$ by $\mathsf{let}\ x = \mathrm{addeval}(u)\ \mathsf{in}\ \mathsf{let}\ y = \mathrm{addeval}(t)\ \mathsf{in}\ \mathsf{out}(x,y)$, every $\mathsf{in}(u,x)$ by $\mathsf{let}\ y = \mathrm{addeval}(u)\ \mathsf{in}\ \mathsf{in}(y,x)$, ...

Similarly to Chapter 4, given a concrete process $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1)$, the set of its traces is defined by:

$$\mathsf{trace}_c((\mathcal{E}_1; \mathcal{P}_1; \Phi_1)) = \{(s, \nu\mathcal{E}_2.\Phi_2) \mid (\mathcal{E}_1; \mathcal{P}_1; \Phi_1) \stackrel{s}{\Rightarrow}_c (\mathcal{E}_2; \mathcal{P}_2; \Phi_2) \text{ for some } (\mathcal{E}_2; \mathcal{P}_2; \Phi_2)\}$$

Note that due to our semantics, we have in fact $\mathcal{E}_1 = \mathcal{E}_2$.

Example 6.1. *Consider the sequence of actions* $\mathsf{tr}' = \nu ax_1.out(c_1, ax_1).in(c_1, \mathsf{proj}_1(\langle ax_1, \mathsf{sdec}(a, b)\rangle)).\nu ax_2.out(c_2, ax_2)$. *For all concrete process A, no trace in* $\mathsf{trace}_c(A)$ *can have* tr' *as sequence of actions. Indeed, for all* Φ, $\mathsf{Message}(\mathsf{proj}_1(\langle ax_1, \mathsf{sdec}(a,b)\rangle)\Phi) = \mathsf{false}$.

Intuitively, the intruder will never be able to compute $\mathsf{proj}_1(\langle ax_1, \mathsf{sdec}(a,b)\rangle)\Phi$ *since the computation of* $\mathsf{sdec}(a,b)$ *will fail and so* $\langle ax_1, \mathsf{sdec}(a,b)\rangle$ *does not represent a bitstring.*

6.1.3 Equivalence

We redefine the static equivalence (Definition 3.1) in this context.

Definition 6.1. *Let $\mathcal{E}$ a set of private names. Let Φ and Φ' two frames. We say that $\nu\mathcal{E}.\Phi$ and $\nu\mathcal{E}.\Phi'$ are* statically equivalent*, written $\nu\mathcal{E}.\Phi \sim_c \nu\mathcal{E}.\Phi'$, when $\mathrm{dom}(\Phi) = \mathrm{dom}(\Phi')$ and when for all terms M, N such that $fvars(M,N) \subseteq \mathrm{dom}(\Phi)$ and $fnames(M,N) \cap \mathcal{E} = \emptyset$, we have:*

- $\mathsf{Message}(M\Phi)$ *if and only if* $\mathsf{Message}(M\Phi')$
- *if* $\mathsf{Message}(M\Phi)$ *and* $\mathsf{Message}(N\Phi)$ *then $M\Phi\downarrow = N\Phi\downarrow$ if and only $M\Phi\downarrow = N\Phi'\downarrow$.*

The second item of Definition 6.1 corresponds to the usual definition of the static equivalence. However, we add the conditions $\mathsf{Message}(M\Phi)$ and $\mathsf{Message}(N\Phi)$ otherwise the equality $M\Phi\downarrow = N\Phi\downarrow$ would be meaningless. Note that we do not require that $\mathsf{Message}(M\Phi')$ or $\mathsf{Message}(N\Phi')$ but it is implied by the first item of the static equivalence.

We recall the definition of trace equivalence in Section 3.1 that we adapt to our set of traces and static equivalence:

Definition 6.2 (trace equivalence $\approx_t$)**.** *Let A and B be concrete processes with the same set of private names $\mathcal{E}$. $A \sqsubseteq_t B$ if for every $(s, \nu\mathcal{E}.\Phi) \in \mathsf{trace}_c(A)$, there exists $(s', \nu\mathcal{E}.\Phi') \in \mathsf{trace}(B)$ such that $s = s'$ and $\nu\mathcal{E}.\Phi \sim_c \nu\mathcal{E}.\Phi'$.*

Two closed extended processes A and B are trace equivalent*, denoted by $A \approx_t B$, if $A \sqsubseteq_t B$ and $B \sqsubseteq_t A$.*

Note that we do not have anymore the condition $fnames(B) \cap bnames(s) = \emptyset$. This condition was essential for the applied pi calculus since a action label could be of the form $\nu c.out(p,c)$ with c a channel name. In our case, only the parameters ax_i can be found in s hence we necessary have $bnames(s) = \emptyset$.

6.2 Symbolic calculus

In order to decide the trace equivalence between two concrete process, we propose, similarly to Section 4.2, a symbolic calculus and its semantics. We consider the constraint system (Definition 4.3) and symbolic process (Definition 4.6) used in Section 4.2.

6.2.1 Semantics

We recall the notations used in the symbolic calculus in Chapter 4.2: We consider a new set $\mathcal{X}^2$ of variables called *second order variables* $X, Y, \ldots$ To avoid conflict between the variables in $\mathcal{X}^2$ and the one we already used to described the protocols and the processes, we denote by $\mathcal{X}^1$ the latter set variables and call them *first order variables*. Note that $\mathcal{X}^1$, $\mathcal{X}^2$, $\mathcal{AX}$ are all disjoint subsets of $\mathcal{X}$. We call *recipe*, usually denoted ξ, the terms in $\mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X}^2 \cup \mathcal{AX})$. We say that a recipe ξ is *closed* (or *ground*) if $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{AX})$. Given a recipe $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{X}^2 \cup \mathcal{AX})$, we denote $param(\xi)$ the set of parameters in ξ, *i.e.* $vars(\xi) \cap \mathcal{AX}$.

We adapt the definition of solutions of a constraint system so that it matches with the concrete semantics (see Figure 6.1).

Definition 6.3 (concrete solution)**.** *A* concrete solution *of a constraint system $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq)$ is a pair of substitutions (σ, θ) such that σ is a mapping from $vars^1(\mathcal{C})$ to $\mathcal{T}(\mathcal{F}, \mathcal{N})$, θ is a mapping from $vars^2(\mathcal{C})$ to $\mathcal{T}(\mathcal{F}, \mathcal{N} \setminus \{\mathcal{E}\}, \mathcal{AX})$, and:*

1. *for all $(X, k \overset{?}{\vdash} x) \in D$, $(X\theta)(\Phi\sigma) = x\sigma$, $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$ and $\mathsf{Message}(x\sigma)$;*
2. *for all $(s \overset{?}{=} s') \in Eq$, $s\sigma\downarrow = s'\sigma\downarrow$, $\mathsf{Message}(s\sigma)$ and $\mathsf{Message}(s'\sigma)$;*
3. *for all $(s \overset{?}{\neq} s') \in Eq$, $s\sigma\downarrow \neq s'\sigma\downarrow$, or $\neg\mathsf{Message}(s\sigma)$, or $\neg\mathsf{Message}(s'\sigma)$.*
4. *for all $(ax_i \rhd u_i) \in \Phi$, $\mathsf{Message}(u_i\sigma)$*

The substitution σ is called the first-order solution *of $\mathcal{C}$ associated to θ, called* second-order solution *of $\mathcal{C}$. The set of solutions of a constraint system $\mathcal{C}$ is denoted $\mathsf{Sol}_c(\mathcal{C})$. A constraint system $\mathcal{C}$ is* satisfiable *if $\mathsf{Sol}_c(\mathcal{C}) \neq \emptyset$.*

Compared to the usual definition of solution (see Definition 4.4), we added the fact that every term in the constraint system has to be a message after instantiation. Moreover, the first order terms of a frame element also has to be a message as it is specified in the rule (OUT_c).

Example 6.2. *Coming back to the Example 4.7, we had* $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$. *But* (σ, θ) *also satisfies all conditions of a concrete solution, hence* $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C}_1)$.

The rules of Figure 6.2 define the semantics of symbolic processes. Similarly to Section 4.2, this relation transforms a symbolic process into a symbolic process. Once again, thanks to this symbolic semantics, we avoid infinite branching due to the inputs of the environment. We added some rules for the choice operator and we removed the rules that was specific to the outputs of channel names.

$$(\mathcal{E}; \{\mathsf{if}\ u = v\ \mathsf{then}\ P_1\ \mathsf{else}\ P_2\} \uplus \mathcal{P}; \Phi; D; Eq) \xrightarrow{\tau}_{cs} (\mathcal{E}; \{P_1\} \uplus \mathcal{P}; \Phi; D; Eq \cup \{u \stackrel{?}{=} v\}) \quad (\text{THEN}_{sc})$$

$$(\mathcal{E}; \{\{\mathsf{if}\ u = v\ \mathsf{then}\ P_1\ \mathsf{else}\ P_2\} \uplus \mathcal{P}; \Phi; D; Eq) \xrightarrow{\tau}_{cs} (\mathcal{E}; \{P_2\} \uplus \mathcal{P}; \Phi; D; Eq \cup \{u \stackrel{?}{\neq} v\}) \quad (\text{ELSE}_{sc})$$

$$(\mathcal{E}; \{\mathsf{out}(u,t).Q_1; \mathsf{in}(v,x).Q_2\} \uplus \mathcal{P}; \Phi; D; Eq) \xrightarrow{\tau}_{cs} (\mathcal{E}; \{Q_1; Q_2\{x \mapsto t\}\} \uplus \mathcal{P}; \Phi; D; Eq \cup \{u \stackrel{?}{=} v\}) \quad (\text{COMM}_{sc})$$

$$(\mathcal{E}; \{P \mid Q\} \uplus \mathcal{P}; \Phi; D; Eq) \xrightarrow{\tau}_{cs} (\mathcal{E}; \{P; Q\} \uplus \mathcal{P}; \Phi; D; Eq) \quad (\text{PAR}_{sc})$$

$$(\mathcal{E}; \{P + Q\} \uplus \mathcal{P}; \Phi; D; Eq) \xrightarrow{\tau}_{cs} (\mathcal{E}; \{P\} \uplus \mathcal{P}; \Phi; D; Eq) \quad (\text{CHOICE}_{sc}\text{-1})$$

$$(\mathcal{E}; \{P + Q\} \uplus \mathcal{P}; \Phi; D; Eq) \xrightarrow{\tau}_{cs} (\mathcal{E}; \{Q\} \uplus \mathcal{P}; \Phi; D; Eq) \quad (\text{CHOICE}_{sc}\text{-1})$$

$$(\mathcal{E}; \{\mathsf{in}(u,x).Q\} \uplus \mathcal{P}; \Phi; D; Eq) \xrightarrow{in(Z,Y)}_{cs} (\mathcal{E}; \{Q\{x \mapsto y\}\} \uplus \mathcal{P}; \Phi; D \cup \{Y, n \stackrel{?}{\vdash} y; Z, n \stackrel{?}{\vdash} z\}; Eq \cup \{z \stackrel{?}{=} u\})$$
if Y, y, Z, z are fresh variables, $n = |\Phi|$ (IN_{sc})

$$(\mathcal{E}; \{\mathsf{out}(u,t).Q\} \uplus \mathcal{P}; \Phi; D; Eq) \xrightarrow{\nu ax_n.out(Z,ax_n)}_{cs} (\mathcal{E}; \{Q\} \uplus \mathcal{P}; \Phi \cup \{ax_{n+1} \rhd t\}; D \cup \{Z, n \stackrel{?}{\vdash} z\}; Eq \cup \{z \stackrel{?}{=} u\})$$
if $ax_n \in \mathcal{AX}$ such that $n = |\Phi|$, Z, z are fresh variables (OUT_{sc})

u, v, t are terms and x is a variable.

Figure 6.2: Concrete symbolic semantics

Note that the rules are almost identical to the symbolic semantics used for the applied pi calculus, except for the new/removed rules. Indeed, all the new requirement of the concrete semantics were added in the definition of a concrete solution of a constraint system instead of being added in the symbolic semantics directly. We define the relations $\xrightarrow{w}_{cs}$ and $\stackrel{\mathsf{tr}}{\Rightarrow}_{cs}$ as usual.

We denote $\mathsf{trace}_{cs}(A)$ the set of concrete symbolic traces of a concrete process as follow:

$$\mathsf{trace}_{cs}(A) = \{(\mathsf{tr}, (\mathcal{E}; \Phi; D; Eq)) \mid \mathcal{A} \stackrel{\mathsf{tr}}{\Rightarrow}_{cs} (\mathcal{E}; \mathcal{P}; \Phi; D; Eq) \text{ for some } \mathcal{E}, \mathcal{P}, \Phi, D, Eq\}$$

When tr is fixed, we also write $(\mathsf{tr}, \Sigma) \in \mathsf{trace}_{cs}(A)$ to define the set Σ as the set of constraint systems $\{\mathcal{C} \mid (\mathsf{tr}, \mathcal{C}) \in \mathsf{trace}_{cs}(A)\}$.

6.2.2 From trace equivalence to concrete symbolic equivalence

Following the results of Chapter 4, we define the concrete symbolic equivalence between two sets of constraint systems.

Definition 6.4 (concrete symbolic equivalence)**.** *Let Σ and Σ' be two sets of constraint systems that contain constraint systems having the same structure. We say that Σ and Σ' are in concrete symbolic equivalence, denoted by $\Sigma \approx_s^c \Sigma'$, if for all $\mathcal{C} \in \Sigma$, for all $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$, there exists $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $\nu\mathcal{E}.\Phi\sigma \sim_c \nu\mathcal{E}.\Phi'\sigma'$ (and conversely for all $\mathcal{C}' \in \Sigma'$) where $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq)$ and $\mathcal{C}' = (\mathcal{E}; \Phi'; D'; Eq')$.*

The definition of $\approx_s^c$ is very close to the definition of $\approx_s$ (see Definition 4.8). Typically, only the solutions used in the equivalence and the static equivalence are different between $\approx_s^c$ and $\approx_s$.

Example 6.3. *Coming back to the Example 4.8, $\Sigma \approx_s^c \Sigma'$ also holds. Moreover $\{\mathcal{C}_1\}$ and $\{\mathcal{C}_1'\}$ are not in concrete symbolic equivalence.*

We establish the link between trace equivalence and concrete symbolic equivalence:

Theorem 6.1. *Let A and B two concrete processes: $A \approx_t B$ if, and only if, for all sequences of symbolic actions tr, for all $(\mathsf{tr}, \Sigma) \in \mathit{trace}_{cs}(A)$, for all $(\mathsf{tr}, \Sigma') \in \mathit{trace}_{cs}(B)$, $\Sigma \approx_s^c \Sigma'$.*

Proof (sketch). The proof of this theorem is an adaptation of the proof of Proposition 4.4. It also relies on soundness and completeness results similar to Lemmas 4.2 and 4.3 that focus on concrete solutions instead of solutions of a constraint system and focus on $\xrightarrow{w}_{cs}$ instead of $\xrightarrow{w}_s$. The proofs of these lemmas are a simple adaptation of the proofs of Lemmas 4.2 and 4.3. □

The focus on the rest of this part is to provide an algorithm that decide that concrete symbolic equivalence between two sets of constraint systems.

6.3 Getting rid of some recipes

Deciding the symbolic equivalence of sets of constraint systems is a difficult problem. One the main issues is that the set of recipes $\mathcal{T}(\mathcal{F}, \mathcal{AX} \cup \mathcal{N})$ that have to be considered for the static equivalence and also for the solutions of a constraint system is infinite. In this section, we aim at reducing this set of recipes. More specifically, we will first show that $\Sigma \approx_s^c \Sigma'$ is equivalent to the symbolic equivalence where we do not consider public names for the intruder ($\approx_s^{\mathcal{T}(F,\mathcal{AX})}$). Then we show that it is equivalent to the symbolic equivalence where we do not consider public names and recipes that are not normalised ($\approx_s^{\Pi_n}$) . Lastly, we will show that there exist Σ_1, Σ_1' sets of constraint systems that only contain constructor terms, called *constructor constraint systems*, such that $\Sigma \approx_s^c \Sigma'$ is equivalent to $\Sigma_1 \approx_s \Sigma_1'$ where $\approx_s$ excludes recipes containing public names or recipe not normalised. Moreover, the predicate $\mathsf{Message}(\cdot)$ is no longer required.

6.3.1 Getting rid of names

One of the infinite components in $\mathcal{T}(\mathcal{F}, \mathcal{AX} \cup \mathcal{N})$ is the infinite set of names $\mathcal{N}$. We show in this subsection that we do not need to consider the names $\mathcal{N}$ in the intruder's recipes.

6.3.1.1 Processes without free names

First of all, we have to distinguish two kinds of names: the ones that are already used by the processes and the ones that might be introduced by the intruder in the trace. In a first step, illustrated in the Example 6.4, we avoid free names in the processes by privatising and disclosing them.

Example 6.4. *Consider the concrete process A defined as follows:*

$$A \;=\; (\{sk\}; \mathsf{out}(c, \mathsf{aenc}(a, \mathsf{pk}(sk))).\mathsf{in}(c, x).\nu n.\mathsf{out}(c, \langle x, n\rangle); \emptyset).$$

We can transform A into a process A' that does not contain any free public name by putting the free names c, a inside the private set of names and adding two frame elements with c and a:

$$A' \;=\; (\{sk, c, a\}; \mathsf{out}(c, \mathsf{aenc}(a, \mathsf{pk}(sk))).\mathsf{in}(c, x).\nu n.\mathsf{out}(c, \langle x, n\rangle); \{ax_1 \rhd c, ax_2 \rhd a\}).$$

Consider now the trace $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathit{trace}_c(A)$ *where* $\mathcal{E} = \{sk\}$, $\mathsf{tr} = \nu ax_1.out(c, ax_1).in(c, ax_1).\nu ax_2.out(c, ax_2)$ *and* $\Phi = \{ax_1 \rhd \mathsf{aenc}(a, \mathsf{pk}(sk)), ax_2 \rhd \langle \mathsf{aenc}(a, \mathsf{pk}(sk)), n\rangle\}$. *There exists a similar trace* $(\mathsf{tr}', \nu\mathcal{E}'.\Phi') \in \mathit{trace}_c(A')$ *where:*

— $\mathcal{E}' = \{sk, c, a\}$

— $\mathsf{tr}' = \nu ax_3.out(ax_1, ax_3).in(ax_1, ax_3).\nu ax_4.out(c, ax_4)$

— $\Phi' = \{ax_1 \rhd c;\ ax_2 \rhd a;\ ax_3 \rhd \mathsf{aenc}(a, \mathsf{pk}(sk));\ ax_4 \rhd \langle \mathsf{aenc}(a, \mathsf{pk}(sk)), n\rangle\}$

We formalise the transformation presented in Example 6.4 in the following lemma.

Lemma 6.1. *Let* $A = (\mathcal{E}_A; \mathcal{P}_A; \Phi_A)$ *and* $B = (\mathcal{E}_B; \mathcal{P}_B; \Phi_B)$ *be two closed concrete processes such that* $\mathcal{E}_A = \mathcal{E}_B$ *and* $\mathrm{dom}(\Phi_A) = \mathrm{dom}(\Phi_B)$. *Let's denote* $n = |\Phi_A|$. *Assume that* A *and* B *contains* $m - 1$ *free names,* i.e. $\mathit{fnames}(A) \cup \mathit{fnames}(B) = \{c_2, \ldots, c_m\}$ *and let* c_1 *be a fresh name in* $\mathcal{N}$. *Let* Φ_0 *the frame* $\{ax_1 \rhd c_1, \ldots, ax_m \rhd c_m\}$, Φ'_A *(resp* Φ'_B*) be the frame obtained from* Φ_A *(resp.* Φ_B*) by replacing every* $(ax_k \rhd u_k) \in \Phi_A$ *(resp.* Φ_B*) by* $(ax_{k+m} \rhd u_k)$. *The following property holds:*

$$A \approx_t B \text{ if, and only if, } (\mathcal{E}_0; \mathcal{P}_A; \Phi_0 \cup \Phi'_A) \approx_t (\mathcal{E}_0; \mathcal{P}_B; \Phi_0 \cup \Phi'_B)$$

where $\mathcal{E}_0 = \mathit{fnames}(A) \cup \mathit{fnames}(B) \cup \mathcal{E}$

Proof (sketch). Let's denote $A' = (\mathcal{E}_0; \mathcal{P}_A; \Phi_0 \cup \Phi'_A)$ and $B' = (\mathcal{E}_0; \mathcal{P}_B; \Phi_0 \cup \Phi'_B)$. The proof consists of applying a simple transformation on the traces of A and B. Considering $(\mathsf{tr}, \nu\mathcal{E}_A.(\Phi_A \cup \Phi^1_A)) \in \mathsf{trace}(A)$, we transform tr into tr' as follows:

— we replace every instance of c_i in tr by ax_i, for all $i \in \{1, \ldots, m\}$.

— we replace every instance of ax_i by ax_{m+i} for all $i > n$

Similarly, we transform Φ^1_A into Φ^2_A by replacing every $(ax_i \rhd u)$ in Φ^1_A by $(ax_{i+m} \rhd u)$. Note that both transformations are bijective. One can easily show that $(\mathsf{tr}', \nu\mathcal{E}_0.(\Phi_0 \cup \Phi'_A \cup \Phi^2_A)) \in \mathsf{trace}_c(A')$. Similarly, if $(\mathsf{tr}, \nu\mathcal{E}_B.(\Phi_B \cup \Phi^1_B)) \in \mathsf{trace}(B)$, then $(\mathsf{tr}', \nu\mathcal{E}_0.(\Phi_0 \cup \Phi'_B \cup \Phi^2_B)) \in \mathsf{trace}_c(B')$ with Φ^2_B is obtained from Φ^1_B by replacing every $(ax_i \rhd u)$ in Φ^1_B by $(ax_{i+m} \rhd u)$.

At last, one can show that $\nu\mathcal{E}.(\Phi_A \cup \Phi^1_A) \sim_c \nu\mathcal{E}.(\Phi_B \cup \Phi^1_B)$ if and only if $\nu\mathcal{E}_0.(\Phi_0 \cup \Phi'_A \cup \Phi^2_A) \sim_c \nu\mathcal{E}_0.(\Phi_0 \cup \Phi'_B \cup \Phi^2_B)$ by applying on the recipes the same transformations used for the sequence of labels. □

Thanks to Lemma 6.1, we can now assume that there is no public names in our constraint system. Indeed, given a concrete process A such that $\mathit{fnames}(A) = \emptyset$, for all $(\mathsf{tr}, \mathcal{C}) \in \mathsf{trace}_{cs}(A)$, $\mathit{fnames}(\mathcal{C}) = \emptyset$. Moreover Lemma 6.1 also indicates that Φ_0 contains at least one frame element $(ax_1 \rhd c_1)$ with $c_1 \in \mathcal{N}$. Thus, if A is the concrete process obtained from Lemma 6.1 then there exists $a \in \mathcal{N}$ such that for all $(\mathsf{tr}, \mathcal{C}) \in \mathsf{trace}_{cs}(A)$, $ax_1\Phi = a$ where $C = (\mathcal{E}; \Phi; D; Eq)$.

6.3.1.2 Removing all names in the recipes

In the previous paragraph we showed how avoid free names in the processes. However, as previously mentioned, the intruder might still introduce new public names in the sequence of labels tr. We show that it is possible to model the public names in $\mathcal{N}$ by $\mathsf{h}(\mathsf{h}(\ldots \mathsf{h}(ax_1) \ldots))$. Consider a constraint system $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq)$ such that $\mathit{fnames}(\mathcal{C}) = \emptyset$. Following Lemma 6.1, we assume that Φ is not empty and $ax_1\Phi \in \mathcal{E}$. Since $\mathcal{N}$ is a countable set, we denote $\mathcal{N} \setminus \mathcal{E} = \{b_1, \ldots, b_n, \ldots\}$. Intuitively, we would represent each b_i by $\mathsf{h}^i(ax_1)$ in the recipe of the intruder where $\mathsf{h}^i(\cdot)$ is i successive applications of h on ax_1. However, this model may not work for all solutions of a constraint system.

Example 6.5. *Consider the constraint system* $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq)$ *where:*

— $\mathcal{E} = \{a, c\}$

— $\Phi = \{ax_1 \rhd a, ax_2 \rhd \mathsf{senc}(c, x)\}$

— $D = \{X, 2 \overset{?}{\vdash} x; Y, 2 \overset{?}{\vdash} y\}$

— $Eq = \{c \stackrel{?}{\neq} \mathsf{sdec}(y, \mathsf{h}(a))\}$

Consider $\theta = \{X \mapsto b_1, Y \mapsto ax_2\}$ *and* $\sigma = \{x \mapsto b_1, y \mapsto \mathsf{senc}(c, b_1)\}$. *We have that* $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$. *Indeed, we trivially have* $\mathsf{Message}(a)$, $\mathsf{Message}(b_1)$ *and* $\mathsf{Message}(\mathsf{senc}(c, b_1))$. *Furthermore* $\neg\mathsf{Message}(\mathsf{sdec}(\mathsf{senc}(c, b_1), \mathsf{h}(a)))$ *since* $\mathsf{sdec}(\mathsf{senc}(c, b_1), \mathsf{h}(a))$ *is in normal form. Thus* $c \stackrel{?}{\neq} \mathsf{sdec}(y, \mathsf{h}(a))$ *is satisfied by* σ.

However, if we replace apply our replacement of free names introduced by the intruder, we obtain that $\theta' = \{X \mapsto \mathsf{h}(ax_1), Y \mapsto ax_2\}$ *and so* $\sigma' = \{x \mapsto \mathsf{h}(a), y \mapsto \mathsf{senc}(c, \mathsf{h}(a))\}$. *Unfortunately* $(\sigma', \theta') \notin \mathsf{Sol}_c(\mathcal{C})$ *since* $\mathsf{Message}(c)$, $\mathsf{Message}(\mathsf{sdec}(\mathsf{senc}(c, \mathsf{h}(a)), \mathsf{h}(a)))$ *and* $c{\downarrow} = \mathsf{sdec}(\mathsf{senc}(c, \mathsf{h}(a)), \mathsf{h}(a))$.

On the other hand, if we consider the replacement $b_k \mapsto \mathsf{h}^{2\times k}(ax_1)$, *then we obtain that* $\theta'' = \{X \mapsto \mathsf{h}(\mathsf{h}(ax_1)), Y \mapsto ax_2\}$ *and so* $\sigma'' = \{x \mapsto \mathsf{h}(\mathsf{h}(a)), y \mapsto \mathsf{senc}(c, \mathsf{h}(\mathsf{h}(a)))\}$. *In such a case,* $(\sigma'', \theta'') \in \mathsf{Sol}_c(\mathcal{C})$.

As illustrated in Example 6.5, we cannot fix the replacement of the public names in advance when checking the symbolic equivalence. The replacement will depend on the solutions that we consider. Let N be a positive integer. Let $a \in \mathcal{E}$. We denote $\sigma_{\mathcal{E},N,a}$ and $\theta_{\mathcal{E},N}$ the replacements defined for all $i \in \mathbb{N}^+$ by $b_i\sigma_{\mathcal{E},N,a} = \mathsf{h}^{i\times N}(a)$ and $b_i\theta_{\mathcal{E},N} = \mathsf{h}^{i\times N}(ax_1)$.

We now show that when checking the concrete symbolic equivalence of sets of constraint systems, we only have to consider the recipes of the intruder that do not contain names. Let's denote $\sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})}$ the concrete static equivalence where we only consider recipe $\xi, \xi' \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$, and let's denote $\approx_s^{\mathcal{T}(F,\mathcal{AX})}$ the symbolic equivalence on concrete constraint systems which only consider the solution (σ, θ) with $\mathit{fnames}(\theta) = \emptyset$, and rely on the static equivalence $\sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})}$. The proof of the following lemma can be found in Appendix C.1.

Lemma 6.2. *Let* Σ *and* Σ' *two sets of constraint systems that contain constraint systems having the same structure. Assume that* $\mathcal{E}$ *is the common set of privates names in* Σ *and* Σ'. *At last, assume that there exists* $a \in \mathcal{E}$ *such that for all* $\mathcal{C} \in \Sigma \cup \Sigma'$, $ax_1\Phi = a$ *with* Φ *the frame of* $\mathcal{C}$.

If $\Sigma \approx_s^{\mathcal{T}(F,\mathcal{AX})} \Sigma'$ *then for all* $\mathcal{C} \in \Sigma$, *for all* $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$, *there exists* N' *such that for all* $N > N'$, *there exist* $\mathcal{C}' \in \Sigma'$ *and a substitution* σ' *such that* $(\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ *and* $\nu\mathcal{E}.\Phi\sigma\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \nu\mathcal{E}.\Phi'\sigma'\sigma_{\mathcal{E},N,a}$.

The previous lemma is not sufficient to prove that $\Sigma \approx_s^{\mathcal{T}(F,\mathcal{AX})} \Sigma'$ implies $\Sigma \approx_s^c \Sigma'$. Indeed, we matched each solution of a constraint system in Σ with a solution in a constraint system in Σ' but the static equivalence does not correspond to the usual static equivalence $\sim_c$. Moreover, for two ground frames Φ and Φ', we do not necessarily have that $\nu\mathcal{E}.\Phi\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \nu\mathcal{E}.\Phi'\sigma_{\mathcal{E},N,a}$ implies $\nu\mathcal{E}.\Phi \sim_c \nu\mathcal{E}.\Phi'$

Example 6.6. *Let* $\mathcal{E} = \{a, c\}$. *Let* $\Phi = \{ax_1 \rhd a, ax_2 \rhd \mathsf{senc}(c, b_1)\}$ *and* $\Phi' = \{ax_1 \rhd a, ax_2 \rhd \mathsf{senc}(c, \mathsf{h}(a))\}$. *Consider* $\xi = \mathsf{sdec}(ax_2, b_1)$. $\mathsf{Message}(\xi\Phi)$ *holds whereas* $\mathsf{Message}(\xi\Phi')$ *does not hold. Hence* $\Phi \not\sim_c \Phi'$.

However, if we assume that $N = 1$, $\Phi\sigma_{\mathcal{E},1,a} = \{ax_1 \rhd a, ax_2 \rhd \mathsf{senc}(c, \mathsf{h}(a))\}$ *and* $\Phi'\sigma_{\mathcal{E},1,a} = \{ax_1 \rhd a, ax_2 \rhd \mathsf{senc}(c, \mathsf{h}(a))\}$. *Since* $\Phi\sigma_{\mathcal{E},1,a} = \Phi'\sigma_{\mathcal{E},1,a}$, *we trivially have* $\Phi\sigma_{\mathcal{E},1,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma_{\mathcal{E},1,a}$. *On the other hand, if we assume that* $N = 2$, *then* $\Phi\sigma_{\mathcal{E},2,a} \not\sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma_{\mathcal{E},2,a}$.

Note that the previous example raises the same issues than Example 6.5: We can not fix in advance the representation of names even for the static equivalence. However, we show in the next lemma how we obtain the static equivalence $\sim_c$ from $\sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})}$. (Proof in Appendix C.1)

Lemma 6.3. *Let* $\mathcal{E}$ *be a set of private names. Let* Φ *and* Φ' *two ground frames with the same domain and* $ax_1\Phi = ax_1\Phi' \in \mathcal{E}$. *If for all* N', *there exists* $N > N'$ *such that* $\Phi\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma_{\mathcal{E},N,a}$ *then* $\Phi \sim_c \Phi'$.

Thanks to Lemmas 6.2 and 6.3, we can now show that $\approx_s^c$ is the same relation than $\approx_s^{\mathcal{T}(F,\mathcal{AX})}$.

Lemma 6.4. *Let Σ and Σ' two finite sets of constraint systems that contain constraint systems having the same structure. Assume that $\mathcal{E}$ is the common set of private names in Σ and Σ'. At last, assume that there exists $a \in \mathcal{E}$ such that for all $\mathcal{C} \in \Sigma \cup \Sigma'$, $ax_1\Phi = a$ with Φ the frame of $\mathcal{C}$. The following property holds:*

$$\Sigma \approx_s^c \Sigma' \text{ if, and only if, } \Sigma \approx_s^{\mathcal{T}(F,\mathcal{AX})} \Sigma'$$

Proof. We first prove the right implication of the equivalence ($\Rightarrow$). Let $\mathcal{C} \in \Sigma$ and let $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ such that for all $X \in \mathrm{dom}(\theta)$, $\mathit{fnames}(X\theta) = \emptyset$. By hypothesis $\Sigma \approx_s^c \Sigma'$ hence there exist $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $\Phi\sigma \sim_c \Phi'\sigma'$. We show that $\nu\mathcal{E}.\Phi\sigma \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \nu\mathcal{E}.\Phi'\sigma'$. Let $\xi, \xi' \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$ such that $\mathit{param}(\xi) \subseteq \mathrm{dom}(\Phi\sigma)$ and $\mathit{param}(\xi') \subseteq \mathrm{dom}(\Phi\sigma)$. $\xi, \xi' \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$ implies that $\xi, \xi' \in \mathcal{T}(\mathcal{F}, \mathcal{AX} \cup \mathcal{N} \setminus \mathcal{E})$ and $\mathit{fnames}(\xi, \xi') \cap \mathcal{E} = \emptyset$. Since $\Phi\sigma \sim_c \Phi'\sigma'$, we can deduce that $\mathsf{Message}(\xi\Phi\sigma)$ is equivalent to $\mathsf{Message}(\xi\Phi'\sigma')$; and if $\mathsf{Message}(\xi\Phi\sigma)$ and $\mathsf{Message}(\xi'\Phi\sigma)$ then $\xi\Phi\sigma\downarrow = \xi'\Phi\sigma\downarrow$ is equivalent to $\xi\Phi'\sigma'\downarrow = \xi'\Phi'\sigma'\downarrow$. Thus $\nu\mathcal{E}.\Phi\sigma \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \nu\mathcal{E}.\Phi'\sigma'$ and so the result holds.

The other inclusion of the symbolic equivalence $\approx_s^{\mathcal{T}(F,\mathcal{AX})}$ is proved symmetrically.

We now prove the left implication ($\Leftarrow$). Let $\mathcal{C} \in \Sigma$ and let $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$. For a constraint system $\mathcal{C}'$, let's denote $\Phi(\mathcal{C}')$ the frame of $\mathcal{C}'$. Thanks to Lemma 6.2, there exists N' such that for all $N > N'$, there exist $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $\Phi(\mathcal{C})\sigma\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi(\mathcal{C}')\sigma'\sigma_{\mathcal{E},N,a}$. Hence, $\{(N, \mathcal{C}', \sigma') \mid (\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $\Phi(\mathcal{C})\sigma\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi(\mathcal{C}')\sigma'\sigma_{\mathcal{E},N,a}\}$ is an infinite set. However, Σ' is a finite set of constraint systems thus there exists $\mathcal{C}' \in \Sigma'$ such that the set $S_2 = \{(N, \sigma') \mid (\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $\Phi(\mathcal{C})\sigma\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi(\mathcal{C}')\sigma'\sigma_{\mathcal{E},N,a}\}$ is infinite. Furthermore, when $\mathcal{C}'$ and θ are fixed, by definition of a solution of a constraint system, we have that for all σ_1, σ_2, $(\sigma_1, \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $(\sigma_2, \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ implies $\sigma_1 = \sigma_2$, *i.e.* the first order solution is fixed by the second order solution. Thus for all $(N_1, \sigma_1), (N_2, \sigma_2) \in S_2$, $\sigma_1 = \sigma_2$. It implies that there exists σ' such that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ and for all $N' > 0$, there exists $N > N'$ such that $\Phi(\mathcal{C})\sigma\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi(\mathcal{C}')\sigma'\sigma_{\mathcal{E},N,a}$. Hence by Lemma 6.3, we conclude that $\Phi(\mathcal{C})\sigma \sim \Phi(\mathcal{C}')\sigma'$.

The other inclusion of the symbolic equivalence $\approx_s^c$ is proved symmetrically. □

6.3.2 Normalised recipe

In the previous subsection, we have shown that we do not have to consider public names in the recipes for the intruder, which reduces a lot the search space for the symbolic equivalence of constraint systems. We will see in this subsection that we can also ignore some recipes that always provide redundant informations for the intruder. Typically, assume that ξ_1, ξ_2 are two recipes such that for all ground frame Φ, their applications always produce the same result, *i.e.* $\xi_1\Phi\downarrow = \xi_2\Phi\downarrow$, then these recipes will not help an intruder to distinguish one frame from another. Thus, we will not consider both recipes but only keep one.

Example 6.7. *Let ξ_1, ξ_2, ξ_3 be the recipes such that:*

1. $\xi_1 = \mathsf{proj}_1(\langle \xi_1', \xi_1'' \rangle)$ *for some* ξ_1', ξ_1'';
2. $\xi_2 = \mathsf{adec}(\mathsf{aenc}(\xi_2', \xi_2''))$ *for some* ξ_2', ξ_2'';
3. $\xi_3 = \mathsf{check}(\mathsf{senc}(\xi_3', \xi_3''))$ *for some* ξ_3', ξ_3''.

In fact, for all frames Φ, $\xi_1\Phi\downarrow = \xi_1'\Phi\downarrow$, $\xi_2\Phi\downarrow = \xi_2'\Phi\downarrow$ and $\neg\mathsf{Message}(\xi_3\Phi)$. Since this is true for any frame, they do not provide new informations for the intruder that might help him distinguishing two sets of constraint systems. Hence we will only consider ξ_1' and ξ_2' and ignore ξ_1, ξ_2, ξ_3.

Following the Example 6.7, we define a new set of recipes, denoted Π_n, and we show that it is complete and sound to only consider this new set of recipes when deciding the symbolic equivalence of sets of constraint systems.

Definition 6.5. *We say that $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{AX} \cup \mathcal{X}^2)$ is a* normalised recipe *if, and only if, for all $\mathsf{f}(\xi_1, \ldots, \xi_n) \in st(\xi)$, $\mathsf{f} \in \mathcal{F}_d$ implies that $\mathsf{root}(\xi_1) \notin \mathcal{F}_c$. We denote Π_n the set of normalised recipes.*

Even if the recipes in Π_n are called normalised recipes, they do not exactly correspond to the normalisation of a recipe by the rewriting system. More specifically $\xi \in \Pi_n$ implies $\xi\downarrow = \xi$ whereas the converse is not true.

Example 6.8. *Consider the recipe* $\xi = \mathsf{check}(\mathsf{senc}(ax_1, ax_2))$. ξ *is a recipe in normal form w.r.t. the rewriting system but* $\xi \notin \Pi_n$.

Intuitively, $\mathcal{T}(\mathcal{F}, \mathcal{AX}) \setminus \Pi_n$ represents all recipes that will either never provide the intruder with any new information or that will never satisfy the predicate $\mathsf{Message}(\cdot)$. The predicate $\mathsf{Message}(\cdot)$ is also easier to verify on the recipes in Π_n as stated in the following lemma. (Proof in Appendix C.1)

Lemma 6.5. *Let* Φ *be a ground frame such that for all* $(ax_i \rhd u_i) \in \Phi$, $\mathsf{Message}(u_i)$. *Let* $\xi \in \Pi_n$ *a ground recipe such that* $param(\xi) \in \mathrm{dom}(\Phi)$. $\xi\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ *if, and only if,* $\mathsf{Message}(\xi\Phi)$ *holds.*

The following lemma shows that the intruder do not lose messages of a frame by only considering the recipes in Π_n. (Proof in Appendix C.1)

Lemma 6.6. *Let* Φ *be a ground frame such that for all* $(ax_i \rhd u_i) \in \Phi$, $\mathsf{Message}(u_i)$. *Let* $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{AX})$. *If* $\mathsf{Message}(\xi\Phi)$ *then there exists a ground recipe* $\xi' \in \Pi_n$ *such that* $\xi\Phi\downarrow = \xi'\Phi\downarrow$, $\mathsf{Message}(\xi'\Phi)$ *and* $param(\xi') \subseteq param(\xi)$

We denote $\sim_{\Pi_n}$ the static equivalence that only consider the recipe in Π_n. We show in the next lemma that the static equivalence is preserved when considering the recipes in Π_n. (Proof in Appendix C.1)

Lemma 6.7. *Let* Φ *and* Φ' *be two ground frames such that for all* $(ax_i \rhd u_i) \in \Phi$ *(resp.* Φ'*),* $\mathsf{Message}(u_i)$. $\Phi \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'$ *is equivalent to* $\Phi \sim_{\Pi_n} \Phi'$.

Note that in the previous lemma, we omitted the set of private names $\mathcal{E}$. Since we do not consider names in the recipe when checking the static equivalence, this set is not needed anymore thus we may ignore it. We denote $\approx_s^{\Pi_n}$ the symbolic equivalence of sets of concrete constraint systems where we only consider solutions (σ, θ) where for all $X \in \mathrm{dom}(\theta)$, $X\theta \in \Pi_n$; and that rely on the static equivalence $\sim_{\Pi_n}$.

Lemma 6.8. *Let* Σ *and* Σ' *two sets of constraint systems that contain constraint systems with the same structure. The following property holds:*

$$\Sigma \approx_s^{\mathcal{T}(F,\mathcal{AX})} \Sigma' \text{ if, and only if, } \Sigma \approx_s^{\Pi_n} \Sigma'$$

Proof. The right implication of this equivalence ($\Rightarrow$) is rather simple. Indeed, for all $\mathcal{C} \in \Sigma$, for all $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$, if for all $X \in \mathrm{dom}(\theta)$, $X\theta \in \Pi_n$ then for all $X \in \mathrm{dom}(\theta)$, $X\theta \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$. Hence with $\Sigma \approx_s^{\mathcal{T}(F,\mathcal{AX})} \Sigma'$, it implies that there exists $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $\Phi\sigma \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma'$, where Φ (resp. Φ') is the frame of $\mathcal{C}$ (resp. $\mathcal{C}'$). By definition of a solution of a constraint system, $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ and $(\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ imply that for all $(ax_i \rhd u_i) \in \Phi\sigma$ (resp. $\Phi'\sigma'$), $\mathsf{Message}(u_i)$. Hence by Lemma 6.7, $\Phi\sigma \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma'$ implies $\Phi\sigma \sim_{\Pi_n} \Phi'\sigma'$ which allows us to conclude.

We now focus on the left implication of this equivalence ($\Leftarrow$). Let $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq) \in \Sigma$ and let $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ such that for all $X \in \mathrm{dom}(\theta)$, $X\theta \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$. We show by induction on k that there exists (σ', θ') such that:

- for all $(X, i \overset{?}{\vdash} x) \in D$,if $i \leq k$ then $(X\theta')(\Phi\sigma') = x\sigma'$, $\mathsf{Message}(x\sigma')$, $x\sigma'\downarrow = x\sigma\downarrow$, $X\theta' \in \Pi_n$ and $param(X\theta') \subseteq \{ax_1, \ldots, ax_i\}$
- for all $(ax_i \rhd u_i) \in \Phi$, if $i \leq k$ then $u_i\sigma\downarrow = u_i\sigma'\downarrow$ and $\mathsf{Message}(u_i\sigma')$.

Base case $k = 0$*:* This case is trivial since for all $(X, i \overset{?}{\vdash} x) \in D$, for all $(ax_j \rhd u_j) \in \Phi$, $i > 0$ and $j > 0$.

Inductive step $k > 0$*:* In such a case, thanks to our induction hypothesis, we know that there exists (σ', θ') satisfying the above properties for $k-1$. Hence, we show how to extend σ' and θ' for frame elements $(ax_k \rhd u_k) \in \Phi$ and deducible constraint $(X, k \overset{?}{\vdash} x) \in D$.

Let $(ax_k \rhd u_k) \in \Phi$. By definition of a constraint system, we know that for all $x \in vars^1(u_k)$, there exists $(X, i \overset{?}{\vdash} x) \in D$ such that $i < k$. By induction hypothesis, we know that $x\sigma'$ is defined, $\mathsf{Message}(x\sigma')$ and $x\sigma'{\downarrow} = x\sigma{\downarrow}$. Thus, we deduce that $u_k\sigma{\downarrow} = u_k(\sigma{\downarrow}){\downarrow} = u_k(\sigma'{\downarrow}){\downarrow} = u_k\sigma'{\downarrow}$. We now prove that $\mathsf{Message}(u_k\sigma')$. We already know by $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ that $\mathsf{Message}(u_k\sigma)$. Moreover, by our induction hypothesis, we also know that for all $x \in vars^1(u_k)$, $\mathsf{Message}(x\sigma')$. Therefore $\mathsf{Message}(u_k\sigma')$ is equivalent to for all $u \in st(u_k)$, $u\sigma'{\downarrow} \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. But $\mathsf{Message}(u_k\sigma)$ implies that $u\sigma{\downarrow} \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. At last, since for all $x \in vars^1(u) \subseteq vars^1(u_k)$, $x\sigma'{\downarrow} = x\sigma{\downarrow}$, we deduce that $u\sigma{\downarrow} = u\sigma'{\downarrow}$ and so $u\sigma'{\downarrow} \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Therefore, $\mathsf{Message}(u_k\sigma')$.

Let $(X, k \overset{?}{\vdash} x) \in D$. We have proved that for all $(ax_i \rhd u_i) \in \Phi$, if $i \leq k$ then $\mathsf{Message}(u_i\sigma')$, $\mathsf{Message}(u_i\sigma)$ and $u_i\sigma'{\downarrow} = u_i\sigma{\downarrow}$. Moreover, $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ implies that $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$. Hence $(X\theta)\Phi\sigma{\downarrow} = (X\theta)\Phi\sigma'{\downarrow}$; and $\mathsf{Message}((X\theta)(\Phi\sigma))$ implies that $\mathsf{Message}((X\theta)(\Phi\sigma'))$. Thus by Lemma 6.6, there exists $\xi \in \Pi_n$ such that $(X\theta)(\Phi\sigma'){\downarrow} = \xi(\Phi\sigma'){\downarrow}$, $\mathsf{Message}(\xi(\Phi\sigma'))$ and $param(\xi) \subseteq param(X\theta)$. We deduce that $param(\xi) \subseteq \{ax_1, \ldots, ax_i\}$, $\mathsf{Message}(\xi(\Phi\sigma'))$ and $\xi(\Phi\sigma'){\downarrow} = (X\theta)\Phi\sigma{\downarrow}$. By defining $X\theta' = \xi$ and $x\sigma' = \xi(\Phi\sigma')$, the result holds.

We have shown that there exists $(\sigma', \theta') \in \mathsf{Sol}_c(\mathcal{C})$ such that for all $X \in \mathrm{dom}(\theta')$, $X\theta \in \Pi_n$; and for all $x \in \mathrm{dom}(\sigma')$, $x\sigma{\downarrow} = x\sigma'{\downarrow}$. Thanks to our hypothesis $\Sigma \approx_s^{\Pi_n} \Sigma'$, we deduce that there exists $\mathcal{C}' = (\mathcal{E}; \Phi'; D'; Eq') \in \Sigma'$ and σ'' a substitution such that $(\sigma'', \theta') \in \mathsf{Sol}_c(\mathcal{C}')$ and $\Phi\sigma' \sim_{\Pi_n} \Phi'\sigma''$. By Lemma 6.7, we deduce that $\Phi\sigma' \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma''$. But for all $X \in \mathrm{dom}(\theta)$, $\mathsf{Message}((X\theta)(\Phi\sigma'))$, $\mathsf{Message}((X\theta')(\Phi\sigma'))$ and $X\theta\Phi\sigma'{\downarrow} = X\theta'\Phi\sigma'{\downarrow}$. Thus by $\Phi\sigma' \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma''$, we deduce that $\mathsf{Message}((X\theta)(\Phi'\sigma''))$ and $X\theta(\Phi'\sigma''){\downarrow} = X\theta'(\Phi'\sigma''){\downarrow}$.

Since $\Phi\sigma{\downarrow} = \Phi\sigma'{\downarrow}$ and for all $x \in vars^1(D)$, $x\sigma{\downarrow} = x\sigma'{\downarrow}$, then we deduce that for all $X \in \mathrm{dom}(\theta)$, $X\theta(\Phi\sigma){\downarrow} = X\theta(\Phi\sigma'){\downarrow} = X\theta'(\Phi\sigma'){\downarrow}$. Therefore, thanks to $\Phi\sigma' \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma''$, we deduce that $X\theta(\Phi'\sigma''){\downarrow} = X\theta'(\Phi'\sigma''){\downarrow}$. Using the similar reasoning as when we showed that $(\sigma', \theta') \in \mathsf{Sol}_c(\mathcal{C})$, we prove that there exists σ''' such that $(\theta, \sigma''') \in \mathsf{Sol}_c(\mathcal{C}')$ and for all $x \in \mathrm{dom}(\sigma'')$, $x\sigma'''{\downarrow} = x\sigma''{\downarrow}$. At last, since $\Phi\sigma{\downarrow} = \Phi\sigma'{\downarrow}$, $\Phi\sigma''{\downarrow} = \Phi\sigma'''{\downarrow}$, $\Phi\sigma' \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma''$ and for all $(ax_i \rhd u_i) \in \Phi\sigma$ (resp. $\Phi\sigma'$, $\Phi'\sigma''$, $\Phi'\sigma'''$), $\mathsf{Message}(u_i)$, we conclude that $\Phi\sigma \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma'''$. Thus the result holds. □

6.3.3 From concrete to constructor constraint systems

The definition of a solution of a concrete constraint system imposes that all first order terms in the constraint systems must be messages, except for terms in the inequations. Hence, when considering a solution (σ, θ) of a concrete constraint system, one could expect the terms in σ to be constructor terms. Moreover, given an equation $u \overset{?}{=} v$ in the constraint system, we will see that it is possible to check if this equation is unsatisfiable, or else to transform it into conjunction of equations that contain only constructor terms. Intuitively, by transforming the terms in our constraint systems into constructor terms, checking the predicate $\mathsf{Message}(\cdot)$ on the frame, equation and inequation will be much easier. Furthermore, since we only consider the recipes in Π_n for the solutions of our constraint systems, checking of the predicate $\mathsf{Message}(\cdot)$ will also become useless thanks to Lemma 6.5.

6.3.3.1 Narrowing

Example 6.9. *Consider the equation* $\mathsf{sdec}(x, a) \overset{?}{=} y$*. Let* σ *be a substitution on constructor terms such that* $\mathsf{Message}(\mathsf{sdec}(x\sigma, a))$*,* $\mathsf{Message}(y\sigma)$ *and* $\mathsf{sdec}(x\sigma, a){\downarrow} = y\sigma{\downarrow}$*.* $\mathsf{Message}(\mathsf{sdec}(x\sigma, a))$ *implies that* $\mathsf{sdec}(x\sigma, a){\downarrow} \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$*. Thus, it is only possible if there exists* u *such that* $x\sigma{\downarrow} = \mathsf{senc}(u, a)$*. Note that in this case,* $\mathsf{sdec}(x\sigma, a){\downarrow} = u$*.*

Hence, we want to replace the equation $\mathsf{sdec}(x,a) \stackrel{?}{=} y$ *by* $E = \left(x \stackrel{?}{=} \mathsf{senc}(x',a) \wedge x' \stackrel{?}{=} y\right)$ *where* x' *is a fresh variable. Note that since there is no more destructors in* E*, a substitution* σ' *on constructor terms satisfies* E *if and only if* $x\sigma' = \mathsf{senc}(x'\sigma', a)$ *and* $x_1\sigma' = y\sigma'$*. Thus, there is no need to check the predicate* $\mathsf{Message}(\cdot)$ *or even normalisation, but only the syntactic equality which easier and faster.*

In Example 6.9, we illustrate the transformation that we will apply on the equations in our constraint system. Simplifying the inequations of a constraint system requires the introduction of universal quantifiers.

Example 6.10. *Coming back to Example 6.9, we saw that* $\mathsf{sdec}(x,a) \stackrel{?}{=} y$ *can be replaced by the conjunction of equations* $x \stackrel{?}{=} \mathsf{senc}(x',a) \wedge x' \stackrel{?}{=} y$*. Actually,* x' *is (implicitly) existentially quantified: the formulas* $\mathsf{sdec}(x,a) \stackrel{?}{=} y$ *and* $\exists x'.x \stackrel{?}{=} \mathsf{senc}(x',a) \wedge x' \stackrel{?}{=} y$ *are equivalent*

Hence if we now consider the inequation $\mathsf{sdec}(x,a) \stackrel{?}{\neq} y$*, one can replace it by the following formula:*

$$\forall x'.x \stackrel{?}{\neq} \mathsf{senc}(x',a) \vee x' \stackrel{?}{\neq} y$$

We introduce the notion of *narrowing* [JK91] that formalises the transformations applied on equations and inequations illustrated in Examples 6.9 and 6.10.

Definition 6.6 (Narrowing)**.** *Let* $\mathcal{R}$ *be our set of rewriting rules. We define the narrowing as a binary relation on* $\mathcal{T}(\mathcal{F} \cup \mathcal{N}, \mathcal{X}^1)$*, denoted* $\rightarrowtail_{\mathcal{R}}$*, such that : for all* $s,t \in \mathcal{T}(\mathcal{F} \cup \mathcal{N}, \mathcal{X}^1)$*,* $s \stackrel{\sigma}{\rightarrowtail}_{\mathcal{R}} t$ *if, and only if, there exists a non variable position* p *of* s *and a renaming of a rewriting rule* $\ell \to r \in \mathcal{R}$ *such that* $\sigma = \mathsf{mgu}(\ell, s_{|p})$ *and* $t = s\sigma[r\sigma]_p$*.*

In order to simplify the proof of correctness, soundness and termination of the transformation, we will only use the innermost strategy for the narrowing.

Example 6.11. *Coming back to Example 6.9,* $\mathsf{sdec}(x,a) \stackrel{\sigma}{\rightarrowtail}_{\mathcal{R}} x'$ *where* $\sigma = \{x \mapsto \mathsf{senc}(x',a)\}$*.*

We extend the narrowing rule to a conjunction of equations between terms such that:

$$s \stackrel{?}{=} v \wedge E \quad \stackrel{\sigma}{\rightarrowtail}_{\mathcal{R}} \quad t \stackrel{?}{=} v' \wedge E'$$

where $s \stackrel{\sigma}{\rightarrowtail}_{\mathcal{R}} t$, $v' = v\sigma$ and $E' = E\sigma$.

From now on, we confuse a substitution $\sigma = \{x_1 \mapsto u_1, \ldots, x_n \mapsto u_n\}$ with the associated conjunction of equation $x_1 \stackrel{?}{=} u_1 \wedge \ldots \wedge x_n \stackrel{?}{=} u_n$. Moreover, we denote $\sigma \vDash_c u \stackrel{?}{=} v$ when $\mathsf{Message}(u\sigma)$, $\mathsf{Message}(v\sigma)$ and $u\sigma{\downarrow} = v\sigma{\downarrow}$. Similarly, we denote $\vDash_c u \stackrel{?}{\neq} v$ when $\neg\mathsf{Message}(u\sigma)$, or $\neg\mathsf{Message}(v\sigma)$ or $u\sigma{\downarrow} \neq v\sigma{\downarrow}$. We extend $\vDash_c$ naturally to conjunctions and disjunctions of equations and inequations with quantified variables. The following lemma shows the soundness and completeness the narrowing rule on conjunction of equations. (Proof in Appendix C.1)

Lemma 6.9. *Let* E *and* E' *be two conjunctions of equations between terms such that* $E \stackrel{\sigma}{\rightarrowtail}_{\mathcal{R}} E'$*. For all substitutions* τ *on ground constructor terms,*

- $\tau \vDash_c E'$ *implies that* $\sigma\tau \vDash_c E$
- $\tau \vDash_c E$ *implies that there exists* τ' *such that* $\tau = (\sigma\tau')|_{\mathrm{dom}(\tau)}$ *and* $\tau' \vDash_c E'$

We also extend the narrowing rule to formulas of the form $\forall \tilde{x} \bigvee_{j=1}^{m} u_j \stackrel{?}{\neq} v_j$ such that:

$$\forall \tilde{x}. \bigvee_{j=1}^{m} u_j \stackrel{?}{\neq} v_j \quad \rightarrowtail_{\mathcal{R}} \quad \forall \tilde{y}. \bigvee_{i=1}^{n} u'_i \stackrel{?}{\neq} v'_i$$

where $E = \bigwedge_{j=1}^{m} u_j \stackrel{?}{=} v_j$, $\sigma \wedge E' = \bigwedge_{i=1}^{n} u'_i \stackrel{?}{=} v'_i$, $E \stackrel{\sigma}{\rightarrowtail}_{\mathcal{R}} E'$, $\tilde{y} = \tilde{x} \cup (vars^1(\sigma) \setminus vars^1(E))$.

The soundness and completeness of this narrowing rule is shown in the following lemma. (Proof in Appendix C.1)

Lemma 6.10. *Let* $\phi = \forall \tilde{x} \bigvee_{j=1}^{m} u_j \stackrel{?}{\neq} v_j$ *and* $\phi' = \forall \tilde{y} \bigvee_{i=1}^{n} u'_j \stackrel{?}{\neq} v'_j$ *two formulas such that* $\phi \rightarrowtail_{\mathcal{R}} \phi'$. *For all substitutions* τ *of constructor terms,* $\tau \vDash_c \phi$ *if and only if* $\tau \vDash_c \phi'$.

6.3.3.2 Constructor constraint systems

We will use the narrowing on equations and inequations presented in the previous paragraph to build constraint systems that only contain constructor terms.

Definition 6.7 (Constructor constraint system)**.** *A constructive constraint system is either* $\bot$ *or a tuple* $(\Phi; D; Eq)$ *where:*

- Φ *is a sequence of the form* $\{ax_1 \rhd t_1, \ldots, ax_n \rhd t_n\}$ *where* t_i *are constructor terms and* ax_i *are variables in* $\mathcal{AX}$*;*
- D *is a set of deducibility constraints of the form* $X, i \stackrel{?}{\vdash} u$, *with* $i \leq n$, $X \in \mathcal{X}^2$ *and* u *is a constructor term.*
- Eq *is a conjunction of formulas of the form* $u \stackrel{?}{=} v$ *or* $\forall \tilde{y}. \bigvee_{j=1}^{m} u_j \stackrel{?}{\neq} v_j$ *where* u, v, u_j, v_j *are constructor terms.*

We also assume that following conditions are satisfied on a constructor constraint system:

1. each variable $X \in \mathcal{X}^2$ occurs at most once in D
2. for every $1 \leq k \leq n$, for every $x \in vars^1(t_k)$, there exists $(X, i \stackrel{?}{\vdash} u) \in D$ such that $x \in vars^1(u)$ and $i < k$.
3. for every free variable x of Eq, there exists $(X, i \stackrel{?}{\vdash} u) \in D$ such that $x \in vars^1(u)$.

Compared to Definition 4.3 , a deducibility constraint allows constructor terms as right hand terms, whereas in Definition 4.3 we only allowed variables. Furthermore, Eq is more general than the conjunction of equations and inequations in Definition 4.3. Note that we do not have the set of private names $\mathcal{E}$ in the definition of a constructor constraint system. Intuitively, we consider that all names that occur in a constructor constraint system are private and so $\mathcal{E}$ is omitted.

Example 6.12. *Let* $\mathcal{C}$ *be the concrete constraint system* $(\mathcal{E}; \Phi \cup \{ax_4 \rhd t\}; D; Eq)$ *where*

- $\mathcal{E} = \{sk_a, sk_{a'}, sk_b, n_b, n_a\}$,
- $\Phi = \{ax_1 \rhd \mathsf{pk}(sk_a), ax_2 \rhd \mathsf{pk}(sk_{a'}), ax_3 \rhd \mathsf{pk}(sk_b)\}$,
- $t = \mathsf{aenc}(\langle \mathsf{proj}_1(\mathsf{adec}(y, sk_b)), \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))$,
- $D = \{Z_1, 3 \stackrel{?}{\vdash} z_1 x;\ Y, 3 \stackrel{?}{\vdash} y;\ Z_2, 3 \stackrel{?}{\vdash} z_2\}$*; and*
- $Eq = \{z_1 \stackrel{?}{=} c;\ \mathsf{proj}_2(\mathsf{adec}(y, sk_b)) \stackrel{?}{=} \mathsf{pk}(sk_a);\ z_2 \stackrel{?}{=} c\}$.

The constructor constraint system $\mathcal{C}'$ *associated to* $\mathcal{C}$ *is the triplet* $(\Phi' \cup \{ax_5 \rhd t'\}; D'; Eq')$ *where*

- $\Phi' = \{ax_1 \rhd c, ax_2 \rhd \mathsf{pk}(sk_a), ax_3 \rhd \mathsf{pk}(sk_{a'}), ax_4 \rhd \mathsf{pk}(sk_b)\}$
- $D' = \{Z_1, 4 \stackrel{?}{\vdash} c;\ Y, 4 \stackrel{?}{\vdash} \mathsf{aenc}(\langle y', \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b));\ Z_2, 4 \stackrel{?}{\vdash} c\}$
- $t' = \mathsf{aenc}(\langle y', \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))$
- $Eq = \top$.

Note that as mentioned in Subsection 6.3.1, we put the free name c in the frame.

We will denote $\sigma \vDash u \stackrel{?}{=} v$ when $u\sigma = v\sigma$. Moreover, we denote $\sigma \vDash u \stackrel{?}{\neq} v$ when $u\sigma \neq v\sigma$. We extend $\vDash$ naturally to conjunctions and disjunctions of (in)equations with quantified variables.

Definition 6.8 (solution of a constructor constraint system)**.** *A* solution *of a constructor constraint system* $\mathcal{C} = (\Phi; D; Eq)$ *consists of a mapping* σ *from* $vars^1(D)$ *to to ground constructor terms and a substitution* θ *mapping* $vars^2(D)$ *to ground recipes such that:*

— *for every* $(X, k \overset{?}{\vdash} u) \in D$, *we have* $X\theta \in \Pi_n$, $(X\theta)(\Phi\sigma)\downarrow = x\sigma$ *and* $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$.

— $\sigma \vDash Eq$

We denote $\mathsf{Sol}(\mathcal{C})$ *the set of solutions of* $\mathcal{C}$.

Example 6.13. *Coming back to the Example 6.12,* $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}')$ *where:*

— $\sigma = \{y' \mapsto \mathsf{h}(c)\}$

— $\theta = \{Z_1 \mapsto ax_1, Y \mapsto \mathsf{aenc}(\langle \mathsf{h}(ax_1), ax_2 \rangle, ax_4), Z_2 \mapsto ax_1\}$

We can adapt the definition of static equivalence and symbolic equivalence to constructor frames and constraint systems.

Definition 6.9 (static equivalence)**.** *Two ground constructor frames* Φ *and* Φ' *are statically equivalent, denoted* $\Phi \sim \Phi'$, *if and only if* $\mathrm{dom}(\Phi) = \mathrm{dom}(\Phi')$ *and for all* $\xi, \xi \in \Pi_n$, *if* $param(\{\xi, \xi'\}) \subseteq \mathrm{dom}(\Phi)$ *then*

— $\xi\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ *is equivalent to* $\xi'\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$

— *if* $\xi\Phi\downarrow, \xi'\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$, *then* $\xi\Phi\downarrow = \xi'\Phi\downarrow$ *is equivalent to* $\xi\Phi'\downarrow = \xi'\Phi'\downarrow$.

Definition 6.10 (symbolic equivalence)**.** *Two sets* Σ *and* Σ' *of constructor constraint systems having the same structure are symbolically equivalent, denoted* $\Sigma \approx_s \Sigma'$, *if and only if for all* $\mathcal{C} \in \Sigma$, *for all* $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, *there exists* $\mathcal{C}' \in \Sigma'$ *and a substitution* σ' *such that* $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ *and* $\Phi\sigma \sim \Phi'\sigma'$ *(and conversely for any* $\mathcal{C}' \in \Sigma'$*) where* Φ *and* Φ' *are the respective frames of* $\mathcal{C}$ *and* $\mathcal{C}'$.

As previously mentioned, we show the relation between symbolic equivalence of sets of constructor constraint system and the symbolic equivalence $\approx_s^{\Pi_n}$ of sets of concrete constraint system. (Proof in Appendix C.1)

Lemma 6.11. *Let* Σ_1, Σ_1' *two sets of concrete constraint systems having the same structure. There exist* Σ_2, Σ_2' *two sets of constructor constraint systems having the same structure such that:*

$$\Sigma_1 \approx_s^{\Pi_n} \Sigma_1' \text{ if and only if } \Sigma_2 \approx_s \Sigma_2'$$

We summarise what we proved along this section in the following theorem.

Theorem 6.2. *Given a decision procedure for the symbolic equivalence of sets of constructor constraint systems, the problem of trace equivalence between two concrete processes is decidable.*

The algorithm proceeds as follows: Given A and B two concrete processes with the same set of names. We first build the two concrete processes A' and B', that do not consider public names, by following Lemma 6.1. Second, we compute the sets $\mathsf{trace}_{cs}(A')$ and $\mathsf{trace}_{cs}(B')$ (possible since A' and B' do not contain replication). Third, we compute the set S of pair of sets of constructor constraint systems such that:

— if $(\mathsf{tr}, \Sigma) \in \mathsf{trace}_{cs}(A')$, $(\mathsf{tr}', \Sigma') \in \mathsf{trace}_{cs}(B')$ and $\mathsf{tr} = \mathsf{tr}'$ then $(\Sigma, \Sigma') \in S$;

— if $(\mathsf{tr}, \Sigma) \in \mathsf{trace}_{cs}(A')$ (resp. $\mathsf{trace}_{cs}(B')$) and for all $(\mathsf{tr}', \Sigma') \in \mathsf{trace}_{cs}(B')$ (resp. $\mathsf{trace}_{cs}(A')$), $\mathsf{tr} \neq \mathsf{tr}'$ then $(\Sigma, \emptyset) \in S$

Four, for every $(\Sigma, \Sigma') \in S$, we compute the sets of constructor constraint systems Σ_1 and Σ_1' from Σ and Σ' respectively by following the proof of Lemma 6.11. At last we apply the decision procedure for symbolic equivalence of sets of constructor constraint systems on Σ_1 and Σ_1'.

The soundness and completeness of this algorithm are ensured by Theorem 6.1 and Lemmas 6.1, 6.4, 6.8 and 6.11.

Note that adding $(\Sigma, \emptyset)$ in the set S does not necessary imply that $A \not\approx_t B$. Indeed, it is possible that all constraint systems in Σ do not have any solution and in such a case, we would have $\Sigma \approx_s^c \emptyset$.

The next chapter is devoted to a decision procedure for symbolic equivalence between two sets of constructor constraint systems.

Chapter 7

A decision procedure for symbolic equivalence

Contents

This chapter is dedicated to the presentation of a decision algorithm for the symbolic equivalence between two sets of constructor constraint systems. The general idea of our decision algorithm is borrowed from earlier work on deducibility constraints: we simplify the constraints until we get a simple form, on which the equivalence problem should be easy. Since we consider pairs of (sets of) constraint systems, the simplification rules should be applied on both (sets of) systems at the same time; when this corresponds to guessing an attacker action, it should be the same rule, which is applied on both (sets of) systems. The second main difference concerns the equivalence checking: we have to keep track of an extended frame, recording some of the deductions of the attacker, and check the static equivalence of all instances, when the constraints are in solved form.

W.r.t. the previous constraint solving algorithms, there are many additional difficulties, which we will point along the chapter. One of the problems is that, when applying the rules in a naive way, the two constraint systems do not necessarily reach a solved form at the same time. So, we may need to apply further rules, even when one of the systems is in solved form, which causes termination issues.

Finally, along the algorithm, we guess for instance whether or not a key is deducible. This introduces negative deducibility constraints, which might be hard to solve. We turn around the difficulty, keeping track of previous choices (e.g., whether a key was deducible or not). This yields matrices of constraint systems: the rows correspond to sets of constraint systems, that share the same structure, but may yield different outputs of the protocol, and different rows correspond to

different guesses of deducibility along the constraint simplification. This complication in the syntax allows some simplifications in the algorithm, since we may take advantage of the bookkeeping of different rows.

Since the correctness, completeness and termination of our algorithm are rather technical, we delayed most of the proofs in the appendices, focusing on the algorithm itself. In the section 7.1, we introduce most of the definitions together with a few examples. The algorithm is explained in the section 7.2. We start with single constraint systems, before extending the rules to pairs of (sets of) constraint systems, and later matrices of constraint systems. Chapter 8 is dedicated to the proof of soundness, completeness and termination of the algorithm.

7.1 Preliminaries

As mentioned earlier, the general idea of our algorithm consists of simplifying the constraint system by applying simplification rules that guess for example an attacker action, or whether a key is deducible or not, ... During the execution of these guesses, we need to be able to keep track of the results of these guesses. However, the definition of a constructor constraint system (Definition 6.7) is too restrictive to contain all the informations we need. For example, while guessing that a key sk is deducible might be represented by adding a deducible constraint $X, i \overset{?}{\vdash} sk$, there is no way to represent, in a constraint system that follows Definition 6.7, that the key sk is not deducible. Therefore, we extend in this section the definitions of constructor constraint systems, frames, ...

7.1.1 Extended frame

In previous chapters, a frame is used to record the sequence of messages (or terms in a symbolic execution) that have been sent by the participants of the protocol. We extend this notion to record some additional informations on attacker's deductions. Typically $\mathsf{sdec}(X, \zeta), i \rhd u$ records that, using a decryption with the recipe ζ, on top of a recipe X, allows one to get u (at stage i). After recording this information in the frame, we may rely on this bookkeeping, and no longer consider a decryption on top of X.

Definition 7.1 (extended frame)**.** *An* extended frame Φ *(resp. a* closed extended frame*) is a sequence* $\{\zeta_1, i_1 \rhd u_1; \ldots; \zeta_n, i_n \rhd u_n\}$ *where:*

- $u_1, \ldots, u_n$ *are constructor terms (resp. ground constructor terms),*
- $i_1, \ldots, i_n$ *are integers, and*
- $\zeta_1, \ldots, \zeta_n$ *are distinct general recipes (resp. ground recipes).*

The domain *of the extended frame* Φ *is* $\mathrm{dom}(\Phi) = \mathcal{AX} \cap \{\zeta_1, \ldots, \zeta_n\}$*. It must be equal to* $\{ax_1, \ldots, ax_m\}$ *for some* m*.* m *is called the* size *of* Φ*. Moreover, an extended frame* Φ *must satisfy that for all* $(\zeta, i \rhd u) \in \Phi$*, if* $\zeta = ax_j$ *for some* j*, then* $i = j$*.*

The indices $i_1, \ldots, i_n$ represent the stages at which a message is known. An attacker could indeed distinguish two processes, simply because some message can be computed earlier in one of the process than in the other: the stage at which messages are available is a relevant information.

For a constructor frame $\Phi = \{ax_1 \rhd u_1; \ldots; ax_m \rhd u_m\}$, the stage is represented by the index of the parameter in $\mathcal{AX}$. Hence, the extended frame obtained from Φ is the following: $\{ax_1, 1 \rhd u_1; \ldots; ax_m, m \rhd u_m\}$. Conversely, for an extended frame Φ of size m, We denote $\mathsf{Init}(\Phi)$ the constructor frame $\{ax_1 \rhd u_1; \ldots; ax_m \rhd u_m\}$ where for all $i \in \{1, \ldots, m\}$, $(ax_i, i \rhd u_i) \in \Phi$.

An extended frame Φ of size m defines a substitution on $\mathrm{dom}(\Phi)$: if $\mathrm{dom}(\Phi) = \{ax_1, \ldots, ax_m\}$ and, for $i = 1, \ldots, m$, $ax_i, j_i \rhd v_i$, then we write again Φ the substitution $\{ax_1 \mapsto v_1, \ldots, ax_m \mapsto v_m\}$. Note that the substitution Φ and $\mathsf{Init}(\Phi)$ are the same. A closed extended frame Φ is *consistent* if, for every $(\zeta, i \rhd u) \in \Phi$, $(\zeta\Phi)\!\downarrow = u$.

Example 7.1. $\Phi = \{ax_1, 1 \rhd \mathsf{senc}(a, b); ax_2, 2 \rhd b; \mathsf{sdec}(ax_1, ax_2), 2 \rhd a; ax_3, 3 \rhd a\}$ *is a closed extended frame; the intermediate component records the deduction of* a *using the recipe* $\mathsf{sdec}(ax_1, ax_2)$ *as early as stage 2. Moreover,* $\mathsf{Init}(\Phi) = \{ax_1 \rhd \mathsf{senc}(a, b); ax_2 \rhd b; ax_3 \rhd a\}$.

Given two extended frames Φ and Φ', we say that Φ and Φ' are statically equivalent, denoted $\Phi \sim \Phi'$, if and only if their associated constructor frames $\mathsf{Init}(\Phi)$ and $\mathsf{Init}(\Phi')$ are statically equivalent, *i.e.* $\mathsf{Init}(\Phi) \sim \mathsf{Init}(\Phi')$.

From now on, we will always work on extended frames, and call them simplify "frame".

7.1.2 Extended constraint systems

As explained in Chapter 6, our constraint systems need not only to represent sets of traces, but also record some information on the attacker's actions that led to these traces. That is why we also include equations between recipes and a set NoUse of obsolete elements in the frame; roughly, a component of the frame is obsolete when the attacker used another recipe to get the message, at an earlier stage (as in the example 7.1). Finally, we also consider negated constraints, in order to enable splitting the set of traces into disjoint sets.

Definition 7.2 (extended constraint system)**.** *An* extended constraint system *is either* $\bot$ *or a tuple* $(S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ *where:*

- S_1 *(resp.* S_2*) is a set of variables in* $\mathcal{X}^1$ *(resp.* $\mathcal{X}^2$*);*
- Φ *is a frame, whose size is some* n *and* NoUse *is a subset of* Φ*;*
- D *is a set of deducible constraints of the form* $X, i \overset{?}{\vdash} u$ *with* $1 \leq i \leq n$*,* $X \in \mathcal{X}^2$ *and* u *is a constructor term.*
- Eq *is a conjunction of formulas of the form* $u \overset{?}{=} v$ *or* $\forall \tilde{y} \cdot [\bigvee_{j=1}^{m} u_j \overset{?}{\neq} v_j]$ *where* u, v, u_j, v_j *are constructor terms.*
- Er *is a conjunction of formulas of the form* $\zeta \overset{?}{=} \zeta'$*, or* $\xi \overset{?}{\neq} \xi'$ *or* $\mathsf{root}(\beta) \overset{?}{\neq} \mathsf{f}$ *where* ζ*,* ζ'*,* ξ*,* ξ' *and* β *are recipe in* Π_n*, and* f *is a constructor symbol.*
- ND *is a conjunction of formulas of the form* $\forall \tilde{x}.[u \overset{?}{\neq} v \vee \bigvee_j k \overset{?}{\not\vdash} w_j]$ *where* u, v, w_j *are constructor terms and* $k \in \mathbb{N}$*.*

We say that an extended constraint system is initial *if* $\mathsf{NoUse} = \emptyset$*,* $ND = \top$*,* $Er = \top$*,* $S_2 = vars^2(D)$*,* $S_1 = vars^1(D)$*,* $vars^1(Eq) \subseteq vars^1(D)$ *and for all* $(\xi, i \rhd u) \in \Phi$*,* $\xi = ax_i$*.*

Intuitively, S_1 is the set of free variables in $\mathcal{X}^1$; we may have to introduce auxiliary variables, that will be (implicitly) existentially quantified, as well as (explicitly) universally quantified variables. Similarly, S_2 is a set of principal recipe variables (in $\mathcal{X}^2$) of the constraint. For readability, we will sometimes omit some of the components of the constraint system, because they are either straightforward from the context or empty.

We also assume the following conditions are satisfied on an extended constraint system:

1. each variable $X \in vars^2(D)$ occurs at most once in D;
2. for every $(\xi, i \rhd u) \in \Phi$, for every $x \in vars^1(u)$, there exists $(X, j \overset{?}{\vdash} v) \in D$ such that $x \in vars^1(v)$ and $j < i$;
3. for every $(\xi, i \rhd u) \in \Phi$, $param(\xi) \subseteq \{ax_1, \ldots, ax_i\}$ and for all $X \in vars^2(\xi)$, there exists $(X, j \overset{?}{\vdash} v) \in D$ such that $j \leq i$.

The second property corresponds to the origination property on first order variable whereas the third property represents the origination property for second order variables. Note that an initial extended constraint systems corresponds in fact to a constructor constraint system where S_1 and S_2 represent all the variables in the constraint systems.

The *structure* of an extended constraint system $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ is given by S_2, Er, $\{(X, i) \mid X, i \overset{?}{\vdash} u \in D\}$, $\{(\xi, i) \mid \xi, i \rhd u \in \Phi\}$ and $\{(\xi, i) \mid \xi, i \rhd u \in \mathsf{NoUse}\}$. Similarly to the structure of constructor constraint system, the structure of an extended constraint system indicates from which symbolic trace the constraint system is coming from but also indicates the guesses on the attacker's actions.

Example 7.2. *Coming back to Example 6.12, the extended constraint system $\mathcal{C}''$ corresponding to the constructor constraint system $\mathcal{C}'$ is the tuple $(S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ where*

— $\Phi = \{ax_1, 1 \rhd c, ax_2, 2 \rhd \mathsf{pk}(sk_a), ax_3, 3 \rhd \mathsf{pk}(sk_{a'}), ax_4, 4 \rhd \mathsf{pk}(sk_b); ax_5, 5 \rhd t\}$

— $D = \{Z_1, 4 \overset{?}{\vdash} c;\ Y, 4 \overset{?}{\vdash} \mathsf{aenc}(\langle y', \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b));\ Z_2, 4 \overset{?}{\vdash} c\}$

— $t = \mathsf{aenc}(\langle y', \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))$

with $S_1 = \{y'\}$, $S_2 = \{Z_1, Y, Z_2\}$, $Eq = Er = ND = \top$ and $\mathsf{NoUse} = \emptyset$.

More examples will be given later.

In order to define what is a solution for such extended constraint systems, we have to give the semantics of the formulas ND, Eq and Er, and also to introduce the notion of path in order to cope with our generalised notion of frame. The formulas ND, Er, Eq are logic formulas built upon elementary formulas using classical connectives. The semantics for the elementary formulas are given below and is extended as expected to general formulas.

Definition 7.3 (solutions of side constraints)**.** *Let $\mathcal{C}$ an extended constraint system with Φ its associated frame. Let θ be a substitution mapping $vars^2(\mathcal{C})$ to ground recipes, and σ be a substitution mapping $vars^1(\mathcal{C})$ to ground constructor terms.*

— $\sigma \vDash (i \overset{?}{\nvdash} u)$ *if and only if $\xi(\Phi\sigma)\!\downarrow \neq u\sigma\!\downarrow$ for any ground recipe $\xi \in \Pi_n$ with $param(\xi) \subseteq \{ax_1, \ldots, ax_i\}$.*

— $\sigma \vDash u \overset{?}{=} v$ *(resp. $u \overset{?}{\neq} v$), if and only if $u\sigma = v\sigma$ (resp. $u\sigma \neq v\sigma$).*

— $\theta \vDash \xi_1 \overset{?}{=} \xi_2$ *(resp. $\theta \vDash \xi_1 \overset{?}{\neq} \xi_2$) if and only if $\xi_1\theta = \xi_2\theta$ (resp. $\xi_1\theta \neq \xi_2\theta$) .*

— $\theta \vDash \mathsf{root}(\xi) \overset{?}{\neq} \mathsf{f}$ *if and only if $\mathsf{root}(\xi\theta) \neq \mathsf{f}$*

Note that the semantics of the formulas Eq are identical to the ones used for constructor constraint system.

Example 7.3. *Let $\Phi = \{ax_1, 1 \rhd \mathsf{senc}(a, x), ax_2, 2 \rhd b\}$ and $\sigma = \{x \mapsto b\}$. Then $\sigma \vDash (1 \overset{?}{\nvdash} a)$ and $\sigma \nvDash (2 \overset{?}{\nvdash} a)$.*

There are possibly several ways to compute the same message, given a frame. All possibles ways of computing such a message are the observable identities that are used in the static equivalence. We need only to consider an arbitrary (but fixed) way of computing a message, as early as possible, the other possible computations being captured by the static equivalence. In other words, if we want to check the symbolic trace equivalence, we need only to consider on the one hand the "canonical" recipes that are used by the attacker, and, on the other hand, the static equivalence of the frames. Assuming such a "canonical" recipe that computes a given message simplifies our proofs.

Now, we can choose our recipe according to its *path*, which is the sequence of destructors applied on the leftmost arguments. This sequence determines the result, regardless of other arguments. Let us precise this point.

Definition 7.4 (path)**.** *Let $\xi \in \Pi_n$ be such that $\mathsf{root}(\xi) \notin \mathcal{F}_c$. The* path *of ξ, denoted $\mathsf{path}(\xi)$, is a word in $\mathcal{F}_d^* \cdot (\mathcal{AX} + \mathcal{X}^2)$ that is recursively defined as follows:*

— $\mathsf{path}(ax) = ax$ *when $ax \in \mathcal{AX}$,*

— $\mathsf{path}(X) = X$ *when $X \in \mathcal{X}^2$, and*

— $\mathsf{path}(\mathsf{f}(\xi_1, \ldots, \xi_n)) = \mathsf{f} \cdot \mathsf{path}(\xi_1)$.

Example 7.4. *Let $\xi = \mathsf{sdec}(\mathsf{sdec}(ax_2, ax_1), \mathsf{sdec}(ax_1, ax_2))$ and $\xi' = \mathsf{sdec}(\mathsf{sdec}(ax_2, \mathsf{sdec}(ax_3, ax_4)), ax_4)$. $\mathsf{path}(\xi) = \mathsf{path}(\xi') = \mathsf{sdec} \cdot \mathsf{sdec} \cdot ax_2$.*

NoUse is a subset of the frame whose use is forbidden, because we changed the canonical recipe. This happens only in the course of our algorithm when we discover that a message can actually be computed at an earlier stage. The following defines the restrictions on the recipes that we consider.

Definition 7.5 (ξ conforms to Φ)**.** *Let Φ be a closed frame, NoUse be a subset of Φ, and ξ be a ground recipe in Π_n. We say that ξ* conforms to *the frame Φ w.r.t. NoUse if :*

— $\forall \zeta \in st(\xi), \forall (\zeta', i \rhd u) \in \Phi, \mathsf{path}(\zeta) = \mathsf{path}(\zeta') \Rightarrow \zeta = \zeta'$.

— $\forall (\zeta, i \rhd u) \in \mathsf{NoUse}$, $\zeta \notin st(\xi)$

Example 7.5. *Let $\Phi = \{ax_1, 1 \rhd \langle a, b\rangle; ax_2, 1 \rhd \mathsf{senc}(a, b); \mathsf{sdec}(ax_2, \mathsf{proj}_2(ax_1)), 1 \rhd a; ax_3, 2 \rhd b; ax_4, 2 \rhd a; ax_5, 3 \rhd \mathsf{senc}(c, a)\}$ and $\mathsf{NoUse} = \{ax_4, 2 \rhd a\}$.*

Implicitly here, we chose a canonical way to compute a, that is recorded in the frame: it consists in decrypting the second message with the second component of the first one. There are other ways of computing a, for instance using the first projection on the first message or using the fourth message. NoUse forbids however this last use (ax_4 is not a recipe that conforms to Φ w.r.t. NoUse). This can be generated when an earlier (at stage 1 instead of stage 2) computation of a is detected and recorded in the frame.

Furthermore, the recipe $\xi = \mathsf{sdec}(ax_2, ax_3)$ does not conform to Φ, while the recipe $\mathsf{sdec}(ax_5, \mathsf{sdec}(ax_2, \mathsf{proj}_2(ax_1)))$ conforms to Φ w.r.t. NoUse.

Note that given an initial frame Φ, *i.e.* $\mathsf{Init}(\Phi) = \Phi$ and the set $\mathsf{NoUse} =$, every ground recipe in Π_n conforms to Φ w.r.t. NoUse.

Definition 7.6 (context w.r.t. Φ)**.** *Let Φ be a frame and ξ be a recipe in Π_n. The* context *of ξ w.r.t. Φ, denoted $\mathsf{C}\lfloor\xi\rfloor_\Phi$, is a term in $\mathcal{T}(\mathcal{F}, \mathcal{F}_d^* \cdot \mathcal{AX} \cup \mathcal{X}^2)$ and is defined recursively as follows:*

— $\mathsf{C}\lfloor\xi\rfloor_\Phi = \mathsf{path}(\xi)$ *if there exists $(\xi', i \rhd u) \in \Phi$ such that $\mathsf{path}(\xi) = \mathsf{path}(\xi')$;*

— $\mathsf{C}\lfloor\xi\rfloor_\Phi = \xi$ *if $\xi \in \mathcal{X}^2$;*

— $\mathsf{C}\lfloor\xi\rfloor_\Phi = \mathsf{f}(\mathsf{C}\lfloor\xi_1\rfloor_\Phi, \ldots, \mathsf{C}\lfloor\xi_n\rfloor_\Phi)$ *if $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$.*

For sake of clarity, when Φ is clear from the context, we denote it by $\mathsf{C}\lfloor\xi\rfloor$.

The context of a recipe w.r.t. a frame represents part of the recipe that are not directly defined by the frame.

Definition 7.7 (direct access mappings)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. We define the* direct access mapping *of $\mathcal{C}$, denoted $\mathsf{acc}^1(\mathcal{C})$ (resp. $\mathsf{acc}^2(\mathcal{C})$), to be a mapping from $(\mathcal{F}_d^* \cdot \mathcal{AX}) \cup \mathcal{X}^2$ to constructor terms (resp. recipe in Π_n) where:*

— *for all $(X, i \overset{?}{\vdash} u) \in D$, we have that $X\mathsf{acc}^1(\mathcal{C}) = u$ (resp. $X\mathsf{acc}^2(\mathcal{C}) = X$)*

— *for all $(\xi, i \rhd u) \in \Phi$, we have that $\mathsf{path}(\xi)\mathsf{acc}^1(\mathcal{C}) = u$ (resp. $\mathsf{path}(\xi)\mathsf{acc}^2(\mathcal{C}) = \xi$).*

Example 7.6. *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system where $\mathsf{NoUse} = \emptyset$, $D = \{X, 1 \overset{?}{\vdash} \langle x, a\rangle \ ; \ Y, 2 \overset{?}{\vdash} b\}$, $\Phi = \{ax_1, 1 \rhd \mathsf{senc}(a, b) \ ; \ ax_2, 2 \rhd b \ ; \ \mathsf{sdec}(ax_1, Y), 2 \rhd a\}$. Let ξ_1, ξ_2 and ξ_3 three recipes such that $\xi_1 = \langle X, Y\rangle$, $\xi_2 = \mathsf{sdec}(ax_1, Y)$ and $\xi_3 = \mathsf{senc}(X, \mathsf{sdec}(ax_1, ax_2))$. We have :*

— $\mathsf{C}\lfloor\xi_1\rfloor = \langle X, Y\rangle$ *and* $\mathsf{C}\lfloor\xi_1\rfloor\mathsf{acc}^1(\mathcal{C}) = \langle\langle x, a\rangle, b\rangle$

— $\mathsf{C}\lfloor\xi_2\rfloor = \mathsf{sdec} \cdot ax_1$ *and* $\mathsf{C}\lfloor\xi_2\rfloor\mathsf{acc}^1(\mathcal{C}) = a$

— $\mathsf{C}\lfloor\xi_3\rfloor = \mathsf{senc}(X, \mathsf{sdec} \cdot ax_1)$ *and* $\mathsf{C}\lfloor\xi_3\rfloor\mathsf{acc}^1(\mathcal{C}) = \mathsf{senc}(\langle x, a\rangle, a)$

Note that for all ground recipe ξ conforms to a ground frame Φ, $\mathsf{C}\lfloor\xi\rfloor_\Phi\mathsf{acc}^2(\mathcal{C}) = \xi$. This illustrates the fact that the context of a recipe represents the canonical way to represent a recipe. Moreover, if Φ is consistent then $\mathsf{C}\lfloor\xi\rfloor_\Phi\mathsf{acc}^1(\mathcal{C})\downarrow = \xi\Phi\downarrow$.

Definition 7.8 ((pre-)solution)**.** *A* solution *of* $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ *consists of a substitution* σ *mapping* $vars^1(\mathcal{C})$ *to ground constructor terms and a substitution* θ *mapping* $vars^2(\mathcal{C})$ *to ground recipes in* Π_n*, such that:*

1. *for every* $X \in vars^2(\mathcal{C})$*,* $X\theta$ *conforms to* $\Phi\theta$ *w.r.t.* NoUse*;*
2. *for every* $(X, i \overset{?}{\vdash} u) \in D$*,* $X\theta(\Phi\sigma)\downarrow = u\sigma\downarrow$ *and* $param(X\theta) \subseteq \{ax_1, \ldots, ax_i\}$*;*
3. $\sigma \vDash ND \wedge Eq$ *and* $\theta \vDash Er$*.*

We denote by $\mathsf{Sol}(\mathcal{C})$ *the set of solutions of* $\mathcal{C}$*. By convention,* $\mathsf{Sol}(\bot) = \emptyset$*. A pair* (σ, θ) *of substitutions that only satisfy the two first items is a* pre-solution *of* $\mathcal{C}$*.*

From the definition of solutions of a constructor constraint system (Definition 6.8), we added the conditions on the conformity of the $X\theta$ w.r.t. the frame and the satisfiability of ND and Er.

Example 7.7. *Consider the constraint of the example 7.2.* $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}')$ *where:*

— $\sigma = \{y' \mapsto \mathsf{h}(c)\}$

— $\theta = \{Z_1 \mapsto ax_1, Y \mapsto \mathsf{aenc}(\langle \mathsf{h}(ax_1), ax_2 \rangle, ax_4), Z_2 \mapsto ax_1\}$

Example 7.8. *Consider the frame of the example 7.5, together with* $D = \{X, 3 \overset{?}{\vdash} \mathsf{senc}(x, a)\}$*,* $Eq = x \overset{?}{\neq} a \wedge x \overset{?}{\neq} b \wedge \forall y_1, y_2.(x \overset{?}{\neq} \langle y_1, y_2 \rangle) \wedge \forall y_1, y_2.(x \overset{?}{\neq} \mathsf{senc}(y_1, y_2))$ *and* $ND = Er = \bot$*. One possible solution is* $\sigma = \{x \mapsto c\}, \theta = \{X \mapsto \mathsf{sdec}(ax_5, \mathsf{sdec}(ax_2, \mathsf{proj}_2(ax_1)))\}$

From now on, we will only work with extended constraint system. Thus, similarly to frames, we will call constraint system an extended constraint system. Given a constraint system $\mathcal{C}$, we will sometimes denote $S_2(\mathcal{C})$, $S_1(\mathcal{C})$, $\Phi(\mathcal{C})$, $D(\mathcal{C})$, ..., $\mathsf{NoUse}(\mathcal{C})$ the different components of $\mathcal{C}$.

7.2 Simplifying a constraint system

As explained in the introduction, our algorithm that decides the equivalence of sets of constraint systems is based on tranformations of such systems until a solved formed is reached. We start by defining and explaining these rules on a single constraint system and then explain how it is extended to pairs of sets of constraint systems (see Section 7.3.1) and later to pairs of matrices of constraint systems (see Section 7.3.2).

The transformation rules are displayed in Figure 7.1 and Figure 7.2 for a single constraint system.

7.2.1 The transformation rules

A simple idea would be to guess the top function symbol of a recipe and replace the recipe variable with the corresponding instance. When the head symbol of a recipe is a constructor and the corresponding term is not a variable, this is nice, since the constraint becomes simpler. This is the purpose of the rule Cons. When the top symbol of a recipe is a destructor, the constraint becomes more complex, introducing new terms, which yields non-termination. Our strategy is different. We do guess the top symbol of a recipe, when it is a constructor (or a parameter ax_i). Otherwise, we switch from the top position of the recipe to the *redex position*. Typically, in case of symmetric encryption, if a ciphertext is in the frame, we will guess whether the decryption key is deducible, and at which stage.

In the transformation rules described below, we only write the components of the constraint systems that are modified during an application of an instance of a rule.

The Cons rule simply guesses whether the top symbol of the recipe is a constructor f. Either it is, and then we can split the constraint, or it is not and we add a insequation on the recipe forbidding it to start with f.

In all the following examples, we apply eagerly some simplifications (such simplifications are formalised and explained in the section 7.2.2), and omit irrelevant parts of the constraint, in order to improve the readability.

$$\underline{\textsc{Cons}(X,\mathsf{f})}: \quad S_2; X, i \overset{?}{\vdash} t; Eq; Er \quad \leadsto \quad \begin{array}{l} S_2'; X_1, i \overset{?}{\vdash} x_1; \cdots; X_n, i \overset{?}{\vdash} x_n; \\ Eq \wedge t \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n); Er \wedge X \overset{?}{=} \mathsf{f}(X_1, \ldots, X_n) \\ X, i \overset{?}{\vdash} t \ ; \ Eq; \ Er \wedge \mathsf{root}(X) \neq \mathsf{f} \end{array}$$

where $x_1, \ldots, x_n, X_1, \ldots X_n$ are fresh variables, and
$S_2' = S_2 \cup \{X_1, \ldots, X_n\}$ if $X \in S_2$ and $S_2' = S_2$ otherwise.

$$\underline{\textsc{Axiom}(X,\mathsf{path})}: \quad \Phi; \ X, i \overset{?}{\vdash} u; \ Eq; \ Er \quad \leadsto \quad \begin{array}{l} \Phi; \ Eq \wedge u \overset{?}{=} v; \ Er \wedge X \overset{?}{=} \xi \\ \Phi; \ X, i \overset{?}{\vdash} u; \ Eq; \ Er \wedge X \overset{?}{\neq} \xi \end{array}$$

If Φ contains $\xi, j \rhd v$ with $i \geq j$, $\mathsf{path}(\xi) = \mathsf{path}$ and $(\xi, j \rhd v) \notin \mathsf{NoUse}$.

$$\underline{\textsc{Dest}(\xi, l \to r, i)}: \quad \Phi; Eq; ND \quad \leadsto \quad \begin{array}{l} \Phi, \mathsf{f}(\xi, X_2, \ldots, X_n), i \rhd w; \ Eq \wedge v \overset{?}{=} u_1 \\ X_2, i \overset{?}{\vdash} u_2; \ \ldots X_n, i \overset{?}{\vdash} u_n; ND \\ \Phi; Eq; ND \wedge \forall \tilde{x} \cdot [v \neq u_1 \vee i \overset{?}{\not\vdash} u_2 \vee \ldots \vee i \overset{?}{\not\vdash} u_n] \end{array}$$

If Φ contains $\xi, j \rhd v$ with $j \leq i$ and $(\xi, j \rhd v) \notin \mathsf{NoUse}$. We denote by $\tilde{x}$ the set of variables that occur in $\mathsf{f}(u_1, \ldots, u_n) \to w$, a fresh renaming of $l \to r$.

Figure 7.1: Transformation rules for satisfiability

Example 7.9. *Consider the constraint system* $\mathcal{C}'' = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ *of Example 7.2. Since* $\left(Y, 4 \overset{?}{\vdash} \mathsf{aenc}(\langle y', \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b))\right) \in D$*, the rule* $\textsc{Cons}(Y, \mathsf{aenc})$ *can be applied to* $\mathcal{C}''$*, guessing whether or not the attacker obtained the message* $\mathsf{aenc}(\langle x, y\rangle, \mathsf{pk}(sk_b))$ *applying a public-key encryption on two previously computed messages. This yields the two constraints* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *where*

$$\mathcal{C}_1 = \begin{cases} S_2(\mathcal{C}_1) : \{Z_1, X_1, X_2, Y, Z_2\} \\ D(\mathcal{C}_1) : \{Z_1, 4 \overset{?}{\vdash} c; \ X_1, 4 \overset{?}{\vdash} x_1; \ X_2, 4 \overset{?}{\vdash} x_2; Z_2, 4 \overset{?}{\vdash} c\} \\ Eq(\mathcal{C}_1) : \mathsf{aenc}(x_1, x_2) \overset{?}{=} \mathsf{aenc}(\langle y', \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b)) \\ Er(\mathcal{C}_1) : Y \overset{?}{=} \mathsf{aenc}(X_1, X_2) \end{cases}$$

and

$$\mathcal{C}_2 = \begin{cases} D(\mathcal{C}_2) : \{Z_1, 4 \overset{?}{\vdash} c; \ Y, 4 \overset{?}{\vdash} \mathsf{aenc}(\langle y', \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b)); \ Z_2, 4 \overset{?}{\vdash} c\} \\ Er(\mathcal{C}_2) : \mathsf{root}(Y) \overset{?}{\neq} \mathsf{aenc} \end{cases}$$

The first constraint can be simplified, solving equations and performing replacements, which yields:

$$\mathcal{C}_1' = \begin{cases} S_2(\mathcal{C}_1) : \{Z_1, X_1, X_2, Y, Z_2\} \\ D(\mathcal{C}_1) : \{Z_1, 4 \overset{?}{\vdash} c; \ X_1, 4 \overset{?}{\vdash} \langle y', \mathsf{pk}(sk_1)\rangle; \ X_2, 4 \overset{?}{\vdash} \mathsf{pk}(sk_b); Z_2, 4 \overset{?}{\vdash} c\} \\ Er(\mathcal{C}_1) : Y \overset{?}{=} \mathsf{aenc}(X_1, X_2) \end{cases}$$

The rule Axiom also guesses whether a trivial recipe (a left member of the frame, typically an axiom ax_i) can be applied. If so, the constraint can simply be removed. Otherwise, we also add an inequation on recipe forbidding it.

Example 7.10. *Continuing with the two constraints (respectively named* C_1' *and* C_2*), obtained in the previous example,* C_1' *yields, by application of* $\textsc{Axiom}(X_2, ax_4)$ *two constraints systems* C_{11}

and C_{22}.

$$\mathcal{C}_{11} = \begin{cases} D(\mathcal{C}_{11}) : \{Z_1, 4 \overset{?}{\vdash} c;\ X_1, 4 \overset{?}{\vdash} \langle y', \mathsf{pk}(sk_1)\rangle;\ Z_2, 4 \overset{?}{\vdash} c\} \\ Eq(\mathcal{C}_{11}) : \mathsf{pk}(sk_b) \overset{?}{=} \mathsf{pk}(sk_b) \\ Er(\mathcal{C}_{11}) : Y \overset{?}{=} \mathsf{aenc}(X_1, X_2) \wedge X_2 \overset{?}{=} ax_4 \end{cases}$$

and

$$\mathcal{C}_{12} = \begin{cases} D(\mathcal{C}_{12}) : \{Z_1, 4 \overset{?}{\vdash} c;\ X_1, 4 \overset{?}{\vdash} \langle y', \mathsf{pk}(sk_1)\rangle;\ X_2, 4 \overset{?}{\vdash} \mathsf{pk}(sk_b); Z_2, 4 \overset{?}{\vdash} c\} \\ Er(\mathcal{C}_{12}) : Y \overset{?}{=} \mathsf{aenc}(X_1, X_2) \wedge X_2 \overset{?}{\neq} ax_4 \end{cases}$$

Again, the first constraint can be simplified into the constraint system C'_{11}*:*

$$\mathcal{C}_{11} = \begin{cases} D(\mathcal{C}_{11}) : \{Z_1, 4 \overset{?}{\vdash} c;\ X_1, 4 \overset{?}{\vdash} \langle y', \mathsf{pk}(sk_1)\rangle;\ Z_2, 4 \overset{?}{\vdash} c\} \\ Er(\mathcal{C}_{11}) : Y \overset{?}{=} \mathsf{aenc}(X_1, X_2) \wedge X_2 \overset{?}{=} ax_4 \end{cases}$$

As already explained, the DEST rule is more tricky. If v is term of the frame, that can be unified with a non variable subterm of a left-hand side of a rewrite rule (for instance v is a ciphertext), we guess whether the rule can be applied to v. This corresponds to the equation $u_1 \overset{?}{=} v$, that yields an instance of w, the right member of the rewrite rule, provided that the rest of the left member is also deducible: in case of symmetric encryption, we get a constraint $X_2, i \overset{?}{\vdash} u_2$.

Example 7.11. *Consider the constraint system* $\mathcal{C}$*, that includes the frame* $\Phi = \{ax_1, 1 \rhd \mathsf{senc}(\langle a, b\rangle, c), ax_2, 2 \rhd cax_3, 3 \rhd \mathsf{senc}(c, a)\}$ *and the deducible constraints* $D = \{X, 3 \overset{?}{\vdash} b\}$ *(the other component being empty) Applying* DEST$(ax_1, \mathsf{sdec}(\mathsf{senc}(x, y), y) \to x, 2)$*, we get two constraint systems* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*:*

$$\mathcal{C}_1 = \begin{cases} \Phi(\mathcal{C}_1) : \Phi \cup \{\mathsf{sdec}(ax_1, X_1), 2 \rhd x\} \\ D(\mathcal{C}_1) : \{X_2, 2 \overset{?}{\vdash} y;\ X, 3 \overset{?}{\vdash} b\} \\ Eq(\mathcal{C}_1) : \mathsf{senc}(x, y) \overset{?}{=} \mathsf{senc}(\langle a, b\rangle, c) \end{cases}$$

and

$$\mathcal{C}_2 = \begin{cases} D(\mathcal{C}_2) : \{X, 3 \overset{?}{\vdash} b\} \\ ND(\mathcal{C}_2) : \forall x, y.(\mathsf{senc}(x, y) \overset{?}{\neq} \mathsf{senc}(\langle a, b\rangle, c) \vee 2 \overset{?}{\not\vdash} y) \end{cases}$$

Basically, we guess here whether the key c *can be deduced at step 2. The second constraint is unsatisfiable, while the first one can be simplified to:*

$$\begin{cases} \Phi(\mathcal{C}_1) = \Phi \cup \{\mathsf{sdec}(ax_1, X_1), 2 \rhd \langle a, b\rangle\} \\ D(\mathcal{C}_1) = \{X_2, 2 \overset{?}{\vdash} c;\ X, 3 \overset{?}{\vdash} b\} \end{cases}$$

Of course, these rules will not be applied without restriction, otherwise we would roughly enumerate all possible attackers recipes and, though this would be complete, this would certainly not terminate. For instance, we are not going to apply CONS(X, f) to $X, i \overset{?}{\vdash} x$ when x is a variable (assuming x does not appear in another constraint), since the very same rule would apply to one of the resulting constraints.

Another set of rules, the *equality rules* of the figure 7.2, guess equalities between right-hand sides of deducibility constraints and/or members of the frame. These rules do not correspond to attacker's actions and they are not necessary if we are only interested in reachability properties. For equivalence properties, it is however necessary to ensure that the observable identities are the same on both systems.

Example 7.12. $\Phi_1 = \{ax_1, 1 \rhd a;\ ax_2, 1 \rhd k_1;\ ax_3, 2 \rhd \mathsf{senc}(x, k);\ ax_4, 3 \rhd \mathsf{senc}(\mathsf{senc}(a, k_1), k)\}$ *and* $\Phi_2 = \{ax_1, 1 \rhd a;\ ax_2, 1 \rhd k_1;\ ax_3, 2 \rhd \mathsf{senc}(x, k);\ ax_4, 3 \rhd \mathsf{senc}(b, k)\}$*. If* $x = \mathsf{senc}(a, k_1)$*,*

then the two frames are not statically equivalent since $ax_2 = ax_3$ is an equality satisfied on the first frame and not on the second. If $x \neq \mathsf{senc}(a, k_1)$, then the two frames are statically equivalent. If, for instance, the deducible constraint associated with both frames is $X, 1 \overset{?}{\vdash} x$, then the rules of the figure 7.1 will not help in finding the witness of non-equivalence.

Finally, the last transformation rule of figure 7.2 guesses the deducible subterms of the frame, which allows to capture a static equivalence algorithm (such as in [AC06] as a particular case).

$$\underline{\textsc{Eq-left-left}(\xi_1, \xi_2)} : \ Eq \ \begin{cases} \nearrow \ Eq \wedge u_1 \overset{?}{=} u_2 \\ \searrow \ Eq \wedge u_1 \overset{?}{\neq} u_2 \end{cases}$$

where $\xi_1, i_1 \rhd u_1,\ \xi_2, i_2 \rhd u_2 \in \Phi$ for some ξ_1, ξ_2, i_1, i_2

$$\underline{\textsc{Eq-left-right}(\xi_1, X_2)} : \ Eq, \mathsf{NoUse} \ \begin{cases} \nearrow \ Eq \wedge u_1 \overset{?}{=} u_2, \mathsf{NoUse} \cup (\xi_1, i_1 \rhd u_1) \\ \searrow \ Eq \wedge u_1 \overset{?}{\neq} u_2, \mathsf{NoUse} \end{cases}$$

where $\xi_1, i_1 \rhd u_1 \in \Phi$ and $X_2, i_2 \overset{?}{\vdash} u_2 \in D$, with $i_2 < i_1$ and $X_2 \in S_2$ for some ξ_1, ξ_2, u_1, u_2

$$\underline{\textsc{Eq-right-right}(X, \xi)} : \ X, i \overset{?}{\vdash} u;\ Eq;\ Er \ \begin{cases} \nearrow \ Eq \wedge u \overset{?}{=} v;\ Er \wedge X \overset{?}{=} \xi \\ \searrow \ X, i \overset{?}{\vdash} u;\ Eq \wedge u \overset{?}{\neq} v;\ Er \end{cases}$$

where $\xi \in \mathcal{T}(\mathcal{F}_c, \mathrm{dom}(\alpha))$ and $v = \xi\alpha$ with $\alpha = \{Y \to w \mid (Y, j \overset{?}{\vdash} w) \in D \wedge j \leq i \wedge Y \in S_2\}$
Moreover, we assume that:

— if $\mathsf{root}(\xi) = \mathsf{f}$ then $Er \not\vdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$

— if $\xi = Y$ then for all $\mathsf{f} \in \mathcal{F}_c$, $Er \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ is equivalent to $Er \vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$.

$$\underline{\textsc{Ded-st}(\xi, \mathsf{f})} : \Phi; Eq; ND \ \begin{cases} \nearrow \ \Phi;\ X_1, m \overset{?}{\vdash} x_1;\ \cdots;\ X_n, m \overset{?}{\vdash} x_n;\ Eq \wedge u \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n);\ ND \\ \searrow \ \Phi;\ Eq; \\ \quad ND \wedge \forall \tilde{x} \cdot [u \neq \mathsf{f}(x_1, \ldots, x_n) \vee m \overset{?}{\not\vdash} x_1 \vee \ldots \vee m \overset{?}{\not\vdash} x_n] \end{cases}$$

If Φ contains $\xi, i \rhd u$ and $(\xi, i \rhd u) \notin \mathsf{NoUse}$. The sequences $\tilde{x} = x_1, \ldots, x_n$, and $X_1, \ldots, X_n$ are sequences of fresh variables and m represents the maximal index that occurs in $\mathcal{C}$.

Figure 7.2: Additional transformation rules for static equivalence

Example 7.13. *Let us come back to the example 7.12. Applying* $\textsc{Eq-left-left}(ax_2, ax_3)$ *to the constraint system whose frame is Φ_1, we get two constraint systems:*

$$\begin{cases} \Phi(\mathcal{C}_1) = \Phi_1 \\ D(\mathcal{C}_1) = \{X, 1 \overset{?}{\vdash} x\} \\ Eq(\mathcal{C}_1) = \mathsf{senc}(x, k) \overset{?}{=} \mathsf{senc}(\mathsf{senc}(a, k_1), k) \end{cases} \qquad \begin{cases} \Phi(\mathcal{C}_2) = \Phi_1 \\ D(\mathcal{C}_2) = \{X, 1 \overset{?}{\vdash} x\} \\ Eq(\mathcal{C}_2) = \mathsf{senc}(x, k) \overset{?}{\neq} \mathsf{senc}(\mathsf{senc}(a, k_1), k) \end{cases}$$

And the case $x = \mathsf{senc}(a, k_1)$ is now distinguished from the case $x \neq \mathsf{senc}(a, k_1)$. We will see later how and why this is sufficient.

Example 7.14. *Consider the two constraint systems*

$$\Phi_1 = \{X, 1ax_1, 1 \rhd a, ax_2, 2 \rhd bax_3, 3 \rhd x_1\}, \quad D_1 = \{X, 1 \overset{?}{\vdash} x_1, \quad Y, 2 \overset{?}{\vdash} x_1, \quad Z, 3 \overset{?}{\vdash} y_1\}$$
$$\Phi_2 = \{X, 1ax_1, 1 \rhd a, ax_2, 2 \rhd bax_3, 3 \rhd x_2\}, \quad D_2 = \{X, 1 \overset{?}{\vdash} x_2, \quad Y, 2 \overset{?}{\vdash} y_2, \quad Z, 3 \overset{?}{\vdash} y_2\}$$

There are redundant constraints in each individual system. However, we need $x_1 = y_1$ and $x_2 = y_2$ in order to get equivalent systems, since the recipes X, Y must yield the same value, according to the first system (hence $x_2 = y_2$) and the recipes Y, Z yield the same value, according to the second system (hence $x_1 = y_1$). The rule Eq-right-right *takes care of such situations: we guess whether different recipes yield the same value and record the result of the guess in the constraint.*

Example 7.15. *Consider the two constraint systems $\Phi_1 = \{ax_1, 1 \rhd \mathsf{pk}(sk_a), ax_2, 2 \rhd \mathsf{aenc}(x, \mathsf{pk}(sk_a))\}$ and $D_1 = \{X, 1 \overset{?}{\vdash} x\}$ on the one hand and $\Phi_2 = \{ax_1, 1 \rhd \mathsf{pk}(sk_a), ax_2, 2 \rhd \mathsf{aenc}(b, \mathsf{pk}(sk_a)\}$ and $D_2 = \{X, 1 \overset{?}{\vdash} x\}$ on the other hand.*

Intuitively, the rules do not help in simplifying any further one of the two constraints. The only relevant possibility would be to try decrypting $\mathsf{aenc}(x, \mathsf{pk}(sk_a))$, but the private key sk_a is not deducible. Though, the two constraint systems are not equivalent since the attacker can construct $\mathsf{aenc}(x, \mathsf{pk}(sk_a))$ (using the recipe $\mathsf{aenc}(X, ax_1)$) and therefore observe the identity $\mathsf{aenc}(X, ax_1) = ax_2$ on Φ_1, which is not possible on Φ_2.

This is the reason of the rule Ded-st*, that guesses the subterms of the frame that can be constructed by the attacker. In the above example, the first constraint system would become*

$$\Phi_1, \{X, 1 \overset{?}{\vdash} x; X_1, 2 \overset{?}{\vdash} x; X_2, 2 \overset{?}{\vdash} \mathsf{pk}(sk_a)\}$$

(the other branch is unsatisfiable), while on the second constraint we get

$$\Phi_2, \{X, 1 \overset{?}{\vdash} x;\ X_1, 2 \overset{?}{\vdash} b;\ X_2, 2 \overset{?}{\vdash} \mathsf{pk}(sk_a)\}$$

Eventually, this last constraint will be proven unsatisfiable, witnessing the non-equivalence of the constraint systems.

Now, before explaining how to apply the rules on pairs of sets of constraint systems, we formalise what we used implicitly in our examples: the constraints are normalised after each transformation step. Next, we define a simple strategy called *strong strategy*, which is sufficient for trace properties, but has to be slightly relaxed for equivalence properties, as we will see in the section 7.4.

7.2.2 Normalisation

The normalisation consists mainly in simplifying the equations and inequations and performing the replacements when relevant.

The normalisation rules are displayed in the figure 7.3. As usual, substitutions are confused with solved conjunctions of equations. We also switch sometimes the order of the components of a constraint, in order to ease the display and omit irrelevant parts of the constraint.

If we do not apply repeatedly the same rule in the same way, the rules of the figure 7.3 terminate on any constraint system.

Now, there are situations in which is clearly not necessary to apply some rules. We say that $\textsc{Cons}(X, f)$ (resp. $\textsc{Axiom}(X, \mathsf{path})$) is *useless on a constraint* $\mathcal{C}$ if it is not applicable or its application results in two constraints $\mathcal{C}_1$ and $\mathcal{C}_2$ such that $\mathcal{C}_1$ simplifies to $\bot$ using the rules of the figure 7.3 and $\mathcal{C}_2$ simplifies to $\mathcal{C}$ using these rules. Similarly, $\textsc{Dest}(\xi, l \to r, i)$ is *useless on a constraint* $\mathcal{C}$ if it has been already applied with the same instance:

— either the frame $\Phi(\mathcal{C})$ contains $f(\xi, \zeta_2, \ldots, \zeta_n), j \rhd w$ and $j \leq i$ and $\mathsf{root}(l) = f$

$$\Phi; D; Er; ND; \mathsf{NoUse}; Eq \wedge \bigwedge_{i=1}^{n} u_i \stackrel{?}{=} v_i \quad \rightsquigarrow \quad \Phi\sigma; D\sigma; Er; ND\sigma; \mathsf{NoUse}\sigma; Eq\sigma \wedge \sigma \quad \text{if } \sigma = \mathsf{mgu}(\bigwedge_{i=1}^{n} u_i \stackrel{?}{=} v_i) \tag{Nuni1}$$

$$\Phi; D; Er; ND; \mathsf{NoUse}; Eq \wedge \bigwedge_{i=1}^{n} u_i \stackrel{?}{=} v_i \quad \rightsquigarrow \quad \bot \quad \text{if } \mathsf{mgu}(\bigwedge_{i=1}^{n} u_i \stackrel{?}{=} v_i) = \bot \tag{Nins1}$$

$$\Phi; D; Eq; ND; \mathsf{NoUse}; Er \wedge \bigwedge_{i=1}^{n} \zeta_i \stackrel{?}{=} \xi_i \quad \rightsquigarrow \quad \Phi\theta; D; Eq; ND; \mathsf{NoUse}\theta; Er\theta \wedge \theta \quad \text{if } \theta = \mathsf{mgu}(\bigwedge_{i=1}^{n} \zeta_i \stackrel{?}{=} \xi_i) \tag{Nuni2}$$

$$\Phi; D; Eq; ND; \mathsf{NoUse}; Er \wedge \bigwedge_{i=1}^{n} \zeta_i \stackrel{?}{=} \xi_i \quad \rightsquigarrow \quad \bot \quad \text{if } \mathsf{mgu}(\bigwedge_{i=1}^{n} \zeta_i \stackrel{?}{=} \xi_i) = \bot \tag{Nins2}$$

$$Eq \wedge \forall \tilde{x}.[\bigvee_{i=1}^{n} u_i \stackrel{?}{\neq} v_i] \quad \rightsquigarrow \quad Eq \qquad \text{if } \mathsf{mgu}(\bigvee_{i=1}^{n} u_i \stackrel{?}{\neq} v_i) = \bot \tag{Nneq1}$$

$$Eq \wedge \forall \tilde{x}.[Eq' \vee u \stackrel{?}{\neq} u] \quad \rightsquigarrow \quad Eq \wedge \forall \tilde{x}.Eq' \tag{Nt1}$$

$$Eq \wedge \forall \tilde{x}.[Eq' \vee x \stackrel{?}{\neq} u] \quad \rightsquigarrow \quad Eq \wedge \forall \tilde{x} \setminus \{x\}.Eq'\sigma \quad \text{if } x \in \tilde{x} \setminus vars(u) \text{ and } \sigma = \{x \to u\} \tag{Nelim1}$$

$$Eq \wedge \forall \tilde{x}.\forall x.Eq' \quad \rightsquigarrow \quad Eq \wedge \forall \tilde{x}.Eq \qquad \text{where } x \notin vars^1(Eq) \tag{Nelim2}$$

$$Eq \wedge \forall \tilde{x}.[Eq' \vee \mathsf{f}(u_1, \ldots, u_n) \stackrel{?}{\neq} \mathsf{f}(v_1, \ldots v_n)] \quad \rightsquigarrow \quad Eq \wedge \forall \tilde{x}.[Eq' \vee \bigvee_{i=1}^{n} u_i \stackrel{?}{\neq} v_i] \tag{Nsplit}$$

$$Eq \wedge u \stackrel{?}{\neq} v \wedge \forall \tilde{x}.[Eq' \vee u \stackrel{?}{\neq} v] \quad \rightsquigarrow \quad Eq \wedge u \stackrel{?}{\neq} v \tag{Nd}$$

$$Er \wedge \zeta \stackrel{?}{\neq} \xi \quad \rightsquigarrow \quad Er \qquad \text{if } \mathsf{mgu}(\zeta, \xi) = \bot \tag{Nneq2}$$

$$Er \wedge \zeta \stackrel{?}{\neq} \zeta \quad \rightsquigarrow \quad \bot \tag{Nt2}$$

$$Er \wedge \mathsf{root}(f(\xi_1, \ldots, \xi_n)) \neq f \quad \rightsquigarrow \quad \bot \tag{Ntop1}$$

$$Er \wedge \mathsf{root}(f(\xi_1, \ldots, \xi_n)) \neq g \quad \rightsquigarrow \quad Er \qquad \text{if } f \neq g \tag{Ntop2}$$

Figure 7.3: Simplification rules for formula on terms

$$Eq \wedge \forall \tilde{x}.[Eq' \vee x \overset{?}{\neq} a] \quad \rightsquigarrow Eq \qquad \text{(Nname)}$$

if $a \in \mathcal{N}$, $(X, i \overset{?}{\vdash} x) \in D$, $\textsc{Axiom}(X, \mathsf{path})$ is useless for any path and
$\textsc{Dest}(\xi, l \to r, i)$ is useless for any $\xi, l \to r$, and
for all $(\zeta, j \rhd v) \in \Phi$, $j \leq i$ and $v \in \mathcal{X}^1$ implies $(\zeta, j \rhd v) \in \mathsf{NoUse}$

$$D \wedge X, i \overset{?}{\vdash} u \quad \rightsquigarrow \quad \bot \qquad \text{(Nnosol)}$$

if $\textsc{Cons}(X, \mathsf{f})$ is useless for all $\mathsf{f} \in \mathcal{F}_c$; and $\textsc{Axiom}(X, \mathsf{path})$ is useless for any path; and
$\textsc{Dest}(\xi, l \to r, i)$ is useless for all $\xi, l \to r$; and
for all $(\zeta, j \rhd v) \in \Phi$, $j \leq i$ and $v \in \mathcal{X}^1$ implies $(\zeta, j \rhd v) \in \mathsf{NoUse}$

Figure 7.4: Two additional simplification rules

— or ND contains $\forall \vec{x}.(v \neq u_1 \vee i \overset{?}{\not\vdash} u_2 \vee \cdots \vee i \overset{?}{\not\vdash} u_n)$, $(\xi, j \rhd v)$ is in Φ and $f(u_1, \ldots, u_n) \to w$ is a renaming of $l \to r$.

— or Φ does not contain any $\xi, j \rhd v$ with $j \leq i$ and $(\xi, j \rhd v) \notin \mathsf{NoUse}$

We further apply two simplification rules, that are displayed in the figure 7.4

Intuitively, the first rule states that x cannot be a name, if it has to be deducible and cannot be obtained from the frame. Indeed, we have assumed that all public names are explicitly disclosed in the frame (see Chapter 6)

Intuitively, the second rule states that, in order to deduce a message, the attacker has either to construct it from deducible messages, or retrieve it from the frame and deducible messages. In other words, any attacker's ground recipe is built on $\mathcal{F}_c, \mathcal{F}_d$ and the ax_i.

Definition 7.9 (normalization)**.** *If $\mathcal{C}$ is a constraint system, we let $\mathcal{C}\downarrow$ be an irreducible form of $\mathcal{C}$, w.r.t. the rules of the figures 7.3 and 7.4.*

In what follows we will therefore assume that every constraint system is eagerly normalised.

7.2.3 A strong strategy

In the previous subsection, we described that sometimes, it is not necessary to applied the rules Dest, Axiom and Cons, thus we defined the notion of a rule being *useless* for a constraint system. In fact, in order to ensure termination, we have to define the condition of uselessness for all the rules in Figure 7.2.

Example 7.16. *Consider the constraint system $\mathcal{C}$ composed of the deducible constraint $D = \{X, 2 \overset{?}{\vdash} x\}$ and the frame $\Phi = \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(a, b);\ ax_3, 3 \rhd \mathsf{senc}(x, b)\}$. The application of the rule* Eq-left-left(ax_2, ax_3) *on $\mathcal{C}$ yields two constraint system $\mathcal{C}_1$ and $\mathcal{C}_2$ such that:*

$$\mathcal{C}_1 = \begin{cases} D(\mathcal{C}_1) : \{X, 1 \overset{?}{\vdash} a\} \\ \Phi(\mathcal{C}_1) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(a, b);\ ax_3, 3 \rhd \mathsf{senc}(a, b)\} \\ Eq(\mathcal{C}_1) : x \overset{?}{=} a \end{cases}$$

and

$$\mathcal{C}_2 = \begin{cases} D(\mathcal{C}_1) : \{X, 1 \overset{?}{\vdash} x\} \\ \Phi(\mathcal{C}_1) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(a, b);\ ax_3, 3 \rhd \mathsf{senc}(x, b)\} \\ Eq(\mathcal{C}_1) : x \overset{?}{\neq} a \end{cases}$$

However, the rule Eq-left-left(ax_2, ax_3)*, as described in Figure 7.2, can still be applicable on both constraint systems* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*. However, the application of* Eq-left-left(ax_2, ax_3) *on* $\mathcal{C}_1$ *yields, after normalisation, the constraint systems* $\mathcal{C}_1$ *and* $\bot$*. Similarly, the application of* Eq-left-left(ax_2, ax_3) *on* $\mathcal{C}_2$ *yields, after normalisation the constraint systems* $\bot$ *and* $\mathcal{C}_2$*. Hence, to ensure the termination, we have to discard this kind of rule application.*

Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. We say that the rules Eq-left-left(ξ_1, ξ_2), Eq-left-right(ξ_1, X_2) and Eq-right-right(X, ξ) are useless on the constraint system $\mathcal{C}$ if their application on $\mathcal{C}$ yields, after normalisation, the constraint systems $\mathcal{C}$ and $\bot$. Moreover, we say that the rule Ded-st(ξ, f) is useless on the constraint system $\mathcal{C}$ if:

— either there exists $X_1, \ldots, X_n \in vars^2(\mathcal{C})$ such that $\mathsf{C}\lfloor \mathsf{f}(X_1, \ldots, X_n)\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C}) = u$ with $(\xi, j \rhd u)$ is in Φ, θ is the most general unifier of the equation in Er.

— or ND contains $\forall \tilde{x}.(u \stackrel{?}{\neq} \mathsf{f}(x_1, \ldots, x_n) \vee m \stackrel{?}{\not\vdash} x_1 \vee \cdots \vee i \stackrel{?}{\not\vdash} x_n)$, $(\xi, j \rhd u)$ is in Φ, $x_1, \ldots, x_n$ are fresh variables and $m = |\Phi|$.

— or Φ does not contain any $\xi, j \rhd u$ and $(\xi, j \rhd u) \notin \mathsf{NoUse}$

Intuitively, the condition of uselessness of the rule Ded-st(ξ, f) describe the fact that this rule was already applied or does not need to be applied. The complexity of the first condition comes from the fact that while the rule Ded-st creates new deducible constraints, these same deducible constraints might be instantiated later on by other rules such as Cons or Axiom.

Example 7.17. *Consider the constraint system* $\mathcal{C}$ *in Example 7.16. The application of the rule* Ded-st(ax_3, senc) *on* $\mathcal{C}$ *yields, after normalisation, two constraint systems* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *such that:*

$$\mathcal{C}_1 = \begin{cases} D(\mathcal{C}_1) : \{X, 1 \stackrel{?}{\vdash} a;\ X_1, 3 \stackrel{?}{\vdash} x;\ X_2, 3 \stackrel{?}{\vdash} b\} \\ \Phi(\mathcal{C}_1) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(a, b);\ ax_3, 3 \rhd \mathsf{senc}(x, b)\} \end{cases}$$

and

$$\mathcal{C}_2 = \begin{cases} D(\mathcal{C}_1) : \{X, 1 \stackrel{?}{\vdash} x\} \\ \Phi(\mathcal{C}_1) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(a, b);\ ax_3, 3 \rhd \mathsf{senc}(x, b)\} \\ ND(\mathcal{C}_1) : \forall x_1.\forall x_2.\mathsf{senc}(x_1, x_2) \stackrel{?}{\neq} \mathsf{senc}(x, b) \vee 3 \stackrel{?}{\not\vdash} x \vee 3 \stackrel{?}{\not\vdash} b \end{cases}$$

The rule Ded-st(ax_3, senc) *is useless on* $\mathcal{C}_2$ *due to the non deducible constraint, and is useless on* $\mathcal{C}_1$ *since* $\mathsf{C}\lfloor \mathsf{senc}(X_1, X_2) \rfloor_{\Phi(\mathcal{C}_1)} \mathsf{acc}^1(\mathcal{C}_1) = \mathsf{senc}(x, b)$*. Now, if we apply the rule* Axiom(X_1, ax_2) *on* $\mathcal{C}_2$*, we obtain two new constraint systems where one of these is the following:*

$$\mathcal{C}_3 = \begin{cases} D(\mathcal{C}_3) : \{X, 1 \stackrel{?}{\vdash} a;\ X_2, 3 \stackrel{?}{\vdash} b\} \\ \Phi(\mathcal{C}_3) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(a, b);\ ax_3, 3 \rhd \mathsf{senc}(\mathsf{senc}(a, b), b)\} \\ Er(\mathcal{C}_3) : X_1 \stackrel{?}{=} ax_2 \\ Eq(\mathcal{C}_3) : x \stackrel{?}{=} \mathsf{senc}(a, b) \end{cases}$$

Ded-st(ax_3, senc) *is useless on* $\mathcal{C}_3$ *since* $\mathsf{C}\lfloor \mathsf{senc}(X_1, X_2)\theta_3 \rfloor_{\Phi(\mathcal{C}_3)} \mathsf{acc}^1(\mathcal{C}_3) = \mathsf{senc}(\mathsf{senc}(a, b), b)$ *with* $\theta_3 = \{X_1 \mapsto ax_2\}$.

Discarding the useless application of a rule allows us to avoid some of the trivial termination problem (such as infinite application of Eq-left-left). However, some of the rules, even without being useless on a constraint system, may yield some simple termination problems.

Example 7.18. *Consider the constraint system* $\mathcal{C}$ *composed of the deducible constraint* $D = \{X, 1 \stackrel{?}{\vdash} x\}$ *and the frame* $\Phi = \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd x\}$*. The application of the rule* Cons(X, senc) *yields the following constraint system* $\mathcal{C}_1$*:*

$$\mathcal{C}_1 = \begin{cases} D(\mathcal{C}_1) : \{X_1, 1 \stackrel{?}{\vdash} x_1;\ X_2, 1 \stackrel{?}{\vdash} x_2\} \\ \Phi(\mathcal{C}_1) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(x_1, x_2)\} \\ Er(\mathcal{C}_1) : X \stackrel{?}{=} \mathsf{senc}(X_1, X_2) \\ Eq(\mathcal{C}_1) : x \stackrel{?}{=} \mathsf{senc}(x_1, x_2) \end{cases}$$

Since we obtained once again a constraint system which a deducible constraint with a variable as right hand term, we can apply the rule Cons *once again, thus leads to a termination problem. Similarly, the application of the rule* Dest$(ax_2, \mathsf{adec}(\mathsf{aenc}(x, \mathsf{pk}(y)), y) \to x, 2)$ *yields the following constraint system* $\mathcal{C}_2$*:*

$$\mathcal{C}_2 = \begin{cases} D(\mathcal{C}_2) : \{X, 1 \stackrel{?}{\vdash} x;\ X_2, 1 \stackrel{?}{\vdash} x_2\} \\ \Phi(\mathcal{C}_2) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{aenc}(x_1, \mathsf{pk}(x_2));\ \mathsf{adec}(ax_2, Y_2), 2 \rhd x_1\} \\ Eq(\mathcal{C}_2) : x \stackrel{?}{=} \mathsf{aenc}(x_1, \mathsf{pk}(x_2)) \end{cases}$$

Hence the rule Dest *can be applied once again on* $\mathsf{adec}(ax_2, Y_2)$ *which leads to a termination problem.*

Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. We say that a rule is applicable on $\mathcal{C}$ when all the conditions needed to apply the rule are fulfilled (see Figure 7.1 and 7.2). To avoid the termination issues illustrated in Example 7.18, we need to consider some additional requirements. We say that a rule can be *strongly applied*, or is *strongly applicable*, when it is not useless on $\mathcal{C}$ and when:

- the rule is Cons(X, f) and either the term t is not a variable, or there exists an atomic statement $(\mathsf{root}(X) \stackrel{?}{\neq} \mathsf{g})$ in Er such that $g \in \mathcal{F}_c$ and $g \neq f$;
- the rule is Axiom(X, path) and the term v is not a variable or there exists $\mathsf{f} \in \mathcal{F}_c$ such that $(\mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f})$ in Er;
- the rule is Dest and the term v is not a variable;
- the rule is Ded-st and the term u is not a variable;
- the rule is Eq-left-left (with no additional condition);
- the rule is Eq-left-right and the terms u_1, u_2 are the same variable.
- the rule is Eq-right-right and $\xi \in vars^2(D)$ and $u = v$, *i.e.* u and v are the same variable;

We will see in the next subsection that our strategy will require to weakened at some point the conditions of strong application of some rules, *i.e.* Cons, Axiom and Eq-right-right.

7.3 Simplifying sets of constraint systems

In the previous subsection, we describe how our transformation rules can be used to transform a single constraint system into two constraint systems. However, since we want to decide the symbolic equivalence of sets of constraint systems, we have to describe how to apply our rules on such sets.

7.3.1 From constraint system to vectors

In fact, for our algorithm, it is crucial to order the constraint systems in a set and to preserve this order. Hence instead of working with sets of constraint system, we will work with vectors of constraint systems. We explain here how our rules can be used on a (pair of) vector of constraint systems (more precisely on a row matrix of constraint systems), assuming that the constraint systems of the vector have the same structure.

Actually, the basic idea is to apply the same transformation rule (with the same parameters) on each constraint system of the vector. Note that, the parameters of a transformation rule only depend on the structure of the underlying constraint system. Thanks to this, the simultaneous application of a transformation rule can be defined in a natural way.

Definition 7.10 (application of a rule on a pair of row matrices)**.** *Let* $M = [\mathcal{C}_1, \ldots, \mathcal{C}_n]$ *and* $M' = [\mathcal{C}'_1, \ldots, \mathcal{C}'_{n'}]$ *be two row matrices having* n *(resp.* n'*) columns and having the same structure. Let* Rule$(\tilde{p})$ *be an instance of a transformation rule. The application of* Rule$(\tilde{p})$ *on the pair* (M, M') *yields two pairs of row matrices* (M_1, M'_1) *and* (M_2, M'_2) *such that :*

— $M_1 = [\mathcal{C}_{1,1}, \ldots, \mathcal{C}_{1,n}]$ *and* $M_2 = [\mathcal{C}_{2,1}, \ldots, \mathcal{C}_{2,n}]$*; whereas* $M'_1 = [\mathcal{C}'_{1,1}, \ldots, \mathcal{C}'_{1,n'}]$ *and* $M'_2 = [\mathcal{C}'_{2,1}, \ldots, \mathcal{C}'_{2,n'}]$*;*

— *for all* $i \in \{1 \ldots n\}$*,* $\mathcal{C}_{1,i}$ *and* $\mathcal{C}_{2,i}$ *are the constraint systems obtained by application of* $\textsc{Rule}(\tilde{p})$ *on* $\mathcal{C}_i$*;*

— *for all* $i \in \{1 \ldots n'\}$*,* $\mathcal{C}'_{1,i}$ *and* $\mathcal{C}'_{2,i}$ *are the constraint systems obtained by application of* $\textsc{Rule}(\tilde{p})$ *on* $\mathcal{C}'_i$*.*

7.3.2 Matrices of constraint systems

The non-deducible constraints introduced by the rules Dest and Ded-st are formally interesting since they allow us to properly divide the solutions of a constraint system. However, note that in Figures 7.1, 7.2 and 7.3, no rule focuses on solving these non-deducible constraints. We illustrate in the next example how we solve them.

Example 7.19. *Consider the constraint system* $\mathcal{C}$ *composed of the deducible constraints* $D = \{X, 2 \overset{?}{\vdash} \mathsf{senc}(a, a)\}$*, the frame* $\Phi = \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(b, a)\}$*,* $S_2 = \{X\}$ *and* $S_1 = \{x\}$*. The application of the rule* $\textsc{Dest}(ax_2, \mathsf{sdec}(\mathsf{senc}(x, y), y) \to x, 2)$ *on* $\mathcal{C}$ *yields two constraint systems* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *such that:*

$$\mathcal{C}_1 = \begin{cases} D(\mathcal{C}_1) : \{X, 2 \overset{?}{\vdash} \mathsf{senc}(a, a);\ Y, 2 \overset{?}{\vdash} a\} \\ \Phi(\mathcal{C}_1) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(b, a);\ \mathsf{sdec}(ax_2, Y), 2 \rhd b\} \end{cases}$$

and

$$\mathcal{C}_2 = \begin{cases} D(\mathcal{C}_2) : \{X, 2 \overset{?}{\vdash} \mathsf{senc}(a, a)\} \\ \Phi(\mathcal{C}_2) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(b, a)\} \\ ND(\mathcal{C}_2) : \forall x_1. \forall x_2. \mathsf{senc}(x_1, x_2) \overset{?}{\neq} \mathsf{senc}(b, a) \vee 2 \overset{?}{\nvdash} a \end{cases}$$

To solve the non-deducible constraint in $\mathcal{C}_2$*, we will use the informations we get from the resolution of* $\mathcal{C}_1$*. Note that the rule* $\textsc{Axiom}(Y, ax_1)$ *is applicable on* $\mathcal{C}_1$ *and yields two constraint systems* $\mathcal{C}_3$ *and* $\mathcal{C}_4$ *such that*

$$\mathcal{C}_3 = \begin{cases} D(\mathcal{C}_3) : \{X, 2 \overset{?}{\vdash} \mathsf{senc}(a, a)\} \\ \Phi(\mathcal{C}_3) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(b, a);\ \mathsf{sdec}(ax_2, Y), 2 \rhd b\} \\ Er(\mathcal{C}_3) : Y \overset{?}{=} ax_1 \end{cases}$$

and

$$\mathcal{C}_4 = \begin{cases} D(\mathcal{C}_3) : \{X, 2 \overset{?}{\vdash} \mathsf{senc}(a, a);\ Y, 2 \overset{?}{\vdash} a\} \\ \Phi(\mathcal{C}_3) : \{ax_1, 1 \rhd a;\ ax_2, 2 \rhd \mathsf{senc}(b, a);\ \mathsf{sdec}(ax_2, Y), 2 \rhd b\} \\ Er(\mathcal{C}_3) : Y \overset{?}{\neq} ax_1 \end{cases}$$

But on $\mathcal{C}_4$*, the successive applications of* $\textsc{Cons}(Y, \mathsf{f})$ *and* $\textsc{Axiom}(Y, \mathsf{path})$ *for any* f *and* path *will yields after normalisation the constraint system* $\bot$*. Hence, when we compare* $\mathcal{C}_3$ *and* $\mathcal{C}_2$*, we can see that the non-deducible constraint in* $\mathcal{C}_2$ *will never be satisfiable and so* $\mathcal{C}_2$ *does not have a solution.*

As illustrated in Example 7.19, to solve the non-deducible constraints of a constraint system, we will use the informations we obtain from the application of the rules on other constraint systems. To keep track of these informations, we regroup the vectors of constraint systems into matrices of constraint systems.

Example 7.20. *Consider the constraint system* $\mathcal{C}$ *of Example 7.19 and consider the vector of constraint systems* $[\mathcal{C}, \mathcal{C}']$ *where* $\mathcal{C}'$ *is some constraint system with the same structure as* $\mathcal{C}$*. By applying* $\textsc{Dest}(ax_2, \mathsf{sdec}(\mathsf{senc}(x, y), y) \to x, 2)$ *on* $[\mathcal{C}, \mathcal{C}']$*, instead of obtaining two vectors* $[\mathcal{C}_1, \mathcal{C}'_1]$

and $[\mathcal{C}_2, \mathcal{C}'_2]$ *where* $\mathcal{C}_1, \mathcal{C}_2$ *are the constraint system of Example 7.19, we obtain the following matrix of constraint system:*

$$\begin{bmatrix} \mathcal{C}_1, \mathcal{C}'_1 \\ \mathcal{C}_2, \mathcal{C}'_2 \end{bmatrix}$$

Then when we applied the rule Axiom(Y, ax_1) *on* $[\mathcal{C}_2, \mathcal{C}'_2]$*, we obtained the following matrix of constraint systems:*

$$\begin{bmatrix} \mathcal{C}_3, \mathcal{C}'_3 \\ \mathcal{C}_4, \mathcal{C}'_4 \\ \mathcal{C}_2, \mathcal{C}'_2 \end{bmatrix}$$

At this point, to check the non-deducible constraint in $\mathcal{C}_2$*, we only have to check the informations we have on the constraint systems in the same column as* $\mathcal{C}_2$*.*

7.3.2.1 Symbolic equivalence of matrices of constraint systems

Note that in Example 7.20, the constraint system $\mathcal{C}_1, \ldots, \mathcal{C}_4$ has the same sets S_2, S_1 and also contains the same subset of deducible constraints whose second order variable is in S_2. This is formalised in the following definition.

Definition 7.11 (shape)**.** *Let* $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ *be a constraint system. The* shape *of* $\mathcal{C}$ *is given by* S_2*, and* $\{(X, i) \mid X, i \overset{?}{\vdash} u \in D \text{ and } X \in S_2\}$*.*

Intuitively, the shape of a constraint system only represents the "actions" of the intruder.

We extend the notion of *same structure* between matrices of constraint systems in the following way. Two matrices M and M' of constraint systems having n (resp. n') lines and m (resp. m') columns have the *same structure* if:

- all the constraint systems stored in M and M' have the same shape;
- $n = n'$, *i.e.* M and M' have the same number of lines; and
- for all $i \in \{1 \ldots n\}$, the constraint systems stored in the i^{th} line of the matrices M and M' have the same structure.

Since our algorithm considers matrices of constraint system, we extend the notion of symbolic equivalence to pairs of matrices of constraint systems.

Definition 7.12 (symbolic equivalence $\approx_s$)**.** *Let* M, M' *be two matrices of symbolic constraint systems having the same structure. We have that* $M \subseteq_s M'$ *if for all* $1 \leq i \leq n$*, for all* $1 \leq j \leq m$*, for all* $(\sigma, \theta) \in \mathsf{Sol}(M_{i,j})$*, there exists* $1 \leq k \leq m'$ *and a substitution* σ' *such that* $(\sigma', \theta) \in \mathsf{Sol}(M'_{i,k})$ *and* $\Phi_{i,j}\sigma \sim \Phi'_{i,k}\sigma'$*.*

If $M \subseteq_s M'$ *and* $M' \subseteq_s M$*, then we say that* M *and* M' *are in symbolic equivalence, denoted by* $M \approx_s M'$*.*

7.3.2.2 Application of the rules on matrices of constraint systems

In fact, introducing the matrices of constraint systems serves a greater purpose than just solving the non-deducible constraints. Indeed, deciding the symbolic equivalence of two row matrices contains two main issues:

- matching an existing solution from one matrix to the other;
- and deciding whether the two resulting frames are statically equivalence or not.

The idea behind matrices with several lines is to keep all the guesses on static equivalence into a single matrix. Intuitively, when we guess the form of the solutions, we split the matrix into two matrices. However, when we guess an equality between terms or a property on static equivalence, we gather the informations in the same matrix. This leads us to consider two kinds of application: *internal* and *external*. The transformation rules Ded-st, Eq-left-left, Eq-left-right and

DEST will be applied internally whereas CONS(X, f), AXIOM(X, path) and EQ-RIGHT-RIGHT(X, ξ) will be applied externally when $X \in S_2$ and internally otherwise (*i.e.* $X \notin S_2$).

Let (M, M') be a pair of matrices of constraint systems having the same structure. In particular, M and M' have the same number of lines, say n. Let $M = [V_1, \ldots, V_n]$ and $M' = [V'_1, \ldots, V'_n]$. Let RULE$(\tilde{p})$ be an instance of a transformation rule and i be an integer representing a line, *i.e.* $1 \leq i \leq n$.

An *internal application* of RULE$(\tilde{p})$ on the i^{th} line of the pair (M, M') yields a pair of matrices $(\tilde{M}, \tilde{M'})$ such that:

$$\tilde{M} = [V_1, \ldots, V_{i-1}, W_{i,1}, W_{i,2}, V_{i+1}, V_n] \quad [V'_1, \ldots, V'_{i-1}, W'_{i,1}, W'_{i,2}, V'_{i+1}, V'_n] = \tilde{M'}$$

where $(W_{i,1}, W'_{i,1})$ and $(W_{i,2}, W'_{i,2})$ are the pair of row matrices obtained by applying RULE$(\tilde{p})$ on (V_i, V'_i). Note that, since the two matrices M and M' have the same structure, the two row matrices V_i and V'_i have also the same structure and we can rely on Definition 7.10. We say that an instance RULE$(\tilde{p})$ of a rule is *internally applicable* on (M, M') on line i if RULE$(\tilde{p})$ is applicable on (V_i, V'_i).

An *external application* of RULE$(\tilde{p})$ on (M, M') yields two pairs of matrices $(\tilde{M}_1, \tilde{M}'_1)$ and $(\tilde{M}_2, \tilde{M}'_2)$ such that:

$$\begin{array}{ll} \tilde{M}_1 = [W_{1,1}, \ldots, W_{n,1}] & [W'_{1,1}, \ldots, W'_{n,1}] = \tilde{M}'_1 \\ \tilde{M}_2 = [W_{1,2}, \ldots, W_{n,2}] & [W'_{1,2}, \ldots, W'_{n,2}] = \tilde{M}'_2 \end{array}$$

where $(W_{i,1}, W'_{i,1})$ and $(W_{i,2}, W'_{i,2})$ are the pair of row matrices obtained by applying RULE$(\tilde{p})$ on (V_i, V'_i) for each $i \in \{1, \ldots, n\}$.

Remark. Unfortunately, all the constraint systems in M and M' do not have necessarily the same structure, but only the same shape. When the external application involved is an instance of a rule CONS, it is easy to see that having the same shape will ensure that the rule can be applied on each row of the matrices. Regarding an external application of the rule AXIOM(X, path), we have to be a little more specific. Since the constraint systems have the same shape and we know that $X \in S_2$, we can ensure that X occurs in each constraint system. However, it could happen that some rows do not contain the required frame element. By convention, in such a pair (V_i, V'_i) of row matrices, the resulting pairs of row matrices are $(W_{i,1}, W'_{i,1}) \stackrel{\mathsf{def}}{=} (\bot, \bot)$ and $(W_{i,2}, W'_{i,2}) \stackrel{\mathsf{def}}{=} (V, V')$.

Example 7.21. *All the rules applied in Example 7.20 are internal rules.*

7.4 Our strategy

In the previous sections, we defined the conditions of application of all the rules described in Figure 7.1 and 7.2. Furthermore, we also introduced the notion of *strong application* in order to solve some basic termination problem. Nevertheless, this *strong application* is not sufficient to ensure termination and soundness. Thus, we describe a strategy on the application of the rule.

By applying the rules described in Figure 7.1 and 7.2, we want to transform the constraint systems into *solved constraint system*, i.e. either $\bot$ or constraint system such that:

1. For all $X, i \stackrel{?}{\vdash} u \in D$, $X \in S_2$ and u is a variable distinct from the right hand side of any other deducible constraint.
2. Eq does not contain variable that are universally quantified, and for all $u \stackrel{?}{\neq} v$ in Eq, we have that u, v do not contain any names.

The first phase of our strategy consists of applying rules to satisfy the first property, without taking care of the inequations. Then, the second phase of our strategy will reduce the constraint

systems into solved ones. Both phases will be described as a succession of different steps (see below).

The description of the strategy is always local to the pair of matrices of constraint systems we consider. Consider for example an pair of matrices (M_0, M'_0) of initial constraint systems. The strategy starts with the first phase. If a rule of the first phase is applicable on (M_0, M'_0), *e.g.* an internal rule, we apply this rule on (M_0, M'_0) which yields a pair of matrices of constraint systems (M_1, M'_1). Then, if a rule of Phase 1 is applicable on (M_1, M'_1), *e.g.* an external rule, we apply this rule on (M_1, M'_1) which yields two pair of matrices of constraint systems (M_{12}, M'_{12}) and (M_{22}, M'_{22}). The strategy continues independently on (M_{12}, M'_{12}) and (M_{22}, M'_{22}). Typically, if a rule of Phase 1 is applicable on (M_{12}, M'_{12}) then we apply this rule on (M_{12}, M'_{12}) and so on. In parallel, if no rule of Phase 1 is applicable on (M_{22}, M'_{22}) then we go into Phase 2 and check if one of the rules of Phase 2 is applicable on (M_{22}, M'_{22}) and so on.

To summarise, the application of our strategy on a pair of matrices of initial constraint systems yields a binary tree (not necessary full) where each node of this tree is a pair of matrices of constraint systems, and two nodes of same height in the binary tree are not necessary in the same phase / step of the strategy.

When describing the phase and strategy, we denote (M, M') the pair of matrices of constraint system on which we wish to apply a rule.

7.4.1 First phase of the strategy

Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$. We associate, to each instance of a rule applied on a constraint system $\mathcal{C}$, an integer, namely its *support*, as follows:

— support of $\textsc{Cons}(X, \mathsf{f})$ is i where $X, i \overset{?}{\vdash} u \in D$

— support of $\textsc{Dest}(\xi, \ell \to r, i)$ is i

— support of $\textsc{Axiom}(X, \mathsf{path})$ is i where $X, i \overset{?}{\vdash} u \in D$

— support of $\textsc{Eq-left-left}(\xi_1, \xi_2)$ is $i = \max(i_1, i_2)$ when $\xi_1, i_1 \rhd u_1 \in \Phi$ and $\xi_2, i_2 \rhd u_2 \in \Phi$

— support of $\textsc{Eq-left-right}(\xi_1, X_2)$ is i_1 when $\xi_1, i_1 \rhd u_1 \in \Phi$

— support of $\textsc{Eq-right-right}(X, \xi)$ is i when $X, i \overset{?}{\vdash} u \in vars^2(D)$

— support of $\textsc{Ded-st}(X, \xi)$, is m (*i.e.* the maximal index that occurs in $\mathcal{C}$)

The first phase is a cycle of five steps. Each cycle is characterised by one parameter: the support s of the rules we will apply during this cycle. The first cycle will consider rules having support 1, then the second cycle will take care of the rules of support 2, and so on. Figure 7.5 represents the evolution between the different steps. The exact description of each step is given below.

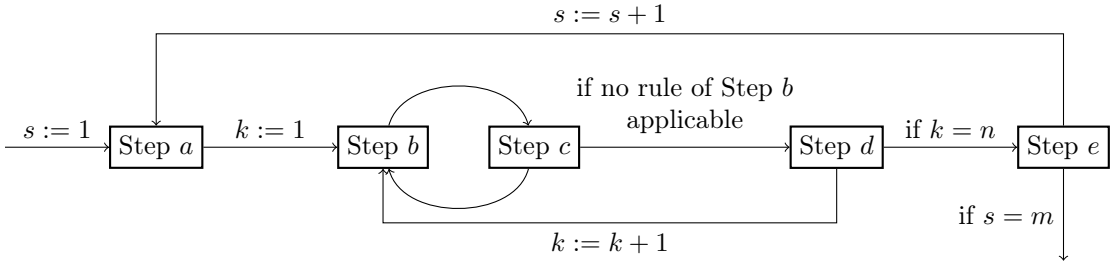
7.4.1.1 Step a

We apply the rules Dest and Eq-left-right, with support equal to s, as long as possible with priority on the rule Eq-left-right. The application of those rules has to be a strong application for at least one constraint system in the line of the pair of matrices on which we apply the rule.

In order to obtain some necessary properties on the matrices of constraint system, when a rule $\textsc{Dest}(\tilde{p})$ or $\textsc{Eq-left-right}(\tilde{p})$ is applied on one line of the matrix, we will apply the same rule with similar parameter on each line of the matrix.

More specifically,

— if $\textsc{Dest}(\xi, \ell \to r, s)$ is applied on a line of (M, M') where (ξ, i) belongs to the structure of a constraint systems of this line, then for all line L' of (M, M') where there exists ξ' such that (ξ', i) belongs to the structure of the constraint systems in L' and $\mathsf{path}(\xi') = \mathsf{path}(\xi)$, we apply $\textsc{Dest}(\xi', \ell \to r, s)$ on L'.



where s is the parameter corresponding to the support, k is the parameter corresponding to the index of the column, n is the number of column in the pair of matrices and m is the size of the frame of one of the constraint systems in the pair of matrices.

Figure 7.5: Representation of the first phase of the strategy

— if Eq-left-right(ξ, X) is applied on a line of (M, M') where (ξ, i) belongs to the structure of the constraint systems of this line, then for all line L' of (M, M') where there exist ξ' such that (ξ', i) belongs to the structure of the constraint systems in L' and $\mathsf{path}(\xi') = \mathsf{path}(\xi)$, we apply Eq-left-right$(\xi', X)$ on L'.

At the end of this step, the frame of any constraint system in the matrices is fixed for the support s, *i.e.* there will be no new frame element added on the frame with a support equal to s.

7.4.1.2 Cycle of steps b, c, d (solving the deducible constraints)

After this first step, the strategy is once again a cycle of three different steps, denoted b, c, d. This time, the parameter of each cycle is the index of the column on which we will apply the rule. Hence at this point, s is the parameter of the main cycle and k is the index of the working column. Each of this cycle alternates steps b and c, and then ends with step d (see Figure 7.5).

Given a pair of matrices (M, M') such that M (resp. M') has n columns (resp. n'), we say that the k^{th} column of (M, M') is either the k^{th} column of M if $k \leq n$; or else the $(k-n)^{\text{th}}$ column of M' if $k > n$. If $n + n' < k$ then the k^{th} column of (M, M') is not defined.

7.4.1.3 Step b

We apply the internal applications of the rules Eq-right-right, Eq-left-left, Cons and Axiom with support less than s, as long as possible. In case $s = |\Phi|$ where Φ is a frame of one of the constraint system in the pair of matrices, we also apply the internal applications of Ded-st. Moreover, the instance of the rule must be strongly applicable on the constraint system in the i^{th} line and k^{th} column of the pair of matrices, where i is the index of the line on which the rule is applied.

7.4.1.4 Step c

For this step, we need to consider the following set: For a constraint system $\mathcal{C}$, we defined $\mathsf{X}^1(\mathcal{C})$ as a subset of $\mathcal{X}^1$ such that:

$$\mathsf{X}^1(\mathcal{C}) = \{x \in \mathcal{X}^1 \mid (Y, j \overset{?}{\vdash} x) \in D(\mathcal{C}), Y \notin S_2(\mathcal{C})\}$$

Furthermore, we define a lexical measure on deducible constraints of a constraint system $\mathcal{C}$, denoted $\mathcal{L}^1_{\mathcal{C}}()$, such that : for all $(X, j \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $vars(u) \cap \mathsf{X}^1(\mathcal{C}) \neq \emptyset$ and $X \in S_2$:

$$\mathcal{L}^1_{\mathcal{C}}(X, j \overset{?}{\vdash} u) = (j, \min\{p \mid u|_p \in \mathsf{X}^1(\mathcal{C})\})$$

If for all constraint system $\mathcal{C}$ in the k^{th} column of (M, M'), $\mathsf{X}^1(\mathcal{C})$ is empty then no rule is applicable for this step. Else, let i_0 be an index of the line of (M, M'), let $\mathcal{C}_0$ be the constraint system in the i_0^{th} line and k^{th} column of (M, M'), and let $(X_0, j_0 \overset{?}{\vdash} u_0) \in D(\mathcal{C}_0)$ such that $vars(u_0) \cap \mathsf{X}^1(\mathcal{C}_0) \neq \emptyset$, $X \in S_2(\mathcal{C}_0)$, and:

$$\mathcal{L}^1_{\mathcal{C}_0}(X_0, j_0 \overset{?}{\vdash} u_0) = \min \left\{ \mathcal{L}^1_{\mathcal{C}}(Z, \ell \overset{?}{\vdash} v) \;\middle|\; \begin{array}{l} \mathcal{C} \text{ is on the } i^{\text{th}} \text{ line and } k^{\text{th}} \text{ column of } (M, M'), \\ i \in \mathbb{N}, (Z, \ell \overset{?}{\vdash} v) \in D(\mathcal{C}), \\ vars(v) \cap \mathsf{X}^1(\mathcal{C}) \neq \emptyset, Z \in S_2(\mathcal{C}) \end{array} \right\}$$

We apply by order of preference:

1. the internal rule Eq-right-right(Y, X_0) on the i_0^{th} line for any Y such that $(Y, j \overset{?}{\vdash} w) \in D(\mathcal{C}_0)$, $j \leq s$, $Y \notin S_2(\mathcal{C}_0)$ and Eq-right-right(Y, X_0) is strongly applicable on $\mathcal{C}_0$;
2. the external rule Cons(X_0, f), for any $\mathsf{f} \in \mathcal{F}_c$ such that Cons(X_0, f) is strongly applicable on $\mathcal{C}_0$;
3. the external rule Axiom(X_0, path), for any path such that Axiom(X_0, path) is strongly applicable on $\mathcal{C}_0$

Intuitively, the choice of $X_0, j_0 \overset{?}{\vdash} u_0$ is done over all the constraint systems in the k^{th} column such that one of the variables appearing in the internal deducible constraint of $\mathcal{C}_0$ is the closest to the root of u_0. This choice is made for the sole purpose that only the application of the external rule Axiom(X_0, f) can instantiate the internal deducible constraints of the constraint systems in the k^{th} column.

7.4.1.5 Step d

We apply the external application of the rules Eq-right-right, Cons and Axiom as long as they are strongly applicable on $M_{i,k}$ by increasing order on the index of the line i (if we assume that k corresponds to an index of the matrix M in the pair (M, M')).

To be more specific, if an instance $R_1(\tilde{p}_1)$ is strongly applicable on $M_{i_1,k}$, $R_2(\tilde{p}_2)$ is strongly applicable on $M_{i_2,k}$, and $i_1 \leq i_2$ then we apply the rule $R_1(\tilde{p}_1)$ on (M, M').

7.4.1.6 Step e (solving the non-deducible constraints)

This last step consists of solving the non-deducible constraints in the matrices. This step will not consist of applying some rules but replacing some constraint systems in the matrices by $\bot$.

Formally, for each constraint system $\mathcal{C}$ in the matrix, if there exists a constraint system $\mathcal{C}'$ in the same column as $\mathcal{C}$, a recipe ξ, such that:

— $(\xi, s \rhd u) \in \Phi(\mathcal{C}')$ for some i and u; and

— for all $(\xi', s \rhd v) \in \Phi(\mathcal{C})$, $\mathsf{path}(\xi) \neq \mathsf{path}(\xi')$

then we replace the constraint system $\mathcal{C}$ in the matrix by $\bot$. This corresponds to solving the non-deducible constraints introduced by the rule Dest.

Furthermore, when s is equal to the size of the frames in (M, M') then for each constraint system $\mathcal{C}$ in (M, M'), if there exists a constraint system $\mathcal{C}'$ in the same column as $\mathcal{C}$ such that for all $(\xi, i \rhd u) \in \Phi(\mathcal{C})$, for all $(\xi', i' \rhd u') \in \Phi(\mathcal{C}')$, for all $\mathsf{f} \in \mathcal{F}_c$,

— $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ and $i = i'$; and

— $ND(\mathcal{C}) \vDash \forall \tilde{x} u \neq \mathsf{f}(x_1, \ldots, x_n) \vee s \overset{?}{\not\vdash} x_1 \vee \ldots \vee s \overset{?}{\not\vdash} x_n$ where $\tilde{x} = x_1 \ldots x_n$ are variables; and

— there exists $X_1, \ldots, X_n \in vars^2(\mathcal{C}')$ such that $\mathsf{C}\lfloor \mathsf{f}(X_1, \ldots, X_n)\theta \rfloor_{\Phi(\mathcal{C}')} \mathsf{acc}^1(\mathcal{C}') = u'$, θ is the most general unifier of the equation in $Er(\mathcal{C}')$

then we replace the constraint system $\mathcal{C}$ in the matrix by $\bot$. This corresponds to solving the non-deducible constraints introduced by the rule Ded-st.

Typically, each constraint system that is replaced by $\bot$ does not have a solution due to the non-deducible constraint.

7.4.2 Second phase of the strategy

As stated at the beginning of this section, this second phase will take care of the inequations. First of all, after the first phase of the strategy, the rules DEST, EQ-LEFT-RIGHT, EQ-LEFT-LEFT and DED-ST will never be applicable anymore for any parameters; thus the only rules that will be applied are CONS, AXIOM and EQ-RIGHT-RIGHT. Furthermore these rules will always be applied as external rules.

The second phase of the strategy is composed of three steps that we denote a,b,c. Figure 7.6 represents the evolution between the different steps.

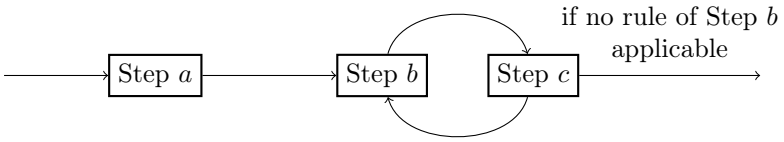


Figure 7.6: Representation of the second phase of the strategy

7.4.2.1 Step a (Getting rid of the universal variables)

We apply the rules $\text{CONS}(\tilde{p})$ and $\text{AXIOM}(\tilde{p})$ as long as they satisfy the following conditions on at least one constraint system of the matrices:

— $\text{CONS}(X, \mathsf{f})$: either $\text{CONS}(X, \mathsf{f})$ is strongly applicable or there exists an atomic statement $t \overset{?}{\neq} w$ in Eq (actually at this stage t will be a variable) for which there exists a universal variable $y \in vars^1(w)$.

— $\text{AXIOM}(X, \mathsf{path})$: either $\text{AXIOM}(X, \mathsf{path})$ is strongly applicable or there exists an atomic statement $u \overset{?}{\neq} w$ in Eq (actually at this stage u will be a variable) for which there exists a universal variable $y \in vars^1(w)$.

We show that at the end of this step, all the universal variables are removed (see Appendix C.4.3.9)

7.4.2.2 The association tables

After Step a, the strategy is once again a cycle of the two last steps, i.e. Steps b and c. Step b consists of matching the inequations on every constraint systems on the matrices using the rules CONS and EQ-RIGHT-RIGHT, while step b instantiates variables using the rule AXIOM. However, for the strategy to terminate, we have to keep track of which inequations were matched using the rule EQ-RIGHT-RIGHT. Hence we introduce the notion of *association table* for disjunction of first order inequations. Intuitively, for each constraint system $\mathcal{C}$ in the matrices, we associate an association table where the index are the disjunctions of inequation between first order terms in $\mathcal{C}$ that comes from an application of the rule EQ-RIGHT-RIGHT. At last, the disjunction of inequation will always be associated to a disjunction of inequation between terms in $\mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}), \mathcal{X}^2)$, i.e. between contexts of recipes.

Normalisation of a disjunction of inequation between contexts of recipe: We define a transformation on formulas of context of recipe, denote $\cdot\wr$, by the following reductions: For all E disjunction of inequations of contexts of recipes, $\mathsf{f} \in \mathcal{F}_c$, and $\xi_i, \zeta_i \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}), \mathcal{X}^2)$,

$$E \vee \mathsf{f}(\xi_1, \ldots, \xi_n) \overset{?}{\neq} \mathsf{f}(\zeta_1, \ldots \zeta_n)] \quad \rightsquigarrow \quad E \vee \xi_1 \overset{?}{\neq} \zeta_1 \vee \ldots \vee \xi_n \overset{?}{\neq} \zeta_n$$

$$E \vee \beta \overset{?}{\neq} \beta \quad \rightsquigarrow \quad E$$

Comparing to the normalisation on formulas of first order term, this normalisation is very simple and slightly reduces formulas. Indeed, we will use these formula only as a way to rebuilt formulas on first order term. Hence, we don't want the formula $\mathsf{g} \cdot ax_1 \stackrel{?}{\neq} ax_2$ to be reduced to false because even if the path are different, it is possible that the associated term aren't, i.e. $(\mathsf{g} \cdot ax_1)\mathsf{acc}^1(\mathcal{C})$ and $ax_2\mathsf{acc}^1(\mathcal{C})$.

At last, we define the maximal parameter for a context of a recipe, denoted $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\beta)$, such that:

$$\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\beta) = \max\{i \mid (w \cdot ax_i) \in st(\beta) \text{ with } w \in \mathcal{F}_d^* \text{ or } (Y, i \stackrel{?}{\vdash} v) \in D \text{ with } Y \in st(\beta)\}$$

Evolution of an association table: Let $\mathcal{C}$ be a constraint system an T its association table. Let C_1, C_2 and T_1, T_2 their respective association table such that $\mathcal{C}_1$ and $\mathcal{C}_2$ are the two constraint system produced by the application of a rule on $\mathcal{C}$. We show here how T_1 and T_2 evolved by default: For all $i \in \{1, 2\}$, for all disjunction D of inequations of first order terms, if $T[D]$ exists and there exists E' and D' such that $Eq(\mathcal{C}_i) = E' \wedge D'$ with $D' = D\mathsf{mgu}(Eq(\mathcal{C}_i))\downarrow$ then we have that $T_i[D'] := T[D]\{X \mapsto \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_i)} \mid X \in \mathrm{dom}(\theta)\}\wr$ where $\theta = \mathsf{mgu}(Er(\mathcal{C}_i))$.

When describing the Step b and c of the strategy, the additional operations on the association tables we may apply will always be done after the default transformation described above. At last, the associations table of each constraint systems in the matrices is empty at the end of Step a.

7.4.2.3 Step b (Splitting the inequations)

In this step, we apply as long as we can the rules Cons and Eq-right-right, then we go to the step c. However, their application depends on the association table of the constraint systems.

We say that Eq-right-right(X, ξ) is applicable for Step b if there exists a constraint system $\mathcal{C}$ in the matrices with its association table T such that Eq-right-right(X, ξ) is not useless on on $\mathcal{C}$, $Eq(\mathcal{C}) = E' \wedge (E'' \vee u \stackrel{?}{\neq} v)$ for some E', E'' and u, v are the terms described in Figure 7.2, and

— either $T[E'' \vee u \stackrel{?}{\neq} v] = \bot$.

— or else $T[E'' \vee u \stackrel{?}{\neq} v] = D \vee X \stackrel{?}{\neq} \xi$ with $(\mathcal{F}_d^* \cdot \mathcal{AX}) \cap st(D) \neq \emptyset$ for some D.

After application of Eq-right-right(X, ξ), if we guessed that the inequation holds, then we add in each association table T' of each constraint system $\mathcal{C}'$ of the matrices the following entry: $T[u' \stackrel{?}{\neq} v'] := X \vee \xi$ where $u' = X\mathsf{acc}^1(\mathcal{C}')$ and $v' = \xi\mathsf{acc}^1(\mathcal{C}')$.

We say that Cons(X, f) is applicable for Step b if there exists a constraint system $\mathcal{C}$ in the matrices with its association table T such that Cons(X, f) is not useless on $\mathcal{C}$, $Eq(\mathcal{C}) = E' \wedge (E'' \vee t \neq u)$ for some E', E'' and t is the term described in Figure 7.1 (actually t is a variable at this stage), and

— either Cons(X, f) is strongly applicable on $\mathcal{C}$;

— or $T[E'' \vee t \stackrel{?}{\neq} u] = \bot$ with $\mathsf{root}(u) = \mathsf{f}$ and either $st(u) \cap \mathcal{N} \neq \emptyset$ or $\mathsf{ind}_{\mathcal{C}}(t) < \mathsf{ind}_{\mathcal{C}}(u)$

— or $T[E''] = D \vee X \stackrel{?}{\neq} \xi$ with $\mathsf{root}(\xi) = \mathsf{f}$ and either $(\mathcal{F}_d^* \cdot \mathcal{AX}) \cap st(\xi) \neq \emptyset$ or $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X) < \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\xi)$.

7.4.2.4 Step c (Instantiating the variables)

In this last step, we apply the rule Axiom(X, path) as long as there is at least one constraint system $\mathcal{C}$ in the matrices satisfying the following conditions:

— either Axiom(X, path) is strongly applicable on $\mathcal{C}$;

— or $Eq(\mathcal{C}) = E \wedge (D \vee u \overset{?}{\neq} a)$ and $T[D \vee u \overset{?}{\neq} a] = \bot$ for some E, D, for some $a \in \mathcal{N}$ and u is the term described in Figure 7.1

7.4.2.5 Removal transformation

During Step b and Step c, for any association table T, if there exists an entry $T[\bigvee_i u_i \overset{?}{\neq} v_i] = \bigvee_i \xi_i \overset{?}{\neq} \xi'_i$ such that ξ_i or ξ'_i is in $(\mathcal{F}_d^* \cdot \mathcal{AX})$ for all i, then we remove this entry, i.e. $T[\bigvee_i u_i \overset{?}{\neq} v_i] := \bot$.

Example 7.22. *Consider the frames Φ_1 and Φ_2 of Example 7.12. Moreover consider the matrix $M = [\mathcal{C}; \mathcal{C}']$ such that:*

$$M = \left[\begin{array}{c|c} \left\{ \begin{array}{l} \{X, 1 \overset{?}{\vdash} x;\ Y, 3 \overset{?}{\vdash} y\} \\ \Phi_1 \\ x \overset{?}{\neq} \mathsf{senc}(a, k_1) \wedge x \overset{?}{\neq} y \end{array} \right. & ; \left\{ \begin{array}{l} \{X, 1 \overset{?}{\vdash} x';\ Y, 3 \overset{?}{\vdash} y'\} \\ \Phi_2 \\ x' \overset{?}{\neq} b \end{array} \right. \end{array} \right]$$

Assume that their respective association tables, denoted T and T' are empty. We can apply the rule Eq-right-right(X, Y) *since $T_1[x \overset{?}{\neq} y] = \bot$. Thus, we obtain two matrices of constraint systems $M_1 = [\mathcal{C}_1; \mathcal{C}'_1]$ and $M_2 = [\mathcal{C}_2; \mathcal{C}'_2]$ such that:*

$$M_1 = \left[\begin{array}{c|c} \bot ; & \left\{ \begin{array}{l} \{X, 1 \overset{?}{\vdash} x';\} \\ \Phi_2 \\ y' \overset{?}{=} x' \wedge x' \overset{?}{\neq} b \wedge \end{array} \right. \end{array} \right]$$

and

$$M_2 = \left[\begin{array}{c|c} \left\{ \begin{array}{l} \{X, 1 \overset{?}{\vdash} x;\ Y, 3 \overset{?}{\vdash} y\} \\ \Phi_1 \\ x \overset{?}{\neq} \mathsf{senc}(a, k_1) \wedge x \overset{?}{\neq} y \end{array} \right. & ; \left\{ \begin{array}{l} \{X, 1 \overset{?}{\vdash} x';\ Y, 3 \overset{?}{\vdash} y'\} \\ \Phi_2 \\ x' \overset{?}{\neq} b \wedge x' \overset{?}{\neq} y' \end{array} \right. \end{array} \right]$$

Moreover, if we denote T_1, T'_1, T_2, T'_2 the respective association tables of $\mathcal{C}_1, \mathcal{C}'_1, \mathcal{C}_2, \mathcal{C}'_2$, then T_1 and T'_1 are empty, $T_2[x \overset{?}{\neq} y] = X \overset{?}{\neq} Y$ and $T'_2[x' \overset{?}{\neq} y'] = X \overset{?}{\neq} Y$. We now focuses on M_2. We can apply Cons(X, senc) *since $x \overset{?}{\neq} \mathsf{senc}(a, k_1)$ is in $Eq(\mathcal{C}_2)$ and $T_2[x \overset{?}{\neq} \mathsf{senc}(a, k_1)] = \bot$. Thus it yields once again two matrices $M_{21} = [\mathcal{C}_{21}; \mathcal{C}_{22}]$ and M_{22} such that:*

$$\mathcal{C}_{21} = \left\{ \begin{array}{l} \{X_1, 1 \overset{?}{\vdash} x_1;\ X_2, 1 \overset{?}{\vdash} x_2;\ Y, 3 \overset{?}{\vdash} y\} \\ \Phi_1 \\ x \overset{?}{=} \mathsf{senc}(x_1, x_2) \wedge \left(x_1 \overset{?}{\neq} a \vee x_2 \overset{?}{\neq} k_1\right) \wedge \mathsf{senc}(x_1, x_2) \overset{?}{\neq} y \\ X \overset{?}{=} \mathsf{senc}(X_1, X_2) \end{array} \right.$$

and

$$\mathcal{C}'_{21} = \left\{ \begin{array}{l} \{X, 1 \overset{?}{\vdash} x'_1;\ X_2, 1 \overset{?}{\vdash} x'_2;\ Y, 3 \overset{?}{\vdash} y'\} \\ \Phi_2 \\ x' \overset{?}{=} \mathsf{senc}(x'_1, x'_2) \wedge \mathsf{senc}(x'_1, x'_2) \overset{?}{\neq} y' \\ X \overset{?}{=} \mathsf{senc}(X_1, X_2) \end{array} \right.$$

If we denote T_{21} and T'_{21} their association tables, then $T_{21}[\mathsf{senc}(x_1, x_2) \overset{?}{\neq} y] = \mathsf{senc}(X_1, X_2) \overset{?}{\neq} Y$ and $T'_{21}[\mathsf{senc}(x'_1, x'_2) \overset{?}{\neq} y'] = \mathsf{senc}(X_1, X_2) \overset{?}{\neq} Y$. At this point, no rule of Step b is applicable thus

we go into Step c and try to apply the rule AXIOM. *Note that* $T_{21}[x_1 \stackrel{?}{\neq} a \vee x_2 \stackrel{?}{\neq} k_1] = \bot$ *hence we apply* AXIOM(X_1, ax_1) *on* M_{21}. *It yields once again two matrices* $M_{211} = [\mathcal{C}_{211}; \mathcal{C}'_{211}]$ *and* M_{212} *such that:*

$$\mathcal{C}_{211} = \begin{cases} \{X_2, 2 \stackrel{?}{\vdash} x_2;\ Y, 3 \stackrel{?}{\vdash} y\} \\ \Phi_1 \\ x_1 \stackrel{?}{=} a \wedge x \stackrel{?}{=} \mathsf{senc}(a, x_2) \wedge x_2 \stackrel{?}{\neq} k_1 \wedge \mathsf{senc}(a, x_2) \stackrel{?}{\neq} y \\ X_1 \stackrel{?}{=} ax_1 \wedge X \stackrel{?}{=} \mathsf{senc}(ax_1, X_2) \end{cases}$$

and

$$\mathcal{C}'_{211} = \begin{cases} \{X, 1 \stackrel{?}{\vdash} x'_1;\ X_2, 2 \stackrel{?}{\vdash} x'_2;\ Y, 3 \stackrel{?}{\vdash} y'\} \\ \Phi_2 \\ x'_1 \stackrel{?}{=} a \wedge x' = \mathsf{senc}(a, x'_2) \wedge \mathsf{senc}(a, x'_2) \stackrel{?}{\neq} y' \\ X_1 \stackrel{?}{=} ax_1 \wedge X \stackrel{?}{=} \mathsf{senc}(ax_1, X_2) \end{cases}$$

If we denote T_{211} *and* T'_{211} *their association tables, then* $T_{211}[\mathsf{senc}(a, x_2) \stackrel{?}{\neq} y] = \mathsf{senc}(ax_1, X_2) \stackrel{?}{\neq} Y$ *and* $T'_{211}[\mathsf{senc}(a, x'_2) \stackrel{?}{\neq} y'] = \mathsf{senc}(ax_1, X_2) \stackrel{?}{\neq} Y$.

We will show that this strategy always terminates on row matrices of initial constraint systems. Section 8.4 describes the essential steps of the proof of this theorem.

Theorem 7.1. *Given a pair of row matrices of initial constraint system* (M, M')*, the application of the strategy described in Subsection 7.4 on* (M, M') *terminates.*

7.4.3 The final test

In this section and the previous one, we described how to apply the rules on a pair of matrices of initial constraint systems and also a strategy that terminates. Thus the last thing we need to define is the final test such that this test succeeds if, and only if, the pair of matrices of constraint systems are symbolically equivalent.

By applying the rules on a initial pair of row matrices, we obtain a tree where every nodes (including the leaves) is a pair of matrices. Thus the final test focuses on the leaves of this tree.

Definition 7.13 (test on the leaves)**.** *Let* (M, M') *be a pair of matrices of constraint systems with* n *lines and* m *(resp.* m'*) columns. We define a predicate on a pair of matrices of constraint systems, denoted* $\mathsf{LeafTest}$*, such that* $\mathsf{LeafTest}(M, M') = \mathsf{true}$ *if, and only if : for all* $i \in \{1 \dots n\}$*,*

$$\exists j \in \{1, \dots, m\}. M_{i,j} \neq \bot \ \text{ is equivalent to } \exists j' \in \{1, \dots, m'\}. M'_{i,j'} \neq \bot$$

With the predicate defined in Definition 7.13, we can finally give one of the main theorem of this paper :

Theorem 7.2. *Let* (M_0, M'_0) *be an initial pair of row matrices of constraint systems. We have that* $M_0 \approx_s M'_0$ *if, and only if, all leaves of a tree, whose root is labeled with* (M_0, M'_0) *and which followed the strategy given in Subsection 7.4, satisfy the predicate* $\mathsf{LeafTest}$.

The next chapter is dedicated to the proof of Theorem 7.1 and 7.2.

Chapter 8

Proof of the decision procedure

Contents

Proving Theorems 7.1 and 7.2 is difficult and highly technical. This is for various reasons. First of all, the large number of transformation rules used in the algorithm usually tends to lengthen the proofs. Indeed, in most of the proofs, we have to do a case analysis on the rules. This is even more true for the termination proof since we have to consider the different rules but also the different phases and steps of the strategy. Second of all, the constraint systems are are composed of several elements which make them difficult to manipulate in the proofs. But most of all, the technicality of the proofs comes from the fact that the soundness and termination proofs overlap. A good example is the Step e of the first phase of the strategy. Without this step, our algorithm is not sound anymore.

This chapter is dedicated to the proofs of Theorem 7.1 and 7.2. More specifically, we provide in Section 8.1 the different properties that the matrices of constraint systems satisfy during the execution of the algorithm. In Section 8.2, we explain the main steps of the proofs of soundness and completeness of our transformation rules but also of our final test on a pair of matrices of constraint systems (see Section 8.3). At last, in Section 8.4, we provide all the necessary measures to prove termination. However, most of the actual proofs are in Appendix C.

8.1 Invariants

8.1.1 Invariants independent from the strategy

First it is easy to see that applying our transformation rules preserves the fact that matrices share the same structure. This is stated in the following lemma (Proof in Appendix C.2.1).

Lemma 8.1. *Let (M, M') be a pair of matrices of constraint systems such that M and M' have the same structure. Any internal (resp. external) application of a rule in Figure 7.1 and/or Figure 7.2 transforms the pair (M, M') on a pair (M_1, M_1') (resp. two pairs (M_1, M_1') and (M_2, M_2')) of matrices having the same structure.*

Initially, we consider two row matrices of initial constraint systems. By applying our rules, the constraint systems we consider become more and more complex. Nevertheless, we will show that the strategy and the rules ensure several invariants on constraint systems that will be used in the proof of completeness, soundness and termination. A constraint system satisfying all the invariants will be called *well-formed* (see Definition 8.2).

Most of the invariants focus are about structural properties of the frame. Those are usually direct consequences of the application of the rules Dest, Axiom and Cons. For example, we will state that for any frame element $(\xi, i \rhd u)$, the path of the recipe ξ is defined and closed. Looking at the rules, this invariant seems almost trivial since the only rule that generates a new frame element is Dest which apply a destructor on a recipe.

Other invariants focus on the formula Er. One of these invariants illustrates the instantiation of a recipe variable X given the equations in Er. Moreover, since some recipes of the frames are not always ground, the invariants of a well-formed constraint system detail by which parameters such recipe can be instantiated. To describe this, we introduce the notion of *maximal parameter of a recipe.*

Definition 8.1 (maximal parameter of a recipe)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system and let $\xi \in \Pi_n$ such that $vars(\xi) \subseteq vars^2(D)$. We define the maximal parameter of ξ in $\mathcal{C}$, denoted $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$, such that:*

$$\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi) = \max\{i \mid ax_i \in st(\xi) \text{ or } (Y, i, \stackrel{?}{\vdash} v) \in D \text{ with } Y \in st(\xi)\}$$

For all constraint systems $\mathcal{C}$, we denote by $\mathsf{mgu}(Eq(\mathcal{C}))$ (resp. $\mathsf{mgu}(Er(\mathcal{C}))$)the most general unifier of all the equations in $Eq(\mathcal{C})$ (resp. $\mathsf{mgu}(Er(\mathcal{C}))$).

Our invariants are not independent. For example, we state that for any frame element $(\xi, i \rhd u)$, the path of the recipe ξ is defined and closed (item 1 Definition 8.2). From this, we can deduce that, given θ and θ' two substitutions and ζ a recipe in Π_n, we have that $\mathsf{C}\lfloor\zeta\rfloor_{\Phi\theta} = \mathsf{C}\lfloor\zeta\rfloor_{\Phi\theta'}$. This comes from the fact that the path $\mathsf{path}(\xi)$ of any frame element $(\xi, i \rhd u)$ in $\Phi\theta$ is closed and so do not depend on the substitution θ or θ'.

Definition 8.2 (well-formed)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. We say that $\mathcal{C}$ is* well-formed *if it satisfies the following properties:*

Invariants on the frame Φ: *for all $(\xi, i \rhd u) \in \Phi$,*

1. $\mathsf{path}(\xi)$ *exists and is closed. Moreover, for all distinct frame elements $(\xi_1, i_1 \rhd u_1)$ and $(\xi_2, i_2 \rhd u_2)$ in Φ, we have that $\mathsf{path}(\xi_1) \neq \mathsf{path}(\xi_2)$.*
2. *if $\mathsf{path}(\xi) = \mathsf{f} \cdot w$ then there exists $(\xi', j \rhd v) \in \Phi$ such that $j \leq i$, $\mathsf{path}(\xi') = w$, $\xi' \in st(\xi)$ and $u \in st(v)$.*
3. $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi) \leq i$
4. *for all $X \in vars^2(\xi)$, for all $x \in vars^1(X\mathsf{acc}^1(\mathcal{C}))$, there exists $(\zeta, k \rhd w) \in \Phi$ such that $k \leq i$ and $x \in vars^1(w)$.*
5. *for all ground substitution λ, if for all $X \in vars^2(\xi)$, we have that $(X\lambda)\Phi\lambda\downarrow = v\lambda$ where $(X, j \stackrel{?}{\vdash} v) \in D$, then $(\xi\lambda)(\Phi\lambda)\downarrow = u\lambda$.*

Other invariants: *let* $\theta = \mathsf{mgu}(Er)$

6. *any inequation in* Er *are:*
   - *either of the form* $X \neq \xi$ *and there exists* $i \in \mathbb{N}$ *and a term* u *such that* $(\xi, i \rhd u) \in \Phi$*;*
   - *or of the form* $\mathsf{root}(X) \neq \mathsf{f}$ *with* $\mathsf{f} \in \mathcal{F}_c$*.*
7. *for all* $X \in vars^2(\mathcal{C})$*,* $\mathsf{C}\lfloor X\theta \rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX} \cup \mathcal{X}^2)$ *and for all* $\zeta \in st(X\theta)$*,* $\mathsf{path}(\zeta) \in \mathcal{F}_d^* \cdot \mathcal{AX}$ *implies that there exists* j *and* v *such that* $(\zeta, j \rhd v) \in \Phi$
8. *For all* $(\zeta, i \rhd u) \in \mathsf{NoUse}$*, there exists* $X \in vars^2(\mathcal{C})$ *such that* $\mathsf{C}\lfloor X\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C}) = u$ *and* $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta) < i$
9. *for all* $(\xi, i \rhd u) \in \Phi$*, for all* $\xi' \in st(\xi)$*,* $\mathsf{C}\lfloor \xi' \rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX} \cup \mathcal{X}^2)$ *and if* $\mathsf{path}(\xi') \in \mathcal{F}_d^* \cdot \mathcal{AX}$ *then there exists* j *and* v *such that* $(\xi', j \rhd v) \in \Phi$
10. *for all* $(X, i \overset{?}{\vdash} u) \in D$*,* $X \notin S_2(\mathcal{C})$ *implies that for all* $x \in vars^1(u)$*, there exists* $(Z, j \overset{?}{\vdash} v) \in D$ *such that* $i < j$ *and* $x \in vars^1(v)$*.*

The next lemma indicates that our transformation rules transform a well-formed constraint system into a pair of constraint systems that are also well-formed (Proof in Appendix C.2.2).

Lemma 8.2. *Any rule in Figure 7.1 and Figure 7.2 transforms a normalised well-formed constraint system into a pair of constraint systems that are also well-formed after normalisation. For the rule* DEST*, we assume that its application is not useless.*

8.1.2 Strategy invariants

In the previous subsection, we described some invariants that are independent of the strategy. Typically, they are very useful to prove some general properties on constraint systems. However, they are not sufficient to establish soundness of our rules or even the termination. The strategy we described in Subsection 7.4 has been essentially designed to ensure termination of our algorithm. However, this strategy also allows us to extract some new invariants that will help us to prove soundness.

Among the invariants, some of them are specific to some steps of the strategy, but we first describe the invariants that are satisfied at any step of the strategy.

Invariant 8.1 (InvGeneral)**.** *Let* M *be a matrix of constraint systems. We say that* M *satisfies the invariant* InvGeneral*, if and only if:*

For all constraint system $\mathcal{C}$ *in* M*, if* $\mathcal{C} \neq \bot$ *then for all* $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$*, for all* $(\xi, i \rhd u) \in \Phi(\mathcal{C})$*,*

1. $ax_i \in st(\xi\theta)$*.*
2. *for all* $\xi' \in \Pi_n$ *with* $\mathsf{root}(\xi') \notin \mathcal{F}_c$*, if* $\mathsf{path}(\xi') = \mathsf{path}(\xi\theta)$ *and* $\xi'(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$*, then* $param(\xi') \not\subseteq \{ax_1, \ldots, ax_{i-1}\}$*.*
3. *for all* $X \in vars^2(\mathcal{C})$*, if* $\mathsf{path}(\xi) \in st(\mathsf{C}\lfloor X\mathsf{mgu}(Er) \rfloor_\Phi)$ *then* $(\xi, i \rhd u) \notin \mathsf{NoUse}$*.*
4. *if* $(\xi, i \rhd u) \notin \mathsf{NoUse}$ *then for all* $\xi' \in st(\xi)$*, if there exists* j *and* v *such that* $(\xi', j \rhd v) \in \Phi$ *then* $(\xi', j \rhd v) \notin \mathsf{NoUse}$*.*

For all constraint systems $\mathcal{C}$*,* $\mathcal{C}'$ *in a same column of* M*, if we denote* $\theta = \mathsf{mgu}(Er(\mathcal{C}))$ *and* $\theta' = \mathsf{mgu}(Er(\mathcal{C}'))$*, then*

5. $\forall X \in S_2(\mathcal{C}), \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}')}$
6. $\forall X \in S_2(\mathcal{C}), \forall \mathsf{f} \in \mathcal{F}_c$*,* $Er(\mathcal{C}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ *implies that* $Er(\mathcal{C}') \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$
7. $\forall X \in S_2(\mathcal{C})$*, for all* $(\xi, i \rhd u) \in \Phi(\mathcal{C})$*, for all* $(\xi', i' \rhd u') \in \Phi(\mathcal{C}')$*, if* $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ *then* $Er(\mathcal{C}) \vDash X \overset{?}{\neq} \xi$ *is equivalent to* $Er(\mathcal{C}') \vDash X \overset{?}{\neq} \xi'$*.*

Typically, Properties 1 and 2 ensures that we use in the frame the minimal recipes w.r.t. the parameters to deduce the key of a cypher or the verification key of a signature. These two properties are given by the application of the rule DEST (Step a of Phase 1) and more specifically by the fact that the cycle of steps in Phase 1 is applied by increasing support.

Property 3 and 4 indicates that during Step a, we always prioritised the application of the rule EQ-LEFT-RIGHT over DEST.

The last three properties established similitudes between constraint systems of a same column. We already know that the shape of the constraint systems are the same but the strategy allows us to be even more specific. Indeed, Property 5 indicates that the actions of the attacker are the same in each constraint system of the column (up to the context) but Property 6 and 7 also indicate that the inequalities corresponding the attacker's actions are the same. These three properties are in fact due to the application on the external rules on the matrices of constraint system. The proof of the following lemma can be found in Appendix C.4.3.8

Lemma 8.3. *Let (M, M') be a pair of row matrices of initial constraint systems having the same structure. Let (M_1, M_1') be a pair of matrices of constraint systems obtained by following the strategy on (M, M'). (M_1, M_1') satisfies* $\mathsf{InvGeneral}$.

The next invariants are more specific to the different steps and phase of the strategy. Thus, they depend on a parameter, *i.e.* the support of the rules.

Invariant 8.2 ($\mathsf{InvVarConstraint}(s)$)**.** *Let $\mathcal{C}$ be a constraint system. We say that $\mathcal{C}$ satisfies* $\mathsf{InvVarConstraint}(s)$ *if $\mathcal{C} = \bot$ or*

1. *for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, if $i \leq s$ then $u \in \mathcal{X}^1$ and $X \in S_2(\mathcal{C})$; and*
2. *for all $(X, i \overset{?}{\vdash} x), (Y, j \overset{?}{\vdash} y) \in D(\mathcal{C})$, if $i \leq s$, $j \leq s$ and $X \neq Y$ then $x \neq y$.*

Invariant 8.3 ($\mathsf{InvVarFrame}(s)$)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. We say that $\mathcal{C}$ satisfies* $\mathsf{InvVarFrame}(s)$ *if and only if for all $(\xi, p \rhd v) \in \Phi$, $p \leq s$ implies for all $X \in vars^2(\xi)$, there exists $q < p$ and $u \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$ such that $(X, q \overset{?}{\vdash} u) \in D$.*

Intuitively, $\mathsf{InvVarConstraint}(s)$ corresponds to the purpose of the first phase of the strategy, i.e. modifying the constraint systems such that all right hand term of deducible constraints are distinct variables. Thus, all constraint systems during Phase 2 will satisfy this invariant.

Invariant 8.4 ($\mathsf{InvNoUse}(s)$)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. We say that $\mathcal{C}$ satisfies* $\mathsf{InvNoUse}(s)$ *if and only if for all $(\xi, p \rhd v) \in \Phi$, $p \leq s$ and $v \in \mathcal{X}^1$ implies $(\xi, p \overset{?}{\vdash} v) \in \mathsf{NoUse}$.*

Invariant 8.5 ($\mathsf{InvDest}(s)$)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. We say that $\mathcal{C}$ satisfies* $\mathsf{InvDest}(s)$ *if and only if for all $(\xi, p \rhd v) \in \Phi$, for all $\mathsf{f} \in \mathcal{F}_d$, $(\xi, p \rhd v) \notin \mathsf{NoUse}$ and $p \leq s$ implies:*

- *either there exists $p' \in \mathbb{N}$ such that*
  - *$s \geq p' \geq p$; and*
  - *$(\xi', p' \rhd v') \in \Phi$ for some ξ' such that $\mathsf{path}(\xi') = \mathsf{f} \cdot \mathsf{path}(\xi)$; and*
  - *for every $p \leq k < p'$, for all σ, $\sigma \vDash ND$ implies that $\sigma \vDash \forall \tilde{x}, v \neq u_1 \vee k \overset{?}{\not\vdash} u_2 \vee \ldots \vee k \overset{?}{\not\vdash} u_n$ where $\mathsf{f}(u_1, \ldots, u_n) \to w$ is a fresh rewriting rule with $vars^1(u_1, \ldots, u_n, w) = \tilde{x}$.*
- *or else for every $p \leq k \leq s$, we have that $ND \vDash \forall \tilde{x}, v \neq u_1 \vee k \overset{?}{\not\vdash} u_2 \vee \ldots \vee k \overset{?}{\not\vdash} u_n$ where $\mathsf{f}(u_1, \ldots, u_n) \to w$ is a fresh rewriting rule with $vars^1(u_1, \ldots, u_n, w) = \tilde{x}$.*

Invariant 8.6 ($\mathsf{InvDedsub}$)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. Let $\theta = \mathsf{mgu}(Er)$ and $m = |\Phi|$. We say that $\mathcal{C}$ satisfies* $\mathsf{InvDedsub}$ *if and only if, for all $(\xi, p \rhd v) \in \Phi$, for all $\mathsf{f} \in \mathcal{F}_c$, $(\xi, p \rhd v) \notin \mathsf{NoUse}$ and $p \leq s$ implies:*

— *either there exists* $X_1, \ldots, X_n \in vars^2(\mathcal{C})$ *such that for all* $i \in \{1, \ldots, n\}$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X_i\theta) \leq s$ *and*

$$\mathsf{C}\lfloor \mathsf{f}(X_1, \ldots, X_n)\theta \rfloor_{\Phi} \mathsf{acc}^1(\mathcal{C}) = v$$

— *or else* $ND \vDash \forall \tilde{x}, v \neq \mathsf{f}(x_1, \ldots, x_n) \vee m \overset{?}{\not\vdash} x_1 \vee \ldots \vee m \overset{?}{\not\vdash} x_n$ *where* $\tilde{x} = x_1 \ldots x_n$ *are fresh.*

Typically, $\mathsf{InvDest}(s)$ ensures that no new subterm can be obtained by applying a destructor while $\mathsf{InvDedsub}$ ensures that no new subterm can be obtained by applying a constructor. The invariants $\mathsf{InvDest}(s)$ and $\mathsf{InvNoUse}(s)$ will be satisfied by any constraint system after Step a of Phase 1 with support s.

The next invariant indicates that no rule was applied with support strictly greater than s. Typically, it is satisfied by all constraint systems during Phase 1 with support smaller than s.

Invariant 8.7 ($\mathsf{InvUntouched}(s)$)**.** *Let* $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ *be a constraint system. We say that* $\mathcal{C}$ *satisfies* $\mathsf{InvUntouched}(s)$ *if and only if*

1. *for all* $(\xi, k \rhd u) \in \Phi$*, if* $s < k$ *then* $\xi = ax_k$*; and*
2. *for all* $(X, k \overset{?}{\vdash} u) \in D$*, if* $s < k$ *then* $X \in S_2$ *and* $X \notin vars^2(Er)$*.*

Lastly, we define an invariant that impact on several constraint systems in the matrices.

Invariant 8.8 ($\mathsf{InvMatrix}(s)$)**.** *Let* M *be a matrix of constraint systems. We say that* M *satisfies* $\mathsf{InvMatrix}(s)$ *if and only if for all* $\mathcal{C}$*,* $\mathcal{C}'$ *two constraint systems in the same column of* M*,*

— $\{\mathsf{path}(\xi), i \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}) \wedge i \leq s\} = \{\mathsf{path}(\xi), i \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}') \wedge i \leq s\}$

— $\{\mathsf{path}(\xi), i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}) \wedge i \leq s\} = \{\mathsf{path}(\xi), i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}') \wedge i \leq s\}$

8.2 Soundness and completeness

This section is dedicated to the proofs the soundness and completeness of our transformation rules.

8.2.1 Preliminaries

In our main proof of completeness and soundness, we usually assume an existing solution in a well-formed constraint system and then we transform this solution such that it become a solution of an another constraint system. In most cases, the transformation consists of replacing a recipe by an other one which deduces the same message. The main issue of this replacement is that the new recipe has to satisfy several properties such that the conformity to the frame, its belonging to Π_n, ...

Example 8.1. *Let a constraint system with a the following frame :*

$$\{ax_1, 1 \rhd \mathsf{senc}(a, b); ax_2, 2 \rhd \mathsf{senc}(b, a); ax_3, 3 \rhd a\}$$

A possible and natural execution of the rules would be to guess that the messages $\mathsf{senc}(a, b)$ *and* $\mathsf{senc}(b, a)$ *are reducible and so by application of the rule* DEST*, we would have a constraint system such that:*

— $\Phi = \{ax_1, 1 \rhd \mathsf{senc}(a, b);\ ax_2, 2 \rhd \mathsf{senc}(b, a);\ ax_3, 3 \rhd a;\ \mathsf{sdec}(ax_1, X) \rhd a;\ \mathsf{sdec}(ax_2, Y) \rhd b\}$

— $D = \{X, 3 \overset{?}{\vdash} b\ ; Y, 3 \overset{?}{\vdash} a\}$

Thus one possible solution for this constraint system would be θ with $X\theta = \mathsf{sdec}(ax_2, ax_3)$ and $Y\theta = ax_3$. We can see that θ belongs to Π_n and also conforms to the frame Φ.

Since the two recipes ax_3 and $\mathsf{sdec}(ax_2, Y)$ both deduce the same message, we could replace any instance of ax_3 by $\mathsf{sdec}(ax_1, X)$ and then forbid the use of the recipe ax_3 (equivalent to adding the frame element $(ax_3, 3 \rhd a)$ into the set NoUse).

Thus to ensure the soundness of this transformation, we need to ensure that we can transform θ in θ' such that θ' is a solution of the constraint system and such that it satisfies the belonging to Π_n and the conformity to the frame. But on this example, the only way to deduce a (for the constraint $Y, 3 \overset{?}{\vdash} a$) without using ax_3 is to use the recipe $\mathsf{sdec}(ax_1, X)$ and the only way to deduce b (for the constraint $X, 3 \overset{?}{\vdash} b$) is to use $\mathsf{sdec}(ax_2, Y)$ which produces a loop. Therefore, on this example, the replacement of the recipe ax_3 by $\mathsf{sdec}(ax_1, X)$, that deduces the same message, does not lead to a solution.

To formalise this, we introduce the following order.

Definition 8.3 (relation $<_\theta$)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system, and θ be a mapping from $vars^2(\mathcal{C})$ to ground recipes such that for all $(X, i \rhd u) \in D$, we have that $param(X\theta) \subseteq \{ax_1, \ldots, ax_i\}$.*

We define a relation on $vars^2(D)$, denoted $<_\theta$, as the smallest relation that is closed by transitivity and such that: $X <_\theta Y$ if $X \in vars^2(\mathsf{C}\lfloor Y\theta \rfloor \mathsf{acc}^2(\mathcal{C}))$ and $X \neq Y$.

Intuitively, $X <_\theta Y$ represents the fact that in the solution θ, $X\theta$ is used in $Y\theta$. Thus, if you replace $X\theta$ by an other recipe, the recipe $Y\theta$ will also need to be changed accordingly in order to conform to the frame.

Example 8.2. *Going back to our previous example, we have that $Y <_\theta X$ since $\mathsf{C}\lfloor Y\theta \rfloor \mathsf{acc}^2(\mathcal{C})) = \mathsf{sdec}(ax_2, Y)$.*

We stated at the beginning of this subsection that the replacement has to preserve the belonging to Π_n. But a simple example with the application of the rule Eq-left-right shows that it is generally not true.

Example 8.3. *Let $\mathcal{C}$ be a constraint system with the following frame:*

$$\Phi = \{ax_1, 1 \rhd a \; ; ax_2, 2 \rhd \mathsf{senc}(a, a) \; ; ax_3, 3 \rhd a\}$$

and the following set of deducibility constraints:

$$D = \{X, 1 \overset{?}{\vdash} \mathsf{senc}(a, a) \; ; Y, 3 \overset{?}{\vdash} a\}$$

One possible solution for this constraint system would be θ with $X\theta = \mathsf{senc}(ax_1, ax_1)$ and $Y\theta = \mathsf{sdec}(ax_2, ax_3)$. We can see that θ belongs to Π_n and also conforms to the frame Φ. By applying the rule Eq-left-right *on $(ax_2, 2 \rhd \mathsf{senc}(a, a))$ and $(X, 1 \overset{?}{\vdash} \mathsf{senc}(a, a))$, the frame element $(ax_2, 2 \rhd \mathsf{senc}(a, a))$ will be added in the set NoUse and thus, we now have to replace each instance of ax_2 with $X\theta$. But in such a case, $Y\theta$ will become the recipe $\mathsf{sdec}(\mathsf{senc}(ax_1, ax_1), ax_3)$ which does not belong to Π_n. Thus, instead of just replacing ax_2 by $\mathsf{senc}(ax_1, ax_2)$, we will replace $Y\theta$ with ax_1.*

8.2.2 Core lemmas

Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system. We denote by $\overline{\mathcal{C}}$ the constraint system $(S_1; S_2; \Phi; D; Eq; Er; \emptyset; \mathsf{NoUse})$, *i.e.* the constraint system obtained from $\mathcal{C}$ by removing the non-deducible constraints. This notation is extended as expected to matrices of constraint systems. We first show the soundness of the normalisation of constraint system (Proof in Appendix C.5)

Lemma 8.4. *Let $\mathcal{C}$ be a constraint system obtained by following the strategy.* $\mathsf{Sol}(\mathcal{C}) = \mathsf{Sol}(\mathcal{C}{\downarrow})$.

As explained in Section 7.2, our algorithm transforms a pair of matrices of constraint systems into one or two pairs of matrices of constraint systems. The main idea behind the soundness and completeness is to locally prove that our transformation preserves the symbolic equivalence between matrices of constraint systems. Since we have several rules, we will need to prove the local preservation of the symbolic equivalence for each rule.

Lemma 8.5 (completeness)**.** *Let $\mathcal{C}$ be a normalised constraint system obtained by following the strategy and* $\textsc{Rule}(\tilde{p})$ *be a transformation rule applicable on $\mathcal{C}$. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the two resulting constraint systems obtained by applying* $\textsc{Rule}(\tilde{p})$ *on $\mathcal{C}$. We denote by Φ, Φ_1 and Φ_2 the respective frames of $\mathcal{C}$, $\mathcal{C}_1$ and $\mathcal{C}_2$ and we denote by S_1 the set of free variable of $\mathcal{C}$.*

For all $i \in \{1,2\}$, for all $(\sigma_i, \theta_i) \in \mathsf{Sol}(\mathcal{C}_i)$, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and $\mathsf{Init}(\Phi)\sigma = \mathsf{Init}(\Phi_i)\sigma_i$ where $\sigma = \sigma_i|_{vars^1(\mathcal{C})}$ and $\theta = \theta_i|_{vars^2(\mathcal{C})}$

Variation: For all $i \in \{1,2\}$, for all $(\sigma_i, \theta_i) \in \mathsf{Sol}(\overline{\mathcal{C}_i})$, $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$ and $\mathsf{Init}(\Phi)\sigma = \mathsf{Init}(\Phi_i)\sigma_i$ where $\sigma = \sigma_i|_{vars^1(\overline{\mathcal{C}})}$ and $\theta = \theta_i|_{vars^2(\overline{\mathcal{C}})}$.

Lemma 8.6 (soundness)**.** *Let $\mathcal{C}$ be a normalised constraint system obtained by following the strategy and* $\textsc{Rule}(\tilde{p})$ *be a transformation rule applicable on $\mathcal{C}$. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the two resulting constraint systems obtained by applying* $\textsc{Rule}(\tilde{p})$ *on $\mathcal{C}$. We denote by Φ, Φ_1 and Φ_2 the respective frames of $\mathcal{C}$, $\mathcal{C}_1$ and $\mathcal{C}_2$ and we denote by S_1 the set of free variables of $\mathcal{C}$.*

Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. There exist σ', θ', and $i_0 \in \{1,2\}$ such that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}_{i_0})$, $\sigma = \sigma'|_{vars^1(\mathcal{C})}$ and $\mathsf{Init}(\Phi)\sigma = \mathsf{Init}(\Phi_{i_0})\sigma'$.

The proofs of Lemma 8.5 and 8.6 can be respectively found in Appendix C.3 and C.5. With the two previous lemmas (Lemma 8.5 and Lemma 8.6), we can see that our rules preserve the set of first-order solution of a constraint system. On the other hand, Lemma 8.6 indicates that it is possible that some second-order solutions are not preserves, for example the solution that use several recipe to deduce the same key. Even if the non-preservation of the whole set of second-order solutions could be surprising at first sight, the idea is to preserve enough second-order solutions to be able to establish symbolic equivalence (Lemma 8.7).

Lemma 8.7. *Let $(\mathcal{C}, \mathcal{C}')$ be a pair of normalised constraint systems having the same structure and obtained by following the strategy. We denote by Φ and Φ' their associated frame. We denote by S_1, S_1' their associated set of free variables. Let* $\textsc{Rule}(\tilde{p})$ *be a transformation rule applicable on $(\mathcal{C}, \mathcal{C}')$. Let $(\mathcal{C}_1, \mathcal{C}_1')$ and $(\mathcal{C}_2, \mathcal{C}_2')$ the two resulting pairs of constraint systems obtained by applying* $\textsc{Rule}(\tilde{p})$ *on $(\mathcal{C}, \mathcal{C}')$, and we denote by Φ_1, Φ_1', Φ_2, and Φ_2' their associated frame.*

Let σ, θ and σ' be three substitutions such that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$, and $\Phi\sigma \sim \Phi'\sigma'$. For all substitution θ',

1. *$(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$ if, and only, if $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$.*
2. *Let $i \in \{1,2\}$, and σ_i be a substitution such that $\sigma|_{S_1} = \sigma_i|_{S_1}$ and $(\sigma_i, \theta') \in \mathsf{Sol}(\mathcal{C}_i)$. Then, $(\sigma_i', \theta') \in \mathsf{Sol}(\mathcal{C}_i')$ for some substitution σ_i' such that $\sigma'|_{S_1'} = \sigma_i'|_{S_1'}$. Moreover, we have that $\mathsf{Init}(\Phi_i)\sigma_i = \mathsf{Init}(\Phi)\sigma$ and $\mathsf{Init}(\Phi_i')\sigma_i' = \mathsf{Init}(\Phi')\sigma'$.*

The proof of Lemma 8.7 can be found in Appendix C.5.

8.2.3 Application to matrices of constraint systems

Using the lemmas 8.5, 8.6 and 8.7, we now establish completeness and soundness for pairs of linear matrices and for pairs of matrices.

Proposition 8.1. *Let V, V' be two row matrices of constraint systems obtained by following the strategy. Let* $\textsc{Rule}(\tilde{p})$ *be a transformation rule applicable on (V, V'). Let (W_1, W_1') and (W_2, W_2') be the two resulting pairs of row matrices of constraints systems obtained by the application of* $\textsc{Rule}(\tilde{p})$.

$$W_1 \approx_s W_1' \text{ and } W_2 \approx_s W_2' \text{ is equivalent to } V \approx_s V'$$

Proof. We prove the two directions of the equivalence separately. We assume w.l.o.g. that V (resp. V') is a row matrix of size n (resp. n'). Let $V = [\mathcal{C}_1, \ldots, \mathcal{C}_n]$ and $V' = [\mathcal{C}'_1 \ldots, \mathcal{C}'_{n'}]$. We know that W_1 and W_2 (resp. W'_1 and W'_2) are row matrices of size n (resp. n'). Let $W_i = [\mathcal{C}^i_1, \ldots, \mathcal{C}^i_n]$ and $W'_i = [\mathcal{C}'^i_1, \ldots, \mathcal{C}'^i_{n'}]$ for $i = 1, 2$.

Let $1 \leq j \leq n$ and $1 \leq k \leq n'$. We denote by Φ_j the frame associated to $\mathcal{C}_j$ and by Φ'_k the frame associated to $\mathcal{C}'_k$. Let $i \in \{1, 2\}$. Similarly, we denote by Φ^i_j the frame associated to $\mathcal{C}^i_j$ and by Φ'^i_k the frame associated to $\mathcal{C}'^i_k$.

Right implication: Let $1 \leq j \leq n$ and let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_j)$. By Lemma 8.6, we know that there exists θ', $i \in \{1, 2\}$ and σ_i such that $(\sigma_i, \theta') \in \mathsf{Sol}(\mathcal{C}^i_j)$ and $\sigma|_{S_1(\mathcal{C}_j)} = \sigma_i|_{S_1(\mathcal{C}_j)}$. By hypothesis, we have that $W_i \approx_s W'_i$. Hence, there exist $1 \leq k \leq n'$ and a substitution σ'_i such that $(\sigma'_i, \theta') \in \mathsf{Sol}(\mathcal{C}'^i_k)$ and $\Phi^i_j \sigma_i \sim \Phi'^i_k \sigma'_i$. Thanks to Lemma 8.5, we deduce that:

$$(\sigma_i|_{vars^1(\mathcal{C}_j)}, \theta'|_{vars^2(\mathcal{C}_j)}) \in \mathsf{Sol}(\mathcal{C}_j) \text{ and } (\sigma'_i|_{vars^1(\mathcal{C}'_k)}, \theta'|_{vars^2(\mathcal{C}'_k)}) \in \mathsf{Sol}(\mathcal{C}'_k).$$

with $\mathsf{Init}(\Phi^i_j)\sigma_i = \mathsf{Init}(\Phi_j)\sigma_i|_{vars^1(\mathcal{C}_j)}$ and $\mathsf{Init}(\Phi'^i_k)\sigma'_i = \mathsf{Init}(\Phi'_k)\sigma'_i|_{vars^1(\mathcal{C}'_k)}$. Note that $\theta'|_{vars^2(\mathcal{C}'_k)} = \theta'|_{vars^2(\mathcal{C}_j)}$ since $\mathcal{C}'_k$ and $\mathcal{C}_j$ have the same structure. Moreover, by hypothesis, we have that $\sigma = \sigma_i|_{vars^1(\mathcal{C}_j)}$. Thus, we can apply Lemma 8.7 on $\mathcal{C}_j$ and $\mathcal{C}'_k$. Since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_j)$, we deduce that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}'_k)$ where $\sigma' = \sigma'_i|_{vars^1(\mathcal{C}'_k)}$. Lastly, since $\Phi^i_j \sigma_i \sim \Phi'^i_k \sigma'_i$, $\mathsf{Init}(\Phi^i_j)\sigma_i = \mathsf{Init}(\Phi_j)\sigma$ and $\mathsf{Init}(\Phi'^i_k)\sigma'_i = \mathsf{Init}(\Phi'_k)\sigma'$, we easily deduce that $\Phi_j \sigma \sim \Phi'_k \sigma'$. The other implication can be done in a similar way.

Left implication: Let $1 \leq j \leq n$, $i \in \{1, 2\}$ and let $(\sigma_i, \theta') \in \mathsf{Sol}(\mathcal{C}^i_j)$. By Lemma 8.5, we know that there exists σ and θ such that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_j)$ with $\sigma = \sigma_i|_{vars^1(\mathcal{C}_j)}$ and $\theta = \theta'|_{vars^2(\mathcal{C}_j)}$. By hypothesis, we have that $V \approx_s V'$. Hence, there exists $1 \leq k \leq n'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}'_k)$ and $\Phi_j \sigma \sim \Phi'_k \sigma'$. Thus, we can apply Lemma 8.7 (second item) on $\mathcal{C}_j$ and $\mathcal{C}'_k$. Since $(\sigma_i, \theta') \in \mathsf{Sol}(\mathcal{C}^i_j)$, we deduce that $(\sigma'_i, \theta') \in \mathsf{Sol}(\mathcal{C}'^i_k)$ for some substitution σ'_i such that $\sigma' = \sigma'_i|_{vars^1(\mathcal{C}'_k)}$. Moreover, we have that $\mathsf{Init}(\Phi^i_j)\sigma_i = \mathsf{Init}(\Phi_j)\sigma$ and $\mathsf{Init}(\Phi'^i_k)\sigma'_i = \mathsf{Init}(\Phi'_k)\sigma'$. Since $\Phi_j \sigma \sim \Phi'_k \sigma'$, we easily deduce that $\Phi^i_j \sigma_i \sim \Phi'^i_k \sigma'_i$. The other implication can be done in a similar way. □

Theorem 8.1 (soundness and completeness for internal rules)**.** *Let M_1, M'_1 be two matrices of constraint systems obtained by following the strategy. Let* $\textsc{Rule}(\tilde{p})$ *be an internal transformation rule applicable on (M_1, M'_1) on the i^{th}. Let (M_2, M'_2) be the resulting pair of matrices of constraint systems obtained by the application of* $\textsc{Rule}(\tilde{p})$*. We have that:*

$$M_2 \approx_s M'_2 \text{ is equivalent to } M_1 \approx_s M'_1$$

Proof. Since M_1 and M'_1 have the same structure, we know that they have the same number of lines, say m. Let $M_1 = [V_1, \ldots, V_m]$ and $M'_1 = [V'_1, \ldots, V'_m]$. Let (W_1, W'_1) and (W_2, W'_2) be the two resulting pairs of row matrices of constraint systems obtained by applying $\textsc{Rule}(\tilde{p})$ on (V_i, V'_i). Hence $M_2 = [V_1, \ldots, V_{i-1}, W_1, W_2, V_{i+1}, \ldots, V_m]$, and $M'_2 = [V'_1, \ldots, V'_{i-1}, W'_1, W'_2, V'_{i+1}, \ldots, V'_m]$.

By Definition 7.12 of the symbolic equivalence of matrices of constraint systems, $M_1 \approx_s M'_1$ is equivalent to $V_j \approx_s V'_j$ for every $j \in \{1, \ldots, m\}$. Thanks to Proposition 8.1, we easily deduce that $V_i \approx_s V'_i$ is equivalent to $W_1 \approx_s W'_1$ and $W_2 \approx_s W'_2$. This allows us to conclude. □

Theorem 8.2 (soundness and completeness for external rules)**.** *Let M, M' be two matrices of well-formed constraint systems that have the same structure. Let* $\textsc{Rule}(\tilde{p})$ *be an external transformation rule applicable on (M, M'). Let (M_1, M'_1) and (M_2, M'_2) be the two resulting pairs of matrices of constraint systems obtained by the application of* $\textsc{Rule}(\tilde{p})$*. We have that:*

$$M_1 \approx_s M'_1 \text{ and } M_2 \approx_s M'_2 \text{ is equivalent to } M \approx_s M'$$

Proof. Since M and M' have the same structure, we know that they have the same number of lines, say m. Let $M = [V_1, \ldots, V_m]$ and $M' = [V'_1, \ldots, V'_m]$. When $\textsc{Rule}(\tilde{p})$ is applicable on (V_i, V'_i), let $(W_{i,1}, W'_{i,1})$ and $(W_{i,2}, W'_{i,2})$ be the two resulting pairs of row matrices of constraint system. Otherwise, let $(W_{i,1}, W'_{i,1}) = (\bot, \bot)$ and $(W_{i,2}, W'_{i,2}) = (V_i, V'_i)$. We have that:

— $M_1 = [W_{1,1}, \ldots, W_{m,1}]$, $M'_1 = [W'_{1,1}, \ldots, W'_{m,1}]$, and
— $M_2 = [W_{1,2}, \ldots, W_{m,2}]$, $M'_2 = [W'_{1,2}, \ldots, W'_{m,2}]$.

By Definition 7.12 of the symbolic equivalence of matrices of constraint systems, $M \approx_s M'$ is equivalent to $V_i \approx_s V'_i$ for every $i \in \{1, \ldots, m\}$. Relying on Proposition 8.1 when the rule is effectively applied on (V_i, V'_i) (when the rule is not applicable, the result trivially holds), we deduce that for every i, $V_i \approx_s V'_i$ is equivalent to $W_{i,1} \approx_s W'_{i,1}$ and $W_{i,2} \approx_s W'_{i,2}$ which is equivalent to $M_1 \approx_s M'_1$ and $M_2 \approx_s M'_2$. This allows us to conclude. □

8.3 Leaves

In Section 8.2 we proved that all of our rules preserve the symbolic equivalence of matrices of constraint systems. Thus, since our strategy terminates (see Section 8.4), we know that for any initial pair of row matrices of constraint systems (M_0, M'_0), we have that $M_0 \approx_s M'_0$ if, and only if, all leaves of a tree, whose root is labeled with (M_0, M'_0), are symbolically equivalent. Thus to prove Theorem 7.2, we will have to prove that the predicate $\mathsf{LeafTest}$ is equivalent to the symbolic equivalence of a leaf.

The proofs of all the lemmas in this section can be found in Appendix C.6.

8.3.1 Shape of the leaves

The purpose of our rules is to transform the constraint systems in the matrices into *simpler constraint systems*. The idea of *simple constraint systems* is not new, and was for example use in [CLCZ10] for their reachability algorithm. They introduce the notion of *solved constraint system* where a constraint system was in solved form if each right hand term of the constraints were a variable. In their algorithm, it was sufficient to prove the existence of at least one solution in a constraint system. But in our case, the right hand term variable condition is not sufficient to prove the symbolic equivalence between matrices. Thus, we introduce a new definition of *solved constraint systems*.

Definition 8.4 (solved constraint system)**.** *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system. We say that $\mathcal{C}$ is a solved constraint system (or is in solved form) if $\mathcal{C}$ satisfies* $\mathsf{InvVarConstraint}(\infty)$, $\mathsf{InvNoUse}(\infty)$, $\mathsf{InvVarFrame}(\infty)$, $\mathsf{InvDest}(\infty)$, $\mathsf{InvDedsub}$ *and:*

1. *Eq is a formula of the form $\bigwedge_k u_k \stackrel{?}{=} v_k \bigwedge_i [\bigvee_j x_{i,j} \stackrel{?}{\neq} w_{i,j}]$ where $w_{i,j} \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^1)$, $x_{i,j} \in \mathcal{X}^1$, $u_k, v_k \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$.*
2. *for all $X \in vars^2(D)$, for all $\mathsf{f} \in \mathcal{F}_c$, for all ξ recipe of Φ, $Er \not\vdash X \stackrel{?}{\neq} \xi$ and $Er \not\vdash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$.*

We say that a constraint system $\bot$ is always in solved formed.

The definition of a solved constraint relies mainly on the invariant that we used for the termination proof. The last property ensure that no inequations on message contains any name. Thus we will be able to easily match those inequations with inequations on other constraint system. It comes directly from the fact that at the end of the strategy, all inequations in the constraint system have an entry in their association table.

Typically, all those properties correspond to the usual properties needed for deciding the symbolic equivalence :

— existence of a solution
— matching the solutions
— deciding the static equivalence

As we said previously, having the constraint systems in solved form on the leaves will help us deciding the symbolic equivalence. But Definition 8.4 only focuses on one constraint system

and does not give any information on the link between constraint systems of the same pair of matrices on a leaf. We already know from Lemma 8.1 that all constraint systems on the matrices have the same shape and furthermore that all constraint systems on the same line have the same structure. But it is not sufficient to show that the predicate LeafTest is equivalent to the symbolic equivalence. In fact, we need to simplify the matrices the same way we simplified the constraint systems. Thus, we define a notion of *solved matrix of constraint systems* which will describe all the links between constraint systems in the matrices.

Definition 8.5 (solved matrix of constraint systems)**.** *Let (M, M') be a pair of matrices of constraint systems. We say that (M, M') is in* solved-form *if (M, M') satisfies* InvMatrix(∞) *and* InvGeneral*, all constraint systems in (M, M') are in solved-form, and*

First order term properties on same column : *for all $\mathcal{C}, \mathcal{C}'$ constraint system on the same column, there exists a variable renaming $\rho : \mathcal{X}^1 \setminus S_1(\mathcal{C}) \to \mathcal{X}^1 \setminus S_1(\mathcal{C}')$ such that:*

1. $\mathsf{mgu}(Eq(\mathcal{C}))_{|S_1(\mathcal{C})}\rho = \mathsf{mgu}(Eq(\mathcal{C}'))_{|S_1(\mathcal{C}')}$*, and* $D(\mathcal{C})\rho = D(\mathcal{C}')$*;*
2. $\{(u\rho, u') \mid (\xi, i \rhd u) \in \Phi \wedge (\xi', i' \rhd u') \in \Phi' \wedge \mathsf{path}(\xi) = \mathsf{path}(\xi')\}$ *is include in* $\{(u, u) \mid u \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)\}$*;*

First order term property : *for all $\mathcal{C}, \mathcal{C}'$ constraint system in (M, M'), there exists a variable renaming $\rho : \mathcal{X}^1 \to \mathcal{X}^1$ such that:*

3. *$Eq(M_{i,j})\rho$ restricted to inequation and $D(M_{i,j})\rho$ are equal to $Eq(M_{i,j'})$ restricted to inequation and $D(M_{i,j'})$.*

The first properties of Definition 8.5 focus on the messages inside the constraint systems. Intuitively, Property 1 and 2 will help us to prove that all the constraint systems have the same set of first order solutions. To understand why this property holds, we need to come back to the creation of new lines in a matrix of constraint systems, and so to the application of internal rules. In Subsection 7.3.2, we describe the application of internal rules as a way to keep the result of the guesses on static equivalence inside a single matrix. Thus, it is natural that the first order solutions of different constraint systems in a same column are the same.

Property 3 indicates that there exists a matching on the inequations of message for all constraint systems in a same line. The purpose of this property is for us to prove that any substitution satisfying the inequations in one constraint systems will be match by an other first order substitution that satisfies the inequations in an other constraint systems.

Lemma 8.8. *Let (M, M') be a pair of matrix obtained at the end strategy. (M, M') is in solved form.*

8.3.2 Proving the symbolic equivalence

This last subsection focuses on the proof of that a pair of matrices of constraint system in solved form are symbolically equivalent if, and only if, they satisfy LeafTest.

First of all, its is not trivial to check if a constraint system in solved form has a solution. Intuitively, this is due to the presence of non deducible constraint in our constraint system, even in solved form. On the other hand, if the non-deducible constraints were to be removed, one can easily show that any solved constraint system would have at least one solution. However, as explain in Section 7.4, Step e of Phase 1 of the strategy were specifically designed to solve the non-deducibility constraints. Indeed, in step e, we only kept the constraint systems whose non-deducible constraints could be satisfied. Hence, on a constraint system on a leaf, we show that any solution that does not consider the non-deducible constraints still satisfies them.

Lemma 8.9. *Let (M, M') be a pair of matrix obtained at the end strategy. For all constraint system $\mathcal{C}$ in M or M',* $\mathsf{Sol}(\mathcal{C}) = \mathsf{Sol}(\overline{\mathcal{C}})$.

Using this lemma, we can now show that any constraint system on a leaf that is different from $\bot$ has at least one solution.

Lemma 8.10. *Let (M, M') be a pair of matrix obtained at the end strategy. Let $\mathcal{C}$ be a constraint system in M or M' different from $\bot$. There exists $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$.*

In this section, we are interested in the symbolic equivalence of matrices of constraint systems. But in fact, when the matrices are in solved one, we can show that any constraint system in the same line of the matrices are symbolically equivalent.

Lemma 8.11. *Let (M, M') be a pair of matrix obtained at the end strategy. Let $\mathcal{C}, \mathcal{C}'$ be two constraint system in the same line in (M, M') ($\mathcal{C}$ and $\mathcal{C}'$ may be contained in the same matrix). If $\mathcal{C} \neq \bot$ and $\mathcal{C}' \neq \bot$ then $\mathcal{C} \approx_s \mathcal{C}'$.*

Using the previous lemma, we can finally prove that the pair of matrices on the leaves are symbolically equivalence if and only if they satisfy the final test.

Theorem 8.3. *Let (M, M') be a pair of matrix obtained at the end strategy. $M \approx_s M'$ if, and only if, $\mathsf{LeafTest}(M, M') = \mathsf{true}$.*

Proof. Assume that M (resp. M') has n lines and m (resp. m') columns. We prove the two implications separately.

Left implication ($\Leftarrow$): Assume that $\mathsf{LeafTest}(M, M') = \mathsf{true}$. Let $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. Let $(\sigma, \theta) \in \mathsf{Sol}(M_{i,j})$. Since $M_{i,j}$ contain a solution, we deduce that $M_{i,j} \neq \bot$. Hence, thanks to $\mathsf{LeafTest}(M, M') = \mathsf{true}$, there exists $j' \in \{1, \ldots, m'\}$ such that $M'_{i,j'} \neq \bot$. By Lemma 8.11, we deduce that $M_{i,j} \approx_s M'_{i,j'}$. Thus, $(\sigma, \theta) \in M_{i,j}$ implies that there exists σ' such that $(\sigma', \theta) \in \mathsf{Sol}(M'_{i,j'})$ and $\Phi(M_{i,j})\sigma \sim \Phi(M'_{i,j'})\sigma'$. The other side of the equivalence is proved symmetrically. Therefore we deduce that $M \approx_s M'$.

Right implication ($\Rightarrow$): Assume that $M \approx_s M'$. We have to show that $\mathsf{LeafTest}(M, M') = \mathsf{true}$. Let $i \in \{1, \ldots, n\}$ and let $j \in \{1, \ldots, m\}$. Assume that $M_{i,j} \neq \bot$. Thanks to Lemma 8.10, we deduce that there exists $(\sigma, \theta) \in \mathsf{Sol}(M_{i,j})$. But $M \approx_s M'$, thus there exists $j' \in \{1, \ldots, m'\}$ and σ' such that $(\sigma', \theta) \in \mathsf{Sol}(M'_{i,j'})$ and $\Phi(M_{i,j})\sigma \sim \Phi(M'_{i,j'})\sigma'$. Since $M'_{i,j'}$ contain at least a solution, we deduce that $M'_{i,j'} \neq \bot$. The other side of the equivalence is proved symmetrically. Therefore, we deduce that $\mathsf{LeafTest}(M, M') = \mathsf{true}$. □

8.4 Termination

In this section, we show how we prove that the strategy explained in Subsection 7.4 terminates. The proofs of all the lemmas in this section can be found in Appendix C.7.

8.4.1 Termination of all steps of Phase 1 of the strategy

In this subsection, we describe all the measures that we use to show the termination of each step of phase 1 of the strategy.

8.4.1.1 Termination of Phase 1, Step a

Step a of the strategy only consists of applying the rules Dest and Eq-left-right. We first define a lexical measure on a constraint system $\mathcal{C}$, denote $\mu_{1.a}(\mathcal{C})$, as follow: $\mu_{1.a}(\bot) = (0, \emptyset, 0, 0)$, otherwise $\mu_{1.a}(\mathcal{C})$ is the following quadruple.

1. $|vars^1(D(\mathcal{C}))|$
2. the multiset $\{\{n \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}) \wedge n = |\{v \in st(u) \mid \mathsf{root}(v) \in \{\mathsf{aenc}, \mathsf{senc}, \langle\rangle, \mathsf{sign}\}\}| \wedge \textsc{Dest}(\xi, \ell \to r, s) \text{ not useless}\}\}$
3. $|\Phi(\mathcal{C})| - |\mathsf{NoUse}(\mathcal{C})|$
4. $|\{(X, \xi) \mid (X, i \rhd u) \in D(\mathcal{C}) \wedge (\xi, j \rhd v) \in \Phi(\mathcal{C}) \setminus \mathsf{NoUse}(\mathcal{C}) \wedge Eq(\mathcal{C}) \not\vdash u \stackrel{?}{\neq} v\}|$

Moreover, we define a measure on pair of matrices of constraint system, denoted $\mu^m_{1.a}(M, M')$, such that such that $\mu^m_{1.a}(M, M') = \{\{\mu_{1.a}(\mathcal{C}) | \mathcal{C} \in M \text{ or } \mathcal{C} \in M'\}\}$.

Intuitively, the first item represents the number of first order variable in the deducible constraint of $\mathcal{C}$. An application of Dest during step a always preserves this number while an application of Eq-left-right may decrease it strictly. The second item of this measure represents the subterms of the frame that may be deducible thanks to the applicable of the rule Dest. Typically when the application of $\text{Dest}(\xi, \ell \to r, s)$ is useless, then no new subterm u in $(\xi, \rhd u)$ can be deduces thanks to Dest. The third item of the measure represents the number of frame elements that are not considered as useless. Lastly, the fourth item represents the possible parameters for which an application of Eq-left-right is not useless.

Lemma 8.12. *Let (M_0, M'_0) be a pair of matrices obtained during Step a of Phase 1 of the strategy. Let $R(\tilde{p})$ one of the possible instances of* Dest *and* Eq-left-right *applicable on (M_0, M'_0). If we denote (M_1, M'_1) the pair of matrices of constraint systems obtained by applying $R(\tilde{p})$ on (M_0, M'_0), then $\mu^m_{1.a}(M_1, M'_1) < \mu^m_{1.a}(M_0, M'_0)$.*

8.4.1.2 Termination of Phase 1, Step b

As stated in Subsection 7.4, after step a, the strategy alternates between step b and step c. Thus, we have to show first that both steps terminates independently and then we have to show that the alternation of both steps terminates too. We remind the steps b and c have two parameter, i.e. the support of the rules s and also the current column k of the matrices.

We define a lexical measure on constraint system. Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system, we define a lexical measure on $\mathcal{C}$, denoted $\mu_{1.b}(\mathcal{C})$, which is composed of :

1. The multiset $\{\{(X, \mathsf{f}) \mid Er \not\vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}, (X, i \stackrel{?}{\vdash} u) \in D, u \in \mathcal{X}^1, \mathsf{f} \in \mathcal{F}_c \text{ and there exists } \mathsf{g} \in \mathcal{F}_c \text{ such that } Er \vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{g}\}\}$.
2. The number of variables in the deducible constraints, i.e. $|vars^1(D)|$.
3. The number of frame elements on which Ded-st is not useless
4. The number of pair of frame elements on which Eq-left-left is not useless
5. The number of function symbols and names in the right term of deducible constraints.
6. The number of (X, f) such that $\mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$ not in Er, $(X, i \stackrel{?}{\vdash} u) \in D$ and $\mathsf{f} \in \mathcal{F}_c$.
7. The number of (X, ξ) such that $(X, i \stackrel{?}{\vdash} u) \in D$, $(\xi, j \stackrel{?}{\vdash} v) \in \Phi$, $j \leq i$ and $X \neq \xi$ is not in Er.
8. The number of deducible constraints, i.e. $|D|$.

By definition, if $\mathcal{C} = \bot$, we say that $\mu_{1.b}(\mathcal{C}) = (\emptyset, 0, 0, 0, 0, 0, 0, 0)$.

Lemma 8.13. *Let $\mathcal{C}$ be a well-formed constraint system satisfying the invariant* $\mathsf{InvVarFrame}(s)$. *Let $R(\tilde{p})$ be any instance of any rules except* Dest *and* Eq-left-right *with support inferior to s. Assume that $R(\tilde{p})$ is strongly applicable on $\mathcal{C}$. Denote $\mathcal{C}_1$ and $\mathcal{C}_2$ the two constraint systems obtained by application of $R(\tilde{p})$ on $\mathcal{C}$. The following property holds :*

$$\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C}) \quad \wedge \quad \mu_{1.b}(\mathcal{C}_2) < \mu_{1.b}(\mathcal{C})$$

We recall that during Step b with parameter s and k where k is the index of the column of (M, M'), an internal rule is applied on the i^{th} line of the matrices (M, M') only if this rule is strongly applicable on constraint system on the i^{th} line and k^{th} column of (M, M'). Hence, for a pair of matrices of constraint system, we define a measure from $\mu_{1.b}(\cdot)$, denoted $\mu^k_{1.b}()$, which is the following multiset:

$$\mu^k_{1.b}(M, M') = \{\{\mu_{1.b}(\mathcal{C}) \mid i \in \mathbb{N} \text{ and } \mathcal{C} \text{ is on the } i^{\text{th}} \text{ line and } k^{\text{th}} \text{ column of } (M, M')\}\}$$

Hence thanks to Lemma 8.13, we can deduce the following corollary.

Corollary 8.1. *Let (M_0, M_0') be a matrix of constraint obtain during Step b of Phase 1 of the strategy with parameter s and k. Let $R(\tilde{p})$ be one of the possible instance of an internal rule of Step b with parameter s and k. If we denote (M_1, M_1') the pair of matrices of constraint systems obtained by applying $R(\tilde{p})$ on (M_0, M_0'), then $\mu^k_{1.b}(M_1, M_1') < \mu^k_{1.b}(M_0, M_0')$*

8.4.1.3 Termination of Phase 1, Step c

As explained in Section 7.4, the purpose of Step c is to definitely remove the internal deducible constraints in the matrices of constraint systems. Whereas the rules applied in Step b are internal, the rules applied in step c are mainly external which lead to more difficulties since an external rule modifies the all matrix. The application condition of a rule during this step of the strategy heavily depend on the subset $\mathsf{X}^1(\mathcal{C})$ which stock the right hand term variable of the deducible constraint whose second order variable are not in $S_2(\mathcal{C})$. Hence the measure that we use to show the termination of this step also depend on it.

Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a constraint system, we define a lexical measure on $\mathcal{C}$, denoted $\mu_{1.c}(\mathcal{C})$, which is composed by :

1. The multiset $\{\{(j,p) \mid (X, i \overset{?}{\vdash} u) \in D(\mathcal{C}), u|_p \in \mathsf{X}^1(\mathcal{C}), \mathsf{ind}_{\mathcal{C}}(u|_p) = j \text{ and } X \in S_2\}\}$.
2. The multiset $\{\{X, Y \mid (X, s \overset{?}{\vdash} u) \in D(\mathcal{C}), (Y, s \overset{?}{\vdash} v) \in D(\mathcal{C}), X, Y \notin S_2(\mathcal{C}) \text{ and } u = v \in \mathcal{X}^1\}\}$.
3. The number of (X, f) such that $Er \not\vdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$, $(X, i \overset{?}{\vdash} u) \in D$, $X \in S_2$, $\mathsf{f} \in \mathcal{F}_c$ and $vars(u) \cap \mathsf{X}^1(\mathcal{C}) \neq \emptyset$.
4. The number of (X, ξ) such that $(X, i \overset{?}{\vdash} u) \in D$, $(\xi, j \rhd v) \in \Phi$, $j \leq i$, $Er \not\vdash X \overset{?}{\neq} \xi$, $X \in S_2$ and $vars(u) \cap \mathsf{X}^1(\mathcal{C}) \neq \emptyset$.

By definition, if $\mathcal{C} = \bot$, we say that $\mu_{1.c}(\mathcal{C}) = (\emptyset, \emptyset, 0, 0)$.

Furthermore, we define the following measure for matrices of constraint systems: Let (M, M') be a pair of matrices of constraint systems. We define a measure from $\mu_{1.c}(\cdot)$, denoted $\mu^k_{1.c}(M, M')$, which is the following multiset:

$$\mu^k_{1.c}(M, M') = \{\{\mu_{1.c}(\mathcal{C}) \mid i \in \mathbb{N} \text{ and } \mathcal{C} \text{ is on the } i^{\text{th}} \text{ line and } k^{\text{th}} \text{ column of } (M, M')\}\}$$

Thanks to this measure, we are now able to show that the Step c of Phase 1 of the strategy terminates with parameter s and k.

Lemma 8.14. *Let (M, M') be a pair of matrices of constraint systems obtained during the Step c of Phase 1 of the strategy with parameter s and k respectively for support and column. Let $R(\tilde{p})$ be the next possible rule applicable according to step c of the strategy and let (M_1, M_1') and (M_2, M_2') be the two pairs of matrices of constraint systems obtained by application of $R(\tilde{p})$ on (M, M') (in the case of $R(\tilde{p})$ being the rule* Eq-right-right*, there is only one pair of constraint system since* Eq-right-right *is applied internally). We have that:*

$$\mu^k_{1.c}(M_1, M_1') < \mu^k_{1.c}(M, M') \quad \wedge \quad \mu^k_{1.c}(M_2, M_2') < \mu^k_{1.c}(M, M')$$

8.4.1.4 Termination of Phase 1, Cycle of steps b and c

We previously showed that the Step b and c terminates. However, since after Step c, we apply once again Step b as part of a cycle, it remains to prove that the cycle Step b + Step c terminates. For this purpose, we define a measure on pair of matrices using $\mathsf{ind}_{\mathcal{C}}(\cdot)$. Let (M, M') be a pair of matrices of constraint systems. Assume w.l.o.g. that the parameter k corresponds to a column of M. We define a measure, denoted $\mu^k_{1.b+c}(M, M')$, such that:

$$\mu^k_{1.b+c}(M, M') = \max \left\{ \mathsf{ind}_{\mathcal{C}}(u) \;\middle|\; \begin{array}{l} i \in \mathbb{N}, X \notin S_2, (X, j \overset{?}{\vdash} u) \in D(\mathcal{C}) \text{ and } \mathcal{C} \text{ is} \\ \text{on the } i^{\text{th}} \text{ line and } k^{\text{th}} \text{ column of } (M, M') \end{array} \right\}$$

We will first show that $\mu^k_{1.b+c}(M, M')$ can not increase for any application of rule during step b and c. However, the strategy of Step c will allow us to show that between the beginning of Step c and the end of Step c, the measure strictly decreases. Indeed, at the beginning of Step c, every internal deducible constraints contain only variable as right hand term. Furthermore, we know that these deducible constraints can either be removed thanks to Eq-right-right or either be instantiated by the application of Axiom. But the choice of the rule and its parameters are determined according to the minimal for $\mathcal{L}^1_\cdot(\cdot)$. Hence, in the case of the rule Axiom, we always instantiate a variable x by a term that can only contains variables appearing strictly at a earlier stage in the constraint system.

Lemma 8.15. *Let (M, M') be a pair of matrices of constraint systems obtained at the end of the Step c of Phase 1 of the strategy with parameter s and k respectively for support and column. Let (M_1, M'_1) be a pair of matrices of constraint systems obtained by application on (M, M') of Steps b and c with the same parameters. $\mu^k_{1.b+c}(M_1, M'_1) < \mu^k_{1.b+c}(M, M')$.*

8.4.1.5 Termination of Phase 1, Step d

As explained in Subsection 7.4, after the cycle of steps $b+c$, it remains to simplify the external deducible constraints such that they only have variables as right hand terms. To do so, we apply the external rules Eq-right-right, Cons and Axiom as long as they are strongly applicable on $M_{i,k}$ by increasing order on the index of the line i (if we assume that k corresponds to an index of the matrix M in the pair (M, M')).

To prove the termination of this cycle, we will use the measure $\mu_{1.b}()$ on constraint system that was used in Step b. Thanks to Lemma 8.13, we know that the measure $\mu_{1.b}()$ strictly decrease for a rule strongly applicable on a constraint system. Assume that n is the number of line in M and M', we define a lexical measure on (M, M'), denoted $\mu^k_{1.d}(M, M')$, as follows:

1. The number $n - i_0$ where i_0 is the maximal index of the line such that for all $i \leq i_0$, $M_{i_0,k} = \bot$ or $M_{i,k}$ satisfies the invariant $\mathsf{InvVarConstraint}(s)$
2. $\mu_{1.b}(M_{i_0+1,k})$
3. The number of (X, path) such that $(X, i \overset{?}{\vdash} u) \in D(M_{\ell,k})$, $(\xi, j \rhd v) \in \Phi(M_{\ell,k})$, $j \leq i$, $X \overset{?}{\neq} \xi$ is not in $Er(M_{\ell,k})$, $X \in S_2$, $\mathsf{path}(\xi) = \mathsf{path}$, $\ell \in \{1, \ldots, n\}$ and there exist $\mathsf{f} \in \mathcal{F}_c$ such that $\mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ in $Er(M_{\ell,k})$

We will assume for this measure that $M_{n',k} = \bot$ where for all $n' > n$.

Lemma 8.16. *Let (M, M') be a pair of matrices of constraint systems obtained during Step d of Phase 1 of the strategy with parameter s and k respectively for support and column. Furthermore, let $R(\tilde{p})$ be the next possible rule applicable according to step d of the strategy and let (M_1, M'_1) and (M_2, M'_2) be the two pairs of matrices of constraint systems obtained by application of $R(\tilde{p})$ on (M, M'). We have that:*

$$\mu^k_{1.d}(M_1, M'_1) < \mu^k_{1.d}(M, M') \quad \wedge \quad \mu^k_{1.d}(M_2, M'_2) < \mu^k_{1.d}(M, M')$$

8.4.1.6 Termination of Phase 1, Step e

This phase does not need a termination proof since it only consists of modifying the constraint systems in the matrices, *i.e.* no rules are applied during Step e.

8.4.2 Association table

We introduced in Subsection 7.4 the notion of association table. We described how they evolved during Step b and c of Phase 2. Hence, we will now show how the association tables are used for the termination proof of the cycles Step $b + c$ of Phase 2.

First of all, we need to show some properties on association tables that are satisfied for each pair of matrices during Step b and c.

Lemma 8.17. *Let (M, M') be a pair of matrices of constraint systems obtained during Step b or Step c of Phase 2 of the strategy. For all constraint systems $\mathcal{C}$, $\mathcal{C}'$ in (M, M') and their respective association tables T, T', for all $\bigvee_i^n x_i \stackrel{?}{\neq} v_i$, for all $\bigvee_j^m \beta_j \stackrel{?}{\neq} \beta'_j$, if $T[\bigvee_i^n x_i \stackrel{?}{\neq} v_i] = \bigvee_j^m \beta_j \stackrel{?}{\neq} \beta'_j$, then we have that:*

$$\left(\bigvee_j^m \beta_j \mathsf{acc}^1(\mathcal{C}) \stackrel{?}{\neq} \beta'_j \mathsf{acc}^1(\mathcal{C})\right)\downarrow = \bigvee_i^n x_i \stackrel{?}{\neq} v_i$$

Moreover, if $T[\bigvee_i^n x_i \stackrel{?}{\neq} v_i] = \bigvee_j^m \beta_j \stackrel{?}{\neq} \beta'_j$ and for all $k \in \{1, \ldots, m\}$, $st(\beta_j, \beta'_j) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$ then there exists $\bigvee_i^{n'} x'_i \stackrel{?}{\neq} v'_i$ such that $T'[\bigvee_i^{n'} x'_i \stackrel{?}{\neq} v'_i] = \bigvee_j^m \beta_j \stackrel{?}{\neq} \beta'_j$.

We now describe the main measure that will never increase during Step b and c of Phase 2. In the next paragraphs, we will show that this measure necessary decrease after one cycle Step $b + c$. For a inequation $u \stackrel{?}{\neq} v$ in $Eq(\mathcal{C})$, we define a measure, denoted $L^1_{\mathcal{C}}(u \stackrel{?}{\neq} v)$ such that $L^1_{\mathcal{C}}(u \stackrel{?}{\neq} v) = \{\{\mathsf{ind}_{\mathcal{C}}(u); \mathsf{ind}_{\mathcal{C}}(v)\}\}$. We extend this measure to disjunction of inequations such that:

$$L^1_{\mathcal{C}}(\bigvee_i^n u_i \stackrel{?}{\neq} v_i) = \max\{L^1_{\mathcal{C}}(u_i \stackrel{?}{\neq} v_i) \mid i = 1 \ldots n\}$$

We defined a similar measure for inequation of context of recipe, denoted $L^2_{\mathcal{C}}()$, such that $L^2_{\mathcal{C}}(\beta \stackrel{?}{\neq} \beta') = \{\{\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\beta); \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\beta')\}\}$. Furthermore, we also extend this notion to disjunction of inequations such that:

$$L^2_{\mathcal{C}}(\bigvee_i^n \beta_i \stackrel{?}{\neq} \beta'_i) = \max\{L^2_{\mathcal{C}}(\beta_i \stackrel{?}{\neq} \beta'_i) \mid i = 1 \ldots n\}$$

The two measures $L^1_{\mathcal{C}}()$ and $L^2_{\mathcal{C}}()$ are related as shown in the next lemma.

Lemma 8.18. *Let $\mathcal{C}$ be a well-formed constraint system satisfying $\mathsf{InvVarConstraint}(\infty)$. Let β, β' be two contexts of recipes such that $\beta, \beta' \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}), \mathcal{X}^2)$. We have that $L^1_{\mathcal{C}}(\beta\mathsf{acc}^1(\mathcal{C}) \stackrel{?}{\neq} \beta'\mathsf{acc}^1(\mathcal{C})) \leq L^2_{\mathcal{C}}(\beta \stackrel{?}{\neq} \beta')$. Furthermore, if $\beta \in (\mathcal{F}_d^* \cdot \mathcal{AX})$ or $\beta' \in (\mathcal{F}_d^* \cdot \mathcal{AX})$, then we have that $L^1_{\mathcal{C}}(\beta\mathsf{acc}^1(\mathcal{C}) \stackrel{?}{\neq} \beta'\mathsf{acc}^1(\mathcal{C})) < L^2_{\mathcal{C}}(\beta \stackrel{?}{\neq} \beta')$.*

We define *the general measure on matrices*, denote $\mu_{gen}()$, such that: Let (M, M') be a pair of matrices of constraint systems, having the same structure and satisfying $\mathsf{InvVarConstraint}(\infty)$. Let $\mathcal{S}$ be the set defined such that

$$\mathcal{S} = \{D \mid T[E] = D \text{ for some association table } T \text{ in } (M, M') \text{ and some } E\}$$

In a pair (M, M'), a same formula on context of recipes may appear several times in the association tables of (M, M'). The set $\mathcal{S}$ represents the set of all these different formulas on contexs of recipes. Let $\mathcal{C}_0$ be a constraint system in (M, M') such that $\mathcal{C}_0 \neq \bot$. We have:

$$\mu^1_{gen}(M, M') = \left\{\left\{ L^1_{\mathcal{C}}(D) \;\middle|\; \begin{array}{l} \mathcal{C} \text{ are its association table } T \text{ are in } M \text{ or } M' \\ Eq(\mathcal{C}) = E \wedge D, \text{ for some } E \text{ and some disjunction } D \\ T[D] = \bot \end{array} \right\}\right\}$$

$$\mu^2_{gen}(M, M') = \{\{\; L^2_{\mathcal{C}_0}(D') \mid \; D' \in \mathcal{S} \;\}\}$$

$$\mu_{gen}(M, M') = \mu^1_{gen}(M, M') \cup \mu^2_{gen}(M, M')$$

Intuitively, $\mu^1_{gen}(M, M')$ represents the measure for the inequation that are not matched by the Eq-right-right rule yet. On the other hand, $\mu^2_{gen}(M, M')$ represents the measure for the inequation that were matched at one point in the strategy by the rule Eq-right-right.

Lemma 8.19. *Let (M, M') be a pair of matrices of constraint systems obtained during Step b or Step c of Phase 2 of the strategy. Let $R(\tilde{p})$ be one of the following external rules:* Axiom, Cons, Eq-right-right. *Let (M_1, M_1') and (M_2, M_2') be the results of the application of $R(\tilde{p})$ on (M, M'). We have that:*

$$\mu_{gen}(M_1, M_1') \leq \mu_{gen}(M, M') \quad \textit{and} \quad \mu_{gen}(M_2, M_2') \leq \mu_{gen}(M, M')$$

Furthermore, for $i = 1, 2$, if a removal transformation was applied on the association table of (M_i, M_i') then $\mu_{gen}(M_i, M_i') < \mu_{gen}(M, M')$.

8.4.3 Termination of all steps of Phase 2 of the strategy

In this subsection, we describe all the measures that we use to show the termination of each step of phase 2 of the strategy.

8.4.3.1 Termination of Phase 2, Step a

This step of the strategy consists of getting rid of the universal variable. Thus, to show that this step terminate, we introduce a measure on the formulas Eq of the constraint systems that checks the positions of all the universal variables in Eq. Let (M, M') be a pair of matrices of constraint systems, we define a lexical measure, denoted $\mu_{2.a}(M, M')$, which is composed of :

1. The multiset of the position of any occurrence of universal variables in $Eq(\mathcal{C})$, for any constraint system $\mathcal{C}$ in M and M', i.e. $\{\{p \mid Eq(\mathcal{C}) = E' \wedge [\forall \tilde{y}.\forall x.E'' \vee u \stackrel{?}{\neq} v] \text{ and } u_{|p} = x \text{ and } \mathcal{C} \text{ in } (M, M')\}\}$
2. The number of (X, f) such that $\mathsf{root}(X) \neq \mathsf{f}$ not in Er, $X, i \stackrel{?}{\vdash} u \in D(\mathcal{C})$, $\mathsf{f} \in \mathcal{F}_c$ and $\mathcal{C}$ in (M, M').
3. The number of (X, ξ) such that $X, i \stackrel{?}{\vdash} u \in D$, $\xi, j \stackrel{?}{\vdash} v \in \Phi$, $j \leq i$, $X \neq \xi$ is not in $Er(\mathcal{C})$ and $\mathcal{C}$ in (M, M').

Thanks to this measure we are now able to prove the termination of Step a of Phase 2.

Lemma 8.20. *Let (M, M') be a pair of matrices of constraint systems obtained during Step a of Phase 2 of the strategy. Let $R(\tilde{p})$ be one instance of the rule* Axiom *or* Cons *such that $R(\tilde{p})$ is strongly applicable on at least one constraint system in M or M'. At last, let (M_1, M_1') and (M_2, M_2') be the two pairs of matrices of constraint systems obtained by application of $R(\tilde{p})$ on (M, M'). We have that:*

$$\mu_{2.a}(M_1, M_1') < \mu_{2.a}(M, M') \quad \wedge \quad \mu_{2.a}(M_2, M_2') < \mu_{2.a}(M, M')$$

8.4.3.2 Termination of Phase 2, Step b

This step consists of applying the rules Cons and Eq-right-right as long as we can by decreasing support. The main difficulty of the termination of this step is the fact that Cons introduces new variables while Eq-right-right removes variables but also introduces new inequations in each constraint systems.

In order to prove the termination of this step, we have to introduce a new measure on a pair of matrices (M, M'), denoted $\mu_{2.b}(M, M')$, defined such that if n is the maximal size of the frame in M, M', then $\mu_{2.b}(M, M')$ is the tuple composed of:

— $\mu_{gen}(M, M')$

— The number of disjunctions D such that $Eq(\mathcal{C}) = D \wedge E$ and $T[D] = \bot$, for some constraint system $\mathcal{C}$ and its associative table T in M or M'.

— The number of disjunctions D such that $Eq(\mathcal{C}) = D \wedge E$, $T[D] = D'$ and $st(D') \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$, for some constraint system $\mathcal{C}$ and its associative table T in M or M'.

— $(\mu^n_{cons}(M,M'), \mu^n_{var}(M,M'), \ldots, \mu^1_{cons}(M,M'), \mu^1_{var}(M,M'))$

— The multiset of (X, f) such that $\mathcal{C}$ is in M or M', $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, $Er(\mathcal{C}) \nVdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and there exists $\mathsf{g} \in \mathcal{F}_c$ such that $Er(\mathcal{C}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$.

where for all $i \in \{1, \ldots, n\}$, $\mu^i_{var}(M,M')$ and $\mu^i_{cons}(M,M')$ are defined as follows:

$$\mu^i_{var}(M,M') = |\{x \in vars^1(\mathcal{C}) \mid \mathcal{C} \text{ in } M \text{ or } M', \text{ and } \mathsf{ind}_{\mathcal{C}}(x) = i\}|$$

$$\mu^i_{cons}(M,M') = \mu^i_{cons,1}(M,M') \cup \mu^i_{cons,2}(M,M')$$

$$\mu^i_{cons,1}(M,M') = \left\{\left\{ (h_a, h_b) \;\middle|\; \begin{array}{l} \mathcal{C} \text{ is in } M \text{ or } M' \text{ with } T \text{ its association table,} \\ Eq(\mathcal{C}) = E \wedge (D \vee x \overset{?}{\neq} v) \text{ and } T[D \vee u \neq v] = \bot, \\ (X, i \overset{?}{\vdash} x) \in D(\mathcal{C}),\ \mathsf{root}(v) \in \mathcal{F}_c, \\ Er(\mathcal{C}) \nVdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{root}(v), \\ \text{either } st(v) \cap \mathcal{N} \neq \emptyset \text{ or } i < \mathsf{ind}_{\mathcal{C}}(v), \\ h_a = \max\{\mathsf{h}(p) \mid v|_p \in \mathcal{N}\}, \\ h_b = \max\{\mathsf{h}(p) \mid v|_p \in \mathcal{X}^1 \wedge i < \mathsf{ind}_{\mathcal{C}}(v|_p)\} \end{array} \right\}\right\}$$

$$\mu^i_{cons,2}(M,M') = \left\{\left\{ (h_a, h_b) \;\middle|\; \begin{array}{l} \mathcal{C} \text{ is in } M \text{ or } M' \text{ with } T \text{ its association table,} \\ Eq(\mathcal{C}) = E \wedge (D) \text{ and } T[D] = D' \vee X \neq \xi, \\ (X, i \overset{?}{\vdash} x) \in D(\mathcal{C}),\ \mathsf{root}(\xi) \in \mathcal{F}_c, \\ Er(\mathcal{C}) \nVdash X \overset{?}{\neq} \mathsf{root}(\xi), \\ \text{either } st(\xi) \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) \neq \emptyset \text{ or } i < \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\xi), \\ h_a = \max\{\mathsf{h}(p) \mid \xi|_p \in \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}\}, \\ h_b = \max\{\mathsf{h}(p) \mid \xi|_p \in \mathcal{X}^2 \wedge i < \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\xi|_p)\} \end{array} \right\}\right\}$$

Intuitively, $\mu^i_{cons}(M,M')$ represents the number of possible applications of the rule Cons that a inequation can trigger. For example, assume that we consider the inequation $x \neq \mathsf{f}(\mathsf{g}(a), y)$. h_a represents the maximal height of a name in the inequation, hence $h_a = 2$. Indeed, as long as the name a is not on the root of the inequation, then an application of the rule Cons will be possible:

$$\begin{array}{cc} x \neq \mathsf{f}(\mathsf{g}(a), y) & \\ \downarrow & \text{Cons on } x \\ x_1 \neq \mathsf{g}(a) \vee x_2 \overset{?}{\neq} y & \\ \downarrow & \text{Cons on } x_1 \\ x_3 \neq a \vee x_2 \overset{?}{\neq} y & \text{No more Cons} \end{array}$$

Similarly, h_b represents the number of possible applications of the rule Cons that an inequation can trigger due to a variable with higher index.

Lemma 8.21. *Let (M,M') be a pair of matrices of constraint systems obtained during Step b of the second phase. Let $R(\tilde{p})$ be an applicable rule and (M_1,M_1') and (M_2,M_2') the two pairs of matrices of constraint systems obtained by application of $R(\tilde{p})$ on (M,M'). We have that:*

$$\mu_{2.b}(M_1,M_1') < \mu_{2.b}(M,M') \quad \textit{and} \quad \mu_{2.b}(M_2,M_2') < \mu_{2.b}(M,M')$$

8.4.3.3 Step c: Instantiating the variables

In this paragraph, we show that Step c terminates and we also shows that the general measure, i.e $\mu_{gen}()$, strictly decrease at some point..

The measure used to prove termination of Step c is very simple. Indeed, during Step c, we only apply the rule Axiom which either decrease the number of deducible constraint or add a

disequation on recipe. Let (M, M') be a pair of matrices of constraint systems, let $\mathcal{C}_0$ a constraint system in M or M' such that $\mathcal{C}_0 \neq \bot$, we define a lexical measure $\mu_{2.c}(M, M')$, as follows:

1. The number of deducible constraints, i.e. $|D(\mathcal{C}_0)|$
2. The number of couple (X, ξ) such that $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C}_0)$, $(\xi, j \rhd u) \in \Phi(\mathcal{C}_0)$ and $Er(\mathcal{C}_0) \nvdash X \overset{?}{\neq} \xi$.

Thanks to this measure, we can now state the termination lemma.

Lemma 8.22. *Let (M, M') be a pair of matrices of constraint systems obtained during Step c of the second phase. Let* Axiom$(\tilde{p})$ *be an applicable rule and (M_1, M_1') and (M_2, M_2') the two pairs of matrices of constraint systems obtained by application of* Axiom$(\tilde{p})$ *on (M, M'). We have that:*

$$\mu_{2.c}(M_1, M_1') < \mu_{2.c}(M, M') \quad \text{and} \quad \mu_{2.c}(M_2, M_2') < \mu_{2.c}(M, M')$$

It remains to show that the measure $\mu_{gen}()$ strictly decrease at some point.

Lemma 8.23. *Let (M, M') be a pair of matrices of constraint systems obtained at the end of Step b of the second phase. Assume that there exists (M_1, M_1') obtained at the end of the next Step b of the second phase such that $(M, M') \rightarrow^* (M_1, M_1')$. At last, assume that there exists (M_0, M_0') obtained at the end of step c of the second phase such that $(M_1, M_1') \rightarrow^* (M_0, M_0')$. In such a case, $\mu_{gen}(M_0, M_0') < \mu_{gen}(M, M')$.*

8.5 Toward a more powerful attacker

In this section, we explore the possibility of extending the power of the attacker to not only equality test between two messages. Indeed, an attacker may in fact have other ways to compare two messages such as comparing the size of messages. Some encryption scheme also allow one to check if two cypher were encrypted under the same key, without necessarily know the key or plain text themselves. In this section, we do not pretend to solve the trace equivalence with such attacker but we show a reduction result that simplify such problem.

8.5.1 Semantic with predicate

To model the new capabilities of the attacker, we will consider a predicate $\mathcal{P}$ of on ground first order terms. For example, in the case of cyphers encrypted with the same key, we would consider that $\mathcal{P}(u, v)$ is true if and only if $u = \mathsf{senc}(u', k)$ and $v = \mathsf{senc}(v', k)$ for some u', v', k. For a substitution σ, and two term u, v, we say $\mathcal{P}(u, v)$ is satisfied, denoted $\sigma \vDash \mathcal{P}(u, v)$ if and only if $\mathcal{P}(u\sigma, v\sigma)$ is true.

With this new predicate available for the intruder, we have to redefine the notion of static equivalence, trace equivalence and so on:

Definition 8.6 (static equivalence w.r.t. $\mathcal{P}$)**.** *Let $\mathcal{E}$ a set of private names. Let Φ and Φ' two frames. We say that $\nu\mathcal{E}.\Phi$ and $\nu\mathcal{E}.\Phi'$ are statically equivalent w.r.t. the predicate $\mathcal{P}$, written $\nu\mathcal{E}.\Phi \sim_c^{\mathcal{P}} \nu\mathcal{E}.\Phi'$, when $\mathrm{dom}(\Phi) = \mathrm{dom}(\Phi')$, when $\nu\mathcal{E}.\Phi \sim_c \nu\mathcal{E}.\Phi'$ and when for all terms M, N such that $fvars(M, N) \subseteq \mathrm{dom}(\Phi)$ and $fnames(M, N) \cap \mathcal{E} = \emptyset$, if $\mathsf{Message}(M\Phi)$ and $\mathsf{Message}(N\Phi)$ then $\mathcal{P}(M\Phi\downarrow, N\Phi\downarrow)$ is true if and only if $\mathcal{P}(M\Phi'\downarrow, N\Phi'\downarrow)$ is true.*

As expected, the static equivalence w.r.t. $\mathcal{P}$ implies the static equivalence while the other implication is not true.

Definition 8.7 (trace equivalence w.r.t. $\mathcal{P}$)**.** *Let A and B be processes with the same set of private names $\mathcal{E}$. $A \sqsubseteq_t^{\mathcal{P}} B$ if for every $(s, \nu\mathcal{E}.\Phi) \in \mathsf{trace}_c(A)$, there exists $(s, \nu\mathcal{E}.\Phi') \in \mathsf{trace}(B)$ such that $\nu\mathcal{E}.\Phi \sim_c^{\mathcal{P}} \nu\mathcal{E}.\Phi'$.*

Two closed processes A and B are trace equivalent w.r.t. the predicate $\mathcal{P}$*, denoted by $A \approx_t^{\mathcal{P}} B$, if $A \sqsubseteq_t^{\mathcal{P}} B$ and $B \sqsubseteq_t^{\mathcal{P}} A$.*

8.5.2 Toward deciding the trace equivalence w.r.t. a predicate

Intuitively, to decide the trace equivalence w.r.t. $\mathcal{P}$, we would have to go through all the lemma and theorem of this part. Hopefully, some of these lemma either does not depend on the static equivalence or only consider the static equivalence as hypothesis. Since the static equivalence w.r.t. $\mathcal{P}$ implies the static equivalence, most of the lemma are easily adaptable.

In particular, we first reduce the problem of trace equivalence w.r.t. the predicate $\mathcal{P}$, to the problem of deciding the symbolic equivalence between set of constructor constraint system. In order to do that, we first have to define again the static equivalence for constructor frame and the symbolic equivalence between set of constraint system.

Definition 8.8 (static equivalence w.r.t. $\mathcal{P}$)**.** *Two ground constructor frames Φ and Φ' are statically equivalent w.r.t. the predicate $\mathcal{P}$, denoted $\Phi \sim^{\mathcal{P}} \Phi'$, if and only if $\mathrm{dom}(\Phi) = \mathrm{dom}(\Phi')$, $\Phi \sim \Phi'$ and for all $\xi, \xi \in \Pi_n$, if $param(\{\xi, \xi'\}) \subseteq \mathrm{dom}(\Phi)$ and $\xi\Phi\downarrow, \xi'\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$, then $\mathcal{P}(\xi\Phi\downarrow, \xi'\Phi\downarrow)$ is equivalent to $\mathcal{P}(\xi\Phi'\downarrow, \xi'\Phi'\downarrow)$.*

Definition 8.9 (symbolic equivalence w.r.t. $\mathcal{P}$)**.** *Two sets Σ and Σ' of constructor constraint systems having the same structure are symbolically equivalent w.r.t. predicate $\mathcal{P}$ denoted $\Sigma \approx_s^{\mathcal{P}} \Sigma'$, if and only if for all $\mathcal{C} \in \Sigma$, for all $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, there exists $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ and $\Phi\sigma \sim^{\mathcal{P}} \Phi'\sigma'$ (and conversely) where Φ and Φ' are the respective frames of $\mathcal{C}$ and $\mathcal{C}'$.*

As mentioned, to show that deciding trace equivalence w.r.t. the predicate $\mathcal{P}$ can be reduced to the problem of deciding symbolic equivalence w.r.t. the predicate $\mathcal{P}$, we follow the same step as for the proof of Theorem 6.2. In particular, we show in Section 6.3.1.2 how it is possible to consider that the attacker does not have access to public name. This is possible thanks to a mobilisation of infinite names. To follow this modelisation, we ask that the predicate we consider satisfy the following property:

Property 8.1. *Let $\mathcal{E}$ be a finite set of names. For all u, v closed constructor terms, there exists $N \in \mathbb{N}$ such that for all $N' > N$, $\mathcal{P}(u, v)$ if and only if $\mathcal{P}(u\sigma_{\mathcal{E},N,a}, v\sigma_{\mathcal{E},N,a})$.*

Intuitively, this property ensures that the terms that model our infinite names cannot be distinguish by the predicate. We can than state the first reduction result:

Lemma 8.24. *Let $\mathcal{P}$ be a predicate on first order term satisfying Property 8.1. Given a decision procedure for the symbolic equivalence w.r.t. $\mathcal{P}$ of sets of constructor constraint systems, the problem of trace equivalence w.r.t. $\mathcal{P}$ between two concrete processes is decidable.*

8.5.3 Toward deciding the symbolic equivalence w.r.t. a predicate

As in the previous section, the idea for deciding the symbolic equivalence w.r.t. a predicate is to use our decision procedure. In particular, we can easily show that the soundness and completeness of our algorithm holds for the symbolic equivalence w.r.t. a predicate $\mathcal{P}$. Indeed, one can note that the core lemmas 8.4, 8.5, 8.6 and 8.7 still hold. Moreover, one can easily adapt the proof of Proposition 8.1 and Theorems 8.1 and 8.2, by replacing the static equivalence $\sim$ by $\sim^{\mathcal{P}}$ and the symbolic equivalence $\approx_s$ by $\approx_s^{\mathcal{P}}$.

Hence, since the soundness and completeness hold even for the symbolic equivalence w.r.t. a predicate, we reduced the problem to the problem of deciding the symbolic equivalence w.r.t. a predicate on the leaves of our algorithm. Although the properties of such leaves are described in detail in Section 8.3, we simplify in this section the formalism of such leaves and extract our main reduction result.

Consider the definition of a solved constraint system (Definition 8.4). We know thanks to Lemma 8.8 that every constraint system in a leaf satisfies this definition. But item 2 indicates that no deductibility constraint is restricted by an equation on second order solution. Moreover, we know that all constraint system in a line of the matrix have same set Er. Hence, the sets Ers of the constraint systems on the leaves does not have any impact on the decidability of

symbolic equivalence w.r.t. $\mathcal{P}$ of two sets on a leaf. Similarly, we know that all equation in Eq is automatically satisfied once the variable in D are instantiate. By extending this reasoning to all the invariant satisfied by the constraint system on a leaf, we can define the notion of simplified constraint system and the notion of solution of simplified constraint system

Definition 8.10. *A* simplified constraint system *is a triple* $(\Phi; D; Eq)$*:*

- Φ *is a simplified frame* $\{\xi_1 \rhd u_1, \ldots, \xi_n \rhd u_n\}$ *where* u_i *are constructor terms and* ξ_i *are recipes.*
- D *is a set of deducible constraints of the form* $X, i \overset{?}{\vdash} x$ *with* $i \in \mathbb{N}$, $X \in \mathcal{X}^2$, $x \in \mathcal{X}^1$.
- Eq *is a set of inequations of the form* $t \overset{?}{\neq} t'$ *where* t, t' *are constructors terms that do not contain names.*

We also assume the following conditions are satisfied on a constraint system:

1. for every $x \in vars^1(D)$, there exists a unique X such that $(X, i \overset{?}{\vdash} x) \in D$, and each variable X occurs at most once in D.
2. $vars^1(\mathcal{C}) \subseteq vars^1(D)$
3. for every $1 \leq k \leq n$, for every $x \in vars^1(t_k)$, if $ax_j \in vars^2\xi_k$ then there exists $(X, i \overset{?}{\vdash} x) \in D$ such that $i < j$.

Given a simplified frame $\Phi = \{\xi_1 \rhd t_1, \ldots, \xi_n \rhd t_n\}$ and a recipe ξ, we say that ξ is build from Φ if there exists a context $C[\__1, \ldots, \__m]$ containing only constructor function symbol and $j_1, \ldots, j_m \in \{1, \ldots, n\}$ such that $\xi = C[\xi_{j_1}, \ldots, \xi_{j_m}]$. Moreover, we denote $\xi\Phi$ the term $C[t_{j_1}, \ldots, t_{j_m}]$.

Definition 8.11 (solution)**.** *A* solution *of a simplified constraint system* $\mathcal{C} = (\Phi; D; Eq)$ *is a pair of substitutions* (σ, θ) *such that* σ *is a mapping from* $vars^1(\mathcal{C})$ *to* $\mathcal{T}(\mathcal{F}_c, \mathcal{N})$*,* θ *is a mapping from* $vars^2(\mathcal{C})$ *to* $\mathcal{T}(\mathcal{F}, \mathcal{A}\mathcal{X})$*, and:*

1. *for all* $(X, k \overset{?}{\vdash} x) \in D$*,* $X\theta$ *is built from* $\Phi\theta$*,* $(X\theta)(\Phi\theta\sigma) = x\sigma$*, and* $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$*;*
2. *for all* $t \overset{?}{\neq} t'$*,* $t\sigma \neq t'\sigma$

The substitution σ *is called the* first-order solution *of* $\mathcal{C}$ *associated to* θ*, called* second-order solution *of* $\mathcal{C}$*. The set of solutions of a constraint system* $\mathcal{C}$ *is denoted* $\mathsf{Sol}(\mathcal{C})$*.*

Note that in the solution of a constraint system, we only consider the recipe that are built from Φ, *i.e.* that are built from a constructor context over Φ. In fact in the constraint system on a leave, it is also the case due to the invariant $\mathsf{InvDest}(\infty)$. Similarly, in the definition of simplified constraint system, we only considered distinct variables as right hand term for variables thanks to the invariant $\mathsf{InvVarConstraint}(\infty)$.

Definition 8.12 (simplified static equivalence)**.** *Let* Φ *and* Φ' *two closed simplified frames having the same structure. We say that* Φ *and* Φ' *are in simplified static equivalence if for all* ξ, ξ' *built from* Φ *(thus also built from* Φ'*),* $\xi\Phi = \xi'\Phi$ *if and only if* $\xi\Phi' = \xi'\Phi'$*. Moreover, we say that* Φ *and* Φ' *are in simplified static equivalent w.r.t. the predicate* $\mathcal{P}$*, denoted* $\Phi \sim^{\mathcal{P}} \Phi'$ *if* Φ *and* Φ' *are in simplified static equivalence and for all* ξ, ξ' *built from* Φ*,* $\mathcal{P}(\xi\Phi, \xi'\Phi)$ *if and only if* $\mathcal{P}(\xi\Phi', \xi'\Phi')$*.*

Definition 8.13 (simplified symbolic equivalence)**.** *Let* Σ *and* Σ' *be two sets of simplified constraint systems that contain constraint systems having the same structure. We say that* Σ *and* Σ' *are in simplified symbolic equivalence, denoted by* $\Sigma \approx_s \Sigma'$*, if for all* $\mathcal{C} \in \Sigma$*, for all* $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$*, there exists* $\mathcal{C}' \in \Sigma'$ *and a substitution* σ' *such that* $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ *and* $\Phi\theta\sigma \sim \Phi'\theta\sigma'$ *(and conversely) where* $\mathcal{C} = (\Phi; D; Eq)$ *and* $\mathcal{C}' = (\Phi'; D'; Eq')$*.*

We say that Σ *and* Σ' *are in simplified symbolic equivalence w.r.t. the predicate* $\mathcal{P}$*, denoted by* $\Sigma \approx_s^{\mathcal{P}} \Sigma'$*, if for all* $\mathcal{C} \in \Sigma$*, for all* $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$*, there exists* $\mathcal{C}' \in \Sigma'$ *and a substitution* σ' *such that* $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ *and* $\Phi\theta\sigma \sim^{\mathcal{P}} \Phi'\theta\sigma'$ *(and conversely) where* $\mathcal{C} = (\Phi; D; Eq)$ *and* $\mathcal{C}' = (\Phi'; D'; Eq')$*.*

With this simpler formalism, we can establish our main reduction result:

Theorem 8.4. *Consider a predicate $\mathcal{P}$ that satisfies Property 8.1. Given an algorithm for deciding the simplified symbolic equivalence w.r.t. $\mathcal{P}$ between two sets of simplified constraint systems Σ and Σ' that contain constraint systems having the same structure and such that there exists $a \in \mathcal{N}$ such that for all $(\Phi; D; Eq) \in \Sigma \cup \Sigma'$, $(ax_1 \rhd a) \in \Phi$, and for all $\mathcal{C}, \mathcal{C}' \in \Sigma \cup \Sigma'$, $\mathcal{C} \approx_s \mathcal{C}'$, $D = D'$ and $Eq = Eq'$ where $\mathcal{C} = (\Phi; D; Eq)$ and $\mathcal{C}' = (\Phi'; D'; Eq')$, we can derive an algorithm for deciding trace equivalence w.r.t. $\mathcal{P}$ between two bounded processes.*

Part III

ProVerif

Chapter 9

Proving more observational equivalences with ProVerif

Contents

ProVerif was first a protocol analyser for trace properties. It implements an algorithm, based on Horn clauses, that accepts protocols written in the applied pi-calculus. The behaviour of the cryptographic primitives is represented by an equational theory and/or rewrite rules. One reason for the success of this tool may come from the fact that ProVerif does not depend on a specific equational theory and allows the user to describe himself the behaviour of his cryptographic primitives.

Since [BAF08], ProVerif can also prove equivalence properties. It focuses on proving equivalences $P \approx Q$ in which P and Q are two variants of the same process obtained by selecting different terms for P and Q. Hence, [BAF08] introduced the notion of *biprocess* and the operator diff where $P(\mathsf{diff}[M_1, M_2])$ represents the two variants $P(M_1)$ and $P(M_2)$, obtained by giving different interpretations to $\mathsf{diff}([M_1, M_2])$, i.e. either selecting M_1 or M_2. In [BAF08], the authors also introduce an operational semantics for biprocesses where $P(\mathsf{diff}(M_1, M_2])$ behaves the same way as $P(M_1)$ and $P(M_2)$ if the intruder cannot distinguish $P(M_1)$ from $P(M_2)$. Furthermore, a reduction of $P(\mathsf{diff}(M_1, M_2])$ necessary becomes a biprocess. On the other hand, if an intruder can distinguish $P(M_1)$ from $P(M_2)$, then the operational semantics specifies that $P(\mathsf{diff}(M_1, M_2])$

gets stuck. Hence, to prove the equivalence between $P(M_1)$ and $P(M_2)$, it is sufficient to prove that $P(\mathsf{diff}[M_1, M_2])$ never gets stuck.

However, the notion of equivalence used by ProVerif is stronger than the usual trace equivalence or observational equivalence. As we described earlier, ProVerif requires that the two processes have the same control structure and that all tests yield the same result in both processes. Thus, for a protocol that does not satisfy this condition, ProVerif will fail to prove equivalence, i.e. it will yield a false attack.

Example 9.1. *Consider for example the two naives protocols:*

$$P : \mathsf{in}(c, x).\mathsf{if}\ x = \mathsf{pk}(sk_A)\ \mathsf{then}\ \mathsf{out}(c, \{s\}_{\mathsf{pk}(sk_A)})\ \mathsf{else}\ \mathsf{out}(c, \{N_p\}_{\mathsf{pk}(sk_A)})$$
$$Q : \mathsf{in}(c, x).\mathsf{if}\ x = \mathsf{pk}(sk_B)\ \mathsf{then}\ \mathsf{out}(c, \{s\}_{\mathsf{pk}(sk_B)})\ \mathsf{else}\ \mathsf{out}(c, \{N_q\}_{\mathsf{pk}(sk_B)})$$

For simplicity, we omitted the name restriction but we assume that all names but c are private. However, the we assume that the intruder knows the public keys of A and B, i.e. $\mathsf{pk}(sk_A)$ *and* $\mathsf{pk}(sk_B)$ *respectively. The protocol P is simply waiting for the public key of the agent A (*$\mathsf{pk}(sk_A)$*) on a channel c, and if he receives it then sends some secret s encrypted with A's public key else a fresh nonce N_p encrypted with A's public key is sent on the channel c. On the other hand, the protocol Q does similar actions but is waiting for the public key of the agent B (*$\mathsf{pk}(sk_B)$*) instead of A. Assuming that the attacker does not have access to the private keys sk_A and sk_B, then the two protocols are equivalent since the attacker cannot differentiate $\{s\}_{\mathsf{pk}(sk_A)}$ and $\{N_q\}_{\mathsf{pk}(sk_A)}$ (resp. $\{s\}_{\mathsf{pk}(sk_B)}$ and $\{N_q\}_{\mathsf{pk}(sk_B)}$).*

However, even if those two protocols can be transformed into a biprocess, i.e.

$$\mathsf{in}(c, x).\mathsf{if}\ x = \mathsf{diff}[\mathsf{pk}(sk_A), \mathsf{pk}(sk_B)]\ \mathsf{then}\ \mathsf{out}(c, \mathsf{diff}[\{s\}_{\mathsf{pk}(sk_A)}, \{s\}_{\mathsf{pk}(sk_B)}])\ \mathsf{else}\ \mathsf{out}(c, \mathsf{diff}[\{N_p\}_{\mathsf{pk}(sk_A)}, \{N_q\}_{\mathsf{pk}(sk_B)}])$$

ProVerif *will yield a false negative due to the conditional. Indeed, if the attacker sends the public key of A, then the left part of the test* $x = \mathsf{diff}[\mathsf{pk}(sk_A), \mathsf{pk}(sk_B)]$ *will succeed when the right part will fail. Since the test does not behave the same way on the right and left side, the biprocess gets stuck and so* ProVerif *will be unable to prove the equivalence. More realistic examples illustrating this false attack can be found in several cases studies, e.g. anonymity of the* private authentication protocol *[AF04] and unlinkability of the* Basic Access Control *protocol of the e-passport protocols (see Chapter 3).*

This chapter address the issue of such false attacks. Our main idea is to extend the behaviour of destructor function symbols, so that we can perform tests directly inside terms. Initially, destructors could only be declared using rewrite rules, such as $\mathsf{dec}(\mathsf{enc}(x, y), y) \to x$, which allowed to perform pattern-matching on terms but no inequality test. We have extended ProVerif with inequalities in the definition of destructors. For instance, we can now define a destructor test as follows:

$$\mathsf{test}(x, x, z, t) \to z$$
$$\mathsf{test}(x, y, z, t) \to t \qquad \text{if } x \neq y$$

With this definition, a term $\mathsf{test}(M_1, M_2, M_3, M_4)$ reduces into M_3 if M_1 and M_2 are equal under the equational theory, and into M_4 if M_1 and M_2 are different under the equational theory. This extension allows us, for example, to replace conditional branching in the processes by $\mathsf{test}(\ldots)$. Coming back to our previous biprocess, we can now write a similar biprocess which has the same control behaviuor for both the left and the right side:

$\mathsf{in}(c, x).$
$\mathsf{let}\ y = \mathsf{test}(x, \mathsf{diff}[\mathsf{pk}(sk_A), \mathsf{pk}(sk_B)], \mathsf{diff}[\{s\}_{\mathsf{pk}(A)}, \{s\}_{\mathsf{pk}(sk_B)}], \mathsf{diff}[\{N_p\}_{\mathsf{pk}(sk_A)}, \{N_q\}_{\mathsf{pk}(sk_B)}])$
$\mathsf{in}\ \mathsf{out}(c, y)$

Since the conditional process is removed, ProVerif will be able to prove the equivalence of such a biprocess. Although the transformation was done on a naive protocol, we show that it also can be done on more realistic examples. Moreover, since such transformations may not be very natural, we provide an implementation that will automatically detect and perform these transformations when possible.

9.1 Model

This section introduces our process calculus, by giving its syntax and its operational semantics. As mentioned above, our work extends the behaviour of destructor symbols our syntax and semantics of terms evaluations change in comparison to the original calculus of ProVerif [BAF08]. However, we did not modify the syntax of processes thus the semantics for processes only differ from the changes led by the modifications in the semantics for term evaluation.

9.1.1 Syntax

Similarly to all our previous works, the messages sent on the network by some agents in a protocol are modelled using an abstract term algebra. We assume an infinite set of names $\mathcal{N}$ and an infinite set of variables $\mathcal{X}$. We also consider a signature $\mathcal{F}$ consisting of a finite set of function symbols with their arity.

We saw that the behaviour of cryptographic primitives could be represented by an equational theory (see Part I), or by a rewriting system (see Part II). However, in ProVerif, both representations are used to describe the behaviour of cryptographic primitives. Hence, as in the calculus of ProVerif [BAF08], we distinguish two categories of function symbols: constructors f and destructors g. Constructors build terms; destructors, defined by rewrite rules, manipulate terms, as detailed below. We denote by $\mathcal{F}_c$ and $\mathcal{F}_d$ the set of constructors and destructors, respectively.

Messages M are terms built from variables, names, and constructors applied to terms. The tool ProVerif being more focused on a practical syntax and semantics, an *equational theory* E can only be described by a finite set of equations $M = N$, where M, N are terms without names.

9.1.1.1 Destructors

In [BAF08], the rewrite rules describing the behaviour of destructors follow the usual definition of a rewrite rule, *i.e.* $\mathsf{g}(M_1, \ldots, M_n) \to M$ where $M_1, \ldots, M_n, M$ are terms built on $\mathcal{T}(\mathcal{F}_c, \mathcal{X})$. However, as mentioned in the introduction, we want to introduce tests directly into terms and more specifically into the definition of destructors. Hence, we introduce *formulas* on messages in order to express these tests. We consider formulas ϕ of the form $\bigwedge_{i=1}^{n} \forall \tilde{x}_i.M_i \neq_E N_i$, where $\tilde{x}$ stand for a sequence of variables $x_1, \ldots, x_k$ and E is the equational theory. We denote by $\top$ and $\bot$ the *true* and *false* formulas, respectively corresponding to an empty conjunction ($n = 0$) and to $x \neq_E x$, for instance. Formulas will be used as side conditions for destructors.

Let σ be a substitution mapping variables to ground terms. We define $\sigma \vDash \phi$ as follows: $\sigma \vDash \bigwedge_{i=1}^{n} \forall \tilde{x}_i.M_i \neq_E N_i$ if and only if for $i = 1, \ldots n$, for all σ_i of domain $\tilde{x}_i$, $M_i\sigma_i\sigma \neq_E N_i\sigma_i\sigma$. More generally, we extend $\sigma \vDash \phi$ to formulas using classic logic connectors with equalities and inequalities. Note that this definition of σ satisfying ϕ is similar to the one given in Part II except that the inequalities and equalities are not syntactic anymore.

In [BAF08] and in Part II, destructors are partial functions defined by rewrite rules; when no rewrite rule can be applied, we say that the destructor fails. However, this formalism does not allow destructors to succeed when one of their arguments fails. We shall need this feature in order to include as many tests as possible in terms. Therefore, we extend the definition of destructors by defining *may-fail messages*, denoted by U, which can be messages M, the special value fail, or a variable u. We separate fail from ordinary messages M so that the equational theory does not apply to fail. May-fail messages represent the possible arguments and result of a destructor. We differentiate variables for may-fail messages, denoted u, v, w and variables for messages, denoted x, y, z. A may-fail variable u can be instantiated by a may-fail term while a message variable x can only be instantiated by a message, and so cannot be instantiated by fail. The syntax of our terms is summarised in Figure 9.1.

For two ground may-fail messages U_1 and U_2, we say that $U_1 =_{\mathsf{E}} U_2$ if and only if $U_1 = U_2 = \mathsf{fail}$ or U_1, U_2 are both messages, denoted M_1, M_2, and $M_1 =_E M_2$.

Example 9.2. *Consider the signature $\mathcal{F}$ and the associated equational theory $\mathsf{E}_{\mathsf{aenc}}$ of Example 2.2. Consider U and V defined as follows:*

$M ::=$	message
x, y, z	variable
a, b, c	name
$\mathsf{f}(M_1, \ldots, M_n)$	constructor application
$U ::=$	may-fail message
M	message
fail	failure
u	may-fail variable

Figure 9.1: Syntax of our terms

— $U = \mathsf{proj}_1(\mathsf{adec}(\mathsf{aenc}(\langle n_1, n_2\rangle, \mathsf{pk}(sk)), sk))$ *and* $V = n_1$*: we have* $U =_{\mathsf{E}_{\mathsf{aenc}}} V$
— $U = \mathsf{proj}_1(\mathsf{adec}(\mathsf{aenc}(\langle n_1, n_2\rangle, \mathsf{pk}(sk)), sk))$*,* $V = n_2$*: we have* $U \neq_{\mathsf{E}_{\mathsf{aenc}}} V$
— $U = n_2$ *and* $V = \mathsf{fail}$*: we have* $U \neq_{\mathsf{E}_{\mathsf{aenc}}} V$

Note that $\mathsf{proj}_1(\mathsf{fail})$ *is not a may-fail message nor a message.*

Given a signature $\mathcal{F}$ and an equational theory E, a destructor g of arity n is defined by a finite set of rewrite rules $\mathsf{g}(U_1, \ldots, U_n) \to U \mid\mid \phi$ where $U_1, \ldots, U_n, U$ are may-fail messages that do not contain any name. Furthermore, ϕ is a formula as defined above that does not contain any name. The variables of U and $fvars(\phi)$ are bound in $U_1, \ldots, U_n$. Note that all variables in $fvars(\phi)$ are necessarily message variables. Variables are subject to renaming. When ϕ is the formula $\top$, we omit the formula.

Example 9.3. *Consider a signature* $\mathcal{F}$ *and an equational theory* E*. A symmetric encryption scheme where the decryption function either properly decrypts a ciphertext using the correct private key, or fails. To model this encryption scheme, we consider, in the signature* $\mathcal{F}$*, the constructor* senc *for encryption, the destructor* sdec *for decryption. Decryption can be defined by the rules:*

— $\mathsf{sdec}(\mathsf{senc}(x, y), y) \to x$ *(decryption succeeds)*
— $\mathsf{sdec}(x, y) \to \mathsf{fail} \mid\mid \forall z.x \neq_{\mathsf{E}} \mathsf{senc}(z, y)$ *(decryption fails, because* x *is not a ciphertext under the correct key)*
— $\mathsf{sdec}(\mathsf{fail}, u) \to \mathsf{fail}$*,* $\mathsf{sdec}(u, \mathsf{fail}) \to \mathsf{fail}$ *(the arguments failed, the decryption also fails)*

To avoid confusion, we will call *equational presentation*, denoted Σ, the signature $\mathcal{F}$ with its associated equational theory E and the sets of rewriting rules that describe the destructors of $\mathcal{F}$. Hence, we denote $\mathrm{def}_\Sigma(\mathsf{g})$ the set of rewrite rule describing g in the signature of Σ. Moreover, we denote $U =_\Sigma V$ (resp. $U \neq_\Sigma V$) the equality (resp. inequality) modulo the equational theory of Σ.

Consider $U_1, \ldots, U_n$ may-fail messages and consider g a destructor of arity n. We say that g rewrites $U_1, \ldots, U_n$ into U, denoted $\mathsf{g}(U_1, \ldots, U_n) \to U$, if there exist $\mathsf{g}(U'_1, \ldots, U'_n) \to U' \mid\mid \phi$ in $\mathrm{def}_\Sigma(\mathsf{g})$, and a substitution σ such that $U'_i\sigma =_\Sigma U_i$ for all $i = 1 \ldots n$, $U'\sigma = U$ and $\sigma \vDash \phi$.

At last, we ask that given an equational presentation Σ, for all destructors g of arity n, $\mathrm{def}_\Sigma(\mathsf{g})$ satisfies the following properties:

P1. For all ground may-fail messages $U_1, \ldots, U_n$, there exists a may-fail message U such that $\mathsf{g}(U_1, \ldots, U_n) \to U$.
P2. For all ground may-fail messages $U_1, \ldots, U_n$, if $\mathsf{g}(U_1, \ldots, U_n) \to V_1$ and $\mathsf{g}(U_1, \ldots, U_n) \to V_2$ for some V_1, V_2 then $V_1 =_\Sigma V_2$.

Property P1 expresses that all destructors are total while Property P2 expresses that all destructors are deterministic (modulo the equational theory). Note that thanks to Property P2, a destructor cannot reduce to fail and a message at the same time. We provide two algorithms that allow us to automatically verify that a destructor satisfies Properties P1 and P2 (See Section 9.3.5).

In Example 9.3, the destructor sdec follows the classical definition of the symmetric decryption. However, thanks to the formulas and the fact that the arguments of a destructor can fail, we can describe the behaviour of new primitives.

Example 9.4. *We define a destructor that tests equality and returns either true or false, as follows:*

$$\begin{array}{l}\mathsf{eq}(x,x) \to \mathsf{true}\\ \mathsf{eq}(x,y) \to \mathsf{false} \mid\mid x \neq_\Sigma y\\ \mathsf{eq}(\mathsf{fail},u) \to \mathsf{fail} \qquad \mathsf{eq}(u,\mathsf{fail}) \to \mathsf{fail}\end{array}$$

This destructor fails when one of its arguments fails. We remark that such a destructor could not be defined in ProVerif *without our extension, because one could not test* $x \neq_\Sigma y$.

9.1.1.2 From usual destructors to our extension

More generally, from a destructor defined, as in [BAF08], by rewrite rules $\mathsf{g}(M_1,\ldots,M_n) \to M$ without side conditions and such that the destructor is considered to fail when no rewrite rule applies, we can build a destructor in our formalism. The algorithm is given in Lemma 9.1 below.

Lemma 9.1. *Consider an equational presentation* Σ. *Let* g *be a destructor of arity* n *described by the following set* $\mathcal{S}$ *of rewrite rules:*

$$\{\mathsf{g}(M_1^i,\ldots,M_n^i) \to M^i \mid i = 1,\ldots,m\}$$

Assume that g *is deterministic,* i.e. $\mathcal{S}$ *satisfies Property P2. The following set* $\mathrm{def}_\Sigma(\mathsf{g})$ *satisfies Properties P1 and P2:*

$$\begin{array}{rl}\mathrm{def}_\Sigma(\mathsf{g}) = & \mathcal{S} \cup \{\mathsf{g}(x_1,\ldots,x_n) \to \mathsf{fail} \mid\mid \phi\}\\ & \cup\{\mathsf{g}(u_1,\ldots,u_{k-1},\mathsf{fail},u_{k+1},\ldots,u_n) \to \mathsf{fail} \mid k = 1,\ldots,n\}\end{array}$$

where $\phi = \bigwedge_{i=1}^m \forall \tilde{y}_i.(x_1,\ldots,x_n) \neq_\Sigma (M_1^i,\ldots,M_n^i)$ *and* $\tilde{y}_i$ *are the variables of* $(M_1^i,\ldots,M_n^i)$, *and* $x_1,\ldots,x_n$ *are message variables.*

Proof. Let $U_1,\ldots,U_n$ be ground may-fail messages. Consider first the case where there exists $i \in \{1,\ldots,n\}$ such that $U_i =_\Sigma \mathsf{fail}$. Since $x_1,\ldots,x_n$ are message variables and for all $i \in \{1,\ldots,m\}$, for all $j \in \{1,\ldots,n\}$, M_j^i is a message, the only rules that can rewrite $U_1,\ldots,U_n$ are $\{\mathsf{g}(u_1,\ldots,u_{k-1},\mathsf{fail},u_{k+1},\ldots,u_n) \to \mathsf{fail} \mid k = 1,\ldots,n\}$. Moreover, let σ be the substitution such that $u_j\sigma = U_j$ for all $j \neq i$. We trivially have that $u_j\sigma =_\Sigma U_j$ and by hypothesis $U_i =_\Sigma \mathsf{fail}$. Thus, $\mathsf{g}(U_1,\ldots,U_n) \to \mathsf{fail}$ and there is no message M such that $\mathsf{g}(U_1,\ldots,U_n) \to M$.

Consider now the case where $U_1,\ldots,U_n$ are all messages, that we rename $M_1,\ldots,M_n$. Thus, $M_1,\ldots,M_n$ can only be rewriten by the rules in $\mathcal{S}\cup\{\mathsf{g}(x_1,\ldots,x_n) \to \mathsf{fail} \mid\mid \phi\}$. Let $i \in \{1,\ldots,m\}$. By definition, we have that $M_1,\ldots,M_n$ is rewritten by the rule $\mathsf{g}(M_1^i,\ldots,M_n^i) \to M^i$ if, and only if, there exists a substitution σ such that $\sigma \vDash \exists\tilde{y}_i.(x_1,\ldots,x_n) =_\Sigma (M_1^i,\ldots,M_n^i)$ and $x_j\sigma = M_j$ for $j = 1\ldots n$ where $\tilde{y}_i = \mathit{fvars}(M_1^i,\ldots,M_n^i)$. Hence, $M_1,\ldots,M_n$ can be rewritten by one of the rules in $\mathcal{S}$ if, and only if, there exists a substitution σ such that $\sigma \vDash \neg\phi$. On the other hand, $M_1,\ldots,M_n$ can be rewritten by $\mathsf{g}(x_1,\ldots,x_n) \to \mathsf{fail} \mid\mid \phi$ if, and only if, $\sigma \vDash \phi$. Hence we deduce that $\mathrm{def}_\Sigma(\mathsf{g})$ satisfies Property P1. Moreover, since M^i is a message for all $i = 1\ldots m$, and there is no substitution σ such that $\sigma \vDash \phi \wedge \neg\phi$, we deduce that $\mathrm{def}_\Sigma(\mathsf{g})$ satisfies Property P2. □

The users can therefore continue defining destructors as before in ProVerif; the tool automatically completes the definition following Lemma 9.1. The users will also be able to define destructors using the new formalism, *i.e.* with side conditions and may-fail messages. However, in this case, it is up to the user to provide a set of rewrite rules that satisfies Properties P1 and P2. Indeed, the algorithms described in Section 9.3.5 only check if a set $\mathcal{S}$ of rewrite rules satisfies both properties but these algorithms do not complete $\mathcal{S}$ if it fails to satisfy the desired properties.

$D ::=$	term evaluations
U	may-fail message
eval $\mathsf{h}(D_1, \ldots, D_n)$	function evaluation

Figure 9.2: Syntax of term evaluation

9.1.1.3 Term Evaluation

A *term evaluation*, denoted by D and defined in Figure 9.2, represents the evaluation of a series of constructors and destructors.

eval $\mathsf{h}(D_1, \ldots, D_n)$ indicates that the function symbol will be evaluated. While all destructor must be preceded by eval , some constructors might also be preceded by eval in a term evaluation. In fact, the reader may ignore the prefix eval since eval h and h have the same semantics. However, eval becomes useful when we convert equations into rewrite rules (see. Section 9.2.2). Typically, eval is used to indicate when a term has been evaluated or not.

In order to avoid distinguishing constructors and destructors in the definition of term evaluation, for f a constructor of arity n, we let $\mathrm{def}_\Sigma(\mathsf{f}) = \{\mathsf{f}(x_1, \ldots, x_n) \to \mathsf{f}(x_1, \ldots, x_n)\} \cup \{\mathsf{f}(u_1, \ldots, u_{i-1}, \mathsf{fail}, u_{i+1}, \ldots, u_n) \to \mathsf{fail} \mid i = 1, \ldots, n\}$. The second part of the union corresponds to the failure cases: the constructor fails if and only if one of its arguments fails. We allow may-fail messages in term evaluations. Since no construct will allow us to bind may-fail variables in processes, only messages M and fail may in fact occur.

9.1.1.4 Processes

At last, processes P, Q, R represent protocols and are defined in Figure 9.3. This syntax corresponds exactly to [BAF08] and is very similar to the syntax of the applied pi calculus. However, note that the condition $\mathsf{if}\ M = N\ \mathsf{then}\ P\ \mathsf{else}\ Q$ is not included in our calculus. Indeed, it was replaced by the term evaluation $\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q$ that evaluates the term evaluation D. If D reduces to a message, it stores the result in x and executes P. Otherwise, D reduces to fail, and Q is executed. A trailing 0 can be omitted after an input or an output. An else branch can be omitted when it is $\mathsf{else}\ 0$.

$P, Q, R ::=$	processes
0	nil
$\mathsf{in}(M, x).P$	input
$\mathsf{out}(M, N).P$	output
$P \mid Q$	parallel composition
$!\,P$	replication
$\nu a.P$	restriction
$\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q$	term evaluation

Figure 9.3: Syntax of our processes

Note that even if the condition $\mathsf{if}\ M = N\ \mathsf{then}\ P\ \mathsf{else}\ Q$ is not included in our calculus, it can now be defined as $\mathsf{let}\ x = \mathsf{equals}(M, N)\ \mathsf{in}\ P\ \mathsf{else}\ Q$, where x is a fresh variable and equals is a binary destructor with the rewrite rules $\{\mathsf{equals}(x, x) \to x, \mathsf{equals}(x, y) \to \mathsf{fail} \mid\mid x \neq_\Sigma y, \mathsf{equals}(\mathsf{fail}, u) \to \mathsf{fail}, \mathsf{equals}(u, \mathsf{fail}) \to \mathsf{fail}\}$. The destructor equals succeeds if and only if its two arguments are equal messages modulo the equational theory and different from fail. We always include this destructor in the signature $\mathcal{F}$.

Example 9.5. *The* private authentication protocol *described in Example 3.10 can be expressed*

in the ProVerif *syntax as follows:*

$$A(sk_a, sk_b) \stackrel{def}{=} \nu n_a.\mathsf{out}(c, \mathsf{aenc}(\langle n_a, \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b))).\mathsf{in}(c, z).0$$

$$\begin{aligned} B(sk_b, sk_a) \stackrel{def}{=}\ & \nu n_b.\mathsf{in}(c, y).\mathsf{let}\ x = \mathsf{adec}(y, sk_b)\ \mathsf{in} \\ & \quad \mathsf{let}\ xn_a = \mathsf{proj}_1(x)\ \mathsf{in} \\ & \quad\quad \mathsf{let}\ z = \mathsf{equals}(\mathsf{proj}_2(x), \mathsf{pk}(ska))\ \mathsf{in} \\ & \quad\quad\quad \mathsf{out}(c, \mathsf{aenc}(\langle xn_a, \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))) \\ & \quad\quad \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(sk_b))) \\ & \quad \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(sk_b))) \\ & \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(skb))) \end{aligned}$$

$$System(sk_a, sk_b) \stackrel{def}{=} A(sk_a, sk_b) \mid B(sk_b, sk_a)$$

9.1.2 Semantics

We define in this subsection the semantics for term evaluation and processes.

9.1.2.1 Term evaluation

Given an equational presentation Σ, given a term evaluation D, a term U is the result of an evaluation of D, denoted $D \downarrow_\Sigma U$, and defined as follows:

$$\begin{array}{l} U \downarrow_\Sigma U \\ \text{eval } \mathsf{g}(D_1, \ldots, D_n) \downarrow_\Sigma U\sigma \\ \quad \text{if } \mathsf{g}(U_1, \ldots, U_n) \to U \mid\mid \phi \in \mathrm{def}_\Sigma(\mathsf{g}), \text{ and } \sigma \text{ is such} \\ \quad \text{that for all } i,\ D_i \downarrow_\Sigma V_i,\ V_i =_\Sigma U_i\sigma \text{ and } \sigma \vDash \phi \end{array}$$

Note that this semantics differs from [BAF08] by the addition of formulas in the destructor definitions and the fact that messages are replaced by may-fail messages.

Thanks to the properties P1 and P2 satisfied by the destructors, we establish the following result:

Lemma 9.2. *Consider an equational presentation Σ. For all ground term evaluations D,*

- *there exists a ground may-fail message U such that $D \downarrow_\Sigma U$.*
- *for all ground may-fail messages U_1, U_2, if $D \downarrow_\Sigma U_1$ and $D \downarrow_\Sigma U_2$ then $U_1 =_\Sigma U_2$.*

Proof. We prove this result by induction on D:

Case $D = U$: In such a case, we have that $D \downarrow_\Sigma U$ hence the result trivially holds.

Case $D =$ eval $\mathsf{g}(D_1, \ldots, D_n)$: By inductive hypothesis, there exist $U_1, \ldots, U_n$ ground may-fail messages such that $D_i \downarrow_\Sigma U_i$ for all $i = 1 \ldots n$. By Property P1, we know that there exists $\mathsf{g}(U'_1, \ldots, U'_n) \to U' \mid\mid \phi$ in $\mathrm{def}_\Sigma(\mathsf{g})$ and a substitution σ such that $U'_i\sigma =_\Sigma U_i$ for $i = 1 \ldots n$ and $\sigma \vDash \phi$ hence $D \downarrow_\Sigma U'\sigma$. Moreover, since $fvars(U') \subseteq fvars(U'_1, \ldots, U'_n)$, we deduce that $U'\sigma$ is ground. Thus the first item is satisfied.

Let V_1, V_2 two may-fail messages such that $D \downarrow_\Sigma V_1$ and $D \downarrow_\Sigma V_2$. $D \downarrow_\Sigma V_1$ implies that there exist $U_1, \ldots, U_n$ ground may-fail messages such that $D_i \downarrow_\Sigma U_i$ for all $i = 1 \ldots n$ and $\mathsf{g}(U_1, \ldots, U_n) \to V_1$. Similarly, $D \downarrow_\Sigma V_2$ implies that there exist $W_1, \ldots, W_n$ ground fail-messages such that $D_i \downarrow_\Sigma W_i$ for all $i = 1 \ldots n$ and $\mathsf{g}(W_1, \ldots, W_n) \to V_2$. By our inductive hypothesis, we deduce that $W_i =_\Sigma U_i$ for all $i = 1 \ldots n$. Hence, by definition of the reduction of may-fail messages by a destructor, we deduce that $\mathsf{g}(U_1, \ldots, U_n) \to V_2$. But $\mathrm{def}_\Sigma(\mathsf{g})$ satisfies Property P2, thus we conclude that $V_1 =_\Sigma V_2$. □

In fact, given an equational presentation Σ' defined as in the original ProVerif, *i.e.* with partial destructors and no side condition, and given Σ the equational presentation adapted to our semantics thanks to Lemma 9.1, we would have: For all ground term evaluations D, for all messages M, $D\downarrow_{\Sigma'} M$ if, and only if, $D\downarrow_{\Sigma} M$.

Example 9.6. *Consider an equational presentation Σ with the equational theory $\mathsf{E}_{\mathsf{aenc}}$ and the destructor* sdec *described in Example 9.3. We have:*

— eval $\mathsf{sdec}(\mathsf{senc}(n, sk), sk)\downarrow_{\Sigma} n$

— eval $\mathsf{sdec}(\mathsf{senc}(n, sk'), sk)\downarrow_{\Sigma} \mathsf{fail}$.

— eval $\mathsf{sdec}(\mathsf{senc}(n, \mathsf{proj}_1(\langle sk, sk'\rangle)), sk)\downarrow_{\Sigma} n$

9.1.2.2 Processes

Similarly to the applied pi calculus, the semantics for processes in ProVerif is defined by a *structural equivalence*, denoted $\equiv$ and some *internal reductions*. However, both relations slightly differ from the applied pi calculus. The structural equivalence in ProVerif is the smallest equivalence relation on processes that is closed under α-conversion of bounded names and variables and that includes the following relations:

$$\begin{array}{ll} P \mid 0 \equiv P & P \equiv P \\ P \mid Q \equiv Q \mid P & Q \equiv P \;\Rightarrow\; P \equiv Q \\ (P \mid Q) \mid R \equiv P \mid (Q \mid R) & P \equiv Q, Q \equiv R \;\Rightarrow\; P \equiv R \\ \nu a.\nu b.P \equiv \nu b.\nu a.P & P \equiv Q \;\Rightarrow\; P \mid R \equiv Q \mid R \\ \nu a.(P \mid Q) \equiv P \mid \nu a.Q & P \equiv Q \;\Rightarrow\; \nu a.P \equiv \nu a.Q \\ \quad \text{if } a \notin \mathit{fnames}(P) & \end{array}$$

Note that similarly to the applied pi calculus, the structural equivalence is closed by application of evaluation contexts, and basic structural rules such as $A \mid 0 \equiv A$, associativity and commutativity of $|$, binding-operator-like behaviour of ν. However, this structural equivalence does not substitute equal terms modulo the equational theory and do not model the replication. Both properties are in fact modelled as internal reduction rules for processes (see below). In [BAF08], they indicate that this weakening of the structural equivalence was designed to simplify the proofs.

Figure 9.4 describes the internal reductions for processes ($\rightarrow_{\Sigma}$). This semantics is different from [BAF08] by the semantics of the rule (Red Fun 2) which previously corresponded to the case where the evaluation term D could not be reduced.

$$\begin{array}{ll} \mathsf{out}(N, M).Q \mid \mathsf{in}(N', x).P \;\rightarrow_{\Sigma}\; Q \mid P\{^{M}/_{x}\} & \\ \quad \text{if } N =_{\Sigma} N' & \text{(Red I/O)} \\ \mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q \rightarrow_{\Sigma} P\{^{M}/_{x}\} & \\ \quad \text{if } D\downarrow_{\Sigma} M & \text{(Red Fun 1)} \\ \mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q \rightarrow_{\Sigma} Q & \\ \quad \text{if } D\downarrow_{\Sigma} \mathsf{fail} & \text{(Red Fun 2)} \\ !\,P \;\rightarrow_{\Sigma}\; P \mid !\,P & \text{(Red Repl)} \\ P \;\rightarrow_{\Sigma}\; Q \;\Rightarrow\; P \mid R \;\rightarrow_{\Sigma}\; Q \mid R & \text{(Red Par)} \\ P \;\rightarrow_{\Sigma}\; Q \;\Rightarrow\; \nu a.P \;\rightarrow_{\Sigma}\; \nu a.Q & \text{(Red Res)} \\ P' \equiv P,\ P \;\rightarrow_{\Sigma}\; Q,\ Q \equiv Q' \;\Rightarrow\; P' \;\rightarrow_{\Sigma}\; Q' & \text{(Red } \equiv\text{)} \end{array}$$

Figure 9.4: Semantics for processes

Both relations $\equiv$ and $\rightarrow_{\Sigma}$ are defined only on closed processes. Furthermore, we denote $\rightarrow_{\Sigma}^{*}$ the reflexive and transitive closure of $\rightarrow_{\Sigma}$. At last, we denote $\rightarrow_{\Sigma}^{*}\equiv$ for its union with $\equiv$. When Σ is clear from the context, we abbreviate $\rightarrow_{\Sigma}$ and $\downarrow_{\Sigma}$ to $\rightarrow$ and $\downarrow$, respectively.

Example 9.7. *Coming back to Example 9.5, consider the process* $P \stackrel{def}{=} \mathsf{out}(c,n).\mathsf{in}(c,z) \mid \nu sk_a.\nu sk_b.B(sk_b,sk_a)$ *and the equational presentation* Σ *with the equational theory* $\mathsf{E}_{\mathsf{aenc}}$ *of Example 2.2. A possible reduction of* P *would be the following:*

$$\begin{array}{lll}
P & \to_\Sigma & \nu sk_a.\nu sk_b.\nu n_b.\big(\mathsf{out}(c,n).\mathsf{in}(c,z) \mid \mathsf{in}(c,y).B'\big) \qquad \textit{(Red} \equiv\textit{)}\\
& \to_\Sigma & \nu sk_a.\nu sk_b.\nu n_b.\big(\mathsf{in}(c,z) \mid B'\{^n/_y\}\big) \qquad \textit{(Red I/O and Res)}\\
& \to_\Sigma & \nu sk_a.\nu sk_b.\nu n_b.\big(\mathsf{in}(c,z) \mid B''\{^n/_y\}\{^u/_x\}\big) \qquad \textit{(Red Fun 1 and Res)}\\
& \to_\Sigma & \nu sk_a.\nu sk_b.\nu n_b.\big(\mathsf{in}(c,z) \mid B'''\{^n/_y\}\{^u/_x\}\{^v/_{xn_a}\}\big) \qquad \textit{(Red Fun 1 and Res)}\\
& \to_\Sigma & \nu sk_a.\nu sk_b.\nu n_b.\big(\mathsf{in}(c,z) \mid \mathsf{out}(c,\mathsf{aenc}(n_b,\mathsf{pk}(sk_b)))\big) \qquad \textit{(Red Fun 2 and Res)}\\
& \to_\Sigma & \nu sk_a.\nu sk_b.\nu n_b.\big(0 \mid 0\big) \qquad \textit{(Red I/O and Res)}
\end{array}$$

where $u = \mathsf{adec}(n, sk_b)$, $v = \mathsf{proj}_1(\mathsf{adec}(n, sk_b))$ *and* B', B'' *and* B''' *are defined as follows:*

$$B' \stackrel{def}{=} \mathsf{let}\ x = \mathsf{adec}(y, sk_b)\ \mathsf{in}\ B''\ \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(skb)))$$

$$B'' \stackrel{def}{=} \mathsf{let}\ xn_a = \mathsf{proj}_1(x)\ \mathsf{in}\ B'''\ \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(skb)))$$

$$\begin{array}{lll}
B''' & \stackrel{def}{=} & \mathsf{let}\ z = \mathsf{equals}(\mathsf{proj}_2(x), \mathsf{pk}(ska))\ \mathsf{in}\\
& & \quad \mathsf{out}(c, \mathsf{aenc}(\langle xn_a, \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a)))\\
& & \ \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(sk_b)))
\end{array}$$

However, if we now consider Σ' *as the equational presentation where* adec *is a destructor and* $\mathrm{def}_{\Sigma'}(\mathsf{adec})$ *is described by the following rules:*

- $\mathsf{adec}(\mathsf{aenc}(x, \mathsf{pk}(y)), y) \to x$ *(decryption succeeds)*
- $\mathsf{adec}(x, y) \to \mathsf{fail} \ ||\ \forall z.x \neq_{\mathsf{E}} \mathsf{aenc}(z, \mathsf{pk}(y))$ *(decryption fails, because* x *is not a ciphertext under the correct public key)*
- $\mathsf{adec}(\mathsf{fail}, u) \to \mathsf{fail}$, $\mathsf{adec}(u, \mathsf{fail}) \to \mathsf{fail}$ *(the arguments failed, the decryption also fails)*

then a possible reduction of P *(we omit the* eval adec*) would be the following:*

$$\begin{array}{lll}
P & \to_{\Sigma'} & \nu sk_a.\nu sk_b.\nu n_b.\big(\mathsf{out}(c,n).\mathsf{in}(c,z) \mid \mathsf{in}(c,y).B'\big) \qquad \textit{(Red} \equiv\textit{)}\\
& \to_{\Sigma'} & \nu sk_a.\nu sk_b.\nu n_b.\big(\mathsf{in}(c,z) \mid B'\{^n/_y\}\big) \qquad \textit{(Red I/O and Res)}\\
& \to_{\Sigma'} & \nu sk_a.\nu sk_b.\nu n_b.\big(\mathsf{in}(c,z) \mid \mathsf{out}(c,\mathsf{aenc}(n_b,\mathsf{pk}(sk_b)))\big) \qquad \textit{(Red Fun 2 and Res)}\\
& \to_{\Sigma'} & \nu sk_a.\nu sk_b.\nu n_b.\big(0 \mid 0\big) \qquad \textit{(Red I/O and Res)}
\end{array}$$

Note that while in the first case, we have $\mathsf{adec}(n, sk_b)\downarrow_\Sigma \mathsf{adec}(n, sk_b)$ *since* adec *is a constructor in* Σ*, we have in the second case* $\mathsf{adec}(n, sk_b)\downarrow_{\Sigma'} \mathsf{fail}$ *due to the definition of* $\mathrm{def}_{\Sigma'}(\mathsf{adec})$*. Hence the application of the internal reduction rule (Red Fun 2). In the first case, the application of the internal reduction rule (Ref Fun 2) is due to the destructor* equals *and more specifically to the fact that* $\mathsf{equals}(\mathsf{proj}_2(\mathsf{adec}(n, sk_b)), \mathsf{pk}(sk_a))\downarrow_\Sigma \mathsf{fail}$.

9.2 Using biprocesses to prove observational equivalence

One of the purposes of ProVerif is to prove the observational equivalence between processes. In Chapter 3, we provide a definition for such equivalence (see Definition 3.5) in the applied pi calculus. While the syntax and semantic of ProVerif are different from the applied pi calculus, the definition of observational equivalence for ProVerif is almost identical to Definition 3.5:

Definition 9.1. *Given an equational presentation* Σ*, the process* P*, built on* Σ*,* emits on M *(*$P \Downarrow M$*) if and only if* $P \to^*_\Sigma \equiv C[\mathsf{out}(M', N).R]$ *for some evaluation context* C *that does not bind* $fnames(M)$ *and* $M =_\Sigma M'$*.*

The observational equivalence*, denoted* $\approx$ *is the largest symmetric relation* $\mathcal{R}$ *between closed processes with the same domain such that* $P\ \mathcal{R}\ Q$ *implies:*

1. *if* $P \Downarrow M$*, then* $Q \Downarrow M$*;*

$$\mathsf{out}(N, M).Q \mid \mathsf{in}(N', x).P \;\rightarrow\; Q \mid P\{x \mapsto M\} \qquad \text{(Red I/O)}$$
$$\text{if } \mathsf{fst}(N) =_\Sigma \mathsf{fst}(N') \text{ and } \mathsf{snd}(N) =_\Sigma \mathsf{snd}(N')$$
$$\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q \rightarrow P\{x \mapsto \mathsf{diff}[M_1, M_2]\} \qquad \text{(Red Fun 1)}$$
$$\text{if } \mathsf{fst}(D) \downarrow_\Sigma M_1,\ \mathsf{snd}(D) \downarrow_\Sigma M_2$$
$$\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q \rightarrow Q \qquad \text{(Red Fun 2)}$$
$$\text{if } \mathsf{fst}(D) \downarrow_\Sigma \mathsf{fail} \text{ and } \mathsf{snd}(D) \downarrow_\Sigma \mathsf{fail}$$

Figure 9.5: Generalized rules for biprocesses

2. *if* $P \rightarrow_\Sigma^* P'$, *then* $Q \rightarrow_\Sigma^* Q'$ *and* $P' \mathcal{R} Q'$ *for some* Q';
3. $C[P] \mathcal{R} C[Q]$ *for all closing evaluation contexts* C.

As in the applied pi calculus, the same difficulties arise when deciding the observational equivalence between two processes. One of the most difficult parts directly comes from the second item of Definition 9.1. Indeed, this condition indicates that each reduction of a process has to be matched in the second process. However, we consider a process algebra with replication, hence there are usually an infinite number of candidates for this mapping.

9.2.1 Biprocesses

To solve this problem, [BAF08] introduces a calculus that represents pairs of processes, called *biprocesses*, that have the same structure and differ only by the terms and term evaluations that they contain. The grammar for the calculus is a simple extension of the grammar of Figures 9.1, 9.2 and 9.3, with additional cases so that $\mathsf{diff}[M, M']$ is a term and $\mathsf{diff}[D, D']$ is a term evaluation.

Given a biprocess P, we define two processes $\mathsf{fst}(P)$ and $\mathsf{snd}(P)$, as follows: $\mathsf{fst}(P)$ is obtained by replacing all occurrences of $\mathsf{diff}[M, M']$ with M and $\mathsf{diff}[D, D']$ with D in P, and similarly, $\mathsf{snd}(P)$ is obtained by replacing $\mathsf{diff}[M, M']$ with M' and $\mathsf{diff}[D, D']$ with D' in P. We define $\mathsf{fst}(D)$, $\mathsf{fst}(M)$, $\mathsf{snd}(D)$, and $\mathsf{snd}(M)$ similarly. A process or context is said to be *plain* when it does not contain diff.

Definition 9.2. *Let* P *be a closed biprocess. We say that* P satisfies observational equivalence *when* $\mathsf{fst}(P) \approx \mathsf{snd}(P)$.

Example 9.8. *Coming back to the private authentication protocol detailed in Example 9.5, we want to verify the anonymity of the participant* A. *In Section 3.3, we saw that the anonymity property is modelled by an observational equivalence between two instances of the protocol: one where* B *is talking to* A *and the other where* B *is talking to* A', *which is modelled, in the* ProVerif *calculus, as follows:*

$$\nu sk_a.\nu sk_{a'}.\nu sk_b.\mathsf{out}(c, \mathsf{pk}(sk_a)).\mathsf{out}(c, \mathsf{pk}(sk_{a'})), \mathsf{out}(c, \mathsf{pk}(sk_b)).System(sk_a, sk_b)$$
$$\approx$$
$$\nu sk_a.\nu sk_{a'}.\nu sk_b.\mathsf{out}(c, \mathsf{pk}(sk_a)).\mathsf{out}(c, \mathsf{pk}(sk_{a'})), \mathsf{out}(c, \mathsf{pk}(sk_b)).System(sk_{a'}, sk_b)$$

We already know that this equivalence holds for the trace equivalence, but in fact it also holds for the observational equivalence. To prove this equivalence using ProVerif, *we first have to transform this equivalence into a biprocess. This is easily done since only the private keys* sk_a *and* $sk_{a'}$ *change between the two processes. Hence, we define the biprocess* P_0 *as follows:*

$$\nu sk_a.\nu sk_{a'}.\nu sk_b.\mathsf{out}(c, \mathsf{pk}(sk_a)).\mathsf{out}(c, \mathsf{pk}(sk_{a'})).\mathsf{out}(c, \mathsf{pk}(sk_b)).System(\mathsf{diff}[sk_a, sk_{a'}], sk_b)$$

Note that $\mathsf{fst}(P_0)$ *and* $\mathsf{snd}(P_0)$ *correspond to the two protocols of the equivalence.*

The semantics for biprocesses is defined as in Figure 9.4 with generalized rules (Red I/O), (Red Fun 1), and (Red Fun 2) given in Figure 9.5. Once again, the difference with [BAF08] is that, in rule (Red Fun 1), we require that $\mathsf{fst}(D)$ and $\mathsf{snd}(D)$ reduce to a message and, in rule (Red Fun

2), $\mathsf{fst}(D)$ and $\mathsf{snd}(D)$ both reduce to fail. In contrast, in [BAF08], in rule (Red Fun 1), $\mathsf{fst}(D)$ and $\mathsf{snd}(D)$ reduced to any message and, in rule (Red Fun 2), $\mathsf{fst}(D)$ and $\mathsf{snd}(D)$ failed to reduce.

The semantics of biprocesses is such that a biprocess reduces if and only if both sides of the biprocess reduce in the same way: a communication succeeds on both sides; a term evaluation succeeds on both sides or fails on both sides. When the two sides of the biprocess reduce in different ways, the biprocess blocks. The following lemma shows that, when both sides of a biprocess always reduce in the same way, then that biprocess satisfies observational equivalence. Its proof can be found in Appendix D.1).

Lemma 9.3. *Let P_0 be a closed biprocess. Suppose that, for all plain evaluation contexts C, all evaluation contexts C', and all reductions $C[P_0] \to^* P$,*

1. *if $P \equiv C'[\mathsf{out}(N, M).Q \mid \mathsf{in}(N', x).R]$, then $\mathsf{fst}(N) =_\Sigma \mathsf{fst}(N')$ if, and only if, $\mathsf{snd}(N) =_\Sigma \mathsf{snd}(N')$; and*
2. *if $P \equiv C'[\mathsf{let}\ x = D\ \mathsf{in}\ Q\ \mathsf{else}\ R]$, then $\mathsf{fst}(D)\downarrow_\Sigma \mathsf{fail}$ if, and only if, $\mathsf{snd}(D)\downarrow_\Sigma \mathsf{fail}$.*

Then P_0 satisfies observational equivalence.

Intuitively, the semantics for biprocesses forces that each reduction of a process has to be matched by the same reduction in the second process. Hence, verifying the second item of Definition 9.1 becomes less problematic since we reduce to one the number of possible candidates.

9.2.2 From equational theories to rewrite rules

Equational theories are very useful when it comes to theoretical results. However, for a practical algorithm, it is easier to work with rewrite rules rather than with equational theories. Hence in [BAF08], the equational theory is transformed into a set of rewrite rules. Typically, for an equational presentation Σ with its associated equational theory, a new equational presentation Σ' with the same function symbols and the empty equational theory is generated. Moreover, Σ' models Σ. In this section, we adapt the definitions used in [BAF08] to our formalism.

9.2.2.1 Generation of Σ'

For an equational presentation Σ' (with the empty equational theory), we define evaluation on open terms as a relation $D \downarrow'_{\Sigma'} (U, \sigma, \phi)$, where σ collects instantiations of D obtained by unification and ϕ collects the side conditions of destructor applications:

$$U \downarrow'_{\Sigma'} (U, \emptyset, \top)$$

$$\begin{array}{l}\text{eval } \mathsf{h}(D_1, \ldots, D_n) \downarrow'_{\Sigma'} (V\sigma_u, \sigma'\sigma_u, \phi'\sigma_u \wedge \phi\sigma_u)\\ \quad \text{if } (D_1, \ldots, D_n) \downarrow'_{\Sigma'} ((U_1, \ldots, U_n), \sigma', \phi'),\\ \quad \mathsf{h}(V_1, \ldots, V_n) \to V \mid\mid \phi \in \mathrm{def}_{\Sigma'}(\mathsf{h}) \text{ and}\\ \quad \sigma_u \text{ is a most general unifier of } (U_1, V_1), \ldots, (U_n, V_n)\end{array}$$

$$\begin{array}{l}(D_1, \ldots, D_n) \downarrow'_{\Sigma'} ((U_1\sigma_n, \ldots, U_{n-1}\sigma_n, U_n), \sigma\sigma_n, \phi\sigma_n \wedge \phi_n)\\ \quad \text{if } (D_1, \ldots, D_{n-1}) \downarrow'_{\Sigma'} ((U_1, \ldots, U_{n-1}), \sigma, \phi) \text{ and } D_n\sigma \downarrow' (U_n, \sigma_n, \phi_n)\end{array}$$

The most general unifier of may-fail messages is computed similarly to the most general unifier of messages, even though specific cases hold due to may-fail variables and message variables: there is no unifier of M and fail, for any message M (including variables x, because these variables can be instantiated only by messages); the most general unifier of u and U is $\{u \mapsto U\}$; the most general unifier of fail and fail is the identity; finally, the most general unifier of M and M' is computed as usual.

Example 9.9. *Consider Σ' the equational presentation with the empty equational theory and containing the destructor* sdec *defined in Example 9.3. Let D and D' be two term evaluations such that $D' = \mathrm{eval}\ \mathsf{sdec}(x, sk)$ and $D = \mathrm{eval}\ \mathsf{sdec}(D', sk')$. We have:*

1. $D' \downarrow'_{\Sigma'} (y, \{x \mapsto \mathsf{senc}(y, sk)\}, \top)$

2. $D' \downarrow'_{\Sigma'} (\mathsf{fail}, \emptyset, \forall y.x \neq_\Sigma \mathsf{senc}(y, sk))$
3. $D \downarrow'_{\Sigma'} (z, \{x \mapsto \mathsf{senc}(\mathsf{senc}(z, sk'), sk)\}, \top)$
4. $D \downarrow'_{\Sigma'} (\mathsf{fail}, \{x \mapsto \mathsf{senc}(y, sk)\}, \forall z.y \neq_\Sigma \mathsf{senc}(z, sk'))$.
5. $D \downarrow'_{\Sigma'} (\mathsf{fail}, \emptyset, \forall y.x \neq_\Sigma \mathsf{senc}(y, sk))$

Note that the evaluations 3 and 4 of D are obtained from the evaluation 1 of D'. On the other hand, the evaluation 5 of D is obtained by the evaluation 2 of D'.

We let addeval$(U_1, \ldots, U_n)$ be the tuple of term evaluations obtained by adding eval before each function symbol of $U_1, \ldots, U_n$. Using these definitions, we recall the definition of an equational presentation Σ' with an empty equational theory modelling another equational presentation Σ with equations:

We consider an auxiliary rewriting system on terms, $\mathcal{S}$, that defines partial normal forms. The rules of $\mathcal{S}$ do not contain names and do not have a single variable on the left-hand side. We say that a term is irreducible by $\mathcal{S}$ when none of the rewrite rules of $\mathcal{S}$ applies to it; we say that the set of terms $\mathcal{M}$ is in normal form relatively to $\mathcal{S}$ and Σ, and write $\mathrm{nf}_{\mathcal{S},\Sigma}(\mathcal{M})$, if and only if all terms of $\mathcal{M}$ are irreducible by $\mathcal{S}$ and, for all subterms N_1 and N_2 of terms of $\mathcal{M}$, if $N_1 =_\Sigma N_2$ then $N_1 = N_2$. Typically, while a term might have several partial normal forms, any equal (sub)terms in $\mathcal{M}$ modulo the equational theory are in the same normal form in $\mathrm{nf}_{\mathcal{S},\Sigma}(\mathcal{M})$. We extend the definition of $\mathrm{nf}_{\mathcal{S},\Sigma}(\cdot)$ to sets of processes: $\mathrm{nf}_{\mathcal{S},\Sigma}(\mathcal{P})$ if and only if the set of terms that appear in processes in $\mathcal{P}$ is in normal form.

Definition 9.3. *Let Σ and Σ' be two equational presentations on the same function symbols. We say that Σ' models Σ if and only if*

1. *The equational theory of Σ' is syntactic equality: $M =_{\Sigma'} N$ if and only if $M = N$.*
2. *The constructors of Σ' are the constructors of Σ; their definition $\mathrm{def}_{\Sigma'}(f)$ contains the rule $f(x_1, \ldots, x_n) \to f(x_1, \ldots, x_n)$, the rewrite rules corresponding to possible failure,* i.e. *for all $i \in \{1 \ldots n\}$, $\mathsf{f}(u_1, \ldots, u_{i-1}, \mathsf{fail}, u_{i+1}, \ldots, u_n) \to \mathsf{fail}$, and perhaps other rules such that there exists a rewriting system $\mathcal{S}$ on terms that satisfies the following properties:*
   - *S1. If $M \to N$ is in $\mathcal{S}$, then $M =_\Sigma N$.*
   - *S2. If $\mathrm{nf}_{\mathcal{S},\Sigma}(\mathcal{M})$, then for any term M there exists M' such that $M' =_\Sigma M$ and $\mathrm{nf}_{\mathcal{S},\Sigma}(\mathcal{M} \cup \{M'\})$.*
   - *S3. If $f(N_1, \ldots, N_n) \to N \mid\mid \phi$ is in $\mathrm{def}_{\Sigma'}(f)$, then $f(N_1, \ldots, N_n) =_\Sigma N$ and $\phi = \top$*
   - *S4. If $f(M_1, \ldots, M_n) =_\Sigma M$ and $\mathrm{nf}_{\mathcal{S},\Sigma}(\{M_1, \ldots, M_n, M\})$, then there exist σ and $f(N_1, \ldots, N_n) \to N$ in $\mathrm{def}_{\Sigma'}(f)$ such that $M = N\sigma$ and $M_i = N_i\sigma$ for all $i \in \{1, \ldots, n\}$.*
3. *The destructors of Σ' are the destructors of Σ, with a rule $\mathsf{g}(U'_1, \ldots, U'_n) \to U' \mid\mid \phi\sigma \wedge \phi'$ in $\mathrm{def}_{\Sigma'}(g)$ for each $\mathsf{g}(U_1, \ldots, U_n) \to U \mid\mid \phi$ in $\mathrm{def}_\Sigma(g)$ and each $\mathrm{addeval}(U_1, \ldots, U_n, U) \downarrow'_{\Sigma'} ((U'_1, \ldots, U'_n, U'), \sigma, \phi')$.*

Stress that, in Item 3, since $U_1, \ldots, U_n$ do not contain any destructor and the side conditions of constructors are always $\top$, ϕ' is necessarily $\top$. The main difference between this definition and the definition of [BAF08] is in Condition 3, since our destructors can now have a formula as side condition. The constructors of Σ' can be computed by the algorithms given in [BAF08], and Item 3 shows how to compute the destructors of Σ'.

Note that, in Σ', the side conditions of destructors still rely on the equational theory of Σ. Moreover, even if the semantics of the function symbols of Σ' are all defined by rewrite rules, we still consider the constructors of Σ as the constructors in Σ'.

Example 9.10. *Consider the equational presentation Σ that has the constructors* senc *and* sdec *with the equations*

$$\mathsf{sdec}(\mathsf{senc}(x, y), y) = x \qquad \mathsf{senc}(\mathsf{sdec}(x, y), y) = x$$

In Σ', we adopt the rewrite rules:

$$\begin{array}{ll} \mathsf{sdec}(x,y) \to \mathsf{sdec}(x,y) & \mathsf{senc}(x,y) \to \mathsf{senc}(x,y) \\ \mathsf{sdec}(\mathsf{senc}(x,y),y) \to x & \mathsf{senc}(\mathsf{sdec}(x,y),y) \to x \\ \mathsf{sdec}(\mathsf{fail},u) \to \mathsf{fail} & \mathsf{senc}(\mathsf{fail},u) \to \mathsf{fail} \\ \mathsf{sdec}(u,\mathsf{fail}) \to \mathsf{fail} & \mathsf{senc}(u,\mathsf{fail}) \to \mathsf{fail} \end{array}$$

We have that Σ' models Σ for the rewriting system $\mathcal{S}$ with rules $\mathsf{sdec}(\mathsf{senc}(x,y),y) \to x$ and $\mathsf{senc}(\mathsf{sdec}(x,y),y) \to x$, and a single normal form for every term.

9.2.2.2 Observation equivalence w.r.t. Σ'

From this point on, we assume that Σ' models Σ. We say that a biprocess P_0 is unevaluated when every term in P_0 is either a message variable or $\mathsf{diff}[a,a]$ for some name a. Hence, every function symbol in P_0 must be in a term evaluation and prefixed by eval. Moreover, we extend equality modulo the equational theory in Σ from terms to biprocesses and term evaluations: $P =_\Sigma P'$ if, and only if, P' can be obtained from P by replacing some of its subterms M (not containing diff or eval) with subterms equal modulo Σ. We define $D =_\Sigma D'$ similarly. We can give two results that show the correspondence between Σ and Σ' in the derivation of biprocesses. The proofs of these results are adapted from [BAF08, Lemmas 1,2].

Lemma 9.4. *Let P_0 be a closed, unevaluated biprocess. If $P_0 \to^*_\Sigma P'_0$, $P'_0 =_\Sigma P'$, and $\mathrm{nf}_{\mathcal{S},\Sigma}(\{P'\})$, then $P_0 \to^*_{\Sigma'} P'$. Conversely, if $P_0 \to^*_{\Sigma'} P'$ then there exists P'_0 such that $P'_0 =_\Sigma P'$ and $P_0 \to^*_\Sigma P'_0$.*

This lemma gives an operational correspondence between $\to_\Sigma$ and $\to_{\Sigma'}$. Using Lemma 9.4, we obtain:

Lemma 9.5. *A closed biprocess P_0 satisfies the conditions of Lemma 9.3 if and only if, for all plain evaluation contexts C, all evaluation contexts C', and all reductions* $\mathrm{unevaluated}(C[P_0]) \to^*_{\Sigma'} P$, *we have*

1. *if $P \equiv C'[\mathsf{out}(N,M).Q \mid \mathsf{in}(N',x).R]$ and $\mathsf{fst}(N) = \mathsf{fst}(N')$, then $\mathsf{snd}(N) =_\Sigma \mathsf{snd}(N')$,*
2. *if $P \equiv C'[\mathsf{let}\ x = D\ \mathsf{in}\ Q\ \mathsf{else}\ R]$ and $\mathsf{fst}(D)\downarrow_{\Sigma'} M_1$ for some M_1, then $\mathsf{snd}(D)\downarrow_\Sigma M_2$ for some M_2,*

as well as the symmetric properties where we swap fst *and* snd.

In [BAF08], a special transition $P \to_{\Sigma',\Sigma} P'$ was defined as $P \to_\Sigma P'$ except that signature Σ' was used for reduction rules (Red I/O) and (Red Fun 1)—signature Σ was still used for (Red Fun 2). In our case, we always use the signature Σ'. The difference is due to the fact that the reference to Σ is now included in the side conditions of the definition of destructors in Σ'.

9.3 Clause generation

In [BAF08], observational equivalence is verified by translating the considered biprocess into a set of Horn clauses, and using a resolution algorithm on these clauses. In this section, we show how we adapt the generation of the clauses to our new formalism.

9.3.1 Patterns and facts

In the clauses, the messages are represented by patterns, with the following grammar:

$p ::=$	patterns
x, y, z, i	variable
$f(p_1, \ldots, p_n)$	constructor application
$a[p_1, \ldots, p_n]$	name

$mp ::=$	may-fail patterns
p	patterns
u, v	may-fail variable
fail	failure

The patterns p are the same as in [BAF08]. The variable i represents a session identifier for each replication of a process. A pattern $a[p_1, \ldots, p_n]$ is assigned to each name of a process P. The arguments $p_1, \ldots, p_n$ allow one to model that a fresh name a is created at execution of $!\,a$. For example, in the process $!\,\mathsf{in}(c', x).\nu a.P$, each name created by νa is represented by $a[i, x]$ where i is the session identifier for the replication and x is the message received as input in $\mathsf{in}(c', x)$. Hence, the name a is represented as a function of i and x. In two different sessions, (i, x) takes two different values, so the two created instances of a ($a[i, x]$) are different.

Since our formalism introduced may-fail messages to describe possible failure of a destructor, we also define *may-fail patterns* to represent the failure in clauses. Similarly to messages and may-fail messages, a may-fail variable u can be instantiated by a pattern or fail, whereas a variable x cannot be instantiated by fail.

Clauses are built from the following predicates:

$F ::=$	facts
$\mathrm{att}'(mp, mp')$	attacker knowledge
$\mathrm{msg}'(p_1, p_2, p_1', p_2')$	output message p_2 on p_1 (resp. p_2' on p_1')
$\mathrm{input}'(p, p')$	input on p (resp. p')
$\mathrm{formula}(\bigwedge_i \forall \tilde{z}_i.p_i \neq_\Sigma p_i')$	formula
bad	bad

Once again, the facts are similar to the ones defined in [BAF08]. Intuitively, $\mathrm{att}'(mp, mp')$ means that the attacker may obtain up in $\mathsf{fst}(P)$ and up' in $\mathsf{snd}(P)$ by the same operations; the fact $\mathrm{msg}'(p_1, p_2, p_1', p_2')$ means that message p_2 may be output on channel p_1 by the process $\mathsf{fst}(P)$ while p_2' may be output on channel p_1' by the process $\mathsf{snd}(P)$ after the same reductions; $\mathrm{input}'(p, p')$ means that an input is possible on channel p in $\mathsf{fst}(P)$ and on channel p' in $\mathsf{snd}(P)$. Note that both facts msg' and input' contain only patterns and not may-fail patterns. Hence channels and terms sent are necessarily messages and so cannot be fail.

The fact $\mathrm{formula}(\phi)$ means that ϕ has to be satisfied. In [BAF08], a fact $\mathrm{nounif}(p, p')$ was used instead: $\mathrm{nounif}(p, p')$ is equivalent to $\mathrm{formula}(\forall \tilde{z}.p \neq_\Sigma p')$ where $\tilde{z}$ are symbols in p and p' that belong to a special set of "general variables". Hence, our formulas can also be encoded as conjunctions of nounif facts. We prefer using formulas, because their meaning is clearer from just reading the formula.

At last, bad serves in detecting violations of observational equivalence: when bad is not derivable, we have observational equivalence.

9.3.2 Clauses for the attacker

The following clauses represent the capabilities of the attacker:

$$\text{For each } a \in \mathit{fnames}(P_0),\ \mathrm{att}'(a[\,], a[\,]) \tag{Rinit}$$

$$\text{For some } b \text{ that does not occur in } P_0,\ \mathrm{att}'(b[x], b[x]) \tag{Rn}$$

$$\mathrm{att}'(\mathsf{fail}, \mathsf{fail}) \tag{Rfail}$$

$$\begin{array}{l}\text{For each function } \mathsf{h}, \text{ for each pair of rewrite rules}\\ \quad \mathsf{h}(U_1, \ldots, U_n) \to U \mid\mid \phi \text{ and } \mathsf{h}(U_1', \ldots, U_n') \to U' \mid\mid \phi'\\ \quad \text{in } \mathrm{def}_{\Sigma'}(\mathsf{h}) \text{ (after renaming of variables)},\\ \quad \mathrm{att}'(U_1, U_1') \wedge \ldots \wedge \mathrm{att}'(U_n, U_n') \wedge \mathrm{formula}(\phi \wedge \phi') \to \mathrm{att}'(U, U')\end{array} \tag{Rf}$$

$$\mathrm{msg}'(x, y, x', y') \wedge \mathrm{att}'(x, x') \to \mathrm{att}'(y, y') \tag{Rl}$$

$$\mathrm{att}'(x,x') \wedge \mathrm{att}'(y,y') \to \mathrm{msg}'(x,y,x',y') \quad \text{(Rs)}$$
$$\mathrm{att}'(x,x') \wedge \to \mathrm{input}'(x,x') \quad \text{(Ri)}$$
$$\mathrm{input}'(x,x') \wedge \mathrm{msg}'(x,z,y',z') \wedge \mathrm{formula}(x' \neq_\Sigma y') \to \mathrm{bad} \quad \text{(Rcom)}$$
$$\mathrm{att}'(x,\mathsf{fail}) \to \mathrm{bad} \quad \text{(Rfailure)}$$

plus symmetric clauses (Rcom$'$) and (Rfailure$'$) obtained from (Rcom) and (Rfailure) by swapping the first and second arguments of att$'$ and input$'$, and the first and third arguments of msg$'$.

The previous clauses are directly inspired from the ones in [BAF08] and correspond to the same semantics adapted to our formalism. Clauses (Rcom) and (Rcom$'$) detect when a communication can occur in one variant of the biprocess and not in the other which yields the non-equivalence due to Condition 1 of Lemma 9.5.

Clause (Rfail) indicates that the attacker knows fail. Clauses (Rf) apply a constructor or a destructor on the attacker's knowledge, given the definition of the destructor in $\mathrm{def}_{\Sigma'}(\mathsf{h})$. Since our destructors may return fail, these clauses and clauses (Rfailure), (Rfailure$'$) detect when a destructor succeeds in one variant of the biprocess and not in the other. In [BAF08], since the destructors were partial, the checking of success or failure of a destructor was written directly into clauses which correspond to the composition of our clauses (Rfailure) and (Rf) (respectively, (Rfailure$'$) and (Rf)).

Stress that, in clause (Rfailure), x is a message variable and so x cannot be instantiated by fail. Similarly, in (Rcom), Rs, Rl and Rf, x, x', y, y' are message variables and so they cannot be instantiated by fail.

9.3.3 Clauses for the protocol

The translation $[\![P]\!]\rho s H$ of a biprocess P is a set of clauses, where ρ is an environment that associates a pair of patterns with each name and variable, s is a sequence of patterns, and H is a sequence of facts. The empty sequence is written $\emptyset$; the concatenation of a pattern p to the sequence s is written s, p; the concatenation of a fact F to the sequence H is written $H \wedge F$.

Intuitively, H represents the hypothesis of the clauses, ρ represents the names and variables that are already associated with a pattern, and s represents the current values of session identifiers and inputs.

When ρ associates a pair of patterns with each name and variable, and f is a constructor, we extend ρ as a substitution by $\rho(\mathsf{f}(M_1,\ldots,M_n)) = (\mathsf{f}(p_1,\ldots,p_n), \mathsf{f}(p'_1,\ldots,p'_n))$ where $\rho(M_i) = (p_i, p'_i)$ for all $i \in \{1,\ldots,n\}$. We denote by $\rho(M)_1$ and $\rho(M)_2$ the components of the pair $\rho(M)$. We let $\rho(\mathsf{diff}[M, M']) = (\rho(M)_1, \rho(M')_2)$.

We define $[\![P]\!]\rho s H$ as follows:

$$[\![0]\!]\rho s H = \emptyset$$
$$[\![!\,P]\!]\rho s H = [\![P]\!]\rho(s,i)H, \text{where } i \text{ is a fresh variable}$$
$$[\![P \mid Q]\!]\rho s H = [\![P]\!]\rho s H \cup [\![Q]\!]\rho s H$$
$$[\![\nu a.P]\!]\rho s H = [\![P]\!](\rho[a \mapsto (a[s], a[s])])s H$$
$$[\![\mathsf{in}(M,x).P]\!]\rho s H =$$
$$\quad [\![P]\!](\rho[x \mapsto (x', x'')])(s, x', x'')(H \wedge \mathrm{msg}'(\rho(M)_1, x', \rho(M)_2, x''))$$
$$\quad \cup \{H \to \mathrm{input}'(\rho(M)_1, \rho(M)_2)\}$$
$$\quad \text{where } x' \text{ and } x'' \text{ are fresh variables}$$
$$[\![\mathsf{out}(M,N).P]\!]\rho s H = [\![P]\!]\rho s H \cup \{H \to \mathrm{msg}'(\rho(M)_1, \rho(N)_1, \rho(M)_2, \rho(N)_2)\}$$

$$
\begin{aligned}
&[\![\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q]\!]\rho s H = \\
&\quad \bigcup \{ [\![P]\!]((\rho\sigma)[x \mapsto (p, p')])(s\sigma, p, p')(H\sigma \wedge \mathrm{formula}(\phi)) \\
&\qquad \mid (\rho(D)_1, \rho(D)_2) \downarrow' ((p, p'), \sigma, \phi) \} \\
&\quad \cup \bigcup \{ [\![Q]\!](\rho\sigma)(s\sigma)(H\sigma \wedge \mathrm{formula}(\phi)) \mid (\rho(D)_1, \rho(D)_2) \downarrow' ((\mathsf{fail}, \mathsf{fail}), \sigma, \phi) \} \\
&\quad \cup \{ H\sigma \wedge \mathrm{formula}(\phi) \to \mathrm{bad} \mid (\rho(D)_1, \rho(D)_2) \downarrow' ((p, \mathsf{fail}), \sigma, \phi) \} \\
&\quad \cup \{ H\sigma \wedge \mathrm{formula}(\phi) \to \mathrm{bad} \mid (\rho(D)_1, \rho(D)_2) \downarrow' ((\mathsf{fail}, p'), \sigma, \phi) \}
\end{aligned}
$$

Comparing with the clauses described in [BAF08], we can recognise the same kind of generated clauses: The process P is translated when both $\rho(D)_1$ and $\rho(D)_2$ succeed; the process Q is translated when both fail; and at last clauses deriving bad are generated when one of $\rho(D)_1$, $\rho(D)_2$ succeeds and the other fails. Since may-fail variables do not occur in D, we can show by induction on the computation of $\downarrow'$ that, when $(\rho(D)_1, \rho(D)_2) \downarrow' ((mp_1, mp_2), \sigma, \phi)$, mp_1 and mp_2 are either fail or a pattern, but cannot be an may-fail variable, so our definition of $[\![\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q]\!]\rho s H$ handles all cases.

However, one of the differences occurs for the translation of $[\![\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q]\!]\rho s H$. It is once again due to the fact that in our formalism, in case of failure, term evaluations reduce into fail, whereas in [BAF08], they did not reduce at all. For example, the clauses representing the cases where a term evaluation D fails on the left side and succeeds on the right side was generated by considering a formula representing the negation of all possible terms reduced by the term evaluation on the left side. In our semantics, this formula is generated thanks to $(\rho(D)_1, \rho(D)_2) \downarrow' ((\mathsf{fail}, p), \sigma, \phi)$.

A more meaningful difference is the presence of only one sequence of patterns s. Indeed, in [BAF08], the translation of a process was written $[\![P]\!]\rho s s' H$ where s and s' were sequences of patterns. Typically, s was reserved for the part of the clauses translating $\mathsf{fst}(P)$ whereas s' was reserved for the part of the clauses translating $\mathsf{snd}(P)$. For instance, in [BAF08], we have:

$$
\begin{aligned}
&[\![\nu a.P]\!]\rho s s' H = [\![P]\!](\rho[a \mapsto (a[s], a[s'])])s s' H \\
&[\![\mathsf{in}(M, x).P]\!]\rho s s' H = \\
&\qquad [\![P]\!](\rho[x \mapsto (x', x'')])(s, x')(s, x'')(H \wedge \mathrm{msg}'(\rho(M)_1, x', \rho(M)_2, x'')) \\
&\quad \cup \{ H \to \mathrm{input}'(\rho(M)_1, \rho(M)_2) \} \\
&\quad \text{where } x' \text{ and } x'' \text{ are fresh variables}
\end{aligned}
$$

As a result, in [BAF08], the arguments of the name a in the first component were only the arguments coming from $\mathsf{fst}(P)$ and the arguments of a in the second component were the arguments coming from $\mathsf{snd}(P)$. In this paper, the arguments of a are the same in both components, and include the arguments coming from both $\mathsf{fst}(P)$ and $\mathsf{snd}(P)$. The difference is subtle, but Example 9.11 below gives an example of a biprocess that satisfies equivalence, and such that the clauses generated in [BAF08] yield a false attack which is avoided by our new clauses.

Example 9.11. *Consider the signature* $\Sigma = \{\mathsf{mac}\}$ *consisting of a MAC of arity 2 with the empty equational theory and the following biprocess:*

$$P = \nu d.!\, \nu k.!\, \nu k'.\mathsf{out}(d, \mathsf{diff}[k, k']) \mid !\, \mathsf{in}(d, x).\nu a.\mathsf{out}(c, \mathsf{mac}(a, x))$$

In this biprocess, the keys k and k' are generated and sent over a private channel d. After receiving these keys on d, they are used to mac a fresh nonce a which is send on a public channel c. The main difference between $\mathsf{fst}(P)$ and $\mathsf{snd}(P)$ is that in $\mathsf{fst}(P)$, an instance of the key k can be sent several times on d whereas in $\mathsf{snd}(P)$, an instance of the key k' is sent only once. However, since the instances of keys k and k' are never revealed to the attacker and the nonces a are freshly generated for each MAC sent on the public channel c, the biprocess P satisfies equivalence.

According to [BAF08], the translation of P will generate the following clauses:

1. $\mathrm{msg}'(d[\,], k[i], d[\,], k'[i, j])$,

2. $\text{input}'(d[\,], d[\,])$, *and*

3. $\text{msg}'(d[\,], x', d[\,], x'') \rightarrow \text{msg}'(c[\,], \mathsf{mac}(a[\ell, x'], x'), c[\,], \mathsf{mac}(a[\ell, x''], x''))$

where i, j, ℓ correspond to the session identifiers of replications.

But using two instances of j, denoted j_1 and j_2, we obtain from Clause 1 the two following facts $\text{msg}'(d[\,], k[i], d[\,], k'[i, j_1])$ *and* $\text{msg}'(d[\,], k[i], d[\,], k'[i, j_2])$. *Hence, using Clause 3, we derive*

— $\text{msg}'(c[\,], \mathsf{mac}(a[\ell, k[i]], k[i]), c[\,], \mathsf{mac}(a[\ell, k'[i, j_1]], k'[i, j_1]))$; *and*

— $\text{msg}'(c[\,], \mathsf{mac}(a[\ell, k[i]], k[i]), c[\,], \mathsf{mac}(a[\ell, k'[i, j_2]], k'[i, j_2]))$.

Thus, we obtain two facts msg' *where the second arguments are equal whereas the fourth arguments are different (since $j_1 \neq j_2$). Using the Clause* (Rl) *and the clause corresponding to the test of failure of destructor* equals, bad *will be derivable.*

This false attack is due to the fact that $a[\ell, k[i]]$ and $a[\ell, k'[i, j]]$ do not have the same level of freshness that is, the same arguments. This is why, to ensure that a name generated have the same level of freshness in $\mathsf{fst}(P)$ *and* $\mathsf{snd}(P)$, *both x and x' are put as argument of a in* $\mathsf{fst}(P)$ *and* $\mathsf{snd}(P)$. *For instance, with our formalism, instead of Clause 3, we obtain from P the clause:*

$$\text{msg}'(d[\,], x', d[\,], x'') \rightarrow \text{msg}'(c[\,], \mathsf{mac}(a[\ell, x', x''], x'), c[\,], \mathsf{mac}(a[\ell, x', x''], x''))$$

By applying this clause on $\text{msg}'(d[\,], k[i], d[\,], k'[i, j_1])$ *and* $\text{msg}'(d[\,], k[i], d[\,], k'[i, j_2])$, *we obtain:*

— $\text{msg}'(c[\,], \mathsf{mac}(a[\ell, k[i], k'[i, j_1]], k[i]), c[\,], \mathsf{mac}(a[\ell, k[i], k'[i, j_1]], k'[i, j_1]))$; *and*

— $\text{msg}'(c[\,], \mathsf{mac}(a[\ell, k[i], k'[i, j_2]], k[i]), c[\,], \mathsf{mac}(a[\ell, k[i], k'[i, j_2]], k'[i, j_2]))$.

In such a case, $\mathsf{mac}(a[\ell, k[i], k'[i, j_1]], k[i]) \neq \mathsf{mac}(a[\ell, k[i], k'[i, j_2]], k[i])$ *and so the previous false attack is avoided.*

This example is a toy example but this kind of false attack can be found when checking the unlinkability of the e-passport protocol (see Section 3.3).

9.3.4 Proving equivalences

Let $\rho_0 = \{a \mapsto (a[\,], a[\,]) \mid a \in \mathit{fnames}(P_0)\}$. We define the set of clauses that corresponds to biprocess P_0 as:

$$\mathcal{R}_{P_0} = [\![\text{unevaluated}(P_0)]\!]\rho_0\emptyset\emptyset \cup \{(\text{Rinit}), (\text{Rn}), \ldots, (\text{Rfailure}), (\text{Rfailure}')\}$$

The following theorem enables us to prove equivalences from these clauses.

Theorem 9.1. *If* bad *is not a logical consequence of $\mathcal{R}_{P_0}$, then P_0 satisfies observational equivalence.*

This theorem shows the soundness of the translation. The proof of this theorem is adapted from the proof of Theorem 3 of [BAF08]. Furthermore, since we use almost the same patterns and facts as in [BAF08], we also use the algorithm proposed in [BAF08] to automatically check if bad is a logical consequence of $\mathcal{R}_{P_0}$, with the only change that we use the unification algorithm for may-fail patterns.

9.3.5 Proving Properties P1 and P2

In this section, we propose two algorithms that respectively verify Properties P1 and P2 of the destructors of Σ. We already know that all deterministic destructors obtained from Lemma 9.1 satisfy both properties. However, since our extension allows a user to define his own destructor with side conditions, it is important to ensure that such destructors satisfy the desired properties.

Consider an equational presentation Σ and assume that Σ' models Σ. Consider the fact $\text{att}(mp)$ where mp is a may-fail pattern. In order to verify that the destructors of Σ satisfy Properties P1 and P2, we generate a new set of clauses defined as follows:

For some a, $\mathrm{att}(a[x])$ (Rname)

$\mathrm{att}(\mathsf{fail})$ (Rfail)

For each constructor f or arity n
$\mathrm{att}(x_1) \wedge \ldots \wedge \mathrm{att}(x_n) \rightarrow \mathrm{att}(\mathsf{f}(x_1, \ldots, x_n))$ (Rcons)

For each destructor g, for each pair of rewrite rules
$\mathsf{g}(U_1, \ldots, U_n) \rightarrow U \;||\; \phi$ and $\mathsf{g}(U'_1, \ldots, U'_n) \rightarrow U' \;||\; \phi'$
in $\mathrm{def}_{\Sigma'}(\mathsf{g})$ (after renaming of variables),
σ is the most general unifier of $\big\{(U, x), (U', y),$
$\big((U_1, \ldots, U_n), (U'_1, \ldots, U'_n)\big)\big\}$ and x, y fresh,
$\mathrm{att}(U_1\sigma) \wedge \ldots \wedge \mathrm{att}(U_n\sigma) \wedge \mathrm{formula}(\phi\sigma \wedge \phi'\sigma \wedge x\sigma \neq_\Sigma y\sigma) \rightarrow \mathrm{bad}$ (Rdeter)

For each destructor g, for each pair of rewrite rules
$\mathsf{g}(U_1, \ldots, U_n) \rightarrow U \;||\; \phi$ and $\mathsf{g}(U'_1, \ldots, U'_n) \rightarrow U' \;||\; \phi'$
in $\mathrm{def}_{\Sigma'}(\mathsf{g})$ (after renaming of variables),
σ is the most general unifier of $\big\{(U, x), (U', \mathsf{fail}),$
$\big((U_1, \ldots, U_n), (U'_1, \ldots, U'_n)\big)\big\}$ and x fresh,
$\mathrm{att}(U_1\sigma) \wedge \ldots \wedge \mathrm{att}(U_n\sigma) \wedge \mathrm{formula}(\phi\sigma \wedge \phi'\sigma) \rightarrow \mathrm{bad}$ (Rdeter2)

For each destructor g, for each $\mathcal{I} \subseteq \{1, \ldots, n\}$
$\forall i \in \mathcal{I}, U_i = x_i$, and $\forall i \notin \mathcal{I}, U_i = \mathsf{fail}$
$x_1, \ldots, x_n$ fresh variables
$\mathrm{att}(x_1) \wedge \ldots \wedge \mathrm{att}(x_n) \wedge \mathit{fails}(U_1, \ldots, U_n) \rightarrow \mathrm{bad}$ (Rtotal)

where $\mathit{fails}(U_1, \ldots, U_n) = \bigwedge_{\substack{\mathsf{g}(V_1,\ldots,V_n)\rightarrow V \;||\; \phi \text{ in } \mathrm{def}_\Sigma(\mathsf{g}) \\ \sigma_u \text{ mgu of } (V_1,\ldots,V_n) \text{ and } (U_1,\ldots,U_n)}} \mathrm{formula}(\forall \tilde{z}.\big[\bigvee_{i\in\mathcal{I}} x_i \neq_\Sigma V_i\sigma_u \vee \neg\phi\sigma_u\big])$
where $\tilde{z}$ are the variables of $V_1\sigma_u, \ldots, V_n\sigma_u$.

Intuitively, the purpose of the clauses (Rcons) and (Rname) is to generate constructor terms. Clauses (Rdeter) and (Rdeter2) check that the destructors satisfy Property P2. Clauses (Rtotal) check that the destructors satisfy Property P1.

Note that in Clauses (Rtotal), the formulas ϕ included in the predicate $\mathrm{formula}(\phi)$ are more complex than the usual conjunction of inequalities. In [BAF08], the solving algorithm applies several simplifications on predicates $\mathrm{formula}(\phi)$ that preserve the solutions of ϕ. However, only the formulas $\bigwedge_i \forall \tilde{z}_i.p_i \neq_\Sigma p'_i$, for some $\tilde{z}_i$, p_i, p'_i, are handled in this algorithm. Hence, we added some simplifications (see Appendix D.1.3) on first order logic such that our extension of ProVerif can now handle predicates $\mathrm{formula}(\phi)$ with a formula ϕ of the form $\bigwedge_i \forall \tilde{x}_i.\big[M_i \neq_\Sigma N_i \vee \bigvee_j \exists \tilde{y}^i_j.L^i_j =_\Sigma O^i_j\big]$.

The algorithms that check Properties P1 and P2 follow the next two lemmas (proofs in Appendix D.1.2):

Lemma 9.6. *We consider* $\mathcal{R}_{P1} = \{(\mathrm{Rname}), (\mathrm{Rfail}), (\mathrm{Rcons}), (\mathrm{Rtotal})\}$. bad *is not a logical consequence of* $\mathcal{R}_{P1}$ *if, and only if, for all* g *destructors of* Σ, g *satisfies Property P1.*

Lemma 9.7. *We consider* $\mathcal{R}_{(\mathrm{P2})} = \{(\mathrm{Rname}), (\mathrm{Rfail}), (\mathrm{Rcons}), (\mathrm{Rdeter}), (\mathrm{Rdeter2})\}$. bad *is not a logical consequence of* $\mathcal{R}_{P2}$ *if, and only if, for all* g *destructors of* Σ, g *satisfies Property P2.*

Note that we do not provide guarantees on the termination of the two algorithms but we rely on the algorithm of [BAF08] to check if bad is derivable or not.

9.4 Automatic modification of the protocol

In this section, we first present the false attack that we want to avoid and then propose an algorithm to automatically generate, from a biprocess P, equivalent biprocesses on which

ProVerif will avoid this kind of false attack.

9.4.1 Targeted false attack

We present a false attack on the anonymity of the *private authentication* protocol due to structural conditional branching, but similar false attack exists on the unlinkability of the *Basic Access Control* protocol.

Example 9.12. *Coming back to the* private authentication *protocol (see Example 9.8), we obtained a biprocess P_0 on which we would ask* ProVerif *to check the equivalence. Unfortunately,* ProVerif *is unable to prove the equivalence of P_0 and yields a false attack. Indeed, consider the evaluation context C defined as follows:*

$$C \stackrel{def}{=} \_ \mid \nu n_i.\mathsf{in}(c, x_{sk_a}).\mathsf{in}(c, x_{sk_{a'}}).\mathsf{in}(c, x_{sk_b}).\mathsf{out}(c, \mathsf{aenc}(\langle n_i, x_{sk_a}\rangle, x_{sk_b}))$$

The process $C[P_0]$ can be derived as follows:

$$\begin{aligned} C[P_0] \to^*_\Sigma\ & \nu n_i.\nu sk_a.\nu sk_{a'}.\nu sk_b.\big(\mathsf{out}(c, \mathsf{aenc}(\langle n_i, \mathsf{pk}(sk_a)\rangle, \mathsf{pk}(sk_b))) \mid System(\mathsf{diff}[sk_a, sk_{a'}], sk_b)\big) \\ \to^*_\Sigma\ & \nu n_i.\nu sk_a.\nu sk_{a'}.\mathsf{let}\ z = \mathsf{equals}(\mathsf{proj}_2(\langle n_i, \mathsf{pk}(sk_a)\rangle)), \mathsf{pk}(\mathsf{diff}[sk_a, sk_{a'}]))\ \mathsf{in} \\ & \quad \mathsf{out}(c, \mathsf{aenc}(\langle n_i, \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(\mathsf{diff}[sk_a, sk_{a'}]))) \\ & \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(sk_b))) \end{aligned}$$

However from this point, the biprocess gets stucked, i.e. *no internal reduction rule is applicable. More specifically, neither the internal rule (Red Fun 1) nor (Red Fun 2) is applicable. Indeed, if we denote $D = \mathsf{equals}(\mathsf{proj}_2(\langle n_i, sk_a\rangle)), \mathsf{pk}(\mathsf{diff}[sk_a, sk_{a'}]))$, we have that $\mathsf{snd}(D)\downarrow_\Sigma \mathsf{fail}$ and $\mathsf{fst}(D)\downarrow_\Sigma \mathsf{pk}(sk_a)$, which contradicts the item 2 of Lemma 9.3. So* ProVerif *cannot prove the equivalence.*

On the other hand, even though a different branch of the let is taken, the process outputs the message $\mathsf{aenc}(\langle n_b, \langle n_a, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))$ in the first variant (in branch of the let) and the message $\mathsf{aenc}(n_b, \mathsf{pk}(sk_b))$ in the second variant (else branch of the let). Intuitively, these two messages are indistinguishable, so in fact the attacker will not be able to determine which branch of the let is taken, and observational equivalence still holds.

As shown in the previous example, ProVerif may return a false attack against the observational equivalence, when a different branch of *let* is taken, but these branches are in fact indistinguishable. In this section, we will show how to use destructors with side conditions in order to avoid such false attacks. Moreover, we will propose an algorithm to automatically transform a biprocess into biprocesses that will avoid such false attacks when such a transformation is possible.

The key idea is to transform term evaluations such as $\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{out}(c, M_1)\ \mathsf{else}\ \mathsf{out}(c, M_2)$ into a computation that always succeeds $\mathsf{let}\ x = D'\ \mathsf{in}\ \mathsf{let}\ m = D''\ \mathsf{in}\ \mathsf{out}(c, m)$. The term evaluation D' will correspond to the value of the evaluation of D when the latter succeeds and a new constant when D fails. Thus we ensure that D' never fails. Moreover, the term evaluation D'' computes either M_1 or M_2 depending on the value of D', *i.e.* depending on whether D succeeds or not. The omitted else 0 branches are never taken. Since the same branch is always taken, the false attack disappears. To do that, we introduce three new destructors glet, gletin, notfail and a constant $\mathsf{c_o}$, which rely on the side conditions that we have added to destructors. These new destructors are defined as follows:

$$\begin{array}{lll} \mathrm{def}_\Sigma(\mathsf{glet}) = & \mathrm{def}_\Sigma(\mathsf{gletin}) = & \mathrm{def}_\Sigma(\mathsf{notfail}) = \\ \quad \mathsf{glet}(x) \to x & \quad \mathsf{gletin}(x, u, v) \to u \mid\mid x \neq_\Sigma \mathsf{c_o} & \quad \mathsf{notfail}(x) \to \mathsf{fail} \\ \quad \mathsf{glet}(\mathsf{fail}) \to \mathsf{c_o} & \quad \mathsf{gletin}(\mathsf{c_o}, u, v) \to v & \quad \mathsf{notfail}(\mathsf{fail}) \to \mathsf{c_o} \\ & \quad \mathsf{gletin}(\mathsf{fail}, u, v) \to \mathsf{fail} & \end{array}$$

One can easily check that $\mathrm{def}_\Sigma(\mathsf{glet})$, $\mathrm{def}_\Sigma(\mathsf{gletin})$ and $\mathrm{def}_\Sigma(\mathsf{notfail})$ satisfy Properties P1 and P2. Intuitively, the destructor glet evaluates its argument and returns either the result of this evaluation when it did not fail or else returns the new constant $\mathsf{c_o}$ instead of the failure constant fail. The

destructor gletin will get the result of glet as first argument and return its third argument if glet returned $\mathsf{c_o}$, and its second argument otherwise. Importantly, glet never fails: it returns $\mathsf{c_o}$ instead of fail. Hence, let $x = D$ in out(c, M_1) else out(c, M_2) can be transformed into let $x =$ eval glet(D) in let $m =$ eval gletin(x, M_1, M_2) in out(c, m): if D succeeds, x has the same value as before, and $x \neq \mathsf{c_o}$, so gletin(x, M_1, M_2) returns M_1; if D fails, $x = \mathsf{c_o}$ and gletin(x, M_1, M_2) returns M_2. The destructor notfail inverses the status of a term evaluation: it fails if and only if its argument does not fail. These destructors will be used in the next section.

Example 9.13. *Coming back to Example 9.12, the false attack occurs due to the following term evaluation:*

$$\begin{array}{l}\mathsf{let}\ z = \mathsf{equals}(\mathsf{proj}_2(x), \mathsf{pk}(\mathsf{diff}[ska, ska'])) \ \mathsf{in}\\ \quad \mathsf{out}(c, \mathsf{aenc}(\langle n_i, \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(\mathsf{diff}[sk_a, sk_{a'}])))\\ \ \mathsf{else}\ \mathsf{out}(c, \mathsf{aenc}(n_b, \mathsf{pk}(sk_b)))\end{array}$$

We transform this term evaluation as explained above:

$$\mathsf{let}\ z = \mathsf{gletin}(\mathsf{glet}(\mathsf{equals}(\mathsf{proj}_2(x), \mathsf{pk}(\mathsf{diff}[ska, ska'])))), M, M')\ \mathsf{in}\ \mathsf{out}(c, z)$$

where $M = \mathsf{aenc}(\langle n_i, \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(\mathsf{diff}[sk_a, sk_{a'}]))$ *and* $M' = \mathsf{aenc}(n_b, \mathsf{pk}(sk_b))$. *Note that with* $x = \langle n_i, \mathsf{pk}(sk_a)\rangle$ *(see Example 9.12), if* D *is the term evaluation* $D = \mathsf{gletin}(\mathsf{glet}(\mathsf{equals}(\mathsf{proj}_2(x)$ $, \mathsf{pk}(\mathsf{diff}[ska, ska'])))), M, M')$*, we obtain that:*

- $\mathsf{fst}(D) \downarrow \mathsf{aenc}(\langle n_i, \langle n_b, \mathsf{pk}(sk_b)\rangle\rangle, \mathsf{pk}(sk_a))$
- $\mathsf{snd}(D) \downarrow \mathsf{aenc}(n_b, \mathsf{pk}(sk_b))$

which corresponds to what $\mathsf{fst}(P_0)$ *and* $\mathsf{snd}(P_0)$ *respectively output. Thanks to this, if we denote by* P_0' *our new biprocess, we obtain that* $\mathsf{fst}(P_0) \approx \mathsf{fst}(P_0')$ *and* $\mathsf{snd}(P_0) \approx \mathsf{snd}(P_0')$. *Furthermore,* ProVerif *will be able to prove that the biprocess* P_0' *satisfies equivalence, i.e.* $\mathsf{fst}(P_0') \approx \mathsf{snd}(P_0')$ *and so* $\mathsf{fst}(P_0) \approx \mathsf{snd}(P_0)$.

The transformation proposed in the previous example can be generalised to term evaluations that perform other actions than just a single output. However, it is possible only if the success branch and the failure branch of the term evaluation both input and output the same number of terms. For example, the biprocess $P =$ let $x = D$ in out(c, M).out(c, M') else out(c, N) cannot be modified into one without else branch even with our new destructors. On the other hand, the success or failure of D can really be observed by adversary, by tracking the number of outputs on channel c, so the failure of the proof of equivalence corresponds to a real attack in this case.

9.4.2 Merging and simplifying biprocesses

To automatically detect and apply this transformation, we define two functions, denoted *merge* and *simpl*. The function *simpl* takes one biprocess as argument and applies the transformation when possible. The function *merge* takes two biprocesses as arguments and detects if those two biprocesses can be merged into one biprocess. Furthermore, if the merging is possible, it returns the merged biprocess. This merged biprocess is expressed using a new operator diff', similar to diff, in such a way that $\mathsf{diff}'[D, D']$ is a term evaluation. Furthermore, we introduce the functions fst' and snd': $\mathsf{fst}'(P)$ replaces each $\mathsf{diff}'[D, D']$ with D in P; similarly, $\mathsf{snd}'(P)$ replaces each $\mathsf{diff}'[D, D']$ with D' in P.

Figures 9.6 and 9.7 define the functions *merge* and *simpl*. The function *merge* is partial whereas *simpl* is total. Both functions are non-deterministic; the implementation may try all possibilities. In the current implementation of ProVerif, we apply the rules (Mlet3) and (Mlet4) only if the rules (Mlet1) and (Mlet2) are not applicable. Moreover, we never merge 0 with a process different from 0. This last restriction is crucial to reduce the number of biprocesses returned by the functions *merge* and *simpl*. Typically, we avoid 0 and let $x = M$ in P else 0 to be merged by the rule (Mlet4).

Case (Mout) detects that both biprocesses output a message while case (Min) detects that both biprocesses input a message. We introduce a *let* for the channels and messages so that they

$$merge(0,0) \stackrel{def}{=} 0 \tag{Mnil}$$

$$merge(\mathsf{out}(M,N).P, \mathsf{out}(M',N').P') \stackrel{def}{=}$$
$$\mathsf{let}\ x = \mathsf{diff}'[M,M']\ \mathsf{in}\ \mathsf{let}\ x' = \mathsf{diff}'[N,N']\ \mathsf{in}\ \mathsf{out}(x,x').merge(P,P') \tag{Mout}$$
where x and x' are fresh variables

$$merge(\mathsf{in}(M,x).P, \mathsf{in}(M',x').P') \stackrel{def}{=}$$
$$\mathsf{let}\ y = \mathsf{diff}'[M,M']\ \mathsf{in}\ \mathsf{in}(y,y').merge(P\{^{y'}/_{x}\}, P'\{^{y'}/_{x'}\}) \tag{Min}$$
where y and y' are fresh variables

$$merge(P_1 \mid \ldots \mid P_n, P'_1, \mid \ldots \mid P'_n) \stackrel{def}{=} Q_1 \mid \ldots \mid Q'_n \tag{Mpar}$$
if $(i_1,\ldots,i_n)$ is a permutation of $(1,\ldots,n)$
and for all $k \in \{1,\ldots,n\}$, $Q_k = merge(P_k, P'_{i_k})$

$$merge(\nu a.P, Q) \stackrel{def}{=} \nu a.merge(P,Q) \tag{Mres}$$
after renaming a such that $a \notin fnames(Q)$

$$merge(!\,\nu a_1 \ldots \nu a_n.!\,P, !\,P') \stackrel{def}{=} !\,\nu a_1 \ldots \nu a_n.merge(!\,P, !\,P') \tag{Mrepl1}$$
after renaming $a_1,\ldots,a_n$ such that $a_1,\ldots,a_n \notin fnames(P')$

$$merge(!\,P, !\,P') \stackrel{def}{=} !\,merge(P,P') \tag{Mrepl2}$$
if there is no P_1 such that $P = !\,P_1$ and no P'_1 such that $P' = !\,P'_1$

$$merge(\mathsf{let}\ x = D\ \mathsf{in}\ P_1\ \mathsf{else}\ P_2, \mathsf{let}\ x' = D'\ \mathsf{in}\ P'_1\ \mathsf{else}\ P'_2) \stackrel{def}{=}$$
$$\mathsf{let}\ y = \mathsf{diff}'[D,D']\ \mathsf{in}\ Q_1\ \mathsf{else}\ Q_2 \text{ if } y \text{ is a fresh variable,} \tag{Mlet1}$$
$Q_1 = merge(P_1\{^{y}/_{x}\}, P'_1\{^{y}/_{x'}\})$, and $Q_2 = merge(P_2, P'_2)$

$$merge(\mathsf{let}\ x = D\ \mathsf{in}\ P_1\ \mathsf{else}\ P_2, \mathsf{let}\ x' = D'\ \mathsf{in}\ P'_1\ \mathsf{else}\ P'_2) \stackrel{def}{=}$$
$$\mathsf{let}\ y = \mathsf{diff}'[D, \mathsf{notfail}(D')]\ \mathsf{in}\ Q_1\ \mathsf{else}\ Q_2 \text{ if } y \text{ is a fresh variable,} \tag{Mlet2}$$
$x' \notin fvars(P'_1)$, $Q_1 = merge(P_1\{^{y}/_{x}\}, P'_2)$, and $Q_2 = merge(P_2, P'_1)$

$$merge(\mathsf{let}\ x = D\ \mathsf{in}\ P_1\ \mathsf{else}\ P_2, P') \stackrel{def}{=} \mathsf{let}\ y = \mathsf{diff}'[D, \mathsf{c_o}]\ \mathsf{in}\ Q\ \mathsf{else}\ P_2 \tag{Mlet3}$$
if y is a fresh variable and $Q = merge(P_1\{^{y}/_{x}\}, P')$

$$merge(\mathsf{let}\ x = D\ \mathsf{in}\ P_1\ \mathsf{else}\ P_2, P') \stackrel{def}{=} \mathsf{let}\ y = \mathsf{diff}'[D, \mathsf{fail}]\ \mathsf{in}\ P_1\{^{y}/_{x}\}\ \mathsf{else}\ Q \tag{Mlet4}$$
if y is a fresh variable and $Q = merge(P_2, P')$

plus symmetric cases (Mres′), (Mrepl1′), (Mlet2′), (Mlet3′), and (Mlet4′) obtained from (Mres), (Mrepl1), (Mlet2), (Mlet3), and (Mlet4) by swapping the first and second arguments of *merge* and diff'.

Figure 9.6: Definition of the function *merge*

can later be replaced by a term evaluation. Case (Mpar) uses the commutativity and associativity of parallel composition to increase the chances of success of *merge*. Cases (Mres) and (Mres′) use $Q \approx \nu a.Q$ when $a \notin \mathit{fnames}(Q)$ to allow merging processes even when a restriction occurs only on one side. Case (Mrepl2) is the basic merging of replicated processes, while Case (Mrepl1) allows merging $!!P$ with $!P'$ (case $n = 0$) because $!P \approx !!P$, and furthermore allows restrictions between the two replications, using $Q \approx \nu a.Q$.

Case (Mlet1) merges two processes that both contain term evaluations, by merging their success branches together and their failure branches together. On the other hand, Case (Mlet2), (Mlet2′) also merge two processes that contain term evaluations, by merging the success branches with the other failure branches.

Cases (Mlet3), (Mlet3′), (Mlet4), (Mlet4′) allow merging a term evaluation with another process P', by merging P' with either the success branch or the failure branch of the term evaluation. (The evaluation of $\mathsf{c_o}$ always succeeds since it is a message.) This merging is useful when ProVerif can prove that the resulting process satisfies equivalence, hence when both sides of the obtained *let* succeed simultaneously. Therefore, rule (Mlet3) is helpful when the term evaluation D always succeeds, and rule (Mlet4) when D always fails. When no such case applies, merging fails.

$$\mathit{simpl}(0) \stackrel{def}{=} 0 \tag{Snil}$$

$$\mathit{simpl}(\mathsf{out}(M,N).P) \stackrel{def}{=} \mathsf{out}(M,N).\mathit{simpl}(P) \tag{Sout}$$

$$\mathit{simpl}(\mathsf{in}(M,x).P) \stackrel{def}{=} \mathsf{in}(M,x).\mathit{simpl}(P) \tag{Sin}$$

$$\mathit{simpl}(P \mid Q) \stackrel{def}{=} \mathit{simpl}(P) \mid \mathit{simpl}(Q) \tag{Smid}$$

$$\mathit{simpl}(\nu a.P) \stackrel{def}{=} \nu a.\mathit{simpl}(P) \tag{Sres}$$

$$\mathit{simpl}(!\,P) \stackrel{def}{=} !\,\mathit{simpl}(P) \tag{Srepl}$$

$$\begin{aligned}&\mathit{simpl}(\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ P') \stackrel{def}{=} \mathsf{let}\ x = \mathrm{eval}\ \mathsf{glet}(D)\ \mathsf{in}\ Q\ \mathsf{else}\ 0\\ &\quad \text{if } Q' = \mathit{merge}(\mathit{simpl}(P), \mathit{simpl}(P')) \text{ and}\\ &\quad Q = Q'\{^{\mathrm{eval}\ \mathsf{gletin}(x,D_1,D_2)}/_{\mathsf{diff}'[D_1,D_2]}\}\end{aligned} \tag{Smerge}$$

$$\begin{aligned}&\mathit{simpl}(\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ P') \stackrel{def}{=} \mathsf{let}\ x = D\ \mathsf{in}\ \mathit{simpl}(P)\ \mathsf{else}\ \mathit{simpl}(P')\\ &\quad \text{if there is no } Q \text{ such that } Q = \mathit{merge}(\mathit{simpl}(P), \mathit{simpl}(P'))\end{aligned} \tag{Slet}$$

Figure 9.7: Definition of the function *simpl*

The function *simpl* proceeds by induction on the process. The only interesting case is (Smerge), which performs the transformation of term evaluations outlined above, when we can merge the success and failure branches. $Q'\{^{\mathrm{eval}\ \mathsf{gletin}(x,D_1,D_2)}/_{\mathsf{diff}'[D_1,D_2]}\}$ means that we replace in Q' every instance of $\mathsf{diff}'[D_1, D_2]$, for some D_1, D_2, with eval $\mathsf{gletin}(x, D_1, D_2)$.

Lemmas 9.8 and 9.9 below show that the functions *merge* and *simpl* preserve observational equivalence. In these two lemmas, we consider processes P and P' that are not necessarily closed. We say that a context C is closing for P when $C[P]$ is closed. Moreover, given two biprocesses P and Q, we say that $P \approx Q$ if, and only if, $\mathsf{fst}(P) \approx \mathsf{fst}(Q)$ and $\mathsf{snd}(P) \approx \mathsf{snd}(Q)$. All results of this section are proved in Appendix D.2.

Lemma 9.8. *Let P and P' be two biprocesses. If $\mathit{merge}(P, P') = Q$, then:*

— *for all contexts C closing for P, $C[P] \approx C[\mathsf{fst}'(Q)]$;*

— *for all contexts C closing for P', $C[P'] \approx C[\mathsf{snd}'(Q)]$.*

Lemma 9.9. *Let P be a biprocess. For all contexts C closing for P, $C[P] \approx C[\mathit{simpl}(P)]$.*

Corollary 9.1. *Let P be a closed biprocess. If $simpl(P)$ satisfies observational equivalence then $\mathsf{fst}(P) \approx \mathsf{snd}(P)$.*

Proof. $simpl(P)$ satisfies observational equivalence implies $\mathsf{fst}(simpl(P)) \approx \mathsf{snd}(simpl(P))$. Since P is closed, we can apply Lemma 9.9 with the empty context, so $simpl(P) \approx P$. Hence by definition of the observation equivalence on biprocesses, we obtain that $\mathsf{fst}(simpl(P)) \approx \mathsf{fst}(P)$ and $\mathsf{snd}(P) \approx \mathsf{snd}(simpl(P))$. We conclude that $\mathsf{fst}(P) \approx \mathsf{snd}(P)$. □

From Corollary 9.1, we can extract our new algorithm. Given a biprocess P as input, we first apply the function *simpl* on P. Since the function *simpl* is total but non-deterministic, we may have several biprocesses as result for $simpl(P)$. If ProVerif is able to prove equivalence on at least one of them, then we conclude that $\mathsf{fst}(P) \approx \mathsf{snd}(P)$.

Corollary 9.2. *Let P and P' be two closed processes. Let $Q = merge(simpl(P), simpl(P'))$. If the biprocess $Q' = Q\{^{\mathsf{diff}[D,D']}/_{\mathsf{diff}'[D,D']}\}$ satisfies observational equivalence, then we have $P \approx P'$.*

Proof. According to Figures 9.6 and 9.7, if P and P' do not contain any diff then so do $simpl(P)$, $simpl(P')$, and $merge(P, P')$. Hence, Q does not contain any diff. Hence, the biprocess $Q' = Q\{^{\mathsf{diff}[D,D']}/_{\mathsf{diff}'[D,D']}\}$ satisfies $\mathsf{fst}(Q') = \mathsf{fst}'(Q)$ and $\mathsf{snd}(Q') = \mathsf{snd}'(Q)$. Since Q' satisfies observational equivalence, we have that $\mathsf{fst}(Q') \approx \mathsf{snd}(Q')$ thus $\mathsf{fst}'(Q) \approx \mathsf{snd}'(Q)$. Since P and P' are closed processes, by Lemma 9.9, we have $P \approx simpl(P)$ and $P' \approx simpl(P')$. Furthermore, by Lemma 9.8, we deduce that $simpl(P) \approx \mathsf{fst}'(Q)$ and $simpl(P') \approx \mathsf{snd}'(Q)$. We conclude that $P \approx P'$. □

Currently, ProVerif can only take a biprocess as input. But thanks to Corollary 9.2, we will be able to loosen this condition. Indeed, consider P and P' two processes (without diff). If we manage to merge them into the biprocess Q' as in Corollary 9.2 and to prove that Q' satisfies equivalence by ProVerif, then we obtain that $P \approx P'$.

9.5 Applications

Currently, we implemented our extension in a development version of ProVerif for typed processes. However, some features still need be adapted to the new formalism before the next stable release, such as the front-end for untyped processes. In any case, we present in this subsection the application of our extension on the private authentication protocol and the e-passport protocol (see Section 3.3).

9.5.1 Successful case study: the *private authentication* protocol

The simplification of processes described in this section was implemented in such a way that the transformation stays hidden from the users. However, since the function *simpl* may return several biprocesses, we do not automatically test the equivalence of all the biprocesses given by the function *simpl*. Furthermore, proving the equivalence requires a lot of times on some biprocesses and might even not terminate. Hence allowing the user to choose whether or not ProVerif has to prove equivalence on the biprocesses returned by the function *simpl* seemed more user-friendly.

By default, ProVerif first tries to prove the equivalence that was provided in the input file without modifying it. Figure 9.8 is a possible input file for the private authentication protocol. In case of non-equivalence, it will compute the function *simpl* on the input biprocess and propose to prove the equivalence on the simplified biprocesses. In the case of the private authentication protocol, ProVerif first states the non-equivalence due to a false attack (see Example 9.12), but then succeeds to prove the observational equivalence on the only biprocess obtained by application of the function *simpl*. Note that ProVerif also proposes to display the biprocesses obtained by application of *simpl* but they are usually difficult to parse due to the variable renaming and the extensive use of the destructors glet and gletin.

```
(* shared-key encryption as equation *)

fun aenc(bitstring, bitstring): bitstring.
fun pk(bitstring): bitstring.

reduc forall x: bitstring, y: bitstring; adec(aenc(x,pk(y)),y) = x.

(* declaration of public name *)

free c: channel.

(* declaration of biprocess *)

let processA(sk_a:bitstring,sk_b:bitstring) =
  new n_a: bitstring;
  out(c, aenc((n_a,pk(sk_a)),pk(sk_b)));
  in(c, x:bitstring);
  0.

let processB(sk_b:bitstring,sk_a:bitstring) =
  in(c, x:bitstring);
  new n_b:bitstring;

  let (n_a:bitstring,pub_a:bitstring) = adec(x, sk_b) in
    if pub_a = pk(sk_a)
    then
      out(c,aenc((n_a,n_b,pk(sk_b)),pk(sk_a)))
    else
      out(c,aenc(n_b,pk(sk_b)))
  else
    out(c,aenc(n_b,pk(sk_b))).

let system(sk_a:bitstring,sk_b:bitstring) =
  !processA(sk_a,sk_b) | !processB(sk_b,sk_a).

process
  new sk_a:bitstring; new sk_b:bitstring; new sk_c:bitstring;
    out(c,pk(sk_a));out(c,pk(sk_b));out(c,pk(sk_c));
    system(choice[sk_a,sk_c],sk_b)
```

Figure 9.8: Input file for the anonymity of the private authentication protocol

9.5.2 Limitations: the *Basic Access Control* protocol

We consider here the unlinkability of the *BAC* protocol. As mentioned in Section 3.4, this security property can be expressed by the following equivalence:

$$!\nu ke.\ \nu km.\ !BAC(ke, km) \approx\ !\nu ke.\ \nu km.BAC(ke, km)$$

In contrast to the equivalence given in Section 3.3, we explicitly denote the keys whom *BAC* depends on and we rely on the observational equivalence and not the trace equivalence. Moreover, $BAC(ke, km)$ is in fact the parallel composition of two processes, one for the passport and one for the reader, each of them depending on the keys ke and km.

Note that the two parts of this equivalence cannot be directly transformed into a biprocess due to the second replication on the left part of the process. However, we can modify slightly the two parts of the equivalence so that they can define a biprocess. Indeed, the following equivalence holds:

$$!\nu ke.\ \nu km.BAC(ke, km) \approx\ !\nu ke'.\nu km'.!\nu ke.\nu km.BAC(ke, km)$$

Intuitively, this non-trivial equivalence holds due to the fact that the keys ke',km' are not used in $BAC(ke, km)$ and that $!!P \approx !P$. Similarly, we have that:

$$!\nu ke.\ \nu km.\ !BAC(ke, km) \approx\ !\nu ke.\nu km.!\nu ke'.\nu km'.BAC(ke, km)$$

Hence, after renaming of the keys ke, km, ke', km', the unlinkability of the BAC protocol can be expressed by the following equivalence:

$$!\,\nu ke.\nu km.!\,\nu ke'.\nu km'.BAC(ke, km) \;\approx\; !\,\nu ke.\nu km.!\,\nu ke'.\nu km'.BAC(ke', km')$$

which is represented by the biprocess $!\,\nu ke.\nu km.!\,\nu ke'.\nu km'.BAC(\mathsf{diff}[ke, ke'], \mathsf{diff}[km, km'])$. Figure 9.9 is a possible input file for the unlinkability of the BAC protocol.

Thanks to our extension, the false attack described in Example 9.11 is now avoided. Moreover, after application of the function *simpl*, ProVerif also avoids a false attack similar to the false attack on the anonymity of the private authentication protocol (due to the test $mac(m_e, km) = m_m$ in the process representing the passport). Hence, in both case studies, ProVerif avoids the false attacks that are due to conditional branching. Unfortunately, even with our extension, ProVerif is still not able to prove unlinkability of the BAC protocol and returns a false attack. This false attack, described below, is caused by the fact that ProVerif necessarily matches an interleaving of actions of one part of a biprocess with exactly the same interleaving of actions of the other part of this biprocess (see semantics of biprocesses).

We denote $P(ke, km)$ the process representing the passport and $R(ke, km)$ the one representing the reader. The intruder proceeds as follows: He starts by engaging the communication with two sessions of the same passport on the one hand, and so necessarily one session of two different passports on the other hand. Hence, it is similar to a biprocess where the left and right parts are represented as follows:

— *Left:* $P(ke_1, km_1) \mid R(ke_1, km_1) \mid P(ke_1, km_1) \mid R(ke_1, km_1)$

— *Right:* $P(ke_1, km_1) \mid R(ke_1, km_1) \mid P(ke_2, km_2) \mid R(ke_2, km_2)$

In the right part of the biprocess, the intruder receives the nonce nt_1 that $P(ke_1, km_1)$ output and sends it to the reader $R(ke_2, km_2)$. Similarly, he sends the nonce nt_2 that $P(ke_2, km_2)$ output and sends it to the reader $R(ke_1, km_1)$. Hence, the readers $R(ke_1, km_1)$ and $R(ke_2, km_2)$ will output respectively the messages $(m_1, mac(m_1, km_1))$ and $(m_2, mac(m_2, km_2)$ where $m_1 = \mathsf{senc}(\langle nr_1, \langle nt_2, kr_1\rangle\rangle, ke_1)$ and $m_2 = \mathsf{senc}(\langle nr_2, \langle nt_1, kr_2\rangle\rangle, ke_2)$. Note that since the intruder swaps the nonces nt_1, nt_2 that was intended to the readers, they are also swapped in m_1, m_2.

Then, the intruder swaps again the messages $(m_1, mac(m_1, km_1))$ and $(m_2, mac(m_2, km_2))$ and sends them to $P(ke_2, km_2)$ and $P(ke_1, km_1)$ respectively. Since $P(ke_1, km_1)$ receives m_2 which is encrypted with ke_2, the decryption $\mathsf{sdec}(m_2, ke_1)$ will fail and so $P(ke_1, km_1)$ will output the error $error_{6A80}$. Similarly, $P(ke_2, km_2)$ will also output the error $error_{6A80}$.

On the other hand, in the left part of the biprocess, since the intruder communicates with two sessions of the same passport, there is only one set of keys (ke_1, km_1) and so both decryptions will succeed. Moreover, since the intruder swaps the messages twice, both passports will be able to check that the nonce nt encrypted in the message they received corresponds to the nonce they previously sent. Hence, both passports will output a message of the form $\mathsf{senc}(m', mac(m', km_1))$. Since $error_{6A80}$ is public, the intruder is able to distinguish $error_{6A80}$ from $\mathsf{senc}(m', mac(m', km_1))$ and so ProVerif returns that the equivalence does not hold.

Intuitively, this false attack is due to the fact that the swaps done in the right part of the biprocess have to be executed in the left part of the biprocess with the two sessions of the same passport. However, when considering the observational equivalence, the actions of the right part of the equivalence could have been matched in the left part of the equivalence by considering a session of a third passport and reader (available thanks to the replication) and then swapping the messages between the first and third passport.

```
(* ePassport Protocol Taken from Arapinis, Ryan, CSF'10 *)

fun enc(bitstring, bitstring): bitstring.
fun mac(bitstring,bitstring): bitstring.
fun dec(bitstring, bitstring): bitstring.

(* Symetric decryption *)

equation forall x: bitstring, y:bitstring; dec(enc(x,y),y) = x.

(* addition of one *)

free c: channel.
free get_challenge: bitstring.
free error_6A80: bitstring.

let reader(ke:bitstring,km:bitstring) =
  out(c,get_challenge);
  in(c,nt:bitstring);
  new nr:bitstring;
  new kr:bitstring;
  let m:bitstring = enc((nr,nt,kr),ke) in
    out(c,(m,mac(m,km)));
    in(c,y:bitstring).

let passport(ke:bitstring, km:bitstring) =
  in(c,x:bitstring);
  if x = get_challenge
  then
    new nt:bitstring;
    out(c,nt);
    in(c,y:bitstring);

    let (m_e:bitstring,m_m:bitstring) = y in
      if mac(m_e,km) = m_m
      then
        let (nr:bitstring,nt':bitstring,kr:bitstring) = dec(m_e,ke) in
          if nt = nt'
          then
            new kt:bitstring;
            out(c,(enc((nt,nr,kt),ke),mac(enc((nt,nr,kt),ke),km)))
          else
            out(c,error_6A80)
        else
          out(c, error_6A80)
      else
        out(c,error_6A80)
    else
      out(c,error_6A80).

process
  !new ke1:bitstring; new km1:bitstring;
    !new ke2:bitstring; new km2:bitstring;
      reader(choice[ke1,ke2],choice[km1,km2])
      |
      passport(choice[ke1,ke2],choice[km1,km2])
```

Figure 9.9: Input file for the unlinkability of the *BAC* protocol

Chapter 10

Conclusion and perpectives

This thesis reports our contributions to the automatic verification of cryptographic protocols. In particular, we focused our attention to privacy-type security properties that play an important role in many modern applications. Some further developments have already been proposed at the end of chapters. We recall the important ones and propose further perspectives.

Algorithms for deciding trace equivalence

In Part II, we designed a new decision procedure for the trace equivalence between bounded processes with possible non-trivial else branches and non-deterministic choices. This procedure is sound, complete and terminates but only accepts a fixed set of cryptographic primitives whose behaviour is expressed by a rewriting system.

The cryptographic primitives are idealised which is sometimes not accurate enough. For example, we do not model the length of messages. However, depending on the padding scheme, encryption scheme may disclose the length of the plaintext and therefore allow to distinguish two ciphers. Hence we would like to add new predicates, *e.g.* a predicate that compares the lengths of terms, in our decision procedure. In fact, such a research track would allow us to use our procedure along with [CC08] for proving computational indistinguishability for a certain class of protocols. Since these predicates would only intervene in the static equivalence, we would like to use the properties of the constraint systems that we obtain at the end of our current procedure, in order to facilitate the development of a decision procedure for trace equivalence with these predicates.

We cannot not currently handle some very interesting cryptographic protocols, such as e-voting protocols, that rely on different cryptographic primitives that the ones we fixed. Hence a natural direction of work would be to allow more cryptographic primitives. In particular, we are interested in the cryptographic primitives with algebraic properties such as exclusive-or or the blind signature. However, compared to adding predicates for the static equivalence, adding these cryptographic primitives will most certainly be a huge challenge, especially if we want to ensure the termination of our procedure.

Tool optimisations

Although the procedure described in Part II terminates, an early implementation showed that the computation of all interleavings of actions results in a significant increase of the execution time of the procedure, even for few sessions of small protocols such as the *Basic Access Control* protocol. We would like to optimise the algorithm such that the execution time remains reasonable, at least for few sessions of any standard protocol. There are two possible directions.

We could apply a partial order reduction on the symbolic traces of the processes [CJM00]. This area seems very promising since the experiments show that there are many interleavings that yield the same state. This is not straightforward, however, when considering equivalence

properties, since we have to match traces from a process to the other process. The reductions have to match on both processes.

An implementation of our algorithm on a parallel machine should be possible. Indeed, the two matrices resulting from an application of a constraint solving rule can be processed independently. Ideally, it should be possible to combine these two optimisations. However, this seems only possible with small enough reductions that do not break the parallelism. Nevertheless, both optimisations would still be implemented since they are both worthy depending on the user's hardware.

Proving more equivalences with ProVerif

In Part III, we proposed an extension of the tool ProVerif that proves more observational equivalences. In particular, we were able to automatically prove the anonymity property on the *private authentication* protocol for an unbounded number of sessions. However, as mentioned in Chapter 9, we are still unable to prove the unlinkability property for the *Basic Access Control* protocol even though we managed to avoid some previously existing false attacks. This a consequence of the matching by ProVerif of traces with the same scheduling in the two variants of the biprocesses. On the one hand, this is probably one of the most frequent cause of false attacks yield by ProVerif, and on the other hand, this is also one of the main reason why ProVerif is very efficient in practice for proving equivalence. To prove even more observational (or trace) equivalence with ProVerif, some approaches could be investigated.

This first approach would consist of applying some small transformations on the processes beforehand that are sound for observational (or trace) equivalence. However, our first tries on the *BAC* protocol seem to show that while such small transformations might exist, they would be very specific to the processes and therefore very hard to automate.

A second direction consists of relaxing a bit the matching of traces *e.g.* by modifying the replication identifiers on the left and right parts of biprocesses. Even if we manage to propose a transformation that is still sound for the observational equivalence, we also would have to preserve the termination "in practice" of ProVerif.

Although ProVerif and our decision procedure for trace equivalence are very different, it would be interesting to study the possible interactions between them. For example, when ProVerif yields a false attack, a possible idea would be to check the possible matched traces for the false attack in a our decision procedure and then inject the result in ProVerif so that it could prove the security properties by relying our the result of the procedure.

Composition results

In Chapter 5, we have shown that protocols can be securely composed w.r.t. privacy-type properties if they are tagged and do not reveal any shared secret. Moreover, we allow a finite number of public or verification keys associated to shared secret keys to be known by the intruder, provided that they were known at the beginning of the protocol. We also showed in this chapter how to apply our composition result to the parallel composition of the *Active Authentication* and *Passive Authentication* protocols from the *e-passport* protocols. However, we had to abstract the symmetric session keys that were generated by the *Basic Access Control* protocol beforehand. Thus, a possible extension of this work will consist of generalising our composition result in order to allow the sequential composition of processes.

In [ADK08], the authors characterise a class of protocols for which secrecy is decidable for an unbounded number of sessions. Typically, they showed how to transform a protocol that is secure for one session, in a protocol that is secure for an unbounded number of sessions. This transformation also relies on tagging of terms but the tags are not constant and are generated by the sessions. Thus, a possible direction of research is to extend their result to privacy-type properties.

All these possible extensions of our work heavily rely on tagged processes. However, most of existing real protocols are not tagged and tagging a process increases the payloads of messages that are sent over the network. Hence following the idea of [KT11], we would like to define an ideal

property that, if shown on each process individually, would ensure the secure composition of these processes. One of the difficulty of this approach would probably be to find the correct balance between the generality of such an ideal property and the difficulty of automatically verifying it.

Bibliography

[3GP10a] 3GPP. Technical specification group core network and terminals; mobile radio interface layer 3 specification; core network protocols; stage 3 (release 9). Technical report, 3rd Generation Partnership Project, 2010. 3GPP TS 24.008 V9.4.0.

[3GP10b] 3GPP. Technical specification group services and system aspects; 3G security; security architecture (release 9). Technical report, 3rd Generation Partnership Project, 2010. 3GPP TS 33.102 V9.3.0.

[3GP11] 3GPP. Technical specification group services and system aspects; 3G security; cryptographic algorithm requirements (release 10). Technical report, 3rd Generation Partnership Project, 2011. 3GPP TS 33.105 V10.0.0.

[ABF04] Martín Abadi, Bruno Blanchet, and Cédric Fournet. Just Fast Keying in the pi calculus. In David Schmidt, editor, *Programming Languages and Systems: 13th European Symposium on Programming (ESOP'04)*, volume 2986 of *LNCS*, pages 340–354, Heidelberg, 2004. Springer.

[AC02] R. Amadio and W. Charatonik. On name generation and set-based analysis in the Dolev-Yao model. In *Proc. of the 13th International Conference on Concurrency Theory (CONCUR'02)*, LNCS, pages 499–514, Brno, Czech Republic, 2002. Springer Verlag.

[AC06] M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. *Theoretical Computer Science*, 387(1-2):2–32, 2006.

[ACRR10] Myrto Arapinis, Tom Chothia, Eike Ritter, and Mark Ryan. Analysing unlinkability and anonymity using the applied pi calculus. In *Proc. 23rd IEEE Computer Security Foundations Symposium (CSF'10)*, pages 107–121. IEEE Computer Society Press, 2010.

[ACS+08] S. Andova, C. Cremers, K. GjøSteen, S. Mauw, S. Mjølsnes, and S. Radomirović. A framework for compositional verification of security protocols. *Information and Computation*, 206(2-4):425–459, 2008.

[ADK08] Myrto Arapinis, Stéphanie Delaune, and Steve Kremer. From one session to many: Dynamic tags for security protocols. In Iliano Cervesato, Helmut Veith, and Andrei Voronkov, editors, *Proceedings of the 15th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'08)*, volume 5330 of *Lecture Notes in Artificial Intelligence*, pages 128–142, Doha, Qatar, November 2008. Springer.

[AF01] M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *Proc. 28th Symposium on Principles of Programming Languages (POPL'01)*, pages 104–115. ACM Press, 2001.

[AF04] Martín Abadi and Cédric Fournet. Private authentication. *Theoretical Computer Science*, 322(3):427–476, 2004.

[AG99] Martín Abadi and Andrew D. Gordon. A calculus for cryptographic protocols: The spi calculus. *Information and Computation*, 148(1):1–70, 1999.

[AMR+12] M. Arapinis, L. Mancini, E. Ritter, M. Ryan, N. Golde, K. Redon, and R. Borgaonkar. New privacy issues in mobile telephony: fix and verification. In *Proc. 19th ACM*

Conference on Computer and Communications Security (CCS'12). ACM Press, 2012. To appear.

[AR00] M. Abadi and P. Rogaway. Reconciling two views of cryptography: the computational soundness of formal encryption. In *Proc. 1rst IFIP International Conference on Theoretical Computer Science*, volume 1872 of *Lecture Notes in Computer Science*, Sendai, Japan, 2000.

[BAF08] Bruno Blanchet, Martín Abadi, and Cédric Fournet. Automated verification of selected equivalences for security protocols. *Journal of Logic and Algebraic Programming*, 75(1):3–51, 2008.

[Bau05] Mathieu Baudet. Deciding security of protocols against off-line guessing attacks. In *Proc. 12th Conference on Computer and Communications Security (CCS'05)*, pages 16–25. ACM Press, 2005.

[Bau07] Mathieu Baudet. *Sécurité des protocoles cryptographiques : aspects logiques et calculatoires*. Phd thesis, École Normale Supérieure de Cachan, France, 2007.

[BCdH10] Mayla Bruso, K. Chatzikokolakis, and J. den Hartog. Formal verification of privacy for RFID systems. In *Proc. 23rd IEEE Computer Security Foundations Symposium (CSF'10)*. IEEE Computer Society Press, 2010.

[BCNP04] B. Barak, R. Canetti, J. Nielsen, and R. Pass. Universally composable protocols with relaxed set-up assumptions. In *Proc. 45th Symposium on Foundations of Computer Science (FOCS'04)*, pages 186–195. IEEE Computer Society Press, 2004.

[Bla01] Bruno Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *14th IEEE Computer Security Foundations Workshop (CSFW-14)*, pages 82–96, Cape Breton, Nova Scotia, Canada, June 2001. IEEE Computer Society.

[Bla04] Bruno Blanchet. Automatic proof of strong secrecy for security protocols. In *Proc. Symposium on Security and Privacy*, pages 86–100. IEEE Comp. Soc. Press, 2004.

[BMU08] Michael Backes, Matteo Maffei, and Dominique Unruh. Zero-knowledge in the applied pi-calculus and automated verification of the direct anonymous attestation protocol. In *IEEE Symposium on Security and Privacy*, pages 202–215, Los Alamitos, 2008. IEEE.

[BS96] F. Baader and K. U. Schulz. Unification in the union of disjoint equational theories: Combining decision procedures. *Journal of Symbolic Computation*, 21(2):211–243, 1996.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proc. 42nd Annual Symposium on Foundations of Computer Science (FOCS'01)*, pages 136–145, Las Vegas (Nevada, USA), 2001. IEEE Computer Society Press.

[CC08] Hubert Comon-Lundh and Véronique Cortier. Computational soundness of observational equivalence. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS'08)*, pages 109–118, Alexandria, Virginia, USA, October 2008. ACM Press.

[CC10] Ştefan Ciobâcă and Véronique Cortier. Protocol composition for arbitrary primitives. In *Proc. of the 23rd IEEE Computer Security Foundations Symposium (CSF'10)*, pages 322–336. IEEE Computer Society Press, 2010.

[CD94] Hubert Comon and Catherine Delor. Equational formulae with membership constraints. *Information and Computation*, 112(2):167–216, August 1994.

[CD09a] Véronique Cortier and Stéphanie Delaune. A method for proving observational equivalence. In *Proceedings of the 22nd IEEE Computer Security Foundations Symposium (CSF'09)*, pages 266–276, Port Jefferson, NY, USA, July 2009. IEEE Computer Society Press.

[CD09b] Véronique Cortier and Stéphanie Delaune. Safely composing security protocols. *Formal Methods in System Design*, 34(1):1–36, February 2009.

[CD12] Véronique Cortier and Stéphanie Delaune. Decidability and combination results for two notions of knowledge in security protocols. *Journal of Automated Reasoning*, 48(4):441–487, April 2012.

[CDK11] Céline Chevalier, Stéphanie Delaune, and Steve Kremer. Transforming password protocols to compose. In *Proc. 31st Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'11)*, Leibniz International Proceedings in Informatics, pages 204–216. Leibniz-Zentrum für Informatik, 2011.

[CDM11] Hubert Comon-Lundh, Stéphanie Delaune, and Jonathan Millen. Constraint solving techniques and enriching the model with equational theories. In Véronique Cortier and Steve Kremer, editors, *Formal Models and Techniques for Analyzing Security Protocols*, volume 5 of *Cryptology and Information Security Series*, pages 35–61. IOS Press, 2011.

[Cio11] Ştefan Ciobâcă. *Automated Verification of Security Protocols with Applications to Electronic Voting*. Thèse de doctorat, Laboratoire Spécification et Vérification, ENS Cachan, France, December 2011.

[CJM00] Edmund M. Clarke, Somesh Jha, and Wilfredo R. Marrero. Partial order reductions for security protocol verification. In *Proceedings of the 6th International Conference on Tools and Algorithms for Construction and Analysis of Systems: Held as Part of the European Joint Conferences on the Theory and Practice of Software, ETAPS 2000*, TACAS '00, pages 503–518, London, UK, UK, 2000. Springer-Verlag.

[CKRT03] Y. Chevalier, R. Küsters, M. Rusinowitch, and M. Turuani. An NP decision procedure for protocol insecurity with XOR. In *Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science (LICS'03)*, pages 261–270, Ottawa (Canada), 2003. IEEE Computer Society Press.

[CLC03] Hubert Comon-Lundh and Véronique Cortier. New decidability results for fragments of first-order logic and application to cryptographic protocols. In Robert Nieuwenhuis, editor, *RTA'03*, volume 2706 of *LNCS*, pages 148–164, Heidelberg, 2003. Springer.

[CLCZ10] Hubert Comon-Lundh, Véronique Cortier, and Eugen Zalinescu. Deciding security properties of cryptographic protocols. application to key cycles. *Transaction on Computational Logic*, 11(2), 2010.

[CLS03] H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of Exclusive Or. In *Proc. 18th Annual IEEE Symposium on Logic in Computer Science (LICS '03)*, pages 271–280, Los Alamitos, CA, 2003. IEEE Computer Society.

[CR05] Y. Chevalier and M. Rusinowitch. Combining intruder theories. In *Proc. 32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 of *LNCS*, pages 639–651. Springer, 2005.

[CR12] Yannick Chevalier and Michaël Rusinowitch. Decidability of equivalence of symbolic derivations. *J. Autom. Reasoning*, 48(2):263–292, 2012.

[Cre08] Cas J.F. Cremers. Unbounded verification, falsification, and characterization of security protocols by pattern refinement. In *CCS '08: Proceedings of the 15th ACM conference on Computer and communications security*, pages 119–128, New York, NY, USA, 2008. ACM.

[CZ06] Véronique Cortier and Eugen Zalinescu. Deciding key cycles for security protocols. In *Proc. 13th Inter. Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'06)*, volume 4246 of *LNCS*, pages 317–331. Springer, 2006.

[DH76] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[DJ90] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In *Handbook of Theoretical Computer Science*, volume B, chapter 6. Elsevier, 1990.

[DJ06] Stéphanie Delaune and Florent Jacquemard. Decision procedures for the security of protocols with probabilistic encryption against offline dictionary attacks. *Journal of Automated Reasoning*, 36(1-2):85–124, January 2006.

[DKR07] S. Delaune, S. Kremer, and M. D. Ryan. Symbolic bisimulation for the applied pi-calculus. In *Proc. 27th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'07)*, pages 133–145, 2007.

[DKR08] Stéphanie Delaune, Steve Kremer, and Mark D. Ryan. Composition of password-based protocols. In *Proc. 21st IEEE Computer Security Foundations Symposium (CSF'08)*, pages 239–251. IEEE Computer Society Press, 2008.

[DKR09] Stéphanie Delaune, Steve Kremer, and Mark D. Ryan. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security*, 17(4):435–487, July 2009.

[DLMS99] N. Durgin, P. Lincoln, J. Mitchell, and A. Scedrov. Undecidability of bounded security protocols. In *Workshop on Formal Methods and Security Protocols*, 1999.

[DSV03] Luca Durante, Riccardo Sisto, and Adriano Valenzano. Automatic testing equivalence verification of spi calculus specifications. *ACM Transactions on Software Engineering and Methodology*, 12(2):222–284, 2003.

[Eng85] Joost Engelfriet. Determinacy implies (observation equivalence = trace equivalence). *Theoretical Computer Science*, 36:21–25, 1985.

[FHG99] F. Javier Thayer Fábrega, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(2/3):191–230, 1999.

[Gam85] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[GML+93] Li Gong, T. Mark, T. Mark A. Lomas, Roger M. Needham, and Jerome H. Saltzer. Protecting poorly chosen secrets from guessing attacks. *IEEE Journal on Selected Areas in Communications*, 11:648–656, 1993.

[GT00] Joshua D. Guttman and F. Javier Thayer. Protocol independence through disjoint encryption. In *Proc. 13th Computer Security Foundations Workshop (CSFW'00)*, pages 24–34. IEEE Comp. Soc. Press, 2000.

[Hut02] H. Huttel. Deciding framed bisimulation. In *4th International Workshop on Verification of Infinite State Systems INFINITY'02*, pages 1–20, 2002.

[ICA04] PKI for machine readable travel documents offering ICC read-only access. Technical report, International Civil Aviation Organization, 2004.

[ISO09] ISO 15408-2: Common Criteria for Information Technology Security Evaluation - Part 2: Security functional components. Final draft, ISO/IEC, July 2009.

[JK91] Jean-Pierre Jouannaud and Claude Kirchner. Solving equations in abstract algebras: A rule-based survey of unification. In *Computational Logic - Essays in Honor of Alan Robinson*, pages 257–321, 1991.

[KT11] Ralf Küsters and Max Tuengerthal. Composition Theorems Without Pre-Established Session Identifiers. In *Proc. 18th Conference on Computer and Communications Security (CCS 2011)*, pages 41–50. ACM Press, 2011.

[Liu11] Jia Liu. A proof of coincidence of labeled bisimilarity and observational equivalence in applied pi calculus. Technical report, Institute of Software Chinese Academy of Sciences, State Key Laboratory of Computer Science, april 2011.

[MNP02] M.Boreale, R. De Nicola, and R. Pugliese. Proof techniques for cryptographic processes. *SIAM Journal on Computing*, 31(3):947–986, 2002.

[MS01] Jonathan K. Millen and Vitaly Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *ACM Conference on Computer and Communications Security*, pages 166–175, 2001.

[MV09] Sebastian Mödersheim and Luca Viganò. Secure pseudonymous channels. In *Proc. 14th European Symposium on Research in Computer Security (ESORICS'09)*, volume 5789 of *LNCS*, pages 337–354. Springer, 2009.

[NH84] Rocco De Nicola and Matthew Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.

[RSG+00] Peter Ryan, Steve Schneider, Michael Goldsmith, Gavin Lowe, and Bill Roscoe. *The Modelling and Analysis of Security Protocols*. Addison Wesley, 2000.

[RT01] M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is NP-complete. In *Proc. 14th Computer Security Foundations Workshop (CSFW'01)*, pages 174–190. IEEE Comp. Soc. Press, 2001.

[TD10] Alwen Tiu and Jeremy E. Dawson. Automating open bisimulation checking for the spi calculus. In *Proc. 23rd IEEE Computer Security Foundations Symposium (CSF'10)*, pages 307–321. IEEE Computer Society Press, 2010.

[Vig06] L. Viganò. Automated security protocol analysis with the avispa tool. In *Proceedings of the XXI Mathematical Foundations of Programming Semantics (MFPS'05)*, volume 155 of *ENTCS*, pages 61–86. Elsevier, 2006.

[Wei99] Christoph Weidenbach. Towards an automatic analysis of security protocols in first-order logic. In Harald Ganzinger, editor, *CADE'99*, volume 1632 of *LNAI*, pages 314–328, Heidelberg, 1999. Springer.

Appendices

Appendix A

From the applied pi calculus to symbolics semantics

A.1 Proofs on relating equivalence

Proposition 3.1. *Let A and B be two closed extended process with* $\mathrm{dom}(A) = \mathrm{dom}(B)$*, and $C[\_] = \nu\tilde{n}.(D \mid \_)$ be an evaluation context closing for A. If $C[A] \overset{\epsilon}{\Rightarrow} A''$ for some process A'', then there exist a closed extended process A', an evaluation context $C' = \nu\tilde{n'}.(D' \mid \_)$ closing for A', and a trace $\mathsf{tr} \in (\mathcal{A} \setminus \{\tau\})^*$ such that $A'' \equiv C'[A']$, $A \overset{\mathsf{tr}}{\Rightarrow} A'$, and for all closed extended process B',*

$$B \overset{\mathsf{tr}}{\Rightarrow} B' \text{ and } \Phi(B') \sim \Phi(A') \text{ imply that } C[B] \overset{\epsilon}{\Rightarrow} C'[B'].$$

Proof. Let A and B be two closed extended processes with $\mathrm{dom}(A) = \mathrm{dom}(B)$ and C be an evaluation context closing for A. Let A'' be such that $C[A] \overset{\epsilon}{\Rightarrow} A''$. We prove the result by induction on the length ℓ of the derivation.

Base case $\ell = 0$: In such a case, we have that $A'' \equiv C[A]$. Let $A' = A$, $C' = C$ and $\mathsf{tr} = \epsilon$, we have that $A'' \equiv C'[A']$, and $A \overset{\epsilon}{\Rightarrow} A'$. Let B' be a closed extended process such that $B \overset{\epsilon}{\Rightarrow} B'$ and $\Phi(B') \sim \Phi(A')$ for some B'. Clearly, we have that $C[B] \overset{\epsilon}{\Rightarrow} C'[B']$ since $C' = C$ and $B \overset{\epsilon}{\Rightarrow} B'$.

Inductive case $\ell > 0$: In such a case, we have that there exists a closed extended process A_1 such that $C[A] \overset{\epsilon}{\Rightarrow} A_1$ with a derivation whose length is smaller than ℓ, and $A_1 \overset{\tau}{\mapsto} A''$. Thus, we can apply our induction hypothesis allowing us to conclude that there exist an extended process A'_1, an evaluation context $C'_1[\_] = \nu\tilde{n'_1}.(D'_1 \mid \_)$ closing for A'_1, and a trace $\mathsf{tr}_1 \in (\mathcal{A} \setminus \{\tau\})^*$ such that $A_1 \equiv C'_1[A'_1]$, $A \overset{\mathsf{tr}_1}{\Rightarrow} A'_1$, and for all closed extended processes B'_1, we have that:

$$B \overset{\mathsf{tr}_1}{\Rightarrow} B'_1 \text{ and } \Phi(B'_1) \sim \Phi(A'_1) \text{ implies that } C[B] \overset{\epsilon}{\Rightarrow} C'_1[B'_1].$$

Since $A_1 \equiv C'_1[A'_1]$ and $A_1 \overset{\tau}{\mapsto} A''$, we have that $C'_1[A'_1] \overset{\tau}{\mapsto} A''$ (internal reduction is closed under structural equivalence). We do a case analysis on the rule involved in this reduction.

Case of an internal reduction in A'_1, i.e. there exists A' such that $A'_1 \overset{\tau}{\mapsto} A'$ and $A'' \equiv C'_1[A']$. In such a case, we have that $C'_1[A'_1] \overset{\tau}{\to} C'_1[A']$. Let $C' = C'_1$ and $\mathsf{tr} = \mathsf{tr}_1$. We have that $A'' \equiv C'_1[A'] = C'[A']$ and $A \overset{\mathsf{tr}_1}{\Rightarrow} A'_1 \overset{\tau}{\mapsto} A'$, *i.e.* $A \overset{\mathsf{tr}}{\Rightarrow} A'$. Lastly, let B' be a closed extended process such that $B \overset{\mathsf{tr}}{\Rightarrow} B'$ and $\Phi(B') \sim \Phi(A')$. We have that $B \overset{\mathsf{tr}_1}{\Rightarrow} B'$ and $\Phi(B') \sim \Phi(A'_1) \equiv \Phi(A')$, and thus relying on our induction hypothesis, we conclude that $C[B] \overset{\epsilon}{\Rightarrow} C'_1[B'] = C'[B']$. This allows us to conclude.

Case of an internal reduction in C'_1, i.e. there exists D'_2 such that $\nu\tilde{n}'_1(D'_1 \mid B) \overset{\tau}{\to} \nu\tilde{n}'_1.(D'_2 \mid B)$ for any process B such that $\Phi(B) \sim \Phi(A'_1)$, and $A'' \equiv \nu\tilde{n}'_1.(D'_2 \mid A'_1)$. In such a case, we have

that $C_1'[A_1'] \overset{\tau}{\mapsto} \nu\tilde{n}_1'.(D_2' \mid A_1')$. Let $A' = A_1'$, $C'[\_] = \nu\tilde{n}_1'.(D_2' \mid \_)$ and $\mathsf{tr} = \mathsf{tr}_1$. We have that $A'' \equiv \nu\tilde{n}_1'.(D_2' \mid A_1') = C'[A']$ and $A \overset{\mathsf{tr}_1}{\Mapsto} A_1' = A'$, *i.e.* $A \overset{\mathsf{tr}}{\Mapsto} A'$. Lastly, let B' be a closed extended process such that $B \overset{\mathsf{tr}}{\Mapsto} B'$ and $\Phi(B') \sim \Phi(A')$. We have that $B \overset{\mathsf{tr}_1}{\Mapsto} B'$ and $\Phi(B') \sim \Phi(A_1') \equiv \Phi(A')$, and thus relying on our induction hypothesis, we have that $C[B] \overset{\epsilon}{\Mapsto} C_1'[B']$. But by our hypothesis on the internal reduction, $\Phi(B') \sim \Phi(A_1')$ implies that $C_1'[B'] \overset{\tau}{\mapsto} \nu\tilde{n}_1'.(D_2' \mid B') = C'[B']$ and so $C[B] \overset{\epsilon}{\Mapsto} C'[B']$. This allows us to conclude.

*Case of a rule (*Comm*) between C_1' (output) and A_1' (input), i.e. $D_1' \equiv \nu\tilde{n}.(\mathsf{out}(c, M).P \mid D)$, $A_1' \equiv A_2' = \nu\tilde{r}.(\mathsf{in}(c, z).Q \mid A_2)$ for some c, M, P, D, A_2', $\tilde{r}$, z, Q, and A_2 such that z is a fresh variable, $fvars(M) \subseteq \mathrm{dom}(A_2')$, $\tilde{r} \cap (fnames(M) \cup fvars(M)) = \emptyset$, and $\tilde{n} \cap (fnames(A_2') \cup fvars(A_2')) = \emptyset$. We assume in addition that names and variables in $\tilde{n}$ do not occur in $fnames(B)$, $fvars(B)$, and* tr. In such a case, we have that

$$\begin{array}{rcl} C_1'[A_1'] & \equiv & \nu\tilde{n}_1'.[\nu\tilde{n}.(\mathsf{out}(c, M).P \mid D) \mid A_2') \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\mathsf{out}(c, M).P \mid D \mid \nu\tilde{r}.(\mathsf{in}(c, z).Q \mid A_2)] \\ & \overset{\tau}{\mapsto} & \nu\tilde{n}_1'.\nu\tilde{n}.[P \mid D \mid \nu\tilde{r}.(Q\{^M/_z\} \mid A_2)] \end{array}$$

and $A'' \equiv \nu\tilde{n}_1', \tilde{n}.[P \mid D \mid \nu\tilde{r}.(Q\{^M/_z\} \mid A_2)]$.

Let $A' = \nu\tilde{r}.(Q\{^M/_z\} \mid A_2)$, $C'[\_] = \nu\tilde{n}_1'.\nu\tilde{n}.(P \mid D \mid \_)$, and $\mathsf{tr} = \mathsf{tr}_1 \cdot \mathsf{in}(c, M)$. We have that $A'' \equiv C'[A']$. By induction hypothesis, we have that $A \overset{\mathsf{tr}_1}{\Mapsto} A_1'$ and $A_1' \overset{\mathsf{in}(c,M)}{\longmapsto} A'$. This allows us to conclude that $A \overset{\mathsf{tr}}{\Mapsto} A'$. Note that A' is a closed extended process ($fvars(M) \subseteq \mathrm{dom}(A_2') = \mathrm{dom}(A_1')$).

Lastly, let B' be a closed extended processes such that $B \overset{\mathsf{tr}}{\Mapsto} B'$ and $\Phi(B') \sim \Phi(A')$. We have that there exists B_1' such that $B \overset{\mathsf{tr}_1}{\Mapsto} B_1' \overset{\mathsf{in}(c,M)}{\longmapsto} B'$. Moreover, we can assume w.l.o.g. that $\tilde{n}$ do not occur in $fnames(B_1')$ and $fvars(B_1')$ since $\tilde{n}$ do not occur in $fnames(B)$, $fvars(B)$ and tr_1. Since $\Phi(B') \sim \Phi(A')$, we have also that $\Phi(B_1') \sim \Phi(A_1')$. Thus, we can apply our induction hypothesis on B_1'. This allows us to deduce that $C[B] \overset{\epsilon}{\Mapsto} C_1'[B_1']$. In order to conclude, it remains to show that $C_1'[B_1'] \overset{\tau}{\mapsto} C'[B']$.

We have seen that $B_1' \overset{\mathsf{in}(c,M)}{\longmapsto} B'$. Hence, we know that there exists $\tilde{m}, P_2, B_2$ such that $B_1' \equiv \nu\tilde{m}.(\mathsf{in}(c, z).P_2 \mid B_2)$, $B' \equiv \nu\tilde{m}.(P_2\{^M/_z\} \mid B_2)$, and $\tilde{m} \cap (fvars(M) \cup fnames(M)) = \emptyset$. Moreover, we have already seen that $\tilde{n} \cap (fnames(B_1') \cup fvars(B_1')) = \emptyset$. Hence, we have that:

$$\begin{array}{rcl} C_1'[B_1'] & \equiv & \nu\tilde{n}_1'.[\nu\tilde{n}.(\mathsf{out}(c, M).P \mid D) \mid B_1'] \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\mathsf{out}(c, M).P \mid D \mid B_1'] \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\mathsf{out}(c, M).P \mid D \mid \nu\tilde{m}.(\mathsf{in}(c, z).P_2 \mid B_2)] \\ & \to & \nu\tilde{n}_1'.\nu\tilde{n}.(P \mid D \mid \nu\tilde{m}.(P_2\{^M/_z\} \mid B_2)) \\ & \equiv & C'[B'] \end{array}$$

*Case of a rule (*Comm*) between C_1' (input) and A_1' (output), i.e. $D_1' \equiv \nu\tilde{n}.(\mathsf{in}(c, z).P \mid D)$, $A_1' \equiv A_2' = \nu\tilde{r}.(\mathsf{out}(c, M).Q \mid A_2)$ for some c, M, P, Q, D, A_2', $\tilde{r}$, z, and A_2 such that z is a fresh variable, $fvars(M) = \emptyset$, $\tilde{n} \cap (fnames(A_2') \cup fvars(A_2')) = \emptyset$, $\tilde{r} \cap (fnames(\mathsf{in}(c, z).P) \cup fvars(\mathsf{in}(c, z).P)) = \emptyset$. We assume in addition that names and variables in $\tilde{n}$ do not occur in $fnames(B)$, $fvars(B)$, and* tr. In such a case, we have that :

$$\begin{array}{rcl} C_1'[A_1'] & \equiv & \nu\tilde{n}_1'.[\nu\tilde{n}.(\mathsf{in}(c, z).P \mid D) \mid A_2'] \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.(\mathsf{in}(c, z).P \mid D \mid A_2') \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\mathsf{in}(c, z).P \mid D \mid \nu\tilde{r}.(\mathsf{out}(c, M).Q \mid A_2)] \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\nu\tilde{r}.(P\{^M/_z\} \mid Q \mid A_2) \mid D] \end{array}$$

and $A'' \equiv \nu\tilde{n}_1'.\nu\tilde{n}.[\nu\tilde{r}.(P\{^M/_z\} \mid Q \mid A_2) \mid D]$.

To determine C', A' and tr, we distinguish several cases depending on the term M:

— *M is a name of channel type and $M \notin \tilde{r}$.* In such a case, we have that $A'' \equiv \nu\tilde{n}_1'.\nu\tilde{n}.[P\{^M/_z\} \mid D \mid \nu\tilde{r}.(Q \mid A_2)]$. Thus, let $C'[\_] = \nu\tilde{n}_1'.\nu\tilde{n}.[P\{^M/_z\} \mid D \mid \_]$, $A' = \nu\tilde{r}.(Q \mid A_2)$ and $\mathsf{tr} = \mathsf{tr}_1 \cdot \mathsf{out}(c, M)$. Clearly, we have that $A'' \equiv C'[A']$ and $A_1' \xmapsto{\mathsf{out}(c,M)} A'$.

Lastly, let B' be a closed process such that $B \overset{\mathsf{tr}}{\Mapsto} B'$ and $\Phi(B') \sim \Phi(A')$. We have that there exists B_1' such that $B \overset{\mathsf{tr}_1}{\Mapsto} B_1' \xmapsto{\mathsf{out}(c,M)} B'$. Moreover, we can assume w.l.o.g. that $\tilde{n}$ do not occur in $\mathit{fnames}(B_1')$, $\mathit{fvars}(B_1')$, and tr_1. We have also that $\Phi(A_1') \sim \Phi(B_1')$. Thus, we can apply our inductive hypothesis on B_1' which means that $C[B] \mapsto^* C_1'[B_1']$. In order to conclude, it remains to show that $C_1'[B_1'] \to C'[B']$.

We have seen that $B_1' \xmapsto{\mathsf{out}(c,M)} B'$. Hence, we know that there exists $\tilde{m}, Q_2, B_2$ such that $B_1' \equiv \nu\tilde{m}.(\mathsf{out}(c, M).Q_2 \mid B_2)$, $B' \equiv \nu\tilde{m}.(Q_2 \mid B_2)$, $M \notin \tilde{m}$ and $\tilde{m} \cap (\mathit{fvars}(D_1') \cup \mathit{fnames}(D_1') = \emptyset$. Therefore, we have that:

$$\begin{array}{rcl} C_1'[B_1'] & \equiv & \nu\tilde{n}_1'.[\nu\tilde{n}.(\mathsf{in}(c, z).P \mid D) \mid B_1') \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\mathsf{in}(c, z).P \mid D \mid B_1'] \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\mathsf{in}(c, z).P \mid D \mid \nu\tilde{m}.(\mathsf{out}(c, M).Q_2 \mid B_2)] \\ & \overset{\tau}{\mapsto} & \nu\tilde{n}_1'.\nu\tilde{n}.[P\{^M/_z\} \mid D \mid \nu\tilde{m}.(Q_2 \mid B_2)] \\ & \equiv & C'[B'] \end{array}$$

— *M is a name of channel type that occurs in $\tilde{r}$.* Let $\tilde{r}'$ be a sequence such that $\tilde{r} = \tilde{r'} \uplus M$. In such a case, we have that $A'' \equiv \nu\tilde{n}_1'.\nu\tilde{n}.\nu M'.[P\{^{M'}/_z\} \mid D \mid \nu\tilde{r}'.(Q\{^{M'}/_M\} \mid A_2\{^{M'}/_M\})]$ where M' is a fresh name of channel type. Let $C'[\_] = \nu\tilde{n}_1'.\nu\tilde{n}.\nu M'.[P\{^{M'}/_z\} \mid D \mid \_]$, $A' = \nu\tilde{r}'.(Q\{^{M'}/_M\} \mid A_2\{^{M'}/_M\})$ and $\mathsf{tr} = \mathsf{tr}_1 \cdot \nu M'.\mathsf{out}(c, M')$. Clearly, we have that $A_1' \equiv \nu\tilde{r}'.\nu M.(\mathsf{out}(c, M).Q \mid A_2) \xmapsto{\nu M'.\mathsf{out}(c,M')} \nu\tilde{r}'.(Q\{^{M'}/_M\} \mid A_2\{^{M'}/_M\}) \equiv A'$ and $A'' \equiv C'[A']$.

Lastly, let B' be a closed process such that $B \overset{\mathsf{tr}}{\Mapsto} B'$ and $\Phi(B') \sim \Phi(A')$. We have that there exists B_1' such that $B \overset{\mathsf{tr}_1}{\Mapsto} B_1' \xmapsto{\nu M'.\mathsf{out}(c,M')} B'$. Moreover, we can assume w.l.o.g. that $\tilde{n}$ do not occur in $\mathit{fnames}(B_1')$, $\mathit{fvars}(B_1')$, and tr_1. We also have that $\Phi(A_1') \sim \Phi(B_1')$. Thus, we can apply our induction hypothesis on B_1' which means that $C[B] \overset{\epsilon}{\mapsto} C_1'[B_1']$. In order to conclude, it remains to show that $C_1'[B_1'] \overset{\tau}{\mapsto} C'[B']$.

We have seen that $B_1' \xmapsto{\nu M'.\mathsf{out}(c,M')} B'$. Hence, we know that there exists $\tilde{m}, Q_2, B_2$ such that $B_1' \equiv \nu M'.\nu\tilde{m}.(\mathsf{out}(c, M').Q_2 \mid B_2)$, $B' \equiv \nu\tilde{m}.(Q_2 \mid B_2)$, $M' \notin \tilde{m}$ and $\tilde{m} \cap (\mathit{fvars}(D_1') \cup \mathit{fnames}(D_1') = \emptyset$. Therefore, we have that (where M'' is a fresh channel name):

$$\begin{array}{rcl} C_1'[B_1'] & \equiv & \nu\tilde{n}_1'.[\nu\tilde{n}.(\mathsf{in}(c, z).P \mid D) \mid B_1') \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\mathsf{in}(c, z).P \mid D \mid B_1'] \\ & \equiv & \nu\tilde{n}_1'.\nu\tilde{n}.[\mathsf{in}(c, z).P \mid D \mid \nu M'.\nu\tilde{m}.(\mathsf{out}(c, M').Q_2 \mid B_2)] \\ & \overset{\tau}{\mapsto} & \nu\tilde{n}_1'.\nu\tilde{n}.\nu M''.[P\{^{M''}/_z\} \mid D \mid \nu\tilde{m}.(Q_2\{^{M''}/_{M'}\} \mid B_2\{^{M''}/_{M'}\})] \\ & \equiv & C'[B'] \end{array}$$

— *M is a term of base type.* In such a case, we have that $A'' \equiv \nu\tilde{n}_1'.\nu\tilde{n}.\nu z.[P \mid D \mid \nu\tilde{r}.(Q \mid A_2 \mid \{^M/_z\})]$. Let $C'[\_] = \nu\tilde{n}_1'.\nu\tilde{n}.\nu z.[P \mid D \mid \_]$, $A' = \nu\tilde{r}.(Q \mid A_2 \mid \{^M/_z\})$ and $\mathsf{tr} = \mathsf{tr}_1 \cdot \nu z.\mathsf{out}(c, z)$. Clearly, we have that $A_1' \equiv \nu\tilde{r}.(\mathsf{out}(c, M).Q \mid A_2) \xmapsto{\nu z.\mathsf{out}(c,z)} \nu\tilde{r}.(Q \mid A_2 \mid \{^M/_z\}) \equiv A'$ and $A'' \equiv C'[A']$.

Lastly, let B' be a closed process such that $B \overset{\mathsf{tr}}{\Mapsto} B'$ and $\Phi(B') \sim \Phi(A')$. We have that there exists B_1' such that $B \overset{\mathsf{tr}_1}{\Mapsto} B_1' \xmapsto{\nu z.\mathsf{out}(c,z)} B'$. Moreover, we can assume w.l.o.g. that $\tilde{n}$ do not occur in $\mathit{fnames}(B_1')$, $\mathit{fvars}(B_1')$, and tr_1. We also have that $\Phi(A_1') \sim \Phi(B_1')$. Thus, we can apply our induction hypothesis on B_1' which means that $C[B] \overset{\tau}{\mapsto} C_1'[B_1']$. In order to conclude, it remains to show that $C_1'[B_1'] \overset{\tau}{\mapsto} C'[B']$.

We have seen that $B_1' \xmapsto{\nu z.\mathsf{out}(c,z)} B'$. Hence, we know that there exists $\tilde{m}, Q_2, B_2$ such that $B_1' \equiv \nu\tilde{m}.(\mathsf{out}(c, M).Q_2 \mid B_2)$, $B' \equiv \nu\tilde{m}.(Q_2 \mid B_2 \mid \{^M/_z\})$ and $\tilde{m} \cap (\mathit{fvars}(D_1') \cup$

$fnames(D'_1) = \emptyset$. Therefore, we have that:

$$\begin{array}{rcl} C'_1[B'_1] & \equiv & \nu\tilde{n}'_1.[\nu\tilde{n}.(\mathsf{in}(c,z).P \mid D) \mid B'_1) \\ & \equiv & \nu\tilde{n}'_1.\nu\tilde{n}.[\mathsf{in}(c,z).P \mid D \mid B'_1] \\ & \equiv & \nu\tilde{n}'_1.\nu\tilde{n}.[\mathsf{in}(c,z).P \mid D \mid \nu\tilde{m}.(\mathsf{out}(c,M).Q_2 \mid B_2)] \\ & \overset{\tau}{\mapsto} & \nu\tilde{n}'_1.\nu\tilde{n}.\nu z.[P \mid D \mid \nu\tilde{m}.(Q_2 \mid B_2 \mid \{^M/_z\})] \\ & \equiv & C'[B'] \end{array}$$

□

A.2 Proofs on symbolic semantics

In this appendix, we provide the complete proof of Proposition 4.2 and 4.3.

Proposition 4.2 (soundness)**.** *Let* $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1; D_1; Eq_1)$*, and* $(\mathcal{E}_2; \mathcal{P}_2; \Phi_2; D_2; Eq_2)$ *be two symbolic processes such that*

— $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1; D_1; Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_2; \mathcal{P}_2; \Phi_2; D_2; Eq_2)$*, and*

— $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$.

Let $\theta_1 = \theta_2|_{vars^2(D_1)}$ *and* $\sigma_1 = \sigma_2|_{vars^1(D_1)}$*. We have that:*

1. $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$*, and*
2. $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$

Proof. The proof consists of a case analysis on the rule involved in the reduction step of:

$$(\mathcal{E}_1; \mathcal{P}_1; \Phi_1; D_1; Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_2; \mathcal{P}_2; \Phi_2; D_2; Eq_2)$$

Case THEN$_s$**:** In such a case there exist u, v, Q_1, and Q_2 such that $D_2 = D_1$, $Eq_2 = Eq_1 \cup \{u \overset{?}{=} v\}$, $\mathcal{P}_1 = \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}$, $\mathcal{P}_2 = \{Q_1\} \uplus \mathcal{P}$, $\mathcal{E}_1 = \mathcal{E}_2$, $\Phi_1 = \Phi_2$, and $\alpha_s = \tau$

1. Since $vars^2(D_1) = vars^2(D_2)$ and $vars^1(D_1) = vars^1(D_2)$, we have $\theta_1 = \theta_2$ and $\sigma_1 = \sigma_2$. Furthermore, $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$ and $\Phi_1 = \Phi_2$ implies that (σ_2, θ_2) satisfies the constraints of $D_2 = D_1$ and the (in)equalities of $Eq_2 = Eq_1 \cup \{u \overset{?}{=} v\}$, and so satisfies Eq_1. At last, with $\mathcal{E}_1 = \mathcal{E}_2$, we conclude that $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$.
2. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_1 \cup \{u \overset{?}{=} v\}))$, we have that $u\sigma_2 =_{\mathsf{E}} v\sigma_2$. With $\sigma_1 = \sigma_2$, we deduce that $u\sigma_1 =_{\mathsf{E}} v\sigma_1$ and so:

$$(\mathcal{E}_1; \{\mathsf{if}\ u\sigma_1 = v\sigma_1\ \mathsf{then}\ Q_1\sigma_1\ \mathsf{else}\ Q_2\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1) \xrightarrow{\tau}_i (\mathcal{E}_1; \{Q_1\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1),$$

i.e. $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$.

The case of the rule ELSE$_s$ can be done in a similar way.

Case COMM$_s$**:** In such a case, there exist p, q, u, x, Q_1, Q_2, and $\mathcal{P}$ such that $D_2 = D_1$, $Eq_2 = Eq_1 \cup \{p \overset{?}{=} q\}$, $\Phi_2 = \Phi_1$, $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_1 = \{\mathsf{out}(p,u).Q_1; \mathsf{in}(q.x).Q_2\} \uplus \mathcal{P}$, $\mathcal{P}_2 = \{Q_1; Q_2\{x \mapsto u\}\} \uplus \mathcal{P}$, and $\alpha_s = \tau$.

1. Since $vars^2(D_1) = vars^2(D_2)$ and $vars^1(D_1) = vars^1(D_2)$, we have $\theta_1 = \theta_2$ and $\sigma_1 = \sigma_2$. Furthermore, $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$, and $\Phi_1 = \Phi_2$ implies that (σ_2, θ_2) satisfies the constraints of $D_2 = D_1$ and the (in)equalities of $Eq_2 = Eq_1 \cup \{p \overset{?}{=} q\}$, and so satisfies Eq_1. At last, with $\mathcal{E}_1 = \mathcal{E}_2$, we conclude that $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$.

2. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_1 \cup \{p \stackrel{?}{=} q\}))$, we have that $p\sigma_2 =_{\mathsf{E}} q\sigma_2$. With $\sigma_1 = \sigma_2$, we deduce that $p\sigma_1 =_{\mathsf{E}} q\sigma_1$. But p and q are of channel type, then so does $p\sigma_1$ and $q\sigma_1$ which means that $p\sigma_1 = q\sigma_1$. Hence, we have that:

$$(\mathcal{E}_1; \{\mathsf{out}(p\sigma_1, u\sigma_1).Q_1\sigma_1; \mathsf{in}(p\sigma_1.x).Q_2\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1) \xrightarrow{\tau}_i (\mathcal{E}_1; \{Q_1\sigma_1; Q_2\sigma_1\{x \mapsto u\sigma_1\}\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1).$$

Since $\sigma_1 = \sigma_2$, $\Phi_1 = \Phi_2$ and $Q_2\sigma_2\{x \mapsto u\sigma_2\} = (Q_2\{x \mapsto u\})\sigma_2$, we can deduce that $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$.

Case $\textsc{Par}_s$**:** In such a case, there exist P, Q, and $\mathcal{P}$ such that $D_2 = D_1$, $Eq_2 = Eq_1$, $\Phi_2 = \Phi_1$, $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_1 = \{P \mid Q\} \uplus \mathcal{P}$, $\mathcal{P}_2 = \{P; Q\} \uplus \mathcal{P}$, and $\alpha_s = \tau$.

1. Since $vars^2(D_1) = vars^2(D_2)$ and $vars^1(D_1) = vars^1(D_2)$, we have $\theta_1 = \theta_2$ and $\sigma_1 = \sigma_2$. Furthermore, we have that $(\mathcal{E}_1; \Phi_1; D_1; Eq_1) = (\mathcal{E}_2; \Phi_2; D_2; Eq_2)$ hence we trivially have that $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$.
2. We have that:

$$(\mathcal{E}_1; \{P\sigma_1 \mid Q\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1) \xrightarrow{\tau}_i (\mathcal{E}_1; \{P\sigma_1; Q\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1)$$

Since $\sigma_1 = \sigma_2$, we can deduce that $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$.

Case $\textsc{In}_s$**.** In such a case, there exist p, x, Q, $\mathcal{P}$ and fresh variables y, z and Y, Z and such that $\alpha_s = \mathsf{in}(Z, Y)$, $\mathcal{P}_1 = \{\mathsf{in}(p, x).Q\} \uplus \mathcal{P}$, $\mathcal{P}_2 = \{Q\{x \mapsto y\}\} \uplus \mathcal{P}$, $D_2 = D_1 \cup \{Y, n \stackrel{?}{\vdash} y \; ; Z, n \stackrel{?}{\vdash} z\}$, $Eq_2 = Eq_1 \cup \{z \stackrel{?}{=} p\}$, $\mathcal{E}_1 = \mathcal{E}_2$ and $\Phi_1 = \Phi_2$ where $n = |\Phi_1|$.

1. We have that $vars^2(D_1) = vars^2(D_2) \setminus \{Y \; ; Z\}$, $vars^1(D_1) = vars^1(D_2) \setminus \{y \; ; z\}$ and $\Phi_1 = \Phi_2$. But $(\mathcal{E}; \Phi_1; D_1; Eq_1)$ is a constraint system hence $vars^1(\Phi_1) \subseteq vars^1(D_1)$ which implies that $\Phi_1\sigma_1 = \Phi_2\sigma_2$. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$ and $D_1 \subseteq D_2$, we have that for all $(X, i \stackrel{?}{\vdash} x) \in D_1$, $(X\theta_2)(\Phi_2\sigma_2) = x\sigma_2$. With $\Phi_1\sigma_1 = \Phi_2\sigma_2$, we deduce that $(X\theta_1)(\Phi_1\sigma_1) = x\sigma_1$. Furthermore, we know that σ_2 satisfies the (in)equalities of $Eq_2 = Eq_1 \cup \{z \stackrel{?}{=} p\}$ and so satisfies also the (in)equalities of Eq_1. Since $(\mathcal{E}; \Phi_1; D_1; Eq_1)$ being a constraint system also implies $vars^1(Eq_1) \subseteq vars^1(D_1)$, we deduce that σ_1 satisfies the (in)equalities of Eq_1 and so we conclude that $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$.
2. Let $M = Y\theta_2$ and $u = y\sigma_2$. Thanks to $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$ and $\mathcal{E}_2 = \mathcal{E}_1$, we have that $fnames(M) \cap \mathcal{E}_1 = \emptyset$. We have also that $u = y\sigma_2 = M(\Phi_2\sigma_2)$. We already know that $\Phi_1\sigma_1 = \Phi_2\sigma_2$, thus we have that $u = M(\Phi_2\sigma_2) = M(\Phi_1\sigma_1)$. Futhermore, by definition of a solution, we have that $fvars(M) \subseteq \mathrm{dom}(\Phi_1)$. Lastly, since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$, we have that $(Z\theta_2)(\Phi_2\sigma_2) = z\sigma_2$, $z\sigma_2 =_{\mathsf{E}} p\sigma_2$ and $Z\theta_2 \in \mathcal{T}(\mathcal{N} \setminus \{\mathcal{E}_2\}, \mathrm{dom}(\Phi_2))$. But p is a term of type channel, thus so does $p\sigma_2$. Since all the function symbol operate on and return term of base type and since all terms in Φ_2 are base type, we can deduce that $(Z\theta_2) \in \mathcal{N} \setminus \{\mathcal{E}_2\}$ and so $Z\theta_2 = p\sigma_2$ with $p\sigma_2 \notin \mathcal{E}_1$ $(\mathcal{E}_1 = \mathcal{E}_2)$. Furthermore, since $\sigma_1 = \sigma_2|_{vars^1(D_1)}$ and $vars^1(D_1) = vars^1(D_2) \setminus \{y \; ; z\}$ where y, z are fresh variables, then p is either a name or a variable in $vars^1(D_1)$ and we have that $p\sigma_2 = p\sigma_1$. Hence, we have that

$$(\mathcal{E}_1; \{\mathsf{in}(p\sigma_1, x).Q\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1) \xrightarrow{in(Z\theta_2, M)}_i (\mathcal{E}_1; \{Q\sigma_1\{x \mapsto u\}\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1),$$

i.e. $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2; \{Q\{x \mapsto y\})\sigma_2\} \uplus \mathcal{P}\sigma_2; \Phi_2\sigma_2)$ since $\mathcal{E}_2 = \mathcal{E}_1$, $\Phi_2\sigma_2 = \Phi_1\sigma_1$ and $\sigma_2 = \sigma_1 \cup \{y \mapsto u \; ; z \mapsto p\sigma_1\}$. Hence, we have that $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$.

Case $\textsc{Out-T}_s$**.** In such a case, there exist p, u, Q, $\mathcal{P}$, fresh variables Z, z and $ax_{n+1} \in \mathcal{AX}$ such that $n = |\Phi_1|$, $\alpha_s = \nu ax_n.out(Z, ax_n)$, $\mathcal{P}_1 = \{\mathsf{out}(p, u).Q\} \uplus \mathcal{P}$, $\mathcal{P}_2 = \{Q\} \uplus \mathcal{P}$, $D_2 = D_1 \cup \{Z, n \stackrel{?}{\vdash} z\}$, $Eq_2 = Eq_1 \cup \{z \stackrel{?}{=} p\}$, $\mathcal{E}_2 = \mathcal{E}_1$, and $\Phi_2 = \Phi_1 \cup \{ax_n \rhd u\}$.

1. We know that Z and z are fresh variables. Hence we have that $vars^2(D_1) = vars^2(D_2) \setminus \{Z\}$, $vars^1(D_1) = vars^1(D_2) \setminus \{z\}$ and $z \notin vars^1(\Phi_1)$. Thus thanks to $\sigma_1 = \sigma_2|_{vars^1(D_1)}$, we deduce that $\Phi_1\sigma_1 = \Phi_1\sigma_2$.
   Furthermore, we know that $(\mathcal{E}_1; \Phi_1; D_1; Eq_1)$ is a constraint system. Hence, with $D_2 = D_1 \cup \{Z, n \overset{?}{\vdash} z\}$ and $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$, we have that for all $(X, i \overset{?}{\vdash} x) \in D_1$, $i \leq n$ and $param(X\theta_2) \subseteq \{ax_1, \ldots, ax_n\}$. Thus thanks to $\mathcal{E}_1 = \mathcal{E}_2$ and $\theta_1 = \theta_2|_{vars^2 D_2}$, we deduce that for all $(X, i \overset{?}{\vdash} x) \in D_1$, $x\sigma_1 = x\sigma_2 = X\theta_2(\Phi_2\sigma_2) = X\theta_1(\Phi_1\sigma_2) = X\theta_1(\Phi_1\sigma_1)$ and $X\theta_1 \in \mathcal{T}(\mathcal{F}, \mathcal{N} \setminus \{\xi\}, \mathcal{AX})$.
   At last, z being fresh also implies that $z \notin vars^1(Eq_1)$. Hence, for all $(s \overset{?}{=} s') \in Eq_1$ (resp. $s \overset{?}{\neq} s'$), we have that $s\sigma_2 = s\sigma_1$ and $s\sigma_1 = s\sigma_2$. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$, we deduce that $s\sigma_2 =_{\mathsf{E}} s'\sigma_2$ which implies $s\sigma_1 =_{\mathsf{E}} s'\sigma_1$. We can conclude that $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$.
2. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$, we have that $(Z\theta_2)(\Phi\sigma_2) = z\sigma_2$, $z\sigma_2 =_{\mathsf{E}} p\sigma_2$ and $(Z\theta_2) \in \mathcal{T}(\mathcal{F}, \mathcal{N} \setminus \{\mathcal{E}_2\} \cup \mathrm{dom}(\Phi_2))$. Since all the function symbol operate on and return term of base type and since all terms of Φ_2 are base type, we can deduce that $(Z\theta_2) \in \mathcal{N} \setminus \{\mathcal{E}_2\}$ and so $Z\theta_2 = p\sigma_2$ with $p\sigma_2 \notin \mathcal{E}_1$ ($\mathcal{E}_1 = \mathcal{E}_2$). Furthermore, since $\sigma_1 = \sigma_1|_{vars^1(D_1)}$ and $vars^1(D_1) = vars^1(D_2) \setminus \{z\}$ where z is a fresh variable, we have that p is either a name or a variable in $vars^1(D_1)$ and we have that $p\sigma_1 = p\sigma_2$.
   We have that:

$$(\mathcal{E}_1; \{\mathsf{out}(p\sigma_1, u\sigma_1).Q\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1) \xrightarrow{\nu ax_{n+1}.out(Z\theta_2, ax_{n+1})}_i (\mathcal{E}_1; \{Q\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1 \cup \{ax_{n+1} \rhd u\sigma_1\}),$$

   i.e. $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\nu ax_{n+1}.out(Z\theta_2, ax_{n+1})}_i (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$ since $\sigma_2 = \sigma_1 \cup \{z \mapsto p\sigma_1\}$.

Case OUT-CH$_s$**.** In such a case, there exist p, c, Q, $\mathcal{P}$, fresh variables Z, Y, z, y and $\mathcal{P}_1 = \{\mathsf{out}(p, c).Q\} \uplus \mathcal{P}$, $\mathcal{P}_2 = \{Q\} \uplus \mathcal{P}$, $\mathcal{E}_2 = \mathcal{E}_1$, $\Phi_2 = \Phi_1$, $D_2 = D_1 \cup \{Z, n \overset{?}{\vdash} z \,; Y, n \overset{?}{\vdash} y\}$, $Eq_2 = Eq_1 \cup \{z \overset{?}{=} p \,; y \overset{?}{=} c\}$, $\alpha_s = out(Z, Y)$ and $c \notin \mathcal{E}_1$ with $n = |\Phi_1|$.

1. We have that $vars^2(D_1) = vars^2(D_2) \setminus \{Z \,; Y\}$, $vars^1(D_1) = vars^1(D_2) \setminus \{z \,; y\}$ and $\Phi_1 = \Phi_2$. But $(\mathcal{E}; \Phi_1; D_1; Eq_1)$ is a constraint system hence $vars^1(\Phi_1) \subseteq vars^1(D_1)$ which implies that $\Phi_1\sigma_1 = \Phi_2\sigma_2$. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$ and $D_1 \subseteq D_2$, we have that for all $(X, i \overset{?}{\vdash} x) \in D_1$, $(X\theta_2)(\Phi_2\sigma_2) = x\sigma_2$. With $\Phi_1\sigma_1 = \Phi_2\sigma_2$, we deduce that $(X\theta_1)(\Phi_1\sigma_1) = x\sigma_1$. Furthermore, we know that σ_2 satisfies the (in)equalities of $Eq_2 = Eq_1 \cup \{z \overset{?}{=} p \,; y \overset{?}{=} c\}$ and so satisfies also the (in)equalities of Eq_1. Since $(\mathcal{E}; \Phi_1; D_1; Eq_1)$ being a constraint system also implies $vars^1(Eq_1) \subseteq vars^1(D_1)$, we deduce that σ_1 satisfies the (in)equalities of Eq_1 and so we conclude that $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$.
2. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$, we have that $(Y\theta_2)(\Phi_2\sigma_2) = y\sigma_2$, $y\sigma_2 = c\sigma_2$ and $(Y\theta_2) \in \mathcal{T}(\mathcal{F}, \mathcal{N} \setminus \{\mathcal{E}_2\} \cup \mathrm{dom}(\Phi_2))$. Since all the function symbols operate on and return terms of base type and since all terms of Φ_2 are base type, we can deduce that $(Y\theta_2) \in \mathcal{N} \setminus \{\mathcal{E}_2\}$ and so $Y\theta_2 = c\sigma_2$ with $c\sigma_2 \notin \mathcal{E}_1$ ($\mathcal{E}_1 = \mathcal{E}_2$). Furthermore, since $\sigma_1 = \sigma_2|_{vars^1(D_1)}$ and $vars^1(D_1) = vars^1(D_2) \setminus \{z \,; y\}$ where z, y are fresh variables, we have that c is either a name or a variable in $vars^1(D_1)$ and we have that $c\sigma_2 = c\sigma_1$.
   Lastly, $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$ also implies that $(Z\theta_2)(\Phi_2\sigma_2) = z\sigma_2$ and $z\sigma_2 =_{\mathsf{E}} p\sigma_2$. We apply the same reasoning we used with $Y\theta_2$ to prove that $Z\theta_2 = p\sigma_1$ with $p\sigma_1 \notin \mathcal{E}_1$.
   Hence, we have that:

$$(\mathcal{E}_1; \{\mathsf{out}(p\sigma_1, c).Q\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1) \xrightarrow{\mathsf{out}(Z\theta_2, Y\theta_2)}_i (\mathcal{E}_1; \{Q\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1).$$

   Hence, we have that $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$.

Case OPEN-CH$_s$**.** In such a case, there exist p, c, Q, $\mathcal{P}$ and fresh variables Z, z such that $p \notin \mathcal{E}_1$, $c \in \mathcal{E}_1$, $\mathcal{P}_1 = \{\mathsf{out}(p, c).Q\} \uplus \mathcal{P}$, $\mathcal{P}_2 = (\{Q\} \uplus \mathcal{P})\{c \mapsto ch\}$, $\mathcal{E}_2 = \mathcal{E}_1$, $\Phi_2 = \Phi_1$, $D_2 = D_1 \cup \{Z, n \stackrel{?}{\vdash} z\}$, $Eq_2 = Eq_1 \cup \{z \stackrel{?}{=} p\}$, and $\alpha_s = \nu ch_m.out(Z, ch_m)$ with $n = |\Phi_1|$.

1. We have that $vars^2(D_1) = vars^2(D_2) \setminus \{Z\}$, $vars^1(D_1) = vars^1(D_2) \setminus \{z\}$ and $\Phi_1 = \Phi_2$. But $(\mathcal{E}; \Phi_1; D_1; Eq_1)$ is a constraint system hence $vars^1(\Phi_1) \subseteq vars^1(D_1)$ which implies that $\Phi_1\sigma_1 = \Phi_2\sigma_2$. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$ and $D_1 \subseteq D_2$, we have that for all $(X, i \stackrel{?}{\vdash} x) \in D_1$, $(X\theta_2)(\Phi_2\sigma_2) = x\sigma_2$. With $\Phi_1\sigma_1 = \Phi_2\sigma_2$, we deduce that $(X\theta_1)(\Phi_1\sigma_1) = x\sigma_1$. Furthermore, we know that σ_2 satisfies the (in)equalities of $Eq_2 = Eq_1 \cup \{z \stackrel{?}{=} p\}$ and so satisfies also the (in)equalities of Eq_1. Since $(\mathcal{E}; \Phi_1; D_1; Eq_1)$ being a constraint system also implies $vars^1(Eq_1) \subseteq vars^1(D_1)$, we deduce that σ_1 satisfies the (in)equalities of Eq_1 and so we conclude that $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$.
2. Since $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$, we have that $(Z\theta_2)(\Phi_2\sigma_2) = z\sigma_2$, $z\sigma_2 = p\sigma_2$ and $(Z\theta_2) \in \mathcal{T}(\mathcal{N} \setminus \{\mathcal{E}_2\}, \mathrm{dom}(\Phi_2))$. Since all the function symbols operate on and return terms of base type and since all terms of Φ_2 are base type, we can deduce that $(Z\theta_2) \in \mathcal{N} \setminus \{\mathcal{E}_2\}$ and so $Z\theta_2 = p\sigma_2$ with $p\sigma_2 \notin \mathcal{E}_1$ ($\mathcal{E}_1 = \mathcal{E}_2$). Furthermore, since $\sigma_1 = \sigma_2|_{vars^1(D_1)}$ and $vars^1(D_1) = vars^1(D_2) \setminus \{z\}$ where z is a fresh variable, then p is either a name or a variable and we have that $p\sigma_2 = p\sigma_1$. Hence, we have that:

$$(\mathcal{E}_1; \{\mathsf{out}(p\sigma_1, c).Q\sigma_1\} \uplus \mathcal{P}\sigma_1; \Phi_1\sigma_1) \xrightarrow{\nu ch_m.out(Z\theta_2, ch_m)}_i (\mathcal{E}_1; (\{Q\sigma_1\} \uplus \mathcal{P}\sigma_1)\{c \mapsto ch_m\}; \Phi_1\sigma_1).$$

Since ch_m is fresh name, we have that $(\{Q\sigma_1\} \uplus \mathcal{P}\sigma_1)\{c \mapsto ch_m\} = \big((\{Q\} \uplus p)\{c \mapsto ch_m\}\big)\sigma_1$. Hence, we have that $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha_s\theta_2}_i (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$ □

Proposition 4.3 (completeness)**.** *Let $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1; D_1; Eq_1)$ be a symbolic process. Let $(\sigma_1, \theta_1) \in \mathsf{Sol}((\mathcal{E}_1; \Phi_1; D_1; Eq_1))$. Let $(\mathcal{E}; \mathcal{P}; \Phi)$ be an intermediate process such that $(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha}_i (\mathcal{E}; \mathcal{P}; \Phi)$. There exist a symbolic process $(\mathcal{E}_2; \mathcal{P}_2; \Phi_2; D_2; Eq_2)$, a pair of substitutions (σ_2, θ_2), and a symbolic action α_s such that:*

1. $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1; D_1; Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_2; \mathcal{P}_2; \Phi_2; D_2; Eq_2)$;
2. $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$;
3. $(\mathcal{E}; \mathcal{P}; \Phi) = (\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2)$; *and*
4. $\alpha_s\theta_2 = \alpha$.

Proof. The proof consists of a case analysis on the rule involved in the reduction step of:

$$(\mathcal{E}_1; \mathcal{P}_1\sigma_1; \Phi_1\sigma_1) \xrightarrow{\alpha}_i (\mathcal{E}; \mathcal{P}; \Phi).$$

Case THEN$_i$**:** In such a case we have that $\mathcal{E} = \mathcal{E}_1$ and there exist u', v', Q'_1, Q'_2, and $\mathcal{P}'$ such that $u' =_{\mathsf{E}} v'$, $\mathcal{P}_1\sigma_1 = \{\mathsf{if}\ u' = v'\ \mathsf{then}\ Q'_1\ \mathsf{else}\ Q'_2\} \uplus \mathcal{P}'$, $\mathcal{P} = \{Q'_1\} \uplus \mathcal{P}'$, $\Phi = \Phi_1\sigma_1$, and $\alpha = \tau$. Hence we deduce that that there exist u, v, Q_1, Q_2, and $\mathcal{P}_0$ such that $\mathcal{P}_1 = \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}_0$, and thus we have that $u\sigma_1 = u'$, $v\sigma_1 = v'$, $Q_1\sigma_1 = Q'_1$, $Q_2\sigma_1 = Q'_2$, and $\mathcal{P}_0\sigma_1 = \mathcal{P}'$. Let $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_2 = \mathcal{Q}_1 \uplus \mathcal{P}_0$, $\Phi_2 = \Phi_1$, $D_2 = D_1$, $Eq_2 = Eq_1 \cup \{u \stackrel{?}{=} v\}$, $\alpha_s = \tau$, $\theta_2 = \theta_1$ and $\sigma_2 = \sigma_1$. We have that:

1. $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1; D_1; Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_2; \mathcal{P}_2; \Phi_2; D_2; Eq_2)$. Indeed, we have that

   $$(\mathcal{E}_1; \{\mathsf{if}\ u = v\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{P}_0; \Phi_1; D_1; Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_1; \{Q_1\} \uplus \mathcal{P}_0; \Phi_1; D_1; Eq_1 \cup \{u \stackrel{?}{=} v\}).$$

2. We have that $vars^2(D_2) = vars^2(D_1)$, $\mathcal{E}_2 = \mathcal{E}_1$, $D_2 = D_1$ and $\Phi_2 = \Phi_1$. To check that (σ_2, θ_2) is a solution, it remains to show that σ_2 satisfies the (in)equalities in $Eq_2 = Eq_1 \cup \{u \stackrel{?}{=} v\}$. But, we have that $u' =_{\mathsf{E}} v'$ and so$u\sigma_1 =_{\mathsf{E}} v\sigma_1$. With $\sigma_1 = \sigma_2$, we deduce that $u\sigma_2 =_{\mathsf{E}} v\sigma_2$ and so $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$.

3. We have that
$$\begin{array}{rcl}(\mathcal{E}_2;\mathcal{P}_2\sigma_2;\Phi_2\sigma_2) &=& (\mathcal{E}_1;(\{Q_1\}\uplus\mathcal{P}_0)\sigma_1;\Phi_1\sigma_1)\\ &=& (\mathcal{E};\{Q_1'\}\uplus\mathcal{P}';\Phi)\\ &=& (\mathcal{E};\mathcal{P};\Phi).\end{array}$$

4. We have that $\alpha_s\theta_2 = \alpha_s = \alpha$.

The case of the rule $\textsc{Else}_i$ can be done in a similar way.

Case $\textsc{Comm}_i$**:** In such a case we have that $\mathcal{E} = \mathcal{E}_1$ and there exist p', u', x, Q_1', Q_2', and $\mathcal{P}'$ such that $\mathcal{P}_1\sigma_1 = \{\mathsf{out}(p',u').Q_1';\mathsf{in}(p',x).Q_2'\}\uplus\mathcal{P}'$, $\mathcal{P} = \{Q_1';Q_2'\{x\mapsto u'\}\}\uplus\mathcal{P}'$, $\Phi = \Phi_1\sigma_1$, and $\alpha = \tau$. Hence we deduce that that there exist u, p, q, Q_1, Q_2, and $\mathcal{P}_0$ such that $\mathcal{P}_1 = \{\mathsf{out}(p,u).Q_1;\mathsf{in}(q,x).Q_2\}\uplus\mathcal{P}_0$, and thus we have that $u\sigma_1 = u'$, $q\sigma_1 = p\sigma_1 = p'$, $Q_1\sigma_1 = Q_1'$, $Q_2\sigma_1 = Q_2'$, and $\mathcal{P}_0\sigma_1 = \mathcal{P}'$. Let $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_2 = \{Q_1;Q_2\{x\mapsto u\}\}\uplus\mathcal{P}_0$, $\Phi_2 = \Phi_1$, $D_2 = D_1$, $Eq_2 = Eq_1\cup\{p\stackrel{?}{=}q\}$, $\alpha_s = \tau$, $\theta_2 = \theta_1$ and $\sigma_2 = \sigma_1$. We have that:

1. $(\mathcal{E}_1;\mathcal{P}_1;\Phi_1;D_1;Eq_1)\xrightarrow{\alpha_s}_s(\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2)$. Indeed, we have that
$$\begin{array}{l}(\mathcal{E}_1;\{\mathsf{out}(p,u).Q_1;\mathsf{in}(q,x).Q_2\}\uplus\mathcal{P}_0;\Phi_1;D_1;Eq_1)\xrightarrow{\alpha_s}_s\\ \qquad\qquad(\mathcal{E}_1;\{Q_1;Q_2\{x\mapsto u\}\}\uplus\mathcal{P}_0;\Phi_1;D_1;Eq_1\cup\{p\stackrel{?}{=}q\})\end{array}$$

2. We have $\mathcal{E}_2 = \mathcal{E}_1$, $D_2 = D_1$, $Eq_2 = Eq_1\cup\{p\stackrel{?}{=}q\}$ and $\Phi_2 = \Phi_1$. To check that (σ_2,θ_2) is a solution, it remains to show that σ_2 satisfies the (in)equalities in $Eq_2 = Eq_1\cup\{p\stackrel{?}{=}q\}$. But we know that $p\sigma_2 = q\sigma_2$ hence $p\sigma_2 =_{\mathsf{E}} q\sigma_2$ and so $(\sigma_2,\theta_2)\in\mathsf{Sol}((\mathcal{E}_2;\Phi_2;D_2;Eq_2))$.
3. Since x is fresh and $x\notin\mathrm{dom}(\sigma_1)$, we have $Q_2\{x\mapsto u\}\sigma_1 = (Q_2\sigma_1)\{x\mapsto u\sigma_1\}$ and so:
$$\begin{array}{rcl}(\mathcal{E}_2;\mathcal{P}_2\sigma_2;\Phi_2\sigma_2) &=& (\mathcal{E}_1;\{Q_1;Q_2\{x\mapsto u\}\}\uplus\mathcal{P}_0)\sigma_1;\Phi_1\sigma_1)\\ &=& (\mathcal{E};\{Q_1';Q_2'\{x\mapsto u'\}\}\uplus\mathcal{P}';\Phi_1')\\ &=& (\mathcal{E};\mathcal{P};\Phi).\end{array}$$

4. We have that $\alpha_s\theta_2 = \alpha_s = \alpha$.

Case $\textsc{In}_i$**:** In such a case we have that $\mathcal{E} = \mathcal{E}_1$, $\Phi = \Phi_1\sigma_1$ and there exist p', x, Q', $\mathcal{P}'$, M and u such that $p'\notin\mathcal{E}_1$, $\mathcal{P}_1\sigma_1 = \{\mathsf{in}(p',x).Q'\}\uplus\mathcal{P}'$, $\mathcal{P} = \{Q'\{x\mapsto u\}\}\uplus\mathcal{P}'$, $M\Phi_1\sigma_1 = u$, $fvars(M)\subseteq\mathrm{dom}(\Phi_1\sigma_1)$, $fnames(M)\cap\mathcal{E}_1 = \emptyset$, and $\alpha = in(p',M)$. Hence, we deduce that there exist p, Q, $\mathcal{P}_0$ such that $\mathcal{P}_1 = \{\mathsf{in}(p,x).Q\}\uplus\mathcal{P}_0$ with $p\sigma_1 = p'$, $Q\sigma_1 = Q'$ and $\mathcal{P}_0\sigma_1 = \mathcal{P}'$.

Let Y and Z be two second order variables and y,z be two fresh first order variables. Let $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_2 = \{Q\{x\mapsto y\}\}\uplus\mathcal{P}_0$, $\Phi_2 = \Phi_1$, $D_2 = D_1\cup\{Y,n\stackrel{?}{\vdash}y\,;Z,n\stackrel{?}{\vdash}z\}$, $Eq_2 = Eq_1\cup\{z\stackrel{?}{=}p\}$ and $\alpha_s = in(Z,Y)$ with $n = |\Phi_1\sigma_1|$. Let θ_2 be the substitution such that $\theta_2 = \theta_1\cup\{Y\mapsto M\,;Z\mapsto p'\}$ and let σ_2 be the substitution such that $\sigma_2 = \sigma_1\cup\{y\mapsto u\,;z\mapsto p'\}$. We have that:

1. $(\mathcal{E}_1;\mathcal{P}_1;\Phi_1;D_1;Eq_1)\xrightarrow{\alpha_s}_s(\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2)$. Indeed, we have that
$$\begin{array}{l}(\mathcal{E}_1;\{\mathsf{in}(p,x).Q\}\uplus\mathcal{P}_0;\Phi_1;D_1;Eq_1)\xrightarrow{\mathsf{in}(Z,Y)}_s\\ \qquad(\mathcal{E}_1;\{Q\{x\mapsto y\}\}\uplus\mathcal{P}_0;\Phi_1;D_1\cup\{Y,n\stackrel{?}{\vdash}y\,;Z,n\stackrel{?}{\vdash}z\};Eq_1\cup\{z\stackrel{?}{=}p\}).\end{array}$$

2. Thanks to $fnames(\theta_1)\cap\mathcal{E}_1 = \emptyset$, $fnames(M)\cap\mathcal{E}_1 = \emptyset$ and $p'\notin\mathcal{E}_1$, we have $\mathrm{dom}(\theta_2) = vars^2(D_2)$ and $fnames(\theta_2)\cap\mathcal{E}_2 = \emptyset$. Hence we have that $Z\theta_2, Y\theta_2\in\mathcal{T}(\mathcal{N}\setminus\mathcal{E}_2,\mathcal{AX})$. Furthermore, since $n = |\Phi_1\sigma| = |\Phi_1|$, $fvars(M)\subset\mathrm{dom}(\Phi_1)$ and p' is a channel name we deduce that $param(Z\theta_2)\subseteq\{ax_1,\ldots,ax_n\}$ and $param(Y\theta_2)\subseteq\{ax_1,\ldots,ax_n\}$.
Now, it remains to show that σ_2 satisfies the constraints in D_2 and the (in)equalities in Eq_2. Since $\sigma_2 = \sigma_1\cup\{y\mapsto u\,;z\mapsto p'\}$ where y,z are fresh variables, it implies that $\Phi_2\sigma_2 = \Phi_1\sigma_1$, $p\sigma_2 = p\sigma_1$ which allows us to conclude since $(Y\theta_2)(\Phi_2\sigma_2) = M(\Phi_1\sigma_1) = u = y\sigma_2$, $(Z\theta_2)(\Phi_2\sigma_2) = p'(\Phi_2\sigma_2) = p' = z\sigma_2 = p\sigma_2$ and $z\sigma_2 = p\sigma_2$ implies $z\sigma_2 =_{\mathsf{E}} p\sigma_2$.

3. We have that

$$\begin{array}{rcl}(\mathcal{E}_2;\mathcal{P}_2\sigma_2;\Phi_2\sigma_2) & = & (\mathcal{E}_1;\{Q\{x\mapsto y\}\sigma_2\}\uplus\mathcal{P}_0\sigma_2;\Phi_1\sigma_1)\\ & = & (\mathcal{E}_1;\{Q\sigma_1\{x\mapsto u\}\}\uplus\mathcal{P}_0\sigma_1;\Phi_1\sigma_1)\\ & = & (\mathcal{E};\{Q'\{x\mapsto u\}\}\uplus\mathcal{P}';\Phi)\\ & = & (\mathcal{E};\mathcal{P};\Phi).\end{array}$$

4. We have that $\alpha_s\theta_2 = In(Z,Y)\theta_2 = In(p',M) = \alpha$.

Case OUT-T$_i$**:** In such a case we have that $\mathcal{E} = \mathcal{E}_1$ and there exist p', u', Q', and $\mathcal{P}'$ such that $p' \notin \mathcal{E}_1$, $\mathcal{P}_1\sigma_1 = \{\mathsf{out}(p',u').Q'\}\uplus\mathcal{P}'$, $\Phi = \Phi_1\sigma_1\cup\{ax_{n+1}\rhd u'\}$ where $n = |\Phi_1\sigma_1|$, $\mathcal{P} = \{Q'\}\uplus\mathcal{P}'$, and $\alpha = \nu ax_{n+1}.out(p',ax_{n+1})$. Since $\mathcal{P}_1\sigma_1 = \{\mathsf{out}(p',u').Q'\}\uplus\mathcal{P}'$, we deduce that there exist u, p, $\mathcal{Q}$ and $\mathcal{P}_0$ such that $\mathcal{P}_1 = \{\mathsf{out}(p,u).Q\}\uplus\mathcal{P}_0$, with $u\sigma_1 = u'$, $p\sigma_1 = p'$, $Q\sigma_1 = Q'$, and $\mathcal{P}_0\sigma_1 = \mathcal{P}'$.

Let Z be a second order variable and z be a fresh first order variable. Let $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_2 = \mathcal{Q}\uplus\mathcal{P}_0$, $\Phi_2 = \Phi_1\cup\{ax_{n+1}\rhd u\}$, $D_2 = D_1\cup\{Z, n-1\overset{?}{\vdash} z\}$, $Eq_2 = Eq_1\cup\{z\overset{?}{=}p\}$, $\alpha_s = \nu ax_{n+1}.out(Z,ax_{n+1})$, $\theta_2 = \theta_1\cup\{Z\mapsto p'\}$ and $\sigma_2 = \sigma_1\cup\{z\mapsto p'\}$. We have that:

1. $(\mathcal{E}_1;\mathcal{P}_1;\Phi_1;D_1;Eq_1)\xrightarrow{\alpha_s}_s(\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2)$. Indeed, we have that

$$(\mathcal{E}_1;\{\mathsf{out}(p,u).Q\}\uplus\mathcal{P}_0;\Phi_1;D_1;Eq_1)\xrightarrow{\nu ax_{n+1}.out(Z,ax_{n+1})}_s (\mathcal{E}_1;\{Q\}\uplus\mathcal{P}_0;\Phi_1\cup\{ax_{n+1}\rhd u\};D_1\cup\{Z,n\overset{?}{\vdash}z\};Eq_1\cup\{z\overset{?}{=}p\}).$$

2. We have $\mathrm{dom}(\theta_2) = vars^2(D_2)$ and $fnames(\theta_2)\cap\mathcal{E}_2 = \emptyset$ since $fnames(\theta_1)\cap\mathcal{E}_1 = \emptyset$ and $p'\notin\mathcal{E}_1$. Furthermore, for all $(X,i\overset{?}{\vdash}x)\in D_1$, we have that $i\leq n$, $X\theta_2 = X\theta_1$, $x\sigma_1 = x\sigma_2$ and $\Phi_1\sigma_1 = \Phi_1\sigma_2$ thus we deduce that $param(X\theta_2)\subseteq\{ax_1,\ldots,ax_i\}$ and $(X\theta_1)(\Phi_1\sigma_1) = x\sigma_1$ implies $(X\theta_2)(\Phi_2\sigma_2) = x\sigma_2$. Moreover, we know have that $Z\theta_2 = p' = z\sigma_2$ thus we deduce that $(Z\theta_2)(\Phi_2\sigma_2) = z\sigma_2$. At last, we know that $p\sigma_1 = p'$ hence $p\sigma_2 = p' = z\sigma_2$, thus we deduce that $p\sigma_2 =_{\mathsf{E}} z\sigma_2$. We can conclude that $(\sigma_2,\theta_2)\in\mathsf{Sol}((\mathcal{E}_2;\Phi_2;D_2;Eq_2))$.
3. We have that

$$\begin{array}{rcl}(\mathcal{E}_2;\mathcal{P}_2\sigma_2;\Phi_2\sigma_2) & = & (\mathcal{E}_1;\{Q\sigma_1\}\uplus\mathcal{P}_0\sigma_1;\Phi_1\sigma_1\cup\{ax_{n+1}\rhd u\sigma_1\})\\ & = & (\mathcal{E};\{Q'\}\uplus\mathcal{P}';\Phi_1\sigma_1\cup\{ax_{n+1}\rhd u'\})\\ & = & (\mathcal{E};\mathcal{P};\Phi)\end{array}$$

4. We have that $\alpha_s\theta_2 = \nu ax_{n+1}.out(Z,ax_{n+1})\theta_2 = \nu ax_{n+1}.out(p',ax_{n+1}) = \alpha$.

Case OUT-CH$_i$**.** In such a case we have that $\mathcal{E} = \mathcal{E}_1$, $\Phi = \Phi_1\sigma_1$ and there exist p', c', Q', and $\mathcal{P}'$ such that $\mathcal{P}'_1 = \{\mathsf{out}(p',c').Q'\}\uplus\mathcal{P}'$, $\mathcal{P} = \{Q'\}\uplus\mathcal{P}'$, $p',c'\notin\mathcal{E}_1$, and $\alpha = out(p',c')$. Hence, we deduce that there exist p, c, Q and $\mathcal{P}_0$ such that $\mathcal{P}_1 = \{\mathsf{out}(p,c).Q\}\uplus\mathcal{P}_0$ with $c\sigma_1 = c'$, $p\sigma_1 = p'$, $\mathcal{P}_0\sigma_1 = \mathcal{P}'$ and $Q\sigma_1 = Q'$.

Let Z and Y be second order variables and z, y be fresh first order variables. Let $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_2 = \mathcal{Q}\uplus\mathcal{P}_0$, $\Phi_2 = \Phi_1$, $D_2 = D_1\cup\{Z,n\overset{?}{\vdash}z\ ;Y,n\overset{?}{\vdash}y\}$, $Eq_2 = Eq_1\cup\{z\overset{?}{=}p\ ;y\overset{?}{=}c\}$, $\alpha_s = out(Z,Y)$, $\theta_2 = \theta_1\cup\{Z\mapsto p'\ ;Y\mapsto c'\}$ and $\sigma_2 = \{z\mapsto p'\ ;y\mapsto c'\}$ with $n = |\Phi_1\sigma_1|$. We have that:

1. $(\mathcal{E}_1;\mathcal{P}_1;\Phi_1;D_1;Eq_1)\xrightarrow{\alpha_s}_s(\mathcal{E}_2;\mathcal{P}_2;\Phi_2;D_2;Eq_2)$. Indeed, we have that

$$(\mathcal{E}_1;\{\mathsf{out}(p,c).Q\}\uplus\mathcal{P}_0;\Phi_1;D_1;Eq_1)\xrightarrow{\mathsf{out}(Z,Y)}_s (\mathcal{E}_2;\{Q\}\uplus\mathcal{P}_0;\Phi_1;D_1\cup\{Z,n\overset{?}{\vdash}z\ ;Y\overset{?}{\vdash}y\};Eq_1\cup\{z\overset{?}{=}p\ ;y\overset{?}{=}c\}).$$

2. First of all, we have $\mathrm{dom}(\theta_2) = vars^2(D_2)$. Second, we know that p' and c' are both channel names such that $p',c'\notin\mathcal{E}_2$ ($\mathcal{E}_1 = \mathcal{E}_2$). Thus, it implies that $Z\theta_2, Y\theta_2\in\mathcal{T}(\mathcal{N}\setminus\mathcal{E}_2,\mathcal{AX})$. Furthermore, we trivially have that $param(Z\theta_2) = param(Y\theta_2) = \emptyset\subseteq\{ax_1,\ldots,ax_n\}$.

Now, it remains to show that σ_2 satisfies the constraints in D_2 and the (in)equalities in Eq_2. Since $\sigma_2 = \{z \mapsto p' ; y \mapsto c'\}$ with z, y fresh variables, it implies that $\Phi_2\sigma_2 = \Phi_1\sigma_1$, $p\sigma_2 = p\sigma_1$ and $c\sigma_2 = c\sigma_1$ which allows us to conclude since

— $(Y\theta_2)(\Phi_2\sigma_2) = c'(\Phi_2\sigma_2) = c' = y\sigma_2$;

— $(Z\theta_2)(\Phi_2\sigma_2) = p'(\Phi_2\sigma_2) = p' = z\sigma_2$;

— $y\sigma_2 =_{\mathsf{E}} c\sigma_2$ and $z\sigma_2 =_{\mathsf{E}} p\sigma_2$ since $c' = c\sigma_1 = c\sigma_2$ and $p' = p\sigma_1 = p\sigma_2$.

3. We have that
$$\begin{array}{rcl}(\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2) & = & (\mathcal{E}_1; \{Q\sigma_1\} \uplus \mathcal{P}_0\sigma_1; \Phi_1\sigma_1) \\ & = & (\mathcal{E}; \{Q'\} \uplus \mathcal{P}'; \Phi) \\ & = & (\mathcal{E}; \mathcal{P}; \Phi)\end{array}$$

4. We have that $\alpha_s\theta_2 = out(Z, Y)\theta_2 = out(p', c') = \alpha$.

Case OPEN-CH$_i$**:** In such a case we have that $\mathcal{E} = \mathcal{E}_1$, $\Phi = \Phi_1\sigma_1$ and there exist p', c, Q', $\mathcal{P}'$ and a fresh channel name ch_n such that $\mathcal{P}_1\sigma_1 = \{\mathsf{out}(p', c).Q'\} \uplus \mathcal{P}'$, $\mathcal{P} = (\{Q'\} \uplus \mathcal{P}')\{c \mapsto ch_m\}$, $p' \notin \mathcal{E}_1$, $c \in \mathcal{E}_1$, and $\alpha = \nu ch_m.out(p, ch_m)$. Hence, we deduce that there exist p, Q and $\mathcal{P}_0$ such that $\mathcal{P}_1 = \{\mathsf{out}(p, c).Q\} \uplus \mathcal{P}_0$ with $p\sigma_1 = p'$, $\mathcal{P}_0\sigma_1 = \mathcal{P}'$ and $Q\sigma_1 = Q'$.

Let Z be a second order variable and z be a fresh first order variable. Let $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_2 = (\{Q\} \uplus \mathcal{P}_0)\{c \mapsto ch_m\}$, $\Phi_2 = \Phi_1$, $D_2 = D_1 \cup \{Z, n \overset{?}{\vdash} z\}$, $Eq_2 = Eq_1 \cup \{z \overset{?}{=} p\}$, $\alpha_s = \nu ch_m.\mathsf{out}(Z, ch_m)$, $\theta_2 = \theta_1 \cup \{Z \mapsto p'\}$, $\sigma_2 = \sigma_1 \cup \{z \mapsto p'\}$ with $n = |\Phi_1\sigma_1|$. We have that:

1. $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1; D_1; Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_2; \mathcal{P}_2; \Phi_2; D_2; Eq_2)$. Indeed, we have that
$$(\mathcal{E}_1; \{\mathsf{out}(p, c).Q\} \uplus \mathcal{P}_0; \Phi_1; D_1; Eq_1) \xrightarrow{\nu ch_m.\mathsf{out}(Z, ch_m)}_s (\mathcal{E}_2; (\{Q\} \uplus \mathcal{P}_0)\{c \mapsto ch_m\}; \Phi_1; D_1 \cup \{Z, n \overset{?}{\vdash} z\}; Eq_1 \cup \{z \overset{?}{=} p\}).$$

2. First of all, we have $\mathrm{dom}(\theta_2) = vars^2(D_2)$. Second, we know that p' is a channel name such that $p' \notin \mathcal{E}_2$ ($\mathcal{E}_1 = \mathcal{E}_2$). Thus, it implies that $Z\theta_2 \in \mathcal{T}(\mathcal{N} \setminus \mathcal{E}_2, \mathcal{AX})$. Furthermore, we trivially have that $param(Z\theta_2) = \emptyset \subseteq \{ax_1, \ldots, ax_n\}$.
Now, it remains to show that σ_2 satisfies the constraints in D_2 and the (in)equalities in Eq_2. Since $\sigma_2 = \{z \mapsto p'\}$ with z fresh variables, it implies that $\Phi_2\sigma_2 = \Phi_1\sigma_1$ and $p\sigma_2 = p\sigma_1$ which allows us to conclude since $(Z\theta_2)(\Phi_2\sigma_2) = p'(\Phi_2\sigma_2) = p' = z\sigma_2$ and $p' = p\sigma_1 = p\sigma_2$ implies $z\sigma_2 =_{\mathsf{E}} p\sigma_2$.

3. We have that
$$\begin{array}{rcl}(\mathcal{E}_2; \mathcal{P}_2\sigma_2; \Phi_2\sigma_2) & = & (\mathcal{E}_1; (\{Q\} \uplus \mathcal{P}_0)\{c \mapsto ch_m\}\sigma_2; \Phi_1\sigma_1) \\ & = & (\mathcal{E}; (\{Q\}\sigma_1 \uplus \mathcal{P}_0\sigma_1)\{c \mapsto ch_m\}; \Phi_1') \\ & = & (\mathcal{E}; (\{Q'\} \uplus \mathcal{P}')\{c \mapsto ch_m\}; \Phi) \\ & = & (\mathcal{E}; \mathcal{P}; \Phi)\end{array}$$

4. We have that $\alpha_s\theta_2 = \nu ch_m.out(Z, ch_m)\theta_2 = \nu ch_m.\mathsf{out}(p', ch_m) = \alpha$.

Case PAR$_i$**:** In such a case we have that $\mathcal{E} = \mathcal{E}_1$ and there exist P', Q', and $\mathcal{P}'$ such that $\mathcal{P}_1\sigma_1 = \{P' \mid Q'\} \uplus \mathcal{P}'$, $\mathcal{P} = \{P'; Q'\} \uplus \mathcal{P}'$, $\Phi = \Phi_1\sigma_1$, and $\alpha = \tau$. Hence we deduce that that there exist P, Q, and $\mathcal{P}_0$ such that $\mathcal{P}_1 = \{P \mid Q\} \uplus \mathcal{P}_0$, and thus we have that $Q\sigma_1 = Q'$, $P\sigma_1 = P'$, and $\mathcal{P}_0\sigma_1 = \mathcal{P}'$. Let $\mathcal{E}_2 = \mathcal{E}_1$, $\mathcal{P}_2 = \{P; Q\} \uplus \mathcal{P}_0$, $\Phi_2 = \Phi_1$, $D_2 = D_1$, $Eq_2 = Eq_1$, $\alpha_s = \tau$, $\theta_2 = \theta_1$ and $\sigma_2 = \sigma_1$. We have that:

1. $(\mathcal{E}_1; \mathcal{P}_1; \Phi_1; D_1; Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_2; \mathcal{P}_2; \Phi_2; D_2; Eq_2)$. Indeed, we have that
$$(\mathcal{E}_1; \{P \mid Q\} \uplus \mathcal{P}_0; \Phi_1; D_1; Eq_1) \xrightarrow{\alpha_s}_s (\mathcal{E}_1; \{P; Q\} \uplus \mathcal{P}_0; \Phi_1; D_1; Eq_1).$$

2. We have that $(\mathcal{E}_2; \Phi_2; D_2; Eq_2) = (\mathcal{E}_1; \Phi_1; D_1; Eq_1)$, $\theta_2 = \theta_1$ and $\sigma_2 = \sigma_1$ and we trivially have that $(\sigma_2, \theta_2) \in \mathsf{Sol}((\mathcal{E}_2; \Phi_2; D_2; Eq_2))$.

3. We have that

$$\begin{array}{lcl}(\mathcal{E}_2;\mathcal{P}_2\sigma_2;\Phi_2\sigma_2) & = & (\mathcal{E}_1;(\{P\sigma_2;Q\sigma_2\}\uplus\mathcal{P}_0\sigma_2);\Phi_2\sigma_2)\\ & = & (\mathcal{E};(\{P\sigma_1;Q\sigma_1\}\uplus\mathcal{P}_0\sigma_1);\Phi_1\sigma_1)\\ (\mathcal{E};(\{P';Q'\}\uplus\mathcal{P}');\Phi) & & \\ & = & (\mathcal{E};\mathcal{P};\Phi).\end{array}$$

4. We have that $\alpha_s\theta_2 = \alpha_s = \alpha$. □

Appendix B

Composition of trace equivalence

B.1 Preliminaries

Lemma 5.4. *If t_1, t_2 are terms (that do not use* $\mathrm{dom}(\rho)$*) in normal form then for all $i \in \{a, b\}$, $t_1 = t_2$ is equivalent to $\delta_i^\rho(t_1) = \delta_i^\rho(t_2)$.*

Proof. The right implication of the lemma is trivial thus, we focus on the left implication: for all $i \in \{a, b\}$,$\delta_i^\rho(t_1) = \delta_i^\rho(t_2)$ implies $t_1 = t_2$. We prove this result by induction on $\max(|t_1|, |t_2|)$.

Base case $\max(|t_1|, |t_2|) = 1$*:* In such a case, we have that $t_1, t_2 \in \mathcal{X} \cup \mathcal{N}$. By definition of δ_a^ρ, we know that $\delta_a^\rho(t_1) = t_1$ and $\delta_a^\rho(t_2) = t_2$. Thus, we can conclude that $\delta_a^\rho(t_1) = \delta_a^\rho(t_2)$ implies $t_1 = t_2$. For the case $i = b$, we do a case analysis on whether $\delta_b^\rho(t_1) = \delta_b^\rho(t_2) = k$ for some $k \in \mathrm{dom}(\rho)$ or not.

Case $\delta_b^\rho(t_1) = k \in \mathrm{dom}(\rho)$*:* By hypothesis, we know that t_2 and t_1 do not use $\mathrm{dom}(\rho)$. Therefore, by definition of δ_b^ρ, we can deduce that, $t_2{\downarrow} = k\rho$ and $t_1{\downarrow} = k\rho$. Since t_1 and t_2 are in normal form, we can conclude that $t_1 = t_2 = k\rho$.

Case $\delta_b^\rho(t_1) \neq k$ *for every* $k \in \mathrm{dom}(\rho)$*:* By definition of δ_b^ρ, we have that $\delta_b^\rho(t_1) = t_1$ and $\delta_b^\rho(t_2) = t_2$, and thus $t_1 = t_2$.

Inductive step $\max(|t_1|, |t_2|) > 1$*:* Assume w.l.o.g. that $|t_1| > 1$. Thus, there exists a function symbol f and terms $u_1, \dots u_n$ such that $t_1 = \mathsf{f}(u_1, \dots u_n)$. Since t_1 is in normal form, we can deduce that $t_1{\downarrow} \neq k\rho$, for every $k \in \mathrm{dom}(\rho)$. We do a case analysis on t_1 :

Case $\mathsf{f} \in \mathcal{F}_d \cup \mathcal{F}_{\mathsf{tag}_d}$ *and* $d \in \{a, b\}$*:* In such a case, we have that $\delta_i^\rho(t_1) = \mathsf{f}(\delta_d^\rho(u_1), \dots, \delta_d^\rho(u_n))$. But $\delta_i^\rho(t_2) = \delta_i^\rho(t_1)$ and by definition of δ_i^ρ, we know that it implies that there exists $v_1, \dots, v_n$ such that $t_2 = \mathsf{f}(v_1, \dots, v_n)$ and $\delta_i^\rho(t_2) = \mathsf{f}(\delta_d^\rho(v_1), \dots, \delta_d^\rho(v_n))$. Thus we have that $\delta_d^\rho(v_j) = \delta_d^\rho(u_j)$ for all $j \in \{1, \dots, n\}$. Furthermore, since t_1 and t_2 are in normal form, we also know that u_j and v_j are in normal form, for every j. But, $\max(|t_1|, |t_2|) > \max(|u_j|, |v_j|)$, for any j, thus by our inductive hypothesis, we can deduce that $u_j = v_j$, for all j and so $t_1 = \mathsf{f}(u_1, \dots, u_n) = \mathsf{f}(v_1, \dots, v_n) = t_2$.

Case $t_1 = \mathsf{f}(\mathsf{tag}_d(w_1), w_2)$*,* $d \in \{a, b\}$ *and* $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$*:* In this case, we know that $\delta_i^\rho(t_1) = \mathsf{f}(\mathsf{tag}_d(\delta_d^\rho(w_1)), \delta_d^\rho(w_2))$. But we know that $\delta_i^\rho(t_2) = \delta_i^\rho(t_1) = \mathsf{f}(\mathsf{tag}_d(\delta_d^\rho(w_1)), \delta_d^\rho(w_2))$. Thus thanks to t_2 being in normal form and by definition of δ_i^ρ, it implies that there exists v_1 and v_2 such that $t_2 = \mathsf{f}(\mathsf{tag}_d(v_1), v_2)$ and so $\delta_i^\rho(t_2) = \mathsf{f}(\mathsf{tag}_d(\delta_d^\rho(v_1)), \delta_d^\rho(v_2))$. Thus, we have that $\delta_d^\rho(v_1) = \delta_d^\rho(u_1)$ and $\delta_d^\rho(v_2) = \delta_d^\rho(u_2)$. Moreover, t_1 and t_2 being in normal form and not using $\mathrm{dom}(\rho)$, so are u_j and v_j for $j \in \{1, 2\}$, so we can apply our inductive hypothesis and conclude that $v_1 = u_1$ and $v_2 = u_2$ and so $t_1 = t_2$.

Case $t_1 = \mathsf{h}(\mathsf{tag}_d(w_1))$ *and* $d \in \{a, b\}$*:* This case is analogous to the previous one and can be handled in a similar way.

Else case: Otherwise, we have that $\mathsf{f} \in \mathcal{F}_0$ but the root symbol of u_1 is not tag_a or tag_b. By definition of δ_i^ρ, we can deduce that $\delta_i^\rho(t_1) = \mathsf{f}(\delta_i^\rho(u_1), \dots, \delta_i^\rho(u_n))$. Since $\delta_i^\rho(t_1) = \delta_i^\rho(t_2)$, we can deduce that the top symbol of t_2 is also f and so there exists $v_1, \dots, v_n$ such that $t_2 = \mathsf{f}(v_1, \dots, v_n)$. In the previous cases, we showed that if $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}, \mathsf{h}\}$ and the top symbol of v_1 is tag_a or

tag_b, then $\delta_i^\rho(t_1) = \delta_i^\rho(t_2)$ implies that the top symbol of u_1 is also tag_a or tag_b. Thus, thanks to our hypothesis, we can deduce that either $\mathsf{f} \notin \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}, \mathsf{h}\}$ or the top symbol of v_1 is different from tag_a and tag_b. Hence by definition of δ_i^ρ, we can deduce that $\delta_i^\rho(t_2) = \mathsf{f}(\delta_i^\rho(v_1), \ldots, \delta_i^\rho(v_n))$ and so $\delta_i^\rho(v_j) = \delta_i^\rho(u_j)$ for all $j \in \{1, \ldots, n\}$. Moreover, t_1 and t_2 being in normal form and not using names in $\mathrm{dom}(\rho)$, implies that so are u_j and v_j for all $j \in \{1, \ldots, n\}$. We can thus invoke our inductive hypothesis and conclude that $u_j = v_j$ for all $j \in \{1, \ldots, n\}$ and so $t_1 = t_2$. □

Lemma 5.5. *Let t_1, t_2 two terms (that do not use $\mathrm{dom}(\rho)$) in normal form. If $\delta_a^\rho(t_1) = \delta_b^\rho(t_2)$ then $t_1 = t_2$.*

Proof. We prove the result by induction on $|\delta_a^\rho(t_1)|$.

Base case $|\delta_a^\rho(t_1)| = 1$: In such a case, we have that $\delta_a^\rho(t_1) \in \mathcal{N} \cup \mathcal{X}$. Assume first that $\delta_a^\rho(t_1) \in \mathcal{X}$ and so $\delta_b^\rho(t_2) \in \mathcal{X}$: By definition of δ_a^ρ and δ_b^ρ, we can deduce that $\delta_a^\rho(t_1) = t_1$ and $\delta_b^\rho(t_2) = t_2$. Thus we conclude that $t_1 = t_2$. Assume now that $\delta_a^\rho(t_1) \in \mathcal{N}$. We need to distinguish two cases :

Case $\delta_a^\rho(t_1) \in \mathrm{dom}(\rho)$: By definition of δ_a^ρ, $\delta_a^\rho(t_1) \in \mathcal{N}$ implies that $\delta_a^\rho(t_1) = t_1$. But we assumed that t_1, t_2 do not use $\mathrm{dom}(\rho)$. Thus this case is impossible.

Case $\delta_a^\rho(t_1) \notin \mathrm{dom}(\rho)$: Once again by definition of δ_a^ρ, we have $\delta_a^\rho(t_1) = t_1$. Furthermore, since t_2 do not use $\mathrm{dom}(\rho)$ and $\delta_b^\rho(t_2) \notin \mathrm{dom}(\rho)$, we also have that $\delta_b^\rho(t_2) = t_2$ by definition of δ_b^ρ. Thus we conclude that $t_1 = t_2$.

Inductive step $|\delta_a^\rho(t_1)| > 1$: In that case, we have that $\delta_a^\rho(t_1) = \mathsf{f}(u_1, \ldots, u_n) = \delta_b^\rho(t_2)$. Assume that $\mathsf{f} \in \mathcal{F}_d \cup \mathcal{F}_{\mathsf{tag}_d}$ with $d \in \{a, b\}$. By definition of δ_a^ρ and δ_b^ρ, we can deduce that $\mathsf{root}(t_1) = \mathsf{f} = \mathsf{root}(t_2)$. Furthermore, if we assume that $t_1 = \mathsf{f}(v_1, \ldots, v_n)$ and $t_2 = \mathsf{f}(w_1, \ldots, w_n)$, we would have $\delta_d^\rho(v_j) = \delta_d^\rho(w_j)$ for all $j \in \{1, \ldots, n\}$. By Lemma 5.4, we deduce that $v_j = w_j$ for all $j \in \{1, \ldots, n\}$. Hence, we conclude that $t_1 = t_2$. Assume now that $\mathsf{f} \in \mathcal{F}_0$. According to the definition of δ_a^ρ and δ_b^ρ, there exists $v_1, \ldots, v_n$ and $w_1, \ldots, w_n$ such that $t_1 = \mathsf{f}(v_1, \ldots, v_n)$, $t_2 = \mathsf{f}(w_1, \ldots, w_n)$ and $\delta_k^\rho(v_j) = \delta_\ell^\rho(w_j)$, for some $k, \ell \in \{a, b\}$. Moreover, t_1 and t_2 being in normal form and not using names in $\mathrm{dom}(\rho)$ imply that so are v_j and w_j for all $j \in \{1, \ldots, n\}$. Now, either $k = \ell$ and so by Lemma 5.4, we have that $v_j = w_j$, else $k \neq \ell$ but then by our inductive hypothesis, we also have $v_j = w_j$. Hence we conclude that $t_1 = t_2$. □

Lemma 5.6. *Let u be a term in normal form that do no use $\mathrm{dom}(\rho)$. We have that for all $i \in \{a, b\}$, $\delta_i^\rho(u)$ is in normal form and $\mathsf{root}(\delta_i^\rho(u)) = \mathsf{root}(u)$.*

Proof. We prove this result by induction on $|u|$.
Base case $|u| = 1$: In such a case, $u \in \mathcal{X} \cup \mathcal{N}$. If $u \in \mathcal{X}$, then for all $i \in \{a, b\}$, $\delta_i^\rho(u) = u$. Since u is in normal form then the result holds. If $u \in \mathcal{N}$, by definition, we also have that $\delta_i^\rho(u) \in \mathcal{N}$ and so $\delta_i^\rho(u)$ is in normal form with the same root as u, namely $\bot$.

Inductive $|u| > 1$: In this case, we have that $u = C[u_1, \ldots, u_n]$ with $u_1, \ldots, u_n$ factors of u. Assume first that C is built upon $\mathcal{F}_j \cup \mathcal{F}_{\mathsf{tag}_j}$, for some $j \in \{a, b\}$. Since u is in normal form, then for all position p of C, we have that $u_{|p}\!\downarrow \notin \mathrm{img}(\rho)$ and so for all $k \in \{a, b\}$, $\delta_k^\rho(u) = C[\delta_j^\rho(u_1), \ldots, \delta_j^\rho(u_n)]$. By inductive hypothesis on $u_1, \ldots, u_n$. We have that $\delta_j^\rho(u_1), \ldots, \delta_j^\rho(u_n)$ are in normal form and $\mathsf{root}(\delta_j^\rho(u_1)) = \mathsf{root}(u_1)$, …, $\mathsf{root}(\delta_j^\rho(u_n)) = \mathsf{root}(u_n)$, thus $\delta_j^\rho(u_1), \ldots, \delta_j^\rho(u_n)$ are factors of $\delta_i^\rho(u)$. Thus, since u is in normal form, by Lemmas 5.4 and 5.3 , we have that $\delta_i^\rho(u)\!\downarrow = C[\delta_j^\rho(u_1), \ldots, \delta_j^\rho(u_n)] = \delta_i^\rho(u)$. Furthermore, we also have that $\mathsf{root}(\delta_i^\rho(u)) = \mathsf{root}(u)$. Assume now that C is built upon $\mathcal{F}_0$. Thus, assume that $u = \mathsf{f}(v_1, \ldots, v_m)$. By definition of δ_a^ρ and δ_b^ρ, there exists $j \in \{a, b\}$ such that $\delta_i^\rho(u) = \mathsf{f}(\delta_j^\rho(v_1), \ldots, \delta_j^\rho(v_m))$. We do a case analysis on f:

Case $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{pk}, \mathsf{sign}, \mathsf{vk}, \mathsf{h}, \langle\,\rangle\}$: In this case, by the rewriting system $\mathcal{O}_0$, we have that $\delta_i^\rho(u)\!\downarrow = \mathsf{f}(\delta_j^\rho(v_1)\!\downarrow, \ldots, \delta_j^\rho(v_m)\!\downarrow)$. Since by inductive hypothesis, $\delta_j^\rho(v_k)$ is in normal form, for all $k \in \{1, \ldots, m\}$, we can deduce that $\delta_i^\rho(u)$ is also in normal form and $\mathsf{root}(\delta_i^\rho(u)) = \mathsf{f} = \mathsf{root}(u)$.

Case $\mathsf{f} = \mathsf{sdec}$: Then by definition of δ_a^ρ and δ_b^ρ, we have that $\delta_i^\rho(u) = \mathsf{sdec}(\delta_j^\rho(v_1), \delta_j^\rho(v_2))$, with $j \in \{a, b\}$. By inductive hypothesis, we have that $\delta_j^\rho(v_1)$ and $\delta_j^\rho(v_2)$ are both in normal form and have the same root as v_1 and v_2 respectively.

Assume first that sdec can not be reduced, *i.e.* $\delta_i^\rho(u)\!\downarrow = \mathsf{sdec}(\delta_j^\rho(v_1)\!\downarrow, \delta_j^\rho(v_2)\!\downarrow)$ and so $\delta_i^\rho(u)\!\downarrow = \mathsf{sdec}(\delta_j^\rho(v_1), \delta_j^\rho(v_2))$. Thus the result holds. Otherwise, if sdec can be reduced, it implies that there

exists w_1, w_2 such that $\delta_j^\rho(v_1) = \mathsf{senc}(w_1, w_2)$ and $\delta_j^\rho(v_2) = w_2$. But by definition of δ_j^ρ, there must exist $k \in \{a, b\}$, and w_1', w_2' such that $\delta_j^\rho(v_1) = \mathsf{senc}(\delta_k^\rho(w_1'), \delta_k^\rho(w_2'))$, $v_1 = \mathsf{senc}(w_1', w_2')$, $w_1 = \delta_k^\rho(w_1')$ and $w_2 = \delta_k^\rho(w_2')$. Thus, we have that $\delta_j^\rho(v_2) = \delta_k^\rho(w_2')$. Thanks to Lemmas 5.4 and 5.5, we can conclude that $v_2 = w_2'$ and so $u = \mathsf{sdec}(\mathsf{senc}(w_1', w_2'), w_2')$. But in such a case, we would have that u is not in normal form which contradicts our hypothesis.

Case $\mathsf{f} = \mathsf{check}$*:* Then by definition of δ_a^ρ and δ_b^ρ, we have that $\delta_i^\rho(u) = \mathsf{check}(\delta_j^\rho(v_1), \delta_j^\rho(v_2))$, with $j \in \{a, b\}$. By inductive hypothesis, we have that $\delta_j^\rho(v_1)$ and $\delta_j^\rho(v_2)$ are both in normal form and have the same root as v_1 and v_2 respectively.

Assume first that check can not be reduced: this case is analogous to the sdec one and can be handled similarly. Otherwise, if check can be reduced, it implies that there exist w_1, w_2 such that $\delta_j^\rho(v_1) = \mathsf{sign}(w_1, w_2)$ and $\delta_j^\rho(v_2) = \mathsf{vk}(w_2)$. But by definition of δ_j^ρ, there must exist $k \in \{a, b\}$, and w_1', w_2' such that $\delta_j^\rho(v_1) = \mathsf{sign}(\delta_k^\rho(w_1'), \delta_k^\rho(w_2'))$, $v_1 = \mathsf{sign}(w_1', w_2')$, $w_1 = \delta_k^\rho(w_1')$ and $w_2 = \delta_k^\rho(w_2')$. Thus, we have that $\delta_j^\rho(v_2) = \mathsf{vk}(\delta_k^\rho(w_2')) = \delta_k^\rho(\mathsf{vk}(w_2'))$. Thanks to Lemmas 5.4 and 5.5, we can conclude that $v_2 = \mathsf{vk}(w_2')$ and so $u = \mathsf{check}(\mathsf{sign}(w_1', w_2'), \mathsf{vk}(w_2')$. But in such a case, we would have that u is not in normal form which contradicts our hypothesis.

Case $\mathsf{f} = \mathsf{adec}$*:* Then by definition of δ_a^ρ and δ_b^ρ, we have that $\delta_i^\rho(u) = \mathsf{adec}(\delta_j^\rho(v_1), \delta_j^\rho(v_2))$, with $j \in \{a, b\}$. By inductive hypothesis, we have that $\delta_j^\rho(v_1)$ and $\delta_j^\rho(v_2)$ are both in normal form and have the same root as v_1 and v_2 respectively.

Assume first that adec cannot be reduced: this case is analogous to the sdec one and can be handled similarly. Otherwise, if adec can be reduced, it implies that there exist w_1, w_2 such that $\delta_j^\rho(v_1) = \mathsf{aenc}(w_1, \mathsf{pk}(w_2))$ and $\delta_j^\rho(v_2) = w_2$. But by definition of δ_j^ρ, there must exist $k \in \{a, b\}$, and w_1', w_2' such that $\delta_j^\rho(v_1) = \mathsf{aenc}(\delta_k^\rho(w_1'), \mathsf{pk}(\delta_k^\rho(w_2')))$, $v_1 = \mathsf{aenc}(w_1', \mathsf{pk}(w_2'))$, $w_1 = \delta_k^\rho(w_1')$ and $w_2 = \delta_k^\rho(w_2')$. Thus, we have that $\delta_j^\rho(v_2) = \delta_k^\rho(w_2')$. Thanks to Lemmas 5.4 and 5.5, we can conclude that $v_2 = w_2'$ and so $u = \mathsf{adec}(\mathsf{aenc}(w_1', \mathsf{pk}(w_2')), w_2')$. But in such a case, we would have that u is not in normal form which contradicts our hypothesis. □

Lemma B.1. *Let ρ be a renaming of name of base type. Let u be a term, C a context (possibly a hole) built over $\mathcal{F}_0$ and $v_1, \ldots, v_m$ terms such that $u = C[v_1, \ldots, v_m]$, $\{v_1, \ldots, v_m\} = Fct_{\mathcal{F}_0}(u)$ and $\mathrm{dom}(\rho)$ do not occur in u. If for all $i \in \{1, \ldots, m\}$, $v_i{\downarrow} \neq k\rho$ for any $k \in \mathrm{dom}(\rho)$, then we have that $\delta_a^\rho(u) = \delta_b^\rho(u)$.*

Proof. We prove this lemma by induction on the syntactic size of $|C|$.

Base case $|C| = 0$ *:* In this case, C is a hole which means that either $\mathsf{root}(u) \notin \mathcal{F}_0$, or u is of the form $\mathsf{f}(\mathsf{tag}_i(u_1), u_2)$ with $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$ and $i \in \{a, b\}$, or u is of the form $\mathsf{h}(\mathsf{tag}_i(u_1))$ with $i \in \{a, b\}$. Remember that $u{\downarrow} \neq k\rho$ for any $k \in \mathrm{dom}(\rho)$. We do a case analysis on u :

Case $u \in \mathcal{N} \cup \mathcal{X}$*:* In such a case, since we assume that $u{\downarrow} \neq k\rho$ for any $k \in \mathrm{dom}(\rho)$, then by definition of δ_a^ρ and δ_b^ρ, we have that $\delta_a^\rho(u) = u = \delta_b^\rho(u)$.

Case $u = \mathsf{f}(u_1, \ldots, u_n)$ with $\mathsf{f} \notin \mathcal{F}_0$: $\mathsf{f} \notin \mathcal{F}_0$ implies that $\mathsf{f} \in \mathcal{F}_d \cup \mathcal{F}_{\mathsf{tag}_d}$ with $d \in \{a, b\}$. Thus, by definition of δ_a^ρ and δ_b^ρ, we have that $\delta_a^\rho(u) = \mathsf{f}(\delta_d^\rho(u_1), \ldots, \delta_d^\rho(u_n)) = \delta_b^\rho(u)$.

Case $u = \mathsf{f}(\mathsf{tag}_d(u_1), u_2)$ *with* $d \in \{a, b\}$ *and* $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$*:* In that case, we have by definition of δ_a^ρ and δ_b^ρ that $\delta_a^\rho(u) = \mathsf{f}(\mathsf{tag}_d(\delta_d^\rho(u_1)), \delta_d^\rho(u_2)) = \delta_b^\rho(u)$.

Case $u = \mathsf{h}(\mathsf{tag}_d(u_1))$ *with* $d \in \{a, b\}$*:* This case is analogous to the previous one and can be handled in a similar way.

Inductive step $|C| > 1$ *:* In this case, we know that $u = \mathsf{f}(u_1, \ldots, u_n)$ with $\mathsf{f} \in \mathcal{F}_0$. But $|C| > 1$ also implies that for all $i \in \{1, \ldots, n\}$, there exists a sub context C_i (possibly a hole) of C such that $u_i = C_i[v_1^i, \ldots, v_{m_i}^i]$ and $\{v_1^i, \ldots, v_{m_i}^i\} = Fct_{\mathcal{F}_0}(u_i)$. Note that $Fct_{\mathcal{F}_0}(u_i) \subseteq Fct_{\mathcal{F}_0}(u)$. Since C_i is a sub context of C, then thanks to our hypothesis, we have that for all $j \in \{1, \ldots, m_i\}$, $v_j^i{\downarrow} \neq k\rho$ for any $k \in \mathrm{dom}(\rho)$. Thus we can apply our inductive hypothesis and deduce that $\delta_a^\rho(u_i) = \delta_b^\rho(u_i)$. Since $C \neq [\_]$ and $\mathsf{f} \in \mathcal{F}_0$, it must be the case that $\mathsf{root}(u_1) \neq \mathsf{tag}_k$ for any $k \in \{a, b\}$, thus by definition of δ_a^ρ and δ_b^ρ, since $\mathsf{root}(u) \in \mathcal{F}_0$, we have that $\delta_a^\rho(u) = \mathsf{f}(\delta_a^\rho(u_1), \ldots, \delta_a^\rho(u_n))$ and $\delta_b^\rho(u) = \mathsf{f}(\delta_b^\rho(u_1), \ldots, \delta_b^\rho(u_n))$. Thanks to the previously established equalities $\delta_a^\rho(u_i) = \delta_b^\rho(u_i)$ for all $i \in \{1, \ldots, n\}$, we can conclude that $\delta_a^\rho(u) = \delta_b^\rho(u)$. □

Lemma B.2. *Let $i \in \{a, b\}$. Let $u \in \mathcal{T}(\mathcal{F}_i \cup \mathcal{F}_0, \mathcal{N} \cup \mathcal{X})$. Let α be a ground substitution such that $fvars(u) \subseteq \mathrm{dom}(\alpha)$ and for all $x \in \mathrm{dom}(\alpha)$, $x\alpha$ is in normal form. Let ρ a renaming of names of base type such that $\mathrm{dom}(\rho)$ do not occur in u or α. We have that:*

— $\delta_i^\rho([u]_i\alpha) = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha)$; *and*

— *If* $\alpha \vDash \mathsf{test}_i([u]_i)$ *then* $\delta_i^\rho([u]_i\alpha)\downarrow = \delta_i^\rho([u]_i\alpha\downarrow)$.

Proof. We prove the two results separately. First of all, we show by induction on $|u|$ that $\delta_i^\rho([u]_i\alpha) = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha)$:

Base case $|u| = 1$*:* In this case, $u \in \mathcal{N} \cup \mathcal{X}$. If $u \in \mathcal{N}$ then we have that $[u]_i = u$ and so $[u]_i\alpha = u$ and $\delta_i^\rho(u) \in \mathcal{N}$. Thus, we have that $\delta_i^\rho([u]_i\alpha) = \delta_i^\rho(u) = \delta_i^\rho(u)\delta_i^\rho(\alpha) = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha)$. Else if $u \in \mathcal{X}$, then we also have that $[u]_i = u$ and $\delta_i^\rho(u) = u$. Thus, $\delta_i^\rho(u)\delta_i^\rho(\alpha) = u\delta_i^\rho(\alpha)$. Since $u \in \mathcal{X}$ and $fvars(u) \subseteq \mathrm{dom}(\alpha)$, we have that $u\delta_i^\rho(\alpha) = \delta_i^\rho(u\alpha)$, thus $\delta_i^\rho([u]_i\alpha) = \delta_i^\rho(u\alpha) = u\delta_i^\rho(\alpha) = \delta_i^\rho(u)\delta_i^\rho(\alpha) = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha)$.

Inductive step $|u| > 1$*:* In this case, $u = \mathsf{f}(u_1, \ldots, u_n)$. We do a case analysis on f.

Case $\mathsf{f} \in \mathcal{F}_i$*:* In such a case, we have that $[u]_i = \mathsf{f}([u_1]_i, \ldots, [u_n]_i)$. But by definition of δ_i^ρ, we have that $\delta_i^\rho([u]_i\alpha) = \mathsf{f}(\delta_i^\rho([u_1]_i\alpha), \ldots, \delta_i^\rho([u_n]_i\alpha))$ and $\delta_i^\rho([u]_i) = \mathsf{f}(\delta_i^\rho([u_1]_i), \ldots, \delta_i^\rho([u_n]_i))$. By our inductive hypothesis, we can deduce that for all $k \in \{1, \ldots, n\}$, we have that $\delta_i^\rho([u_k]_i\alpha) = \delta_i^\rho([u_k]_i)\delta_i^\rho(\alpha)$. Thus, $\delta_i^\rho([u]_i\alpha) = \mathsf{f}(\delta_i^\rho([u_1]_i), \ldots, \delta_i^\rho([u_n]_i))\delta_i^\rho(\alpha) = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha)$.

Case $\mathsf{f} \in \{\mathsf{aenc}, \mathsf{sign}, \mathsf{senc}\}$*:* In this case $n = 2$, and by definition of $[u]_i$, we have that $[u]_i = \mathsf{f}(\mathsf{tag}_i([u_1]_i), [u_2]_i)$. Thus, we have that $\delta_i^\rho([u]_i) = \mathsf{f}(\mathsf{tag}_i(\delta_i^\rho([u_1]_i)), \delta_i^\rho([u_2]_i))$ and $\delta_i^\rho([u]_i\alpha) = \mathsf{f}(\mathsf{tag}_i(\delta_i^\rho([u_1]_i\alpha)), \delta_i^\rho([u_2]_i\alpha))$. But by our inductive hypothesis, we can deduce that $\delta_i^\rho([u_k]_i\alpha) = \delta_i^\rho([u_k]_i)\delta_i^\rho(\alpha)$, for all $k \in \{1, 2\}$. Thus, $\delta_i^\rho([u]_i\alpha) = \mathsf{f}(\mathsf{tag}_i(\delta_i^\rho([u_1]_i)\delta_i^\rho(\alpha)), \delta_i^\rho([u_2]_i)\delta_i^\rho(\alpha))$ and so we conclude that $\delta_i^\rho([u]_i\alpha) = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha)$.

Case $\mathsf{f} = \mathsf{h}$*:* This case is analogous to the previous one and can be handled in a similar way.

Case $\mathsf{f} \in \{\mathsf{sdec}, \mathsf{adec}, \mathsf{check}\}$*:* In this case $n = 2$, and by definition of $[u]_i$, we have that $[u]_i = \mathsf{untag}_i(\mathsf{f}([u_1]_i, [u_2]_i))$. Thus, we have that $\delta_i^\rho([u]_i) = \mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i), \delta_i^\rho([u_2]_i)))$ and $\delta_i^\rho([u]_i\alpha) = \mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i\alpha), \delta_i^\rho([u_2]_i\alpha)))$. Once again, with our inductive hypothesis, we can deduce that $\delta_i^\rho([u_k]_i\alpha) = \delta_i^\rho([u_k]_i)\delta_i^\rho(\alpha)$, for $k \in \{1, 2\}$, and so we can conclude that $\delta_i^\rho([u]_i\alpha) = \mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i), \delta_i^\rho([u_2]_i)))\delta_i^\rho(\alpha) = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha)$.

Else case: In this case, by definition of $[u]_i$, we have that $[u]_i = \mathsf{f}([u_1]_i, \ldots, [u_n]_i)$ and $\delta_i^\rho([u]_i) = \mathsf{f}(\delta_i^\rho([u_1]_i), \ldots, \delta_i^\rho([u_n]_i))$. Thus, this case is similar to the case $\mathsf{f} \in \mathcal{F}_i$. Hence the result holds.

We now prove the second property, *i.e.* if $\alpha \vDash \mathsf{test}_i([u]_i)$, then $\delta_i^\rho([u]_i\alpha)\downarrow = \delta_i^\rho([u]_i\alpha\downarrow)$. Once again, we prove the results by induction on $|u|$:

Base case $|u| = 1$*:* In this case, $u \in \mathcal{N} \cup \mathcal{X}$. In both cases, we have that $[u]_i = u$ and $\mathsf{test}_i(u) = \mathsf{true}$. If $u \in \mathcal{N}$, we know that $\delta_i^\rho(u) \in \mathcal{N}$ and so $\delta_i^\rho(u)\downarrow = \delta_i^\rho(u)$. But we also have that $u\alpha\downarrow = u\alpha = u$. Thus, we conclude that $\delta_i^\rho(u\alpha)\downarrow = \delta_i^\rho(u)\downarrow = \delta_i^\rho(u) = \delta_i^\rho(u\alpha\downarrow)$. Else, if $u \in \mathcal{X}$, by hypothesis on α, we deduce that $u\alpha\downarrow = u\alpha$. Furthermore, by Lemma 5.6, we also know that $\delta_i^\rho(u\alpha\downarrow)\downarrow = \delta_i^\rho(u\alpha\downarrow)$. Thus, we conclude that $\delta_i^\rho(u\alpha\downarrow) = \delta_i^\rho(u\alpha\downarrow)\downarrow = \delta_i^\rho(u\alpha)\downarrow$.

Inductive step $|u| > 1$*:* In this case, we have $u = \mathsf{f}(u_1, \ldots, u_n)$. We do a case analysis on f.

Case $\mathsf{f} \in \mathcal{F}_i$*:* In such a case, we have that $[u]_i = \mathsf{f}([u_1]_i, \ldots, [u_n]_i)$. Hence, we deduce that $\delta_i^\rho([u]_i\alpha) = \mathsf{f}(\delta_i^\rho([u_1]_i\alpha), \ldots, \delta_i^\rho([u_n]_i\alpha))$ and so $\delta_i^\rho([u]_i\alpha)\downarrow = \mathsf{f}(\delta_i^\rho([u_1]_i\alpha)\downarrow, \ldots, \delta_i^\rho([u_n]_i\alpha)\downarrow)\downarrow$. But $\mathsf{test}_i([u]_i) = \bigwedge_{j=1}^n \mathsf{test}_i([u_j]_i)$ which means that $\alpha \vDash \mathsf{test}_i([u_j]_i)$, for all j. Thus, by our inductive hypothesis on $u_1, \ldots, u_n$, we deduce that $\delta_i^\rho([u]_i\alpha)\downarrow = \mathsf{f}(\delta_i^\rho([u_1]_i\alpha\downarrow), \ldots, \delta_i^\rho([u_n]_i\alpha\downarrow))\downarrow = \delta_i^\rho(\mathsf{f}([u_1]_i\alpha\downarrow, \ldots, [u_n]_i\alpha\downarrow))\downarrow$.

Let's denote $t = \mathsf{f}([u_1]_i\alpha\downarrow, \ldots, [u_n]_i\alpha\downarrow)$. We can assume that there exists a context C built on $\mathcal{F}_i$ such that $t = C[t_1, \ldots, t_m]$ with $Fct(t) = \{t_1, \ldots, t_m\}$ and $t_1, \ldots, t_m$ are in normal form. Thus, by Lemma 5.2, there exists a context D (possibly a hole) such that $t\downarrow = D[t_{j_1}, \ldots, t_{j_k}]$ with $j_1, \ldots, j_k \in \{0, \ldots, m\}$ and $t_0 = n_{min}$. But since $t_1, \ldots, t_m$ are all in normal form and

thanks to Lemma 5.6, we know that for all $k \in \{0,\dots,m\}$, $\delta_i^\rho(t_k)$ is also in normal form and its root is not in $\mathcal{F}_i$. Moreover, thanks to Lemma 5.4, we know that $t_p = t_q$ is equivalent to $\delta_i^\rho(t_p) = \delta_i^\rho(t_q)$, for all $p,q \in \{0,\dots,m\}$. Hence, we can apply Lemma 5.3 such that $C[\delta_i^\rho(t_1),\dots,\delta_i^\rho(t_n)]\!\downarrow = D[\delta_i^\rho(t_{j_1}),\dots,\delta_i^\rho(t_{j_k})]$. But since C and D are both built upon $\mathcal{F}_i$, we have that $C[\delta_i^\rho(t_1),\dots,\delta_i^\rho(t_n)]\!\downarrow = \delta_i^\rho(C[t_1,\dots,t_n])\!\downarrow$ and $D[\delta_i^\rho(t_{j_1}),\dots,\delta_i^\rho(t_{j_k})] = \delta_i^\rho(D[t_{j_1},\dots,t_{j_k}])$. Hence, we can deduce that $\delta_i^\rho(t)\!\downarrow = \delta_i^\rho(t\!\downarrow)$. But we already know that $t\!\downarrow = [u]_i\alpha\!\downarrow$ and $\delta_i^\rho(t)\!\downarrow = \delta_i^\rho([u]_i\alpha)\!\downarrow$. Thus, we can conclude that $\delta_i^\rho([u]_i\alpha)\!\downarrow = \delta_i^\rho([u]_i\alpha\!\downarrow)$.

Case $\mathsf{f} \in \{\mathsf{senc},\mathsf{aenc},\mathsf{sign}\}$*:* In such a case, we have that $[u]_i = \mathsf{f}(\mathsf{tag}_i([u_1]_i),[u_2]_i)$ and $\mathsf{test}_i([u]_i) = \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i)$. By definition of the rewriting system $\mathcal{O}_0$, we have that $[u]_i\alpha\!\downarrow = \mathsf{f}(\mathsf{tag}_i([u_1]_i\alpha\!\downarrow),[u_2]_i\alpha\!\downarrow)$. Moreover, $\delta_i^\rho([u]_i) = \mathsf{f}(\mathsf{tag}_i(\delta_i^\rho([u_1]_i)),\delta_i^\rho([u_2]_i))$ and so we have $\delta_i^\rho([u]_i\alpha)\!\downarrow = \mathsf{f}(\mathsf{tag}_i(\delta_i^\rho([u_1]_i\alpha)\!\downarrow),\delta_i^\rho([u_2]_i\alpha)\!\downarrow)$. By our inductive hypothesis on u_1 and u_2, we have that $\delta_i^\rho([u_k]_i\alpha)\!\downarrow = \delta_i^\rho([u_k]_i\alpha\!\downarrow)$, for $k \in \{1,2\}$. Hence, we can deduce that $\delta_i^\rho([u]_i\alpha)\!\downarrow = \mathsf{f}(\mathsf{tag}_i(\delta_i^\rho([u_1]_i\alpha\!\downarrow)),\delta_i^\rho([u_2]_i\alpha\!\downarrow)) = \delta_i^\rho(\mathsf{f}(\mathsf{tag}_i([u_1]_i\alpha\!\downarrow),[u_2]_i\alpha\!\downarrow))$. Thus, $\delta_i^\rho([u]_i\alpha)\!\downarrow = \delta_i^\rho([u]_i\alpha\!\downarrow)$.

Case $\mathsf{f} = \mathsf{h}$*:* This case is analogous to the previous one and can be handled in a similar way.

Case $\mathsf{f} \in \{\mathsf{pk},\mathsf{vk},\langle\,\rangle\}$*:* In this case, we have that $[u]_i = \mathsf{f}([u_1]_i,\dots,[u_n]_i)$ with $n \in \{1,2\}$, and $\mathsf{test}_i([u]_i) = \wedge_{j=1}^n \mathsf{test}_i([u_j]_i)$. By definition of $\mathcal{O}_0$, we have that $[u]_i\alpha\!\downarrow = \mathsf{f}([u_1]_i\alpha\!\downarrow,[u_2]_i\alpha\!\downarrow)$. Thus, this case is similar to the senc case and can be handled similarly.

Case $\mathsf{f} \in \{\mathsf{sdec},\mathsf{adec},\mathsf{check}\}$*:* In such a case, we have that $[u]_i = \mathsf{untag}_i(\mathsf{f}([u_1]_i,[u_2]_i))$ and $\mathsf{test}_i([u]_i) = \big(\mathsf{tag}_i(\mathsf{untag}_i(\mathsf{f}([u_1]_i,[u_2]_i))) = \mathsf{f}([u_1]_i,[u_2]_i)\big) \wedge \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i)$. But by hypothesis, we know that $\alpha \vDash \mathsf{test}_i([u]_i)$, thus $\mathsf{tag}_i(\mathsf{untag}_i(\mathsf{f}([u_1]_i,[u_2]_i)))\alpha\!\downarrow = \mathsf{f}([u_1]_i,[u_2]_i)\alpha\!\downarrow$. Hence, we deduce that the root function symbol f can be reduced and the root symbol of the plain text is tag_i, *i.e.* there exists v_1, v_2 such that

- if $\mathsf{f} = \mathsf{sdec}$ then $[u_1]_i\alpha\!\downarrow = \mathsf{senc}(\mathsf{tag}_i(v_1),v_2)$, $[u_2]_i\alpha\!\downarrow = v_2$ and $[u]_i\alpha\!\downarrow = v_1$. It implies that $\delta_i^\rho([u_1]_i\alpha\!\downarrow) = \mathsf{senc}(\mathsf{tag}_i(\delta_i^\rho(v_1)),\delta_i^\rho(v_2))$ and so we can deduce that $\delta_i^\rho([u]_i\alpha\!\downarrow) = \delta_i^\rho(v_1) = \mathsf{untag}_i(\mathsf{sdec}(\delta_i^\rho([u_1]_i\alpha\!\downarrow),\delta_i^\rho([u_2]_i\alpha\!\downarrow)))\!\downarrow$
- $\mathsf{f} = \mathsf{adec}$: $[u_1]_i\alpha\!\downarrow = \mathsf{aenc}(\mathsf{tag}_i(v_1),\mathsf{pk}(v_2))$, $[u_2]_i\alpha\!\downarrow = v_2$ and $[u]_i\alpha\!\downarrow = v_1$. It implies that $\delta_i^\rho([u_1]_i\alpha\!\downarrow) = \mathsf{aenc}(\mathsf{tag}_i(\delta_i^\rho(v_1)),\mathsf{pk}(\delta_i^\rho(v_2)))$ and so we can deduce that $\delta_i^\rho([u]_i\alpha\!\downarrow) = \delta_i^\rho(v_1) = \mathsf{untag}_i(\mathsf{adec}(\delta_i^\rho([u_1]_i\alpha\!\downarrow),\delta_i^\rho([u_2]_i\alpha\!\downarrow)))\!\downarrow$
- $\mathsf{f} = \mathsf{check}$: $[u_1]_i\alpha\!\downarrow = \mathsf{sign}(\mathsf{tag}_i(v_1),v_2)$, $[u_2]_i\alpha\!\downarrow = \mathsf{vk}(v_2)$ and $[u]_i\alpha\!\downarrow = v_1$. It implies that $\delta_i^\rho([u_1]_i\alpha\!\downarrow) = \mathsf{sign}(\mathsf{tag}_i(\delta_i^\rho(v_1)),\delta_i^\rho(v_2))$ and so we can deduce that $\delta_i^\rho([u]_i\alpha\!\downarrow) = \delta_i^\rho(v_1) = \mathsf{untag}_i(\mathsf{check}(\delta_i^\rho([u_1]_i\alpha\!\downarrow),\delta_i^\rho([u_2]_i\alpha\!\downarrow)))\!\downarrow$

In all cases, the following equality holds: $\mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i\alpha\!\downarrow),\delta_i^\rho([u_2]_i\alpha\!\downarrow)))\!\downarrow = \delta_i^\rho([u]_i\alpha\!\downarrow)$. But by inductive hypothesis, we know that $\delta_i^\rho([u_k]_i\alpha\!\downarrow) = \delta_i^\rho([u_k]_i\alpha)\!\downarrow$, for $k \in \{1,2\}$. Thus, since we also have that $\delta_i^\rho([u]_i\alpha)\!\downarrow = \mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i\alpha)\!\downarrow,\delta_i^\rho([u_2]_i\alpha)\!\downarrow))\!\downarrow$, we can conclude that $\delta_i^\rho([u]_i\alpha)\!\downarrow = \mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i\alpha\!\downarrow),\delta_i^\rho([u_2]_i\alpha\!\downarrow)))\!\downarrow = \delta_i^\rho([u]_i\alpha\!\downarrow)$.

Case $\mathsf{f} = \mathsf{proj}_j$*,* $j = 1,2$*:* In this case $n = 1$, and $[u]_i = \mathsf{f}([u_1]_i)$. Since $\alpha \vDash \mathsf{test}_i([u]_i)$, we have that there exists v_1, v_2 such that $[u_1]_i\alpha\!\downarrow = \langle v_1, v_2\rangle$ and so $\delta_i^\rho([u]_i\alpha\!\downarrow) = \delta_i^\rho(v_j)$. But by inductive hypothesis, we have that $\delta_i^\rho([u_1]_i\alpha)\!\downarrow = \delta_i^\rho([u_1]_i\alpha\!\downarrow) = \langle\delta_i^\rho(v_1),\delta_i^\rho(v_2)\rangle$. Hence, $\delta_i^\rho([u]_i\alpha)\!\downarrow = \mathsf{f}(\delta_i^\rho([u_1]_i\alpha))\!\downarrow = \mathsf{f}(\delta_i^\rho([u_1]_i\alpha)\!\downarrow)\!\downarrow = \delta_i^\rho(v_j)\!\downarrow$. But we showed that $\delta_i^\rho(v_j) = \delta_i^\rho([u]_i\alpha\!\downarrow)$, thus by Lemma 5.6, $\delta_i^\rho(v_j)$ is in normal form. Hence, we have that $\delta_i^\rho([u]_i\alpha)\!\downarrow = \delta_i^\rho(v_j) = \delta_i^\rho([u]_i\alpha\!\downarrow)$ which allows us to conclude. □

Corollary B.1. *Let* $i \in \{a,b\}$*. Let* $u,v \in \mathcal{T}(\mathcal{F}_i \cup \mathcal{F}_0, \mathcal{N} \cup \mathcal{X})$*. Let* α *be a ground substitution such that* $\mathit{fvars}(u) \subseteq \mathrm{dom}(\alpha)$ *and for all* $x \in \mathrm{dom}(\alpha)$*,* $x\alpha$ *is in normal form. Let* ρ *a renaming of names of base type such that* $\mathrm{dom}(\rho)$ *do not occur in* u, v *or* α*. We have that : If* $\alpha \vDash \mathsf{test}_i([u]_i) \wedge \mathsf{test}_i([v]_i)$ *then* $[u]_i\alpha\!\downarrow = [v]_i\alpha\!\downarrow$ *is equivalent to* $\delta_i^\rho([u]_i)\delta_i^\rho(\alpha)\!\downarrow = \delta_i^\rho([v]_i)\delta_i^\rho(\alpha)\!\downarrow$*.*

Proof. Thanks to Lemma 5.4, $[u]_i\alpha\!\downarrow = [v]_i\alpha\!\downarrow$ is equivalent to $\delta_i^\rho([u]_i\alpha\!\downarrow) = \delta_i^\rho([v]_i\alpha\!\downarrow)$. But thanks to Lemma B.2, we have that $\delta_i^\rho([u]_i\alpha\!\downarrow) = \delta_i^\rho([u]_i\alpha)\!\downarrow = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha)\!\downarrow$ and $\delta_i^\rho([v]_i\alpha\!\downarrow) = \delta_i^\rho([v]_i\alpha)\!\downarrow = \delta_i^\rho([v]_i)\delta_i^\rho(\alpha)\!\downarrow$. Thus, the result holds. □

Lemma B.3. *Let $i \in \{a, b\}$. Let $u, v \in \mathcal{T}(\mathcal{F}_i \cup \mathcal{F}_0, \mathcal{N} \cup \mathcal{X})$. Let α be a ground substitution such that $fvars(u) \subseteq \mathrm{dom}(\alpha)$ and for all $x \in \mathrm{dom}(\alpha)$, $x\alpha$ is in normal form. Let ρ a renaming of names of base type such that $\mathrm{dom}(\rho)$ do not occur in u, v or α. We have that :*

$$\alpha \vDash \mathsf{test}_i([u]_i) \text{ is equivalent to } \delta_i^\rho(\alpha) \vDash \mathsf{test}_i(\delta_i^\rho([u]_i))$$

Proof. We prove this result by induction on $|u|$:

Base case $|u| = 1$: In this case, we have that $u \in \mathcal{N} \cup \mathcal{X}$, and thus $[u]_i, \delta_i^\rho([u]_i) \in \mathcal{N} \cup \mathcal{X}$. But then by definition, $\mathsf{test}_i([u]_i) = \mathsf{true}$ and $\mathsf{test}_i(\delta_i^\rho([u]_i)) = \mathsf{true}$. Thus the result trivially holds.

Inductive step $|u| > 1$: Then, we have that $u = \mathsf{f}(u_1, \ldots, u_n)$. We do a case analysis on f:

Case $\mathsf{f} \in \mathcal{F}_i \cup \{\mathsf{pk}, \mathsf{vk}, \langle\ \rangle\}$: In this case, we have that $[u]_i = \mathsf{f}([u_1]_i, \ldots, [u_n]_i)$ and $\delta_i^\rho([u]_i) = \mathsf{f}(\delta_i^\rho([u_1]_i), \ldots, \delta_i^\rho([u_n]_i))$. Thus, we deduce that $\mathsf{test}_i([u]_i) = \bigwedge_{j=1}^n \mathsf{test}_i([u_j]_i)$ and $\mathsf{test}_i(\delta_i^\rho([u]_i)) = \bigwedge_{j=1}^n \mathsf{test}_i(\delta_i^\rho([u_j]_i))$. By inductive hypothesis on $u_1, \ldots, u_n$, the result holds.

Case $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$: In this case, we have that $[u]_i = \mathsf{f}(\mathsf{tag}_i([u_1]_i), [u_2]_i)$ and $\delta_i^\rho([u]_i) = \mathsf{f}(\mathsf{tag}_i(\delta_i^\rho([u_1]_i)), \delta_i^\rho([u_2]_i))$. Thus, we deduce that $\mathsf{test}_i(\delta_i^\rho([u]_i)) = \mathsf{test}_i(\delta_i^\rho([u_1]_i)) \wedge \mathsf{test}_i(\delta_i^\rho([u_2]_i))$ and $\mathsf{test}_i([u]_i) = \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i)$. By inductive hypothesis on u_1, u_2, the result holds.

Case $\mathsf{f} = \mathsf{h}$: This case is analogous to the previous one and can be handled in a similar way.

Case $\mathsf{f} \in \{\mathsf{sdec}, \mathsf{adec}, \mathsf{check}\}$: In this case, we have that $[u]_i = \mathsf{untag}_i(\mathsf{f}([u_1]_i, [u_2]_i))$ and $\delta_i^\rho([u]_i) = \mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i), \delta_i^\rho([u_2]_i)))$. Thus, we deduce that:

— $\mathsf{test}_i([u]_i) = \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i) \wedge \mathsf{tag}_i(\mathsf{untag}_i(\mathsf{f}([u_1]_i, [u_2]_i))) = \mathsf{f}([u_1]_i, [u_2]_i)$

— $\mathsf{test}_i(\delta_i^\rho([u]_i)) = \mathsf{test}_i(\delta_i^\rho([u_1]_i)) \wedge \mathsf{test}_i(\delta_i^\rho([u_2]_i)) \wedge \mathsf{tag}_i(\mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i), \delta_i^\rho([u_2]_i)))) = \mathsf{f}(\delta_i^\rho([u_1]_i), \delta_i^\rho([u_2]_i))$

Whether we assume that $\alpha \vDash \mathsf{test}_i([u]_i)$ or $\delta_i^\rho(\alpha) \vDash \mathsf{test}_i(\delta_i^\rho([u]_i))$, we have by inductive hypothesis that $\alpha \vDash \mathsf{test}_i([u_k]_i)$ for $k \in \{1, 2\}$. Thus by Lemma B.2, it implies that $\delta_i^\rho([u_k]_i\alpha\downarrow) = \delta_i^\rho([u_k]_i)\delta_i^\rho(\alpha)\downarrow$, for $k \in \{1, 2\}$. We do a case analysis on f :

— $\mathsf{f} = \mathsf{sdec}$: $\alpha \vDash \mathsf{tag}_i(\mathsf{untag}_i(\mathsf{f}([u_1]_i, [u_2]_i))) = \mathsf{f}([u_1]_i, [u_2]_i)$ is equivalent to there exists v_1, v_2 such that $[u_1]_i\alpha\downarrow = \mathsf{senc}(\mathsf{tag}_i(v_1), v_2)$ and $[u_2]_i\alpha\downarrow = v_2$. But by Lemma 5.4, it is equivalent to $\delta_i^\rho([u_1]_i\alpha\downarrow) = \mathsf{senc}(\mathsf{tag}_i(\delta_i^\rho(v_1)), \delta_i^\rho(v_2))$ and $\delta_i^\rho([u_2]_i\alpha\downarrow) = \delta_i^\rho(v_2)$. Thus, it is equivalent to $\delta_i^\rho([u_1]_i)\delta_i^\rho(\alpha)\downarrow = \mathsf{senc}(\mathsf{tag}_i(\delta_i^\rho(v_1)), \delta_i^\rho(v_2))$ and $\delta_i^\rho([u_2]_i)\delta_i^\rho(\alpha)\downarrow = \delta_i^\rho(v_2)$. Hence it is equivalent to $\delta_i^\rho(\alpha) \vDash \mathsf{tag}_i(\mathsf{untag}_i(\mathsf{f}(\delta_i^\rho([u_1]_i), \delta_i^\rho([u_2]_i)))) = \mathsf{f}(\delta_i^\rho([u_1]_i), \delta_i^\rho([u_2]_i))$

— $\mathsf{f} = \mathsf{adec}$ and $\mathsf{f} = \mathsf{check}$: Similar to case $\mathsf{f} = \mathsf{sdec}$.

Case $\mathsf{f} \in \{\mathsf{proj}_1, \mathsf{proj}_2\}$: In this case, we have that $[u]_i = \mathsf{f}([u_1]_i)$ and $\delta_i^\rho([u]_i) = \mathsf{f}(\delta_i^\rho([u_1]_i))$. Thus, we deduce that :

— $\mathsf{test}_i([u]_i) = \mathsf{test}_i([u_1]_i) \wedge \langle \mathsf{proj}_1([u_1]_i), \mathsf{proj}_2([u_1]_i)\rangle = [u_1]_i$

— $\mathsf{test}_i(\delta_i^\rho([u]_i)) = \mathsf{test}_i(\delta_i^\rho([u_1]_i)) \wedge \langle \mathsf{proj}_1(\delta_i^\rho([u_1]_i)), \mathsf{proj}_2(\delta_i^\rho([u_1]_i))\rangle = \delta_i^\rho([u_1]_i)$

Whether we assume that $\alpha \vDash \mathsf{test}_i([u]_i)$ or $\delta_i^\rho(\alpha) \vDash \mathsf{test}_i(\delta_i^\rho([u]_i))$, we have by inductive hypothesis that $\alpha \vDash \mathsf{test}_i([u_1]_i)$. Thus by Lemma B.2, it implies that $\delta_i^\rho([u_1]_i\alpha\downarrow) = \delta_i^\rho([u_1]_i)\delta_i^\rho(\alpha)\downarrow$.

But $\alpha \vDash \langle \mathsf{proj}_1([u_1]_i), \mathsf{proj}_2([u_1]_i)\rangle = [u_1]_i$ is equivalent to there exists v_1, v_2 such that $[u_1]_i\alpha\downarrow = \langle v_1, v_2\rangle$, which is equivalent to $\delta_i^\rho([u_1]_i\alpha\downarrow) = \langle \delta_i^\rho(v_1), \delta_i^\rho(v_2)\rangle$ thanks to Lemma 5.4. We showed that it is equivalent to $\delta_i^\rho([u_1]_i)\delta_i^\rho(\alpha)\downarrow = \langle \delta_i^\rho(v_1), \delta_i^\rho(v_2)\rangle$, which allows us to conclude that $\alpha \vDash \langle \mathsf{proj}_1([u_1]_i), \mathsf{proj}_2([u_1]_i)\rangle = [u_1]_i$ is equivalent $\delta_i^\rho(\alpha) \vDash \langle \mathsf{proj}_1(\delta_i^\rho([u_1]_i)), \mathsf{proj}_2(\delta_i^\rho([u_1]_i))\rangle = \delta_i^\rho([u_1]_i)$. □

We are now focus on the message that will be send the network during the execution of the processes. For a term u that does not contain any tag, we defined in Section 5.1, a way to construct a term that is properly tagged (*i.e.* $[u]_i$). Hence, for a term properly tagged, we would never have $\mathsf{senc}(n, k)$ where n and k are both nonces, for example. Instead, we would have $\mathsf{senc}(\mathsf{tag}_i(n), k)$. However, even if we can force the processes to properly tag their term, we do not have any control

on what the intruder can build. Typically, if the intruder is able to deduce n and k, he is allowed to send to a process the term $\mathsf{senc}(n,k)$. Similarly, while we can restrict our processes to only apply vk and pk on a nonce, we can not restrict the intruder from using those cryptographic primitive with terms different from a nonce.

Thus, we define the notion of *flawed tagged subterm* of a term.

Definition B.1. *Let u be a ground term in normal form. We define the flawed tagged subterm of u, denoted $\mathsf{Flawed}(u)$, is the set of subterms $v \in st(u)$ such that either:*

- $v = \mathsf{f}(u_1, u_2)$ *with* $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$ *and* $\mathsf{root}(u_1) \notin \{\mathsf{tag}_a, \mathsf{tag}_b\}$ *for some u_1, u_2; or*
- $v = \mathsf{h}(u)$ *and* $\mathsf{root}(u_1) \notin \{\mathsf{tag}_a, \mathsf{tag}_b\}$ *for some u; or*
- $v = \mathsf{f}(u)$ *with* $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ *and* $u \notin \mathcal{N}$ *for some u; or*
- $v = \mathsf{f}(u_1, \ldots, u_n)$ *with* $\mathsf{f} \in \{\mathsf{sdec}, \mathsf{adec}, \mathsf{check}, \mathsf{proj}_1, \mathsf{proj}_2\}$ *for some $u_1, \ldots, u_n$.*

Note that all subterm with a destructor is considered as flawed. Indeed, we saw in Section 5.1 that a process P_A tagged with the tag a may output the instantiation of $[u]_a$ only if it satisfies the the instantiation of the conditional $\mathsf{test}_{[u]_a}(a)$. But as already mentioned in Subsection 5.1.1.2, $\mathsf{test}_{[u]_a}(a)$ verifies that the computation done by P_A on $[u]_a$ succeed properly, i.e. the destructors from $\mathsf{sdec}, \mathsf{adec}, \mathsf{check}, \mathsf{proj}_1, \mathsf{proj}_2, \mathsf{untag}_a, \mathsf{untag}_b$ are reduced before sending the instantiation of $[u]_a$ over the network. But once again, we cannot restrict the intruder from misusing those cryptographic primitives. Thus, he is allowed to send $\mathsf{sdec}(n,k)$ to a process. Hence, to sum up, $\mathsf{Flawed}(u)$ represents all the subterm of u that the intruder is sure to have computed (he may have computed more that $\mathsf{Flawed}(u)$ in u).

Example B.1. *Consider the term $u = \langle \mathsf{senc}(\mathsf{tag}_a(\mathsf{proj}_1(\mathsf{sdec}(n,k))), k), \mathsf{aenc}(a, \mathsf{pk}(\langle k, k' \rangle)))\rangle$. We have that u is in normal form and*

$$\mathsf{Flawed}(u) = \{\mathsf{proj}_1(\mathsf{sdec}(n,k)); \mathsf{sdec}(n,k); \mathsf{aenc}(a, \mathsf{pk}(\langle k, k' \rangle)); \mathsf{pk}(\langle k, k' \rangle)\}$$

We formalise the property that any flawed subterm has to be computed by the intruder with the following result:

Lemma B.4. *Let $i \in \{a, b\}$. Let $u \in \mathcal{T}(\mathcal{F}_i \cup \mathcal{F}_0, \mathcal{N} \cup \mathcal{X})$ such that for all $v \in st(u)$, $\mathsf{root}(v) = \mathsf{f}$ with $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ implies that there exists $v' \in \mathcal{N}$ such that $v = \mathsf{f}(v')$. Let α be a ground substitution such that $fvars(u) \subseteq \mathrm{dom}(\alpha)$ and for all $x \in \mathrm{dom}(\alpha)$, $x\alpha$ is in normal form. We have that if $\alpha \vDash \mathsf{test}_i([u]_i)$ then for all $t \in \mathsf{Flawed}([u]_i\alpha\downarrow)$, there exists $x \in fvars([u]_i)$ such that $t \in \mathsf{Flawed}(x\alpha)$.*

Proof. We prove the result by induction on $|u|$.

Base case $|u| = 1$: In this case, we have that $u \in \mathcal{X} \cup \mathcal{N}$ and so $[u]_i = u$. If $u \in \mathcal{N}$, then $u\alpha \in \mathcal{N}$ and $[u]_i\alpha\downarrow \in \mathcal{N}$, which means that $\mathsf{Flawed}([u]_i\alpha\downarrow) = \emptyset$. Thus the results holds. Else $u \in \mathcal{X}$ and so $[u]_i = u \in \mathrm{dom}(\alpha)$ which means that the result trivially holds.

Inductive step $|u| > 1$: Then, $u = \mathsf{f}(u_1, \ldots, u_n)$. We do a case analysis on f.

Case $\mathsf{f} \in \mathcal{F}_i$: In this case, $[u]_i = \mathsf{f}([u_1]_i, \ldots, [u_n]_i)$ and $[u]_i\alpha\downarrow = \mathsf{f}([u_1]_i\alpha\downarrow, \ldots, [u_n]_i\alpha\downarrow)\downarrow$. By definition, we know that for all $t \in \mathsf{Flawed}([u]_i\alpha\downarrow)$, $\mathsf{root}(t) \notin \mathcal{F}_a \cup \mathcal{F}_b$. Thus, thanks to Lemma 5.2, we can deduce that for all $t \in \mathsf{Flawed}([u]_i\alpha\downarrow)$, there exists $k \in \{1, \ldots, n\}$ such that $t \in \mathsf{Flawed}([u_k]_i\alpha\downarrow)$. By hypothesis, $\alpha \vDash \mathsf{test}_i([u]_i)$ and so $\alpha \vDash \mathsf{test}_i([u_k]_i)$. Thus, by inductive hypothesis, we know that there exists $x \in fvars([u_k]_i)$ such that $t \in st(x\alpha)$. But $x \in fvars([u_k]_i)$ implies $x \in fvars([u]_i)$, thus the result holds.

Case $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$: In this case, $[u]_i = \mathsf{f}(\mathsf{tag}_i([u_1]_i), [u_2]_i)$. Furthermore, $[u]_i\alpha\downarrow = \mathsf{f}(\mathsf{tag}_i([u_1]_i\alpha\downarrow), [u_2]_i\alpha\downarrow)$. At last, $\alpha \vDash \mathsf{test}_i([u]_i)$ implies that $\alpha \vDash \mathsf{test}_i([u_k]_i)$, for all $k = 1, 2$. But, by definition, $\mathsf{Flawed}([u]_i\alpha\downarrow) = \mathsf{Flawed}([u_1]_i\alpha\downarrow) \cup \mathsf{Flawed}([u]_i\alpha\downarrow)$ and so by our inductive hypothesis on u_1 and u_2, the result holds.

Case $\mathsf{f} = \mathsf{h}$: This case is analogous to the previous one and can be handled in a similar way.

Case $\mathsf{f} = \langle\ \rangle$: In this case, $[u]_i = \mathsf{f}([u_1]_i, [u_2]_i)$. Furthermore, $[u]_i\alpha\downarrow = \mathsf{f}([u_1]_i\alpha\downarrow, [u_2]_i\alpha\downarrow)$. At last, $\alpha \vDash \mathsf{test}_i([u]_i)$ implies that $\alpha \vDash \mathsf{test}_i([u_k]_i)$, for all $k = 1, 2$. But, by definition,

$\mathsf{Flawed}([u]_i\alpha\downarrow) = \mathsf{Flawed}([u_1]_i\alpha\downarrow) \cup \mathsf{Flawed}([u]_i\alpha\downarrow)$ and so by our inductive hypothesis on u_1 and u_2, the result holds.

Case $\mathsf{f} = \{\mathsf{vk}, \mathsf{pk}\}$*:* In this case, we know by hypothesis that $u = \mathsf{f}(v)$ with $v \in \mathcal{N}$. Thus $[u]_i = u$ and $\mathsf{Flawed}(u) = \emptyset$. Thus the result trivially holds.

Case $\mathsf{f} \in \{\mathsf{sdec}, \mathsf{adec}, \mathsf{check}\}$*:* In this case, we have that $[u]_i = \mathsf{untag}_i(\mathsf{f}([u_1]_i, [u_2]_i))$ and $\mathsf{test}_i([u]_i) = \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i) \wedge \mathsf{tag}_i(\mathsf{untag}_i([u]_i)) = [u]_i$. But by hypothesis, we know that $\alpha \vDash \mathsf{test}_i([u]_i)$ and more specifically $\mathsf{tag}_i(\mathsf{untag}_i([u]_i))\alpha\downarrow = [u]_i\alpha\downarrow$. It implies that there exists v_1, v_2 such that $[u_1]_i\alpha\downarrow = \mathsf{g}(\mathsf{tag}_i(v_1), v_2)$ and $[u]_i\alpha\downarrow = v_1$, with $\mathsf{g} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$. Thus, for all $t \in \mathsf{Flawed}([u]_i\alpha\downarrow)$, $t \in \mathsf{Flawed}([u_1]_i\alpha\downarrow)$. Since $\alpha \vDash \mathsf{test}_i([u_1]_i)$, the result holds by inductive hypothesis.

Case $\mathsf{f} = \mathsf{proj}_j$*,* $j \in \{1, 2\}$*:* In this case, we have that $[u]_i = \mathsf{f}([u_1]_i)$ and $\mathsf{test}_i([u]_i) = \mathsf{test}_i([u_1]_i) \wedge \langle \mathsf{proj}_1([u_1]_i), \mathsf{proj}_2([u_1]_i)\rangle = [u_1]_i$. Hence, $\alpha \vDash \mathsf{test}_i([u]_i)$ implies that there exist v_1, v_2 such that $[u_1]_i\alpha\downarrow = \langle v_1, v_2\rangle$ and $[u]_i\alpha\downarrow = v_j$. Thus, for all $t \in \mathsf{Flawed}([u]_i\alpha\downarrow)$, $t \in \mathsf{Flawed}([u_1]_i\alpha\downarrow)$. Since $\alpha \vDash \mathsf{test}_i([u_1]_i)$ by hypothesis, our inductive hypothesis allows us to conclude. □

Corollary B.2. *Let* $i \in \{a, b\}$*. Let* $u \in \mathcal{T}(\mathcal{F}_i \cup \mathcal{F}_0, \mathcal{N} \cup \mathcal{X})$ *such that for all* $v \in st(u)$*,* $\mathsf{root}(v) = \mathsf{f}$ *with* $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ *implies that there exists* $v' \in \mathcal{N}$ *such that* $v = \mathsf{f}(v')$*. Let* α *be a ground substitution such that* $fvars(u) \subseteq \mathrm{dom}(\alpha)$ *and for all* $x \in \mathrm{dom}(\alpha)$*,* $x\alpha$ *is in normal form. Let* ρ *a renaming of names of base type such that* $\mathrm{dom}(\rho)$ *do not occur in* u *or* α*. We have that if* $\delta_i^\rho(\alpha) \vDash \mathsf{test}_i(\delta_i^\rho([u]_i))$*, then for all* $t \in \mathsf{Flawed}(\delta_i^\rho([u]_i)\delta_i^\rho(\alpha)\downarrow)$*, there exists* $x \in fvars(\delta_i^\rho([u]_i)$ *such that* $t \in \mathsf{Flawed}(x\delta_i^\rho(\alpha))$*.*

Proof (sketch). This result is a corollary of Lemma B.4. Actually, this can be proved in a similar way since the transformation δ_i^ρ does not change the structure of the term. □

We use the previous Lemmas to prove some properties on well-tagged frames. Let $(\mathcal{E}; \mathcal{P}; \Phi)$ be a closed intermediate process such that $\Phi = \{ax_1 \rhd u_1, \ldots, ax_n \rhd u_n\}$. We define a lexicographic measure on terms M, denoted $\mathcal{M}(M)$, where $fvars(M) \subseteq \mathrm{dom}(\Phi)$ and $fnames(M) \cap \mathcal{E} = \emptyset$ such that : $\mathcal{M}(M) = (\max\{i \mid ax_i \in fvars(M)\}, |M|)$.

Lemma B.5. *Let* $(\mathcal{E}; \mathcal{P}; \Phi)$ *be a closed intermediate process such that* $\nu\mathcal{E}.\Phi$ *is well-tagged. Let's denote* $\Phi = \{ax_1 \rhd u_1;\ \ldots;\ ax_n \rhd u_n\}$*. We have that for all* $i \in \{1, \ldots, n\}$*, for all* $t \in \mathsf{Flawed}(u_i\downarrow)$*, there exists* M *such that* $fvars(M) \subseteq \{ax_1, \ldots, ax_{i-1}\}$*,* $fnames(M) \cap \mathcal{E} = \emptyset$ *and* $t \in \mathsf{Flawed}(M\Phi\downarrow)$*.*

Proof. We prove this result for any ground term u well-tagged up to ax_i, $i \in \{1, \ldots, n\}$. By definition, u being well-tagged up to ax_i, $i \in \{1, \ldots, n\}$ implies that there exists a term v, a substitution α and $c \in \{a, b\}$ such that:

- for all $\mathsf{vk}(t), \mathsf{pk}(t') \in st(v)$, $t, t' \in \mathcal{N}$
- $u = [v]_c\alpha$; and
- $\alpha \vDash \mathsf{test}_c([v]_c)$; and
- for all $x \in \mathrm{dom}(\alpha)$, either v is not a variable and $x\alpha$ is well-tagged up to ax_i; or there exists M such that $fvars(M) \subseteq \{ax_1, \ldots, ax_{i-1}\}$, $fnames(M) \cap \mathcal{E} = \emptyset$ and $M\Phi = x\alpha$.

The proof is done by induction $|u|$.

Base case $|u| = 1$*:* In this case, $u \in \mathcal{N}$ which implies $u\downarrow = u$, and thus by definition $\mathsf{Flawed}(u\downarrow) = \emptyset$. Hence the result trivially holds.

Inductive step $|u| > 1$*:* Let $t \in \mathsf{Flawed}(u\downarrow)$. Since for all $\mathsf{vk}(t), \mathsf{pk}(t') \in st(v)$, $t, t' \in \mathcal{N}$, and $u = [v]_c\alpha$ and $\alpha \vDash \mathsf{test}_c([v]_c)$, we can apply Lemma B.4 to v and $\alpha\downarrow$. Thus, we have that there exists $x \in fvars([v]_c)$ such that $t \in \mathsf{Flawed}(x\alpha\downarrow)$. But since u is well-tagged up to ax_i, we know that either (a) there exists M such that $fvars(M) \subseteq \{ax_1, \ldots, ax_{i-1}\}$, $fnames(M) \cap \mathcal{E} = \emptyset$ and $M\Phi = x\alpha$. Hence $t \in \mathsf{Flawed}(x\alpha\downarrow) = \mathsf{Flawed}(M\Phi\downarrow)$ and so the result holds in such a case.

Or, (b) v is not a variable and thus neither is $[v]_c$. Moreover, $x\alpha$ is well-tagged up to ax_i. Now because v is not a variable and $x \in fvars([v]_c)$, we have that $|x\alpha| < |[v]_c\alpha| = |u|$, we conclude by applying our inductive hypothesis. □

Lemma B.6. *Let $(\mathcal{E}, \mathcal{P}, \Phi)$ be a closed intermediate process such that $\nu\mathcal{E}.\Phi$ is well-tagged. We have that for all M such that $fnames(M) \cap \mathcal{E} = \emptyset$ and $fvars(M) \subseteq \mathrm{dom}(\Phi)$, for all $\mathsf{f}(u_1, \ldots, u_n) \in \mathsf{Flawed}(M\Phi\downarrow)$, there exists $M_1, \ldots, M_n$ such that $fvars(M_i) \subseteq \mathrm{dom}(\Phi)$, $fnames(M_i) \cap \mathcal{E} = \emptyset$, $M_i\Phi\downarrow = u_i$, and $\mathcal{M}(M_i) < \mathcal{M}(M)$, for all $i \in \{1, \ldots, n\}$.*

Proof. We prove this result by induction on $\mathcal{M}(M)$.
Base case $\mathcal{M}(M) = (0, 0)$*:* A term with $|M| = 0$ is impossible so the result trivially holds.

Inductive step $\mathcal{M}(M) = (i, 1)$*:* In this case, either we have that $M \in \mathcal{N}$ or $M = ax_i$. If $M \in \mathcal{N}$, then we have $M\Phi\downarrow = M \in \mathcal{N}$ and $\mathsf{Flawed}(M\Phi\downarrow) = \emptyset$. Thus the result holds. If $M = ax_i$, then by Lemma B.5, for all $\mathsf{f}(t_1, \ldots, t_m) \in \mathsf{Flawed}(w_i\Phi\downarrow)$, there exists M' such that $fvars(M') \subseteq \{w_1, \ldots, w_{i-1}\}$, $fnames(M) \cap \mathcal{E} = \emptyset$ and $\mathsf{f}(t_1, \ldots, t_m) \in \mathsf{Flawed}(M'\Phi\downarrow)$. But $\mathcal{M}(M') < \mathcal{M}(M)$, thus by our inductive hypothesis, we can deduce that there exists $M_1, \ldots, M_m$ such that $fvars(M_i) \subseteq \mathrm{dom}(\Phi)$, $fnames(M_i) \cap \mathcal{E} = \emptyset$, $M_i\Phi\downarrow = t_i$ and $\mathcal{M}(M_i) < \mathcal{M}(M') < \mathcal{M}(M)$, for all $i \in \{1, \ldots, m\}$.

Inductive step $\mathcal{M}(M) > (i, 1)$*:* We have that $M = \mathsf{f}(M_1, \ldots, M_n)$. Let $t = \mathsf{g}(t_1, \ldots, t_m) \in \mathsf{Flawed}(M\Phi\downarrow)$. We do a case analysis on f.

Case $\mathsf{f} \in \mathcal{F}_\ell \cup \mathcal{F}_{\mathsf{tag}_\ell}$, $\ell \in \{a, b\}$*:* In this case, $M\Phi\downarrow = \mathsf{f}(M_1\Phi\downarrow, \ldots, M_n\Phi\downarrow)\downarrow$. By definition of $t \in \mathsf{Flawed}(M\Phi\downarrow)$, we know that $\mathsf{root}(t) \notin \mathcal{F}_\ell \cup \mathcal{F}_{\mathsf{tag}_\ell}$. Thus, thanks to Lemma 5.2, we can deduce that there exists $k \in \{1, \ldots, n\}$ such that $t \in \mathsf{Flawed}(M_k\Phi\downarrow)$. But $\mathcal{M}(M_k) < \mathcal{M}(M)$, thus, by inductive hypothesis, we know that there exists $M'_1, \ldots, M'_m$ such that $fvars(M'_j) \subseteq \Phi$, $fnames(M'_j) \cap \mathcal{E} = \emptyset$, $M'_j\Phi\downarrow = t_i$ and $\mathcal{M}(M'_j) < \mathcal{M}(M_k) < \mathcal{M}(M)$, for all $j \in \{1, \ldots, m\}$. Hence the result holds.

Case $\mathsf{f} = \langle\ \rangle$*:* In such a case, $M\Phi\downarrow = \mathsf{f}(M_1\Phi\downarrow, M_2\Phi\downarrow)$. Furthermore by definition, we have $\mathsf{Flawed}(M\Phi\downarrow) = \mathsf{Flawed}(M_1\Phi\downarrow) \cup \mathsf{Flawed}(M_2\Phi\downarrow)$. Since $\mathcal{M}(M_1) < \mathcal{M}(M)$, $\mathcal{M}(M_2) < \mathcal{M}(M)$ and $t \in \mathsf{Flawed}(M_1\Phi\downarrow) \cup \mathsf{Flawed}(M_2\Phi\downarrow)$, we conclude by applying our inductive hypothesis on M_1 (or M_2).

Case $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$*:* In this case, $M\Phi\downarrow = \mathsf{f}(M_1\Phi\downarrow)$. If $M_1\Phi\downarrow \in \mathcal{N}$, then we have that $\mathsf{Flawed}(M\Phi\downarrow) = \emptyset$, else $\mathsf{Flawed}(M\Phi\downarrow) = \{M\Phi\downarrow\} \cup \mathsf{Flawed}(M_1\Phi\downarrow)$. If $t = M\Phi\downarrow$, then we have $t_1 = M_1\Phi\downarrow$. Since $\mathcal{M}(M_1) < \mathcal{M}(M)$, then the result holds; else we conclude by applying our inductive hypothesis on M_1.

Case $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$*:* In such a case, $M\Phi\downarrow = \mathsf{f}(M_1\Phi\downarrow, M_2\Phi\downarrow)$. We need to distinguish if $\mathsf{root}(M_1\Phi\downarrow) \in \{\mathsf{tag}_a, \mathsf{tag}_b\}$ or not.

If $\mathsf{root}(M_1\Phi\downarrow) \in \{\mathsf{tag}_a, \mathsf{tag}_b\}$, then there exists $\ell \in \{a, b\}$ and u_1 such that $M_1\Phi\downarrow = \mathsf{tag}_\ell(u_1)$. Thus, $\mathsf{Flawed}(M_1\Phi\downarrow) = \mathsf{Flawed}(u_1)$. But by definition, we have that $\mathsf{Flawed}(M\Phi\downarrow) = \mathsf{Flawed}(u_1) \cup \mathsf{Flawed}(M_2\Phi\downarrow)$. Thus, $t \in \mathsf{Flawed}(M\Phi\downarrow)$ implies that $t \in \mathsf{Flawed}(M_1\Phi\downarrow)$ or $t \in \mathsf{Flawed}(M_2\Phi\downarrow)$. Since $\mathcal{M}(M_1) < \mathcal{M}(M)$ and $\mathcal{M}(M_2) < \mathcal{M}(M)$, we conclude by applying our inductive hypothesis on M_1 or M_2.

Else $\mathsf{root}(M_1\Phi\downarrow) \notin \{\mathsf{tag}_a, \mathsf{tag}_b\}$. In such a case, we have that $\mathsf{Flawed}(M\Phi\downarrow) = \mathsf{Flawed}(M_1\Phi\downarrow) \cup \mathsf{Flawed}(M_2\Phi\downarrow) \cup \{M\Phi\downarrow\}$. If $t = M\Phi\downarrow$, we have that $t_1 = M_1\Phi\downarrow$, $t_2 = M_2\Phi\downarrow$ and $\mathcal{M}(M_1) < \mathcal{M}(M)$, $\mathcal{M}(M_2) < \mathcal{M}(M)$. Thus the result holds. If $t \in \mathsf{Flawed}(M_1\Phi\downarrow) \cup \mathsf{Flawed}(M_2\Phi\downarrow)$, we conclude by applying our inductive hypothesis on M_1 or M_2.

Case $\mathsf{f} = \mathsf{h}$*:* This case is analogous to the previous one and can be handled similarly.

Case $\mathsf{f} \in \{\mathsf{sdec}, \mathsf{adec}, \mathsf{check}\}$*:* For all those functions, we have to distinguish two cases: Either f is reduced in $M\Phi\downarrow$, or not.

If f is not reduced, then we have that $M\Phi\downarrow = \mathsf{f}(M_1\downarrow, M_2\downarrow)$. By definition we have that $\mathsf{Flawed}(M\Phi\downarrow) = \{M\Phi\downarrow\} \cup \mathsf{Flawed}(M_1\Phi\downarrow) \cup \mathsf{Flawed}(M_2\Phi\downarrow)$. Thus if $t = M\Phi\downarrow$, we have that $t_1 = M_1\Phi\downarrow$, $t_2 = M_2\Phi\downarrow$ and $\mathcal{M}(M_1) < \mathcal{M}(M)$, $\mathcal{M}(M_2) < \mathcal{M}(M)$. Therefore the result holds. Else if $t \in \mathsf{Flawed}(M_1\Phi\downarrow)$ or $t \in \mathsf{Flawed}(M_2\Phi\downarrow)$, since $\mathcal{M}(M_1) < \mathcal{M}(M)$, $\mathcal{M}(M_2) < \mathcal{M}(M)$, we can conclude by applying our inductive hypothesis on M_1 or M_2.

If f is reduced, then we have that $M_1\Phi\downarrow = \mathsf{f}'(u_1, u_2)$ with $M\Phi\downarrow = u_1$ and $\mathsf{f}' \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}\}$. If $\mathsf{root}(u_1) = \mathsf{tag}_\ell$, with $\ell \in \{a, b\}$ then we have that there exists u'_1 such that $u_1 = \mathsf{tag}_\ell(u'_1)$,

$\mathsf{Flawed}(M\Phi\downarrow) = \mathsf{Flawed}(u_1')$ and $\mathsf{Flawed}(M_1\Phi\downarrow) = \mathsf{Flawed}(u_1') \cup \mathsf{Flawed}(u_2)$. Thus $\mathsf{Flawed}(M\Phi\downarrow) \subseteq \mathsf{Flawed}(M_1\Phi\downarrow)$. If $\mathsf{root}(u_1) \neq \mathsf{tag}_\ell$, for all $\ell \in \{a, b\}$, then we have that $\mathsf{Flawed}(M_1\Phi\downarrow) = \{M_1\Phi\downarrow\} \cup \mathsf{Flawed}(u_1) \cup \mathsf{Flawed}(u_2)$ and $\mathsf{Flawed}(M\Phi\downarrow) = \mathsf{Flawed}(u_1)$. Thus, we also have that $\mathsf{Flawed}(M\Phi\downarrow) \subseteq \mathsf{Flawed}(M_1\Phi\downarrow)$. Since in both cases, we have that $\mathsf{Flawed}(M\Phi\downarrow) \subseteq \mathsf{Flawed}(M_1\Phi\downarrow)$ and $\mathcal{M}(M_1) < \mathcal{M}(M)$, we can conclude by applying our inductive hypothesis on M_1.

Case $\mathsf{f} = \mathsf{proj}_j$, $j \in \{1, 2\}$*:* One again, we have to distinguish two cases: Either f is reduced in $M\Phi\downarrow$, or not. If f is not reduced, then we have that $M\Phi\downarrow = \mathsf{f}(M_1\Phi\downarrow)$. By definition we have that $\mathsf{Flawed}(M\Phi\downarrow) = \{M\Phi\downarrow\} \cup \mathsf{Flawed}(M_1\Phi\downarrow)$. Thus if $t = M\Phi\downarrow$, we have that $t_1 = M_1\Phi\downarrow$ and $\mathcal{M}(M_1) < \mathcal{M}(M)$. Therefore the result holds. Else if $t \in \mathsf{Flawed}(M_1\Phi\downarrow)$, since $\mathcal{M}(M_1) < \mathcal{M}(M)$, we can conclude by applying our inductive hypothesis on M_1 or M_2.

If f is reduced, then we have that $M_1\Phi\downarrow = \langle u_1, u_2\rangle$ with $M\Phi\downarrow = u_j$. Thus we have that $\mathsf{Flawed}(M_1\Phi\downarrow) = \mathsf{Flawed}(u_1) \cup \mathsf{Flawed}(u_2)$ and $\mathsf{Flawed}(M\Phi\downarrow) = \mathsf{Flawed}(u_j)$. Hence, we have that $\mathsf{Flawed}(M\Phi\downarrow) \subseteq \mathsf{Flawed}(M_1\Phi\downarrow)$. Since $\mathcal{M}(M_1) < \mathcal{M}(M)$, we can conclude by applying our inductive hypothesis on M_1. □

Lemma B.7. *Let u be a ground term in normal form. We have that there exists a context C (possibly a hole) built on $\{\langle\ \rangle\}$, and $u_1, \ldots, u_m$ such that $u = C[u_1, \ldots, u_m]$, and for all $i \in \{1, \ldots, m\}$,*

- *either $u_i \in \mathsf{Flawed}(u)$;*
- *or $u_i \in Fct_{\mathcal{F}_0}(u)$ and $\delta_a^\rho(u_i) = \delta_b^\rho(u_i)$,*
- *or $u_i = \mathsf{f}(n)$ for some $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ and $n \in \mathcal{N}$,*
- *or $u_i \in \mathcal{N}$.*

Proof. Let u a ground term in normal form and let $\{v_1, \ldots, v_n\} = Fct_{\mathcal{F}_0}(u)$. Thus there exists a context D (possibly a hole) built on $\mathcal{F}_0$ such that $u = D[v_1, \ldots, v_n]$. We prove the result by induction on $|D|$.

Base case $|D| = 0$: We show that the result holds with $C = \_$. $|D| = 0$ implies that $Fct_{\mathcal{F}_0}(u) = u$. Assume first that $u\downarrow \notin \mathrm{img}(\rho)$. Hence by Lemma B.1, we have $\delta_a^\rho(u) = \delta_b^\rho(u)$ and so the result holds. Assume now that $u\downarrow \in \mathrm{img}(\rho)$. In such a case, since u is in normal form, we have $u \in \mathcal{N}$ which allow us to conclude.

Inductive step $|D| > 0$: There exist $\mathsf{f} \in \mathcal{F}_0$, and $v_1, \ldots, v_k$ such that $u = \mathsf{f}(u_1, \ldots, u_k)$. We do a case analysis on f.

Case $\mathsf{f} = \langle\ \rangle$*:* In such a case, there exists D_1, D_2 context (possibly holes) built on $\mathcal{F}_0$ such that $D = \langle D_1, D_2\rangle$, $u_i = D_i[v_1^i, \ldots, v_{n_i}^i]$ with $\{v_1^i, \ldots, v_{n_i}^i\} = Fct_{\mathcal{F}_0}(u_i)$ and $|D_i| < |D|$, for all $i \in \{1, 2\}$. By inductive hypothesis on u_1 and u_2, we have that there exists C_1 and C_2 context built on $\{\langle\ \rangle\}$ such that $u_1 = C_1[u_1^1, \ldots, u_{m_1}^1]$, $u_2 = C_2[u_1^2, \ldots, u_{m_2}^2]$ and for all i, j,

- either $u_j^i \in \mathsf{Flawed}(u_i)$, but we have that $\mathsf{Flawed}(u) = \mathsf{Flawed}(u_1) \cup \mathsf{Flawed}(u_2)$ so $u_j^i \in \mathsf{Flawed}(u)$;
- or $u_j^i \in Fct_{\mathcal{F}_0}(u_i)$ and $\delta_a^\rho(u_j^i) = \delta_b^\rho(u_j^i)$, but we have that $Fct_{\mathcal{F}_0}(u) = Fct_{\mathcal{F}_0}(u_1) \uplus Fct_{\mathcal{F}_0}(u_2)$ thus $u_j^i \in Fct_{\mathcal{F}_0}(u)$
- or $u_j^i = \mathsf{f}(n)$ for some $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ and $n \in \mathcal{N}$,
- or $u_j^i \in \mathcal{N}$.

Thus, with the context $C = \langle C_1, C_2\rangle$, and $u = C[u_1^1, \ldots, u_{m_1}^1, u_1^2, \ldots, u_{m_2}^2]$, the result holds.

Case $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ *and* $u = \mathsf{f}(n)$ *for some* $n \in N$*:* With $C = \_$ as context, the result trivially holds.

Otherwise: By definition, we have that $\mathsf{Flawed}(u) = \{u\} \cup \bigcup_{i=1}^k \mathsf{Flawed}(u_i)$. Thus, since $u \in \mathsf{Flawed}(u)$, then with $C = \_$ as context, the result trivially holds. □

B.2 Proof for the first result

In this section, we will focus on the lemmas needed for the proof of Theorem 5.1. Typically, the first section of the Appendix is useful for the proofs of both main results. Thus, this section is independent of section B.3.

Lemma 5.9. *Let $\mathcal{E}$ be a set of names and $\Phi = \{ax_1 \rhd u_1, \ldots, ax_n \rhd u_n\}$ such that $\nu\mathcal{E}.\Phi$ is a derived well-tagged frame in normal form. Let ρ be a renaming such that* $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$ *and* $\mathrm{dom}(\rho) \cap \mathit{fnames}(\Phi) = \emptyset$*. If one of the two following conditions is satisfied:*

(a) for all $k \in \mathrm{img}(\rho)$, $\nu\mathcal{E}.\Phi \not\vdash k$, $\nu\mathcal{E}.\Phi \not\vdash \mathsf{pk}(k)$ and $\nu\mathcal{E}.\Phi \not\vdash \mathsf{vk}(k)$

(b) for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash k$, $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash \mathsf{pk}(k)$ and $\nu\mathcal{E}.\delta^\rho(\Phi) \not\vdash \mathsf{vk}(k)$;

then for all M such that $\mathit{fvars}(M) \subseteq \mathrm{dom}(\Phi)$ and $\mathit{fnames}(M) \cap \mathcal{E} = \emptyset$, for all $i \in \{a, b\}$, $\delta_i^\rho(M\Phi\!\downarrow) = M\delta^\rho(\Phi)\!\downarrow$.

Proof. We prove this result by induction on $\mathcal{M}(M)$:

Base case $\mathcal{M}(M) = (0, 0)$: There exists no term M such that $|M| = 0$, thus the result holds.

Inductive step $\mathcal{M}(M) > (0, 0)$: We first prove there exists $i \in \{a, b\}$ such that $\delta_i^\rho(M\Phi\!\downarrow) = M\delta^\rho(\Phi)\!\downarrow$ and then we show that $\delta_a^\rho(M\Phi\!\downarrow) = \delta_b^\rho(M\Phi\!\downarrow)$

Assume first that $|M| = 1$, i.e. either $M \in \mathcal{N}$ or there exists $j \in \{1, \ldots, n\}$ such that $M = ax_j$.

Let us first suppose $M \in \mathcal{N}$. In that case, $M\beta\!\downarrow = M$ for any susbsitution β. Now, because by hypotheses $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$, and $\mathit{fnames}(M) \cap \mathcal{E} = \emptyset$, we necessarily have $M\!\downarrow \neq k\rho$ for any $k \in \mathrm{dom}(\rho)$. Thus, by definition $\delta_j^\rho(M\Phi\!\downarrow) = \delta_j^\rho(M) = M = M\delta^\rho(\Phi)\!\downarrow$ for any $d \in \{a, b\}$.

Let's now assume that there exists $j \in \{1, \ldots, n\}$ such that $M = ax_j$, and suppose that $\mathsf{col}(ax_j) = i \in \{a, b\}$. According to the definition of $\delta^\rho(\Phi)$, we have that $ax_j\delta^\rho(\Phi) = \delta_i^\rho(ax_j\Phi)$. Since u_j is in normal form, then by Lemma 5.6, we know that $\delta_i^\rho(ax_j\Phi)$ is also in normal form. Thus, we have that $\delta_i^\rho(M\Phi\!\downarrow) = M\delta^\rho(\Phi)\!\downarrow$.

Otherwise, if $|M| > 1$, then there exists f and $M_1, \ldots, M_n$ such that $M = \mathsf{f}(M_1, \ldots, M_n)$. We do a case analysis on f:

Case $\mathsf{f} \in \mathcal{F}_\ell \cup \mathcal{F}_{\mathsf{tag}_\ell}$, $\ell \in \{a, b\}$: In this case, let $t = \mathsf{f}(M_1\Phi\!\downarrow, \ldots, M_n\Phi\!\downarrow)$. Since $\mathsf{f} \in \mathcal{F}_\ell$, then there exists a context C built upon $\mathcal{F}_\ell$ such that $t = C[u_1, \ldots, u_m]$ and $u_1, \ldots, u_m$ are factor of t in normal form. By Lemma 5.2, we know that there exists a context D (possibly a hole) over $\mathcal{F}_0$ such that $t\!\downarrow = D[u_{i_1}, \ldots, u_{i_k}]$ with $i_1, \ldots, i_k \in \{0, \ldots, m\}$ and $u_0 = n_{min}$. But thanks to Lemma 5.3, 5.4 and 5.6, we also that $C[\delta_\ell^\rho(u_1), \ldots, \delta_\ell^\rho(u_m)]\!\downarrow = D[\delta_\ell^\rho(u_{i_1}), \ldots, \delta_\ell^\rho(u_{i_k})]$. But C and D are both built on $\mathcal{F}_\ell \cup \mathcal{F}_{\mathsf{tag}_\ell}$, thus by definition of δ_ℓ^ρ, we have that $\delta_\ell^\rho(t)\!\downarrow = C[\delta_\ell^\rho(u_1), \ldots, \delta_\ell^\rho(u_m)]\!\downarrow$ and $\delta_\ell^\rho(t\!\downarrow) = D[\delta_\ell^\rho(u_{i_1}), \ldots, \delta_\ell^\rho(u_{i_k})]$. Hence, the equality, $\delta_\ell^\rho(t\!\downarrow) = \delta_\ell^\rho(t)\!\downarrow$, holds. But $t\!\downarrow = M\Phi\!\downarrow$ which means that $\delta_\ell^\rho(M\Phi\!\downarrow) = \delta_\ell^\rho(t)\!\downarrow$.

At last $\delta_\ell^\rho(t)\!\downarrow = \delta_\ell^\rho(\mathsf{f}(M_1\Phi\!\downarrow, \ldots, M_n\Phi\!\downarrow))\!\downarrow = \mathsf{f}(\delta_\ell^\rho(M_1\Phi\!\downarrow), \ldots, \delta_\ell^\rho(M_n\Phi\!\downarrow))\!\downarrow$. Since we have that $\mathcal{M}(M_1) < \mathcal{M}(M)$, …, $\mathcal{M}(M_n) < \mathcal{M}(M)$, we can apply our inductive hypothesis on $M_1, \ldots, M_n$ and so $\delta_\ell^\rho(t)\!\downarrow = \mathsf{f}(M_1\delta^\rho(\Phi)\!\downarrow, \ldots, M_n\delta^\rho(\Phi)\!\downarrow)\!\downarrow = \mathsf{f}(M_1, \ldots, M_n)\delta^\rho(\Phi)\!\downarrow$. Thus we can conclude that $\delta_\ell^\rho(M\Phi\!\downarrow) = \delta_\ell^\rho(t)\!\downarrow = M\delta^\rho(\Phi)\!\downarrow$.

Case $\mathsf{f} \in \{\mathsf{senc}, \mathsf{aenc}, \mathsf{sign}, \langle\ \rangle, \mathsf{pk}, \mathsf{vk}, \mathsf{h}\}$: In such a case, $M\Phi\!\downarrow = \mathsf{f}(M_1\Phi\!\downarrow, \ldots, M_n\Phi\!\downarrow)$. By applying our inductive hypothesis on $M_1, \ldots, M_n$, we have $\delta_a^\rho(M_k\Phi\!\downarrow) = \delta_b^\rho(M_k\Phi\!\downarrow)$, for all $k \in \{1, \ldots, n\}$. Thus we have that $\delta_i^\rho(M\Phi\!\downarrow) = \mathsf{f}(\delta_j^\rho(M_1\Phi\!\downarrow), \ldots, \delta_j^\rho(M_n\Phi\!\downarrow))$ with $j \in \{a, b\}$, for all $i \in \{a, b\}$. Thus by applying again our inductive hypothesis on $M_1, \ldots, M_n$, we have that $\delta_i^\rho(M\Phi\!\downarrow) = \mathsf{f}(M_1\delta^\rho(\Phi)\!\downarrow, \ldots, M_n\delta^\rho(\Phi)\!\downarrow) = M\delta^\rho(\Phi)\!\downarrow$.

Case $\mathsf{f} \in \{\mathsf{sdec}, \mathsf{adec}, \mathsf{check}\}$: If we first assume that the root symbol f is not reduced in $M\Phi\!\downarrow$ then the proof is similar to the previous case. Thus, we focus on the case where the root occurence of f is reduced. In this case, there exists v_1, v_2 such that

— $M_1\Phi\!\downarrow = \mathsf{senc}(v_1, v_2)$, $M_2\Phi\!\downarrow = v_2$ and $M\Phi\!\downarrow = v_1$. According to the definition of δ_a^ρ and δ_b^ρ, we know that there exists $i \in \{a, b\}$ such that $\delta_i^\rho(\mathsf{senc}(v_1, v_2)) = \mathsf{senc}(\delta_i^\rho(v_1), \delta_i^\rho(v_2))$. For such i, we have that $\mathsf{sdec}(\delta_i^\rho(M_1\Phi\!\downarrow), \delta_i^\rho(M_2\Phi\!\downarrow))\!\downarrow = \delta_i^\rho(M\Phi\!\downarrow)$. But by applying our inductive hypothesis on M_1 and M_2, we obtain $\delta_i^\rho(M\Phi\!\downarrow) = \mathsf{sdec}(M_1\delta^\rho(\Phi)\!\downarrow, M_2\delta^\rho(\Phi)\!\downarrow)\!\downarrow = M\delta^\rho(\Phi)\!\downarrow$.

— $M_1\Phi\downarrow = \mathsf{aenc}(v_1, \mathsf{pk}(v_2))$, $M_2\Phi\downarrow = v_2$ and $M\Phi\downarrow = v_1$: According to the definition of δ^ρ_a and δ^ρ_b, there exists $i \in \{a, b\}$ such that $\delta^\rho_i(\mathsf{aenc}(v_1, \mathsf{pk}(v_2))) = \mathsf{aenc}(\delta^\rho_i(v_1), \mathsf{pk}(\delta^\rho_i(v_2)))$. For such i, we have that $\mathsf{adec}(\delta^\rho_i(M_1\Phi\downarrow), \delta^\rho_i(M_2\Phi\downarrow))\downarrow = \delta^\rho_i(M\Phi\downarrow)$. But by applying our inductive hypothesis on M_1 and M_2, we obtain $\delta^\rho_i(M\Phi\downarrow) = \mathsf{adec}(M_1\delta^\rho(\Phi)\downarrow, M_2\delta^\rho(\Phi)\downarrow)\downarrow = M\delta^\rho(\Phi)\downarrow$.

— $M_1\Phi\downarrow = \mathsf{sign}(v_1, v_2)$, $M_2\Phi\downarrow = \mathsf{vk}(v_2)$ and $M\Phi\downarrow = v_1$: According to the definition of δ^ρ_a and δ^ρ_b, there exists $i \in \{a, b\}$ such that $\delta^\rho_i(\mathsf{sign}(v_1, v_2)) = \mathsf{sign}(\delta^\rho_i(v_1), \delta^\rho_i(v_2))$. For such i, we have that $\mathsf{adec}(\delta^\rho_i(M_1\Phi\downarrow), \delta^\rho_i(M_2\Phi\downarrow))\downarrow = \delta^\rho_i(M\Phi\downarrow)$. But by applying our inductive hypothesis on M_1 and M_2, we obtain $\delta^\rho_i(M\Phi\downarrow) = \mathsf{adec}(M_1\delta^\rho(\Phi)\downarrow, M_2\delta^\rho(\Phi)\downarrow)\downarrow = M\delta^\rho(\Phi)\downarrow$.

Case $\mathsf{f} \in \mathsf{proj}_j$, $j = 1, 2$*:* Similarly to the previous case, we only focus on the case where f is reduced in $M\Phi\downarrow$. In such a case, there exists u_1, u_2 such that $M_1\Phi\downarrow = \langle u_1, u_2\rangle$ and $M\Phi\downarrow = u_j$. Thus for all $i \in \{1, 2\}$, we have that $\delta^\rho_i(M\Phi\downarrow) = \delta^\rho_i(u_j)$ and $\delta^\rho_i(M_1\Phi\downarrow) = \langle \delta^\rho_i(u_1), delta^\rho_i(u_2)\rangle$. Hence we deduce that $\mathsf{f}(\delta^\rho_i(M_1\Phi\downarrow))\downarrow = \delta^\rho_i(u_j) = \delta^\rho_i(M\Phi\downarrow)$. By inductive hypothesis on M_1, there exist i_0 such that $\delta^\rho_{i_0}(M_1\Phi\downarrow) = M_1\delta^\rho(\Phi)\downarrow$. Therefore, we deduce that $\delta^\rho_{i_0}(M\Phi\downarrow) = \mathsf{f}((M_1\delta^\rho(\Phi)\downarrow))\downarrow = M\delta^\rho(\Phi)\downarrow$.

It remains to prove that $\delta^\rho_a(M\Phi\downarrow) = \delta^\rho_b(M\Phi\downarrow)$. We proved that there exists $i_0 \in \{a, b\}$ such that $\delta^\rho_{i_0}(M\Phi\downarrow) = M\delta^\rho(\Phi)\downarrow$. But by Lemma B.7, we know that there exists a context C built over $\{\langle\rangle\}$, and $v_1, \ldots, v_m$ terms such that $M\Phi\downarrow = C[v_1, \ldots, v_m]$ and for all $i \in \{1, \ldots, m\}$:

— either $v_i \in \mathsf{Flawed}(M\Phi\downarrow)$

— or $v_i \in Fct_{\mathcal{F}_0}(M\Phi\downarrow)$ and $\delta^\rho_a(v_i) = \delta^\rho_b(v_i)$.

— or $v_i = \mathsf{f}(n)$ for some $f \in \{\mathsf{pk}, \mathsf{vk}\}$ and $n \in \mathcal{N}$,

— or $v_i \in \mathcal{N}$.

First of all, note that C being built upon $\{\langle\rangle\}$ means that v_i is deducible in Φ, for all $i \in \{1, \ldots, m\}$. Furthermore, we also have that $\delta^\rho_{i_0}(M\Phi\downarrow) = C[\delta^\rho_{i_0}(v_1), \ldots, \delta^\rho_{i_0}(v_m)]$. But we previously proved that $\delta^\rho_{i_0}(M\Phi\downarrow) = M\delta^\rho(\Phi)\downarrow$, thus $\delta^\rho_{i_0}(v_i)$ is deducible from $\delta^\rho(\Phi)$, for all $i \in \{1, \ldots, m\}$.

Case $v_i \in \mathsf{Flawed}(M\Phi\downarrow)$*:* There exists $w_1, \ldots, w_\ell$ terms and a function symbol f such that $v_i = \mathsf{f}(w_1, \ldots, w_\ell)$. By Lemma B.6, there exists $N_1, \ldots, N_\ell$ such that for all $k \in \{1, \ldots, \ell\}$, $\mathcal{M}(N_k) < \mathcal{M}(M)$ and $N_k\Phi\downarrow = w_k$. Hence, by applying inductive hypothesis on $N_1, \ldots, N_\ell$, we obtain that $\delta^\rho_a(N_k\Phi\downarrow) = \delta^\rho_b(N_k\Phi\downarrow)$, for all $k \in \{1, \ldots, \ell\}$. Thus, thanks to v_i being in normal form, we can conclude that $\delta^\rho_a(v_i) = \delta^\rho_b(v_i)$.

Case $v_i \in Fct_{\mathcal{F}_0}(M\Phi\downarrow)$*:* we also have $\delta^\rho_a(v_i) = \delta^\rho_b(v_i)$.

Case $v_i = \mathsf{f}(n)$ *for some* $f \in \{\mathsf{pk}, \mathsf{vk}\}$ *and* $n \in \mathcal{N}$*:* By hypothesis, we know that either $\Phi \not\vdash \mathsf{f}(k)$, for all $k \in \mathrm{img}(\rho)$; or $\delta^\rho(\Phi) \not\vdash \mathsf{f}(k)$, for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$. Since we showed that v_i is deducible from Φ and $\delta^\rho_{i_0}(v_i)$ is deducible from $\delta^\rho(\Phi)$, both hypotheses imply that $n \notin \mathrm{img}(\rho)$ and so $\delta^\rho_a(v_i) = \delta^\rho_b(v_i)$.

Case $v_i \in \mathcal{N}$*:* By hypothesis we know that either $\Phi \not\vdash k$, for all $k \in \mathrm{img}(\rho)$; or $\delta^\rho(\Phi) \not\vdash k$, for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$. Since we showed that v_i is deducible from Φ and $\delta^\rho_{i_0}(v_i)$ is deducible from $\delta^\rho(\Phi)$, both hypotheses imply that $v_i \notin \mathrm{img}(\rho)$ and so $\delta^\rho_a(v_i) = \delta^\rho_b(v_i)$. □

Lemma 5.10 (Soundness)**.** *Let* $S = (\mathcal{E}_S; \mathcal{P}_S; \Phi_S)$, $S' = (\mathcal{E}'_S; \mathcal{P}'_S; \Phi'_S)$ *and* $D = (\mathcal{E}_D; \mathcal{P}_D; \Phi_D)$ *be three bounded intermediate processes. Assume that* $S \xrightarrow{\ell} S'$, Φ_S *is well-tagged and there exists a derived well-tagged multi-set of processes* $(\mathcal{P}_0, \alpha)$ *and a renaming* ρ*, such that*

— $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_S$, $\mathrm{dom}(\rho) \cap fnames(\mathcal{P}_S, \Phi_S) = \emptyset$*; and*

— $\mathcal{E}_S = \mathcal{E}_D$, $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$*; and*

— $\mathcal{P}_S = \mathcal{P}_0\alpha$ *and* $\mathcal{P}_D\downarrow = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha\downarrow)\downarrow$*; and*

— *for all traces* (tr, Φ) *of* D*, for all* $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\Phi \not\vdash k$, $\Phi \not\vdash \mathsf{pk}(k)$ *and* $\Phi \not\vdash \mathsf{vk}(k)$.

There exists a bounded intermediate process $D' = (\mathcal{E}'_D; \mathcal{P}'_D; \Phi'_D)$*, a derived well tagged multi-set of processes* $(\mathcal{P}'_0, \alpha')$ *such that* Φ'_S *is well-tagged and:*

— $\mathcal{E}'_S = \mathcal{E}_S = \mathcal{E}'_D$ *and* $\Phi'_D\downarrow = \delta^\rho(\Phi'_S\downarrow)$*; and*

— $\mathcal{P}'_S\downarrow = \mathcal{P}'_0\alpha'\downarrow$ *and* $\mathcal{P}'_D\downarrow = \delta^\rho(\mathcal{P}'_0)\delta^\rho(\alpha'\downarrow)\downarrow$*; and*

— $D \xrightarrow{\ell} D'$.

Proof. We show this result by case-by-case analysis on the rule:

Case of the rule THEN*:* In this case, there exists ϕ formula and Q_1, Q_2 plain processes and $\mathcal{Q}$ a multi-set of processes such that $\mathcal{P}_S = \{\mathsf{if}\ \phi\ \mathsf{then}\ Q_1\ \mathsf{else}\ Q_2\} \uplus \mathcal{Q}$, $\mathcal{P}'_S = \{Q_1\} \uplus \mathcal{Q}$, $\mathcal{E}_S = \mathcal{E}'_S$ and $\Phi_S = \Phi'_S$ with $i = \mathsf{col}(\mathcal{Q}_1) = \mathsf{col}(\mathcal{Q}_2)$ and ϕ is a conjunction of equation. Furthermore, we have that for all equation $u = v$ of ϕ, $u =_{\mathsf{E}} v$.

Since $\mathcal{P}_S = \mathcal{P}_0\alpha$, then there exists Q_1^0, Q_2^0 plain processes and $\mathcal{Q}^0$ a multi-set of processes such that $Q_1^0\alpha = Q_1$, $Q_2^0\alpha = Q_2$ and $\mathcal{Q}^0\alpha = \mathcal{Q}$. Furthermore, either (a) there exists u such that ϕ is the formula $\mathsf{test}_i([u]_i)\alpha$, or (b) there exists u_1, u_2 such that ϕ is the formula $[u_1]_i\alpha = [u_2]_i\alpha$ and $\alpha \vDash \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i)$.

But we also have $\mathcal{P}_D{\downarrow} = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha{\downarrow}){\downarrow}$, which means that there exists ϕ' formula and Q'_1, Q'_2 plain processes and $\mathcal{P}'$ a multi-set of processes such that $\mathcal{P}_D = \{\mathsf{if}\ \phi'\ \mathsf{then}\ Q'_1\ \mathsf{else}\ Q'_2)\} \uplus \mathcal{Q}'$ with $Q'_1{\downarrow} = \delta_i^\rho(Q_1^0)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$, $Q'_2{\downarrow} = \delta_i^\rho(Q_2^0)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$, $\mathcal{Q}'{\downarrow} = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha{\downarrow}){\downarrow}$ and $\Phi_D{\downarrow} = \delta^\rho(\Phi_S{\downarrow})$. Furthermore, in case (a) $\phi'{\downarrow}$ is the formula $\mathsf{test}_i(\delta_i^\rho([u]_i))\delta_i^\rho(\alpha{\downarrow}){\downarrow}$; and in case (b) ϕ' is the formula $u' = v'$ where $u{\downarrow} = \delta_i^\rho([u_1]_i)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$ and $v'{\downarrow} = \delta_i^\rho([u_2]_i)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$.

In case (a), since for all equation in $u = v$ in ϕ, $u =_{\mathsf{E}} v$, then $u{\downarrow} = v{\downarrow}$. Thus it is equivalent to $(\alpha{\downarrow}) \vDash \mathsf{test}_i([u]_i)$. But by Lemma B.3, we know that this is equivalent to $\delta_i^\rho(\alpha{\downarrow}) \vDash \mathsf{test}_i(\delta_i^\rho([u]_i))$. Thus we have that for all equation $u = v$ of $\mathsf{test}_i(\delta_i^\rho([u]_i))$, $u\delta_i^\rho(\alpha{\downarrow}){\downarrow} = v\delta_i^\rho(\alpha{\downarrow}){\downarrow}$. Since $\phi'{\downarrow}$ is the formula $\mathsf{test}_i(\delta_i^\rho([u]_i))\delta_i^\rho(\alpha{\downarrow}){\downarrow}$, we can conclude that for all equation $u = v$ of ϕ', $u{\downarrow} = v{\downarrow}$ and so $u =_{\mathsf{E}} v$.

In case (b), we know that $\alpha{\downarrow} \vDash \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i)$ and $[u_1]_i\alpha{\downarrow} = [u_2]_i\alpha{\downarrow}$. Thus by Corollary B.1, we can deduce that $\delta_i^\rho([u_1]_i)\delta_i^\rho(\alpha{\downarrow}){\downarrow} = \delta_i^\rho([u_2]_i)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$ which means that $u'{\downarrow} = v'{\downarrow}$ and so $u' =_{\mathsf{E}} v'$.

ϕ' being satisfied allows us to deduce that $D \xrightarrow{\ell} (\mathcal{E}_D; \{Q'_1\} \uplus \mathcal{Q}'; \Phi_D)$. But we know that on one hand $Q_1^0\alpha = Q_1$ and $\mathcal{Q}^0\alpha = \mathcal{Q}$, and on the other hand, $Q'_1{\downarrow} = \delta_i^\rho(Q_1^0)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$, $\mathcal{Q}'{\downarrow} = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha{\downarrow}){\downarrow}$ and $\Phi_D{\downarrow} = \delta^\rho(\Phi_S{\downarrow})$. Thus with $\alpha' = \alpha$ and $\mathcal{P}'_0 = \mathcal{Q}_1^0 \uplus \mathcal{Q}^0$, the result holds.

Case of the rule ELSE*:* This case similar to the rule THEN.

Case of the rule COMM*:* In this case, there exists p, u, x terms, Q_1, Q_2 processes, and $\mathcal{Q}$ multi-set of processes such that $\mathcal{P}_S = \{\mathsf{out}(p, u).Q_1; \mathsf{in}(p, x).Q_2\} \uplus \mathcal{Q}$, $\mathcal{P}'_S = \{Q_1; Q_2\{x \mapsto u\}\} \uplus \mathcal{Q}$, $\mathcal{E}_S = \mathcal{E}'_S$ and $\Phi_S = \Phi'_S$. Assume that $\mathsf{col}(\mathsf{out}(p, u).Q_1) = i$ and $\mathsf{col}(\mathsf{in}(p, x).Q_2) = j$. We distinguish two cases:

Case $p \in \mathcal{E}_s$: Because processes of different color do not share private channel, it is necessary the case that $i = j$.

Since $\mathcal{P}_S = \mathcal{P}_0\alpha$, then there exists Q_1^0, Q_2^0 processes, $\mathcal{Q}^0$ multi-set of processes, v term such that $Q_1^0\alpha = Q_1$, $Q_2^0\alpha = Q_2$, $\mathcal{Q}^0\alpha = \mathcal{Q}$ and $u = [v]_i\alpha$. Furthermore, since $(\mathcal{P}_0, \alpha)$ is an original well tagged process, we have that $\alpha \vDash \mathsf{test}_i([v]_i)$.

But we also have that $\mathcal{P}_D{\downarrow} = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha{\downarrow}){\downarrow}$, which means that there exists p'', p', u', x' terms, Q'_1, Q'_2 processes, and $\mathcal{Q}'$ multi-set of processes such that $\mathcal{P}_D = \{\mathsf{out}(p', u').Q'_1; \mathsf{in}(p'', x).Q'_2\} \uplus \mathcal{Q}'$, $\mathcal{E}_D = \mathcal{E}_S$ with $Q'_1{\downarrow} = \delta_i^\rho(Q_1^0)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$, $Q'_2{\downarrow} = \delta_i^\rho(Q_2^0)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$, $\mathcal{Q}'{\downarrow} = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha{\downarrow}){\downarrow}$, $\Phi_D{\downarrow} = \delta^\rho(\Phi_S{\downarrow})$, $p'' = \delta_i^\rho(p) = p'$ and $u'{\downarrow} = \delta_i^\rho([v]_i)\delta_i^\rho(\alpha{\downarrow}){\downarrow}$.

Therefore, we have that $(\mathcal{E}_D; \mathcal{P}_D; \Phi_D) \xrightarrow{\ell} (\mathcal{E}_D; \{Q'_1; Q'_2\{x \mapsto u'\}\} \uplus \mathcal{P}'; \Phi_D)$.

Let's denote $\alpha' = \alpha \cup \{x \mapsto u\}$ with $\mathsf{col}(x) = i$. Since $Q_2 = Q_2^0\alpha$, we have that $Q_2\{x \mapsto u\} = Q_2^0\alpha'$. Furthermore, $Q'_2\{x \mapsto u'\}{\downarrow} = (Q'_2{\downarrow})\{x \mapsto (u'{\downarrow})\}{\downarrow}$. But thanks to Lemma B.2 and $(\alpha{\downarrow}) \vDash \mathsf{test}_i([v]_i)$, we have $\delta_i^\rho([v]_i)\delta_i^\rho(\alpha{\downarrow}){\downarrow} = \delta_i^\rho([v]_i(\alpha{\downarrow})){\downarrow} = \delta_i^\rho([v]_i\alpha{\downarrow}) = \delta_i^\rho(u{\downarrow})$. Thus, we deduce that $Q'_2\{x \mapsto u'\}{\downarrow} = \delta_i^\rho(Q_2^0)\delta_i^\rho(\alpha{\downarrow})\{x \mapsto \delta_i^\rho(u{\downarrow})\}{\downarrow} = \delta_i^\rho(Q_2^0)\delta_i^\rho(\alpha'{\downarrow}){\downarrow}$.

Let $\mathcal{P}'_0 = \{Q_1^0; Q_2^0\} \uplus \mathcal{Q}^0$. Since x doesn't appears in Q_1^0 and $\mathcal{Q}^0$, we conclude that $\mathcal{P}'_0\alpha' = \mathcal{P}'_S$ and $\delta^\rho(\mathcal{P}'_0)\delta^\rho(\alpha'{\downarrow}){\downarrow} = \mathcal{P}'_D{\downarrow}$. Hence the result holds.

Case $p \notin \mathcal{E}_s$: Since $\mathcal{P}_S = \mathcal{P}_0\alpha$, then there exists Q_1^0, Q_2^0 processes, $\mathcal{Q}^0$ multi-set of processes, v term such that $Q_1^0\alpha = Q_1$, $Q_2^0\alpha = \mathcal{Q}_2$, $Q^0\alpha = \mathcal{Q}$ and $u = [v]_i\alpha$. Furthermore, since $(\mathcal{P}_0, \alpha)$ is an original well tagged process, we have that $\alpha \vDash \mathsf{test}_i([v]_i)$.

But we also have that $\mathcal{P}_D\downarrow = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha\downarrow)\downarrow$, which means that there exists p'', p', u', x' terms, Q_1', Q_2' processes and $\mathcal{Q}'$ multi-set of processes such that $\mathcal{P}_D = \{\mathsf{out}(p', u').Q_1'; \mathsf{in}(p'', x).Q_2'\} \uplus \mathcal{Q}'$, $\mathcal{E}_D = \mathcal{E}_S$ with $Q_1'\downarrow = \delta_i^\rho(Q_1^0)\delta_i^\rho(\alpha\downarrow)\downarrow$, $Q_2'\downarrow = \delta_j^\rho(Q_2^0)\delta_j^\rho(\alpha\downarrow)\downarrow$, $\mathcal{Q}'\downarrow = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha\downarrow)\downarrow$, $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$, $p'' = \delta_j^\rho(p)$, $p' = \delta_i^\rho(p)$ and $u'\downarrow = \delta_i^\rho([v]_i)\delta_i^\rho(\alpha\downarrow)\downarrow$.

We assumed that all names in $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho)$ are of base type. Thus, we can deduce that $\delta_i^\rho(p) = \delta_j^\rho(p) = p$ and so $p'' = p'$. Therefore, we have that $(\mathcal{E}_D; \mathcal{P}_D; \Phi_D) \xrightarrow{\ell} (\mathcal{E}_D; \mathcal{Q}_1' \uplus \mathcal{Q}_2'\{x \mapsto u'\} \uplus \mathcal{P}'; \Phi_D)$.

Moreover, we have have $\delta_a^\rho(u\downarrow) = \delta_b^\rho(u\downarrow)$. Indeed by hypothesis, we know that for all trace $(\mathsf{tr}, \nu\mathcal{E}_D.\Phi)$ of D, $\nu\mathcal{E}_D.\Phi \not\vdash k$, $\nu\mathcal{E}_D.\Phi \not\vdash \mathsf{pk}(k)$ and $\nu\mathcal{E}_D.\Phi \not\vdash \mathsf{vk}(k)$, for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$. But $S \xrightarrow{\nu ax_n.\mathsf{out}(p, ax_n)} (\mathcal{E}_S; \{Q_1; \mathsf{in}(p, x).Q_2\}; \Phi_S \cup \{ax_n \rhd u\})$. Since $u = [v]_i\alpha$, we have that $\Phi_S \cup \{ax_n \rhd u\}$ is a frame well-formed. Moreover, we know that $u'\downarrow = \delta_i^\rho([v]_i)\delta_i^\rho(\alpha\downarrow)\downarrow$, thus thanks to Lemma B.2 and $(\alpha\downarrow) \vDash \mathsf{test}_i([v]_i)$, we deduce that $u'\downarrow = \delta_i^\rho([v]_i(\alpha\downarrow))\downarrow = \delta_i^\rho([v]_i\alpha\downarrow) = \delta_i^\rho(u\downarrow)$. If we denote $\Phi = \Phi_S \cup \{ax_n \rhd u\}$, we therefore have that $\delta^\rho(\Phi\downarrow) = \Phi_D\downarrow \cup \{ax_n \rhd u'\downarrow\}$. On the other hand, we also have $D \xrightarrow{\nu ax_n.\mathsf{out}(p, ax_n)} (\mathcal{E}_S; \{Q_1'; \mathsf{in}(p, x).Q_2'\}; \Phi_D \cup \{ax_n \rhd u'\})$. Thus, thanks to Lemma 5.9 and since $u\downarrow = ax_n\Phi\downarrow$, we obtain that $\delta_a^\rho(u\downarrow) = ax_n\delta^\rho(\Phi\downarrow)\downarrow = \delta_b^\rho(u\downarrow)$.

Let's denote $\alpha' = \alpha \cup \{x \mapsto u\}$. Since $Q_2 = Q_2^0\alpha$, we have that $Q_2\{x \mapsto u\} = Q_2^0\alpha'$. Furthermore, $Q_2'\{x \mapsto u'\}\downarrow = (Q_2'\downarrow)\{x \mapsto (u'\downarrow)\}\downarrow$. But we already showed that $u'\downarrow = \delta_i^\rho(u\downarrow)$, thus, we deduce that $Q_2'\{x \mapsto u'\}\downarrow = \delta_j^\rho(Q_2^0)\delta_j^\rho(\alpha\downarrow)\{x \mapsto \delta_i^\rho(u\downarrow)\}\downarrow$. Since $\delta_a^\rho(u\downarrow) = \delta_b^\rho(u\downarrow)$, we have that $Q_2'\{x \mapsto u'\}\downarrow = \delta_j^\rho(Q_2^0)\delta_j^\rho(\alpha'\downarrow)\downarrow$.

Let $\mathcal{P}_0' = \{Q_1^0; Q_2^0\} \uplus \mathcal{Q}^0$. Since x doesn't appears in Q_1^0 and $\mathcal{Q}^0$, we conclude that $\mathcal{P}_0'\alpha' = \mathcal{P}_S'$ and $\delta^\rho(\mathcal{P}_0')\delta^\rho(\alpha'\downarrow)\downarrow = \mathcal{P}_D'\downarrow$. Hence the result holds.

Case of the rule In*:* In this case, there exists p, x, u, M terms, Q_1 a process, $\mathcal{Q}$ a multi-set of processes such that $p \notin \mathcal{E}_S$, $M\Phi_S\downarrow = u$, $\mathit{fvars}(M) \subseteq \mathrm{dom}(\Phi_S)$ and $\mathit{fnames}(M) \cap \mathcal{E}_S = \emptyset$. Furthermore, we have that $\mathcal{P}_S = \{\mathsf{in}(p, x).Q_1\} \uplus \mathcal{Q}$, $\mathcal{P}_S' = \{Q_1\{x \mapsto u\}\} \uplus \mathcal{Q}$, $\Phi_S = \Phi_S'$, $\mathcal{E}_S = \mathcal{E}_S'$ and $\ell = in(p, M)$. Since $\mathcal{P}_S = \mathcal{P}_0\alpha$, then there exists Q_1^0 a process and $\mathcal{Q}^0$ a multi-set of processes such that $Q_1^0\alpha = Q_1$ and $\mathcal{Q}^0\alpha = \mathcal{Q}$. Assume that $\mathsf{col}(\mathsf{in}(p, x).Q_1) = i \in \{a, b\}$.

But we also have that $\mathcal{P}_D\downarrow = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha\downarrow)\downarrow$, which means that there exists p' term, Q_1' a process, $\mathcal{Q}'$ a multi-set of processes such that $\mathcal{P}_D = \{\mathsf{in}(p', x).Q_1'\} \uplus \mathcal{Q}'$, $\mathcal{E}_D = \mathcal{E}_S$ with $Q_1'\downarrow = \delta_i^\rho(Q_1^0)\delta_i^\rho(\alpha\downarrow)\downarrow$, $\mathcal{Q}'\downarrow = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha\downarrow)\downarrow$, $p' = \delta_i^\rho(p)$ and $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$.

p' and p are both channel type term and we assumed that all the names in $\mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$ are names of base type. Thus we have that $p' = p$. Furthermore, $\mathcal{E}_D = \mathcal{E}_S$ which means that $p' \notin \mathcal{E}_D$ and $\mathit{fnames}(M) \cap \mathcal{E}_D = \emptyset$. We also have that $\Phi_D\downarrow = \delta(\Phi_S\downarrow)$ which means that $\mathrm{dom}(\Phi_D) = \mathrm{dom}(\Phi_S)$ and so $\mathit{fvars}(M) \subseteq \mathrm{dom}(\Phi_D)$. Thus, we can deduce that $(\mathcal{E}_D; \mathcal{P}_D; \Phi_D) \xrightarrow{\ell} (\mathcal{E}_D; \{Q_1'\{x \mapsto u'\}\} \uplus \mathcal{Q}'; \Phi_D)$ where $u' = M\Phi_D$.

By hypothesis, we assumed that for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\nu\mathcal{E}.\Phi_D \not\vdash k$, $\nu\mathcal{E}.\Phi_D' \not\vdash \mathsf{pk}(k)$, $\nu\mathcal{E}.\Phi_D \not\vdash \mathsf{vk}(k)$. Since $\delta^\rho(\Phi_S\downarrow) = \Phi_D\downarrow$, we can apply Lemma 5.9 and so $\delta_a^\rho(M(\Phi_S\downarrow)\downarrow) = \delta_b^\rho(M(\Phi_S\downarrow)\downarrow) = M\delta^\rho(\Phi_S\downarrow)\downarrow = M\Phi_D\downarrow$. But $M(\Phi_S\downarrow)\downarrow = M\Phi_S\downarrow = u\downarrow$ and $M(\Phi_D\downarrow)\downarrow = u'\downarrow$. Thus, we have that $\delta_a^\rho(u\downarrow) = \delta_b^\rho(u\downarrow) = u'\downarrow$. Since $Q_1'\downarrow = \delta_i^\rho(\mathcal{Q}_1^0)\delta_i^\rho(\alpha\downarrow)\downarrow$, we deduce that $Q_1'\{x \mapsto u'\}\downarrow = \delta_i^\rho(Q_1^0)\delta_i^\rho(\alpha\downarrow)\{x \mapsto \delta_i^\rho(u\downarrow)\}\downarrow$.

At last, let $\mathcal{P}_0' = \{Q_1^0\} \uplus \mathcal{Q}^0$ and let $\alpha' = \alpha \cup \{x \mapsto u\}$. We have that $\delta^\rho(\alpha'\downarrow) = \delta^\rho(\alpha\downarrow) \cup \{x \mapsto \delta_i^\rho(u\downarrow)\}$. Thus, we conclude that $Q_1'\{x \mapsto u'\}\downarrow = \delta_i^\rho(\mathcal{Q}_1^0)\delta_i^\rho(\alpha'\downarrow)\downarrow$ and since x does not appear in $\mathcal{Q}^0$, $\mathcal{Q}'\downarrow = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha\downarrow)\downarrow = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha'\downarrow)\downarrow$. Hence the result holds.

Case of the rule Out-T*:* In this case, there exists u, p terms, Q_1 a process, and $\mathcal{Q}$ a multi-set of processes such that $\mathcal{P}_S = \{\mathsf{out}(p, u).Q_1\} \uplus \mathcal{Q}$, $\mathcal{P}_S' = \{Q_1\} \uplus \mathcal{Q}$, $\mathcal{E}_S = \mathcal{E}_S'$ and $\Phi_S' = \Phi_S \cup \{ax_n \rhd u\}$. Furthermore, we have that $\ell = \nu ax_n.\mathsf{out}(p, ax_n)$, $p \notin \mathcal{E}_S$ and ax_n is a parameter such that $n - 1 = |\Phi_S|$. Assume that $\mathsf{col}(\mathsf{out}(p, u).Q_1) = i \in \{a, b\}$.

Since $\mathcal{P}_S = \mathcal{P}_0\alpha$, then there exists Q_1^0 a process, $\mathcal{Q}^0$ a multi-set of processes and v a term such that $Q_1^0\alpha = Q_1$, $\mathcal{Q}^0\alpha = \mathcal{Q}$ and $u = [v]_i\alpha$ where $i = \mathsf{col}(\mathsf{out}(p, u).\mathcal{Q}_1)$. Furthermore, since $(\mathcal{P}_0, \alpha)$ is an original well tagged process, we have that $\alpha \vDash \mathsf{test}_i([v]_i)$. Hence we deduce that Φ_S' is well-formed.

But we also have that $\mathcal{P}_D\downarrow = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha\downarrow)\downarrow$, which means that there exists p', u' terms, Q_1' a

process and $\mathcal{Q}'$ a multi-set of processes such that $\mathcal{P}'_D = \{\mathsf{out}(p', u').Q'_1\} \uplus \mathcal{Q}'$, $\mathcal{E}_D = \mathcal{E}_S$ with $Q'_1\downarrow = \delta^\rho_i(Q^0_1)\delta^\rho_i(\alpha\downarrow)\downarrow$, $\mathcal{Q}'\downarrow = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha\downarrow)\downarrow$, $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$, $p' = \delta^\rho_i(p)$ and $u'\downarrow = \delta^\rho_i([v]_i)\delta^\rho_i(\alpha\downarrow)\downarrow$.

p' and p are both channel type term and we assumed that $\mathrm{dom}(\rho)$ and $\mathrm{img}(\rho)$ only contains name of base type. Thus, we have that $p' = p$. Furthermore, $\mathcal{E}_D = \mathcal{E}_S$ which means that $p' \notin \mathcal{E}_D$. We also have that $\Phi_D\downarrow = \delta(\Phi_S\downarrow)$ which means that $|\Phi_D| = |\Phi_S| = n - 1$. Thus, we can deduce that $(\mathcal{E}_D; \mathcal{P}_D; \Phi_D) \xrightarrow{\ell} (\mathcal{E}_D; \{Q'_1\} \uplus \mathcal{Q}'; \Phi_D \cup \{w_n \rhd u'\})$.

Thanks to Lemma B.2 and $(\alpha\downarrow) \vDash \mathsf{test}_i([v]_i)$, we have $\delta^\rho_i([v]_i)\delta^\rho_i(\alpha\downarrow)\downarrow = \delta^\rho_i([v]_i(\alpha\downarrow))\downarrow = \delta^\rho_i([v]_i\alpha\downarrow) = \delta^\rho_i(u\downarrow)$. But $u'\downarrow = \delta^\rho_i([v]_i)\delta^\rho_i(\alpha\downarrow)\downarrow$, which means that $u'\downarrow = \delta^\rho_i(u\downarrow)$. Since $\mathsf{col}(ax_n) = i$, we can deduce that $\Phi'_D\downarrow = \delta^\rho(\Phi'_S\downarrow)$. We conclude with $\mathcal{P}'_0 = \{Q^0_1\} \uplus \mathcal{Q}^0$ and $\alpha' = \alpha$. We conclude that $\mathcal{P}'_0\alpha' = \mathcal{P}'_S$ and $\delta^\rho(\mathcal{P}'_0)\delta^\rho(\alpha'\downarrow)\downarrow = \mathcal{P}'_D\downarrow$. Hence the result holds.

Case of the rule Out-Ch*:* Obvious since $\mathrm{dom}(\rho)$ and $\mathrm{img}(\rho)$ only contains name of base type

Case of the rule Open-Ch*:* Obvious since $\mathrm{dom}(\rho)$ and $\mathrm{img}(\rho)$ only contains name of base type

Case of the rule Par*:* In this case, there exists Q_1, Q_2 processes and $\mathcal{Q}$ a multi-set of processes such that $\mathcal{P}_S = \{Q_1 \mid Q_2\} \uplus \mathcal{Q}$, $\mathcal{P}'_S = \{Q_1; Q_2\} \uplus \mathcal{Q}$, $\mathcal{E}_S = \mathcal{E}'_S$ and $\Phi_S = \Phi'_S$. Assume that $\mathsf{col}(\mathcal{Q}_1) = \mathsf{col}(\mathcal{Q}_2) = i$.

Since $\mathcal{P}_S = \mathcal{P}_0\alpha$, then there exists Q^0_1, Q^0_2 and $\mathcal{Q}^0$ processes such that $Q^0_1\alpha = Q_1$, $Q^0_2\alpha = Q_2$, and $\mathcal{Q}^0\alpha = \mathcal{Q}$.

But we also have that $\mathcal{P}_D\downarrow = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha\downarrow)\downarrow$, which means that there exists Q'_1, Q'_2 processes and $\mathcal{Q}'$ multi-set of processes such that $\mathcal{P}'_D = \{Q_1; Q'_2\} \uplus \mathcal{Q}'$, $\mathcal{E}_D = \mathcal{E}_S$ with $Q'_1\downarrow = \delta^\rho_i(Q^0_1)\delta^\rho_i(\alpha\downarrow)\downarrow$, $Q'_2\downarrow = \delta^\rho_i(Q^0_2)\delta^\rho_i(\alpha\downarrow)\downarrow$, $\mathcal{Q}'\downarrow = \delta^\rho(\mathcal{Q}^0)\delta^\rho(\alpha\downarrow)\downarrow$, and $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$. Therefore, we have that $(\mathcal{E}_D; \mathcal{P}_D; \Phi_D) \xrightarrow{\ell} (\mathcal{E}_D; \{Q'_1; Q'_2\} \uplus \mathcal{P}'; \Phi_D)$.

Let $\alpha' = \alpha$ and $\mathcal{P}'_0 = \{Q^0_1; Q^0_2\} \uplus \mathcal{Q}^0$. We conclude that $\mathcal{P}'_0\alpha' = \mathcal{P}'_S$ and $\delta^\rho(\mathcal{P}'_0)\delta^\rho(\alpha'\downarrow)\downarrow = \mathcal{P}'_D\downarrow$. Hence the result holds. □

Lemma 5.11 (Completeness)**.** *Let $S = (\mathcal{E}_S; \mathcal{P}_S; \Phi_S)$, $D = (\mathcal{E}_D; \mathcal{P}_D; \Phi_D)$ and $D' = (\mathcal{E}'_D; \mathcal{P}'_D; \Phi'_D)$ and be three bounded intermediate processes. Assume that $D \xrightarrow{\ell} D'$, Φ_S is a derived well-tagged frame and there exists a derived well-tagged multi-set of processes $(\mathcal{P}_0, \alpha)$ and a renaming ρ, such that*

- $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_S$, $\mathrm{dom}(\rho) \cap \mathit{fnames}(\mathcal{P}_S, \Phi_S) = \emptyset$*; and*
- $\mathcal{E}_S = \mathcal{E}_D$, $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$*; and*
- $\mathcal{P}_S = \mathcal{P}_0\alpha$ *and* $\mathcal{P}_D\downarrow = \delta^\rho(\mathcal{P}_0)\delta^\rho(\alpha\downarrow)\downarrow$*; and*
- *for all traces (tr, Φ) of D, for all $k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)$, $\Phi \not\vdash k$, $\Phi \not\vdash \mathsf{pk}(k)$ and $\Phi \not\vdash \mathsf{vk}(k)$.*

There exists a bounded intermediate process $S' = (\mathcal{E}'_S; \mathcal{P}'_S; \Phi'_S)$, a derived well tagged multi-set of processes $(\mathcal{P}'_0, \alpha')$ such that Φ'_S is a derived well-tagged frame and:

- $\mathcal{E}'_S = \mathcal{E}_S = \mathcal{E}'_D$ *and* $\Phi'_D\downarrow = \delta^\rho(\Phi'_S\downarrow)$*; and*
- $\mathcal{P}'_S\downarrow = \mathcal{P}'_0\alpha'\downarrow$ *and* $\mathcal{P}'_D\downarrow = \delta^\rho(\mathcal{P}'_0)\delta^\rho(\alpha'\downarrow)\downarrow$*; and*
- $S \xrightarrow{\ell} S'$.

Proof. The proof of this Lemma is almost identical to the proof of Lemma 5.10. Indeed, in the proof of Lemma 5.10, we used Lemma B.2 to show that $\alpha \vDash \mathsf{test}_i([u]_i)$ implies that $\delta^\rho_i(\alpha) \vDash \mathsf{test}_i(\delta^\rho_i([u]_i))$. But Lemma B.2 shows that those two properties are equivalent. The same goes for Corollary B.1. At last, the conditions of Lemma 5.9 are fulfilled in both lemmas thus, it can be also used in this proof. □

B.3 Proof of second result

In this section, we will focus on the proof of Theorem 5.2. We will assume, as in Section B.2, that processes and frames are colored by a or b.

We will consider the two following frames $\Phi_a = \{ax_1^a \rhd u_1, \ldots, ax_n^a \rhd u_n\}$ and $\Phi_b = \{ax_1^b \rhd u'_1, \ldots, ax_n^b \rhd u'_n\}$ such that for all $j \in \{1, \ldots, n\}$, $u_j = u'_j = \mathsf{f}(k)$, for some $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$. Furthermore assume that Φ_a (resp. Φ_b) is colored by a (resp. b).

Lemma 5.13. *Let Φ and Φ' two frames in normal form such that $\mathrm{dom}(\Phi) = \mathrm{dom}(\Phi')$. Assume that Φ and Φ' have the same colors, i.e. for all $(ax \rhd u) \in \Phi$, for all $(ax' \rhd u') \in \Phi'$, $ax = ax'$ implies $\mathsf{col}(ax \rhd u) = \mathsf{col}(ax' \rhd u')$.*

Let $\mathcal{E}$ be a set of names and let ρ a renaming such that $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$ and $\mathrm{dom}(\rho) \cap fnames(\Phi, \Phi') = \emptyset$. Let's denote $\Phi_+ = \Phi_a \uplus \Phi_b \uplus \Phi$ and $\Phi'_+ = \Phi_a \uplus \Phi_b \uplus \Phi'$.

If the following properties are satisfied:

- *$\nu\mathcal{E}.\Phi_+$, $\nu\mathcal{E}.\Phi'_+$ are well-tagged*
- *$\nu\mathcal{E}.\delta^\rho(\Phi_+) \sim \nu\mathcal{E}.\delta^\rho(\Phi'_+)$*
- *for all $u \in \{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$, $\nu\mathcal{E}.\delta^\rho(\Phi_+) \vdash u$ or $\nu\mathcal{E}.\delta^\rho(\Phi'_+) \vdash u$ implies that $u \in \mathrm{img}(\delta^\rho(\Phi_a \uplus \Phi_b))$*

then for all M such that $fnames(M) \cap \mathcal{E} = \emptyset$ and $fvars(M) \subseteq \mathrm{dom}(\Phi_+)$, there exists M_a and M_b such that $fnames(M_a, M_b) \cap \mathcal{E} = \emptyset$, $fvars(M_a, M_b) \subseteq \mathrm{dom}(\Phi_+)$ and:

1. *$\delta_a^\rho(M\Phi_+\downarrow) = M_a\delta^\rho(\Phi_+)\downarrow$ and $\delta_a^\rho(M\Phi'_+\downarrow) = M_a\delta^\rho(\Phi'_+)\downarrow$*
2. *$\delta_b^\rho(M\Phi_+\downarrow) = M_b\delta^\rho(\Phi_+)\downarrow$ and $\delta_b^\rho(M\Phi'_+\downarrow) = M_b\delta^\rho(\Phi'_+)\downarrow$*

Proof. We prove this lemma by induction on $\mathcal{M}(M)$.

Base case $\mathcal{M}(M) = (0,0)$: There exists no term M such that $|M| = 0$, thus the result holds.

Inductive step $\mathcal{M}(M) > (0,0)$: The first step of the proof will be to show that there exists $c \in \{a, b\}$, a terms M_c such that $fnames(M_c) \cap \mathcal{E} = \emptyset$, $fvars(M_c) \subseteq \mathrm{dom}(\Phi_+)$, $\delta_c^\rho(M\Phi_+\downarrow) = M_c\delta^\rho(\Phi_+)\downarrow$ and $\delta_c^\rho(M\Phi'_+\downarrow) = M_c\delta^\rho(\Phi'_+)\downarrow$. Then the second step will consist in showing that there exists another term M_d, with $d \in \{a, b\}$ and $c \neq d$ that verifies the wanted properties.

First step: Assume first that $|M| = 1$. In such a case, we have that either $M \in \mathcal{N}$ or there exists $ax \in \mathrm{dom}(\Phi_+)$ such that $M = ax$.

If $M \in \mathcal{N}$, then by hypothesis, we know that $M \notin \mathcal{E}$. Since $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}$, we can deduce that $\delta_i^\rho(M) = M$, for all $i \in \{a, b\}$. Furthermore, $M \in \mathcal{N}$ also implies that $M\Phi_+\downarrow = M$, $M\Phi'_+\downarrow = M$, $M\delta^\rho(\Phi_+)\downarrow = M$ and $M\delta^\rho(\Phi'_+)\downarrow = M$. Thus, the result holds.

Else $ax \in \mathrm{dom}(\Phi_+)$. Since $\mathrm{dom}(\Phi_+) = \mathrm{dom}(\Phi'_+)$, we know that there exists u, u' such that $(ax \rhd u) \in \Phi_+$ and $(ax \rhd u') \in \Phi'_+$. Furthermore, we assumed that Φ and Φ' have the same colors, then so do Φ_+ and Φ'_+. Hence we have that $\mathsf{col}(ax \rhd u) = \mathsf{col}(ax \rhd u')$. Let's denote $i = \mathsf{col}(ax \rhd u)$. By definition of $\delta^\rho(\Phi_+)$ and $\delta^\rho(\Phi'_+)$, we have that $ax\delta^\rho(\Phi_+) = \delta_i^\rho(ax\Phi_+)$ and $ax\delta^\rho(\Phi'_+) = \delta_i^\rho(ax\Phi'_+)$. Note that we assumed that Φ and Φ' are both in normal form, thus so are Φ_+ and Φ'_+. Hence, we can conclude that $M\delta^\rho(\Phi_+)\downarrow = \delta_i^\rho(M\Phi_+\downarrow)$ and $M\delta^\rho(\Phi'_+)\downarrow = \delta_i^\rho(M\Phi'_+\downarrow)$.

Assume now that $|M| > 1$. It implies that there exists $M_1, \ldots, M_n$ term and a function symbol f such that $M = \mathsf{f}(M_1, \ldots, M_n)$. We do a case analysis on f.

Case $\mathsf{f} \in \mathcal{F}_i \cup \mathcal{F}_{\mathsf{tag}_i}$, $i \in \{a, b\}$: In such a case, let $t = \mathsf{f}(M_1\Phi_+\downarrow, \ldots, M_n\Phi_+\downarrow)$ and $t' = \mathsf{f}(M_1\Phi'_+\downarrow, \ldots, M_n\Phi'_+\downarrow)$. Since $\mathsf{f} \in \mathcal{F}_i \cup \mathcal{F}_{\mathsf{tag}_i}$, we have that $\delta_i^\rho(t) = \mathsf{f}(\delta_i^\rho(M_1\Phi_+\downarrow), \ldots, \delta_i^\rho(M_n\Phi_+\downarrow))$ and $\delta_i^\rho(t') = \mathsf{f}(\delta_i^\rho(M_1\Phi'_+\downarrow), \ldots, \delta_i^\rho(M_n\Phi'_+\downarrow))$. But we have that $\mathcal{M}(M_1) < \mathcal{M}(M)$, $\ldots$, $\mathcal{M}(M_n) < \mathcal{M}(M)$. Thus we can apply or inductive hypothesis on $M_1, \ldots, M_n$ and so there exists $M_1^i, \ldots, M_n^i$ such that for all $k \in \{1, \ldots, n\}$, $fnames(M_k^i) \cap \mathcal{E} = \emptyset$, $fvars(M_k^i) \subseteq \mathrm{dom}(\Phi_+)$, $\delta_i^\rho(M_k\Phi_+\downarrow) = M_k^i\delta^\rho(\Phi_+)\downarrow$ and $\delta_i^\rho(M_k\Phi'_+\downarrow) = M_k^i\delta^\rho(\Phi'_+)\downarrow$. Hence, we have that $\delta_i^\rho(t)\downarrow = \mathsf{f}(M_1^i, \ldots, M_n^i)\delta^\rho(\Phi_+)\downarrow$ and $\delta_i^\rho(t')\downarrow = \mathsf{f}(M_1^i, \ldots, M_n^i)\delta^\rho(\Phi'_+)\downarrow$.

On the other hand, $t = \mathsf{f}(M_1\Phi_+\downarrow, \ldots, M_n\Phi_+\downarrow)$ implies that there exists C context built on $\mathcal{F}_i \cup \mathcal{F}_{\mathsf{tag}_i}$ and $u_1, \ldots, u_m$ terms in normal form such that $t = C[u_1, \ldots, u_m]$ with $u_1, \ldots, u_m$ factors of t. Thanks to Lemma 5.2, there exists a context D (possibly a hole) built on $\mathcal{F}_i \cup \mathcal{F}_{\mathsf{tag}_i}$ such that $t\downarrow = D[u_{j_1}, \ldots, u_{j_k}]$ with $j_1, \ldots, j_k \in \{0, \ldots, m\}$ and $u_0 = n_{min}$. But using Lemmas 5.3, 5.4 and 5.6, we can deduce that $C[\delta_i^\rho(u_1), \ldots, \delta_i^\rho(u_m)]\downarrow = D[\delta_i^\rho(u_{j_1}), \ldots, \delta_i^\rho(u_{j_k})]$. Since C and D are

both built upon $\mathcal{F}_i \cup \mathcal{F}_{\mathsf{tag}_i}$, we can conclude that $\delta_i^\rho(C[u_1,\dots,u_m])\!\downarrow = \delta_i^\rho(D[u_{j_1},\dots,u_{j_k}])$ and so $\delta_i^\rho(t)\!\downarrow = \delta_i^\rho(t\!\downarrow)$.

Similarly, we have that $t' = \mathsf{f}(M_1\Phi'_+\!\downarrow,\dots,M_n\Phi'_+\!\downarrow)$ implies that $\delta_i^\rho(t')\!\downarrow = \delta_i^\rho(t'\!\downarrow)$. Since we proved that $\delta_i^\rho(t)\!\downarrow = \mathsf{f}(M_1^i,\dots,M_n^i)\delta^\rho(\Phi_+)\!\downarrow$ and $t\!\downarrow = M\Phi_+\!\downarrow$, we can deduce that $\delta_i^\rho(M\Phi_+\!\downarrow) = \mathsf{f}(M_1^i,\dots,M_n^i)\delta^\rho(\Phi_+)\!\downarrow$. Similarly, we have that $\mathsf{f}(M_1^i,\dots,M_n^i)\delta^\rho(\Phi'_+)\!\downarrow = \delta_i^\rho(M\Phi'_+\!\downarrow)$. Hence, the result holds with $M_i = \mathsf{f}(M_1^i,\dots,M_n^i)$.

Case $\mathsf{f} \in \{\mathsf{senc},\mathsf{aenc},\mathsf{sign},\langle\rangle\}$*:* By definition of $\mathcal{F}_0$, we know that $M\Phi_+\!\downarrow = \mathsf{f}(M_1\Phi_+\!\downarrow, M_2\Phi_+\!\downarrow)$ and $M\Phi'_+\!\downarrow = \mathsf{f}(M_1\Phi'_+\!\downarrow, M_2\Phi'_+\!\downarrow)$. Thanks to Lemma 5.6, we can deduce that for all $i \in \{a,b\}$, $\mathsf{root}(\delta_i^\rho(M_1\Phi_+\!\downarrow)) = \mathsf{root}(M_1\Phi_+\!\downarrow)$ and $\mathsf{root}(\delta_i^\rho(M_1\Phi'_+\!\downarrow)) = \mathsf{root}(M_1\Phi'_+\!\downarrow)$. But $\mathcal{M}(M_1) < \mathcal{M}(M)$, thus by our inductive hypothesis, there exists M_1^i such that $\delta_i^\rho(M_1\Phi_+\!\downarrow) = M_1^i\delta^\rho(\Phi_+)\!\downarrow$ and $\delta_i^\rho(M_1\Phi'_+\!\downarrow) = M_1^i\delta^\rho(\Phi'_+)\!\downarrow$. Hence for all $j \in \{a,b\}$, $\mathsf{root}(M_1\Phi_+\!\downarrow) = \mathsf{tag}_j$ is equivalent to $\mathsf{root}(M_1^i\delta^\rho(\Phi_+)\!\downarrow) = \mathsf{tag}_j$, which is also equivalent to $\mathsf{tag}_j(\mathsf{untag}_j(M_1^i))\delta^\rho(\Phi_+)\!\downarrow = M_1^i\delta^\rho(\Phi_+)\!\downarrow$. But by hypothesis $\nu\mathcal{E}.\delta^\rho(\Phi_+) \sim \nu\mathcal{E}.\delta^\rho(\Phi'_+)$, thus $\mathsf{tag}_j(\mathsf{untag}_j(M_1^i))\delta^\rho(\Phi_+)\!\downarrow = M_1^i\delta^\rho(\Phi_+)\!\downarrow$ is equivalent to $\mathsf{tag}_j(\mathsf{untag}_j(M_1^i))\delta^\rho(\Phi'_+)\!\downarrow = M_1^i\delta^\rho(\Phi'_+)\!\downarrow$, which is equivalent to $\mathsf{root}(M_1^i\delta^\rho(\Phi'_+)\!\downarrow) = \mathsf{tag}_j$. Hence, we deduce that for all $j \in \{a,b\}$, $\mathsf{root}(M_1\Phi_+\!\downarrow) = \mathsf{tag}_j$ is equivalent to $\mathsf{root}(M_1\Phi'_+\!\downarrow) = \mathsf{tag}_j$.

This equivalence and the definition of δ_a^ρ and δ_b^ρ allow us to deduce that for all $i \in \{a,b\}$, there exists $j \in \{a,b\}$ such that $\delta_i^\rho(M\Phi_+\!\downarrow) = \mathsf{f}(\delta_j^\rho(M_1\Phi_+\!\downarrow), \delta_j^\rho(M_2\Phi_+\!\downarrow))$ and $\delta_i^\rho(M\Phi'_+\!\downarrow) = \mathsf{f}(\delta_j^\rho(M_1\Phi'_+\!\downarrow), \delta_j^\rho(M_2\Phi'_+\!\downarrow))$. By our inductive hypothesis on M_1 and M_2, we deduce that there exists M_1^j and M_2^j such that $\delta_j^\rho(M_k\Phi_+\!\downarrow) = M_k^j\delta^\rho(\Phi_+)\!\downarrow$ and $\delta_j^\rho(M_k\Phi'_+\!\downarrow) = M_k^j\delta^\rho(\Phi'_+)\!\downarrow$, for $k \in \{1,2\}$. Hence, we have $\delta_i^\rho(M\Phi_+\!\downarrow) = \mathsf{f}(M_1^j, M_2^j)\delta^\rho(\Phi_+)\!\downarrow$ and $\delta_i^\rho(M\Phi'_+\!\downarrow) = \mathsf{f}(M_1^j, M_2^j)\delta^\rho(\Phi'_+)\!\downarrow$. So the result holds.

Case $\mathsf{f} \in \{\mathsf{h},\mathsf{pk},\mathsf{vk}\}$*:* This case is analogous to the previous one and can be handled similarly.

Case $\mathsf{f} = \mathsf{sdec}$*:* We have that $M = \mathsf{f}(M_1, M_2)$. Thus, we can apply our inductive hypothesis on M_1 and M_2, which means that for all $i \in \{a,b\}$, for all $k \in \{1,2\}$, there exists $M_k^i\delta^\rho(\Phi_+)\!\downarrow = \delta_i^\rho(M_k\Phi_+\!\downarrow)$ and $M_k^i\delta^\rho(\Phi'_+)\!\downarrow = \delta_i^\rho(M_k\Phi'_+\!\downarrow)$. Let's first focus on $M\Phi_+\!\downarrow$. We need to distinguish several cases:

(a) If the root occurence of sdec cannot be reduced, then $M\Phi_+\!\downarrow = \mathsf{sdec}(M_1\Phi_+\!\downarrow, M_2\Phi_+\!\downarrow)$. Thus for all $i \in \{a,b\}$, $\delta_i^\rho(M\Phi_+\!\downarrow) = \mathsf{sdec}(\delta_i^\rho(M_1\Phi_+\!\downarrow), \delta_i^\rho(M_2\Phi_+\!\downarrow)) = \mathsf{sdec}(M_1^i\delta^\rho(\Phi_+)\!\downarrow, M_2^i\delta^\rho(\Phi_+)\!\downarrow)$. Thanks to Lemma 5.6, we know that $\delta_i^\rho(M\Phi_+\!\downarrow)$ is in normal form which means that $\delta_i^\rho(M\Phi_+\!\downarrow) = \mathsf{f}(M_1^i, M_2^i)\delta^\rho(\Phi_+)\!\downarrow$

(b) If the root occurence of sdec can be reduced and $\mathsf{root}(M\Phi_+\!\downarrow) = \mathsf{tag}_j$, for some $j \in \{a,b\}$ then there exists u_1, u_2 such that $M_1\Phi_+\!\downarrow = \mathsf{senc}(\mathsf{tag}_j(u_1), u_2)$ and $M_2\Phi_+\!\downarrow = u_2$. Hence we have that $\mathsf{sdec}(\delta_j^\rho(M_1\Phi_+\!\downarrow), \delta_j^\rho(M_2\Phi_+\!\downarrow))\!\downarrow = \delta_j^\rho(\mathsf{tag}_j(u_1)) = \delta_j^\rho(M\Phi_+\!\downarrow)$. On the other hand, we also have $\mathsf{sdec}(\delta_j^\rho(M_1\Phi_+\!\downarrow), \delta_j^\rho(M_2\Phi_+\!\downarrow))\!\downarrow = \mathsf{sdec}(M_1^j\delta^\rho(\Phi_+)\!\downarrow, M_2^j\delta^\rho(\Phi_+)\!\downarrow)\!\downarrow$. Hence, we have that $\delta_j^\rho(M\Phi_+\!\downarrow) = \mathsf{sdec}(M_1^j, M_2^j)\delta^\rho(\Phi_+)\!\downarrow$.

(c) Else, the root occurence of sdec can be reduced and for all $j \in \{a,b\}$, $\mathsf{root}(M\Phi_+\!\downarrow) \neq \mathsf{tag}_j$. In such a case, there exist u_1, u_2 such that $M_1\Phi_+\!\downarrow = \mathsf{senc}(u_1, u_2)$, $M_2\Phi_+\!\downarrow = u_2$. Moreover, for all $i \in \{a,b\}$, $\delta_i^\rho(M_1\Phi_+\!\downarrow) = \mathsf{senc}(\delta_i^\rho(u_1), \delta_i^\rho(u_2))$ and $\delta_i^\rho(M_2\Phi_+\!\downarrow) = \delta_i^\rho(u_2)$. Hence, we have that for all $i \in \{a,b\}$, $\mathsf{sdec}(\delta_i^\rho(M_1\Phi_+\!\downarrow), \delta_i^\rho(M_2\Phi_+\!\downarrow))\!\downarrow = \delta_i^\rho(M\Phi_+\!\downarrow)$. At last, since $\mathsf{sdec}(\delta_i^\rho(M_1\Phi_+\!\downarrow), \delta_i^\rho(M_2\Phi_+\!\downarrow))\!\downarrow = \mathsf{sdec}(M_1^i, M_2^i)\delta^\rho(\Phi_+)\!\downarrow$, we have that $\mathsf{sdec}(M_1^i, M_2^i)\delta^\rho(\Phi_+)\!\downarrow = \delta_i^\rho(M\Phi_+\!\downarrow)$.

We could do the same case analysis for $M\Phi'_+\!\downarrow$ and we would obtain similar results. Since in two cases (a) and (c), the result holds for all $i \in \{a,b\}$, it only remains to show that for all $j \in \{a,b\}$, the root occurence of sdec is reduced in $M\Phi_+\!\downarrow$ and $\mathsf{root}(M\Phi_+\!\downarrow) = \mathsf{tag}_j$, if and only if, the root occurence of sdec is reduced in $M\Phi'_+\!\downarrow$ and $\mathsf{root}(M\Phi'_+\!\downarrow) = \mathsf{tag}_j$.

We already showed that the root occurence of sdec is reduced in $M\Phi_+\!\downarrow$ and $\mathsf{root}(M\Phi_+\!\downarrow) = \mathsf{tag}_j$ imply that $\delta_j^\rho(M\Phi_+\!\downarrow) = \mathsf{sdec}(M_1^j, M_2^j)\delta^\rho(\Phi_+)\!\downarrow$. Thus we have $\mathsf{sdec}(M_1^j, M_2^j)\delta^\rho(\Phi_+)\!\downarrow = \mathsf{tag}_j(\mathsf{untag}_j(\mathsf{sdec}(M_1^j, M_2^j)))\delta^\rho(\Phi_+)\!\downarrow$. But by hypothesis, $\nu\mathcal{E}.\delta^\rho(\Phi_+) \sim \nu\mathcal{E}.\delta^\rho(\Phi'_+)$. Hence we have $\mathsf{tag}_j(\mathsf{untag}_j(\mathsf{sdec}(M_1^j, M_2^j)))\delta^\rho(\Phi'_+)\!\downarrow = \mathsf{sdec}(M_1^j, M_2^j)\delta^\rho(\Phi'_+)\!\downarrow$. Thus there exists v_1, v_2 such that $M_1^j\delta^\rho(\Phi'_+)\!\downarrow = \mathsf{senc}(\mathsf{tag}_j(v_1), v_2)$ and $M_2^j\delta^\rho(\Phi'_+)\!\downarrow = v_2$, which means that $\delta_j^\rho(M_1\Phi'_+\!\downarrow) = \mathsf{senc}(\mathsf{tag}_j(v_1), v_2)$ and $\delta_j^\rho(M_2\Phi'_+\!\downarrow) = v_2$. Thanks to Lemmas 5.4 and 5.6, we deduce that there

exists v'_1 and v'_2 such that $M_1\Phi'_+{\downarrow} = \mathsf{senc}(\mathsf{tag}_j(v'_1), v'_2)$, $M_2\Phi'_+{\downarrow} = v'_2$, $v_2 = \delta^\rho_j(v'_2)$ and $v_1 = \delta^\rho_j(v'_1)$. We can conclude that the root occurence of sdec is reduced in $M\Phi'_+{\downarrow}$ and $\mathsf{root}(M\Phi'_+{\downarrow}) = \mathsf{tag}_j$. The other implication is symmetrical to this one. Hence our result holds.

Case $\mathsf{f} \in \{\mathsf{adec}, \mathsf{check}\}$*:* This case is similar to the case $\mathsf{f} = \mathsf{sdec}$.

Case $\mathsf{f} \in \mathsf{proj}_k$, $k \in \{1, 2\}$*:* In such a case, we have $M = \mathsf{f}(M_1)$. Thus we can apply our inductive hypothesis on M_1 which means that for all $i \in \{a, b\}$, there exists $M_1^i\delta^\rho(\Phi_+){\downarrow} = \delta^\rho_i(M_1\Phi_+{\downarrow})$ and $M_1^i\delta^\rho(\Phi'_+){\downarrow} = \delta^\rho_i(M_1\Phi'_+{\downarrow})$. Let's first focus on $M\Phi_+{\downarrow}$. We need to distinguish two cases:

(a) If the root occurence of f cannot be reduced, then $M\Phi_+{\downarrow} = \mathsf{f}(M_1\Phi_+{\downarrow})$. Thus for all $i \in \{a, b\}$, $\delta^\rho_i(M\Phi_+{\downarrow}) = \mathsf{f}(\delta^\rho_i(M_1\Phi_+{\downarrow})) = \mathsf{f}(M_1^i\delta^\rho(\Phi_+){\downarrow})$. Thanks to Lemma 5.6, we know that $\delta^\rho_i(M\Phi_+{\downarrow})$ is in normal form which means that $\delta^\rho_i(M\Phi_+{\downarrow}) = \mathsf{sdec}(M_1^i)\delta^\rho(\Phi_+){\downarrow}$.

(b) Else, the root occurence of f can be reduced. In such a case there exists u_1, u_2 such that $M_1\Phi_+{\downarrow} = \langle u_1, u_2\rangle$. Hence for all $i \in \{a, b\}$, we have $\delta^\rho_i(M_1\Phi_+{\downarrow}) = \langle\delta^\rho_i(u_1), \delta^\rho_i(u_2)\rangle$ and so $\mathsf{f}(\delta^\rho_i(M_1\Phi_+{\downarrow})){\downarrow} = \delta^\rho_i(u_k)$. Thus, we conclude that $\mathsf{f}(M_1^i)\delta^\rho(\Phi_+){\downarrow} = \delta^\rho_i(M\Phi_+{\downarrow})$

We could do the same case analysis for $M\Phi'_+{\downarrow}$ and we would obtain similar results. Since the result holds in both cases, we can conclude.

Second step: Thanks to the first step, we showed that there exists $c \in \{a, b\}$, and a term M_c such that $\mathit{fnames}(M_c) \cap \mathcal{E} = \emptyset$, $\mathit{fvars}(M_c) \subseteq \mathrm{dom}(\Phi_+)$, $\delta^\rho_c(M\Phi_+{\downarrow}) = M_c\delta^\rho(\Phi_+){\downarrow}$ and $\delta^\rho_c(M\Phi'_+{\downarrow}) = M_c\delta^\rho(\Phi'_+){\downarrow}$. Let $d \in \{a, b\}$ such that $d \neq c$.

By Lemma B.7, we know that there exists two contexts C, C' (possibly holes) built on $\{\langle\ \rangle\}$, and $u_1, \ldots, u_m, v_1, \ldots, v_n$ such that $M\Phi_+{\downarrow} = C[u_1, \ldots, u_m]$ and $M\Phi'_+{\downarrow} = C'[v_1, \ldots, v_n]$. Hence, we have that $\delta^\rho_c(M\Phi_+{\downarrow}) = C[\delta^\rho_c(u_1), \ldots, \delta^\rho_c(u_m)]$ and $\delta^\rho_c(M\Phi'_+{\downarrow}) = C'[\delta^\rho_c(v_1), \ldots, \delta^\rho_c(v_n)]$. Our hypothesis implies $M_c\delta^\rho(\Phi_+){\downarrow} = C[\delta^\rho_c(u_1), \ldots, \delta^\rho_c(u_m)]$ and $M_c\delta^\rho(\Phi'_+){\downarrow} = C'[\delta^\rho_c(v_1), \ldots, \delta^\rho_c(v_n)]$. But $\nu\mathcal{E}.\delta^\rho(\Phi_+) \sim \nu\mathcal{E}.\delta^\rho(\Phi'_+)$. Therefore, since C and C' are both built upon $\{\langle\ \rangle\}$, we deduce that $C = C'$ and $n = m$. Note that it also implies that there exists $D_1, \ldots, D_n$ contexts built on $\{\mathsf{proj}_1, \mathsf{proj}_2\}$ such that for all $k \in \{1, \ldots, n\}$, $D_k(M_c)\delta^\rho(\Phi_+){\downarrow} = \delta^\rho_c(u_k)$ and $D_k(M_c)\delta^\rho(\Phi'_+){\downarrow} = \delta^\rho_c(v_k)$.

Lemma B.7 also tells us that for all $k \in \{1, \ldots, n\}$,

— either $u_k \in \mathsf{Flawed}(M\Phi_+{\downarrow})$;

— or $u_k \in \mathit{Fct}_{\mathcal{F}_0}(M\Phi_+{\downarrow})$ and $\delta^\rho_a(u_k) = \delta^\rho_b(u_k)$,

— or $u_k = \mathsf{f}(n)$ for some $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ and $n \in \mathcal{N}$,

— or $u_k \in \mathcal{N}$.

Thus, we build M_d such that $M_d = C[N_1, \ldots, N_n]$ where, for all $k \in \{1, \ldots, n\}$,

(a) if $\delta^\rho_a(u_k) = \delta^\rho_b(u_k)$ then $N_k = D_k(M_c)$

(b) else if $u_k \in \mathsf{Flawed}(M\Phi_+{\downarrow})$, then by Lemma B.6, there exists f and $M_1, \ldots, M_\ell$ such that for all $i \in \{1 \ldots, \ell\}$, $\mathcal{M}(M_i) < \mathcal{M}(M)$ and $u_k = \mathsf{f}(M_1\Phi_+{\downarrow}, \ldots, M_\ell\Phi_+{\downarrow})$. But by our inductive hypothesis on $M_1, \ldots, M_\ell$, we have that there exists $M_1^d, \ldots, M_\ell^d$ such that for all $i \in \{1, \ldots, \ell\}$, $M_i^d\delta^\rho(\Phi_+){\downarrow} = \delta^\rho_d(M_i\Phi_+{\downarrow})$ and $M_i^d\delta^\rho(\Phi'_+){\downarrow} = \delta^\rho_d(M_i\Phi'_+{\downarrow})$. Therefore $\mathsf{f}(M_1^d, \ldots, M_\ell^d)\delta^\rho(\Phi_+){\downarrow} = \delta^\rho_d(\mathsf{f}(M_1\Phi_+{\downarrow}, \ldots, M_\ell\Phi_+{\downarrow}))$ and $\mathsf{f}(M_1^d, \ldots, M_\ell^d)\delta^\rho(\Phi'_+){\downarrow} = \delta^\rho_d(\mathsf{f}(M_1\Phi'_+{\downarrow}, \ldots, M_\ell\Phi'_+{\downarrow}))$.

We define $N_k = \mathsf{f}(M_1^d, \ldots, M_\ell^d)$. We know that $N_k\delta^\rho(\Phi_+){\downarrow} = \delta^\rho_d(u_k)$, thus it remains to show that $N_k\delta^\rho(\Phi'_+){\downarrow} = \delta^\rho_d(v_k)$. Our inductive hypothesis on $M_1, \ldots, M_\ell$ allows us to show that there exists $M_1^c, \ldots, M_\ell^c$ such that $\mathsf{f}(M_1^c, \ldots, M_\ell^c)\delta^\rho(\Phi_+){\downarrow} = \delta^\rho_c(u_k)$. But $\delta^\rho_c(u_k) = D_k(M_c)\delta^\rho(\Phi_+){\downarrow}$. Since $\delta^\rho(\Phi_+) \sim \delta^\rho(\Phi'_+)$, we deduce that $D_k(M_c)\delta^\rho(\Phi'_+){\downarrow} = \mathsf{f}(M_1^c, \ldots, M_\ell^c)\delta^\rho(\Phi'_+){\downarrow}$. Hence we have that $\delta^\rho_c(v_k) = \delta^\rho_c(\mathsf{f}(M_1\Phi'_+{\downarrow}, \ldots, M_\ell\Phi'_+{\downarrow}))$. By applying Lemma 5.4, we obtain that $\mathsf{f}(M_1\Phi'_+{\downarrow}, \ldots, M_\ell\Phi'_+{\downarrow}) = v_k$ and so $N_k\delta^\rho(\Phi'_+){\downarrow} = \delta^\rho_d(v_k)$

(c) else if $u_k \in \mathsf{f}(n)$ *for some* $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ *and* $n \in \mathcal{N}$, then $D_k(M_c)\delta^\rho(\Phi_+){\downarrow} = \mathsf{f}(\delta^\rho_c(n))$, i.e. $\mathsf{f}(\delta^\rho_c(n))$ is deducible. If $n \notin \mathrm{img}(\rho)$ then $\delta^\rho_a(u_k) = \delta^\rho_b(u_k)$ and so it is the same as case (a). Else by hypothesis on $\delta^\rho(\Phi_+)$, we know that $n \in \mathrm{img}(\rho)$ implies that $\mathsf{f}(\delta^\rho_c(n)) \in \mathrm{img}(\delta^\rho(\Phi_a \uplus \Phi_b))$. By construction of Φ_a and Φ_b, we know that there exist $(ax^d_\ell \rhd \mathsf{f}(\delta^\rho_d(n)), ax^c_\ell \rhd \mathsf{f}(\delta^\rho_c(n))) \in \delta^\rho(\Phi_a \uplus \Phi_b)$ with $\mathsf{col}(ax^d_\ell) = d$ and $\mathsf{col}(ax^c_\ell) = c$ and some $\ell \in \mathbb{N}$.

We define $N_k = ax^d_\ell$. It remains to show that $ax^d_\ell\delta^\rho(\Phi'_+)\downarrow = \delta^\rho_d(v_k)$. But we have $ax^c_\ell\delta^\rho(\Phi_+)\downarrow = D_k(M_c)\delta^\rho(\Phi_+)\downarrow$. Thanks to our hypothesis $\delta^\rho(\Phi_+) \sim \delta^\rho(\Phi'_+)$, we deduce that $ax^c_\ell\delta^\rho(\Phi'_+)\downarrow = D_k(M_c)\delta^\rho(\Phi'_+)\downarrow$. By definition of Φ'_+, we have that $ax^c_\ell\delta^\rho(\Phi'_+)\downarrow = \mathsf{f}(\delta^\rho_c(n))$ which means that $\delta^\rho_c(v_k) = D_k(M_c)\delta^\rho(\Phi'_+)\downarrow = \delta^\rho_c(\mathsf{f}(n))$. By applying Lemma 5.4, we obtain that $v_k = \mathsf{f}(n)$. At last, by definition of Φ'_+, we have that $ax^d_\ell\delta^\rho(\Phi'_+)\downarrow = \delta^\rho_d(\mathsf{f}(n)) = \delta^\rho_d(v_k)$.

(d) Else $u_k \in \mathcal{N}$. But by hypothesis on $\delta^\rho(\Phi_+)$, we know that $n \in \mathrm{img}(\rho)$ implies $\delta^\rho_c(n)$ is not deducible from $\delta^\rho(\Phi_+)$. Thus $u_k \notin \mathrm{img}(\rho)$ and so $\delta^\rho_a(u_k) = \delta^\rho_b(u_k)$, which leads to $N_k = D_k(M_c)$, i.e. same as case (a).

We showed that for such $M_d = C[N_1, \ldots, N_n]$, we have that $M_d\delta^\rho(\Phi_+)\downarrow = \delta^\rho_d(C[u_1, \ldots, u_n])$ and $M_d\delta^\rho(\Phi'_+)\downarrow = \delta^\rho_d(C[v_1, \ldots, v_n])$, which leads to $M_d\delta^\rho(\Phi_+)\downarrow = \delta^\rho_d(M\Phi_+\downarrow)$ and $M_d\delta^\rho(\Phi'_+)\downarrow = \delta^\rho_d(M\Phi'_+\downarrow)$. □

For the next two lemmas, *i.e.* soundness and completeness, we consider $\mathcal{E}_0$ a set of private names and P_a, P_b two processes without name restriction or replication, and coloured respectively by a and b. At last consider ρ a renaming such that:

— $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_0$ and $\mathrm{dom}(\rho)$ do not appear in P_a, P_b, Φ_a or Φ_b.

— for all $i \in \{a, b\}$, there exists P'_i built on $\mathcal{F}_i \cup \mathcal{F}_0$ such that $P_i = [P'_i]_i$.

— for all $i \in \{a, b\}$, $\mathit{fnames}(P_i) \cap \mathcal{C}h \subseteq \mathrm{dom}(\rho_{\mathcal{C}h_i})$ and $\mathrm{img}(\rho_{\mathcal{C}h_i})$ does not appear in P_a or P_b.

We will denote $D = (\mathcal{E}_0; \{P_a\rho_{\mathcal{C}h_a}, P_b\rho^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_a \uplus \Phi_b\rho^{-1})$ and $S = (\mathcal{E}_0; \{P_a, P_b\}; \Phi_a \uplus \Phi_b)$. We assume that P_a and P_b do not share any private channel name, *i.e.* $\mathit{fnames}(P_a) \cap \mathit{fnames}(P_b) \cap \mathcal{C}h \cap \mathcal{E}_0 = \emptyset$, and do not use variable of channel type.

Lemma 5.14 (Soundness)**.** *If for all* $(\mathsf{tr}, \Phi) \in \mathit{trace}(D)$*, for all* $u \in \{\mathsf{vk}(k), \mathsf{pk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$*,* $\nu\mathcal{E}_0.\Phi \vdash u$ *implies that* $u \in \mathrm{img}(\Phi_a \uplus \Phi_b\rho^{-1})$*, then for all* $S \xrightarrow{w} (\mathcal{E}_0; \mathcal{P}_S; \Phi_S)$ *such that* w *does not contain any* τ *action that corresponds to an internal communication between two processes of different colours, we have that* Φ_S *is well-tagged and there exists* $D \xrightarrow{w'} (\mathcal{E}; \mathcal{P}_D; \Phi_D)$ *such that* $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$.

Furthermore, if $w = \ell_1.\ldots.\ell_n$ *then* $w' = \ell'_1.\ldots.\ell'_n$ *such that for all* $k \in \{1, \ldots, n\}$*,*

— *if* $\ell_k = \nu ax_m.out(c, ax_m)$ *is an output coming from a process coloured by* $i \in \{a, b\}$*, then* $\ell'_k = \nu ax_m.\mathsf{out}(c\rho_{\mathcal{C}h_i}, ax_m)$

— *if* $\ell_k = in(c, M)$ *is an input coming from a process coloured by* $i \in \{a, b\}$*, then* $\ell'_k = in(c\rho_{\mathcal{C}h_i}, M_i)$ *with* $M_i\Phi_D\downarrow = \delta^\rho_i(M\Phi_S\downarrow)$.

— *if* $\ell_k = out(c, d)$ *is an output coming from a process coloured by* $i \in \{a, b\}$ *with* d *a channel name, then* $\ell'_k = \mathsf{out}(c\rho_{\mathcal{C}h_i}, d\rho_{\mathcal{C}h_i})$

— *if* $\ell_k = \nu ch_m.out(c, ch_m)$ *is an output coming from a process coloured by* $i \in \{a, b\}$ *with* ch_m *a channel name, then* $\ell'_k = \nu ch_m.out(c\rho_{\mathcal{C}h_i}, ch_m)$

— *if* $\ell_k = \tau$*, then* $\ell'_k = \tau$.

Proof. We have that $S \xrightarrow{w} (\mathcal{E}_0; \mathcal{P}'_S; \Phi'_S)$. We will show by induction on $|w|$ the properties stated in the lemma but also that there exists $(\mathcal{P}_a, \alpha_a)$ and $(\mathcal{P}_b, \alpha_b)$ two original well-tagged multi-sets of processes such that $\mathsf{col}(\mathcal{P}_a) = a$, $\mathsf{col}(\mathcal{P}_b) = b$, $\mathcal{P}'_S = \mathcal{P}_a\alpha_a \uplus \mathcal{P}_b\alpha_b$ and $\mathcal{P}'_D = \mathcal{P}^a_D \uplus \mathcal{P}^b_D$ with for all $i \in \{a, b\}$, $\mathcal{P}^i_D\downarrow = \delta^\rho_i(\mathcal{P}_i)\rho_{\mathcal{C}h_i}\delta^\rho_i(\alpha_i\downarrow)\downarrow$.

Base case $|w| = 0$*:* In this case, we need to verify that S and D satisfy the wanted properties. By hypothesis on S and D, we have $S = (\mathcal{E}_0; \mathcal{P}_S; \Phi_S)$ and $D = (\mathcal{E}_0; \mathcal{P}_D; \Phi_D)$ such that $\mathcal{P}_S = \{[P'_a]_a, [P'_b]_b\}$, $\Phi_S = \Phi_a \uplus \Phi_b$, $\mathcal{P}_D = \{\delta^\rho_a([P'_a]_a)\rho_{\mathcal{C}h_a}, \delta^\rho_b([P'_b]_b)\rho_{\mathcal{C}h_b}\}$ and $\Phi_D = \delta^\rho(\Phi_S)$.

Thus by definition of an original well-tagged multi-set of processes, we define $\mathcal{P}_a = \{[P'_a]_a\}$, $\alpha_a = id$, $\mathcal{P}_b = \{[P'_b]_b\}$, $\alpha_b = id$. Hence, we have $\delta^\rho_i(\alpha_i\downarrow) = id$ and so $\mathcal{P}^i_D\downarrow = \delta^\rho_i(\mathcal{P}_i)\rho_{\mathcal{C}h_i}\downarrow$, for all $i \in \{a, b\}$.

Furthermore, we know by hypothesis that for all $(w \rhd u) \in \Phi_A \cup \Phi_B$, $u = \mathsf{f}(n)$ for some $\mathsf{f} \in \{\mathsf{pk}, \mathsf{vk}\}$ and $n \in \mathcal{N}$. Hence we have that $[u]_i = u$ and $\mathsf{test}_i(u) = \mathsf{true}$ for all $i \in \{a, b\}$, which implies that Φ_S is well-tagged and in normal form.

At last, $\delta^\rho_a(\Phi_A) = \Phi_A$ and $\delta^\rho_b(\Phi_B) = \Phi_B\rho_0^{-1}$, hence we have that $\delta^\rho(\Phi_S\downarrow) = \Phi_D\downarrow$.

Inductive case $|w| > 0$*:* In this case, we have that $w = w_1.\ell$ and so $S \xrightarrow{w_1} (\mathcal{E}_0; \mathcal{P}''_S; \Phi''_S) \xrightarrow{\ell} (\mathcal{E}_0; \mathcal{P}'_S; \Phi'_S)$ with no internal communication between two processes of different colors. Hence, by inductive hypothesis on w_1, we have that there exists $D \xrightarrow{w'_1} (\mathcal{E}_0; \mathcal{P}''_D; \Phi''_D)$, $(\mathcal{P}'_a, \alpha'_a)$ and $(\mathcal{P}'_b, \alpha'_b)$ two original well-tagged multi-sets of processes such that:

— $\Phi''_D\downarrow = \delta^\rho(\Phi''_S\downarrow)$

— Φ''_S is well-tagged

— $\mathcal{P}''_S = \mathcal{P}'_a\alpha'_a \uplus \mathcal{P}'_b\alpha'_b$ and $\mathcal{P}''_D = {\mathcal{P}^a_D}' \uplus {\mathcal{P}^b_D}'$ with for all $i \in \{a, b\}$, ${\mathcal{P}^i_D}'\downarrow = \delta^\rho_i(\mathcal{P}'_i)\rho_{\mathcal{C}h_i}\delta^\rho_i(\alpha'_i\downarrow)\downarrow$.

— for all $k \in \{1, \ldots, |w| - 1\}$, ℓ_k satisfies the desired properties.

We proceed by case analysis on the rule applied for the transition $(\mathcal{E}_0; \mathcal{P}''_S; \Phi''_S) \xrightarrow{\ell} (\mathcal{E}_0; \mathcal{P}'_S; \Phi'_S)$. Since by hypothesis, we know that there is no internal communication between two processes of different colors, and $\mathcal{P}''_S = \mathcal{P}'_a\alpha'_a \uplus \mathcal{P}'_b\alpha'_b$ we can assume that a rule is applied on $\mathcal{P}'_i\alpha'_i$, with $i \in \{a, b\}$. Let $j \in \{a, b\}$ such that $i \neq j$

Case of the rule THEN*:* In this case, by definition of $(\mathcal{P}'_i, \alpha'_i)$, there exists ϕ formula and Q_1, Q_2 processes and $\mathcal{Q}'_i \subseteq \mathcal{P}'_i$ such that $\mathcal{P}''_S = \{\texttt{if } \phi \texttt{ then } [Q_1]_i\alpha'_i \texttt{ else } [Q_2]_i\alpha'_i\} \uplus \mathcal{Q}'_i\alpha'_i \uplus \mathcal{P}'_j\alpha'_j$, $\mathcal{P}'_S = \{[Q_1]_i\alpha'_i\} \uplus \mathcal{Q}'_i\alpha'_i \uplus \mathcal{P}'_j\alpha'_j$, $\Phi'_S = \Phi''_S$ and ϕ a conjunction of equations $(u = v)$. By definition, we have that $([Q_1]_i, \alpha'_i)$ and $([Q_2]_i, \alpha'_i)$ are both original well-tagged processes.

Furthermore, we have that for all equation $(u = v) \in \phi$, $u =_{\mathsf{E}} v$; and either (a) there exists u such that ϕ is the formula $\mathsf{test}_i([u]_i)\alpha'_i$, or (b) there exists u_1, u_2 such that ϕ is the formula $[u_1]_i\alpha'_i = [u_2]_i\alpha'_i$ and $\alpha'_i \vDash \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i)$.

But we also have ${\mathcal{P}^i_D}'\downarrow = \delta^\rho_i(\mathcal{P}'_i)\rho_{\mathcal{C}h_i}\delta^\rho_i(\alpha'_i\downarrow)\downarrow$, which means that there exists ϕ' formula and Q'_1, Q'_2 processes and $\mathcal{Q}^i_D \subseteq {p^i_D}'$ such that $\mathcal{P}''_D = \{\texttt{if } \phi' \texttt{ then } Q'_1 \texttt{ else } Q'_2)\} \uplus \mathcal{Q}^i_D \uplus {\mathcal{P}^j_D}'$ with $Q'_1\downarrow = \delta^\rho_i([Q_1]_i)\rho_{\mathcal{C}h_i}\delta^\rho_i(\alpha'_i\downarrow)\downarrow$, $Q'_2\downarrow = \delta^\rho_i([Q_2]_i)\rho_{\mathcal{C}h_i}\delta^\rho_i(\alpha'_i\downarrow)\downarrow$. Furthermore, in case (a) $\phi'\downarrow$ is the formula $\mathsf{test}_i(\delta^\rho_i([u]_i))\delta^\rho_i(\alpha'_i\downarrow)\downarrow$; and in case (b) ϕ' is the formula $(u' = v')$ where $u'\downarrow = \delta^\rho_i([u_1]_i)\delta^\rho_i(\alpha'_i\downarrow)\downarrow$ and $v'\downarrow = \delta^\rho_i([u_2]_i)\delta^\rho_i(\alpha'_i\downarrow)\downarrow$.

In case (a), For all equation $(u' = v') \in \phi$, $u' =_{\mathsf{E}} v'$ which is equivalent to $u'\downarrow = v'\downarrow$. Thus it is equivalent to $(\alpha'_i\downarrow) \vDash \mathsf{test}_i([u]_i)$. But by Lemma B.3, we know that this is equivalent to $\delta^\rho_i(\alpha'_i\downarrow) \vDash \mathsf{test}_i(\delta^\rho_i([u]_i))$. Thus we have that for all equation $(u' = v')$ of $\mathsf{test}_i(\delta^\rho_i([u]_i))$, $u'\delta^\rho_i(\alpha'_i\downarrow)\downarrow = v'\delta^\rho_i(\alpha'_i\downarrow)\downarrow$. Since $\phi'\downarrow$ is the formula $\mathsf{test}_i(\delta^\rho_i([u]_i))\delta^\rho_i(\alpha'_i\downarrow)\downarrow$, we can conclude that for all equation $(u' = v')$ of ϕ', $u'\downarrow = v'\downarrow$ and so $u' =_{\mathsf{E}} v'$.

In case (b), we know that $\alpha\downarrow \vDash \mathsf{test}_i([u_1]_i) \wedge \mathsf{test}_i([u_2]_i)$ and $[u_1]_i\alpha'_i\downarrow = [u_2]_i\alpha\downarrow$. Thus by Corollary B.1, we can deduce that $\delta^\rho_i([u_1]_i)\delta^\rho_i(\alpha'_i\downarrow)\downarrow = \delta^\rho_i([u_2]_i)\delta^\rho_i(\alpha'_i\downarrow)\downarrow$ which means that $u'\downarrow = v'\downarrow$ and so $u' =_{\mathsf{E}} v'$.

ϕ' being satisfied in both cases allows us to deduce that $(\mathcal{E}_0; \mathcal{P}''_D; \Phi''_D) \xrightarrow{\ell} (\mathcal{E}_0; \{Q'_1\} \uplus \mathcal{Q}^i_D \uplus {\mathcal{P}^j_D}'; \Phi''_D)$.

Thus, we have $\Phi'_D = \Phi''_D$, but $\Phi'_S = \Phi''_S$, hence we have that Φ'_S is well-tagged and $\Phi'_D\downarrow = \delta^\rho(\Phi'_S\downarrow)$. Furthermore, let $\ell' = \ell = \tau$ and $\mathcal{P}_i = \{[Q_1]_i\} \uplus \mathcal{Q}'_i$, $\alpha_i = \alpha'_i$, $\mathcal{P}_j = \mathcal{P}'_j$, $\alpha_j = \alpha'_j$. By definition of $[Q_1]_i$, we have that $(\mathcal{P}_i, \alpha_i)$ and $(\mathcal{P}_j, \alpha_j)$ are both original well-tagged multisets of processes.

We already showed that $\mathcal{P}_S = \mathcal{P}_i\alpha_i \uplus \mathcal{P}_j\alpha_i$ and thanks to $Q'_1\downarrow = \delta^\rho_i([Q_1]_i)\rho_{\mathcal{C}h_i}\delta^\rho_i(\alpha'_i\downarrow)\downarrow$, we can deduce that $\mathcal{P}_D\downarrow = \delta^\rho_i(\mathcal{P}_i)\rho_{\mathcal{C}h_i}\delta^\rho_i(\alpha_i\downarrow)\downarrow \uplus \delta^\rho_j(\mathcal{P}_j)\rho_{\mathcal{C}h_j}\delta^\rho_j(\alpha_j\downarrow)\downarrow$. Hence the result holds.

Case of the rule ELSE*:* This case similar to the rule THEN.

Case of the rule COMM*:* In this case, by definition of $(\mathcal{P}'_i, \alpha'_i)$, there exists u, x, p terms, Q_1, Q_2 processes and $\mathcal{Q}'_i \subseteq \mathcal{P}'_i$ such that $\mathcal{P}''_S = \{\mathsf{out}(p, [u]_i\alpha'_i).[Q_1]_i\alpha_i; \mathsf{in}(p, x).[Q_2]_i\alpha'_i\} \uplus \mathcal{Q}'_i\alpha'_i \uplus \mathcal{P}'_j\alpha'_j$, $\mathcal{P}'_S = \{[Q_1]_i\alpha'_i; [Q_2]_i\alpha'_i\{x \mapsto [u]_i\alpha_i\}\} \uplus \mathcal{Q}'_i\alpha'_i \uplus \mathcal{P}'_j\alpha'_j$, $\Phi'_S = \Phi''_S$ and $\alpha'_i \vDash \mathsf{test}_i([u]_i)$.

First of all, we trivially have that Φ'_S is well-tagged.

Furthermore, we have ${\mathcal{P}^i_D}'\downarrow = \delta^\rho_i(\mathcal{P}'_i)\rho_{\mathcal{C}h_i}\delta^\rho_i(\alpha'_i\downarrow)\downarrow$, which means that there exists p', u' terms and Q'_1, Q'_2, processes such that $\mathcal{P}''_D = \{\mathsf{out}(p', u').Q'_1; \mathsf{in}(p', x).Q'_2\} \uplus \mathcal{Q}^i_D \uplus {\mathcal{P}^j_D}'$ with $Q'_1\downarrow =$

$\delta_i^\rho([Q_1]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i'\downarrow)\downarrow$, $Q_2'\downarrow = \delta_i^\rho([Q_2]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i'\downarrow)\downarrow$, $p' = p'\downarrow = \delta_i^\rho(p)\rho_{\mathcal{C}h_i} = p\rho_{\mathcal{C}h_i}$ and $u'\downarrow = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha_i'\downarrow)\downarrow$. Hence for $\ell' = \ell = \tau$, we have that $(\mathcal{E}_0; \mathcal{P}_D''; \Phi_D'') \xrightarrow{\ell'} (\mathcal{E}_0; Q_1' \uplus Q_2'\{x \mapsto u'\} \uplus \mathcal{Q}_D^i \uplus \mathcal{P}_D^{j\,\prime}; \Phi_D'')$.

Let $\Phi_D' = \Phi_D''$. Since $\Phi_S' = \Phi_S''$, then by hypothesis, we have that $\Phi_D'\downarrow = \delta^\rho(\Phi_S'\downarrow)$. Let's denote $\alpha_i = \alpha_i'\{x \mapsto [u]_i\alpha_i\}$. We already know that $\alpha_i' \vDash \mathsf{test}_i([u]_i)$. Thus thanks to Lemma B.2 and $(\alpha_i'\downarrow) \vDash \mathsf{test}_i([u]_i)$, we deduce that $u'\downarrow = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha_i'\downarrow)\downarrow = \delta_i^\rho([u]_i(\alpha_i'\downarrow))\downarrow = \delta_i^\rho([u]_i\alpha_i'\downarrow)$. This implies that $\delta_i^\rho(\alpha_i\downarrow) = \delta_i^\rho(\alpha_i')\{x \mapsto u'\downarrow\}$. But $Q_2'\{x \mapsto u'\}\downarrow = (Q_2'\downarrow)\{x \mapsto (u'\downarrow)\}\downarrow$, hence $Q_2'\{x \mapsto u'\}\downarrow = \delta_i^\rho([Q_2]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i\downarrow)\downarrow$.

Moreover, since $x \notin \mathit{fvars}([Q_1]_i) \cup \mathit{fvars}(\mathcal{Q}_i')$, we can deduce that $[Q_1]_i\alpha_i' = [Q_1]_i\alpha_i$, $Q_2'\downarrow = \delta_i^\rho([Q_2]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i\downarrow)\downarrow$ and $\mathcal{Q}_i'\alpha_i' = \mathcal{Q}_i'\alpha_i$. Thus, we have that $(\{[Q_1]_i, [Q_2]_i\} \uplus \mathcal{Q}_i', \alpha_i)$ is an original well-tagged multi-set of processes. Hence the result holds.

Case of the rule IN*:* In this case, by definition of $(\mathcal{P}_i', \alpha_i')$, there exists x, p, M terms, Q_1 process, $\mathcal{Q}_i' \subseteq \mathcal{P}_i'$, $M\Phi_S'' = u$, $\mathit{fvars}(M) \subseteq \mathrm{dom}(\Phi_S'')$ and $\mathit{fnames}(M) \cap \mathcal{E}_0 = \emptyset$ such that $\mathcal{P}_S'' = \{\mathsf{in}(p, x).[Q_1]_i\alpha_i'\} \uplus \mathcal{Q}_i'\alpha_i' \uplus \mathcal{P}_j'\alpha_j'$, $\mathcal{P}_S' = \{[Q_1]_i\alpha_i'\{x \mapsto u\}\} \uplus \mathcal{Q}_i'\alpha_i' \uplus \mathcal{P}_j\alpha_j'$, $\Phi_S' = \Phi_S''$ and $\alpha_i' \vDash \mathsf{test}_i([u]_i)$. We trivially have that Φ_S' is well-tagged.

Furthermore, we have $\mathcal{P}_D^{i\,\prime}\downarrow = \delta_i^\rho(\mathcal{P}_i')\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i'\downarrow)\downarrow$, which means that there exists p' term and Q_1' process such that $\mathcal{P}_D'' = \{\mathsf{in}(p', x).Q_1'\} \uplus \mathcal{Q}_D^i \uplus \mathcal{P}_D^{j\,\prime}$ with $Q_1'\downarrow = \delta_i^\rho([Q_1]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i'\downarrow)\downarrow$ and $p' = p'\downarrow = \delta_i^\rho(p)\rho_{\mathcal{C}h_i} = p\rho_{\mathcal{C}h_i}$.

By hypothesis, we assumed that for all $v \in \{k, \mathsf{vk}(k), \mathsf{pk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$, $\nu\mathcal{E}_0.\Phi_D'' \vdash v$ implies that $v \in \mathrm{img}(\Phi_D)$, where $\Phi_D = \delta^\rho(\Phi_a \uplus \Phi_b)$. Thus, thanks to Lemma 5.13, we have that there exists a term M_i such that $\mathit{fvars}(M_i) \subseteq \mathrm{dom}(\Phi_D'')$, $\mathit{fnames}(M_i) \cap \mathcal{E}_0 = \emptyset$ and $M_i\Phi_D''\downarrow = \delta_i^\rho(M\Phi_S''\downarrow)$. Thus, with $\ell' = \mathsf{in}(p\rho_{\mathcal{C}h_i}, M_i)$, we can deduce that $(\mathcal{E}_0; \mathcal{P}_D''; \Phi_D'') \xrightarrow{\ell'} (\mathcal{E}_0; Q_1'\{x \mapsto u'\} \uplus \mathcal{Q}_D^i \uplus \mathcal{P}_D^{j\,\prime}; \Phi_D'')$ where $u' = M_i\Phi_D''$.

But $M\Phi_S''\downarrow = u\downarrow$ and $M_i\Phi_D''\downarrow = u'\downarrow$. Thus, we have that $\delta_i^\rho(u\downarrow) = u'\downarrow$. Since $Q_1'\downarrow = \delta_i^\rho([Q_1]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i'\downarrow)\downarrow$, we deduce that $Q_1'\{x \mapsto u'\}\downarrow = \delta_i^\rho([Q_1]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i'\downarrow)\{x \mapsto \delta_i^\rho(u\downarrow)\}\downarrow$.

Let $\alpha_i = \alpha_i'\{x \mapsto u\}$, we have that that $\delta_i^\rho(\alpha_i\downarrow) = \delta_i^\rho(\alpha_i'\downarrow)\{x \mapsto \delta_i^\rho(u\downarrow)\}$ and so $Q_1'\{x \mapsto u'\}\downarrow = \delta_i^\rho([Q_1]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i\downarrow)\downarrow$ and $[Q_1]_i\alpha_i'\{x \mapsto u\} = [Q_1]_i\alpha_i$. Moreover, since $x \notin \mathit{fvars}(\mathcal{Q}_i')$, we can deduce that , $Q_1'\downarrow = \delta_i^\rho([Q_1]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i\downarrow)\downarrow$ and $\mathcal{Q}_i'\alpha_i' = \mathcal{Q}_i'\alpha_i$. Thus, we have that $(\{[Q_1]_i\} \uplus \mathcal{Q}_i', \alpha_i)$ is an original well-tagged multi-set of processes. Hence the result holds.

Case of the rule OUT-T*:* In such a case, by definition of $(\mathcal{P}_i', \alpha_i')$, there exists u, p terms and Q_1 process, $\mathcal{Q}_i' \subseteq \mathcal{P}_i'$ such that $\mathcal{P}_S'' = \{\mathsf{out}(p, [u]_i\alpha_i').[Q_1]_i\alpha_i'\} \uplus \mathcal{Q}_i'\alpha_i' \uplus \mathcal{P}_j'\alpha_j'$, $\mathcal{P}_S' = \{[Q_1]_i\alpha_i'\} \uplus \mathcal{Q}_i'\alpha_i' \uplus \mathcal{P}_j'\alpha_j'$, $\Phi_S' = \Phi_S'' \uplus \{ax_n \rhd [u]_i\alpha_i'\}$ and $\alpha_i' \vDash \mathsf{test}_i([u]_i)$. Furthermore, we have that $\ell = \nu ax_n.\mathsf{out}(p, ax_n)$.

Moreover, we have $\mathcal{P}_D^{i\,\prime}\downarrow = \delta_i^\rho(\mathcal{P}_i')\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i'\downarrow)\downarrow$, which means that there exists p', u' terms and Q_1' process such that $\mathcal{P}_D'' = \{\mathsf{out}(p', u').Q_1'\} \uplus \mathcal{Q}_D^i \uplus \mathcal{P}_D^{j\,\prime}$ with $Q_1'\downarrow = \delta_i^\rho([Q_1]_i)\rho_{\mathcal{C}h_i}\delta_i^\rho(\alpha_i'\downarrow)\downarrow$, $p' = p'\downarrow = \delta_i^\rho(p)\rho_{\mathcal{C}h_i} = p\rho_{\mathcal{C}h_i}$ and $u'\downarrow = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha_i'\downarrow)\downarrow$. Hence for $\ell' = \nu w_n.\mathsf{out}(p\rho_{\mathcal{C}h_i}, w_n)$, we have that $(\mathcal{E}_0; \mathcal{P}_D''; \Phi_D'') \xrightarrow{\ell'} (\mathcal{E}_0; \{Q_1'\} \uplus \mathcal{Q}_D^i \uplus \mathcal{P}_D^{j\,\prime}; \Phi_D')$.

We first need to show that Φ_S' is well-tagged. Since Φ_S'' is well tag, we only need to focus on the new term $[u]_i\alpha_i'$. Let $x \in \mathrm{dom}(\alpha_i')$, we know that x was initially a variable from S. Thus, x was introduced by an input $\mathsf{in}(c, x)$ for some c and so there exists a transition ℓ_x in w_1 such that this transition reduces $\mathsf{in}(c, x)$. If $\ell_x = \mathsf{in}(c, M)$ (i.e. the rule IN), then we trivially have that the result holds with M; else $\ell_x = \tau$ (i.e. the rule COMM). But in the case $\ell_x = \tau$, we know that the output comes from a process colored by i and so there exists v such that $x\alpha_i' = [v]_i\alpha_i'$. Thus with a simple induction on the size of w_1, we can show that Φ_S' is well-tagged.

Since all variables in $[u]_i$ are colored by i, then thanks to Lemma B.2 and $(\alpha_i'\downarrow) \vDash \mathsf{test}_i([u]_i)$, $\delta_i^\rho([u]_i)\delta_i^\rho(\alpha_i'\downarrow)\downarrow = \delta_i^\rho([u]_i(\alpha_i'\downarrow))\downarrow = \delta_i^\rho([u]_i\alpha_i'\downarrow) = \delta_i^\rho(u\downarrow)$. But $u'\downarrow = \delta_i^\rho([u]_i)\delta_i^\rho(\alpha_i'\downarrow)\downarrow$, which means that $u'\downarrow = \delta_i^\rho(u\downarrow)$. Since $\mathsf{col}(w_n) = i$, we can conclude that $\Phi_D'\downarrow = \delta^\rho(\Phi_S'\downarrow)$. Hence the result holds.

At last, let $\alpha_i = \alpha_i'$, we have that $(\{[Q_1]_i\} \uplus \mathcal{Q}_i', \alpha_i)$ is an original well-tagged multi-set of processes. Hence our result holds.

Case of the rule Out-Ch*:* Obvious since $\mathrm{dom}(\rho)$ and $\mathrm{img}(\rho)$ only contain names of base type and so if $\ell = \mathsf{out}(c, d)$, since c and d are public then $\ell' = \mathsf{out}(c\rho_{\mathcal{C}h_i}, d\rho_{\mathcal{C}h_i})$

Case of the rule Open-Ch*:* Obvious since $\mathrm{dom}(\rho)$ and $\mathrm{img}(\rho)$ only contain names of base type and so if $\ell = \nu ch\mathsf{out}(c, ch)$, since c is public and $\mathrm{dom}(\rho_{\mathcal{C}h_i})$ is only composed of public channels, then $\ell' = \nu ch_m.\mathsf{out}(c\rho_{\mathcal{C}h_i}, ch_m)$
Case of the rule Par*:* Obvious □

Lemma 5.15 (Completeness)**.** *Let $D \xrightarrow{w} (\mathcal{E}_0; \mathcal{P}_D; \Phi_D)$. Let Φ_+ be a ground well-tagged frame such that $\Phi_+ = \Phi_a \uplus \Phi_b \uplus \Phi$ for some Φ, and Φ_+, Φ_D have the same colours. If the following properties are satisfied:*

- *$w = \ell_1 . \ldots . \ell_n$*
- *$\nu\mathcal{E}_0.\Phi_D \sim \nu\mathcal{E}_0.\delta^\rho(\Phi_+)$*
- *for all $u \in \{k, \mathsf{vk}(k), \mathsf{pk}(k) \mid k \in \mathrm{img}(\rho) \cup \mathrm{dom}(\rho)\}$, $\nu\mathcal{E}_0.\delta^\rho(\Phi_+) \vdash u$ or $\nu\mathcal{E}_0.\Phi_D \vdash u$ implies that $u \in \mathrm{img}(\Phi_a \uplus \Phi_b\rho^{-1})$*
- *for all $k \in \{1, \ldots, n\}$, if $\ell_k = \mathsf{in}(c, M_k)$ with $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a, b\}$ then there exists M_k^i such that $M_k\delta^\rho(\Phi_+\downarrow)\downarrow = \delta_i^\rho(M_k^i\Phi_+\downarrow)$.*

then there exists a label $w' = \ell'_1 . \ldots . \ell'_n$ and a well-tagged frame Φ_S such that $S \xrightarrow{w'} (\mathcal{E}; \mathcal{P}_S; \Phi_S)$, $\Phi_D\downarrow = \delta^\rho(\Phi_S\downarrow)$, and for all $k \in \{1, \ldots, n\}$,

- *if $\ell_k = \nu ax.out(c, ax)$ with $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a, b\}$ then $\ell'_k = \nu ax.out(c\rho_{\mathcal{C}h_i}^{-1}, ax)$*
- *if $\ell_k = in(c, M_k)$ with $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a, b\}$, then $\ell'_k = in(c\rho_{\mathcal{C}h_i}^{-1}, M_k^i)$.*
- *if $\ell_k = out(c, d)$ with $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a, b\}$ and d a channel name, then $\ell'_k = out(c, d)\rho_{\mathcal{C}h_i}^{-1}$*
- *if $\ell_k = \nu ch_m.out(c, ch_m)$ with $c \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $i \in \{a, b\}$ and ch_m a channel name, then $\ell'_k = \nu ch_m.out(c\rho_{\mathcal{C}h_i}^{-1}, ch_m)$*
- *if $\ell_k = \tau$ then $\ell'_k = \tau$*

Proof. The proof of this Lemma is very similar to the proof of Lemma 5.14. Indeed, in the proof of Lemma 5.14, we used Lemma B.2 to show that $\alpha \vDash \mathsf{test}_i([u]_i)$ implies that $\delta_i^\rho(\alpha) \vDash \mathsf{test}_i(\delta_i^\rho([u]_i))$. But Lemma B.2 shows that those two properties are equivalent. The same goes for Corollary B.1

The only difference is that in the case of the rule In, Lemma 5.13 cannot be called. Intuitively, Lemma 5.13 allows us to show that for all recipes applied on Φ'_S, we can create an equivalent recipe for Φ'_D; but not the other way around. On the other hand, the new hypothesis is added in this lemma (the last one) which is the corresponding result of Lemma 5.13.

Indeed we have $\nu\mathcal{E}_0.\delta^\rho(\Phi_+) \sim \nu\mathcal{E}_0.\Phi'_D$. But with our inductive step, we would have $\Phi''_D = \Phi'_D$, Φ''_S well tagged and $\Phi''_D\downarrow = \delta^\rho(\Phi''_S\downarrow)$. Thus we have that $\nu\mathcal{E}_0.\delta^\rho(\Phi_+\downarrow) \sim \nu\mathcal{E}.\delta^\rho(\Phi''_S\downarrow)$. Let M_k be the recipe from an input. By hypothesis, we have that $M_k\delta^\rho(\Phi_+\downarrow)\downarrow = \delta_i^\rho(M_k^i\Phi_+\downarrow)$. But by Lemma 5.13, there exists M such that $M\delta^\rho(\Phi_+\downarrow)\downarrow = \delta_i^\rho(M_k^i\Phi_+\downarrow)$ and $M\delta^\rho(\Phi''_S\downarrow)\downarrow = \delta_i^\rho(M_k^i\Phi''_S\downarrow)$. But $M_k\delta^\rho(\Phi_+\downarrow)\downarrow = M\delta^\rho(\Phi_+\downarrow)\downarrow$ implies $M_k\delta^\rho(\Phi''_S\downarrow)\downarrow = M\delta^\rho(\Phi''_S\downarrow)\downarrow$, which allows us to conclude that $M_k\delta^\rho(\Phi''_S\downarrow)\downarrow = M_k\Phi''_D\downarrow = \delta_i^\rho(M_k^i\Phi''_S\downarrow)$. □

We conclude this appendix with the proof of Theorem 5.2.

Theorem 5.2. *Let $\overline{P_A}, \overline{P'_A}$ (resp. $\overline{P_B}, \overline{P'_B}$) be two sequences of plain processes built $\mathcal{F}_a \cup \mathcal{F}_0$ (resp. $\mathcal{F}_b \cup \mathcal{F}_0$). Let $\mathcal{K}_0$ be a finite set of names of base type, and C and C' be two composition contexts. Let $\mathcal{K}'_0$ be a subset of $\mathcal{K}_0$ such that $\mathcal{K}'_0 = names(P_A, P'_A) \cap names(P_B, P'_B) \cap \mathcal{K}_0$. Let $\Phi_0 = \{ax_1 \rhd \mathsf{f}_1(k_1), \ldots, ax_n \rhd \mathsf{f}_n(k_n)\}$ with $\mathsf{f}_i \in \{\mathsf{pk}, \mathsf{vk}\}$, and $k_i \in \mathcal{K}_0$ for any $i \in \{1, \ldots, n\}$. Assume that the processes $(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_0)$ and $(\mathcal{K}_0; C[[\overline{P_B}]_b]; \Phi_0)$ (resp. $(\mathcal{K}_0; C[[\overline{P'_A}]_a]; \Phi_0)$, and $(\mathcal{K}_0; C[[\overline{P'_B}]_b]; \Phi_0)$):*

- *do not reveal any key in $\{k, \mathsf{pk}(k), \mathsf{vk}(k) \mid k \in \mathcal{K}'_0\}$ unless if the key occurs explicitly in Φ_0; and*
- *do not reveal any shared key in C (resp. C');*

Lastly, we assume that plain processes $\overline{P_A}, \overline{P'_A}$ and $\overline{P_B}, \overline{P'_B}$ do not use variable of channel type. In such a case,

$$\frac{(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a]; \Phi_0) \quad (\mathcal{K}_0; C[[\overline{P_B}]_b]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_B}]_b]; \Phi_0)}{(\mathcal{K}_0; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b]; \Phi_0)}$$

Proof. Before we start the proof, we rename Φ_0 into Φ_a. We colour Φ_a by a. If we assume that $\Phi_a = \{ax_1^a \rhd u_1, \ldots, ax_n^a \rhd u_n\}$, we build the frame Φ_b, coloured by b such that $\Phi_b = \{ax_1^b \rhd u_1, \ldots, ax_n^b \rhd u_n\}$. At last, let ρ_0 be the bijective renaming such that $\mathrm{img}(\rho_0) = \mathcal{K}'_0$ and $\mathrm{dom}(\rho_0)$ are composed of fresh names.

Let $\mathcal{C}h_a$ and $\mathcal{C}h_b$ be two sets of fresh channel type names. Furthermore, let $\rho_{\mathcal{C}h_a}$ be a bijective renaming from the public channel of $(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_0)$ and $(\mathcal{K}_0; C'[[\overline{P'_A}]_a]; \Phi_0)$ to $\mathcal{C}h_a$. We define $\rho_{\mathcal{C}h_b}$ in the same way.

We know by hypothesis that $(\mathcal{K}_0; C[[\overline{P_A}]_a]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a]; \Phi_0)$ and $(\mathcal{K}_0; C[[\overline{P_B}]_b]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_B}]_b]; \Phi_0)$. But the trace equivalence is stable under renaming. Thus, we have:

$$\begin{array}{c}(\mathcal{K}_0; C[[\overline{P_A}]_a \rho_{\mathcal{C}h_a}]; \Phi_a) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a \rho_{\mathcal{C}h_a}]; \Phi_a) \\ (\mathcal{K}_0\rho_0^{-1}; C[[\overline{P_B}]_b \rho_{\mathcal{C}h_b}\rho_0^{-1}]; \Phi_b\rho_0^{-1}) \approx_t (\mathcal{K}_0\rho_0^{-1}; C'[[\overline{P'_B}]_b \rho_{\mathcal{C}h_b}\rho_0^{-1}]; \Phi_b\rho_0^{-1})\end{array}$$

But $\mathrm{img}(\rho_0) = \mathcal{K}'_0$ and $\mathcal{K}'_0$ represents the sets of shared named of $\mathcal{K}_0$ betweenP_A, P'_A and P_B, P'_B. Hence, since $\mathrm{dom}(\Phi_a) \cap \mathrm{dom}(\Phi_b) = \emptyset$, we can compose the two equivalences such that if we denote $D = (\mathcal{K}_0 \cup \mathcal{K}_0\rho_0^{-1}; C[[\overline{P_A}]_a \rho_{\mathcal{C}h_a}] \mid C[[\overline{P_B}]_b \rho_{\mathcal{C}h_b}\rho_0^{-1}]; \Phi_a \uplus \Phi_b\rho_0^{-1})$ and $D' = (\mathcal{K}_0 \cup \mathcal{K}_0\rho_0^{-1}; C'[[\overline{P'_A}]_a \rho_{\mathcal{C}h_a}] \mid C'[[\overline{P'_B}]_b \rho_{\mathcal{C}h_b}\rho_0^{-1}]; \Phi_a \uplus \Phi_b\rho_0^{-1})$, we have that $D \approx_t D'$.

Let's denote now $S = (\mathcal{K}_0; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]; \Phi_a \uplus \Phi_b)$ and $S' = (\mathcal{K}_0; C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b]; \Phi_a \uplus \Phi_b)$.

We will show that $S \approx_t S'$. Indeed, since $\Phi_0 = \Phi_a$ and Φ_b have exactly the same terms, then $S \approx_t S'$ is equivalent to $(\mathcal{K}_0; C[[\overline{P_A}]_a \mid [\overline{P_B}]_b]; \Phi_0) \approx_t (\mathcal{K}_0; C'[[\overline{P'_A}]_a \mid [\overline{P'_B}]_b]; \Phi_0)$.

To avoid confusion, for a sequence of actions w including the τ actions, we will denote $\widetilde{w}$ the sequence of actions w where we removed the τ actions.

Let $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(S)$, by Lemma 5.12, there exists a renaming ρ and two bounded intermediate processes $S_1 = (\mathcal{E}_1; \{P_a, P_b\}; \Phi_a \uplus \Phi_b)$ and $D_1 = (\mathcal{E}_1; \{P_a\rho_{\mathcal{C}h_a}, P_b\rho^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_a \uplus \Phi_b\rho^{-1})$ such that

1. $\rho_{|\,\mathrm{dom}(\rho_0)} = \rho_0$, $\mathrm{dom}(\rho) \cup \mathrm{img}(\rho) \subseteq \mathcal{E}_1$ and $\mathrm{dom}(\rho)$ does not appear in $\{P_a, P_b\}$
2. for all $i \in \{a, b\}$, there exists P'_i built on $\mathcal{F}_i \cup \mathcal{F}_0$ such that such that $P_i = [P'_i]_i$
3. there exists Φ_1 such that $(\mathsf{tr}, \nu\mathcal{E}_1.\Phi_1) \in \mathsf{trace}(S_1)$ and $\nu\mathcal{E}_1.\Phi_1 \sim \nu\mathcal{E}.\Phi$
4. for all $(\mathsf{tr}', \nu\mathcal{E}_1.\Phi') \in \mathsf{trace}(D_1)$, there exists $\nu\mathcal{E}''.\Phi''$ such that $(\mathsf{tr}', \nu\mathcal{E}''.\Phi'') \in \mathsf{trace}(D)$ and $\nu\mathcal{E}_1.\Phi' \sim \nu\mathcal{E}''.\Phi''$

We colour P_a and P_b by a and b respectively.

In order to apply Lemma 5.14, we need to get rid of the possible internal communications between two processes of different colours in $(\mathsf{tr}, \nu\mathcal{E}_1.\Phi_1)$. Since $(\mathsf{tr}, \nu\mathcal{E}_1.\Phi_1) \in \mathsf{trace}(S_1)$, we have that $S_1 \stackrel{\mathsf{tr}}{\Rightarrow} (\mathcal{E}_1; \mathcal{P}; \Phi_1)$, for some $\mathcal{P}$. Thus, there exists w_S such that $\widetilde{w} = \mathsf{tr}$ and $S_1 \xrightarrow{w_S} (\mathcal{E}_1; \mathcal{P}; \Phi_1)$.

We show by induction on $\#\{\tau \in w_S\}$ that there exists w_S^+ such that $S_1 \xrightarrow{w_S^+} (\mathcal{E}_1; \mathcal{P}; \Phi_1 \uplus \Phi_+)$, for some Φ_+ and w_S^+ does not contain any τ action corresponding to an internal communications between two processes of different colours:
Base case $\#\{\tau \in w_S\} = 0$: The result trivially holds since there is no τ action.

Inductive step $\#\{\tau \in w_S\} > 0$: Assume that there is an internal communication between two processes of different colours (if not the result holds trivially). Thus there exists w_1, w_2 such that $w_S = w_1.\tau.w_2$ and $S_1 \xrightarrow{w_1} (\mathcal{E}_1; \mathcal{P}_1; \Phi_2) \xrightarrow{\tau} (\mathcal{E}_1; \mathcal{P}_2; \Phi_2) \xrightarrow{w_2} (\mathcal{E}_1; \mathcal{P}; \Phi_1)$. Since τ is an internal communication, then $\mathcal{P}_1 = \{\mathsf{in}(c, x).P_1; \mathsf{out}(c, u).P_2\} \uplus \mathcal{Q}$ and $\mathcal{P}_2 = \{P_1\{x \mapsto u\}; P_2\} \uplus \mathcal{Q}$, for some $P_1, P_2, \mathcal{Q}, c, x, u$. But by hypothesis, we know that processes of different colours do not share private channels. Thus c is a public channel. Hence, we have that $(\mathcal{E}_1; \mathcal{P}_1; \Phi_2) \xrightarrow{\nu ax.\mathsf{out}(c, ax)} P'$

where $P' = (\mathcal{E}_1; \{\mathsf{in}(c,x).P_1; P_2\} \uplus \mathcal{Q}; \Phi_2 \uplus \{ax \rhd u\})$. Once again, since c is a public channel, we now have that $P' \xrightarrow{\mathsf{in}(c,ax)} (\mathcal{E}_1; \mathcal{P}_2; \Phi_2 \uplus \{ax \rhd u\})$. A simple induction on the rest of the trace allows us to show that $(\mathcal{E}_1; \mathcal{P}_2; \Phi_2 \uplus \{ax \rhd u\}) \xrightarrow{w_2} (\mathcal{E}_1; \mathcal{P}; \Phi_1 \uplus \{ax \rhd u\})$. Since we removed the τ action without adding new ones, we can apply our inductive hypothesis on $w_1.\nu ax.\mathsf{out}(c,ax).\mathsf{in}(c,ax).w_2$ in order to conclude.

We showed that $S_1 \xrightarrow{w_S^+} (\mathcal{E}_1; \mathcal{P}; \Phi_1 \uplus \Phi_+)$, for some Φ_+ and w_S^+ does not contain any τ action corresponding to an internal communication between two processes of different colours. Let's denote $\Phi_S = \Phi_1 \uplus \Phi_+$, we have that $(\widetilde{w_S^+}, \nu\mathcal{E}_1.\Phi_S) \in \mathsf{trace}(S_1)$. Thanks to Lemma 5.14, we can deduce that Φ_S is well-tagged and there exists $D_1 \xrightarrow{w_D^+} (\mathcal{E}_1; \mathcal{P}_D; \Phi_D)$ such that $\Phi_D{\downarrow} = \delta^\rho(\Phi_S{\downarrow})$. Furthermore, if $w_S^+ = \ell_1 . \ldots . \ell_n$ then $w_D^+ = \ell'_1 . \ldots . \ell'_n$ such that for all $k \in \{1, \ldots, n\}$,

- if $\ell_k = \nu ax.\mathsf{out}(c,ax)$ is an output coming from a process coloured by $i \in \{a,b\}$, then $\ell'_k = \nu ax.\mathsf{out}(c\rho_{\mathcal{C}h_i}, ax)$
- if $\ell_k = \mathsf{in}(c,M)$ is an input coming from a process coloured by $i \in \{a,b\}$, then $\ell'_k = \mathsf{in}(c\rho_{\mathcal{C}h_i}, M_i)$ with $M_i\Phi_D{\downarrow} = \delta_i^\rho(M\Phi_S{\downarrow})$.
- if $\ell_k = \mathsf{out}(c,d)$ is an output coming from a process coloured by $i \in \{a,b\}$ with d a channel name, then $\ell'_k = \mathsf{out}(c\rho_{\mathcal{C}h_i}, d\rho_{\mathcal{C}h_i})$
- if $\ell_k = \nu ch_m.out(c, ch_m)$ is an output coming from a process coloured by $i \in \{a,b\}$ with ch_m a channel name, then $\ell'_k = \nu ch_m.out(c\rho_{\mathcal{C}h_i}, ch_m)$
- if $\ell_k = \tau$, then $\ell'_k = \tau$.

Hence we have that $(\widetilde{w_D^+}, \nu\mathcal{E}_1.\Phi_D) \in \mathsf{trace}(D_1)$. Thus, thanks to the property 4 established earlier with Lemma 5.12, there exists $\nu\mathcal{E}^1.\Phi_D^1$ such that $(\nu\mathcal{E}^1.\Phi_D^1) \in \mathsf{trace}(D)$ and $\nu\mathcal{E}^1.\Phi_D^1 \sim \nu\mathcal{E}_1.\Phi_D$ Furthermore, we showed that $D \approx_t D'$, hence we deduce that there exists $\nu\mathcal{E}'^1.\Phi_D'^1$ such that $(\widetilde{w_D^+}, \nu\mathcal{E}'^1.\Phi_D'^1) \in \mathsf{trace}(D')$ and $\nu\mathcal{E}^1.\Phi_D^1 \sim \nu\mathcal{E}'^1.\Phi_D'^1$.

Once again by Lemma 5.12, there exists a renaming ρ' and two bounded intermediate processes $S'_1 = (\mathcal{E}'_1; \{P'_a, P'_b\}; \Phi_a \uplus \Phi_b)$ and $D'_1 = (\mathcal{E}'_1; \{P'_a\rho_{\mathcal{C}h_a}, P'_b\rho^{-1}\rho_{\mathcal{C}h_b}\}; \Phi_a \uplus \Phi_b\rho^{-1})$ such that

1. $\rho'_{|\,\mathrm{dom}(\rho_0)} = \rho_0$, $\mathrm{dom}(\rho') \cup \mathrm{img}(\rho') \subseteq \mathcal{E}'_1$ and $\mathrm{dom}(\rho')$ does not appear in $\{P'_a, P'_b\}$
2. for all $i \in \{a,b\}$, there exists P''_i built on $\mathcal{F}_i \cup \mathcal{F}_0$ such that such that $P'_i = [P''_i]_i$
3. there exists Φ'_D such that $(\widetilde{w_D^+}, \nu\mathcal{E}'_1.\Phi'_D) \in \mathsf{trace}(D'_1)$ and $\nu\mathcal{E}'_1.\Phi'_D \sim \nu\mathcal{E}'^1.\Phi_D'^1$
4. for all $(\mathsf{tr}', \nu\mathcal{E}'_1.\Phi') \in \mathsf{trace}(S'_1)$, there exists $\nu\mathcal{E}''.\Phi''$ such that $(\mathsf{tr}', \nu\mathcal{E}''.\Phi'') \in \mathsf{trace}(S)$ and $\nu\mathcal{E}'_1.\Phi' \sim \nu\mathcal{E}''.\Phi''$

Since our processes are bounded intermediate process, we can assume that $\mathcal{E}_1 = \mathcal{E}'_1$ and $\rho = \rho'$ (if not we can apply some renaming on private name in $\mathcal{E}_1$ or $\mathcal{E}'_1$ in order to make them equal). Thus, $(\widetilde{w_D^+}, \nu\mathcal{E}_1.\Phi'_D) \in \mathsf{trace}(D'_1)$.

Let's summarise what we have proved so far. We had $(\mathsf{tr}, \nu\mathcal{E}.\Phi) \in \mathsf{trace}(S)$, then obtain a trace $(\mathsf{tr}, \nu\mathcal{E}_1.\Phi_1) \in \mathsf{trace}(S_1)$ such that $\nu\mathcal{E}.\Phi \sim \nu\mathcal{E}_1.\Phi_1$. We modified this trace into $(\widetilde{w_S^+}, \Phi_1 \uplus \Phi_+)$ where visible actions $\nu ax.\mathsf{out}(c,ax).\mathsf{in}(c,ax)$ replaced some τ actions. Then Lemma 5.12 allowed us to build the trace $(\widetilde{w_D^+}, \nu\mathcal{E}_1.\Phi_D) \in \mathsf{trace}(D_1)$ which only modify the terms in the trace but not the actions themselves. At last, we obtain, thanks to our hypothesis, that there exists Φ'_D such that $(\widetilde{w_D^+}, \nu\mathcal{E}_1.\Phi'_D) \in \mathsf{trace}(D'_1)$ such that $\nu\mathcal{E}_1.\Phi_D \sim \nu\mathcal{E}_1.\Phi'_D$.

$(\widetilde{w_D^+}, \nu\mathcal{E}_1.\Phi'_D) \in \mathsf{trace}(D'_1)$ implies that there exists $w_D'^+$ such that $\widetilde{w_D'^+} = \widetilde{w_D^+}$ and $D'_1 \xrightarrow{w_D'^+} (\mathcal{E}_1; \mathcal{P}'; \Phi'_D)$. It is possible that there exists $\ell \in \tau^*$ such that $\nu ax.\mathsf{out}(c_i,ax).\ell.\mathsf{in}(c_j,ax) \in w_D'^+$ where $c_i \in \mathrm{img}(\rho_{\mathcal{C}h_i})$, $c_j \in \mathrm{img}(\rho_{\mathcal{C}h_j})$ and $i \neq j$. But, thanks to Lemma 5.16, we can assume that this case does not occur and so there is no τ action between $\nu ax.\mathsf{out}(c_i,ax)$ and $\mathsf{in}(c_j,ax)$, for any $ax \in \mathrm{dom}(\Phi'_D)$.

Since $\nu\mathcal{E}_1.\Phi_D \sim \nu\mathcal{E}_1.\Phi'_D$, Φ_S is a well-tagged frame and $\Phi_D{\downarrow} = \delta^\rho(\Phi_S{\downarrow})$, we can apply Lemma 5.15 on $D'_1 \xrightarrow{w_D'^+} (\mathcal{E}_1; \mathcal{P}'; \Phi'_D)$ which allow us to deduce that there exists a well-tagged frame

Φ'_S such that $S'_1 \xrightarrow{w'^+_S} (\mathcal{E}_0; \mathcal{P}''; \Phi'_S)$, $\Phi'_D{\downarrow} = \delta^\rho(\Phi'_S{\downarrow})$, and if $w'^+_D = \ell_1.\ldots.\ell_n$ then $w'^+_S = \ell'_1.\ldots.\ell'_n$ and for all $k \in \{1, \ldots, n\}$,

— if $\ell_k = \nu ax.\mathsf{out}(c, ax)$ with $c \in \mathcal{C}h_i$, $i \in \{a, b\}$ then $\ell'_k = \nu ax.\mathsf{out}(c\rho^{-1}_{\mathcal{C}h_i}, ax)$

— if $\ell_k = \mathsf{in}(c, M_k)$ with $c \in \mathcal{C}h_i$, $i \in \{a, b\}$, then $\ell'_k = \mathsf{in}(c\rho^{-1}_{\mathcal{C}h_i}, M^i_k)$.

— if $\ell_k = \mathsf{out}(c, d)$ with $c \in \mathcal{C}h_i$, $i \in \{a, b\}$ and d a channel name, then $\ell'_k = \mathsf{out}(c\rho^{-1}_{\mathcal{C}h_i}, d\rho^{-1}_{\mathcal{C}h_i})$

— if $\ell_k = \tau$ then $\ell'_k = \tau$

where the couples (M_k, M^i_k) were generated by the application of Lemma 5.14 earlier on the trace $\widetilde{w^+_S}$. Note that it is possible because the channels (public and private) of processes with different colours are disjoints in D_1 and D'_1. For example, if we have $\mathsf{in}(c, M) \in w^+_D$ and $c \in \mathcal{C}h_a$, then we know for sure that the input was done by a process coloured by a in D'_1 and in D_1.

Hence, by construction of w^+_S and w'^+_S, we have in fact that $w^+_S = w'^+_S$ and so $(\widetilde{w^+_S}, \nu\mathcal{E}_1.\Phi'_S) \in \mathsf{trace}(S'_1)$. Thanks to Corollary 5.4, we can also deduce that $\nu\mathcal{E}_1.\Phi_S \sim \nu\mathcal{E}_1.\Phi'_S$.

But $\Phi_S = \Phi_1 \uplus \Phi_+$ and since $\mathrm{dom}(\Phi_S) = \mathrm{dom}(\Phi'_S)$, there exists Φ'_1 and Φ'_+ such that $\mathrm{dom}(\Phi'_1) = \mathrm{dom}(\Phi_1)$, $\mathrm{dom}(\Phi_+) = \mathrm{dom}(\Phi'_+)$ and $\Phi'_S = \Phi'_1 \uplus \Phi'_+$. Since the transformation between w'^+_D and w'^+_S only modifies the terms of the sequence of actions and not the actions themselves, and since we assume that there is no τ action between $\nu ax.\mathsf{out}(c_a, ax)$ and $\mathsf{in}(c_b, ax)$ in w'^+_S, for any $ax \in \mathrm{dom}(\Phi'_+)$, a simple induction on $|\mathrm{dom}(\Phi'_+)|$ allows us to show that $(\widetilde{w_S}, \nu\mathcal{E}_1.\Phi'_1) \in \mathsf{trace}(S'_1)$ and so $(\mathsf{tr}, \nu\mathcal{E}_1.\Phi'_1) \in \mathsf{trace}(S'_1)$. Moreover, since we have that $\nu\mathcal{E}_1.(\Phi_1 \uplus \Phi_+) \sim \nu\mathcal{E}_1.(\Phi'_1 \uplus \Phi'_+)$, we can deduce that $\nu\mathcal{E}_1.\Phi_1 \sim \nu\mathcal{E}_1.\Phi'_1$. At last, thanks to the property 4 obtained by application of Lemma 5.12, we deduce that there exists $\nu\mathcal{E}'.\Phi'$ such that $(\mathsf{tr}, \nu\mathcal{E}'.\Phi') \in \mathsf{trace}(S)$ and $\nu\mathcal{E}'.\Phi' \sim \nu\mathcal{E}_1.\Phi'_1$. Since we have $\nu\mathcal{E}.\Phi \sim \nu\mathcal{E}_1.\Phi_1$, we conclude that $(\mathsf{tr}, \nu\mathcal{E}'.\Phi') \in \mathsf{trace}(S)$ and $\nu\mathcal{E}.\Phi \sim \nu\mathcal{E}'.\Phi'$. Hence the result holds. □

Appendix C

Decision procedure of trace equivalence

C.1 Getting rid of some recipes

C.1.1 Getting rid of public names in the recipes

Let N be a positive integer. Let $a \in \mathcal{E}$. We denote $\sigma_{\mathcal{E},N,a}$ and $\theta_{\mathcal{E},N}$ the substitution defined such that for all $i \in \mathbb{N}^+$, $b_i\sigma_{\mathcal{E},N,a} = \mathsf{h}^{i\times N}(a)$ and $b_i\theta_{\mathcal{E},N} = \mathsf{h}^{i\times N}(ax_1)$. At last, we denote $\mathsf{len}_\mathsf{h}(u) = \max\{k \mid \mathsf{h}^k(u') \in st(u) \text{ for some } u'\}$. Intuitively, $\mathsf{len}_\mathsf{h}(u)$ represents the longest successive application of h in u.

Lemma C.1. *Let $\mathcal{E}$ be a finite set on names and $a \in \mathcal{E}$. Let $N \in \mathbb{N}^+$. For all $u, v \in \mathcal{T}(\mathcal{F},\mathcal{N})$, if $u\sigma_{\mathcal{E},N,a} = v\sigma^1_{\mathcal{E},N,a}$, $\mathsf{len}_\mathsf{h}(u) < N$ and $\mathsf{len}_\mathsf{h}(v) < N$ then $u = v$.*

Proof. We prove the result by induction on $|u\sigma_{\mathcal{E},N,a}|$.

Base case $|u\sigma_{\mathcal{E},N,a}| = 1$: In such a case, $u\sigma^1_{\mathcal{E},N,a} \in \mathcal{N}$. Since $N > 0$, we deduce that $fnames(u) \cap (\mathcal{N} \setminus \mathcal{E}) = \emptyset$. Hence $u \in \mathcal{E}$ and so $u\sigma_{\mathcal{E},N,a}$. Similarly, we have that $v\sigma_{\mathcal{E},N,a} = v$ and so $u = v$.

Inductive step $|u\sigma_{\mathcal{E},N,a}| > 1$: Otherwise we have $u\sigma_{\mathcal{E},N,a} = \mathsf{f}(u_1, \ldots, u_n)$. We do a case analysis on f:

- Case $\mathsf{f} = \mathsf{h}$: In such a case, there exists $k \in \mathbb{N}^+$ such that $u\sigma_{\mathcal{E},N,a} = \mathsf{h}^k(u_1)$ and $\mathsf{root}(u_1) \neq \mathsf{h}$ for some u_1. If $u_1 \neq a$ then by definition of $\sigma_{\mathcal{E},N,a}$, we deduce that there exists u'_1 such that $u'_1\sigma_{\mathcal{E},N,a} = u_1$ and $u = \mathsf{h}^k(u'_1)$. Similarly, we have that there exists $v'_1\sigma_{\mathcal{E},N,a} = u_1$ and $v = \mathsf{h}^k(v'_1)$. By inductive hypothesis, we deduce that $u'_1 = v'_1$ and so $u = \mathsf{h}^k(u'_1) = \mathsf{h}^k(v'_1) = v$.
  Assume now that $u_1 = a$. In such a case, there exists $i \in \mathbb{N}^+$ and $j \leq k$ such that $u = \mathsf{h}^j(b_i)$ and $k = j + i \times N$. Similarly, there exists $i' \in \mathbb{N}^+$ and $j' \leq k$ such that $v = \mathsf{h}^{j'}(b_{i'})$ and $k = j' + i' \times N$. But by hypothesis, we know that $\mathsf{len}_\mathsf{h}(u) < N$ and $\mathsf{len}_\mathsf{h}(v) < N$ hence $j < N$ and $j' < N$. Thus $j + i \times N = j' + i' \times N$ implies that $i = i'$ and $j = j'$. We conclude that $u = v$.
- Case $\mathsf{f} \neq \mathsf{h}$: By definition of $\sigma_{\mathcal{E},N,a}$, there exists $u'_1, \ldots, u'_n, v'_1, \ldots, v'_n$ such that $u = \mathsf{f}(u'_1, \ldots, u'_n)$, $v = \mathsf{f}(v'_1, \ldots, v'_n)$ and so for all $i \in \{1, \ldots, n\}$, $u'_i\sigma_{\mathcal{E},N,a} = u_i = v'_i\sigma_{\mathcal{E},N,a}$. By our inductive hypothesis on u'_i, v'_i, we conclude that $u'_i = v'_i$ for all $i \in \{1, \ldots, n\}$ and so $u = v$. □

In Lemma C.1, the conditions on the number of occurrence of h in u and v is crucial. Indeed, if we consider $N = 1$ and the terms $u = \mathsf{senc}(\mathsf{h}(a), b_1)$, $v = \mathsf{senc}(b_1, b_1)$ then we obtain $u\sigma_{\mathcal{E},N,a} = v\sigma_{\mathcal{E},N,a} = \mathsf{senc}(\mathsf{h}(a), \mathsf{h}(a))$. This is an issue when it comes to check if a message or not. Intuitively, our representation of public names should not change the validity of a term. However, if we consider the term $t = \mathsf{sdec}(u, \mathsf{h}(a))$ then $\mathsf{Message}(t)$ does not hold while $\mathsf{Message}(t\sigma_{\mathcal{E},N,a})$ holds.

Using the conditions on the number of occurrence of h in terms, we will see in the next lemma that such example can be avoided.

Lemma C.2. *Let $\mathcal{E}$ ba finite set of names and $a \in \mathcal{E}$. Let $N \in \mathbb{N}^+$. For all $u \in \mathcal{T}(\mathcal{F},\mathcal{N})$, if $\mathsf{len}_{\mathsf{h}}(u) < N$ then:*

— $\mathsf{Message}(u)$ *if, and only if,* $\mathsf{Message}(u\sigma_{\mathcal{E},N,a})$*; and*

— $(u\sigma_{\mathcal{E},N,a})\downarrow = (u\downarrow)\sigma_{\mathcal{E},N,a}$

Proof. We prove both results at the same time by induction on $|u|$:

Base case $|u| = 1$: In such a case, we have that $u \in \mathcal{N}$. If $u \in \mathcal{E}$ then $u\sigma_{\mathcal{E},N,a} = u$ and so the results trivially holds. Else there exists $i \in \mathbb{N}^+$ such that $u = b_i$. Thus $u\sigma_{\mathcal{E},N,a} = \mathsf{h}^{i\times N}(a)$. But $\mathsf{h} \in \mathcal{F}_c$ hence $(u\sigma_{\mathcal{E},N,a})\downarrow = u\sigma_{\mathcal{E},N,a} = (u\downarrow)\sigma_{\mathcal{E},N,a}$. Moreover, $\mathsf{Message}(b_i)$ and $\mathsf{Message}(\mathsf{h}^{i\times N}(a))$ both hold. Thus the result holds.

Inductive step $|u| > 1$: Otherwise $u = \mathsf{f}(u_1,\dots,u_n)$ with $\mathsf{f} \in \mathcal{F}$. We first prove that $(u\sigma_{\mathcal{E},N,a})\downarrow = (u\downarrow)\sigma_{\mathcal{E},N,a}$ by case analysis on f.

— Case 1, $\mathsf{f} \in \mathcal{F}_c$: In such a case, we have that $(u\sigma_{\mathcal{E},N,a})\downarrow = \mathsf{f}(u_1\sigma_{\mathcal{E},N,a}\downarrow,\dots,u_n\sigma_{\mathcal{E},N,a}\downarrow)$. By inductive hypothesis on $u_1,\dots,u_n$, we deduce that $(u\sigma_{\mathcal{E},N,a})\downarrow = \mathsf{f}((u_1\downarrow)\sigma_{\mathcal{E},N,a},\dots,(u_n\downarrow)\sigma_{\mathcal{E},N,a})$ and so $(u\sigma_{\mathcal{E},N,a})\downarrow = \mathsf{f}(u_1,\dots,u_n)\downarrow\sigma_{\mathcal{E},N,a}$.

— Case 2, $\mathsf{f} \in \mathcal{F}_d$: In such a case, $u\downarrow = \mathsf{f}(u_1\downarrow,\dots,u_n\downarrow)\downarrow$. Moreover, $u\sigma_{\mathcal{E},N,a}\downarrow = \mathsf{f}(u_1\sigma_{\mathcal{E},N,a}\downarrow,\dots,u_n\sigma_{\mathcal{E},N,a}\downarrow)\downarrow$. By inductive hypothesis on $u_1,\dots,u_n$, we deduce that $u\sigma_{\mathcal{E},N,a}\downarrow = \mathsf{f}(u_1\downarrow\sigma_{\mathcal{E},N,a},\dots,u_n\downarrow\sigma_{\mathcal{E},N,a})\downarrow$.

Consider $\mathsf{f}(v_1,\dots,v_n) \to v$ a fresh instance of the rewrite rule of f. f is reduced in $u\downarrow$ is equivalent to $(u_1\downarrow,\dots,u_n\downarrow)$ is unifiable with $(v_1,\dots,v_n)$. Similarly, f is reduce in $u\sigma_{\mathcal{E},N,a}\downarrow$ is equivalent to $(u_1\downarrow\sigma_{\mathcal{E},N,a},\dots,u_n\downarrow\sigma_{\mathcal{E},N,a})$ is unifiable with $(v_1,\dots,v_n)$. We prove by case analysis on the rewrite rule involve that f is reduced in $u\downarrow$ is equivalent to f is reduce in $u\sigma_{\mathcal{E},N,a}\downarrow$:

— $\mathsf{f} = \mathsf{sdec}$: In such a case, we have that $u = \mathsf{sdec}(u_1,u_2)$. Hence f is reduced in $u\downarrow$ is equivalent to $u_1\downarrow = \mathsf{senc}(v_1,v_2)$ and $u_2\downarrow = v_2$ for some v_1,v_2. Moreover, f is reduced in $u\sigma_{\mathcal{E},N,a}\downarrow$ is equivalent to $u_1\downarrow\sigma_{\mathcal{E},N,a} = \mathsf{senc}(v_1',v_2')$ and $u_2\downarrow\sigma_{\mathcal{E},N,a} = v_2'$ for some v_1',v_2'. First of all, $u_1\downarrow = \mathsf{senc}(v_1,v_2)$ and $u_2\downarrow = v_2$ implies that $u_1\downarrow = \mathsf{senc}(v_1\sigma_{\mathcal{E},N,a},v_2\sigma_{\mathcal{E},N,a})$ and $u_2\downarrow\sigma_{\mathcal{E},N,a} = v_2\sigma_{\mathcal{E},N,a}$. Hence f is reduced in $u\downarrow$ implies f is reduced in $u\sigma_{\mathcal{E},N,a}\downarrow$. Secondly, by definition of $\sigma_{\mathcal{E},N,a}$, $u_1\downarrow\sigma_{\mathcal{E},N,a} = \mathsf{senc}(v_1',v_2')$ implies that there exits v_1'',v_2'' such that $u_1\downarrow = \mathsf{senc}(v_1'',v_2'')$ and $v_1''\sigma_{\mathcal{E},N,a} = v_1'$ and $v_2''\sigma_{\mathcal{E},N,a} = v_2' = u_2\downarrow\sigma_{\mathcal{E},N,a}$. By Lemma C.1, we obtain that $v_2'' = u_2\downarrow$ and so $u_1\downarrow = \mathsf{senc}(v_1',u_2\downarrow)$. Hence f is reduced in $u\sigma_{\mathcal{E},N,a}\downarrow$ implies that f is reduced in $u\downarrow$.

— $\mathsf{f} \in \{\tilde{,}\mathsf{adec},\mathsf{proj}_1,\mathsf{proj}_2\}$: Similar to the case $\mathsf{f} = \mathsf{sdec}$.

We have shown that f is reduced in $u\downarrow$ is equivalent to f is reduce in $u\sigma_{\mathcal{E},N,a}\downarrow$. Since all rewrite rules are subterm convergent, we deduce that $(u\sigma_{\mathcal{E},N,a})\downarrow = (u\downarrow)\sigma_{\mathcal{E},N,a}$.

At last, $\mathsf{Message}(u)$ is equivalent to $\mathsf{Message}(u_i)$ for all $i \in \{1,\dots,n\}$ and $u\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. By inductive hypothesis, we know that $\mathsf{Message}(u_i)$ if and only if $\mathsf{Message}(u_i\sigma_{\mathcal{E},N,a})$. Furthermore, we have shown that $(u\sigma_{\mathcal{E},N,a})\downarrow = (u\downarrow)\sigma_{\mathcal{E},N,a}$ and we know that for all $i \in \mathbb{N}^+$, $b_i\sigma_{\mathcal{E},N,a} \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. We can deduce that $u\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ is equivalent to $(u\downarrow)\sigma_{\mathcal{E},N,a} \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ which is equivalent to $(u\sigma_{\mathcal{E},N,a})\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Hence we conclude that $\mathsf{Message}(u)$ if and only if $\mathsf{Message}(u\sigma_{\mathcal{E},N,a})$. □

Corollary C.1. *Let $\mathcal{E}$ be a finite set of names. Let $N \in \mathbb{N}^+$. Let Φ be a ground frame such that $ax_1\Phi = a \in \mathcal{E}$. For all $\xi \in \mathcal{T}(\mathcal{F},\mathcal{AX}\cup\mathcal{N})$, if $param(\xi) \subseteq \mathrm{dom}(\Phi)$ and $\mathsf{len}_{\mathsf{h}}(\xi\Phi) < N$ then:*

— $\mathsf{Message}(\xi\Phi)$ *if and only if* $\mathsf{Message}((\xi\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a}))$

— $(\xi\Phi\downarrow)\sigma_{\mathcal{E},N,a} = (\xi\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a})\downarrow$

Proof. By definition of $\sigma_{\mathcal{E},N,a}$ and $\theta_{\mathcal{E},N}$, we have that for all $i \in \mathbb{N}^+$, $b_i\sigma_{\mathcal{E},N,a} = b_i\theta_{\mathcal{E},N}\Phi = (b_i\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a})$. More over, for all $ax_k \in \mathrm{dom}(\Phi)$, $ax_k\Phi\sigma_{\mathcal{E},N,a} = (ax_k\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a})$. Hence we have $\xi\Phi\sigma_{\mathcal{E},N,a} = (\xi\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a})$. We conclude by applying Lemma C.2 to $\xi\Phi$. □

Lemma 6.2. *Let Σ and Σ' two sets of constraint systems that contain constraint systems having the same structure. Assume that $\mathcal{E}$ is the common set of privates names in Σ and Σ'. At last, assume that there exists $a \in \mathcal{E}$ such that for all $\mathcal{C} \in \Sigma \cup \Sigma'$, $ax_1\Phi = a$ with Φ the frame of $\mathcal{C}$.*

If $\Sigma \approx_s^{\mathcal{T}(F,\mathcal{AX})} \Sigma'$ then for all $\mathcal{C} \in \Sigma$, for all $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$, there exists N' such that for all $N > N'$, there exist $\mathcal{C}' \in \Sigma'$ and a substitution σ' such that $(\sigma', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $\nu\mathcal{E}.\Phi\sigma\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \nu\mathcal{E}.\Phi'\sigma'\sigma_{\mathcal{E},N,a}$.

Proof. Let $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq) \in \Sigma$ and let $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$. Let N' be the integer equal to the number of occurrence of the function symbol h in Σ, Σ' and θ. Let $N > N'$. Before proving the result, we show the following proposition:

Proposition: For all $(\mathcal{E}; \Phi'; D'; Eq') \in \Sigma \cup \Sigma'$, for all substitution σ', if for all $(X, k \overset{?}{\vdash} x) \in D'$, $(X\theta)(\Phi'\sigma')$ and $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$, then

— for all $(X, k \overset{?}{\vdash} x) \in D'$, $\mathsf{len_h}(x\sigma') \leq N_k(\mathcal{C}')$

— for all $(ax_k \rhd u_k) \in \Phi$, $\mathsf{len_h}(u_i\sigma') \leq N_{k-1}(\mathcal{C}') + M_k(\mathcal{C}')$

where $M_k(\mathcal{C}')$ is the number of occurrence in $\{u_j \mid (ax_j \rhd u_j) \in \Phi' \wedge j \leq k\}$ and $N_k(\mathcal{C}')$ is the number of occurrence in $\{Y\theta \mid (Y, j \overset{?}{\vdash} y) \in D' \wedge j \leq k\}$ plus $M_k(\mathcal{C}')$. We assume that $M_0 = N_0 = 0$ We prove these both results at the same time by induction on k.

Base case $k = 1$: In such a case, by hypothesis on Φ' we know that $ax_1\Phi' = a$. Hence $\mathsf{len_h}(a\sigma') = 0 = M_1$. Let $(X, 1 \overset{?}{\vdash} x) \in D'$. Since $param(X\theta) \subseteq \{ax_1\}$, $(X\theta)(\Phi'\sigma') = x\sigma'$ and $ax_1\Phi' = a$, we deduce that $\mathsf{len_h}(x\sigma') = \mathsf{len_h}(X\theta)$. But N_1 is bigger than $\mathsf{len_h}(X\theta)$. Thus we deduce that $\mathsf{len_h}(X\theta) \leq N_1$ and so $\mathsf{len_h}(x\sigma') \leq N_1$.

Inductive step $k > 1$: Let $(ax_k \rhd u_k) \in \Phi'$. Let $\mathsf{h}^j(u) \in st(u_k\sigma)$ for some j and u. Either (a) there exists $u' \in st(u_k)$ and j' such that $u' = \mathsf{h}^{j'}(x)$ and $x\sigma = \mathsf{h}^{j-j'}(u)$ with $x \in \mathcal{X}^1$; or (b) there exists $x \in vars^1(u_k)$ such that $\mathsf{h}^j(u) \in st(x\sigma')$.

In case (a), by definition of $M_k(\mathcal{C}')$, we have $j' \leq M_k(\mathcal{C}')$ and so $j' + N_{k-1}(\mathcal{C}') \leq M_k(\mathcal{C}') + N_{k-1}(\mathcal{C}')$. Moreover, by definition of a constraint system, we know that for all $x \in vars^1(u_k)$, there exists $(X, i \overset{?}{\vdash} x) \in D'$ such that $i < k$. By inductive hypothesis on x, we deduce that $\mathsf{len_h}(x\sigma') \leq N_i(\mathcal{C}')$. Since $N_i(\mathcal{C}') \leq N_{k-1}(\mathcal{C}')$, we deduce that $N_{k-1}(\mathcal{C}')+M_k(\mathcal{C}') \geq j'+N_{k-1}(\mathcal{C}') \geq j' + N_i(\mathcal{C}') \geq j' + \mathsf{len_h}(x\sigma') \geq j' + j - j' = j$. Hence, it implies that $N_{k-1}(\mathcal{C}') + M_k(\mathcal{C}') \geq j$. In case (b), once again by inductive hypothesis on x, we deduce that $\mathsf{len_h}(x\sigma') \leq N_i(\mathcal{C}')$. Since $N_i(\mathcal{C}') \leq N_{k-1}(\mathcal{C}')$ and $j \leq \mathsf{len_h}(x\sigma')$ then $j \leq N_{k-1}(\mathcal{C}') + M_k(\mathcal{C}')$.

Since we have showed that for all $\mathsf{h}^j(u) \in st(u_k\sigma')$, $j \leq N_{k-1}(\mathcal{C}') + M_k(\mathcal{C}')$, we conclude that $\mathsf{len_h}(u_k\sigma') \leq M_k(\mathcal{C}') + N_{k-1}(\mathcal{C}')$

Let $(X, k \overset{?}{\vdash} x) \in D'$, by hypothesis on θ and σ', we have that $(X\theta)(\Phi'\sigma') = x\sigma'$ and $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$. Let $\mathsf{h}^j(u) \in st(x\sigma')$ for some j and u. Either (a) there exists $\xi \in st(X\theta)$ and j' such that $u' = \mathsf{h}^{j'}(ax_i)$, $ax_i\Phi\sigma' = \mathsf{h}^{j-j'}(u)$ for some $i \leq k$; or (b) there exists $ax_i \in \mathrm{dom}(\Phi')$ such that $\mathsf{h}^j(u) \in st(ax_i\Phi'\sigma')$; or (c) $\mathsf{h}^j(u) \in st(X\theta)$.

In case (a), we have proved that $\mathsf{len_h}(ax_i\Phi'\sigma') \leq M_k(\mathcal{C}')$ and by definition of $N_k(\mathcal{C}')$, we deduce that $N_k(\mathcal{C}') \geq j' + M_k(\mathcal{C}')$ and so $N_k(\mathcal{C}') \geq j' + \mathsf{len_h}(ax_i\Phi\sigma')$. Since $\mathsf{len_h}(ax_i\Phi\sigma') \geq j - j'$, we conclude that $N_k(\mathcal{C}') \geq j$. In case (b), thanks to $\mathsf{len_h}(ax_i\Phi'\sigma') \leq M_k(\mathcal{C}')$, we directly conclude that $j \leq N_k(\mathcal{C}')$. In case (c), the definition of $N_k(\mathcal{C}')$ implies that $j \leq N_k(\mathcal{C}')$.

Since we have showed that for all $\mathsf{h}^j(u) \in st(x\sigma')$, $j \leq N_k(\mathcal{C}')$, we conclude that $\mathsf{len_h}(x\sigma') \leq N_k(\mathcal{C}')$.

Main result: Let $\theta' = \theta\theta_{\mathcal{E},N}$ and $\sigma' = \sigma\sigma_{\mathcal{E},N,a}$. We show that $(\sigma', \theta') \in \mathsf{Sol}_c(\mathcal{C})$:

— let $(X, k \overset{?}{\vdash} x) \in D$. $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ implies that $(X\theta)(\Phi\sigma) = x\sigma$, $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$ and $\mathsf{Message}(x\sigma)$. Moreover, by our proposition, $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ also implies that $\mathsf{len_h}(x\sigma) \leq N_k(\mathcal{C}) < N$. By Corollary C.1, $\mathsf{Message}(x\sigma)$ implies that $\mathsf{Message}((X\theta\theta_{\mathcal{E},N})(\Phi\sigma\sigma_{\mathcal{E},N,a}))$. Hence, we deduce that $(X\theta')(\Phi\sigma') = x\sigma\sigma_{\mathcal{E},N,a} = x\sigma'$ and $\mathsf{Message}(x\sigma')$.

At last, since $param(b_i\theta_{\mathcal{E},N}) = \{ax_1\}$ for all $i \in \mathbb{N}^+$, we conclude that $param(X\theta') \subseteq \{ax_1, \ldots, ax_k\}$.

— let $(s \stackrel{?}{=} s')$ or $(s \stackrel{?}{\neq} s')$ in Eq. Thanks to our proposition, we know that for all $x \in vars^1(s, s')$, $\mathsf{len_h}(x\sigma) \leq N_k$ with $k = |\Phi|$. But N is strictly bigger that N_k plus the number of occurrence of h in s and s'. Hence, we deduce that $\mathsf{len_h}(s\sigma) < N$ and $\mathsf{len_h}(s'\sigma) < N$. By Lemma C.2, we deduce that $\mathsf{Message}(s\sigma)$ is equivalent to $\mathsf{Message}(s\sigma\sigma_{\mathcal{E},N,a})$; and $\mathsf{Message}(s'\sigma)$ is equivalent to $\mathsf{Message}(s'\sigma\sigma_{\mathcal{E},N,a})$; and $s\sigma{\downarrow}\sigma_{\mathcal{E},N,a} = (s\sigma\sigma_{\mathcal{E},N,a}){\downarrow}$; and $s'\sigma{\downarrow}\sigma_{\mathcal{E},N,a} = (s'\sigma\sigma_{\mathcal{E},N,a}){\downarrow}$. But thanks to Lemma C.1, $(s\sigma{\downarrow})\sigma_{\mathcal{E},N,a} = (s'\sigma{\downarrow})\sigma_{\mathcal{E},N,a}$ is equivalent to $s\sigma{\downarrow} = s'\sigma{\downarrow}$. Therefore we can deduce that $s\sigma{\downarrow} = s'\sigma{\downarrow}$, $\mathsf{Message}(s\sigma)$ and $\mathsf{Message}(s'\sigma)$ is equivalent to $s\sigma'{\downarrow} = s'\sigma'{\downarrow}$, $\mathsf{Message}(s\sigma')$ and $\mathsf{Message}(s'\sigma')$. Thus, if $(s \stackrel{?}{=} s')$ (resp. $(s \stackrel{?}{\neq} s')$) is in Eq then $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ implies that $s\sigma'{\downarrow} = s'\sigma'{\downarrow}$, $\mathsf{Message}(s\sigma')$ and $\mathsf{Message}(s'\sigma')$ (resp. $s\sigma'{\downarrow} \neq s'\sigma'{\downarrow}$, or $\neg\mathsf{Message}(s\sigma')$ or $\neg\mathsf{Message}(s'\sigma')$).

— let $(ax_i \rhd u_i) \in \Phi$. By our proposition, we know that $\mathsf{len_h}(u_i\sigma) < N$ hence thanks to Lemma C.1, $(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C})$ implies that $\mathsf{Message}(u_i\sigma\sigma_{\mathcal{E},N,a})$ and so $\mathsf{Message}(u_i\sigma')$

It conclude the proof that $(\sigma', \theta') \in \mathsf{Sol}_c(\mathcal{C})$.

By hypothesis, $\Sigma \approx_s^{\mathcal{T}(F,\mathcal{A}\mathcal{X})} \Sigma'$ hence there exists a constraint system $\mathcal{C}' = (\mathcal{E}; \Phi'; D'; Eq')$ and a substitution σ'' such that $(\sigma'', \theta') \in \mathsf{Sol}_c(\mathcal{C})$ and $\nu\mathcal{E}.\Phi\sigma' \sim_{\mathcal{T}(\mathcal{F},\mathcal{A}\mathcal{X})} \nu\mathcal{E}.\Phi'\sigma''$. We now show that there exists σ''' such that $\sigma'' = \sigma'''\sigma_{\mathcal{E},N,a}$ and $(\sigma''', \theta) \in \mathsf{Sol}_c(\mathcal{C})$.

We define σ''' such that $\mathrm{dom}(\sigma''') = \mathrm{dom}(\sigma'')$ and for all $(X, k \stackrel{?}{\vdash} x) \in D'$, $x\sigma''' = (X\theta)(\Phi\sigma''')$. Such substitution exists thanks to the origination property of the constraint system $\mathcal{C}'$. We prove by induction on k that for all $(X, k \stackrel{?}{\vdash} x) \in D'$, $x\sigma'''\sigma_{\mathcal{E},N,a} = x\sigma''$, $\mathsf{Message}(x\sigma''')$; and for all $(ax_k \rhd u_k) \in \Phi'$, $\mathsf{Message}(u_k\sigma''')$.

Base case $k = 0$: This case is impossible.

Base case $k > 0$: Let $(ax_k \rhd u_k) \in \Phi'$. By the origination property of a constraint system, we know that for all $y \in vars^1(u_k)$, there exists $(Y, i \stackrel{?}{\vdash} y) \in D'$ such that $i < k$. Hence by our inductive hypothesis, we deduce that for all $y \in vars^1(u_k)$, $y\sigma'''\sigma_{\mathcal{E},N,a} = y\sigma''$. Hence we deduce that $u_k\sigma'''\sigma_{\mathcal{E},N,a} = u_k\sigma''$. Thanks to our proposition, we know that $\mathsf{len}_<(u_k\sigma''')N$. Moreover, we have $(\sigma'', \theta') \in \mathsf{Sol}_c(\mathcal{C}')$ which implies $\mathsf{Message}(u_k\sigma'')$. Hence by Lemma C.2, we deduce that $\mathsf{Message}(u_k\sigma''')$.

Let $(X, k \stackrel{?}{\vdash} x) \in D'$. Let $(ax_i \rhd u_i) \in \Phi'$ such that $ax_i \in X\theta'$ and so $i \leq k$. By our inductive hypothesis on the variable of u_i, we deduce that $u_i\sigma'''\sigma_{\mathcal{E},N,a} = u_i\sigma''$. Hence $(X\theta')(\Phi\sigma'') = (X\theta')(\Phi\sigma'''\sigma_{\mathcal{E},N,a})$. Since $\theta' = \theta\theta_{\mathcal{E},N}$, then $x\sigma'' = (X\theta')(\Phi\sigma'') = (X\theta)(\Phi\sigma''')\sigma_{\mathcal{E},N,a} = x\sigma'''\sigma_{\mathcal{E},N,a}$. Moreover $(\sigma'', \theta') \in \mathsf{Sol}_c(\mathcal{C}')$ implies $\mathsf{Message}(x\sigma'')$. But, thanks to our proposition, we know that $\mathsf{len_h}(x\sigma''') \leq N_k(\mathcal{C}') < N$ therefore by Lemma C.2, we deduce that $\mathsf{Message}(x\sigma'')$ implies $\mathsf{Message}(x\sigma'''\sigma_{\mathcal{E},N,a})$ which implies $\mathsf{Message}(x\sigma''')$.

It remains to show that for all $(s \stackrel{?}{=} s') \in \Phi'$, $s\sigma'''{\downarrow} = s'\sigma'''{\downarrow}$, $\mathsf{Message}(s\sigma''')$ and $\mathsf{Message}(s'\sigma''')$. We know by $(\sigma'', \theta') \in \mathsf{Sol}_c(\mathcal{C}')$ that $s\sigma''{\downarrow} = s'\sigma''{\downarrow}$, $\mathsf{Message}(s\sigma'')$ and $\mathsf{Message}(s'\sigma'')$. Since $\mathsf{len}_<(s\sigma''')N$, $\mathsf{len}_<(s'\sigma''')N$ and $\sigma'''\sigma_{\mathcal{E},N,a}$, we can apply Lemma C.2 which allows us to conclude. For all $(s \stackrel{?}{\neq} s') \in \Phi'$, we similarly prove that $s\sigma'''{\downarrow} \neq s'\sigma'''{\downarrow}$, or $\neg\mathsf{Message}(s\sigma''')$ or $\neg\mathsf{Message}(s'\sigma''')$. Thus we conclude that $(\sigma''', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$.

We have proved that there exists $\mathcal{C}' \in \Sigma'$ and a substitution σ''' such that $(\sigma''', \theta) \in \mathsf{Sol}_c(\mathcal{C}')$ and $\nu\mathcal{E}.\Phi\sigma\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{A}\mathcal{X})} \nu\mathcal{E}.\Phi'\sigma'''\sigma_{\mathcal{E},N,a}$. Hence the result holds. □

Lemma 6.3. *Let $\mathcal{E}$ be a set of private names. Let Φ and Φ' two ground frames with the same domain and $ax_1\Phi = ax_1\Phi' \in \mathcal{E}$. If for all N', there exists $N > N'$ such that $\Phi\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{A}\mathcal{X})} \Phi'\sigma_{\mathcal{E},N,a}$ then $\Phi \sim_c \Phi'$.*

Proof. Let $\xi, \xi' \in \mathcal{T}(\mathcal{F}, \mathcal{AX} \cup \mathcal{N} \setminus \mathcal{E})$ such that $param(\xi) \subseteq \mathrm{dom}(\Phi)$, $param(\xi') \subseteq \mathrm{dom}(\Phi')$ and $fnames(\xi, \xi') \cap \mathcal{E} = \emptyset$. Let N' the number of occurrence of h in ξ, ξ', Φ and Φ'. By hypothesis the exists $N > N'$ such that $\Phi\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma_{\mathcal{E},N,a}$. Since $N > N'$, we have that $\mathsf{len_h}(\xi\Phi) < N$, $\mathsf{len_h}(\xi'\Phi) < N$, $\mathsf{len_h}(\xi\Phi') < N$ and $\mathsf{len_h}(\xi'\Phi') < N$.

— Thanks to Corollary C.1 and $\mathsf{len_h}(\xi\Phi) < N$, $\mathsf{Message}(\xi\Phi)$ is equivalent to $\mathsf{Message}((\xi\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N}))$. But $\xi\theta_{\mathcal{E},N} \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$ and $param(\xi\theta_{\mathcal{E},N}) \subseteq \mathrm{dom}(\Phi) = \mathrm{dom}(\Phi\sigma_{\mathcal{E},N,a})$. Hence we deduce that $\Phi\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma_{\mathcal{E},N,a}$ implies that $\mathsf{Message}((\xi\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N}))$ is equivalent to $\mathsf{Message}((\xi\theta_{\mathcal{E},N})(\Phi'\sigma_{\mathcal{E},N}))$.

Once again by Corollary C.1 and $\mathsf{len_h}(\xi\Phi') < N$, we deduce that $\mathsf{Message}((\xi\theta_{\mathcal{E},N})(\Phi'\sigma_{\mathcal{E},N}))$ is equivalent to $\mathsf{Message}(\xi\Phi')$. Thus $\mathsf{Message}(\xi\Phi)$ is equivalent to $\mathsf{Message}(\xi\Phi')$.

— Assume that $\xi\Phi{\downarrow} = \xi'\Phi{\downarrow}$, $\mathsf{Message}(\xi\Phi)$ and $\mathsf{Message}(\xi'\Phi)$. Corollary C.1, $\mathsf{len_h}(\xi\Phi) < N$ and $\mathsf{len_h}(\xi'\Phi) < N$ imply that $(\xi\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a}){\downarrow} = (\xi\Phi{\downarrow})\sigma_{\mathcal{E},N,a}$ and $(\xi'\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a}){\downarrow} = (\xi'\Phi{\downarrow})\sigma_{\mathcal{E},N,a}$. Hence from $\xi\Phi{\downarrow} = \xi'\Phi{\downarrow}$, we deduce that $(\xi\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a}){\downarrow} = (\xi'\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a}){\downarrow}$. But $(\xi\theta_{\mathcal{E},N}), (\xi\theta_{\mathcal{E},N}) \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$. Moreover, thanks to by Corollary C.1, we also have $\mathsf{Message}((\xi'\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a}))$ and $\mathsf{Message}((\xi\theta_{\mathcal{E},N})(\Phi\sigma_{\mathcal{E},N,a}))$. Thus we deduce that $\Phi\sigma_{\mathcal{E},N,a} \sim_{\mathcal{T}(\mathcal{F},\mathcal{AX})} \Phi'\sigma_{\mathcal{E},N,a}$ implies that $(\xi\theta_{\mathcal{E},N})(\Phi'\sigma_{\mathcal{E},N,a}){\downarrow} = (\xi'\theta_{\mathcal{E},N})(\Phi'\sigma_{\mathcal{E},N,a}){\downarrow}$.

Once again by Corollary C.1, $\mathsf{len_h}(\xi\Phi') < N$ and $\mathsf{len_h}(\xi'\Phi') < N$, $(\xi\theta_{\mathcal{E},N})(\Phi'\sigma_{\mathcal{E},N,a}){\downarrow} = (\xi'\theta_{\mathcal{E},N})(\Phi'\sigma_{\mathcal{E},N,a}){\downarrow}$ implies that $(\xi\Phi'{\downarrow})\sigma_{\mathcal{E},N,a} = (\xi'\Phi'{\downarrow})\sigma_{\mathcal{E},N,a}$. Thus thanks to Lemma C.1, we deduce that $\xi\Phi'{\downarrow} = \xi'\Phi'{\downarrow}$. The proof of the other side of the static equivalence can be done symmetrically.

We can conclude that $\Phi \sim_c \Phi'$. □

C.1.2 Normalised recipe

For a recipe ξ and a frame Φ, we say that $\mathsf{root}(\xi)$ is *not reduced* if $\mathsf{f}(\xi_1, \ldots, \xi_n)\Phi{\downarrow} = \mathsf{f}(\xi_1\Phi{\downarrow}, \ldots, \xi_n\Phi{\downarrow})$ and $\mathsf{root}(\xi) \in \mathcal{F}_d$ with $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$ for some $\xi_1, \ldots, \xi_n$. We establish three properties on the recipes in Π_n:

Lemma C.3. *Let Φ be a ground frame such that for all $(ax_i \rhd u_i) \in \Phi$, $\mathsf{Message}(u_i)$. Let ξ be a recipe such that $param(\xi) \subseteq \mathrm{dom}(\Phi)$, $\xi\Phi{\downarrow} \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. If for every $\xi' \in st(\xi)$,*

$$\xi' = \mathsf{f}(\mathsf{g}(\xi_1, \ldots, \xi_n), \ldots, \beta_m),\ \mathsf{f} \in \mathcal{F}_d \text{ and } \mathsf{g} \in \mathcal{F}_c \text{ implies } \mathsf{f} \text{ is not reduced,}$$

then, either $\mathsf{root}(\xi) \in \mathcal{F}_c$ or $\mathsf{root}(\xi)$ is not reduced.

Proof. We prove this result by induction on the size of ξ.

Base case: $|\xi| = 1$. In such a case, $\xi \in \mathrm{dom}(\Phi)$ and so $\xi\Phi{\downarrow} \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ by hypothesis on the frame. Hence the result trivially holds.

Induction step: $|\xi| > 1$. If $\mathsf{root}(\xi) \in \mathcal{F}_c$ or $\mathsf{root}(\xi)$ is not reduced then the property trivially holds. Else we have that $\xi = \mathsf{f}(\xi_1, \ldots, \xi_m)$ with $\mathsf{f} \in \mathcal{F}_d$ and f is reduced. We show that this case is impossible.

f is reduced implies that there is a rewrite rule $\mathsf{f}(u_1, \ldots, u_n) \to u$ such that $\mathsf{f}(u_1, \ldots, u_n)$ and $\mathsf{f}(\xi_1\Phi{\downarrow}, \ldots, \xi_n\Phi{\downarrow})$ are unifiable. Since $\xi\Phi{\downarrow} \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $\xi\Phi{\downarrow} \in st(\xi_1\Phi{\downarrow})$ (since $u \in st(u_1)$), we have that $\xi_1\Phi{\downarrow} \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. By applying our induction hypothesis on ξ_1, we deduce that either $\mathsf{root}(\xi_1) \in \mathcal{F}_c$ or $\mathsf{root}(\xi_1)$ is not reduced.

— If $\mathsf{root}(\xi_1) \in \mathcal{F}_c$ then by hypothesis on the subterms of ξ, we deduce that f is not reduced which is in contradiction with the hypothesis.

— If $\mathsf{root}(\xi_1)$ is not reduced, then $\mathsf{root}(\xi_1\Phi{\downarrow}) \in \mathcal{F}_d$. This contradicts the fact that $\mathsf{f}(u_1, \ldots, u_n)$ and $\mathsf{f}(\xi_1\Phi{\downarrow}, \ldots, \xi_n\Phi{\downarrow})$ are unifiable since we have that $\mathsf{root}(u_1) \in \mathcal{F}_c$.

This allows us to conclude that either $\mathsf{root}(\xi) \in \mathcal{F}_c$ or $\mathsf{root}(\xi)$ is not reduced. □

The following corollary is a direct consequence of Lemma C.3 since by definition of $\xi \in \Pi_n$, there is no $\xi' \in st(\xi)$ of form $\mathsf{f}(\xi_1, \ldots, \xi_n)$ with $\mathsf{f} \in \mathcal{F}_d$ and $\mathsf{root}(\xi_1) \in \mathcal{F}_c$.

Corollary C.2. *Let Φ be a ground frame such that for all $(ax_i \rhd u_i) \in \Phi$, $\mathsf{Message}(u_i)$. Let $\xi \in \Pi_n$ such that $param(\xi) \subseteq \mathrm{dom}(\Phi)$ and $\xi\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. We have that either $\mathsf{root}(\xi) \in \mathcal{F}_c$ or $\mathsf{root}(\xi)$ is not reduced.*

Lemma 6.5. *Let Φ be a ground frame such that for all $(ax_i \rhd u_i) \in \Phi$, $\mathsf{Message}(u_i)$. Let $\xi \in \Pi_n$ a ground recipe such that $param(\xi) \in \mathrm{dom}(\Phi)$. $\xi\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ if, and only if, $\mathsf{Message}(\xi\Phi)$ holds.*

Proof. We prove this result by induction on the size of ξ.

Base case: $|\xi| = 1$. In such a case, $\xi \in \mathrm{dom}(\Phi)$ thus there exists $(ax_i \rhd u_i) \in \Phi$ such $ax_i = \xi$ and $\xi\Phi\downarrow = u_i\downarrow$. But $\mathsf{Message}(u_i)$ holds thus by definition, $u_i\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Hence, the result holds.

Induction case: $|\xi| > 1$. In such a case, we have that $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$ with $\mathsf{f} \in \mathcal{F}_c \cup \mathcal{F}_d$. Assume first that $\mathsf{f} \in \mathcal{F}_c$. In such a case, $\xi_i\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ for every $i \in \{1 \ldots n\}$. Hence, we can apply our induction hypothesis on each ξ_i. This allows us to conclude.

Assume now that $\mathsf{f} \in \mathcal{F}_d$. By hypothesis, we have that $\xi\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and so f is reduced by the rewriting system. Let $\mathsf{f}(u_1, \ldots, u_n) \to u$ be the rewrite rule involved in $\mathsf{f}(\xi_1\Phi\downarrow, \ldots, \xi_n\Phi\downarrow) \to \xi\Phi\downarrow$. We distinguish two cases:

— *Case 1:* $\xi_1\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Since $vars(\{u_2, \ldots, u_n\}) \subseteq vars(u_1)$, we deduce that for every $i \in \{1, \ldots, n\}$, $\xi_i\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Hence, we easily conclude by applying our induction hypothesis.

— *Case 2:* $\xi_1\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Thanks to Corollary C.2, we deduce that either $\mathsf{root}(\xi_1) \in \mathcal{F}_c$ or $\mathsf{root}(\xi_1)$ is not reduced. Since $\xi \in \Pi_n$, we have that $\mathsf{root}(\xi_1) \notin \mathcal{F}_c$, hence $\mathsf{root}(\xi_1)$ is not reduced. It implies that $\mathsf{root}(\xi_1\Phi\downarrow) \in \mathcal{F}_d$. By definition of a rewriting rule, we know that $\mathsf{root}(u_1) \in \mathcal{F}_c$. This contradicts the fact that $\mathsf{f}(u_1, \ldots, u_n)$ and $\mathsf{f}(\xi_1\Phi\downarrow, \ldots, \xi_n\Phi\downarrow)$ are unifiable. Hence, this case is impossible. □

Lemma C.4. *Let Φ be a ground frame such that for all $(ax_i \rhd u_i) \in \Phi$, $\mathsf{Message}(u_i)$. Let ξ be a recipe such that $param(\xi) \subseteq \mathrm{dom}(\Phi)$ and $\xi \notin \Pi_n$. If for all $\mathsf{f}(\xi_1, \ldots, \xi_n) \in st(\xi)$, $\mathsf{f} \in \mathcal{F}_d$ and $\mathsf{root}(\xi_1) \in \mathcal{F}_c$ imply f is not reduced, then $\xi\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$.*

Proof. We prove this result by induction on the size of ξ.

Base case: $|\xi| = 1$. In such a case, we have that $\xi \in \mathrm{dom}(\Phi)$, and thus $\xi \in \Pi_n$. Hence, there is nothing to prove.

Induction step: $|\xi| > 1$. In such a case, we have that $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$ with $\mathsf{f} \in \mathcal{F}_c \cup \mathcal{F}_d$. We distinguish several cases:

— *Case 1,* $\mathsf{f} \in \mathcal{F}_c$*:* In such a case, we have that $\xi_i \notin \Pi_n$ for some $i \in \{1, \ldots, n\}$. Assume w.l.o.g. that $\xi_1 \notin \Pi_n$. Hence, by applying our induction hypothesis on ξ_1, we deduce that $\xi_1\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Since $\xi\Phi\downarrow = \mathsf{f}(\xi_1\Phi\downarrow, \ldots, \xi_n\Phi\downarrow)$, we have that $\xi\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. The same reasoning holds when $i \in \{2, \ldots, n\}$.

— *Case 2,* f *is not reduced:* In such a case, we have that $\mathsf{root}(\xi\Phi\downarrow) \in \mathcal{F}_d$. Hence, we have that $\xi\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. This allows us to conclude.

— *Case 3,* $\mathsf{f} \in \mathcal{F}_d$ *and* f *is reduced.* In such a case, by hypothesis, we have that $\mathsf{root}(\xi_1) \notin \mathcal{F}_c$. Let $\mathsf{f}(u_1, \ldots, u_n) \to u$ be the rewrite rule involved in $\mathsf{f}(\xi_1\Phi\downarrow, \ldots, \xi_n\Phi\downarrow) \to \xi\Phi\downarrow$. Since $\xi \notin \Pi_n$, we have that $\xi_i \notin \Pi_n$ for some $i \in \{1, \ldots, n\}$. We distinguish two cases:

1. If $\xi_1 \notin \Pi_n$ then by applying our induction hypothesis on ξ_1, we have that $\xi_1\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$.
2. Otherwise, if $\xi_i \notin \Pi_n$ for some $i \in \{2, \ldots, n\}$, then by applying our induction hypothesis on ξ_i, we have that $\xi_i\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$, and thus $\xi_1\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ since $vars(u_i) \subseteq vars(u_1)$ and u_1, u_i are constructor terms.

Hence, in both cases, we have that $\xi_1\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Thanks to Lemma C.3, we know that either $\mathsf{root}(\xi_1) \in \mathcal{F}_c$ or $\mathsf{root}(\xi_1)$ is not reduced. We have already shown that $\mathsf{root}(\xi_1) \notin \mathcal{F}_c$, thus we have that $\mathsf{root}(\xi_1)$ is not reduced, and thus $\mathsf{root}(\xi_1\Phi\downarrow) \in \mathcal{F}_d$. This contradicts the fact that $\mathsf{root}(\xi) \in \mathcal{F}_d$ and $\mathsf{root}(\xi)$ is reduced. Thus, this case is impossible. □

Lemma 6.6. *Let Φ be a ground frame such that for all $(ax_i \rhd u_i) \in \Phi$, $\mathsf{Message}(u_i)$. Let $\xi \in \mathcal{T}(\mathcal{F}, \mathcal{N} \cup \mathcal{AX})$. If $\mathsf{Message}(\xi\Phi)$ then there exists a ground recipe $\xi' \in \Pi_n$ such that $\xi\Phi\downarrow = \xi'\Phi\downarrow$, $\mathsf{Message}(\xi'\Phi)$ and $param(\xi') \subseteq param(\xi)$*

Proof. We prove this result by induction on the number of subterm ξ' of ξ such that ξ' is of the form $\mathsf{g}(\xi_1, \ldots, \xi_n)$, $\mathsf{g} \in \mathcal{F}_d$ and $\mathsf{root}(\xi_1) \in \mathcal{F}_c$. We denote this number $N(\xi)$

Base case $N(\xi) = 0$: In such a case, we have that $\xi \in \Pi_n$ and so the result holds.

Inductive step $N(\xi) > 0$: Let $p \in \mathcal{P}os(\xi)$ such that $\xi|_p = \mathsf{g}(\xi_1, \ldots, \xi_n)$, $\mathsf{g} \in \mathcal{F}_d$ and $\mathsf{root}(\xi_1) \in \mathcal{F}_c$. By hypothesis, we have $\mathsf{Message}(\xi\Phi)$ hence it implies that $\mathsf{g}(\xi_1, \ldots, \xi_n)\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and so g is reduced. Since $\mathsf{root}(\xi_1) \in \mathcal{F}_c$, we have $\xi_1 = \mathsf{f}(\alpha_1, \ldots, \alpha_m)$ for some $\mathsf{f} \in \mathcal{F}_c$ and some recipe $\alpha_1, \ldots, \alpha_m$. However, according to the rewriting system, g is reduced implies that $p|_\xi\Phi\downarrow = \alpha_1\Phi\downarrow$.

Let $\xi' = \xi[\alpha_1]_p$. We deduce that $\xi'\Phi\downarrow = \xi\Phi\downarrow$. Moreover, $\alpha \in st(\xi)$ and $\mathsf{Message}(\xi\Phi)$ implies that $\mathsf{Message}(\xi'\Phi)$ and $param(\xi') \subseteq param(\xi)$. At last, since $N(\xi') < N(\xi)$ then by inductive hypothesis on ξ', there exists $\xi'' \in \Pi_n$ such that $\xi''\Phi\downarrow = \xi'\Phi\downarrow = \xi\Phi\downarrow$, $\mathsf{Message}(\xi''\Phi)$ and $param(\xi'') \subseteq param(\xi') \subseteq param(\xi)$. □

Lemma C.5. *Let Φ, Φ' be ground frames of same domain and such that for all $(ax_i \rhd u_i) \in \Phi$ (resp. Φ'), $\mathsf{Message}(u_i)$. Assume that $\Phi \sim_2 \Phi'$. Let $\xi_1, \xi_2, \xi_3 \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$ such that $param(\{\xi_1, \xi_2, \xi_3\}) \subseteq \mathrm{dom}(\Phi)$ and for all $\mathsf{f}(\alpha_1, \ldots, \alpha_n) \in st(\xi_i)$ $(i = 1, 2, 3)$, $\mathsf{root}(\alpha_1) \in \mathcal{F}_c$ implies f is not reduced by Φ. The following properties hold:*

1. *$\mathsf{Message}(\xi_3\Phi)$ implies $\mathsf{Message}(\xi_3\Phi')$; and*
2. *if $\mathsf{Message}(\xi_1\Phi)$, $\mathsf{Message}(\xi_2\Phi)$, $\xi_1\Phi\downarrow = \xi_2\Phi\downarrow$ then $\xi_1\Phi'\downarrow = \xi_2\Phi'\downarrow$.*

Proof. Assume that $\mathsf{Message}(\xi_3\Phi)$ thus $\xi_3\Phi\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. In such a case, thanks to Lemma C.4, we have that $\xi_3 \in \Pi_n$. Since $\Phi \sim_2 \Phi'$, we easily conclude that $\mathsf{Message}(\xi_3\Phi')$.

Now, assume that $\mathsf{Message}(\xi_1\Phi)$, $\mathsf{Message}(\xi_2\Phi)$ and $\xi_1\Phi\downarrow = \xi_2\Phi\downarrow$. First, if ξ_1 and ξ_2 are in Π_n, then the result trivially holds. Otherwise, we prove that $\xi_1\Phi'\downarrow = \xi_2\Phi'\downarrow$ by induction on $\max(|\xi_1|, |\xi_2|)$.

Base case $\max(|\xi_1|, |\xi_2|) = 1$*:* In such a case, we have that $\xi_1, \xi_2 \in \mathrm{dom}(\Phi)$ and so $\xi_1, \xi_2 \in \Pi_n$. We already proved that the result holds.

Induction step $\max(|\xi_1|, |\xi_2|) > 1$*:* We assume w.l.o.g. that $\xi_1 \notin \Pi_n$. In such a case, thanks to Lemma C.4, we have that $\xi_1\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and since $\xi_1\Phi\downarrow = \xi_2\Phi\downarrow$, we also have that $\xi_2\Phi\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Thanks to Lemma C.3, we have that:

— either $\mathsf{root}(\xi_1)$ is not reduced or $\mathsf{root}(\xi_1) \in \mathcal{F}_c$; and

— either $\mathsf{root}(\xi_2)$ is not reduced or $\mathsf{root}(\xi_2) \in \mathcal{F}_c$.

Hence, we deduce that $\xi_1 = \mathsf{f}_1(\alpha_1, \ldots, \alpha_m)$ and $\xi_2 = \mathsf{f}_2(\beta_1, \ldots, \beta_k)$ for some $\mathsf{f}_1, \alpha_1, \ldots, \alpha_m$ and $\mathsf{f}_2, \beta_1, \ldots, \beta_k$. Moreover, we also deduce that

$$\xi_1\Phi\downarrow = \mathsf{f}_1(\alpha_1\Phi\downarrow, \ldots, \alpha_m\Phi\downarrow) \text{ and } \xi_2\Phi\downarrow = \mathsf{f}_2(\beta_1\Phi\downarrow, \ldots, \beta_k\Phi\downarrow).$$

Since $\xi_1\Phi\downarrow = \xi_2\Phi\downarrow$, it implies that $\mathsf{f}_1 = \mathsf{f}_2$, $k = m$ and $\alpha_i\Phi\downarrow = \beta_i\Phi\downarrow$ for every $i \in \{1, \ldots, m\}$. Moreover, $\mathsf{Message}(\xi_1\Phi)$ implies $\mathsf{Message}(\alpha_i\Phi)$ and $\mathsf{Message}(\xi_2\Phi)$ implies $\mathsf{Message}(\beta_i\Phi)$ for all $i \in \{1, \ldots, m\}$. By applying our induction hypothesis on α_i, β_i, we have that $\alpha_i\Phi'\downarrow = \beta_i\Phi'\downarrow$ for every $i \in \{1, \ldots, m\}$. This allows us to deduce that $\mathsf{f}_1(\alpha_1\Phi'\downarrow, \ldots, \alpha_k\Phi'\downarrow) = \mathsf{f}_2(\beta_1\Phi'\downarrow, \ldots, \beta_k\Phi'\downarrow)$, and thus $\xi_1\Phi'\downarrow = \xi_2\Phi'\downarrow$. □

Lemma 6.7. *Let Φ and Φ' be two ground frames such that for all $(ax_i \rhd u_i) \in \Phi$ (resp. Φ'), $\mathsf{Message}(u_i)$. $\Phi \sim_{\mathcal{T}(\mathcal{F}, \mathcal{AX})} \Phi'$ is equivalent to $\Phi \sim_{\Pi_n} \Phi'$.*

Proof. The right implication ($\Rightarrow$) is an easy case. Indeed, for all $\xi, \xi' \in \Pi_n$, by definition of Π_n, we have $\xi, \xi' \in \mathcal{T}(\mathcal{F}, \mathcal{AX})$. Hence, since $\Phi \sim_{\mathcal{T}(\mathcal{F}, \mathcal{AX})} \Phi'$, $\mathsf{Message}(\xi\Phi)$ is equivalent to $\mathsf{Message}(\xi\Phi')$. Furthermore, if $\mathsf{Message}(\xi\Phi)$ and $\mathsf{Message}(\xi'\Phi)$ then $\xi\Phi\downarrow = \xi'\Phi\downarrow$ is equivalent to $\xi\Phi'\downarrow = \xi'\Phi'\downarrow$. Hence $\Phi \sim_{\Pi_n} \Phi'$.

We now prove the left implication ($\Leftarrow$) of the equivalence. Let $\xi_1, \xi_2 \in \mathcal{T}(\mathcal{F}, \mathcal{A}\mathcal{X})$ such that $param(\xi_1, \xi_2) \subseteq \mathrm{dom}(\phi)$. We show that

— $\mathsf{Message}(\xi_1\Phi)$ implies $\mathsf{Message}(\xi_1\Phi')$; and

— if $\mathsf{Message}(\xi_1\Phi)$ and $\mathsf{Message}(\xi_2\Phi)$ then $\xi_1\Phi\downarrow = \xi_2\Phi\downarrow$ implies $\xi_1\Phi'\downarrow = \xi_2\Phi'\downarrow$.

Let n_1 (resp. n_2) be the number of subterms of the form $\mathsf{f}(\alpha_1, \ldots, \alpha_n) \in st(\xi_1)$ (resp. $st(\xi_2)$) such that $\mathsf{f} \in \mathcal{F}_d$, $\mathsf{root}(\alpha_1) \in \mathcal{F}_c$ and f is reduced by Φ. Let $n = \max(n_1, n_2)$ and $m = |\xi_1| + |\xi_2|$. We prove this result by induction on (n, m) with the lexicographic ordering. We assume that $\mathsf{Message}(\xi_1\Phi)$ and $\mathsf{Message}(\xi_2\Phi)$.

Base case : $n = 0$. In such a case, for all $\mathsf{f}(\alpha_1, \ldots, \alpha_n) \in st(\{\xi_1, \xi_2\})$, $\mathsf{root}(\alpha_1) \in \mathcal{F}_c$ and $\mathsf{f} \in \mathcal{F}_d$ imply f is not reduced. Hence, we apply Lemma C.5 and so the result holds.

Induction step: $n > 0$. There exists w.l.o.g $p \in \mathcal{P}os(\xi_1)$ such that $\xi_1|_p = \xi$ where $\xi = \mathsf{f}(\alpha_1, \ldots, \alpha_n)$, $\mathsf{f} \in \mathcal{F}_d$, $\mathsf{root}(\alpha_1) \in \mathcal{F}_c$ and f is reduced by Φ. Thus, there exists a rewriting rule that can be applied on the position p of ξ_1. We consider the different rewriting rules in turn:

— $\mathsf{f} = \mathsf{sdec}$ with $\xi = \mathsf{sdec}(\mathsf{senc}(\beta_1, \beta_2), \alpha_2)$ and $\alpha_2\Phi\downarrow = \beta_2\Phi\downarrow$. Moreover $\mathsf{Message}(\xi_1\Phi)$ implies $\mathsf{Message}(\alpha_2\Phi)$ and $\mathsf{Message}(\beta_2\Phi)$;

— $\mathsf{f} = \mathsf{adec}$ with $\xi = \mathsf{adec}(\mathsf{aenc}(\beta_1, \beta_2), \alpha_2)$ and $\beta_2\Phi\downarrow = (\mathsf{pk}(\alpha_2))\Phi\downarrow$. Moreover $\mathsf{Message}(\xi_1\Phi)$ implies $\mathsf{Message}(\mathsf{pk}(\alpha_2)\Phi)$ and $\mathsf{Message}(\beta_2\Phi)$;

— $\mathsf{f} = \mathsf{check}$ with $\xi = \mathsf{check}(\mathsf{sign}(\beta_1, \beta_2), \alpha_2)$ and $\alpha_2\Phi\downarrow = \mathsf{vk}(\beta_2)\Phi\downarrow$. Moreover $\mathsf{Message}(\xi_1\Phi)$ implies $\mathsf{Message}(\alpha_2\Phi)$ and $\mathsf{Message}(\mathsf{vk}(\beta_2)\Phi)$;

— $\mathsf{f} = \mathsf{proj}_1$ with $\xi = \mathsf{proj}_1(\langle\beta_1, \beta_2, \rangle\rangle$. Moreover, $\mathsf{Message}(\xi_1\Phi)$ implies $\mathsf{Message}(\beta_1\Phi)$;

— $\mathsf{f} = \mathsf{proj}_2$ with $\xi = \mathsf{proj}_2(\langle\beta_2, \beta_1, \rangle\rangle$. Moreover, $\mathsf{Message}(\xi_1\Phi)$ implies $\mathsf{Message}(\beta_1\Phi)$.

In the first three cases, we can apply our induction hypothesis in order to deduce that:

— $\mathsf{f} = \mathsf{sdec}$, $\alpha_2\Phi'\downarrow = \beta_2\Phi'\downarrow$; or

— $\mathsf{f} = \mathsf{adec}$ and $\beta_2\Phi'\downarrow = (\mathsf{pk}(\alpha_2))\Phi'\downarrow$; or

— $\mathsf{f} = \mathsf{check}$ and $\alpha_2\Phi'\downarrow = (\mathsf{vk}(\beta_2))\Phi'\downarrow$.

Hence, we can deduce that $\xi\Phi'\downarrow = \beta_1\Phi'\downarrow$. Note that this result trivially holds when $\mathsf{f} \in \{\mathsf{proj}_1, \mathsf{proj}_2\}$. Hence, we have that:

$$\xi_1\Phi\downarrow = (\xi_1[\beta_1]_p)\Phi\downarrow \text{ and } \xi_1[\Phi']\downarrow = (\xi_1[\beta_1]_p)\Phi'\downarrow$$

Moreover, since $\xi\Phi'\downarrow = \beta_1\Phi'\downarrow$, $\mathsf{Message}(\xi_1\Phi)$ and $\mathsf{Message}(\beta_1\Phi)$, we have that $\mathsf{Message}(\xi_1[\beta_1]_p\Phi)$.

We apply now our induction hypothesis on $\xi_1[\beta_1]_p$ and ξ_2. Note that even if the first measure does not decrease, we necessarily have that $|\xi_1[\beta_1]| < |\xi_1|$. This allows us to obtain that $\mathsf{Message}(\xi_1[\beta_1]_p\Phi')$. Moreover, if $\xi_1\Phi\downarrow = \xi_2\Phi\downarrow$, then we have that $\xi_1[\beta_1]_p\Phi\downarrow = \xi_2\Phi\downarrow$ and so by our induction hypothesis, we deduce that $\xi_1[\beta_1]_p\Phi'\downarrow = \xi_2\Phi'\downarrow$, and thus $\xi_1\Phi'\downarrow = \xi_2\Phi'\downarrow$. The other side of the static equivalence is proved symmetrically. This allows us to conclude. □

C.1.3 Constructor constraint system

Lemma C.6. *Let $s \in \mathcal{T}(\mathcal{F} \cup \mathcal{N}, \mathcal{X}^1)$. Assume that $s' \in st(s)$ such that $\mathsf{root}(s') \in \mathcal{F}_d$ and all strict subterms of s' is constructive. If for all fresh rewriting rule $\ell \to r$, ℓ is not unifiable with s', then for all mapping τ from $vars^1(s)$ to ground constructor terms, $\neg\mathsf{Message}(s\tau)$.*

Proof. Let τ a mapping from $vars^1(s)$ to ground constructor terms. Assume that $\mathsf{Message}(s\tau)$. By definition of $\mathsf{Message}(\cdot)$, we can deduce that $s'\tau\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. But $\mathsf{root}(s') \in \mathcal{F}_d$, thus $s'\tau$ has to be reduced. Let's assume that $s' = f(u_1, \ldots, u_n)$. We know that all strict subterms of s' only contain constructor; but τ is a mapping to ground constructor terms. Therefore, we have $u_i\tau\downarrow = u_i\tau$, for $i = 1 \ldots n$, which means that $s'\tau$ is reducible implies that s' is unifiable with a fresh rewriting rule. It implies a contradiction with our hypothesis and so $\neg\mathsf{Message}(s\tau)$. □

Lemma C.7. *Let $s, t \in \mathcal{T}(\mathcal{F} \cup \mathcal{N}, \mathcal{X}^1)$ such that $s \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} t$. For all mapping τ (resp. τ') from a set of variable including $vars^1(s)$ (resp. $vars^1(t)$) to ground constructor terms, we have:*

- $\mathsf{Message}(s\tau)$ *implies that there exists τ' such that $\tau = (\sigma\tau')|_{\mathrm{dom}(\tau)}$,* $\mathsf{Message}(t\tau')$ *and $s\tau\downarrow = t\tau'\downarrow$.*
- $\mathsf{Message}(t\tau')$ *implies that there exists τ such that $\tau = \sigma\tau'$,* $\mathsf{Message}(s\tau)$*, and $s\tau\downarrow = t\tau'\downarrow$*

Proof. Let $s, t \in \mathcal{T}(\mathcal{F} \cup \mathcal{N}, \mathcal{X}^1)$ such that $s \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} t$. Let τ (resp. τ') a mapping from $vars(s)$ (resp. $vars(t)$) to ground constructor terms. We prove the two result separately:

- If $\mathsf{Message}(s\tau)$: By hypothesis, we know that $s \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} t$ and so there exists a fresh rewriting rule $\ell \to r$ and a position p non variable of s such that:

  $$\sigma = \mathsf{mgu}(\ell, s_{|p}) \text{ and } t = s\sigma[r\sigma]_p$$

  By definition of a rewriting rule, we deduce that $r\sigma\downarrow = \ell\sigma\downarrow = s_{|p}\sigma\downarrow$. Thus, it implies that $t\downarrow = s\sigma[r\sigma]_p\downarrow = s\sigma[s_{|p}\sigma]_p\downarrow = s\sigma\downarrow$.

  But by Lemma C.6, we know that $\mathsf{Message}(s\tau)$ implies that there exists a fresh rewriting rule $\ell' \to r'$ such that ℓ' and $s_{|p}\tau$ are unifiable. Since we consider rewriting rule such that a destructor can be reduced by only one rewriting rule, we deduce that $\ell \to r$ is the same rule as $\ell' \to r'$ and so ℓ and $s_{|p}\tau$ are unifiable. Since $s_{|p}\tau$ is closed, there exists τ'' such that $\mathrm{dom}(\tau'') = vars^1(\ell)$ and $\ell\tau'' = s_{|p}\tau$. ℓ being fresh also implies that $\mathrm{dom}(\tau'') \cap \mathrm{dom}(\tau) = \emptyset$ and so $\ell\tau\tau' = s_{|p}\tau\tau'$. But $\sigma = \mathsf{mgu}(\ell, s_{|p})$, which means that there exists a mapping τ' such that $\tau\tau'' = \sigma\tau'$. Hence $\tau = (\sigma\tau')|_{\mathrm{dom}(\tau)}$.

  But, we already know that $t\downarrow = s\sigma\downarrow$, thus $t\tau'\downarrow = s\sigma\tau'\downarrow = s\tau\downarrow$.

  At last, we check that $\mathsf{Message}(t\tau')$. Let $t' \in st(t\tau')$. If $t' \in st(r\sigma\tau')$, then $t' \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ since we know that σ and τ' are both mapping to ground constructor terms and r is also a constructor term. If $t' \notin st(r\sigma\tau')$ then a simple induction on the position p' of t' on t allows us to conclude that $t'\downarrow = s_{|p'}\sigma\tau'\downarrow$ and so $t'\downarrow = s_{|p'}\tau\downarrow$. But $\mathsf{Message}(s\tau)$ implies $s_{|p'}\tau\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and so $t'\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$.

- As proved in the previous point, $s \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} t$ implies that $t\downarrow = s\sigma\downarrow$. Thus $t\tau'\downarrow = s\sigma\tau'\downarrow = s\tau\downarrow$ with $\tau = \sigma\tau'$.

  Let $s' \in st(s\tau)$, if $s' \in st(\ell\tau)$, then either s' is a strict subterm of $\ell\tau$ or $s' = \ell\tau$. In the first case, we know by the innermost strategy that all strict subterm are constructive terms which means that $\mathsf{Message}(s')$; in the second case, we know that $\ell\tau\downarrow = r\tau\downarrow$. Since $\mathsf{Message}(t\tau')$ and $\tau = \sigma\tau'$, we can conclude that $\ell\tau\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$.

  At last, if $s' \notin st(\ell\tau)$, then a simple induction on the position p' of s' on s allows us to conclude that $s'\downarrow = t_{|p'}\tau'\downarrow$ and so $\mathsf{Message}(s')$. □

Lemma 6.9. *Let E and E' be two conjunctions of equations between terms such that $E \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} E'$. For all substitutions τ on ground constructor terms,*

- $\tau \vDash_c E'$ *implies that $\sigma\tau \vDash_c E$*
- $\tau \vDash_c E$ *implies that there exists τ' such that $\tau = (\sigma\tau')|_{\mathrm{dom}(\tau)}$ and $\tau' \vDash_c E'$*

Proof. This Lemma is almost a direct implication of Lemma C.7. Let $E = \bigwedge_{i=1\ldots n} u_i \overset{?}{=} v_i$ be a conjunction of equations betweens terms. Let τ be a mapping from $vars^1(E)$ to ground constructor terms, we have $\tau \vDash_c E$ iff :

$$\forall i \in \{1 \ldots n\}, u_i\tau\downarrow = v_i\tau\downarrow \wedge \mathsf{Message}(u_i\tau) \wedge \mathsf{Message}(v_i\tau)$$

Assume now that $E \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} E'$ and $j \in \{1 \ldots n\}$ such that $u_j \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} t$.

— Assume that there exists a mapping τ from $vars^1(E)$ to ground constructor terms such that $\tau \vDash_c E$. It implies that $\mathsf{Message}(u_j\tau)$ and so by Lemma C.7, we know that there exists a mapping τ' from $vars(E')$ to ground constructor terms such that $\tau = (\sigma\tau')|_{\mathrm{dom}(\tau)}$, $u_j\tau\downarrow = t\tau'\downarrow$ and $\mathsf{Message}(t\tau')$. By hypothesis, we have $u_i\tau\downarrow = v_i\tau\downarrow$, for $i = 1 \dots n$ and so $(u_i\sigma)\tau'\downarrow = (v_i\sigma)\tau'\downarrow$. In the case $i = j$, $u_j\tau\downarrow = v_j\tau\downarrow$ implies $t\tau'\downarrow = (v_j\sigma)\tau'\downarrow$. At last, by hypothesis, we also have $\mathsf{Message}(u_i\tau)$ and $\mathsf{Message}(v_i\tau)$, for $i = \{1 \dots n\}$ which means that $\mathsf{Message}((u_i\sigma)\tau')$ and $\mathsf{Message}((v_i\sigma)\tau')$. In the case of $i = j$, we know that $\mathsf{Message}(t\tau')$ by Lemma C.7. We can conclude that $\tau' \vDash_c E'$.

— Assume that there exists a mapping τ' from $vars^1(E')$ to ground constructor terms such that $\tau' \vDash_c E'$. It implies that $\mathsf{Message}(t\tau')$ and so by Lemma C.7, we have that $\tau = \sigma\tau'$, $t\tau'\downarrow = u_j\tau\downarrow$ and $\mathsf{Message}(u_j\tau)$. But $\tau' \vDash E'$ also implies $u_j\tau\downarrow = t\tau'\downarrow = (v_j\sigma)\tau'\downarrow = v_j\tau\downarrow$ and $\mathsf{Message}(v_j\tau)$. At last, we know that $(u_i\sigma)\tau'\downarrow = (vi\sigma)\tau'\downarrow$, $\mathsf{Message}((u_i\sigma)\tau')$ and $\mathsf{Message}((v_i\sigma)\tau')$, for $i = 1 \dots n \neq j$ which allows us to conclude that $\tau \vDash_c E$. □

Lemma 6.10. *Let $\phi = \forall\tilde{x} \bigvee_{j=1}^m u_j \stackrel{?}{\neq} v_j$ and $\phi' = \forall\tilde{y} \bigvee_{i=1}^n u'_j \stackrel{?}{\neq} v'_j$ two formulas such that $\phi \rightarrowtail_{\mathcal{R}} \phi'$. For all substitutions τ of constructor terms, $\tau \vDash_c \phi$ if and only if $\tau \vDash_c \phi'$.*

Proof. Assume first that $\tau \nvDash_c \phi'$. There exists a substitution τ' on constructor terms such that $\mathrm{dom}(\tau') = \tilde{y}$ and $\tau\tau' \nvDash \bigvee_{i=1}^n u'_i \stackrel{?}{\neq} v'_i$. This implies that $\tau\tau' \vDash_c \sigma \wedge E'$. By hypothesis, we have $E \stackrel{\sigma}{\rightarrowtail}_{\mathcal{R}} E'$, hence $\mathrm{dom}(\sigma) \cap fvars(E') = \emptyset$. Thus $\tau\tau' \vDash_c \sigma E'$ implies that there exists τ'' such that $\tau\tau' = \sigma\tau''$ and $\tau'' \vDash_c E'$. Hence by lemma 6.9, we deduce that $\sigma\tau'' \vDash E$ and so $\sigma\tau'' \nvDash \bigvee_{j=1}^m u_j \stackrel{?}{\neq} v_j$. Since $\sigma\tau'' = \tau\tau'$, $\mathrm{dom}(\tau) = fvars(\phi)$ and $\tilde{x} \subseteq \mathrm{dom}(\tau')$, we conclude that $\tau \nvDash_c \phi$.

Assume now that $\tau \nvDash_c \phi$. There exists a substitution τ' on ground constructor terms such that $\mathrm{dom}(\tau') = \tilde{x}$ and $\tau\tau' \nvDash_c \bigvee_{j=1}^m u_j \stackrel{?}{\neq} v_j$. This implies that $\tau\tau' \vDash E$. By hypothesis, we have $E \stackrel{\sigma}{\rightarrowtail}_{\mathcal{R}} E'$, hence thanks to Lemma 6.9, we deduce that there exists τ'' such that $\tau\tau' = (\sigma\tau'')|_{\mathrm{dom}(\tau\tau')}$ and $\tau'' \vDash E'$. Hence, $\sigma\tau'' \vDash_c E' \wedge \sigma$. Since $\mathrm{dom}(\tau) = fvars(\phi')$, $\mathrm{dom}(\sigma\tau'') \supseteq \tilde{y} \cup fvars(\phi')$ and $\tau\tau' = (\sigma\tau'')|_{\mathrm{dom}(\tau\tau')}$, we deduce that there exists $\tau''' = (\sigma\tau'')|_{\tilde{y}}$ such that $\tau\tau''' \nvDash_c \bigvee_{i=1}^n u'_i \stackrel{?}{\neq} v'_i$. Hence we conclude that $\tau \nvDash_c \phi'$. □

Lemma 6.11. *Let Σ_1, Σ'_1 two sets of concrete constraint systems having the same structure. There exist Σ_2, Σ'_2 two sets of constructor constraint systems having the same structure such that:*

$$\Sigma_1 \approx_s^{\Pi_n} \Sigma'_1 \text{ if and only if } \Sigma_2 \approx_s \Sigma'_2$$

Proof. We first show how we obtain a constructor constraint system from a concrete constraint system. Then we will show that the solutions are preserved and finally we will prove the wanted equivalence.

Building the constructor constraint systems: Let $\mathcal{C} = (\mathcal{E}; \Phi; D; Eq)$ be a concrete constraint system where $\Phi = \{ax_1 \rhd t_1, \dots ax_n \rhd t_n\}$. Let $y_1, \dots, y_n$ fresh variables. Assume that $Eq = E_e \wedge D$ where $D = \bigwedge_{j=1}^m u_j \stackrel{?}{\neq} v_j$ and E_e is a conjunction of equations. We associate a constructor constraint system $\mathcal{C}'$ or $\bot$ to $\mathcal{C}$ as follows:

Let $E_0 = E_e \wedge \bigwedge_{i=1}^n y_i \stackrel{?}{=} t_i$. Assume that $E_0 \stackrel{\sigma_1}{\rightarrowtail}_{\mathcal{R}} E_1 \stackrel{\sigma_2}{\rightarrowtail}_{\mathcal{R}} \dots \stackrel{\sigma_\ell}{\rightarrowtail}_{\mathcal{R}} E_\ell$ and for all E', for all σ', $E_\ell \not\stackrel{\sigma'}{\rightarrowtail}_{\mathcal{R}} E'$. E_ℓ exists since the narrowing strictly decrease the number of destructor symbols. If E_ℓ contains a function symbol then we associate $\bot$ to $\mathcal{C}$.

Else, since applying a narrowing on a conjunction of equation does not change the number of conjunction, and for all $i \in \{1, \dots, n\}$, y_i are fresh variables, we deduce that $E_\ell = E'_e \wedge \bigwedge_{i=1}^n y_i \stackrel{?}{=} t'_i$ for some E'_e, $t'_1, \dots, t'_n$. Hence, we associate $\mathcal{C}' = (\Phi'; D'; Eq')$ to $\mathcal{C}$ where

1. $\alpha = \sigma_1 \dots \sigma_\ell$
2. $D' = D\alpha$

3. $\Phi' = \{ax_1 \rhd t'_1, \ldots, ax_n \rhd t'_n\}$
4. $Eq' = E'_e \wedge \bigwedge_{k \in I} \phi_k$

and $I \subseteq \{1 \ldots m\}$ satisfies:

— for all $k \in I$, $u_k\alpha \overset{?}{\neq} v_k\alpha \rightarrowtail_{\mathcal{R}} \ldots \rightarrowtail_{\mathcal{R}} \phi_k$, ϕ_k does not contain destructor symbols.

— for all $k \notin I$, for all ϕ, $u_k\alpha \overset{?}{\neq} v_k\alpha \rightarrowtail_{\mathcal{R}} \ldots \rightarrowtail_{\mathcal{R}} \phi$ implies ϕ contain at least a destructor symbol.

By construction, $\mathcal{C}'$ is a constructor constraint system. However, it remains to prove the following properties:

— for every $1 \leq k \leq n$, for every $z \in vars^1(t_k\alpha)$, there exists $(X, i \overset{?}{\vdash} x\alpha) \in D'$ such that $z \in vars^1(x\alpha)$ and $i < k$.

— for every the free variables z of Eq', there exists $(X, i \overset{?}{\vdash} x\alpha) \in D'$ such that $z \in vars^1(x\alpha)$.

Our rewriting system is subterm hence for all $s \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} t$, for all $x \in vars^1(t)$, either $x \in vars^1(s)$ and $x \notin \mathrm{dom}(\sigma)$; or else there exists $y \in vars^1(s)$ such that $x \in vars^1(y\sigma)$. Thus, by extension to conjunction of equations, we have for all $\bigwedge_{k=1}^m u_k \overset{?}{=} v_k \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} \bigwedge_{k=1}^m u'_k \overset{?}{=} v'_k$, for all $k \in \{1, \ldots, m\}$, for all $x \in vars^1(u'_k)$ (resp. $vars^1(v'_k)$), either $x \in vars^1(u_k)$ (resp. $vars^1(v_k)$) and $x \notin \mathrm{dom}(\sigma)$; or else there exists $y \in vars^1(u'_k)$ (resp. $vars^1(v_k)$) such that $x \in vars^1(y\sigma)$.

By applying a simple induction on ℓ, we can deduce that for all $i \in \{1, \ldots, n\}$, for all $x \in vars^1(t'_i)$, either (a) $x \in vars^1(t_i)$ and $x \notin \mathrm{dom}(\sigma)$; or else (b) there exists $y \in vars^1(t_i)$ such that $x \in vars^1(y\sigma_1 \ldots \sigma_\ell)$. In case (a), by the origination property of concrete constraint systems, we deduce that there exists $(X, j \overset{?}{\vdash} x) \in D$ such that $j < i$. Since $x\alpha = x$, we deduce that $(X, j \overset{?}{\vdash} x) \in D'$ and so the result holds. In case (b), by the origination property of concrete constraint systems, we deduce that there exists $(X, j \overset{?}{\vdash} y) \in D$ such that $j < i$ and so $(X, j \overset{?}{\vdash} y\alpha) \in D'$. Since $x \in vars^1(y\alpha)$, the results holds. The proof is similar for the variables in E'_e.

At last, consider two formulas ϕ and ϕ' such that $\phi \rightarrowtail_{\mathcal{R}} \phi'$. By definition, we know that there exists $E, E', \sigma, \tilde{x}, \tilde{y}$ such that: $\phi = \forall \tilde{x}.\neg E$, $\phi' = \forall \tilde{x}.\tilde{y}.\neg(\sigma \wedge E')$, $E \overset{\sigma}{\rightarrowtail}_{\mathcal{R}} E'$ and $\tilde{y} = vars^1(\sigma) \setminus (E)$. We show that $fvars(\phi') \subseteq fvars(\phi)$. Let $z \in fvars(\phi')$. Hence, $z \notin \tilde{x}$ and either (a) $z \in vars^1(\sigma)$ or (b) $z \in vars^1(E')$. In case (a), since $\tilde{y} = vars^1(\sigma) \setminus (E)$, we deduce that $z \in vars^1(E)$. Moreover, with $z \notin \tilde{x}$, we deduce that $z \in fvars(\phi)$. In case (b), we already proved that either $z \in vars^1(E)$ or else there exists $y \in vars^1(E)$ such that $x \in vars^1(y\sigma)$. We do the same reasoning as in case (a) and we conclude that $z \in fvars(\phi)$.

For all $k \in I$, by applying a simple induction on $u_k\alpha \overset{?}{\neq} v_k\alpha \rightarrowtail_{\mathcal{R}} \ldots \rightarrowtail_{\mathcal{R}} \phi_k$, we deduce that $fvars(\phi_k) \subseteq vars^1(u_k\alpha, v_k\alpha)$. Thus by applying the same reasoning as for the variables of t'_i, we can conclude that $fvars(Eq') \subseteq vars^1(D')$.

Preserving the solution: We say that $\mathsf{Sol}(\bot) = \emptyset$. Let's denote $\mathsf{Sol}_c^{\Pi_n}(\mathcal{C}) = \{(\sigma, \theta) \in \mathsf{Sol}_c(\mathcal{C}) \mid \forall X \in \mathrm{dom}(\theta), X\theta \in \Pi_n\}$. We show the two following properties:

1. for all $(\sigma, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C})$, there exists σ' such that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ and $\Phi\sigma\downarrow = \Phi'\sigma'$;
2. for all $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}')$, there exists σ such that $(\sigma, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C})$ and $\Phi\sigma\downarrow = \Phi'\sigma'$.

where Φ' is the frame of $\mathcal{C}'$.

First property: Let $(\sigma, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C})$. By definition, for all $i \in \{1, \ldots, n\}$, $\mathsf{Message}(t_i\sigma)$. Let's denote $\tau = \{y_1 \mapsto t_i\sigma; \ldots; y_n \mapsto t_n\sigma\} \cup \sigma$. We have that $\tau \vDash_c \bigwedge_{i=1}^n y_i \overset{?}{=} t_i$. Moreover, $(\sigma, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C})$ also implies that $\sigma \vDash_c Eq$ hence $\tau \vDash_c E_0$. At last, since $(\sigma, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C})$ implies that for all $(X, i \overset{?}{\vdash} x) \in D$, $\mathsf{Message}(x\sigma)$, we deduce that $(\tau\downarrow) \vDash_c E_0$ and $(\tau\downarrow)$ is a mapping from $vars^1(E_0)$ to ground constructor terms. Thanks to Lemma 6.9, we deduce with a simple induction on ℓ that there exists τ' such that $\tau\downarrow = (\sigma_1 \ldots \sigma_\ell\tau')_{\mathrm{dom}(\tau)}$ and $\tau' \vDash E_\ell$. Thanks to Lemma C.6, we deduce that E_ℓ does not contain a destructor and so $\mathcal{C}' \neq \bot$.

Let σ' be the substitution τ' restricted to the domain $\mathrm{dom}(\tau') \setminus \{y_1, \dots y_n\}$. We verify that $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$. $\tau\downarrow \vDash E_0$ and $\tau' \vDash_c E_\ell$ imply that for all $k \in \{1, \dots, n\}$, $t_k\tau\downarrow = y_k\tau\downarrow = y_k\alpha\tau' = y_k\tau' = t'_k\tau'$. Hence we deduce that $\Phi\tau\downarrow = \Phi'\tau'$. Since the variables $y_1, \dots, y_n$ are not in Φ nor Φ', $\Phi\sigma\downarrow = \Phi'\sigma'$. Furthermore, we know that for all $(X, i \overset{?}{\vdash} x) \in D$, $(X\theta)(\Phi\sigma) = x\sigma$. But $x\sigma\downarrow = x\tau\downarrow = x\sigma_1 \dots \sigma_\ell\tau' = x\alpha\tau' = x\alpha\sigma'$. Hence, we have that $(X\theta)(\Phi'\tau')\downarrow = (X\theta)(\Phi\sigma\downarrow)\downarrow = (X\theta)(\Phi\sigma)\downarrow = x\alpha\sigma'$. Moreover, we know that $\tau' \vDash_c E'_e$ and E'_e only contains constructor terms. Thus $\tau' \vDash E'_e$. Since $y \notin vars^1(E'_e)$, we deduce that $\sigma' \vDash E'_e$.

At last, let $k \in I$. Since $\tau \vDash_c u_k \overset{?}{\neq} v_k$ and for all $x \in \mathrm{dom}(\tau)$, $\mathsf{Message}(x\tau)$, we deduce that $\tau\downarrow \vDash_c u_k \overset{?}{\neq} v_k$ which implies that $\alpha\sigma' \vDash_c u_k \overset{?}{\neq} v_k$ and so $\sigma' \vDash_c u_k\alpha \overset{?}{\neq} v_k\alpha$. Thanks to Lemma 6.10, we deduce that $\sigma' \vDash_c \phi_k$. Since ϕ_k only contains constructor terms, we deduce that $\sigma' \vDash \phi_k$. This conclude the proof of $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$.

Second property: Let $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$. By definition, we have that σ' is a mapping to ground constructor terms. Let denote $\tau' = \{y_1 \mapsto t'_i\sigma'; \dots; y_n \mapsto t'_n\sigma'\} \cup \sigma'$. Hence $\tau' \vDash \wedge \bigwedge_{i=1}^n y_i \overset{?}{=} t'_i$. Moreover, $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ implies that $\sigma' \vDash E'_e \wedge \bigwedge_{k\in I} \phi_k$. Thus, we deduce that $\tau' \vDash E_\ell \wedge \bigwedge_{k\in I} \phi_k$. Since all terms in E_ℓ are constructor term, $\tau' \vDash_c E_\ell \wedge \bigwedge_{k\in I} \phi_k$.

By Lemma 6.10, $\tau' \vDash_c \bigwedge_{k\in I} \phi_k$ implies that $\tau' \vDash_c \bigwedge_{k\in I} u_k\alpha \overset{?}{\neq} v_k\alpha$. Let $k \notin I$, we know that for all ϕ, $u_k\alpha \overset{?}{\neq} v_k\alpha \rightarrowtail_{\mathcal{R}} \phi$ implies ϕ contains a destructor. Assume that no narrowing can be apply on ϕ (ϕ exists since the narrowing strictly reduces the number of destructors). Let $w \overset{?}{\neq} w' \in \phi$ such that w contains a destructor. By Lemma C.6, we deduce that for all mapping to ground constructor terms, and more specifically τ', $\neg\mathsf{Message}(w\tau')$. Hence, we deduce that $\tau' \vDash_c \bigwedge_{k=1}^m u_k\alpha \overset{?}{\neq} v_k\alpha$ and so $\alpha\tau' \vDash_c D$.

Thanks to Lemma 6.9, $\tau' \vDash_c E_\ell$ implies that $\sigma_1 \dots \sigma_\ell\tau' \vDash_c E_0$ and so $\alpha\tau' \vDash_c E_0$. Hence, $\alpha\tau' \vDash_c E_e$ and for all $i \in \{1, \dots, n\}$, $\mathsf{Message}(t_i\alpha\tau')$. Moreover, since $y_1, \dots y_n$ are not variable of α, we deduce for all $i \in \{1 \dots, n\}$, $t_i\alpha\tau'\downarrow = y_i\alpha\tau'\downarrow = y_i\tau'\downarrow = t'_i\tau'$.

Let σ be the substitution such that for all $(X, i \overset{?}{\vdash} x) \in D$, $(X\theta)(\Phi\sigma) = x\sigma$. We prove by induction on i that $\mathsf{Message}(x\sigma)$, $x\sigma\downarrow = x\alpha\tau'$ and $\mathsf{Message}(t_i\sigma)$.

The base case $i = 0$ is trivial since there is no deducible constraint with $i = 0$.

Inductive step $i > 0$: $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$ implies that $param(X\theta) \subseteq \{ax_1, \dots, ax_i\}$. But thanks to the origination property of a constraint system, we know that for all $j \leq i$, for all $y \in vars^1(t_j)$, there exists $(Y, k \overset{?}{\vdash} y) \in D$ such that $k < j \leq i$. Hence by our inductive hypothesis, $y\sigma\downarrow = y\alpha\tau'$ and $\mathsf{Message}(y\sigma)$. We already proved that $\mathsf{Message}(t_j\alpha\tau')$ and $t_j\alpha\tau'\downarrow = t'_j\tau'$ thus $\mathsf{Message}(t_j\sigma)$ and $t_j\sigma\downarrow = t'_j\tau'$. Therefore $x\sigma\downarrow = (X\theta)(\Phi\sigma)\downarrow = (X\theta)(\Phi\alpha\tau')\downarrow = (X\theta)(\Phi'\tau')\downarrow = x\alpha\tau'$. At last, since $X\theta \in \Pi_n$ and $x\sigma\downarrow = x\alpha\tau' \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ then by Lemma 6.5, we deduce that $\mathsf{Message}(x\sigma)$.

At last, since for all $x \in \mathrm{dom}(\sigma)$, $x\sigma\downarrow = x\alpha\tau'$ and $\mathsf{Message}(x\sigma)$ then $\alpha\tau' \vDash_c E_e$ implies $\sigma \vDash_c E_e$; and $\alpha\tau' \vDash_c D$ implies $\sigma \vDash_c D$. This conclude the proof of $(\sigma, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C})$.

The main result: Let Σ_1, Σ'_1 two sets of concrete constraint systems having the same structure. We build Σ_2, Σ'_2 the two sets of constructor constraint system associated to Σ_1, Σ'_1 respectively.

— Assume that $\Sigma_2 \approx_s \Sigma'_2$: Let $\mathcal{C}_1 \in \Sigma_1$, let $(\sigma_1, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C}_1)$. We have shown that there exists $\mathcal{C}_2 \in \Sigma_2$ and a substitution σ_2 such that $(\sigma_2, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$ and $\Phi_1\sigma_1\downarrow = \Phi_2\sigma_2$, where Φ_1 and Φ_2 are the respective frames of $\mathcal{C}_1$ and $\mathcal{C}_2$. Thus by our hypothesis, we deduce that there exists $\mathcal{C}'_2 \in \Sigma'_2$ and a substitution σ'_2 such that $(\sigma'_2, \theta) \in \mathsf{Sol}(\mathcal{C}'_2)$ and $\Phi_2\sigma_2 \sim \Phi'_2\sigma'_2$, where Φ'_2 is the frame of $\mathcal{C}'_2$.

We have shown that there exists $\mathcal{C}'_1 \in \Sigma'_1$ and a substitution σ'_1 such that $(\sigma'_1, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C}'_1)$ such that $\Phi'_1\sigma'_1\downarrow = \Phi'_2\sigma'_2$, where Φ'_1 is the frame of $\mathcal{C}'_1$.

It remains to show that $\Phi_1\sigma_1 \sim_{\Pi_n} \Phi'_1\sigma'_1$. First of all, $(\sigma_1, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C}_1)$ and $(\sigma'_1, \theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C}_2)$ implies that for all $(ax_i \rhd u) \in \Phi_1\sigma_1$ (resp. $\Phi'_1\sigma'_1$), $\mathsf{Message}(u)$. Let $\xi, \xi' \in \Pi_n$ such that $param(\{\xi, \xi'\}) \subseteq \mathrm{dom}(\Phi_1\sigma_1)$. Thanks to Lemma 6.5, $\mathsf{Message}(\xi\Phi_1\sigma_1)$ is equivalent to

$\xi\Phi_1\sigma_1\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. But we proved $\Phi_1\sigma_1\downarrow \sim \Phi'_1\sigma'_1\downarrow$ thus $\xi\Phi_1\sigma_1\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ is equivalent to $\xi\Phi'_1\sigma'_1\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. One again by Lemma 6.5, $\xi\Phi'_1\sigma'_1\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ is equivalent to $\mathsf{Message}(\xi\Phi'_1\sigma'_1)$ and so $\mathsf{Message}(\xi\Phi_1\sigma_1)$ is equivalent to $\mathsf{Message}(\xi\Phi'_1\sigma'_1)$. Assume now that $\mathsf{Message}(\xi\Phi_1\sigma_1)$ and $\mathsf{Message}(\xi'\Phi_1\sigma_1)$. It implies that $\xi\Phi_1\sigma_1\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ and $\xi'\Phi_1\sigma_1\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Thus, since $\Phi_1\sigma_1\downarrow \sim \Phi'_1\sigma'_1\downarrow$, we deduce that $\xi\Phi_1\sigma_1\downarrow = \xi'\Phi_1\sigma_1\downarrow$ is equivalent to $\xi\Phi'_1\sigma'_1\downarrow = \xi'\Phi'_1\sigma'_1\downarrow$. Therefore, we deduce that $\Phi_1\sigma_1 \sim_{\Pi_n} \Phi'_1\sigma'_1$.
This conclude the proof of $\Sigma_1 \approx_s^{\Pi_n} \Sigma'_1$.

— Assume that $\Sigma_1 \approx_s \Sigma'_1$: Let $\mathcal{C}_2 \in \Sigma_2$, let $(\sigma_2,\theta) \in \mathsf{Sol}(\mathcal{C}_2)$. We have shown that there exists $\mathcal{C}_1 \in \Sigma_1$ and a substitution σ_1 such that $(\sigma_1,\theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C}_2)$ and $\Phi_1\sigma_1\downarrow = \Phi_2\sigma_2$, where Φ_1 and Φ_2 are the respective frames of $\mathcal{C}_1$ and $\mathcal{C}_2$. Thus by our hypothesis, we deduce that there exists $\mathcal{C}'_1 \in \Sigma'_1$ and a substitution σ'_1 such that $(\sigma'_1,\theta) \in \mathsf{Sol}(\mathcal{C}'_1)$ and $\Phi_1\sigma_1 \sim_{\Pi_n} \Phi'_1\sigma'_1$, where Φ'_1 is the frame of $\mathcal{C}'_1$.
We have shown that there exists $\mathcal{C}'_2 \in \Sigma'_2$ and a substitution σ'_2 such that $(\sigma'_2,\theta) \in \mathsf{Sol}(\mathcal{C}'_2)$ such that $\Phi'_1\sigma'_1\downarrow = \Phi'_2\sigma'_2$, where Φ'_2 is the frame of $\mathcal{C}'_2$.
It remains to show that $\Phi_2\sigma_2 \sim \Phi'_2\sigma'_2$. First of all, $(\sigma_1,\theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C}_1)$ and $(\sigma'_1,\theta) \in \mathsf{Sol}_c^{\Pi_n}(\mathcal{C}_2)$ implies that for all $(ax_i \rhd u) \in \Phi_1\sigma_1$ (resp. $\Phi'_1\sigma'_1$), $\mathsf{Message}(u)$. Let $\xi,\xi' \in \Pi_n$ such that $param(\{\xi,\xi'\}) \subseteq \mathrm{dom}(\Phi_2\sigma_2)$. Thanks to Lemma 6.5, $\mathsf{Message}(\xi\Phi_1\sigma_1)$ is equivalent to $\xi(\Phi_1\sigma_1)\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Since $\Phi_1\sigma_1\downarrow = \Phi_2\sigma_2$, we deduce that $\mathsf{Message}(\xi\Phi_1\sigma_1)$ is equivalent to $\xi\Phi_2\sigma_2\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Similarly, $\mathsf{Message}(\xi\Phi'_1\sigma'_1)$ is equivalent to $\xi\Phi'_2\sigma'_2\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Since $\Phi_1\sigma_1 \sim_{\Pi_n} \Phi'_1\sigma'_1$, we deduce that $\xi\Phi_2\sigma_2\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ is equivalent to $\xi\Phi'_2\sigma'_2\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Assume know that $\xi\Phi_2\sigma_2\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ and $\xi\Phi'_2\sigma'_2\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. We already proved that is was equivalent to $\mathsf{Message}(\xi\Phi_1\sigma_1)$ and $\mathsf{Message}(\xi'\Phi_1\sigma_1)$. Moreover, we also proved that $\xi\Phi_2\sigma_2\downarrow = \xi\Phi_1\sigma_1\downarrow$ and $\xi'\Phi_2\sigma_2\downarrow = \xi'\Phi_1\sigma_1\downarrow$. Hence, thanks to $\Phi_1\sigma_1 \sim_{\Pi_n} \Phi'_1\sigma'_1$, we deduce that $\xi\Phi_2\sigma_2\downarrow = \xi'\Phi_2\sigma_2\downarrow$ is equivalent to $\xi\Phi'_1\sigma'_1\downarrow = \xi'\Phi'_1\sigma'_1\downarrow$. At last, since $\xi\Phi'_1\sigma'_1\downarrow = \xi\Phi'_2\sigma'_2\downarrow$ and $\xi'\Phi'_1\sigma'_1\downarrow = \xi'\Phi'_2\sigma'_2\downarrow$, we conclude that $\xi\Phi_2\sigma_2\downarrow = \xi'\Phi_2\sigma_2\downarrow$ is equivalent to $\xi\Phi'_2\sigma'_2\downarrow = \xi'\Phi'_2\sigma'_2\downarrow$. Therefore we deduce that $\Phi_2\sigma_2 \sim \Phi'_2\sigma'_2$.
This conclude the proof of $\Sigma_2 \approx_s^{\Pi_n} \Sigma'_2$. □

Note that

C.2 General invariants

C.2.1 Structure invariant

Lemma 8.1. *Let (M,M') be a pair of matrices of constraint systems such that M and M' have the same structure. Any internal (resp. external) application of a rule in Figure 7.1 and/or Figure 7.2 transforms the pair (M,M') on a pair (M_1,M'_1) (resp. two pairs (M_1,M'_1) and (M_2,M'_2)) of matrices having the same structure.*

Proof. Let $\mathcal{C}$ and $\mathcal{C}'$ be two constraint systems. Recall that the transformations rules Dest, Eq-left-left, Eq-left-right and Ded-st are applied internally whereas the rules Cons(X,f), Axiom(X,path) and Eq-right-right(X,ξ) are applied externally when $X \in S_2$ and internally otherwise (*i.e.* $X \notin S_2$).

Let Rule$(\tilde{p})$ be a rule and let $\mathcal{C}_1,\mathcal{C}_2$ (resp. $\mathcal{C}'_1,\mathcal{C}'_2$) be the two constraint systems obtained by application of R on $\mathcal{C}$ (resp. $\mathcal{C}'$). We assume that the application of Rule$(\tilde{p})$ is done simultaneously on $\mathcal{C}$ and $\mathcal{C}'$ (*i.e.* the possible fresh second order variables created are the same in both applications). We show that :

1. if $\mathcal{C}$ and $\mathcal{C}'$ have the same structure (resp. shape) then $\mathcal{C}_1,\mathcal{C}'_1$ and $\mathcal{C}_2,\mathcal{C}'_2$ have the same structure (resp. shape);
2. if Rule$(\tilde{p})$ is applied internally on $\mathcal{C}$, then $\mathcal{C},\mathcal{C}_1$ and $\mathcal{C}_2$ have the same shape.

With theses properties, the result directly holds. We prove theses properties by case analysis on the rule Rule$(\tilde{p})$. To simplify the notation, we will use the notation $S_1(\mathcal{C}), S_2(\mathcal{C}), \Phi(\mathcal{C}), \ldots$ while we refer to the different elements of a constraint system $\mathcal{C}$.

Case $\text{RULE}(\tilde{p}) = \text{CONS}(X, \mathsf{f})$: We assume that the application of $R(\tilde{p})$ was done simultaneously, thus if $\mathsf{ar}(\mathsf{f}) = n$, we denote by $X_1, \ldots, X_n$ the fresh second order variables used in $\mathcal{C}_1$ and $\mathcal{C}'_1$.

Assume first that $\mathcal{C}$ and $\mathcal{C}'$ have the same structure. Thus we have that $S_2(\mathcal{C}) = S_2(\mathcal{C}')$. By definition of $\text{CONS}(X, \mathsf{f})$, we have that $S_2(\mathcal{C}) = S_2(\mathcal{C}_2)$ and $S_2(\mathcal{C}') = S_2(\mathcal{C}'_2)$ which means that $S_2(\mathcal{C}_2) = S_2(\mathcal{C}'_2)$. Furthermore, If $X \in S_2(\mathcal{C}) = S_2(\mathcal{C}')$, then $S_2(\mathcal{C}_1) = S_2(\mathcal{C}) \cup \{X_1, \ldots, X_n\} = S_2(\mathcal{C}') \cup \{X_1, \ldots, X_n\} = S_2(\mathcal{C}'_1)$. Otherwise, we have that $S_2(\mathcal{C}_1) = S_2(\mathcal{C}) = S_2(\mathcal{C}') = S_2(\mathcal{C}'_1)$.

We also have that $Er(\mathcal{C}) = Er(\mathcal{C}')$. But by definition of $\text{CONS}(X, \mathsf{f})$, $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$ and $Er(\mathcal{C}'_1) = Er(\mathcal{C}') \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$. Thus we have that $Er(\mathcal{C}'_1) = Er(\mathcal{C}_1)$. Similarly, we have that $Er(\mathcal{C}_2) = Er(\mathcal{C}'_2)$.

Since the frame is not modified by the rule $\text{CONS}(X, \mathsf{f})$, then we have $\{(\xi, i) \mid \xi, i \rhd \Phi(\mathcal{C})\} = \{(\xi, i) \mid \xi, i \rhd \Phi(\mathcal{C}')\}$ implies that $\{(\xi, i) \mid \xi, i \rhd \Phi(\mathcal{C}_1)\} = \{(\xi, i) \mid \xi, i \rhd \Phi(\mathcal{C}'_1)\}$. The same holds for $\mathcal{C}_2$ and $\mathcal{C}'_2$.

At last, we have $D(\mathcal{C}_1) = D(\mathcal{C}) \cup \{X_1, i \stackrel{?}{\vdash} x_1, \ldots, X_n, i \stackrel{?}{\vdash} x_n\}$ and $D(\mathcal{C}'_1) = D(\mathcal{C}') \cup \{X_1, i \stackrel{?}{\vdash} x'_1, \ldots, X_n, i \stackrel{?}{\vdash} x'_n\}$ where $x_1, \ldots, x_n, x'_1, \ldots, x'_n$ are fresh variables. Thus, $\{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C})\} = \{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}')\}$ implies that $\{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}_1)\} = \{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}'_1)\}$. The case for $\mathcal{C}_2$ and $\mathcal{C}'_2$ is trivial since $D(\mathcal{C}_2) = D(\mathcal{C})$ and $D(\mathcal{C}'_2) = D(\mathcal{C}')$.

We can conclude that if $\mathcal{C}$ and $\mathcal{C}'$ have the same structure then $\mathcal{C}_1$ and $\mathcal{C}'_1$ (resp. $\mathcal{C}_2$ and $\mathcal{C}'_2$) also have the same structure. Similarly, we show that if $\mathcal{C}$ and $\mathcal{C}'$ have the same shape then $\mathcal{C}_1$ and $\mathcal{C}'_1$ (resp. $\mathcal{C}_2$ and $\mathcal{C}'_2$) also have the same shape.

At last, if $\text{CONS}(X, \mathsf{f})$ is applied internally (*i.e.* $X \notin S_2(\mathcal{C})$), then we have that $S_2(\mathcal{C}) = S_2(\mathcal{C}_1) = S_2(\mathcal{C}_2)$. Furthermore since $X_1, \ldots, X_n$ are fresh then $X_1, \ldots, X_n \notin S_2(\mathcal{C})$ and so $\mathcal{C}, \mathcal{C}_1$ and $\mathcal{C}_2$ have the same shape.

Case $\text{RULE}(\tilde{p}) = \text{AXIOM}(X, \mathsf{path})$: The rule $\text{AXIOM}(X, \mathsf{path})$ does not modify S_2 and either it does not modify D, in the case of $\mathcal{C}_2$ and $\mathcal{C}'_2$, or it removes this constraint in the case of $\mathcal{C}_1$, and $\mathcal{C}'_1$. Thus, we easily have that if $\mathcal{C}$ and $\mathcal{C}'$ have the same shape, then $\mathcal{C}_1$ and $\mathcal{C}'_1$ (resp. $\mathcal{C}_2$ and $\mathcal{C}'_2$) also have the same shape. Furthermore, in the case of an internal application of $\text{AXIOM}(X, \mathsf{path})$, we have that $X \notin S_2(\mathcal{C})$. If $(X, i \stackrel{?}{\vdash} u) \in D(\mathcal{C})$, then we have that $(X, i) \notin \{(Y, j) \mid (Y, j \stackrel{?}{\vdash} v \in D(\mathcal{C}) \wedge Y \in S_2(\mathcal{C})\}$. Thus we can also deduce that $\mathcal{C}, \mathcal{C}_1$ and $\mathcal{C}_2$ have the same shape when the rule $\text{AXIOM}(X, \mathsf{path})$ is applied internally.

At last, assume that $\mathcal{C}$ and $\mathcal{C}'$ have the same structure. Thus, we have that $\{(\xi, i) \mid \xi, i \rhd u \in \Phi(\mathcal{C})\} = \{(\xi, i) \mid \xi, i \rhd u \in \Phi(\mathcal{C}')\}$. But if there exists $(\xi, i \rhd u) \in \Phi(\mathcal{C})$ with $\mathsf{path}(\xi) = \mathsf{path}$ then it implies that there exists u' such that $(\xi, i \rhd u') \in \Phi(\mathcal{C}')$. With this, we can deduce that $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \stackrel{?}{=} \xi$ and $Er(\mathcal{C}'_1) = Er(\mathcal{C}') \wedge X \stackrel{?}{=} \xi$. Since $Er(\mathcal{C}) = Er(\mathcal{C}')$, we can conclude that $Er(\mathcal{C}_1) = Er(\mathcal{C}'_1)$ and so $\mathcal{C}_1$ and $\mathcal{C}'_1$ have the same structure. A similar reasoning allows us to conclude that $\mathcal{C}_2$ and $\mathcal{C}'_2$ have the same structure.

Case $\text{RULE}(\tilde{p}) = \text{DEST}(\xi, \ell \to r, i)$: The application of this rule on $\mathcal{C}$ (resp $\mathcal{C}'$) only adds a new non deducibility constraint on $\mathcal{C}_2$ (resp. $\mathcal{C}'_2$). Thus, we trivially have that $\mathcal{C}, \mathcal{C}_2$ have the same shape; and $\mathcal{C}, \mathcal{C}'$ have the same structure (resp. shape) implies that $\mathcal{C}_2, \mathcal{C}'_2$ have the same structure (resp. shape).

We consider now the constraint systems $\mathcal{C}_1$ and $\mathcal{C}'_1$. Since $\text{DEST}(\xi, \ell \to r, i)$ is applied simultaneously on $\mathcal{C}$ and $\mathcal{C}'$, then there exists $X_2, \ldots, X_n$ fresh second order variables such that $D(\mathcal{C}_1) = D(\mathcal{C}) \cup \{X_2, i \stackrel{?}{\vdash} u_2;\ \ldots;\ X_n, i \stackrel{?}{\vdash} u_n\}$ and $D(\mathcal{C}'_1) = D(\mathcal{C}') \cup \{X_2, i \stackrel{?}{\vdash} u'_2;\ \ldots;\ X_n, i \stackrel{?}{\vdash} u'_n\}$ where $f(u_1, \ldots u_n) \to w$ and $f(u'_1, \ldots, u'_n) \to w'$ are fresh renaming of $\ell \to r$. Thus we can deduce that $\{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C})\} = \{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}')\}$ implies that $\{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}_1)\} = \{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}'_1)\}$.

Furthermore, we also have that $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C}) \cup \{\mathsf{f}(\xi, X_2, \ldots, X_n), i \rhd w\}$ and $\Phi(\mathcal{C}'_1) = \Phi(\mathcal{C}') \cup \{\mathsf{f}(\xi, X_2, \ldots, X_n), i \rhd w'\}$. Thus, we deduce that $\{(\xi, i) \mid \xi, i \rhd u \in \Phi(\mathcal{C})\} = \{(\xi, i) \mid \xi, i \rhd u \in \Phi(\mathcal{C}')\}$ implies that $\{(\xi, i) \mid \xi, i \rhd u \in \Phi(\mathcal{C}_1)\} = \{(\xi, i) \mid \xi, i \rhd u \in \Phi(\mathcal{C}'_1)\}$.

Since $S_2(\mathcal{C}) = S_2(\mathcal{C}_1)$ and $S_2(\mathcal{C}') = S_2(\mathcal{C}'_1)$ by definition of DEST, we can conclude that if $\mathcal{C}, \mathcal{C}'$ have the same structure then $\mathcal{C}_1, \mathcal{C}'_1$ also have the same structure.

At last, the facts that $S_2(\mathcal{C}) = S_2(\mathcal{C}_1)$ and $X_2, \ldots, X_n$ are fresh variables (*i.e.* $X_2, \ldots, X_n \notin S_2(\mathcal{C})$) also imply that $\{(X,i) \mid X, i \overset{?}{\vdash} u \in D(\mathcal{C}) \wedge X \in S_2(\mathcal{C})\} = \{(X,i) \mid X, i \overset{?}{\vdash} u \in D(\mathcal{C}_1) \wedge X \in S_2(\mathcal{C}_1)\}$. Thus, we can conclude that $\mathcal{C}, \mathcal{C}_1$ have the same shape.

Case $\text{RULE}(\tilde{p}) = \text{EQ-LEFT-LEFT}(\xi_1, \xi_2)$: This rule only modifies $Eq(\mathcal{C})$ and $Eq(\mathcal{C}')$ thus the result trivially holds.

Case $\text{RULE}(\tilde{p}) = \text{EQ-LEFT-RIGHT}(\xi_1, X_2)$: The application of this rule modifies $Eq(\mathcal{C})$, $Eq(\mathcal{C}')$ and adds an frame element on $\mathsf{NoUse}(\mathcal{C})$ and $\mathsf{NoUse}(\mathcal{C}')$. Hence, it is easy to see that the rule does not modify the shape of the constrain systems. Assume now that $\mathcal{C}$ and $\mathcal{C}'$ have the same structure. It implies that $\{\xi, i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C})\} = \{\xi, i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}')\}$ and $\{\xi, i \mid (\xi, i \rhd u) \in \Phi(\mathcal{C})\} = \{\xi, i \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}')\}$. Hence, we have that there exists u, u' such that $(\xi_1, i_1 \rhd u) \in \Phi(\mathcal{C})$ and $(\xi_1, i_1 \rhd u') \in \Phi(\mathcal{C}')$. Thus, by definition of the rule, we have that $\mathsf{NoUse}(\mathcal{C}_1) = \mathsf{NoUse}(\mathcal{C}) \cup \{\xi_1, i_1 \rhd u\}$ and $\mathsf{NoUse}(\mathcal{C}'_1) = \mathsf{NoUse}(\mathcal{C}') \cup \{\xi_1, i_1 \rhd u'\}$. It implies that $\{\xi, i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}_1)\} = \{\xi, i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}'_1)\}$ and so $\mathcal{C}_1$, $\mathcal{C}'_1$ have the same structure.

Since only $Eq(\mathcal{C}_2)$, $Eq(\mathcal{C}'_2)$ are modified from $Eq(\mathcal{C})$, $Eq(\mathcal{C}')$, we easily deduce that $\mathcal{C}_2$,$\mathcal{C}_2$ have the same structure.

Case $\text{RULE}(\tilde{p}) = \text{EQ-RIGHT-RIGHT}(X, \xi)$: By the application of this rule on $\mathcal{C}$ and $\mathcal{C}'$, we have that $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \xi$ and $Er(\mathcal{C}'_1) = Er(\mathcal{C}') \wedge X \overset{?}{=} \xi$. Furthermore, the constraint $X, i \overset{?}{\vdash} u \in D(\mathcal{C})$ (resp. $X, i \overset{?}{\vdash} u' \in D(\mathcal{C}')$) is removed in $D(\mathcal{C}_1)$ (resp. $D(\mathcal{C}_2)$). Thus, we can easily see that if $\mathcal{C}, \mathcal{C}'$ have the same structure (resp. shape) then $\mathcal{C}_1, \mathcal{C}'_1$ and $\mathcal{C}_2, \mathcal{C}'_2$ also have the same structure (resp. shape).

In the case were $\text{EQ-RIGHT-RIGHT}(X, \xi)$ is applied internally, we have that $X \notin S_2(\mathcal{C})$ which means that $(X,i) \notin \{(Y,j) \mid Y, j \overset{?}{\vdash} v \in D(\mathcal{C}) \wedge Y \in S_2(\mathcal{C})\}$. Thus, we have that $\{(Y,j) \mid Y, j \overset{?}{\vdash} v \in D(\mathcal{C}) \wedge Y \in S_2(\mathcal{C})\} = \{(Y,j) \mid Y, j \overset{?}{\vdash} v \in D(\mathcal{C}_1) \wedge Y \in S_2(\mathcal{C}_1)\}$. This allows us to conclude that $\mathcal{C}$ and $\mathcal{C}_1$ have the same shape when $\text{EQ-RIGHT-RIGHT}(X, \xi)$ is applied internally. The result trivially holds for $\mathcal{C}_2$ and $\mathcal{C}'_2$.

Case $\text{RULE}(\tilde{p} = \text{DED-ST}(\xi, \mathsf{f})$: The application of this rule on $\mathcal{C}$ (resp $\mathcal{C}'$) only adds a new non deducibility constraint on $\mathcal{C}_2$ (resp. $\mathcal{C}'_2$). Thus, we trivially have that $\mathcal{C}, \mathcal{C}_2$ have the same shape; and $\mathcal{C}, \mathcal{C}'$ have the same structure (resp. shape) implies that $\mathcal{C}_2, \mathcal{C}'_2$ have the same structure (resp. shape).

We consider now the constraint systems $\mathcal{C}_1$ and $\mathcal{C}'_1$. Since $\text{DED-ST}(\xi, \mathsf{f})$ is applied simultaneously on $\mathcal{C}$ and $\mathcal{C}'$, then $X_1, \ldots, X_n$ fresh second order variables such that $D(\mathcal{C}_1) = D(\mathcal{C}) \cup \{X_1, i \overset{?}{\vdash} x_1;\ \ldots;\ X_n, i \overset{?}{\vdash} x_n\}$ and $D(\mathcal{C}'_1) = D(\mathcal{C}') \cup \{X_1, i \overset{?}{\vdash} x_1;\ \ldots;\ X_n, i \overset{?}{\vdash} x_n\}$ where $x_1, \ldots, x_n$ are fresh variables. Thus we can deduce that $\{(X,i) \mid X, i \overset{?}{\vdash} u \in D(\mathcal{C})\} = \{(X,i) \mid X, i \overset{?}{\vdash} u \in D(\mathcal{C}')\}$ implies that $\{(X,i) \mid X, i \overset{?}{\vdash} u \in D(\mathcal{C}_1)\} = \{(X,i) \mid X, i \overset{?}{\vdash} u \in D(\mathcal{C}'_1)\}$.

Since $S_2(\mathcal{C}) = S_2(\mathcal{C}_1)$ and $S_2(\mathcal{C}') = S_2(\mathcal{C}'_1)$ by definition of DED-ST, we can conclude that if $\mathcal{C}, \mathcal{C}'$ have the same structure then $\mathcal{C}_1, \mathcal{C}'_1$ also have the same structure.

At last, the facts that $S_2(\mathcal{C}) = S_2(\mathcal{C}_1)$ and $X_2, \ldots, X_n$ were fresh variables (*i.e.* $X_2, \ldots, X_n \notin S_2(\mathcal{C})$) also imply that $\{(X,i) \mid X, i \overset{?}{\vdash} u \in D(\mathcal{C}) \wedge X \in S_2(\mathcal{C})\} = \{(X,i) \mid X, i \overset{?}{\vdash} u \in D(\mathcal{C}_1) \wedge X \in S_2(\mathcal{C}_1)\}$. Thus, we can conclude that $\mathcal{C}, \mathcal{C}_1$ have the same shape. □

C.2.2 Well-formed invariant

Lemma C.8. *Let $\xi, \zeta \in \Pi_n$ and let $X \in \mathcal{X}^2$. Let θ and Θ be two substitutions such that $\theta = \{X \mapsto \zeta\}$ and $\Theta = \{X \mapsto \mathsf{path}(\zeta)\}$. The following property holds:*

$$\mathsf{path}(\xi\theta) = \mathsf{path}(\xi)\Theta$$

Proof. We prove this result by induction on $|\mathsf{path}(\xi)|$:

Base case $|\mathsf{path}(\xi)| = 1$*:* In such a case, either $\xi \in \mathcal{A}\mathcal{X}$ or $\xi \in \mathcal{X}^2$. If $\xi \in \mathcal{A}\mathcal{X}$, then $\xi\theta = \xi$, $\mathsf{path}(\xi) = \xi$ and $\xi\Theta = \xi$. Thus, we have $\mathsf{path}(\xi\theta) = \mathsf{path}(\xi)\Theta$. Otherwise $\xi \in \mathcal{X}^2$ and $\mathsf{path}(\xi) = \xi$. We distinguish two cases: $\xi = X$ or $\xi \neq X$. If $\xi = X$ then $\mathsf{path}(X\theta) = \mathsf{path}(\zeta) = X\Theta = \mathsf{path}(X)\Theta$. Else, we have that $\xi\theta = \xi$. But $\mathsf{path}(\xi) = \xi$ and so $\mathsf{path}(\xi)\Theta = \mathsf{path}(\xi)$. Thus we conclude $\mathsf{path}(\xi\theta) = \mathsf{path}(\xi)\Theta$.

Inductive step $|\mathsf{path}(\xi)| > 1$*:* Otherwise, there exist $\xi_1, \ldots, \xi_n \in \Pi_n$ and $\mathsf{f} \in \mathcal{F}$ such that $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$ and $\mathsf{path}(\xi) = \mathsf{f} \cdot \mathsf{path}(\xi_1)$. Thus $\xi\theta = \mathsf{f}(\xi_1\theta, \ldots, \xi_n\theta)$ and so $\mathsf{path}(\xi\theta) = \mathsf{f} \cdot \mathsf{path}(\xi_1\theta)$. Using our induction hypothesis on $\mathsf{path}(\xi_1)$, we have that $\mathsf{path}(\xi_1\theta) = \mathsf{path}(\xi_1)\Theta$ which allows us to conclude that $\mathsf{path}(\xi\theta) = \mathsf{f} \cdot \mathsf{path}(\xi_1)\Theta = \mathsf{path}(\xi)\Theta$. □

Lemma C.9. *Let $\mathcal{C}$ be a well formed constraint system and Φ its associated frame. Let $\xi \in \Pi_n$ such that $\mathsf{C}\lfloor\xi\rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$. Let θ, Θ be two substitutions such that $\theta = \{X \to \xi'\}$, $\Theta = \{X \to \mathsf{C}\lfloor\xi'\rfloor_\Phi\}$ and $\mathsf{C}\lfloor\xi'\rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$. The following property holds:*

$$\mathsf{C}\lfloor\xi\theta\rfloor_\Phi = \mathsf{C}\lfloor\xi\rfloor_\Phi\Theta$$

Proof. We prove this result by induction on $|\mathsf{C}\lfloor\xi\rfloor_\Phi|$:

Base case $|\mathsf{C}\lfloor\xi\rfloor_\Phi| = 1$*:* In such a case, either $\mathsf{C}\lfloor\xi\rfloor_\Phi \in (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X})$ or $\xi \in \mathcal{X}^2$. If $\xi \in \mathcal{X}^2 \setminus \{X\}$, then $\xi\theta = \xi$ and so $\mathsf{C}\lfloor\xi\theta\rfloor_\Phi = \xi = \mathsf{C}\lfloor\xi\rfloor_\Phi\Theta$. Thus the result holds. On the other hand, if $\xi = X$, then $\xi\theta = \xi'$ and $\mathsf{C}\lfloor\xi\rfloor_\Phi = X$ and so $\mathsf{C}\lfloor\xi\theta\rfloor_\Phi = \mathsf{C}\lfloor\xi'\rfloor_\Phi = X\Theta$.

Assume now that $\mathsf{C}\lfloor\xi\rfloor_\Phi \in (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X})$. In such case, there exists $(\zeta, i \rhd u) \in \Phi$ such that $\mathsf{path}(\zeta) = \mathsf{path}(\xi)$. But since $\mathsf{path}(\xi) \in (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X})$, we also have that $\mathsf{path}(\xi) = \mathsf{path}(\xi\theta)$, which means that $\mathsf{C}\lfloor\xi\theta\rfloor_\Phi = \mathsf{C}\lfloor\xi\rfloor_\Phi = \mathsf{C}\lfloor\xi\rfloor_\Phi\Theta$.

Inductive step $|\mathsf{C}\lfloor\xi\rfloor_\Phi| > 1$*:* By hypothesis, we know that $\mathsf{C}\lfloor\xi\rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$, thus, we have that $\mathsf{root}(\mathsf{C}\lfloor\xi\rfloor_\Phi) \in \mathcal{F}_c$ and so $\mathsf{root}(\xi) \in \mathcal{F}_c$. Assume that $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$. By definition of a context, we have: $\mathsf{C}\lfloor\xi\rfloor_\Phi = \mathsf{f}(\mathsf{C}\lfloor\xi_1\rfloor_\Phi, \ldots, \mathsf{C}\lfloor\xi_n\rfloor_\Phi)$. Thus we can applied our inductive hypothesis on ξ_i, for $i \in \{1 \ldots n\}$. This allows us to deduce that $\mathsf{C}\lfloor\xi_i\theta\rfloor_\Phi = \mathsf{C}\lfloor\xi_i\rfloor_\Phi\Theta$. Hence, we have that

$$\begin{array}{rcl}
\mathsf{C}\lfloor\xi\theta\rfloor_\Phi & = & \mathsf{C}\lfloor\mathsf{f}(\xi_1, \ldots, \xi_n)\theta\rfloor_\Phi \\
& = & \mathsf{f}(\mathsf{C}\lfloor\xi_1\theta\rfloor_\Phi, \ldots, \mathsf{C}\lfloor\xi_n\theta\rfloor_\Phi) \\
& = & \mathsf{f}(\mathsf{C}\lfloor\xi_1\rfloor_\Phi, \ldots, \mathsf{C}\lfloor\xi_n\rfloor_\Phi)\Theta \\
& = & \mathsf{C}\lfloor\mathsf{f}(\xi_1, \ldots, \xi_n)\rfloor_\Phi\Theta \\
& = & \mathsf{C}\lfloor\xi\rfloor_\Phi\Theta
\end{array}$$

This allows us to conclude. □

Lemma 8.2. *Any rule in Figure 7.1 and Figure 7.2 transforms a normalised well-formed constraint system into a pair of constraint systems that are also well-formed after normalisation. For the rule* Dest*, we assume that its application is not useless.*

Proof. Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a normalised well-formed constraint system and let $\textsc{Rule}(\tilde{p})$ be a rule. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the two constraint systems obtained by application of $\textsc{Rule}(\tilde{p})$ on $\mathcal{C}$. In the case where the normalisation of $\mathcal{C}_1$ and $\mathcal{C}_2$ is $\bot$, the result trivially holds thus we will assume that $\mathcal{C}_1{\downarrow} \neq \bot$ and $\mathcal{C}_2{\downarrow} \neq \bot$. We show the result by base analysis on the rule $\textsc{Rule}(\tilde{p})$.

Case $\textsc{Rule}(\tilde{p}) = \textsc{Cons}(X, \mathsf{f})$: The rule Cons only adds the inequation $\mathsf{root}(X) \neq \mathsf{f}$ on $Er(\mathcal{C}_2)$. Thus, we have that $\mathcal{C}_2{\downarrow} = \mathcal{C}_2$ and $\mathcal{C}_2$ trivially satisfies all the properties of Definition 8.2, except

for the property 6. But $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge \mathsf{root}(X) \neq \mathsf{f}$. Since $\mathcal{C}$ is a well-formed constraint system, then $Er(\mathcal{C})$ satisfies Property 6. Furthermore, since $\mathsf{f} \in \mathcal{F}_c$, then $\mathsf{root}(X) \neq \mathsf{f}$ also satisfies Property 6. We can conclude that $\mathcal{C}_2$ is a well-formed constraint system.

On the other hand, we have that $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$, $D(\mathcal{C}_1) = D(\mathcal{C}) \backslash \{X, i \stackrel{?}{\vdash} t\} \cup \{X_1, i \stackrel{?}{\vdash} x_1;\ \ldots;\ X_n, i \stackrel{?}{\vdash} x_n\}$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n)$ and $\Phi(\mathcal{C}) = \Phi(\mathcal{C}_1)$. Let $\sigma = \mathsf{mgu}(t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$ and $\theta = \mathsf{mgu}(X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n))$. By hypothesis, we know that $\mathcal{C}$ is normalised which means that $X \notin \mathrm{dom}(\mathsf{mgu}(Er(\mathcal{C})))$ and $vars^1(t) \cap \mathrm{dom}(\mathsf{mgu}(Eq(\mathcal{C}))) = \emptyset$. Since $X_1, \ldots, X_n, x_1, \ldots, x_n$ are fresh variables, we can deduce $\mathsf{mgu}(Er(\mathcal{C}_1)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$ and $\mathsf{mgu}(Eq(\mathcal{C}_1)) = \mathsf{mgu}(Eq(\mathcal{C}))\sigma$.

Furthermore, $\mathcal{C}$ normalised also implies that $\Phi(\mathcal{C})\mathsf{mgu}(Er(\mathcal{C}))\mathsf{mgu}(Eq(\mathcal{C})) = \Phi(\mathcal{C})$ and also that $D(\mathcal{C})\mathsf{mgu}(Eq(\mathcal{C})) = D(\mathcal{C})$. Thus, we can deduce that $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$ and $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C}_1)\sigma$. We now prove the different properties one by one.

Let $(\xi, j \rhd u) \in \Phi(\mathcal{C}_1{\downarrow})$. Since $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$, we know that there exists $(\xi', j \rhd u') \in \Phi(\mathcal{C})$ such that $\xi = \xi'\theta$ and $u = u'\sigma$.

1. this property is direct from Lemma C.8 and $\mathcal{C}$ being a well-formed constraint system.
2. this property is direct from Lemma C.8 and $\mathcal{C}$ being a well-formed constraint system.
3. We know that $\xi = \xi'\theta$ where $\theta = \mathsf{mgu}(X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n))$. But $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) = i = \mathsf{param}^{\mathcal{C}_1{\downarrow}}_{\mathsf{max}}(X_j)$, we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi) = \mathsf{param}^{\mathcal{C}_1{\downarrow}}_{\mathsf{max}}(\xi')$. Since $\mathcal{C}$ is well-formed, we have $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi) \leq j$ and so $\mathsf{param}^{\mathcal{C}_1{\downarrow}}_{\mathsf{max}}(\xi') \leq j$.
4. Let $Y \in vars^2(\xi)$ and $y \in vars^1(Y\mathsf{acc}^1(\mathcal{C}_1{\downarrow}))$. If $Y \in \{X_1, \ldots, X_n\}$ then there exists z such that $y \in vars^1(z\sigma)$ and $z \in vars^1(t)$. Thus $z \in vars^1(X\mathsf{acc}^1(\mathcal{C}))$. But $\xi = \xi'\theta$ thus it implies that $X \in vars^2(\xi')$. Since $\mathcal{C}$ is well formed, we deduce that there exists $(\zeta, k \rhd w) \in \Phi(\mathcal{C})$ such that $z \in vars^1(w)$ and $k \leq j$. We already showed that $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$ hence $y \in vars^1(w\sigma)$ with $(\zeta\theta, k \stackrel{?}{\vdash} w\sigma) \in \Phi(\mathcal{C}_1{\downarrow})$. The result holds
   If $Y \notin \{X_1, \ldots, X_n\}$, then $Y \in vars^2(\xi')$ and $Y \neq X$. Hence $Y \in vars^2(D(\mathcal{C}))$. Since $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C}_1)\sigma$, $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$ and $\mathcal{C}$ is well-formed, we deduce the result.
5. Let λ be a ground substitution such that for all $Y \in vars^2(\xi)$, $(Y\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda{\downarrow} = v\lambda$ where $(Y, k \stackrel{?}{\vdash} v) \in D(\mathcal{C}_1{\downarrow})$. Let λ' the substitution such that $\lambda' = \theta\sigma\lambda$. We show that for all $Z \in vars^2(\xi')$, $(Z\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = w\lambda'$ where $(Z, k \stackrel{?}{\vdash} w) \in D(\mathcal{C})$. Let $Z \in vars^2(\xi')$. Since $\xi = \xi'\theta$, we have to distinguish two cases :
   — Either $Z = X$: In this case, we have that $Z\lambda' = X\theta\sigma\lambda = \mathsf{f}(X_1, \ldots, X_n)\lambda$. But in such a case, we have that $X_1, \ldots, X_n \in vars^2(\xi)$ and so by hypothesis, we have that $(X_k\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda{\downarrow} = x_k\lambda$, for all $k \in \{1, \ldots, n\}$. Since $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$, we can deduce that $(X_k\lambda)\Phi(\mathcal{C})\lambda'{\downarrow} = x_k\lambda = x_k\lambda'$. Thus $(Z\lambda')(\Phi(\mathcal{C})\lambda'){\downarrow} = \mathsf{f}(x_1, \ldots, x_n)\lambda' = t\lambda'$.
   — Or $Z \in vars^2(\xi) \backslash \{X_1, \ldots, X_n\}$: In such a case, we have that $Z\theta\sigma = Z$ and so $Z\lambda' = Z\lambda$. Furthermore, we know that $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$ and so $\Phi(\mathcal{C}_1{\downarrow})\lambda = \Phi(\mathcal{C})\lambda'$. Thus, we have that $(Z\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = (Z\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda{\downarrow}$. By hypothesis, there exists $Z, k \stackrel{?}{\vdash} v \in D(\mathcal{C}_1{\downarrow})$ such that $(Z\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda{\downarrow} = v\lambda$. But $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C}_1)\sigma$ and $D(\mathcal{C}_1) = D(\mathcal{C}) \backslash \{X, i \stackrel{?}{\vdash} t\} \cup \{X_1, i \stackrel{?}{\vdash} x_1;\ \ldots;\ X_n, i \stackrel{?}{\vdash} x_n\}$. Thus there exists $Z, k \stackrel{?}{\vdash} v' \in D(\mathcal{C})$ such that $v = v'\sigma$ and so $v\lambda = v'\lambda'$. We can conclude that $(Z\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = v'\lambda'$ with $Z, k \stackrel{?}{\vdash} v' \in D(\mathcal{C})$.

   By hypothesis, we know that $\mathcal{C}$ is a well-formed constraint system and so we deduce that $(\xi'\lambda')(\Phi(\mathcal{C})\lambda'){\downarrow} = u'\lambda'$. But $\xi'\lambda' = \xi\lambda$, $u'\lambda' = u\lambda$ and $\Phi(\mathcal{C})\lambda' = \Phi(\mathcal{C}_1{\downarrow})\lambda$. Thus we conclude that $(\xi\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda = u\lambda$.
6. Let $(\zeta_1 \stackrel{?}{\neq} \zeta_2) \in Er(\mathcal{C}_1{\downarrow})$. By the normalisation, we know that there exists $(\zeta_1' \stackrel{?}{\neq} \zeta_2') \in Er(\mathcal{C}_1)$ such that $\zeta_1'\theta = \zeta_1$ and $\zeta_2'\theta = \zeta_2$. Moreover, by the rule Cons, the sets of inequations of

$Er(\mathcal{C})$ and $Er(\mathcal{C}_1)$ are the same thus $(\zeta_1' \stackrel{?}{\neq} \zeta_2') \in Er(\mathcal{C})$. Since $\mathcal{C}$ is well formed, we deduce that $\zeta_1' \in \mathcal{X}^2$ and there exists $k \in \mathbb{N}$ and a term u such that $(\zeta_2', k \rhd u) \in \Phi(\mathcal{C})$. Since $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$, we deduce that $(\zeta_2, k \rhd u\sigma) \in \Phi(\mathcal{C}_1\downarrow)$.
But $\mathsf{path}(\zeta_2')$ exists thanks to $\mathcal{C}$ being well-formed (Property 1) thus $\mathsf{root}(()\zeta_2') \notin \mathcal{F}_c$. Since $\mathsf{f} \in \mathcal{F}_c$ and $\zeta_1 = \zeta_1'\theta$, we deduce that $\zeta_1' \neq X$ otherwise the inequation would have disappeared by the normalisation rule (Nneq2). Thus $\zeta_1 \in \mathcal{X}^2$ and so the result holds.

Let $(\mathsf{root}(\zeta) \stackrel{?}{\neq} \mathsf{g}) \in Er(\mathcal{C}_1\downarrow)$. Thanks to the normalisation, we know that $\zeta \in \mathcal{X}^2$ otherwise the inequation would have disappeared by the normalisation rules (Ntop2). Hence the result holds.

7. Let $Y \in vars^2(\mathcal{C}_1\downarrow)$. If $Y \in \{X_1, \ldots, X_n\}$, then since $\theta = \{X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)\}$ and Y is fresh, we have that $Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = Y$ and so we deduce that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = Y \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$. Otherwise, we have that $Y \in vars^2(\mathcal{C})$. Note that $\mathcal{C}$ is well-formed, thus we deduce that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$. But the path of any recipe in $\Phi(\mathcal{C})$ is closed which means that any context with $\Phi(\mathcal{C}_1\downarrow)$ and $\Phi(\mathcal{C})$ are the same. More specifically, we have $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C}_1\downarrow)}$. Note that $\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$, $\theta = \{X \mapsto \mathsf{f}(X_1, \ldots, X_n)\}$ and $\mathsf{f} \in \mathcal{F}_c$. Thus by Lemma C.9, we have that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\theta \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$.
Let $\zeta \in st(Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))$ such that $\mathsf{path}(\zeta) \in \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}$. Since $\theta = \{X \mapsto \mathsf{f}(X_1, \ldots, X_n)\}$ with $\mathsf{f} \in \mathcal{F}_c$ and $\mathsf{mgu}(Er(\mathcal{C}_1)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$, then there exists $\zeta' \in st(Y\mathsf{mgu}(Er(\mathcal{C})))$ such that $\zeta = \zeta'\theta$. Since $\mathcal{C}$ is well-formed, then there exists k and u such that $(\zeta', k \rhd u) \in \Phi(\mathcal{C})$. But $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$. Hence $(\zeta, k \rhd u\sigma) \in \Phi(\mathcal{C}_1\downarrow)$. Hence the result holds.
8. Assume that $(\xi, j \rhd u) \in \mathsf{NoUse}(\mathcal{C}_1\downarrow)$. Since $\mathsf{NoUse}(\mathcal{C}_1\downarrow) = \mathsf{NoUse}(\mathcal{C})\theta\sigma$, we have that $(\xi', j \rhd u') \in \mathsf{NoUse}(\mathcal{C})$. Since $\mathcal{C}$ is a well-formed constraint system, we deduce that there exists $Y \in vars^2(\mathcal{C})$ such that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C}) = u'$ and $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}))) < j$. We have seen that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\theta$ when proving the previous item. Furthermore, we know that $D(\mathcal{C}_1\downarrow) = D(\mathcal{C}_1)\sigma$ and $D(\mathcal{C}_1) = D(\mathcal{C})\backslash\{X, i \stackrel{?}{\vdash} t\} \cup \{X_1, i \stackrel{?}{\vdash} x_1;\ \ldots;\ X_n, i \stackrel{?}{\vdash} x_n\}$. Thus, we deduce that $\mathsf{acc}^1(\mathcal{C})\sigma = \theta\mathsf{acc}^1(\mathcal{C}_1\downarrow)$. This allows us to conclude that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)}\mathsf{acc}^1(\mathcal{C}_1\downarrow) = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\theta\mathsf{acc}^1(\mathcal{C}_1\downarrow) = u'\sigma = u$.
At last, we already have that $Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = Y\mathsf{mgu}(Er(\mathcal{C}))\theta$. Thus, since $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) = \mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(\mathsf{f}(X_1, \ldots, X_n)) = i$, then $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}))) < j$ implies that $\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) < j$.
9. Similar to Property 7
10. Let $(Z, k \stackrel{?}{\vdash} u) \in D(\mathcal{C}_1\downarrow)$. Assume that $Z \notin S_2(\mathcal{C}_1)$ and let $x \in vars^1(u)$. If $X \in S_2(\mathcal{C})$ then it implies that there exists $(Z, k \stackrel{?}{\vdash} u') \in D(\mathcal{C})$ such that $u = u'\sigma$. There exists a variable z such that $x \in vars^1(z\sigma)$ and $z \in vars^1(u')$. Since $\mathcal{C}$ is well-formed, we deduce that there exists $(Y, p \stackrel{?}{\vdash} w') \in D(\mathcal{C})$ such that $z \in vars^1(w')$ and $p < k$. Thus $x \in vars^1(w'\sigma)$. If $Y \neq X$ then we deduce that $(Y, p \stackrel{?}{\vdash} w'\sigma) \in D(\mathcal{C}_1\downarrow)$ and so the result holds. If $Y = X$, then $t = w'$ and $p = i$. Since $\sigma = \mathsf{mgu}(t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$, we deduce that there exists $\ell \in \{1 \ldots, n\}$ such that $x \in vars^1(x_\ell\sigma)$. But $(X_\ell, i \stackrel{?}{\vdash} x_\ell) \in D(\mathcal{C}_1)$ with $k < i$ and so the result holds.
Assume now that $X \notin S_2(\mathcal{C})$. If $Z \notin \{X_1, \ldots, X_n\}$ then the proof is similar to the case where $X \in S_2(\mathcal{C})$ and so the result holds. If $X \in \{X_1, \ldots, X_n\}$ then there exists $\ell \in \{1, \ldots, n\}$ such that $x \in vars^1(x_\ell\sigma)$. Since $\sigma = \mathsf{mgu}(t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$, we deduce that $x \in vars^1(t\sigma)$. Hence, there exists z such that $x \in vars^1(z\sigma)$ and $z \in vars^1(t)$. Once again, thanks to $\mathcal{C}$ being-well formed, there exists $(Y, p \stackrel{?}{\vdash} w') \in D(\mathcal{C})$ such that $p < k$ and $z \in vars^1(w')$ and so $x \in vars^1(w'\sigma)$. But $(Y, p \stackrel{?}{\vdash} w'\sigma) \in D(\mathcal{C}_1\downarrow)$ thus the result holds.

Case $\textsc{Rule}(\tilde{p}) = \textsc{Axiom}(X, \mathsf{path})$: By definition of Axiom, we have $X, i \overset{?}{\vdash} u \in D(\mathcal{C})$ and $\xi, j \rhd v$ in $\Phi(\mathcal{C})$ such that $i \geq j$, $\mathsf{path}(\xi) = \mathsf{path}$ and $(\xi, j \rhd v) \notin \mathsf{NoUse}(\mathcal{C})$. The rule Axiom only adds the inequation $X \overset{?}{\neq} \xi$ on $Er(\mathcal{C}_2)$. Thus, we have that $\mathcal{C}_2{\downarrow} = \mathcal{C}_2$ and $\mathcal{C}_2$ trivially satisfies all the properties of Definition 8.2, except for the property 6. But $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge X \overset{?}{\neq} \xi$. Since $\mathcal{C}$ is a well-formed constraint system, then $Er(\mathcal{C})$ satisfies Property 6. Furthermore, $X \overset{?}{\neq} \xi$ also satisfies Property 6 by definition. We can conclude that $\mathcal{C}_2$ is a well-formed constraint system.

On the other hand, we have that $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \xi$, $D(\mathcal{C}_1) = D(\mathcal{C}) \backslash \{X, i \overset{?}{\vdash} u\}$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u \overset{?}{=} v$ and $\Phi(\mathcal{C}) = \Phi(\mathcal{C}_1)$. Let $\sigma = \mathsf{mgu}(u \overset{?}{=} v)$ and $\theta = \mathsf{mgu}(X \overset{?}{=} \xi)$. By hypothesis, $\mathcal{C}$ is a normalized constraint system which means that $(\{X\} \cup vars^2(\xi)) \cap \mathrm{dom}(\mathsf{mgu}(Er(\mathcal{C}))) = \emptyset$ and $vars^1(u, v) \cap \mathrm{dom}(\mathsf{mgu}(Eq(\mathcal{C}))) = \emptyset$. Thus, we can deduce $\mathsf{mgu}(Er(\mathcal{C}_1)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$ and $\mathsf{mgu}(Eq(\mathcal{C}_1)) = \mathsf{mgu}(Eq(\mathcal{C}))\sigma$. Since $\mathcal{C}$ is normalized, we have that $\Phi(\mathcal{C})\mathsf{mgu}(Er(\mathcal{C}))\mathsf{mgu}(Eq(\mathcal{C})) = \Phi(\mathcal{C})$ and $D(\mathcal{C})\mathsf{mgu}(Eq(\mathcal{C})) = D(\mathcal{C})$. Thus, we can deduce that $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$ and $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C})\sigma \backslash \{X, i \overset{?}{\vdash} u\sigma\}$. We now prove the different properties one by one.

Let $(\zeta, k \rhd w) \in \Phi(\mathcal{C}_1{\downarrow})$. Since $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$, we know that there exists $(\zeta', k \rhd w') \in \Phi(\mathcal{C})$ such that $\zeta = \zeta'\theta$ and $w = w'\sigma$.

1. this property is direct from Lemma C.8 and $\mathcal{C}$ being a well-formed constraint system.
2. this property is direct from Lemma C.8 and $\mathcal{C}$ being a well-formed constraint system.
3. We know that $\zeta = \zeta'\theta$ where $\theta = \{X \mapsto \xi\}$. But $\mathcal{C}$ is a well-formed constraint system hence we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi) \leq j$. Since $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) = i \geq j \geq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$, we deduce that $\mathsf{param}^{\mathcal{C}_1{\downarrow}}_{\mathsf{max}}(\zeta) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta') \leq j$. Hence the result holds.
4. Let $Y \in vars^2(\zeta)$ and $y \in vars^1(Y\mathsf{acc}^1(\mathcal{C}_1{\downarrow}))$. $Y \in vars^2(D(\mathcal{C}_1{\downarrow}))$ which means that $Y \in vars^2(D(\mathcal{C}))$. Since $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C}_1)\sigma$, $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$ and $\mathcal{C}$ is well-formed, we deduce the result.
5. Let λ be a substitution such that for all $Y \in vars^2(\zeta)$, $(Y\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda{\downarrow} = r\lambda$ where $(Y, \ell \overset{?}{\vdash} r) \in D(\mathcal{C}_1{\downarrow})$. Let λ' the substitution such that $\lambda' = \theta\sigma\lambda$. We show that for all $Y \in vars^2(\zeta')$, $(Y\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = r\lambda'$ where $(Y, \ell \overset{?}{\vdash} r) \in D(\mathcal{C})$. Let $Y \in vars^2(\zeta')$. Since $\zeta = \zeta'\theta$, we have to distinguish two cases :
   — Either $Y = X$: In this case, we have that $\xi \in st(\zeta)$. Thus, by hypothesis, we have that for all $Z \in vars^2(\xi)$, $(Z\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda{\downarrow} = t\lambda$, where $(Z, m \overset{?}{\vdash} t) \in D(\mathcal{C}_1{\downarrow})$. But $(Z, m \overset{?}{\vdash} t) \in D(\mathcal{C}_1{\downarrow})$ implies that there exist t' such that $(Z, m \overset{?}{\vdash} t') \in D(\mathcal{C})$ and $t = t'\sigma$. Since $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$, we can deduce that $(Z\lambda)\Phi(\mathcal{C})\lambda'{\downarrow} = t'\sigma\lambda = t'\lambda'$. At last, $\theta = \{X \mapsto \xi\}$ implies that $Z\theta = Z$ and so $Z\lambda = Z\lambda'$. Thus we have that $(Z\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = t\lambda'$. Since $\mathcal{C}$ is a well-formed constraint system, we can deduce that $(\xi\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = v\lambda'$. Since $X\theta = \xi$ and $\lambda' = \theta\sigma\lambda$, we have that $\xi\lambda' = X\lambda'$. With $u\sigma = v\sigma$, we can conclude that $(X\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = u\lambda'$.
   — Or $Y \in vars^2(\zeta)$: In such a case, we have that $Y\theta\sigma = Y$ and so $Y\lambda' = Y\lambda$. Furthermore, we know that $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\theta\sigma$ and so $\Phi(\mathcal{C}_1{\downarrow})\lambda = \Phi(\mathcal{C})\lambda'$. Thus, we have that $(Y\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = (Y\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda{\downarrow}$. By hypothesis, there exists $Y, \ell \overset{?}{\vdash} r \in D(\mathcal{C}_1{\downarrow})$ such that $(Y\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda{\downarrow} = r\lambda$. However, we have that $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C})\sigma \backslash \{X, i \overset{?}{\vdash} u\sigma\}$ and $Y \neq X$. Thus there exists $Y, \ell \overset{?}{\vdash} r' \in D(\mathcal{C})$ such that $r = r'\sigma$ and so $r\lambda = r'\lambda'$. We can conclude that $(Y\lambda')\Phi(\mathcal{C})\lambda'{\downarrow} = r'\lambda'$ with $Y, \ell \overset{?}{\vdash} r' \in D(\mathcal{C})$.

   By hypothesis, we know that $\mathcal{C}$ is well-formed and so $(\zeta'\lambda')(\Phi(\mathcal{C})\lambda'){\downarrow} = w'\lambda'$. We have that $\zeta'\lambda' = \zeta\lambda$, $w'\lambda' = w\lambda$ and $\Phi(\mathcal{C})\lambda' = \Phi(\mathcal{C}_1{\downarrow})\lambda$. Thus we conclude that $(\zeta\lambda)\Phi(\mathcal{C}_1{\downarrow})\lambda = w\lambda$.

6. Let $(\zeta_1 \overset{?}{\neq} \zeta_2) \in Er(\mathcal{C}_1\downarrow)$. By the normalisation, we know that there exists $(\zeta_1' \overset{?}{\neq} \zeta_2') \in Er(\mathcal{C}_1)$ such that $\zeta_1'\theta = \zeta_1$ and $\zeta_2'\theta = \zeta_2$. Moreover, by the rule Axiom, the sets of inequations of $Er(\mathcal{C})$ and $Er(\mathcal{C}_1)$ are the same thus $(\zeta_1' \overset{?}{\neq} \zeta_2') \in Er(\mathcal{C})$. Since $\mathcal{C}$ is well formed, we deduce that $\zeta_1' \in \mathcal{X}^2$ and there exists $k \in \mathbb{N}$ and a term u such that $(\zeta_2', k \rhd u) \in \Phi(\mathcal{C})$. Since $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$, we deduce that $(\zeta_2, k \rhd u\sigma) \in \Phi(\mathcal{C}_1\downarrow)$.

   But thanks to $\mathcal{C}$ being well-formed (Property 1), $\mathsf{path}(\zeta_2')$ and $\mathsf{path}(\xi)$ exists, are closed and if $\mathsf{path}(\zeta_2') = \mathsf{path}(\xi')$ then $\zeta_2' = \xi'$. But $\xi = \xi'\theta$, $\zeta_2 = \zeta_2'\theta$ $\mathsf{path}(\zeta_2') = \mathsf{path}(\zeta_2)$ and $\mathsf{path}(\xi) = \mathsf{path}(\xi')$. Thus we deduce that $\zeta_2 = \xi$ implies that $\zeta_1 \neq \xi$ otherwise $\mathcal{C}_1\downarrow = \bot$ by the normalisation rule (Nt2).

   Moreover, by definition of the path of recipe, $\mathsf{path}(\xi) \neq \mathsf{path}(\zeta_2)$ implies $\xi \neq \zeta_2$. Thus we deduce that $\zeta_1 \neq \xi$ otherwise the inequation would have disappeared by the normalisation rule (Nneq2). Thus $\zeta_1 \in \mathcal{X}^2$ and so the result holds.

7. Let $Y \in vars^2(\mathcal{C}_1\downarrow)$. We have $\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$ with $\theta = \{X \mapsto \xi\}$. Furthermore, we know that $\mathsf{path}(\xi) \in \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}$ since $\mathcal{C}$ is well-formed. Let $\Theta = \{X \mapsto \mathsf{path}(\xi)\}$. By Lemma C.9, we have that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C}_1\downarrow)}\Theta$. However, we have that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}$ and since $\mathcal{C}$ is well-formed, we also have that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$. Since $\mathsf{path}(\xi) \in \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}$, we conclude that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$.

   Let $\zeta \in st(Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))$ such that $\mathsf{path}(\zeta) \in \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}$. Since $\theta = \{X \mapsto \xi\}$ and $\mathsf{mgu}(Er(\mathcal{C}_1)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$, then (a) either $\zeta \in st(\xi)$ with $(\xi, i \rhd v) \in \Phi(\mathcal{C})$ and $(\xi, i \rhd v\sigma) \in \Phi(\mathcal{C}_1\downarrow)$ or (b) there exists $\zeta' \in st(Y\mathsf{mgu}(Er(\mathcal{C})))$ such that $\zeta = \zeta'\theta$. In both cases, since $\mathcal{C}$ is well-formed (Property 7), then there exists k and u such that $(\zeta', k \rhd u) \in \Phi(\mathcal{C})$. But $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$. Hence $(\zeta, k \rhd u\sigma) \in \Phi(\mathcal{C}_1\downarrow)$. Hence the result holds.

8. Assume that $(\zeta, k \rhd w) \in \mathsf{NoUse}(\mathcal{C}_1\downarrow)$. Since $\mathsf{NoUse}(\mathcal{C}_1\downarrow) = \mathsf{NoUse}(\mathcal{C})\theta\sigma$, we have that $(\zeta', k \rhd w') \in \mathsf{NoUse}(\mathcal{C})$. Since $\mathcal{C}$ is well-formed, we deduce that there exists $Y \in vars^2(\mathcal{C})$ such that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C}) = w'$ and $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}))) < k$. We have seen that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\Theta$ when proving the previous point. We have that $\mathsf{path}(\xi)\mathsf{acc}^1(\mathcal{C})\sigma = v\sigma = u\sigma = X\mathsf{acc}^1(\mathcal{C})\sigma$. Thus, we deduce that $\Theta\mathsf{acc}^1(\mathcal{C})\sigma = \mathsf{acc}^1(\mathcal{C})\sigma$. Furthermore, we know that $D(\mathcal{C}_1\downarrow) = D(\mathcal{C})\sigma\backslash\{X, i \overset{?}{\vdash} u\sigma\}$ and $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$ which means that $\mathsf{acc}^1(\mathcal{C})\sigma = \Theta\mathsf{acc}^1(\mathcal{C}_1\downarrow)$. This allows us to conclude that $w = w'\sigma = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C})\sigma = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)}\mathsf{acc}^1(\mathcal{C}_1\downarrow)$.

   Lastly, since $\mathcal{C}$ is a well formed constraint system, we know that for all $Z \in vars^2(\xi)$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z) \leq j \leq i = \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X)$. Since $Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = Y\mathsf{mgu}(Er(\mathcal{C}))\theta$, then $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}))) < k$ implies that $\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))) < k$.

9. Similar to Property 7

10. Let $(Z, k \overset{?}{\vdash} t) \in D(\mathcal{C}_1\downarrow)$. Assume that $Z \notin S_2(\mathcal{C}_1)$ and let $x \in vars^1(u)$. It implies that there exists $(Z, k \overset{?}{\vdash} t') \in D(\mathcal{C})$ such that $t = t'\sigma$. Hence, there exists a variable z such that $x \in vars^1(z\sigma)$ and $z \in vars^1(t')$. Since $\mathcal{C}$ is well-formed, we deduce that there exists $(Y, p \overset{?}{\vdash} w') \in D(\mathcal{C})$ such that $z \in vars^1(w')$ and $p < k$. Thus $x \in vars^1(w'\sigma)$. If $Y \neq X$ then we deduce that $(Y, p \overset{?}{\vdash} w'\sigma) \in D(\mathcal{C}_1\downarrow)$ and so the result holds. If $Y = X$, then $u = w'$ and $p = i$. Since $\sigma = \mathsf{mgu}(u \overset{?}{=} v)$, we deduce that $x \in vars^1(v\sigma)$. But $(\xi, j \overset{?}{\vdash} v\sigma) \in \Phi(\mathcal{C}_1\downarrow)$ with $j \leq i$. Thus by the origination property of a constraint system, there exists $(Y_2, p_2 \overset{?}{\vdash} w_2) \in D(\mathcal{C}_1\downarrow)$ such that $p_2 < j$ and $x \in vars^1(w_2)$. Since $p_2 < j \leq i < k$ then the result holds.

Case $\textsc{Rule}(\tilde{p}) = \textsc{Dest}(\xi, \ell \to r, i)$: By definition of Dest, we have that $(\xi, j \rhd v) \in \Phi(\mathcal{C})$ such that $j \leq i$, $(\xi, j \rhd v) \notin \mathsf{NoUse}(\mathcal{C})$, $X_2, \ldots, X_n$ fresh variables and $\mathsf{f}(u_1, \ldots, u_n) \to w$ a fresh

renaming of $\ell \to r$. The rule DEST only adds a non-deducibility constraint in $\mathcal{C}_2$. Hence, we have that $\mathcal{C}_2\downarrow = \mathcal{C}_2$ and since $\mathcal{C}$ is well formed, we easily deduce that $\mathcal{C}_2$ is also well-formed.

On the other hand, we have that $D(\mathcal{C}_1) = D(\mathcal{C})\backslash\{X_2, i \overset{?}{\vdash} u_2;\ \ldots;\ X_n, i \overset{?}{\vdash} u_n\}$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u_1 \overset{?}{=} v$ and $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C}) \cup \{\mathsf{f}(\xi, X_2, \ldots, X_n), i \rhd w\}$. By hypothesis, we know that $\mathcal{C}$ is normalised which means that $vars^1(v) \cap \mathrm{dom}(\mathsf{mgu}(Eq(\mathcal{C}))) = \emptyset$. Let $\sigma = \mathsf{mgu}(u_1 \overset{?}{=} v)$. Since $\mathsf{f}(u_1, \ldots, u_n) \to w$ is a fresh renaming of $\ell \to r$ and for all $k \in \{1, \ldots, k\}$, $vars^1(u_k) \subseteq vars^1(u_1)$, we can deduce $\mathsf{mgu}(Eq(\mathcal{C}_1)) = \mathsf{mgu}(Eq(\mathcal{C}))\sigma$. Furthermore, $\mathcal{C}$ normalised also implies that $\Phi(\mathcal{C})\mathsf{mgu}(Eq(\mathcal{C})) = \Phi(\mathcal{C})$ and $D(\mathcal{C})\mathsf{mgu}(Eq(\mathcal{C})) = D(\mathcal{C})$. Thus, we can deduce that $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\sigma \cup \{\mathsf{f}(\xi, X_2, \ldots, X_n) \rhd w\sigma\}$ and $D(\mathcal{C}_1\downarrow) = D(\mathcal{C})\sigma \cup \{X_2, i \overset{?}{\vdash} u_2\sigma;\ \ldots, X_n, i \overset{?}{\vdash} u_n\sigma\}$. We now prove the different properties one by one.

Let $(\zeta, k \rhd u) \in \Phi(\mathcal{C}_1\downarrow)$. Since $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\sigma \cup \{\mathsf{f}(\xi, X_2, \ldots, X_n) \rhd w\sigma\}$, we know that either there exists u' such that $(\zeta, k \rhd u') \in \Phi(\mathcal{C})$ with $u'\sigma = u$, or $(\zeta, k \rhd u) = (\mathsf{f}(\xi, X_2, \ldots, X_n), i \rhd w\sigma)$

1. If $(\zeta, k \rhd u) = (\mathsf{f}(\xi, X_2, \ldots, X_n), i \rhd w\sigma)$, we have that $\mathsf{path}(\zeta) = \mathsf{f} \cdot \mathsf{path}(\xi)$. Since $\mathcal{C}$ is well-formed, we have that $\mathsf{path}(\xi)$ exists and is closed. Thus $\mathsf{path}(\zeta)$ exists and is closed.
   Furthermore, since the rule DEST is never applied if its application is useless, then we deduce that the frame $\Phi(\mathcal{C})$ does not contain $\mathsf{f}(\xi, \zeta_2, \ldots, \zeta_n), j \rhd w'$ and $j \leq i$ and $\mathsf{root}(\ell) = \mathsf{f}$. Thus, with $\mathcal{C}$ being well formed hence satisfies Property 1, we can conclude that for all distinct frame elements $(\xi_1, i_1 \rhd u_1)$ and $(\xi_2, i_2 \rhd u_2)$ in $\Phi(\mathcal{C}_1\downarrow$, $\mathsf{path}(\xi_1) \neq \mathsf{path}(\xi_2)$.
2. If $(\zeta, k \rhd u) = (\mathsf{f}(\xi, X_2, \ldots, X_n), i \rhd w\sigma)$, since we know that $(\xi, j \rhd v\sigma) \in \Phi(\mathcal{C}_1\downarrow)$, the result trivially holds. Otherwise, we have that $(\zeta, k \rhd u') \in \Phi(\mathcal{C})$ and since $\mathcal{C}$ is well-formed, we easily conclude.
3. If $(\zeta, k \rhd u) = \mathsf{f}(\xi, X_2, \ldots, X_n)$ then $\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(\zeta) = \max(\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(\xi), i)$. But $\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(\xi) = \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$ thus since $\mathcal{C}$ is well-formed, we deduce that $\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(\xi) \leq j$. Since $j \leq i$ by definition, we conclude that $\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(\zeta) = i$ thus the result holds.
4. Let $Y \in vars^2(\zeta)$ and $y \in vars^1(Y\mathsf{acc}^1(\mathcal{C}_1\downarrow))$. If $Y \in \{X_2, \ldots, X_n\}$ then $y \in vars^1(v\sigma)$. Indeed, our rewrite rule satisfies $vars^1(u_2, \ldots, u_n) \subseteq vars^1(u_1)$ thus with $\sigma = \mathsf{mgu}(u_1, v)$ and $y \in vars^1(u_2\sigma, \ldots, u_n\sigma)$ the result holds.
   If $Y \notin \{X_2, \ldots, X_n\}$ then $Y \in vars^2(D(\mathcal{C}))$ and so there exists z such that $y \in vars^1(z\sigma)$ and $z \in vars^1(Y\mathsf{acc}^1(\mathcal{C}))$. Since $\mathcal{C}$ is well-formed, we deduce that there exists $(\beta, m \rhd t) \in \Phi(\mathcal{C})$ such that $m \leq k$ and $z \in vars^1(t)$. But $(\beta, m \rhd t\sigma) \in \Phi(\mathcal{C}_1\downarrow)$ hence $y \in vars^1(t\sigma)$. Thus the result holds.
5. Let λ be a substitution such that for all $Y \in vars^2(\zeta)$, we have that $(Y\lambda)\Phi(\mathcal{C}_1\downarrow)\lambda\downarrow = r\lambda$ where $(Y, k \overset{?}{\vdash} r) \in D(\mathcal{C}_1\downarrow)$. Let $\lambda' = \sigma\lambda$. Actually, $Y \in vars^2(\zeta)$ and $(\zeta, k \rhd u') \in \Phi(\mathcal{C})$ implies that there exists r' such that $(Y, k \overset{?}{\vdash} r') \in D(\mathcal{C})$ and $r'\sigma = r$. Since $Y\sigma\lambda = Y\lambda$ and $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\sigma \cup \{\mathsf{f}(\xi, X_2, \ldots, X_n) \rhd w\sigma\}$, we can deduce that $(Y\lambda')\Phi(\mathcal{C})\sigma\lambda\downarrow = (Y\lambda')\Phi(\mathcal{C})\lambda'\downarrow = r'\lambda'$. Since $\mathcal{C}$ is well-formed, we have that $(\zeta\lambda')(\Phi(\mathcal{C})\lambda')\downarrow = u'\lambda'$. Since $\zeta\sigma\lambda = \zeta\lambda$ and $u'\lambda' = u\lambda$, we can deduce that $(\zeta\lambda)\Phi(\mathcal{C}_1\downarrow)\lambda\downarrow = u\lambda$.
6. $Er(\mathcal{C}_1\downarrow) = Er(\mathcal{C})$ thus the result trivially holds.
7. Let $X \in vars^2(\mathcal{C}_1\downarrow)$. We have that $Er(\mathcal{C}_1\downarrow) = Er(\mathcal{C})$. Let $\theta = \mathsf{mgu}(Er(\mathcal{C}))$. We show that $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_1\downarrow)} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$. If $X \in \{X_2, \ldots, X_n\}$, then the result trivially holds. Otherwise, we have that $X \in vars^2(\mathcal{C})$. Since $\mathcal{C}$ is a well-formed constraint system, we know that $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$. Thus for all $\Xi \in st(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})})$, $\mathsf{root}(\Xi) \notin \mathcal{F}_d$. Since $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\sigma \cup \{\mathsf{f}(\xi, X_2, \ldots, X_n) \rhd w\sigma\}$ and $\mathsf{f} \in \mathcal{F}_d$, we conclude that $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$.
   Let $\beta \in st(X\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)))$ and $\mathsf{path}(\beta) \in \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}$. Since $\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = \mathsf{mgu}(Er(\mathcal{C}))$, then $\beta \in st(X\mathsf{mgu}(Er(\mathcal{C})))$. Thanks to $\mathcal{C}$ being well-formed, we deduce that there exists k, u such that $(\beta, k \rhd u) \in \Phi(\mathcal{C})$ and so $(\beta, k \rhd u\sigma) \in \Phi(\mathcal{C}_1\downarrow)$. Hence the result holds.
8. Assume that $(\zeta, k \rhd u) \in \mathsf{NoUse}(\mathcal{C}_1\downarrow)$. Since $\mathsf{NoUse}(\mathcal{C}_1\downarrow) = \mathsf{NoUse}(\mathcal{C})\sigma$, we know that there exists u' such that $(\zeta, k \rhd u') \in \Phi(\mathcal{C})$ and $u'\sigma = u$. Since $\mathcal{C}$ is a well formed constraint

system, we know that there exists $X \in vars^2(\mathcal{C})$ such that $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C}) = u'$ and $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\mathsf{mgu}(Er(\mathcal{C}))) < k$. Since $\mathsf{mgu}(Er(\mathcal{C})) = \mathsf{mgu}(Er(\mathcal{C}_1{\downarrow}))$ and $D(\mathcal{C})\sigma \subseteq D(\mathcal{C}_1{\downarrow})$, we deduce that $\mathsf{param}^{\mathcal{C}_1{\downarrow}}_{\mathsf{max}}(X\mathsf{mgu}(Er(\mathcal{C}_1{\downarrow}))) = \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\mathsf{mgu}(Er(\mathcal{C}))) < k$

In the previous point, we have shown that $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_1{\downarrow})}$. Furthermore, we have $\mathsf{acc}^1(\mathcal{C}_1{\downarrow}) = \mathsf{acc}^1(\mathcal{C})\sigma \cup \{\mathsf{f}\cdot\mathsf{path}(\xi) \mapsto w\sigma;\ X_2 \mapsto u_2\sigma;\ \ldots;\ X_n \mapsto u_n\sigma\}$. Actually, $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_1{\downarrow})}$ implies that $\mathsf{f}\cdot, \mathsf{path}(\xi), X_2, \ldots, X_n \notin st(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_1{\downarrow})})$. Hence, we have that $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_1{\downarrow})}\mathsf{acc}^1(\mathcal{C}_1{\downarrow}) = \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C})\sigma = u'\sigma = u$.

9. Similar to Property 7

10. Let $(Z, k \overset{?}{\vdash} u) \in D(\mathcal{C}_1{\downarrow})$. Assume that $Z \notin S_2(\mathcal{C}_1)$ and let $x \in vars^1(u)$. If $Z \notin \{X_2, \ldots, X_n\}$ then there exists $(Z, k \overset{?}{\vdash} u') \in D(\mathcal{C})$ such that $u = u'\sigma$. There exists a variable z such that $x \in vars^1(z\sigma)$ and $z \in vars^1(u')$. Since $\mathcal{C}$ is well-formed, we deduce that there exists $(Y, p \overset{?}{\vdash} t') \in D(\mathcal{C})$ such that $z \in vars^1(t')$ and $k < p$. Thus $x \in vars^1(t'\sigma)$. But $(Y, p \overset{?}{\vdash} t'\sigma) \in D(\mathcal{C}_1{\downarrow})$ hence the result holds.

    If $Z \in \{X_2, \ldots, X_n\}$ then there exists $\ell \in \{2, \ldots, n\}$ such that $x \in vars^1(x_\ell\sigma)$. Since $\sigma = \mathsf{mgu}(u_1 \overset{?}{=} v)$, $\mathsf{f}(u_1, \ldots, u_n) \to w$ is a renaming of $\ell \to r$ and $vars^1(u_\ell) \subseteq vars^1(u_1)$, we deduce that $x \in vars^1(v\sigma)$. By the origination property of a constraint system, $(\xi, j \rhd v\sigma) \in \Phi(\mathcal{C}_1{\downarrow})$ and $x \in vars^1(v\sigma)$ implies that there exists $(Y, p \overset{?}{\vdash} t) \in D(\mathcal{C}_1{\downarrow})$ such that $p < j$ and $x \in vars^1(t)$. But $j \leq i$ hence $p < i = k$ and so the result holds.

Case $\textsc{Rule}(\tilde{p}) = \textsc{Eq-left-left}(\xi_1, \xi_2)$: The rule Eq-left-left only adds the insequation $u_1 \overset{?}{\neq} u_2$ in $\mathcal{C}_2$. Since $\mathcal{C}$ is well-formed, we easily deduce that $\mathcal{C}_2{\downarrow}$ is also well-formed.

On the other hand, we have that $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$, $D(\mathcal{C}_1) = D(\mathcal{C})$, $\mathsf{NoUse}(\mathcal{C}_1) = \mathsf{NoUse}(\mathcal{C})$, $Er(\mathcal{C}) = Er(\mathcal{C}_1)$ and $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u_1 \overset{?}{=} u_2$. Since $\mathcal{C}$ is a well formed constraint, we have $vars^1(u_1, u_2) \cap \mathrm{dom}(\mathsf{mgu}(Eq(\mathcal{C}))) = \emptyset$. Let $\sigma = \mathsf{mgu}(u_1 \overset{?}{=} u_2)$. We have that $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\sigma$, $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C})\sigma$ and $\mathsf{NoUse}(\mathcal{C}_1{\downarrow}) = \mathsf{NoUse}(\mathcal{C})\sigma$. We have also that $\mathsf{acc}^1(\mathcal{C}_1{\downarrow}) = \mathsf{acc}^1(\mathcal{C})\sigma$. Thus, we easily deduce that $\mathcal{C}_1{\downarrow}$ is a well-formed constraint system.

Case $\textsc{Rule}(\tilde{p}) = \textsc{Eq-left-right}(\xi_1, X_2)$: This case is similar to the rule Eq-left-left. Let $\sigma = \mathsf{mgu}(u_1 \overset{?}{=} u_2)$. We have that $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\sigma$, $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C})\sigma$ and $\mathsf{acc}^1(\mathcal{C}_1{\downarrow}) = \mathsf{acc}^1(\mathcal{C})\sigma$. On the other hand, we have that $\mathsf{NoUse}(\mathcal{C}_1{\downarrow}) = \mathsf{NoUse}(\mathcal{C})\sigma \cup \{\xi_1, i_1 \rhd u_1\sigma\}$. Thus $\mathcal{C}_1$ easily satisfies all the properties of Definition 8.2 except the property 8. We know that $u_1\sigma = u_2\sigma$ and $X_2, i_2 \overset{?}{\vdash} u_2\sigma \in D(\mathcal{C}_1{\downarrow})$. Furthermore, since $\mathcal{C}$ is normalised, we have that $X_2\mathsf{mgu}(Er(\mathcal{C})) = X_2\mathsf{mgu}(Er(\mathcal{C}_1{\downarrow})) = X_2$ and so $\mathsf{C}\lfloor X_2\mathsf{mgu}(Er(\mathcal{C}_1{\downarrow}))\rfloor_{\Phi(\mathcal{C}_1{\downarrow})} = X_2$. Lastly, since $X_2\mathsf{acc}^1(\mathcal{C}_1) = u_2\sigma$, we deduce that $\mathsf{C}\lfloor X_2\mathsf{mgu}(Er(\mathcal{C}_1{\downarrow}))\rfloor_{\Phi(\mathcal{C}_1{\downarrow})}\mathsf{acc}^1(\mathcal{C}_1{\downarrow}) = u_1\sigma$. Moreover, by definition of the rule Eq-left-right, $i_1 > i_2$ hence we deduce that $\mathsf{param}^{\mathcal{C}_1{\downarrow}}_{\mathsf{max}}(X_2) < i_1$.

For any frame element other than $(\xi_1, i_1 \rhd u_1\sigma)$ the result holds since $\mathcal{C}$ is well-formed, $\Phi(\mathcal{C}_1{\downarrow}) = \Phi(\mathcal{C})\sigma$, $D(\mathcal{C}_1{\downarrow}) = D(\mathcal{C})\sigma$ and $\mathsf{acc}^1(\mathcal{C}_1{\downarrow}) = \mathsf{acc}^1(\mathcal{C})\sigma$.

Case $\textsc{Rule}(\tilde{p}) = \textsc{Eq-right-right}(X, \xi)$: By definition of the rule Eq-right-right, we have that $X, i \overset{?}{\vdash} u \in D(\mathcal{C})$, $\xi \in \mathcal{T}(\mathcal{F}_c, vars^2(D(\mathcal{C}))$, and $\xi\mathsf{acc}^1(\mathcal{C}) = v$. The rule Eq-right-right only adds the insequation $u \overset{?}{\neq} v$ in $Eq(\mathcal{C}_2)$. Thus, we have that $\mathcal{C}_2{\downarrow} = \mathcal{C}_2$. Since $\mathcal{C}$ is well-formed, we also have that $\mathcal{C}_2{\downarrow}$ is a well-formed constraint system.

On the other hand, we have that $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \xi$, $D(\mathcal{C}_1) = D(\mathcal{C})\backslash\{X, i \overset{?}{\vdash} u\}$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u \overset{?}{=} v$ and $\Phi(\mathcal{C}) = \Phi(\mathcal{C}_1)$. By hypothesis, $\mathcal{C}$ is normalised which means that $(\{X\} \cup vars^2(\xi)) \cap \mathrm{dom}(\mathsf{mgu}(Er(\mathcal{C}))) = \emptyset$ and $vars^1(u, v) \cap \mathrm{dom}(\mathsf{mgu}(Eq(\mathcal{C}))) = \emptyset$. Let $\sigma = \mathsf{mgu}(u \overset{?}{=} v)$ and $\theta = \mathsf{mgu}(X \overset{?}{=} \xi)$. We deduce that $\mathsf{mgu}(Er(\mathcal{C}_1)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$ and $\mathsf{mgu}(Eq(\mathcal{C}_1)) = \mathsf{mgu}(Eq(\mathcal{C}))\sigma$. Furthermore, since $\mathcal{C}$ is normalized, we have that $\Phi(\mathcal{C})\mathsf{mgu}(Er(\mathcal{C}))\mathsf{mgu}(Eq(\mathcal{C})) =$

$\Phi(\mathcal{C})$ and $D(\mathcal{C})\mathsf{mgu}(Eq(\mathcal{C})) = D(\mathcal{C})$. Thus, we can deduce that $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$ and $D(\mathcal{C}_1\downarrow) = D(\mathcal{C})\sigma\backslash\{X, i \overset{?}{\vdash} u\sigma\}$. We now prove the different properties one by one.

Let $(\zeta, k \rhd w) \in \Phi(\mathcal{C}_1\downarrow)$. Since $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$, we know that there exists $(\zeta', k \rhd w') \in \Phi(\mathcal{C})$ such that $\zeta = \zeta'\theta$ and $w = w'\sigma$.

1. this property is direct from Lemma C.8 and $\mathcal{C}$ being a well-formed constraint system.
2. this property is direct from Lemma C.8 and $\mathcal{C}$ being a well-formed constraint system.
3. We know that $\zeta = \zeta'\theta$ where $\theta = \mathsf{mgu}(X \overset{?}{=} \xi)$. By definition of the rule Eq-right-right, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) = i$. Since $\mathcal{C}$ is a well-formed constraint system hence we deduce that $\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(\zeta'\theta) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta') \leq j$. Hence the result holds.
4. Let $Y \in vars^2(\zeta)$ and $y \in vars^1(Y\mathsf{acc}^1(\mathcal{C}_1\downarrow)))$. $Y \in vars^2(D(\mathcal{C}_1\downarrow))$ implies that $Y \in vars^2(D(\mathcal{C}))$. Since $D(\mathcal{C})\sigma \subseteq D(\mathcal{C}_1\downarrow)$, we deduce that there exists $z \in vars^1(Y\mathsf{acc}^1(\mathcal{C}))$ such that $y \in vars^1(z\sigma)$. Since $\mathcal{C}$ is well-formed, we deduce that there exists $(\beta, j \rhd v) \in \Phi(\mathcal{C})$ such that $j \leq k$ and $z \in vars^1(v)$. Hence $(\beta\theta, j \rhd v\sigma) \in \Phi(\mathcal{C}_1\downarrow)$ and $y \in vars^1(v\sigma)$. Thus the result holds.
5. Let λ be a substitution such that for all $Y \in vars^2(\zeta)$, $(Y\lambda)\Phi(\mathcal{C}_1\downarrow)\lambda\downarrow = r\lambda$ where $(Y, \ell \overset{?}{\vdash} r) \in D(\mathcal{C}_1\downarrow)$. Let λ' the substitution such that $\lambda' = \theta\sigma\lambda$. We show that for all $Y \in vars^2(\zeta')$, $(Y\lambda')\Phi(\mathcal{C})\lambda'\downarrow = r\lambda'$ where $(Y, \ell \overset{?}{\vdash} r) \in D(\mathcal{C})$. Let $Y \in vars^2(\zeta')$. Since $\zeta = \zeta'\theta$, we have to distinguish two cases :
   — Either $Y = X$: In this case, we have that $\xi \in st(\zeta)$. Thus, by hypothesis, we have that for all $Z \in vars^2(\xi)$, $(Z\lambda)\Phi(\mathcal{C}_1\downarrow)\lambda\downarrow = t\lambda$, where $(Z, m \overset{?}{\vdash} t) \in D(\mathcal{C}_1\downarrow)$. But $(Z, m \overset{?}{\vdash} t) \in D(\mathcal{C}_1\downarrow)$ implies that there exist t' such that $(Z, m \overset{?}{\vdash} t') \in D(\mathcal{C})$ and $t = t'\sigma$. Since $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$, we deduce that $(Z\lambda)\Phi(\mathcal{C})\lambda'\downarrow = t'\sigma\lambda = t'\lambda'$. Moreover, $\theta = \{X \mapsto \xi\}$ implies that $Z\theta = Z$ and so $Z\lambda = Z\lambda'$. Thus we have that $(Z\lambda')\Phi(\mathcal{C})\lambda'\downarrow = t'\lambda' = Z\mathsf{acc}^1(\mathcal{C})\lambda'$. Since $\xi \in \mathcal{T}(\mathcal{F}_c, vars^2(D(\mathcal{C})))$, we deduce that $(\xi\lambda')\Phi(\mathcal{C})\lambda'\downarrow = \xi\mathsf{acc}^1(\mathcal{C})\lambda' = v\lambda'$. Since $X\theta = \xi$ and $\lambda' = \theta\sigma\lambda$, we have that $\xi\lambda' = X\lambda'$. With $u\sigma = v\sigma$, we can conclude that $(X\lambda')\Phi(\mathcal{C})\lambda'\downarrow = u\lambda'$.
   — Or $Y \in vars^2(\zeta)$: In such a case, we have that $Y\theta\sigma = Y$ and so $Y\lambda' = Y\lambda$. Furthermore, we know that $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$ and so $\Phi(\mathcal{C}_1\downarrow)\lambda = \Phi(\mathcal{C})\lambda'$. Thus, we have that $(Y\lambda')\Phi(\mathcal{C})\lambda'\downarrow = (Y\lambda)\Phi(\mathcal{C}_1\downarrow)\lambda\downarrow$. By hypothesis, there exists $Y, \ell \overset{?}{\vdash} r \in D(\mathcal{C}_1\downarrow)$ such that $(Y\lambda)\Phi(\mathcal{C}_1\downarrow)\lambda\downarrow = r\lambda$. Since $D(\mathcal{C}_1\downarrow) = D(\mathcal{C})\sigma\backslash\{X, i \overset{?}{\vdash} u\sigma\}$ and $Y \neq X$, there exists $Y, \ell \overset{?}{\vdash} r' \in D(\mathcal{C})$ such that $r = r'\sigma$ and so $r\lambda = r'\lambda'$. We can conclude that $(Y\lambda')\Phi(\mathcal{C})\lambda'\downarrow = r'\lambda'$ with $Y, \ell \overset{?}{\vdash} r' \in D(\mathcal{C})$.

   By hypothesis, we know that $\mathcal{C}$ well-formed. Hence, we have that $(\zeta'\lambda')(\Phi(\mathcal{C})\lambda')\downarrow = w'\lambda'$. Since $\zeta'\lambda' = \zeta\lambda$, $w'\lambda' = w\lambda$ and $\Phi(\mathcal{C})\lambda' = \Phi(\mathcal{C}_1\downarrow)\lambda$, we conclude that $(\zeta\lambda)\Phi(\mathcal{C}_1\downarrow)\lambda = w\lambda$.
6. We know that $\xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{A}\mathcal{X})$ hence this case is similar to the proof of Property 6 for the rule Cons.
7. Let $Y \in vars^2(\mathcal{C}_1\downarrow)$. We have $\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$ with $\theta = \{X \mapsto \xi\}$. Furthermore, we know that $\xi \in \mathcal{T}(\mathcal{F}_c, vars^2(D(\mathcal{C})))$ and so $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$. By Lemma C.9, we have that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C}_1\downarrow)}\theta$. We have that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}$ and $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$ (since $\mathcal{C}$ is well-formed). Since we have that $\xi \in \mathcal{T}(\mathcal{F}_c, vars^2(D(\mathcal{C})))$, we conclude that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$.
   Let $\zeta \in st(Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)))$ such that $\mathsf{path}(\zeta) \in \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}$. We know that $\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$. But $\theta = \mathsf{mgu}(X \overset{?}{=} \xi)$ with $\xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{A}\mathcal{X})$. Hence $\mathsf{path}(\zeta) \in \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}$ implies that there exists $\zeta' \in st(Y\mathsf{mgu}(Er(\mathcal{C})))$ such that $\zeta = \zeta'\theta$. Since $\mathcal{C}$ is well-formed, we deduce

that there exists k, w such that $(\zeta', k \rhd w) \in \Phi(\mathcal{C})$. Thus $(\zeta'\theta, k \rhd w\sigma) \in \Phi(\mathcal{C}_1\downarrow)$ and so the result holds.

8. Assume that $(\zeta, k \rhd w) \in \mathsf{NoUse}(\mathcal{C}_1\downarrow)$. Since $\mathsf{NoUse}(\mathcal{C}_1\downarrow) = \mathsf{NoUse}(\mathcal{C})\theta\sigma$, we have that $(\zeta', k \rhd w') \in \mathsf{NoUse}(\mathcal{C})$. Since $\mathcal{C}$ is well-formed, we deduce that there exists $Y \in vars^2(\mathcal{C})$ such that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C}) = w'$ and $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}))) < k$. As shown in the previous point, we have that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)} = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\theta$. Furthermore, we have $\xi\mathsf{acc}^1(\mathcal{C})\sigma = v\sigma = u\sigma = X\mathsf{acc}^1(\mathcal{C})\sigma$. Thus, we deduce that $\theta\mathsf{acc}^1(\mathcal{C})\sigma = \mathsf{acc}^1(\mathcal{C})\sigma$. Moreover, we know that $D(\mathcal{C}_1\downarrow) = D(\mathcal{C})\sigma\backslash\{X, i \overset{?}{\vdash} u\sigma\}$ and $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\theta\sigma$ which means that $\mathsf{acc}^1(\mathcal{C})\sigma = \theta\mathsf{acc}^1(\mathcal{C}_1\downarrow)$. We can conclude that $\mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))\rfloor_{\Phi(\mathcal{C}_1\downarrow)}\mathsf{acc}^1(\mathcal{C}_1\downarrow) = \mathsf{C}\lfloor Y\mathsf{mgu}(Er(\mathcal{C}))\rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C})\sigma = w'\sigma = w$. Lastly, for all $Z \in vars^2(\xi)$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z) \leq i = \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X)$. Since $Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow)) = Y\mathsf{mgu}(Er(\mathcal{C}))\theta$, we conclude that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}))) < k$ implies that $\mathsf{param}^{\mathcal{C}_1\downarrow}_{\mathsf{max}}(Y\mathsf{mgu}(Er(\mathcal{C}_1\downarrow))) < k$.

9. Similar to Property 7

10. Let $(Z, k \overset{?}{\vdash} t) \in D(\mathcal{C}_1\downarrow)$. Assume that $Z \notin S_2(\mathcal{C}_1)$ and let $x \in vars^1(u)$. It implies that there exists $(Z, k \overset{?}{\vdash} t') \in D(\mathcal{C})$ such that $t = t'\sigma$. Hence, there exists a variable z such that $x \in vars^1(z\sigma)$ and $z \in vars^1(t')$. Since $\mathcal{C}$ is well-formed, we deduce that there exists $(Y, p \overset{?}{\vdash} w') \in D(\mathcal{C})$ such that $z \in vars^1(w')$ and $p < k$. Thus $x \in vars^1(w'\sigma)$. If $Y \neq X$ then we deduce that $(Y, p \overset{?}{\vdash} w'\sigma) \in D(\mathcal{C}_1\downarrow)$ and so the result holds. If $Y = X$, then $u = w'$ and $p = i$. Since $\sigma = \mathsf{mgu}(u \overset{?}{=} v)$, we deduce that $x \in vars^1(v\sigma)$. Moreover, by construction of v, it implies that there exists $(Y', p' \overset{?}{\vdash} u') \in D(\mathcal{C}_1\downarrow)$ such that $Y' \in vars^2(\xi)$, $p' \leq i$ and $x \in vars^1(u')$. Since $p' \leq i = p < k$, we deduce that $p' < k$ and so the result holds.

Case $\textsc{Rule}(\tilde{p}) = \textsc{Ded-st}(\xi, \mathsf{f})$: The rule $\textsc{Ded-st}$ only adds a non-deducibility constraint in $\mathcal{C}_2$. Thus, we have that $\mathcal{C}_2\downarrow = \mathcal{C}_2$ and since $\mathcal{C}$ is a well formed constraint system, we easily deduce that $\mathcal{C}_2$ is also well-formed.

On the other hand, we have that $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$, $D(\mathcal{C}_1) = D(\mathcal{C}) \cup \{X_1, m \overset{?}{\vdash} x_1;\ \ldots;\ X_n, m \overset{?}{\vdash} x_n\}$, $\mathsf{NoUse}(\mathcal{C}_1) = \mathsf{NoUse}(\mathcal{C})$, $Er(\mathcal{C}) = Er(\mathcal{C}_1)$ and $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n)$ where $x_1, \ldots, x_n$ are fresh variables. Since $\mathcal{C}$ is well-formed, we have $vars^1(u) \cap \mathrm{dom}(\mathsf{mgu}(Eq(\mathcal{C}))) = \emptyset$. Let $\sigma = \mathsf{mgu}(u \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n))$. We have that $\Phi(\mathcal{C}_1\downarrow) = \Phi(\mathcal{C})\sigma$, $D(\mathcal{C}_1\downarrow) = D(\mathcal{C})\sigma \cup \{X_1, m \overset{?}{\vdash} x_1\sigma;\ \ldots;\ X_n, m \overset{?}{\vdash} x_n\sigma\}$ and $\mathsf{NoUse}(\mathcal{C}_1\downarrow) = \mathsf{NoUse}(\mathcal{C})\sigma$. Since the variables $X_1, \ldots, X_n$ do not appear in the frame, the facts that $\mathcal{C}$ is well-formed and $\sigma = \mathsf{mgu}(u \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n))$ implies that $\mathcal{C}_1\downarrow$ is also a well-formed constraint system. □

Lemma C.10. *Let $\mathcal{C}$ be a well-formed constraint system. We have for all $(Y, p \overset{?}{\vdash} u) \in D$, for all $x \in vars^1(u)$, there exists $(X, q \overset{?}{\vdash} v) \in D$ such that $x \in vars^1(v)$, $q \leq p$ and $X \in S_2$.*

Proof. Let $\mathcal{C}$ be a well-formed constraint system and $\mathcal{C}_1$, $\mathcal{C}_2$ the two constraint systems obtained by application of the rule on $\mathcal{C}$.

For any rule, only disequations and non-deducibility constraint are added on $\mathcal{C}_2$. Thus, we trivially have that $D(\mathcal{C}) = D(\mathcal{C}_2)$, $\Phi(\mathcal{C}) = \Phi(\mathcal{C}_2)$ and $S_2(\mathcal{C}) = S_2(\mathcal{C}_2)$. Therefore, we can conclude that $\mathcal{C}_2$ satisfies the property.

We focus now $\mathcal{C}_1$. First of all, we prove that the application of a substitution preserves the property. Let σ be a substitution such that $\mathrm{dom}(\sigma) \cap \mathrm{img}(\sigma) = \emptyset$. Let $(Y, p \overset{?}{\vdash} u\sigma) \in D(\mathcal{C}\sigma)$, let $x \in vars^1(u\sigma)$.

— if $x \in vars^1(u)$, then by hypothesis, there exists $(X, q \overset{?}{\vdash} v) \in D(\mathcal{C})$ such that $x \in vars^1(v)$, $q \leq p$ and $X \in S_2(\mathcal{C})$. But $\mathrm{dom}(\sigma) \cap \mathrm{img}(\sigma) = \emptyset$, which means that $x \in vars^1(v\sigma)$. Since $S_2(\mathcal{C}) = S_2(\mathcal{C}\sigma)$, the result holds.

— if $x \notin vars^1(u)$, it means that $x \in \mathrm{img}(\sigma)$ and that there exists $y \in vars^1(u)$ such that $x \in vars^1(y\sigma)$. Thus by hypothesis, we have that there exists $(X, q \overset{?}{\vdash} v) \in D(\mathcal{C})$ such that $y \in vars^1(v)$, $q \leq p$ and $X \in S_2(\mathcal{C})$. Therefore, we have $x \in vars^1(v\sigma)$ which prove the result.

We prove the result by case analysis on the rule applied on a constraint system :

Rule Cons: The substitution $\sigma = \mathsf{mgu}(t \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n))$ was applied on $\mathcal{C}$, with $x_1, \ldots, x_n$ fresh variables. Hence we know that $\mathcal{C}\sigma$ satisfies the property. Let first assume that the rule Cons was applied on $(X, i \overset{?}{\vdash} t)$ such that $X \notin S_2(\mathcal{C})$. In such a case, we have that $S_2(\mathcal{C}) = S_2(\mathcal{C}_1)$. On $\mathcal{C}_1$, the deducible constraints $(X_k, i \overset{?}{\vdash} x_k\sigma)$ are added, for all $j \in \{1, \ldots, n\}$. Since $\sigma = \mathsf{mgu}(t \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n))$, we know that $vars^1(x_1\sigma, \ldots, x_n\sigma) = vars^1(t\sigma)$. Thus for all $x \in vars^1(x_k\sigma)$, there exists $(Y, q \overset{?}{\vdash} v\sigma) \in D(\mathcal{C})$ such that $x \in vars^1(v)$, $q \leq i$ and $Y \in S_2(\mathcal{C}\sigma) = S_2(\mathcal{C}_1)$, which proves the result.

If we assume now that $X \in S_2(\mathcal{C})$, by application of the rule, we have $S_2(\mathcal{C}_1) = S_2(\mathcal{C}) \cup \{X_1, \ldots X_n\}$. Thus for all $(Y, p \overset{?}{\vdash} u\sigma) \in D(\mathcal{C}_1)$, for all $x \in u\sigma$, we know by hypothesis that there exists $(Z, q \overset{?}{\vdash} v\sigma) \in D(\mathcal{C}\sigma)$ such that $x \in vars^1(v\sigma)$, $q \leq p$ and $Z \in S_2(\mathcal{C}\sigma)$. If $Z \neq X$ then the result holds, else we know that $\sigma = \mathsf{mgu}(t \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n))$ and so $vars^1(x_1\sigma, \ldots, x_n\sigma) = vars^1(t\sigma)$. Therefore, there exists $k \in \{1 \ldots n\}$ such that $x \in vars^1(x_k\sigma)$, which also proves the result.

Rule Axiom(X, path): Assume that the rule is applied on $(X, i \overset{?}{\vdash} u) \in D$ and $(\xi, j \rhd v) \in \Phi$ with $\mathsf{path}(\xi) = \mathsf{path}$. The substitution $\sigma = \mathsf{mgu}(u \overset{?}{=} v)$ was applied on $\mathcal{C}$ thus we know that $\mathcal{C}\sigma$ satisfies the property. Furthermore, the deducible constraint $X, i \overset{?}{\vdash} u$ was removed from $D(\mathcal{C})$ in $\mathcal{C}_1$. If $X \notin S_2(\mathcal{C}_1)$ then the result trivially holds. Else, let $(Y, p \overset{?}{\vdash} w\sigma) \in D(\mathcal{C}_1)$ such that $Y \notin S_2(\mathcal{C}_1)$. By hypothesis, we know that for all $x \in vars^1(w\sigma)$, there exists $(Z, q \overset{?}{\vdash} t\sigma) \in D(\mathcal{C}\sigma)$ such that $x \in vars^1(t\sigma)$, $Z \in S_2(\mathcal{C}\sigma)$ and $q \leq p$. If $Z \neq X$ then the result holds, else $x \in vars^1(u\sigma)$ implies that $x \in vars^1(v\sigma)$ since $\sigma = \mathsf{mgu}(u \overset{?}{=} v)$. But the rule tells us that $j \leq i$ and so $j \leq p$. Furthermore, by Definition of a constraint system, we know that there exists $(Z', k \overset{?}{\vdash} u') \in D(\mathcal{C}_1)$ such that $x \in vars^1(u')$ and $k < j$. But $(Z', k \overset{?}{\vdash} u') \in D(\mathcal{C}\sigma)$ and so by hypothesis, there exists $(Y', k' \overset{?}{\vdash} v') \in D(\mathcal{C}\sigma)$ such that $x \in vars^1(v')$, $k' \leq k$ and $Y' \in S_2(\mathcal{C}\sigma)$. Since $k' \leq k < i$ then we have $Y' \neq X$ which implies that $(Y', k' \overset{?}{\vdash} v') \in D(\mathcal{C}_1)$ and $Y' \in S_2(\mathcal{C}_1)$. Hence the result holds.

Rule Dest$(\xi, \ell \to r, i)$: Assume that the rule is applied on $(\xi, j \rhd v) \in \Phi(\mathcal{C})$ with $\mathsf{f}(u_1, \ldots, u_n) \to w$ a fresh variant of $\ell \to r$. The substitution $\sigma = \mathsf{mgu}(v \overset{?}{=} u_1)$ was applied on $\mathcal{C}$ thus we know that $\mathcal{C}\sigma$ satisfies the property. Furthermore, only deducible constraints $(X_i, j \overset{?}{\vdash} u_i\sigma)$ were added on $\mathcal{C}_1$ such that $X_i \notin S_2(\mathcal{C}_1)$, for all $i \in \{2, \ldots, n\}$. Since by definition of our rewriting rules, $vars^1(u_2, \ldots, u_n) \subseteq vars^1(u_1)$, we have for all $i \in \{2 \ldots n\}$, for all $x \in vars^1(u_i\sigma)$, $x \in vars^1(v\sigma)$. Thus by Definition of a constraint system, we have that there exists $(Z, k \overset{?}{\vdash} t) \in D(\mathcal{C}_1)$ such that $x \in vars^1(t)$, $k < j$and so $k < i$. But $k < i$ implies that $(Z, k \overset{?}{\vdash} t) \in D(\mathcal{C}\sigma)$. Hence there exists $(Z', k' \overset{?}{\vdash} t') \in D(\mathcal{C}\sigma)$ such that $Z' \in S_2(\mathcal{C}\sigma)$, $k' \leq k'$ and $x \in vars^1(\mathcal{C}\sigma)$. But, it implies that $(Z, k' \overset{?}{\vdash} t') \in D(\mathcal{C}_1)$ and $Z' \in S_2(\mathcal{C}_1)$. Hence the result holds.

Rule Eq-left-left, Eq-left-right and Eq-right-right: For those rules, $D(\mathcal{C}\sigma) = D(\mathcal{C}_1)$ and $S_2(\mathcal{C}\sigma) = S_2(\mathcal{C}_1)$ hence the result trivially holds.

Rule Ded-st(ξ, f): Assume that the rule is applied on $(\xi, i \rhd u)$. The substitution $\sigma = \mathsf{mgu}(u \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$ was applied on $\mathcal{C}$ with $x_1, \ldots, x_n$ fresh variables. Thus we know that $\mathcal{C}\sigma$ satisfies the property. Furthermore, only deducible constraints $(X_i, m \stackrel{?}{\vdash} x_i\sigma)$ were added on $\mathcal{C}_1$ such that $X_i \notin S_2(\mathcal{C}_1)$, for all $i \in \{1, \ldots, n\}$. Since $\sigma = \mathsf{mgu}(u \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$, we have for all $i \in \{1 \ldots n\}$, for all $x \in vars^1(x_i\sigma)$, $x \in vars^1(u\sigma)$. Thus by definition of a constraint system, we have that there exists $(X, k \stackrel{?}{\vdash} t) \in D(\mathcal{C}_1)$ with $x \in vars^1(t)$, $t < i$ and so $t < m$. But $t < m$ implies that $(X, k \stackrel{?}{\vdash} t) \in D(\mathcal{C}\sigma)$ which means that there exists $(X', k' \stackrel{?}{\vdash} t') \in D(\mathcal{C}\sigma)$ with $x \in vars^1(t)$, $X' \in S_2(\mathcal{C}\sigma)$ and $k' \leq k$ and so $k' < m$. It implies that $(X', k', \stackrel{?}{\vdash} t') \in D(\mathcal{C}_1)$ and $X' \in S_2(\mathcal{C}_1)$. Hence the result holds. □

C.3 Proof of completeness

Lemma 8.5 (completeness)**.** *Let $\mathcal{C}$ be a normalised constraint system obtained by following the strategy and* Rule$(\tilde{p})$ *be a transformation rule applicable on $\mathcal{C}$. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the two resulting constraint systems obtained by applying* Rule$(\tilde{p})$ *on $\mathcal{C}$. We denote by Φ, Φ_1 and Φ_2 the respective frames of $\mathcal{C}$, $\mathcal{C}_1$ and $\mathcal{C}_2$ and we denote by S_1 the set of free variable of $\mathcal{C}$.*

For all $i \in \{1, 2\}$, for all $(\sigma_i, \theta_i) \in \mathsf{Sol}(\mathcal{C}_i)$, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and $\mathsf{Init}(\Phi)\sigma = \mathsf{Init}(\Phi_i)\sigma_i$ where $\sigma = \sigma_i|_{vars^1(\mathcal{C})}$ and $\theta = \theta_i|_{vars^2(\mathcal{C})}$

Variation: For all $i \in \{1, 2\}$, for all $(\sigma_i, \theta_i) \in \mathsf{Sol}(\overline{\mathcal{C}_i})$, $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$ and $\mathsf{Init}(\Phi)\sigma = \mathsf{Init}(\Phi_i)\sigma_i$ where $\sigma = \sigma_i|_{vars^1(\overline{\mathcal{C}})}$ and $\theta = \theta_i|_{vars^2(\overline{\mathcal{C}})}$.

Proof. We prove this lemma by case analysis on the transformation rule that is used to transform $\mathcal{C}$ on $\mathcal{C}_1$, $\mathcal{C}_2$. In each situation where some conditions are added on the resulting constraint system (without modifying the conditions that are already present in $\mathcal{C}$), the result trivially holds. This remark allows one to conclude for the rules Eq-left-left, Eq-left-right, Ded-st, Dest, and the case $i = 2$ for the rules Cons, Axiom and Eq-right-right. Therefore, it remains to prove the result for the remaining cases, *i.e.* rule Cons when $i = 1$, rule Axiom when $i = 1$ and rule Eq-right-right when $i = 1$.

Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$. We consider the remaining cases using the notation introduced in Figure 7.1.

Rule Cons(X, f), $i = 1$: Assume that $X, k \stackrel{?}{\vdash} t \in D(\mathcal{C})$ and so $D(\mathcal{C}_1) = \{X_1, k \stackrel{?}{\vdash} x_1;\ \ldots;\ X_n, k \stackrel{?}{\vdash} x_n\} \cup D(\mathcal{C}) \setminus \{X, k \stackrel{?}{\vdash} t\}$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n)$ and $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$, $\Phi(\mathcal{C}) = \Phi(\mathcal{C}_1)$ with $x_1, \ldots, x_n$ and $X_1, \ldots, X_n$ fresh variables.

Let $(\sigma_1, \theta_1) \in \mathsf{Sol}(\mathcal{C}_1)$. By definition of a solution of a constraint system, we know that:

1. $(X_j\theta_1)(\Phi(\mathcal{C}_1)\sigma_1)\downarrow = x_j\sigma_1\downarrow$ and $param(X_j\theta_1) \subseteq \{ax_1, \ldots, ax_i\}$ for any $j \in \{1, \ldots, n\}$;
2. $\sigma_1 \models Eq(\mathcal{C}) \wedge t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n) \wedge ND(\mathcal{C})$ and so $t\sigma_1\downarrow = \mathsf{f}(x_1\sigma_1, \ldots, x_n\sigma_1)\downarrow$;
3. $\theta_1 \models Er(\mathcal{C}) \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$ and so $X\theta_1 = \mathsf{f}(X_1\theta_1, \ldots, X_n\theta_1)$.

Hence, we have that:

$$\begin{array}{rcl}
(X\theta_1)(\Phi(\mathcal{C})\sigma_1)\downarrow & = & \mathsf{f}(X_1\theta_1, \ldots, X_n\theta_1)(\Phi(\mathcal{C})\sigma_1)\downarrow \\
& = & \mathsf{f}((X_1\theta_1)\Phi(\mathcal{C})\sigma_1\downarrow, \ldots, (X_n\theta_1)\Phi(\mathcal{C})\sigma_1\downarrow)\downarrow \\
& = & \mathsf{f}(x_1\sigma_1\downarrow, \ldots, x_n\sigma_1\downarrow)\downarrow \\
& = & \mathsf{f}(x_1\sigma_1, \ldots, x_n\sigma_1)\downarrow \\
& = & t\sigma_1\downarrow
\end{array}$$

Moreover, thanks to $param(X_j\theta_1) \subseteq \{ax_1, \ldots, ax_i\}$ for $j \in \{1, \ldots, n\}$, we have that $param(X\theta_1) \subseteq \{ax_1, \ldots, ax_i\}$. This allows us to conclude that$(\sigma_1|_{vars^1(\mathcal{C})}, \theta_1|_{vars^2(\mathcal{C})}) \in \mathsf{Sol}(\mathcal{C})$.

Rule Axiom(X, path), $i = 1$. Assume that $(X, k \overset{?}{\vdash} u) \in D(\mathcal{C})$ and $(\xi, j \rhd v) \in \Phi(\mathcal{C})$ with $j \leq k$ and $\mathsf{path}(\xi) = \mathsf{path}$. Thus, we have $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \xi$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u \overset{?}{=} v$, $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{(X, k \overset{?}{\vdash} u)\}$ and $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$

Let $(\sigma_1, \theta_1) \in \mathsf{Sol}(\mathcal{C}_1)$. By definition of a solution of a constraint system, we know that:

1. $\sigma_1 \models Eq(\mathcal{C}) \wedge u \overset{?}{=} v \wedge ND(\mathcal{C})$, and so $u\sigma_1 = v\sigma_1$.
2. $\theta_1 \models Er(\mathcal{C}) \wedge X \overset{?}{=} \xi$, and so $X\theta_1 = \xi\theta_1$.

Hence, we have that $(X\theta_1)(\Phi\sigma_1)\downarrow = (\xi\theta_1)(\Phi\sigma_1)\downarrow$. Moreover, since $\mathcal{C}$ is well-formed (Definition 8.2, item 3), we deduce that $\mathsf{param}^{\xi}_{\mathsf{max}}(()) \leq j$ hence for all $Y \in vars^2(\xi)$, there exists $q \leq j$ and w such that $(Y, q \overset{?}{\vdash} w) \in D(\mathcal{C})$. Since X an ξ are unifiable, we also deduce that $X \notin vars^2(\xi)$. Hence $(Y, q \overset{?}{\vdash} w) \in D(\mathcal{C}_1)$. Thanks to $(\sigma_1, \theta_1) \in \mathsf{Sol}(\mathcal{C}_1)$, $Y\theta_1\Phi(\mathcal{C}_1)\sigma_1\downarrow = w\sigma_1$. Hence, thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 5) and $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$, we deduce that $\xi\theta_1\Phi(\mathcal{C})\sigma_1\downarrow = v\sigma_1 = u\sigma_1$.

At last, since $\mathsf{param}^{\xi}_{\mathsf{max}}(()) \leq j$ and for all $Y \in vars^2(\xi)$, $(\sigma_1, \theta_1) \in \mathsf{Sol}(\mathcal{C}_1)$ also indicates that $param(Y\theta_1) \subseteq \{ax_1, \ldots, ax_j\}$ and so $param(\xi\theta_1) \subseteq \{ax_1, \ldots, ax_j\}$. At last, with $j \leq k$ and $X\theta_1 = \xi\theta_1$, we conclude that $param(X\theta_1) \subseteq \{ax_1, \ldots, ax_k\}$. This allow us to conclude that $(\sigma_1|_{vars^1(\mathcal{C})}, \theta_1|_{vars^2(\mathcal{C})}) \in \mathsf{Sol}(\mathcal{C})$.

Rule Eq-right-right(X, ξ), $i = 1$. Assume that $(X, k \overset{?}{\vdash} u) \in D(\mathcal{C})$ and $\xi \in \mathcal{T}(\mathcal{F}_c, \mathrm{dom}(\alpha))$ with $\alpha = \{Y \to w \mid (Y, j \overset{?}{\vdash} w) \in D(\mathcal{C}) \wedge j \leq i \wedge Y \in S_2\}$. Thus, we have $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \xi$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u \overset{?}{=} v$, $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{(X, k \overset{?}{\vdash} u)\}$ and $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$.

Let $(\sigma_1, \theta_1) \in \mathsf{Sol}(\mathcal{C}_1)$. By definition of a solution of a constraint system, we know that:

1. $\sigma_1 \models Eq \wedge u \overset{?}{=} v \wedge ND$, and so $u\sigma_1\downarrow = v\sigma_1\downarrow$.
2. $\theta_1 \models Er \wedge X \overset{?}{=} \xi$, and so $X\theta_1 = \xi\theta_1$.

Hence, we have that $(X\theta_1)(\Phi(\mathcal{C}_1)\sigma_1)\downarrow = (\xi\theta_1)(\Phi(\mathcal{C}_1)\sigma_1)\downarrow$. Moreover, according to Figure 7.2, we have that $\xi \in \mathcal{T}(\mathcal{F}_c, \mathrm{dom}(\alpha))$ and $v = \xi\alpha$

Since $(\sigma_1, \theta_1) \in \mathsf{Sol}(\mathcal{C}_1)$, we have that for all $(Y, j \overset{?}{\vdash} w) \in D(\mathcal{C}_1)$, $(Y\theta_1)\Phi(\mathcal{C})\sigma_1\downarrow = w\sigma_1$. Since $\xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$, we can deduce that $(\xi\theta_1)(\Phi(\mathcal{C})\sigma_1)\downarrow = v\sigma_1$. This allows us to deduce that $(X\theta_1)(\Phi(\mathcal{C})\sigma_1)\downarrow = v\sigma_1\downarrow$ and so $(X\theta_1)(\Phi(\mathcal{C})\sigma_1)\downarrow = u\sigma_1\downarrow$. Furthermore, we also know that for all $(Y, j \overset{?}{\vdash} w) \in D$, if $Y \in vars^2(\xi)$, then $j \leq i$ which means that $param(Y\theta_1) \subseteq \{ax_1, \ldots, ax_i\}$. Thus we have:

$$param(X\theta_1) = param(\xi\theta_1) \subseteq \{ax_1, \ldots, ax_i\}.$$

This allows us to conclude that $(\sigma_1|_{vars^1(\mathcal{C})}, \theta_1|_{vars^2(\mathcal{C})}) \in \mathsf{Sol}(\mathcal{C})$. □

C.4 Strategy Invariants

C.4.1 Preliminaries

We write $\mathcal{C} \to^* \mathcal{C}'$ when $\mathcal{C}'$ is obtained from $\mathcal{C}$ by applying a sequence of transformation rules.

Lemma C.11. *Let $\mathcal{C}$ and $\mathcal{C}'$ be two normalised well-formed constraint systems such that $\mathcal{C} \to^* \mathcal{C}'$. Let $\theta = \mathsf{mgu}(Er(\mathcal{C}))$, $\theta' = \mathsf{mgu}(Er(\mathcal{C}'))$ and $\sigma' = \mathsf{mgu}(Eq(\mathcal{C}'))$. The following property holds: for all $X \in vars^2(\mathcal{C})$, $\mathsf{C}\lfloor X\theta \rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')$ and $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(X\theta') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta)$.*

Proof. We prove this result by induction on the length N of the derivation $\mathcal{C} \rightarrow^* \mathcal{C}'$.

Base case $N = 0$: In such a case, $\mathcal{C} = \mathcal{C}'$. Thus, we have that $\theta = \theta'$ and $\mathsf{acc}^1(\mathcal{C}) = \mathsf{acc}^1(\mathcal{C}')$. Therefore, we have for all $X \in vars^2(\mathcal{C})$, $\mathsf{C}\lfloor X\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi'} \mathsf{acc}^1(\mathcal{C}')\sigma'$. Since $\mathcal{C}$ is normalised, we have that $\mathrm{dom}(\sigma') \cap \mathrm{img}(\mathsf{acc}^1(\mathcal{C}')) = \emptyset$, which means that $\mathsf{C}\lfloor X\theta' \rfloor_{\Phi'} \mathsf{acc}^1(\mathcal{C}')\sigma' = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi'} \mathsf{acc}^1(\mathcal{C}')$ and so $\mathsf{C}\lfloor X\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi'} \mathsf{acc}^1(\mathcal{C}')$. Furthermore, since $\theta = \theta'$ and $\mathcal{C} = \mathcal{C}'$, we trivially have that $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(X\theta') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta)$. Hence the result holds.

Inductive case $N > 0$: In such a case, we have that $\mathcal{C} \rightarrow^* \mathcal{C}'' \rightarrow \mathcal{C}'$ for some normalised constraint system $\mathcal{C}''$. By Lemma 8.2, we know that $\mathcal{C}''$ is also well-formed. Let $\theta'' = \mathsf{mgu}(Er(\mathcal{C}''))$ and $\sigma'' = \mathsf{mgu}(Eq(\mathcal{C}''))$. By inductive hypothesis, we know that for all $X \in vars^2(\mathcal{C})$, $\mathsf{C}\lfloor X\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma'' = \mathsf{C}\lfloor X\theta'' \rfloor_{\Phi''} \mathsf{acc}^1(\mathcal{C}'')$ and $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(X\theta'') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta)$. The application of a rule on $\mathcal{C}''$ produced two constraint systems $\mathcal{C}_1$ and $\mathcal{C}_2$. We show the result by case analysis on the rule applied on $\mathcal{C}''$ and we distinguish two cases depending on whether $\mathcal{C}' = \mathcal{C}_1$ or $\mathcal{C}' = \mathcal{C}_2$.

Case $\mathcal{C}'' = \mathcal{C}_2$ for any rule: According to Figure 7.1 and Figure 7.2, for any rule, only inequations or non deducibility constraint are added in $\mathcal{C}_2$, or some frame elements are marked as NoUse. Hence, we have that $\theta'' = \theta'$, $\sigma'' = \sigma'$, , $\Phi'' = \Phi'$ and $D'' = D'$. Thus, $\mathsf{C}\lfloor X\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma'' = \mathsf{C}\lfloor X\theta'' \rfloor_{\Phi''} \mathsf{acc}^1(\mathcal{C}'')$ implies $\mathsf{C}\lfloor X\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi'} \mathsf{acc}^1(\mathcal{C}')$. Moreover, we also deduce $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(X\theta') = \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(X\theta'')$. Since $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(X\theta'') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta)$, we conclude that $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(X\theta') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta)$ and so the result holds.

We now consider the case where $\mathcal{C}'' = \mathcal{C}_1$, and we consider each rule in turn:

Rule $\textsc{Cons}(X, \mathsf{f})$: Let $Y \in vars^2(\mathcal{C})$. The rule described in Figure 7.1 tells us that:

— $Eq' = Eq'' \wedge t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n)$.

— $Er' = Er'' \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$

— $(X, i \stackrel{?}{\vdash} t) \in D(\mathcal{C}'')$

Since $\mathcal{C}''$ is normalised, it means that $vars(t) \cap \mathrm{dom}(\sigma'') = \emptyset$. Furthermore, $x_1, \ldots, x_n$ are fresh variables, and so $\{x_1, \ldots, x_n\} \cap \mathrm{dom}(\sigma'') = \emptyset$. Thus, $\mathsf{mgu}(Eq'' \wedge t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n)) = \sigma''\mathsf{mgu}(t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$. Let $\Sigma = \mathsf{mgu}(t \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$ (it exists otherwise the normalised constraint system $\mathcal{C}'$ would be $\bot$), we have $\sigma' = \sigma''\Sigma$. Since $X_1, \ldots, X_n$ are also fresh variables, thus if we denote $\Theta = \mathsf{mgu}(X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n))$, then $\theta' = \theta''\Theta$.

According to Figure 7.1, $(X, i \stackrel{?}{\vdash} t) \in D(\mathcal{C}'')$ implies $(X_k, i \stackrel{?}{\vdash} x_k\Sigma) \in D(\mathcal{C}')$ for all $k \in \{1, \ldots, n\}$. Hence, we have that $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(X) = \mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(X\Theta)$. Since only $X, i \stackrel{?}{\vdash} t$ was removed from $D(\mathcal{C}'')$, we deduce that $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta''\Theta) = \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'')$. Hence $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta') = \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\theta)$.

From the first equality, we deduce that $\mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma''\Sigma$. Therefore by our inductive hypothesis, we have $\mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''} \mathsf{acc}^1(\mathcal{C}'')\Sigma$. But since no frame element was added on Φ', thus we have $\mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi'}$. Furthermore, since $\mathcal{C}''$ is well-formed, we know that $\mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}), \mathcal{X}^2)$ and since $\mathsf{C}\lfloor \mathsf{f}(X_1, \ldots, X_n) \rfloor_{\Phi'} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}), \mathcal{X}^2)$, we can deduce by Lemma C.9 that $\mathsf{C}\lfloor Y\theta''\Theta \rfloor_{\Phi'} = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi'}\{X \rightarrow \mathsf{C}\lfloor \mathsf{f}(X_1, \ldots, X_n) \rfloor_{\Phi'}\}$ and so $\mathsf{C}\lfloor Y\theta''\Theta \rfloor_{\Phi'} = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''}\Theta$

At last, since the constraint with the variable X was removed in D' and since we consider $\mathcal{C}'$ normalised, we have :

— for all $Z \in vars^2(D'') \setminus \{X\}$, $Z\Theta\mathsf{acc}^1(\mathcal{C}') = Z\mathsf{acc}^1(\mathcal{C}') = Z\mathsf{acc}^1(\mathcal{C}'')\Sigma$

— for all $i \in \{1, \ldots, n\}$, $X_i\mathsf{acc}^1(\mathcal{C}') = x_i\Sigma$ and so $X\mathsf{acc}^1(\mathcal{C}'')\Sigma = t\Sigma = \mathsf{f}(x_1, \ldots, x_n)\Sigma = X\Theta\mathsf{acc}^1(\mathcal{C}')$

Thus, we deduce that $\mathsf{acc}^1(\mathcal{C}'')\Sigma = \Theta\mathsf{acc}^1(\mathcal{C}')$, which implies, thanks to $\mathsf{C}\lfloor Y\theta''\Theta \rfloor_{\Phi'} = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''}\Theta$, that : $\mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''} \mathsf{acc}^1(\mathcal{C}'')\Sigma = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'} \mathsf{acc}^1(\mathcal{C}')$ and so $\mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'} \mathsf{acc}^1(\mathcal{C}')$.

Rule $\textsc{Axiom}(X, \mathsf{path})$: Let $Y \in vars^2(\mathcal{C})$. The rule described in Figure 7.1 tells us that:

— $Eq' = Eq'' \wedge u \stackrel{?}{=} v$.

— $Er' = Er'' \wedge X \stackrel{?}{=} \xi$

— $(X, i \stackrel{?}{\vdash} u) \in D(\mathcal{C}'')$, $(\xi, j \rhd v) \in \Phi(\mathcal{C}'')$ and $\mathsf{path}(\xi) = \mathsf{path}$

Since $\mathcal{C}''$ is normalised, it means that $(vars(u) \cup vars(v)) \cap \mathrm{dom}(\sigma'') = \emptyset$ which means that $\mathsf{mgu}(Eq'' \wedge u \stackrel{?}{=} v) = \sigma''\mathsf{mgu}(u \stackrel{?}{=} v)$. Let $\Sigma = \mathsf{mgu}(u \stackrel{?}{=} v)$; we have $\sigma' = \sigma''\Sigma$. For the same reason, if we denoted $\Theta = \mathsf{mgu}(X \stackrel{?}{=} \xi)$, we have $\theta' = \theta''\Theta$. No element has been added into the frame $\mathcal{C}'$ (from $\mathcal{C}''$), which means that $\mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'} = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta''\Theta \rfloor_{\Phi''}$. By Lemma C.9 and since $\mathcal{C}''$ is well-formed, we deduce that $\mathsf{C}\lfloor Y\theta''\Theta \rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''}\{X \to \mathsf{C}\lfloor \xi \rfloor_{\Phi''}\}$.

Thanks to $\mathcal{C}$ being well formed (Definition 8.2, item 3), we deduce that $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(\xi) \leq j$. Moreover, since $\Theta = \mathsf{mgu}(X \stackrel{?}{=} \xi)$, we deduce that $X \notin vars^2(\xi)$ and so $(\xi, j \rhd v\Sigma) \in \Phi(\mathcal{C}')$. The deducible constraint $(X, i \stackrel{?}{\vdash} u)$ being the only one removed from $D(\mathcal{C}'')$, we deduce that $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(\xi) = \mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(\xi)$. Hence $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(X) = i \geq j \geq \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(\xi)$. Therefore we deduce $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta''\Theta) \leq \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'')$. Thus we deduce that $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta') = \mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta''\Theta) \leq \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\theta)$.

At last, since the constraint with the variable X was removed in D' and $\mathcal{C}'$ normalised, we have:

— for all $Z \in vars(D'') \setminus \{X\}$, $Z\{X \to \mathsf{C}\lfloor \xi \rfloor_{\Phi''}\}\mathsf{acc}^1(\mathcal{C}') = Z\mathsf{acc}^1(\mathcal{C}') = Z\mathsf{acc}^1(\mathcal{C}'')\Sigma$

— $X\mathsf{acc}^1(\mathcal{C}'')\Sigma = u\Sigma = v\Sigma = X\{X \to \mathsf{C}\lfloor \xi \rfloor_{\Phi''}\}\mathsf{acc}^1(\mathcal{C}')$

Thus, we deduce that $\mathsf{acc}^1(\mathcal{C}'')\Sigma = \{X \to \mathsf{C}\lfloor \xi \rfloor_{\Phi''}\}\mathsf{acc}^1(\mathcal{C}')$. Hence, thanks to our inductive hypothesis (applied on $\mathcal{C}''$ and Y), we have that $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma'' = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')$, and we deduce that $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta \rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma''\Sigma = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')\Sigma = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''}\{X \to \mathsf{C}\lfloor \xi \rfloor_{\Phi''}\}\mathsf{acc}^1(\mathcal{C}') = \mathsf{C}\lfloor Y\theta''\Theta \rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}') = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')$.

Rule $\textsc{Dest}(\xi, \ell \to r, i)$: Let $Y \in vars^2(\mathcal{C})$. The rule described in Figure 7.1 tells us that:

— $Eq' = Eq'' \wedge u \stackrel{?}{=} u_1$.

— $Er' = Er''$ and so $\theta' = \theta''$.

Since $\mathcal{C}'$ is normalised, it means that $vars(u) \cap \mathrm{dom}(\sigma'') = \emptyset$. Furthermore, we know that all variable in u_1 are fresh variables, which means that $\mathsf{mgu}(Eq'' \wedge u \stackrel{?}{=} u_1) = \sigma''\mathsf{mgu}(u \stackrel{?}{=} u_1)$. Let $\Sigma = \mathsf{mgu}(u \stackrel{?}{=} u_1)$. We have that $\sigma' = \sigma''\Sigma$.

Since $\theta' = \theta''$ and no deducible constraint is removed from $D(\mathcal{C}'')$ to $D(\mathcal{C}')$, we trivially have that $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'') = \mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta')$ and so $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\theta)$.

The frame element $(\mathsf{f}(\xi, X_1, \ldots, X_n), i \rhd w)$ with $f \in \mathcal{F}_d$ was added in Φ', but since $\mathcal{C}''$ is well-formed, we know that $\mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''} \in \mathcal{T}(\mathcal{F}_c, (\mathcal{F}_d^* \cdot \mathcal{AX}) \cup \mathcal{X}^2)$, thus we deduce that $\mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi'} = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'}$.

At last, since only constraints with fresh variable X_k were added in D' and since $\mathcal{C}'$ is normalised, we have that $\mathsf{acc}^1(\mathcal{C}')_{|\,\mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))} = \mathsf{acc}^1(\mathcal{C}'')\Sigma$. With this last property, we can use the inductive hypothesis (on $\mathcal{C}''$ and Y). We obtain that $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma'' = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')$, and so $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta \rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma''\Sigma = \mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')\Sigma = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')\Sigma = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')_{|\,\mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))}$.

Since $\mathsf{C}\lfloor Y\theta'' \rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'}$, then for all $Z \in vars^2(\mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'})$, $Z \in \mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))$. Moreover, it also implies that for all $\mathsf{path} \in st(\mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'})$, $\mathsf{path} \in \mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))$. Hence, we deduce that $\mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')_{|\,\mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))} = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')$ and so $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')$.

Rules Eq-left-right and Eq-left-left: Let $Y \in vars^2(\mathcal{C})$. The rule described in Figure 7.1 tells us that $Eq' = Eq'' \wedge u_1 \stackrel{?}{=} u_2$ and $Er' = Er''$, hence $\theta'' = \theta'$. Since $\mathcal{C}''$ is normalised, we have that $(vars(u_1) \cup vars(u_2)) \cap \mathrm{dom}(\sigma'') = \emptyset$ which means that $\mathsf{mgu}(Eq'' \wedge u_1 \stackrel{?}{=} u_2) = \sigma''\mathsf{mgu}(u_1 \stackrel{?}{=} u_2)$. Let $\Sigma = \mathsf{mgu}(u_1 \stackrel{?}{=} u_2)$. We have that $\sigma' = \sigma''\Sigma$.

Neither the frame nor the constraints changed between $\mathcal{C}''$ and $\mathcal{C}'$, which means that $\mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'} = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi''}$. Since $\mathcal{C}'$ is normalised, we also have $\mathsf{acc}^1(\mathcal{C}') = \mathsf{acc}^1(\mathcal{C}'')\Sigma$. By the inductive hypothesis (applied on $\mathcal{C}''$ and Y), we have that $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma'' = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')$ and so $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma''\Sigma = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')\Sigma = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')$.

Furthermore, since $\theta' = \theta''$ and no deducible constraint is removed from $D(\mathcal{C}'')$ to $D(\mathcal{C}')$, we trivially have that $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'') = \mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta')$ and so $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\theta)$.

Rule Eq-right-right(X, ξ): Let $Y \in vars^2(\mathcal{C})$. The rule described in Figure 7.1 tells us that:

- $Eq' = Eq'' \wedge u \stackrel{?}{=} v$.
- $Er' = Er'' \wedge X \stackrel{?}{=} \xi$
- $(X, i \stackrel{?}{\vdash} u) \in D(\mathcal{C}'')$

where $\xi \in \mathcal{T}(\mathcal{F}_c, \mathrm{dom}(\alpha))$ and $v = \xi\alpha$ with $\alpha = \{Y \to u \mid (Y, j \stackrel{?}{\vdash} u) \in D(\mathcal{C}'') \wedge j \leq i \wedge Y \in S_2\}$. But $\mathsf{acc}^1(\mathcal{C}'')_{|\,\mathrm{dom}(\alpha)} = \alpha$. Hence, we have that $v = \xi\mathsf{acc}^1(\mathcal{C}'')$.

Since $\mathcal{C}''$ is normalised, it means that $(vars(u) \cup vars(v)) \cap \mathrm{dom}(\sigma'') = \emptyset$ which means that $\mathsf{mgu}(Eq'' \wedge u \stackrel{?}{=} v) = \sigma''\mathsf{mgu}(u \stackrel{?}{=} v)$. Let $\Sigma = \mathsf{mgu}(u \stackrel{?}{=} v)$. We have that $\sigma' = \sigma''\Sigma$. For the same reason, we have that $\theta' = \theta''\Theta$ where $\Theta = \{X \to \xi\}$. No element has been added into the frame $\mathcal{C}'$ (w.r.t. $\mathcal{C}''$). Hence, we have that $\mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'} = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta''\Theta\rfloor_{\Phi''}$. By Lemma C.9, $\mathcal{C}''$ well formed and $\xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$, we deduce that $\mathsf{C}\lfloor Y\theta''\Theta\rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''}\Theta$.

Since for all $Z \in vars^2(\xi)$, $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Z) \leq i = \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(X)$ and $\Theta = \{X \to \xi\}$, we deduce that $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(X\Theta) \leq \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(X)$. Therefore $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta''\Theta) \leq \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'')$. Thus we deduce that $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta') = \mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta''\Theta) \leq \mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\theta)$.

At last, since the constraint with the variable X was removed in D' and $\mathcal{C}'$ normalised, we have:

- for all $Z \in vars(D'') \setminus \{X\}$, $Z\Theta\mathsf{acc}^1(\mathcal{C}') = Z\mathsf{acc}^1(\mathcal{C}') = Z\mathsf{acc}^1(\mathcal{C}'')\Sigma$
- $X\mathsf{acc}^1(\mathcal{C}'')\Sigma = u\Sigma = v\Sigma = X\Theta\mathsf{acc}^1(\mathcal{C}'')\Sigma = X\Theta\mathsf{acc}^1(\mathcal{C}')$

Thus, we deduce that $\mathsf{acc}^1(\mathcal{C}'')\Sigma = \Theta\mathsf{acc}^1(\mathcal{C}')$. Hence, thanks to our inductive hypothesis (applied on $\mathcal{C}''$ and Y), we have that $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma'' = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')$. From this, we deduce that $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma''\Sigma = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')\Sigma = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''}\Theta\mathsf{acc}^1(\mathcal{C}') = \mathsf{C}\lfloor Y\theta''\Theta\rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}')$. Hence we conclude that $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')$.

Rule Ded-st: Let $Y \in vars^2(\mathcal{C})$. The rule described in Figure 7.1 tells us that:

- $Eq' = Eq'' \wedge u \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n)$.
- $Er' = Er''$ and so $\theta' = \theta''$.

Since $\mathcal{C}''$ is normalised, this means that $vars(u) \cap \mathrm{dom}(\sigma'') = \emptyset$. Furthermore, we know that the variables x_i are fresh variables, which means that $\mathsf{mgu}(Eq'' \wedge u \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n)) = \sigma''\mathsf{mgu}(u \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$. Let $\Sigma = \mathsf{mgu}(u \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$. We have that $\sigma' = \sigma''\Sigma$.

Since $\theta' = \theta''$ and no deducible constraint are removed from $D(\mathcal{C}'')$ to $D(\mathcal{C}')$, we trivially have that $\mathsf{param}^{\mathcal{C}''}_{\mathsf{max}}(Y\theta'') = \mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta')$ and so $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y\theta') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\theta)$.

No element has been added into the frame $\mathcal{C}'$ (w.r.t. $\mathcal{C}''$). Hence, we have that $\mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi'} = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}$.

At last, since only constraints with fresh variable X_i were added in D' and since $\mathcal{C}'$ normalised, we have that $\mathsf{acc}^1(\mathcal{C}')_{|\,\mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))} = \mathsf{acc}^1(\mathcal{C}'')\Sigma$. With this last property, we can use the inductive hypothesis (applied on $\mathcal{C}''$ and Y). We obtain that $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma'' = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')$, and so $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma''\Sigma = \mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''}\mathsf{acc}^1(\mathcal{C}'')\Sigma = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}'')\Sigma = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')_{|\,\mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))}$.

Since $\mathsf{C}\lfloor Y\theta''\rfloor_{\Phi''} = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}$, then for all $Z \in vars^2(\mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'})$, $Z \in \mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))$. Moreover, it also implies that for all $\mathsf{path} \in st(\mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'})$, $\mathsf{path} \in \mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))$. Hence, we deduce that $\mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')_{|\,\mathrm{dom}(\mathsf{acc}^1(\mathcal{C}''))} = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')$ and so $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C})\sigma' = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}')$.

□

Lemma C.12. *Let $\mathcal{C}$ and $\mathcal{C}'$ be two normalised well-formed constraint systems such that $\mathcal{C} \to^* \mathcal{C}'$. Let $\sigma = \mathsf{mgu}(Eq(\mathcal{C}'))$, and $(\xi, i \rhd u) \in \Phi$. There exist $(\xi', i \rhd u') \in \Phi'$ such that $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ and $u' = u\sigma$.*

Proof. According to the rules described in Figure 7.1 and 7.2 and the fact that $\mathcal{C}$ is well-formed, the path $\mathsf{path}(\xi)$ of a frame element $(\xi, i \rhd u)$ is never modified. The only operation that affect this frame element is the normalisation of a constraint system, i.e. the most general unifier of Eq is applied on u (idem for Er). Thus, if $\sigma = \mathsf{mgu}(Eq(\mathcal{C}'))$, we can conclude that $u = u'\sigma$. □ □

Lemma C.13. *Let $\mathcal{C}$ be a normalised well-formed constraint system. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two normalised well formed constraint systems such that $\mathcal{C} \to^* \mathcal{C}_1$ and $\mathcal{C} \to^* \mathcal{C}_2$. Let ρ be a variable renaming from $\mathcal{X}^1$ to $\mathcal{X}^1$. Let $ax \in \mathcal{AX}$ such that $\{ax, i \rhd u_1\} \in \Phi_1$ and $\{ax, i \rhd u_2\} \in \Phi_2$. Let $w \in \mathcal{F}_d^*$, $\{\xi_1, i_1 \rhd v_1\} \in \Phi_1$ and $\{\xi_2, i_2 \rhd v_2\} \in \Phi_2$ such that $\mathsf{path}(\xi_1) = \mathsf{path}(\xi_2) = w \cdot ax$. We have:*

$$\text{if } u_1\rho = u_2 \text{ then } v_1\rho = v_2$$

Proof. We prove the result by induction on $|w|$:

Base case $|w| = 0$: In such a case, then $\xi_1 = \xi_2 = ax$. Thus by Property 1 of a well-formed constraint system, we know that $u_1 = v_1$ and $u_2 = v_2$ and so the result trivially holds.

Inductive step $|w| > 0$: Assume that $w = \mathsf{f} \cdot w'$ and $u_1\rho = u_2$. By Property 2 of a well-formed constraint system, we know that there exists $(\xi_1', i_1' \rhd v_1') \in \Phi_1$ and $(\xi_2', i_2' \rhd v_2') \in \Phi_2$ such that $\mathsf{path}(\xi_1') = \mathsf{path}(\xi_2') = w' \cdot ax$. Thus by our inductive hypothesis, we know that $v_1'\rho = v_2'$. Furthermore, by definition of the rule DEST (the only rule that can add an element into the frame), we know that there exists a position p (actually for our rewriting rules $p = 1$) such that $v_1 = v_1'|_p$ and $v_2 = v_2'|_p$. Hence, we have that $v_1\rho = (v_1'|_p)\rho = (v_1'\rho)|_p = v_2'|_p = v_2$. □

C.4.2 Preservation of the strategy invariants by the rules

Lemma C.14. *Let $\mathcal{C}$ be a well formed constraint system satisfying $\mathsf{InvVarConstraint}(s)$. Let $R(\tilde{p})$ be an occurrence of the rule* CONS*, or* AXIOM *or* EQ-RIGHT-RIGHT *with support $s' \leq s$. Let $\mathcal{C}_1, \mathcal{C}_2$ be the two constraint systems obtained by applying $R(\tilde{p})$ on $\mathcal{C}$. We have that for all $i \in \{1, 2\}$, $\mathcal{C}_i$ satisfies the invariant $\mathsf{InvVarConstraint}(s)$.*

Proof. According to the definitions of the three rules, $\mathcal{C}_2$ only differs from $\mathcal{C}$ by an addition of an inequality on recipes. Thus, we trivially deduce that $\mathcal{C}_2$ satisfies $\mathsf{InvVarConstraint}(s)$. We prove the result for $\mathcal{C}_1$ by case analysis on the rule applied:

Rule $\text{CONS}(X, \mathsf{f})$*:* Since the support of the rule is s', then there exists $(X, s' \overset{?}{\vdash} u) \in D(\mathcal{C})$. Moreover, $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$ hence, we deduce $u \in \mathcal{X}^1$. But $D(\mathcal{C}_1) = D(\mathcal{C})\sigma \backslash \{X, s' \overset{?}{\vdash} u\sigma\} \cup \{X_1, s' \overset{?}{\vdash} x_1\sigma; \ldots; X_n, s' \overset{?}{\vdash} x_n\sigma\}$ where $\sigma = \mathsf{mgu}(u \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n))$ and $x_1, \ldots, x_n, X_1, \ldots, X_n$ are fresh variables. Since $s' \leq s$ then $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$ also implies that $X \in S_2$. Thus, by definition of the rule CONS, $X_1, \ldots, X_n \in S_2$.

Moreover, since $u \in \mathcal{X}^1$, we deduce that $\sigma = \{u \mapsto \mathsf{f}(x_1, \ldots, x_n)\}$. Hence for all $i \in \{1, \ldots, n\}$, $x_i\sigma = x_i$. Moreover, since $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$, we deduce that for all $(Y, j \rhd y) \in D$ such that $Y \neq X$, $y\sigma = y$. Thus, thanks to $x_1, \ldots, x_n$ and $X_1, \ldots, X_n$ being fresh, the result holds.

Rule $\text{AXIOM}(X, \mathsf{path})$*:* Since the support of the rule is s', then there exists $(X, s' \overset{?}{\vdash} u) \in D(\mathcal{C})$ and $(\xi, j \rhd v) \in \Phi(\mathcal{C})$ with $j \leq s'$. Moreover, $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$ hence, we deduce $u \in \mathcal{X}^1$. But $D(\mathcal{C}_1) = D(\mathcal{C})\sigma \backslash \{X, s' \overset{?}{\vdash} u\sigma\}$ where $\sigma = \mathsf{mgu}(u \overset{?}{=} v)$. Thus the first property is trivially satisfied.

Moreover, $u \in \mathcal{X}^1$ implies that either (a) $\sigma = \{u \mapsto v\}$ or (b) $v \in \mathcal{X}^1$ and $\sigma = \{v \mapsto u\}$. In case (a), since $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$, we deduce that for all $(Z, k \overset{?}{\vdash} z) \in D(\mathcal{C})$ and $k \leq s$,

if $Z \neq X$ then $(Z, k \overset{?}{\vdash} z \in D(\mathcal{C}_1)$ hence the result holds. In case (b), $v \in \mathcal{X}^1$ implies, by the origination property, that there exists $(Y, k \overset{?}{\vdash} v') \in D(\mathcal{C})$ such that $k < j$ and $v \in vars^1(v')$. But $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$ thus $v = v'$. Hence for all $(Z_1, \ell_1 \overset{?}{\vdash} z_1) \in D(\mathcal{C}_1)$, if $\ell_1 \leq s$ then either $Z_1 \neq Y$ and so $(Z_1, \ell_1 \overset{?}{\vdash} z_1) \in D(\mathcal{C})$, or $Z_1 = Y$ and $v' = u$. By relying on $\mathcal{C}$ satisfying $\mathsf{InvVarConstraint}(s)$, the result holds.

Rule Eq-right-right(X, ξ)*:* Proof similar to the rule Axiom. □

Lemma C.15. *Let $\mathcal{C}$ be a well formed constraint system satisfying $\mathsf{InvUntouched}(s)$. Let $R(\tilde{p})$ be an occurrence of a rule with support $s' \leq s$. Let $\mathcal{C}_1, \mathcal{C}_2$ be the two constraint systems obtained by applying $R(\tilde{p})$ on $\mathcal{C}$. We have that for all $i \in \{1, 2\}$, $\mathcal{C}_i$ satisfies the invariant $\mathsf{InvUntouched}(s)$.*

Proof. The rule Dest is the only one that add frame element. But the support of the rule being s', Dest can only introduce frame element of the form $(\zeta, s' \rhd w)$. Thus since $s' \leq s$, we deduce deduce that for all $(\xi, k \rhd u) \in \Phi(\mathcal{C}_1)$ (resp. $\Phi(\mathcal{C}_2)$), if $s < k$ then $\xi = ax_k$.

Similarly, the only rules that add elements in $Er(\mathcal{C})$ are Cons, Axiom and Eq-right-right. In case of Cons and Eq-right-right, the result trivially holds by definition of the rules and the fact that $s' \leq s$. In case of application of the rule Axiom(X, path), by definition, we know that there exists $(X, s' \overset{?}{\vdash} u) \in D(\mathcal{C})$ and $(\xi, k \rhd v) \in \Phi(\mathcal{C})$ with $k \leq s'$. But since $\mathcal{C}$ is well formed (Definition 8.2, item 3), we know that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi) \leq k$ which implies that for all $Y \in vars^2(\xi)$, $(Y, \ell \overset{?}{\vdash} w) \in D(\mathcal{C})$ implies that $\ell \leq k$. Hence any new (in)equations in Er only contain variables Y such that $\mathsf{param}^{\mathcal{C}_1}_{\mathsf{max}}(Y) \leq s' \leq s$. Hence the result holds. □

Lemma C.16. *Let $\mathcal{C}$ be a well formed constraint system satisfying $\mathsf{InvNoUse}(s)$ (resp. $\mathsf{InvDest}(s)$, $\mathsf{InvVarFrame}(s)$ and $\mathsf{InvDedsubs}$). Let $R(\tilde{p})$ be an occurrence of a rule different from Dest, or the rule Dest with support $s' > s$. Let $\mathcal{C}_1, \mathcal{C}_2$ be the two constraint systems obtained by applying $R(\tilde{p})$ on $\mathcal{C}$. We have that for all $i \in \{1, 2\}$, $\mathcal{C}_i$ satisfies the invariant $\mathsf{InvNoUse}(s)$ (resp. $\mathsf{InvDest}(s)$, $\mathsf{InvVarFrame}(s)$ and $\mathsf{InvDedsubs}$).*

Proof. We prove the different invariant by case analysis on the rule applied.

Rule Cons(X, f)*:* The rule Cons only adds the inequation $\mathsf{root}(X) \neq \mathsf{f}$ on $Er(\mathcal{C}_2)$. Thus, $\mathcal{C}_2$ trivially satisfies all the wanted invariants.

On the other hand, we have that $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})\theta\sigma$, $\mathsf{NoUse}(\mathcal{C}_1) = \mathsf{NoUse}(\mathcal{C})\theta\sigma$, $ND(\mathcal{C}_1) = ND(\mathcal{C})\sigma$ and $D(\mathcal{C}_1) = D(\mathcal{C})\sigma \backslash \{X, i \overset{?}{\vdash} t\sigma\} \cup \{X_1, i \overset{?}{\vdash} x_1\sigma;\ \ldots;\ X_n, i \overset{?}{\vdash} x_n\sigma\}$ where $\sigma = \mathsf{mgu}(t \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n))$ and $\theta = \mathsf{mgu}(X \overset{?}{=} \mathsf{f}(X_1, \ldots, X_n))$. We now prove the different invariants one by one.

Let $(\xi, p \rhd v) \in \Phi(\mathcal{C}_1)$ such that $p \leq s$. Since $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})\sigma$, there exists ξ' and v' such that $(\xi', p \rhd v') \in \Phi(\mathcal{C})$, $\xi'\theta = \xi$ and $v'\sigma = v$:

- $\mathsf{InvNoUse}(s)$: Assume that $v \in \mathcal{X}^1$. In such a case, $v'\sigma = v$ implies $v' \in \mathcal{X}^1$. But $\mathcal{C}$ satisfies $\mathsf{InvNoUse}(s)$ hence $(\xi', p \rhd z) \in \mathsf{NoUse}(\mathcal{C})$. With $\mathsf{NoUse}(\mathcal{C}_1) = \mathsf{NoUse}(\mathcal{C})\theta\sigma$, we deduce that $(\xi, p \rhd v) \in \mathsf{NoUse}(\mathcal{C}_1)$ thus the result holds.
- $\mathsf{InvVarFrame}(s)$: Let $Y \in vars^2(\xi)$. In such a case, $\xi'\theta = \xi$ implies that there exists $Y' \in vars^2(\xi')$ such that $Y \in vars^2(Y'\theta)$. But $\mathcal{C}$ satisfies $\mathsf{InvVarFrame}(s)$ implies that there exists $q < p$ and $u \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$ such that $(Y', q \overset{?}{\vdash} u) \in D(\mathcal{C})$. If $Y' \neq X$ then $Y' = Y$ and so $(Y, q \overset{?}{\vdash} u\sigma) \in D(\mathcal{C}_1)$ thus the result holds. If $Y' = X$ then $Y \in \{X_1, \ldots, X_n\}$ and $s' = q$. But for all $k \in \{1, \ldots, n\}$, we have $(X_k, s' \overset{?}{\vdash} x_k\sigma) \in D(\mathcal{C}_1)$. With $s' = q < p$ then the result holds.
- $\mathsf{InvDest}(s)$: The result is direct from $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})\theta\sigma$, $\mathsf{NoUse}(\mathcal{C}_1) = \mathsf{NoUse}(\mathcal{C})\theta\sigma$, $ND(\mathcal{C}_1) = ND(\mathcal{C})\sigma$ and the fact that $\mathcal{C}$ satisfies the invariant $\mathsf{InvDest}(s)$.

— InvDedsub: Assume that $|\Phi(\mathcal{C})| = m$. Since $\mathcal{C}$ satisfies the invariant, then for all $\mathsf{g} \in \mathcal{F}_c$, $(\xi', p \rhd v') \in \mathsf{NoUse}(\mathcal{C})$ implies that either (a) there exists $Y_1, \ldots, Y_k \in vars^2(\mathcal{C})$ such that for all $i \in \{1, \ldots, k\}$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y_i \mathsf{mgu}(Er(\mathcal{C}))) \leq s$ and $\mathsf{C}\lfloor \mathsf{g}(Y_1, \ldots, Y_m)\mathsf{mgu}(Er(\mathcal{C}))\rfloor \mathsf{acc}^1(\mathcal{C}) = v'$, or else (b) $ND(\mathcal{C}) \vDash \forall \tilde{x}. v' \stackrel{?}{\neq} \mathsf{g}(x_1, \ldots, x_n) \vee m \stackrel{?}{\not\vdash} x_1 \vee \ldots \vee m \stackrel{?}{\not\vdash} x_k$ where $x_1, \ldots, x_k$ are fresh variables.

In case (a), since $\mathsf{mgu}(Er(\mathcal{C}_1)) = \mathsf{mgu}(Er(\mathcal{C}))\theta$ and $\mathsf{f}(X_1, \ldots, X_n) = \mathsf{C}\lfloor \mathsf{f}(X_1, \ldots, X_n)\rfloor \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$ then thanks to Lemma C.9, we deduce that $\mathsf{C}\lfloor \mathsf{g}(Y_1, \ldots, Y_k)\mathsf{mgu}(Er(\mathcal{C}_1))\rfloor = \mathsf{C}\lfloor \mathsf{g}(Y_1, \ldots, Y_k)\mathsf{mgu}(Er(\mathcal{C}))\rfloor\theta$. But for all $Z \in vars^2(\mathcal{C})$, $Z\mathsf{acc}^1(\mathcal{C})\sigma = Z\theta\mathsf{acc}^1(\mathcal{C}_1)$ thus $\mathsf{C}\lfloor \mathsf{g}(Y_1, \ldots, Y_k)\mathsf{mgu}(Er(\mathcal{C}_1))\rfloor \mathsf{acc}^1(\mathcal{C}_1) = \mathsf{C}\lfloor \mathsf{g}(Y_1, \ldots, Y_k)\mathsf{mgu}(Er(\mathcal{C}))\rfloor \mathsf{acc}^1(\mathcal{C})\sigma = v'\sigma = v$. Hence the result holds.

In case (b), since $ND(\mathcal{C}_1) = ND(\mathcal{C})\sigma$ then $ND(\mathcal{C}_1) \vDash \forall \tilde{x}. v \stackrel{?}{\neq} \mathsf{g}(x_1, \ldots, x_n) \vee m \stackrel{?}{\not\vdash} x_1 \vee \ldots \vee m \stackrel{?}{\not\vdash} x_k$. Hence the result holds.

Rule $\textsc{Dest}(\xi, \ell \to r, s')$ *and* $s' > s$ *:* Since the rule only adds a frame element $(\zeta, s' \rhd w)$ for some ζ, w and applies a substitution on first order term, then the result holds by relying on $\mathcal{C}$ verifying each invariant respectively.

The proofs of all the others rules are similar to the rule Cons. Note that for the rule $\textsc{Ded-st}(\xi, \mathsf{f})$, if $\mathcal{C}$ satisfies the invariant InvDedsub, then it implies that the application of the rule $\textsc{Ded-st}(\xi, \mathsf{f})$ was in fact useless hence according to the strategy, such application could not have happen. Hence the result holds. □

Lemma C.17. *Let M be a well-formed matrix of constraint systems satisfying* $\mathsf{InvMatrix}(s)$*. Let $R(\tilde{p})$ be an occurrence of a rule different from* Dest *and* Eq-left-right*, or* Dest *with support $s' > s$, or* Eq-left-right *with support $s' > s$. Let M_1, M_2 be the two matrix of constraint systems obtained by applying $R(\tilde{p})$ on M (if $R(\tilde{p})$ is an internal rule, only M_1 exists). We have that for all $i \in \{1, 2\}$, M_i satisfies the invariant* $\mathsf{InvMatrix}(s)$*.*

Proof. Note that the invariant $\mathsf{InvMatrix}(s)$ mainly focus on the path of the frames. But thanks to M being well-formed we have that for all $\mathcal{C} \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$, $\mathcal{C}$ is well-formed. Hence by Definition 8.2, item 1, we know that all path of the frame element are closed. Hence relying on Lemma C.8, we trivially deduce that the result holds for the rule Cons, Axiom, Eq-right-right, Eq-left-left and Ded-st. Hence it remains to prove that the result holds for Eq-left-right with support $s' > s$ and Dest with support $s' > s$. But in both cases, the only possible modifications on the frames or the sets NoUse only occurs on frame elements with support stricly bigger than s. Hence the result holds. □

Lemma C.18. *Let $\mathcal{C}$ be a well formed constraint system. Let $s \in \mathbb{N}$, if $\mathcal{C}$ satisfies* InvDedsub*s (resp.* $\mathsf{InvVarFrame}(s)$*,* $\mathsf{InvNoUse}(s)$*,* $\mathsf{InvDest}(s)$ *and* $\mathsf{InvVarConstraint}(s)$*) then for all $s' \leq s$, $\mathcal{C}$ satisfies* InvDedsub*s' (resp.* $\mathsf{InvVarFrame}(s')$*,* $\mathsf{InvNoUse}(s')$*,* $\mathsf{InvDest}(s')$ *and* $\mathsf{InvVarConstraint}(s')$*).*

If $\mathcal{C}$ satisfies $\mathsf{InvUntouched}(s)$ *then for all $s' \geq s$, $\mathcal{C}$ satisfies* $\mathsf{InvUntouched}(s')$*.*

Proof. Direct from the definition of the invariants. □

Lemma C.19. *Let M be a well formed constraint system satisfying* InvGeneral*. Let $R(\tilde{p})$ be an occurrence of a rule different from* Dest *and* Eq-left-right*. Let M_1, M_2 be the two matrix of constraint systems obtained by applying $R(\tilde{p})$ on M (if $R(\tilde{p})$ is an internal rule, only M_1 exists). We have that for all $i \in \{1, 2\}$, M_i satisfies the invariant* InvGeneral*.*

Let $R(\tilde{p}$ be a occurrence of the rule Dest *or* Eq-left-right*. Let M' be the matrix of constraint systems obtained by applying $R(\tilde{p})$ on M. We have that M' satisfies Properties 5, 6 and 7 of the invariant* InvGeneral*.*

C.4.3 Invariants specific to different steps and phases of the strategy

In this subsection, we will show several invariants that are specific to each phase and step of the strategy. In fact, the invariants are interconnected. Intuitively, we have to define an invariant for each step and also an invariant for the link between each step, i.e. we have to show that the invariant of the end of a step corresponds to the invariant at the beginning of the next step.

Let (M_0, M_0') be a pair of matrices of constraint systems. We say that a pair of matrices of constraint systems (M_1, M_1') is obtained from (M_0, M_0') by applying Step i of Phase j of the strategy with parameters s (and k), for some $i \in \{a, b, c, d, e\}$, $j \in \{1, 2\}$ if $(M_0, M_0') \to^* (M_1, M_1')$ and the rules applied follow exactly the description of Step i of Phase j with parameters s (and k) given in Section 7.4.

Moreover, we will say that (M_1, M_1') is obtained from (M_0, M_0') at the end of Step i of Phase j of the strategy with parameters s (and k) if $(M_0, M_0') \to^* (M_1, M_1')$, the rules applied follow exactly the description of Step i of Phase j with parameters s (and k) given in Section 7.4 and no rule following the description of Step i of Phase j with parameters s (and k) is applicable on (M_1, M_1').

C.4.3.1 Invariants of Phase 1

Property C.1. *We say that a pair of matrices of constraint systems (M, M') satisfy* $\mathsf{PP1}(s)$ *if M and M' have the same structure, satisfy* $\mathsf{InvMatrix}(s)$ *and* $\mathsf{InvGeneral}$*, and for all constraint system $\mathcal{C}$ in M or M', $\mathcal{C}$ satisfies the invariants* $\mathsf{InvVarConstraint}(s)$*,* $\mathsf{InvVarFrame}(s)$*,* $\mathsf{InvDest}(s)$*,* $\mathsf{InvNoUse}(s)$ *and* $\mathsf{InvUntouched}(s)$*.*

Moreover, if Φ is a frame of a constraint system in M or M' and $s = |\Phi|$ then (M, M') satisfies also $\mathsf{InvDedsub}$*.*

Lemma C.20. *Let (M, M') be a pair of row matrices of initial constraint systems having the same structure. (M, M') satisfies* $\mathsf{PP1}(0)$*.*

Proof. We start by proving that M and M' satisfy $\mathsf{InvGeneral}$ (Definition 8.1). First of all, Item 5, 6 and 7 trivially hold since M and M' are row matrices, *i.e.* there is only one constraint system in each column of M and M'. Furthermore, by definition of an initial constraint system, we know that for all $\mathcal{C}$ different from $\bot$, for all $(\xi, i \rhd u) \in \Phi(\mathcal{C})$, $\xi = ax_i$. Hence Item 1 is trivially true. More over, for all $\xi' \in \Pi_n$ with $\mathsf{root}(\xi') \notin \mathcal{F}_c$, if $\mathsf{path}(\xi') = \mathsf{path}(\xi\theta)$ then $\mathsf{path}(\xi\theta) = \mathsf{path}(ax_i\theta) = \mathsf{path}(ax_i) = ax_i$. Hence, $\mathsf{path}(\xi') = ax_i$ implies $ax_i = \xi'$ and so Item 2 holds. Since $\xi = ax_i$, then Item 4 also holds. At last, $\mathcal{C}$ is an initial constraint system also implies $\mathsf{NoUse}(\mathcal{C}) = \emptyset$. Hence Item 3 holds.

The invariants $\mathsf{InvMatrix}(s)$, $\mathsf{InvVarConstraint}(0)$, $\mathsf{InvVarFrame}(0)$, $\mathsf{InvDest}(0)$, $\mathsf{InvNoUse}(0)$ are trivially satisfied since their no deducible constraint $(X, i \overset{?}{\vdash} u)$ or frame element $(\xi, i \overset{?}{\vdash} u)$ such that $i \leq 0$.

It remains to prove that (M, M') satisfies the invariant $\mathsf{InvUntouched}(0)$. We already know that for all constraint system $\mathcal{C}$ in M or M'. If $\mathcal{C}$ is different from $\bot$ then for all $(\xi, i \rhd u) \in \Phi(\mathcal{C})$, $\xi = ax_i$. Furthermore we know that $Er(\mathcal{C}) = \top$. At last, by definition of an initial constraint system $vars^2(D(\mathcal{C})) \subseteq S_2(\mathcal{C})$. Hence $\mathcal{C}$ satisfies the invariant $\mathsf{InvUntouched}(0)$. □

C.4.3.2 Invariants of Phase 1, Step a

Property C.2. *We say that a pair of matrices of constraint systems (M, M') satisfy* $\mathsf{PP1Sa}(s)$ *if M and M' have the same structure, satisfy* $\mathsf{InvMatrix}(s-1)$ *and* $\mathsf{InvGeneral}$*, and for all constraint system $\mathcal{C}$ in M or M', if $\mathcal{C} \neq \bot$ then*

- *$\mathcal{C}$ invariants* $\mathsf{InvDest}(s-1)$*,* $\mathsf{InvVarFrame}(s-1)$*,* $\mathsf{InvNoUse}(s-1)$*,* $\mathsf{InvUntouched}(s)$*; and*
- *for all $(\xi, s \rhd u) \in \Phi(\mathcal{C})$ with $u \in \mathcal{X}^1$, there exists $X \in S_2(\mathcal{C})$ and $\ell < s$ such that $(X, \ell \overset{?}{\vdash} u) \in D(\mathcal{C})$; and*

— *for all* $(\xi, s \rhd u) \in \Phi(\mathcal{C})$*, either* $\xi \in \mathcal{AX}$ *or there exists* $X_2, \ldots, X_n \in \mathcal{X}^2 \setminus S_2(\mathcal{C})$, $\mathsf{f} \in \mathcal{F}_d$ *and* $(\xi', p \rhd v) \in \Phi(\mathcal{C})$ *such that* $\xi = \mathsf{f}(\xi', X_2, \ldots, X_n)$ *and* $p \leq s$*; and*

— *for all* $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$*, for all* $\mathsf{f} \in \mathcal{F}_c$*, for all* $(\xi, j \rhd v) \in \Phi(\mathcal{C})$*,* $Er(\mathcal{C}) \nvdash X \overset{?}{\neq} \xi$ *and* $Er(\mathcal{C}) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$*; and*

— *for all* $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$*,* $X \in S_2(\mathcal{C})$ *implies* $i = s$ *and there exists a unique frame element* $(\mathsf{g}(\xi_1, \ldots, \xi_n), j \rhd v) \in \Phi(\mathcal{C})$ *and* $k \in \{2, \ldots, n\}$ *such that* $j = s$ *and* $\xi_k = X$

Lemma C.21. *Let* (M, M') *be a pair of matrices of constraint systems satisfies* $\mathsf{PP1}(s-1)$*. For all pair of matrices of constraint systems* (M_1, M_1') *obtained during Step a of the first phase on* (M, M') *with support* s*,* (M_1, M_1') *satisfies* $\mathsf{PP1Sa}(s)$*.*

Proof. Let (M_1, M_1') be a pair of matrices of constraint systems obtained during Step a of the first phase on (M, M'). We show by induction on the size N of the branch of (M_1, M_1') that (M_1, M_1') satisfies the wanted properties. plus a new one : for all $\mathcal{C}$ in (M_1, M_1'),

6. for all $x \in vars^2(\{u \mid X, i \overset{?}{\vdash} u \wedge i < s\})$, if for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $X \in S_2(\mathcal{C})$, $i < s$ implies $x \neq u$, then for all $u \in \{v \mid (\xi, i \rhd v) \in \Phi(\mathcal{C}) \text{ or } (X, i \rhd v) \in D(\mathcal{C})\}$, for all position p, if $u|_p = x$ then there exists p' such that $p = p' \cdot 1$ and $u|_{p'} = \mathsf{pk}(x)$.

This property stated that when a variable is never a right hand term of a deducible constraint, then this variable is always used under the constructor pk.

Base case $N = 0$*:* In such a case we have that $(M_1, M_1') = (M, M')$. Hence, we trivially have that M_1 and M_1' satisfy $\mathsf{InvMatrix}(s-1)$ and $\mathsf{InvGeneral}$. Furthermore, we also have that for all $\mathcal{C}$ in M_1 and M_1', $\mathcal{C}$ satisfies the invariant $\mathsf{InvVarFrame}(s-1)$ and $\mathsf{InvNoUse}(s-1)$. Furthermore, thanks to Lemma C.18, we also have that $\mathcal{C}$ satisfies $\mathsf{InvUntouched}(s)$. We know prove the other properties:

6. We know that $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s-1)$ hence for all $x \in vars^2(\{u \mid X, i \overset{?}{\vdash} u \wedge i < s\})$, there exists $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $x = u$ and $X \in S_2(\mathcal{C})$. Hence the property holds.
3. for all $(\xi, s \rhd x) \in \Phi(\mathcal{C})$, thanks to the property of origination of a constraint system, we know that there exists $(X, \ell \rhd u) \in D(\mathcal{C})$ such that $\ell < s$ and $x \in vars^1(u)$. But $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s-1)$ which means that $u \in \mathcal{X}^1$ and so $x = u$.
4. Since $\mathcal{C}$ satisfies $\mathsf{InvUntouched}(s-1)$, we know that for all $(\xi, s \rhd u) \in \Phi(\mathcal{C})$, $\xi = ax_s \in \mathcal{AX}$.
5. Since $\mathcal{C}$ satisfies $\mathsf{InvUntouched}(s-1)$, we know that for all $(X, k \overset{?}{\vdash} u) \in D(\mathcal{C})$, if $k \geq s$ then $X \notin vars^2(Er(\mathcal{C}))$. Hence for all $\mathsf{f} \in \mathcal{F}_c$, for all recipe ξ on a frame element of $\Phi(\mathcal{C})$, we have that $Er(\mathcal{C}) \nvdash X \overset{?}{\neq} \xi$ and $Er(\mathcal{C}) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$.

Inductive step $N > 0$*:* In such a case, there exists a couple of matrices of constraint systems (M_2, M_2') such that (M_2, M_2') is the father of (M_1, M_1'). By our inductive hypothesis, we know that (M_2, M_2') satisfies the properties stated by the lemma. For all $\mathcal{C}$ in (M_1, M_1'), there exists a constraint system $\mathcal{C}'$ in (M_2, M_2') such that $\mathcal{C}' \to \mathcal{C}$.

1. We know that M_2 and M_2' satisfy $\mathsf{InvMatrix}(s-1)$, $M_2 \to M_1$, $M_2' \to M_1$. Since the rule applied are of support s, then thanks to Lemma C.17 we have that M_1 and M_1' satisfy $\mathsf{InvMatrix}(s-1)$.

   Thanks to Lemma C.19, we already know that Property 5 of $\mathsf{InvGeneral}$ is satisfied. Thus, it remains to prove the others properties. Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and let $(\xi, i \rhd u) \in \Phi(\mathcal{C})$. Thanks to Lemma 8.5, we know that there exists $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$ such that $\theta' = \theta|_{vars^2(\mathcal{C}')}$ and $\sigma' = \sigma|_{vars^1(\mathcal{C}')}$ We do a case analysis on the rule applied

   — Case Eq-left-right: In such a case, we have that $Er(\mathcal{C}') = Er(\mathcal{C})$ and if $\Sigma = \mathsf{mgu}(Eq(\mathcal{C}))$, we have that $\Phi(\mathcal{C}) = \Phi(\mathcal{C}')\Sigma$ and $D(\mathcal{C}) = D(\mathcal{C}')\Sigma$. Hence we have $\theta = \theta'$. Thus since, by hypothesis, $\mathcal{C}'$ satisfies the Property 1 of $\mathsf{InvGeneral}$, we have $param(\xi\theta') \subseteq \{ax_1, \ldots, ax_i\}$ and so $param(\xi\theta) \subseteq \{ax_1, \ldots, ax_i\}$.

Let $\xi' \in \Pi_n$ with $\mathsf{root}(\xi') \notin \mathcal{F}_c$, $\mathsf{path}(\xi') = \mathsf{path}(\xi\theta)$ and $\xi'(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Since $\theta = \theta'$, we have that $\mathsf{path}(\xi') = \mathsf{path}(\xi\theta')$. Furthermore, $\sigma' = \sigma|_{vars^1(\mathcal{C}')}$ implies that $\xi'(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C}')\sigma')\downarrow$. Hence by hypothesis, since $\mathcal{C}$ satisfies Property 2 of $\mathsf{InvGeneral}$, we have $param(\xi') \not\subseteq \{ax_1, \ldots, ax_{i-1}\}$.

— Case DEST when the guess is negative: The proof is similar to the case EQ-LEFT-RIGHT.

— Case DEST: Otherwise, we have that $Er(\mathcal{C}') = Er(\mathcal{C})$ and if $\Sigma = \mathsf{mgu}(Eq(\mathcal{C}))$, we have that $\Phi(\mathcal{C}) = \Phi(\mathcal{C}')\Sigma \cup \{\mathsf{g}(\zeta, X_2, \ldots, X_n), s \rhd w\}$ and $D(\mathcal{C}) = D(\mathcal{C}')\Sigma \cup \{X_i, s \rhd v_i\}_{i=2..n}$ where $X_2, \ldots, X_2$ are fresh variables, $(\zeta, j \rhd t) \in \Phi(\mathcal{C})$ and $j \leq s$. Let's denote $\zeta' = \mathsf{g}(\zeta, X_2, \ldots, X_n)$.

Since $\mathcal{C}'$ satisfies $\mathsf{InvGeneral}$, we already know that $\mathsf{param}^{\zeta}_{\mathsf{max}}(()\theta) \leq j$. Furthermore, by definition of $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we have that $\mathsf{param}^{\mathsf{X}}_{\mathsf{max}}(()_k\theta) \leq s$, for all $k = 2..n$. Hence we can conclude that $param(\zeta'\theta) \subseteq \{ax_1, \ldots, ax_s\}$.

We now show that $ax_s \in st(\zeta'\theta)$. If $j = s$, then we have that $ax_s \in st(\zeta\theta)$ since $\mathcal{C}'$ satisfies $\mathsf{InvGeneral}$. Thus we conclude that $ax_s \in st(\zeta'\theta)$. Else $j < s$. $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$ implies that $\sigma' \vDash ND(\mathcal{C}')$. But we know that $\mathcal{C}'$ also satisfies $\mathsf{InvDest}(s-1)$. Hence, $j < s$ and $\sigma' \vDash ND(\mathcal{C}')$ implies that there exists no recipe $(\xi_2, \ldots, \xi_n) \in \Pi_n$ such that $param(\xi_2, \ldots, \xi_n) \subseteq \{ax_1, \ldots, ax_{s-1}\}$ such that $\mathsf{g}(\zeta\theta', \xi_2, \ldots, \xi_n)(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. But we know that $\mathsf{g}(\zeta\theta', \xi_2, \ldots, \xi_n)(\Phi(\mathcal{C}')\sigma')\downarrow = \mathsf{g}(\zeta\theta, \xi_2, \ldots, \xi_n)(\Phi(\mathcal{C})\sigma)\downarrow$. At last, since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies $\zeta'\theta(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$, we can conclude that there exists $k \in \{2, \ldots, n\}$ such that $ax_s \in st(X_k\theta)$ and so $ax_s \in st(\zeta')$. Thus, $\mathcal{C}$ satisfies Properties 1 and 2 of $\mathsf{InvGeneral}$.

It remains to prove Property 3 and 4 of the invariant $\mathsf{InvGeneral}$. Let $X \in vars^2(\mathcal{C})$ such that $\mathsf{path}(\xi) \in st(\mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C}))\rfloor_\Phi)$. We already know that the rule EQ-LEFT-RIGHT and DEST do not modify Er. Furthermore, all new second order variables introduced by the DEST are not instantiated during Step a. Hence if $\mathcal{C}''$ is the constraint system in M or M' such that $\mathcal{C}'' \to^* \mathcal{C}$, then we have we have that $\mathsf{path}(\xi) \in st(\mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C}))\rfloor_\Phi)$ implies that $X \in vars^2(\mathcal{C}'')$ and $\mathsf{path}(\xi) \in st(\mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C}''))\rfloor_\Phi)$. Furthermore, since $\mathcal{C}''$ satisfies $\mathsf{InvUntouched}(s-1)$, we can deduce that $i < s$. Hence, by hypothesis on $\mathcal{C}''$, we have that $(\xi, i \rhd u'') \notin \mathsf{NoUse}(\mathcal{C}'')$ where $u''\mathsf{mgu}(Eq(\mathcal{C}'')) = u$. At last, since during Step a with support s, EQ-LEFT-RIGHT only add frame element of the form $(\xi', s \rhd v)$ and $i < s$, we can conclude that $(\xi, i \rhd u) \notin \mathsf{NoUse}(\mathcal{C})$. Hence $\mathcal{C}$ satisfies Property 3 of $\mathsf{InvGeneral}$.

Assume now that $(\xi, i \rhd u) \notin \mathsf{NoUse}(\mathcal{C})$ and let $\xi' \in st(\xi)$ such that $(\xi', j \rhd v) \in \Phi(\mathcal{C}) \cap \mathsf{NoUse}(\mathcal{C})$. Since EQ-LEFT-RIGHT only add frame element of the form $(\zeta, s \rhd w)$ and $\mathcal{C}'$ satisfies Property 4 of $\mathsf{InvGeneral}$, we can deduce that $i = j = s$. Thus, the only way this case occur is if DEST was applied on a frame element which belong to NoUse or if EQ-LEFT-RIGHT was applied on $(\xi', j \rhd v)$ after that DEST was applied on it. But this case is impossible since it would imply that $v \in \mathcal{X}^1$ and we now by definition of DEST that u is a strict subterm of v. Hence we have that $(\xi', j \rhd v) \notin \mathsf{NoUse}(\mathcal{C})$ and so $\mathcal{C}$ satisfies Property 4 of $\mathsf{InvGeneral}$.

2. Since $\mathcal{C}'$ in (M_2, M_2'), $\mathcal{C}'$ satisfies $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvNoUse}(s-1)$ and $\mathsf{InvUntouched}(s)$. Thanks to Lemmas C.15 and C.16, we can deduce that $\mathcal{C}$ satisfies $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvNoUse}(s-1)$ and $\mathsf{InvUntouched}(s)$.

6. Let $x \in vars^2(\{u \mid X, i \overset{?}{\vdash} u \wedge i < s\})$ such that for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $i < s$ and $X \in S_2(\mathcal{C})$, $x \neq u$. We know that $\mathcal{C}' \to \mathcal{C}$ hence we do a case analysis on the rule applied DEST or EQ-LEFT-RIGHT:

Case EQ-LEFT-RIGHT(X, ξ)*:* We focus on the son which modifies the terms in the constraint systems, i.e. when the equality guess is true. Since the rule EQ-LEFT-RIGHT(X, ξ) is applicable then there exists i, u_0, v_0 such that $(X, i \overset{?}{\vdash} u_0) \in D(\mathcal{C}')$, $X \in S_2(\mathcal{C}')$ and $(\xi, s \rhd v_0) \in \Phi(\mathcal{C}')$. Let $\sigma = \mathsf{mgu}(u, v)$. We know that $\Phi(\mathcal{C}) = \Phi(\mathcal{C}')\sigma$ and $D(\mathcal{C}) = D(\mathcal{C}')\sigma$. But $x \in vars^2(\{(u \mid X, i \overset{?}{\vdash} u) \in D(\mathcal{C}) \wedge i < s\})$, thus it implies that $x \notin \mathrm{dom}(\sigma)$ and

so $x' \in vars^2(\{(u \mid X, i \overset{?}{\vdash} u) \in D(\mathcal{C}') \wedge i < s\})$. Furthermore, it also implies that for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}')$ such that $i < s$ and $X \in S_2(\mathcal{C}')$, $x \neq u$. Indeed, if there exists $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C}')$ then $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ which is a contradiction with our hypothesis.
Let $t \in \{v \mid (\xi, i \rhd v) \in \Phi(\mathcal{C}) \text{ or } (X, i \rhd v) \in D(\mathcal{C})\}$, thus there exists $t' \in \{v \mid (\xi, i \rhd v) \in \Phi(\mathcal{C}') \text{ or } (X, i \rhd v) \in D(\mathcal{C}')\}$ such that $t'\sigma = t$. Let p a position such that $t|_p = x$.
If $x \notin \text{img}(\sigma)$, then we can deduce that $t'|_p = x$. Hence, by our inductive hypothesis, we have that there exists p' such that $p = p' \cdot 1$ and $t'|_{p'} = \mathsf{pk}(x)$ and so $t'\sigma|_{p'} = \mathsf{pk}(x)$.
If $x \in \text{img}(\sigma)$, then $x \in u_0$ or $x \in v_0$. But by our inductive hypothesis we have that for all p, if $u_0|_p = x$ then there exists p' such that $p = p' \cdot 1$ and $u_0|_{p'} = \mathsf{pk}(x)$. Hence by definition of the mgu, for all $y \in \text{dom}(\sigma)$, $x \in vars(y\sigma)$ implies that either (a) $x = y\sigma$ or (b) there for all p, if $y\sigma|_p = x$ then there exists p' such that $p = p' \cdot 1$ and $y\sigma|_{p'} = \mathsf{pk}(x)$.

Case (a): In such a case, we have that for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}')$ such that $i < s$ and $X \in S_2(\mathcal{C}')$, $y \neq u$. Indeed, if there exists $(X, i \overset{?}{\vdash} y) \in D(\mathcal{C}')$ then $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ which is a contradiction with our hypothesis. Hence, $t|_p = x$ implies that $t'|_p = y$ or $t'|_p = x$. If $t'|_p = x$ then the result holds similarly to the case $x \notin \text{img}(\sigma)$. If $t'|_p = y$, we know by our inductive hypothesis that $t'|_{p'} = \mathsf{pk}(y)$ with $p = p' \cdot 1$ and so $t'\sigma|_{p'} = \mathsf{pk}(x)$. Hence the result holds.
Case (b): Otherwise, $t|_p = x$ implies that $t'|_p = x$ or there exists p', p'' such that $p = p' \cdot p''$ and $t'|_{p'} = y$ and $y\sigma|_{p''} = x$. But by hypothesis on y, there exists p''' such that $p'' = p''' \cdot 1$ and $y\sigma|_{p'''} = \mathsf{pk}(x)$, hence we have that $t'\sigma|_{p' \cdot p'''} = \mathsf{pk}(x)$. Hence the result holds.

Case $\textsc{Dest}(\xi, \ell \to r, s)$*:* Once again, we focus on the son which may instantiate the terms in the constraint system, i.e. when the guess is positive. Since $\textsc{Dest}(\xi, \ell \to r, s)$ is applicable then there exists $i \leq s, u_0$ such that $(\xi, i \rhd u_0) \in \Phi(\mathcal{C}')$. First of all, we deduce $u_0 \notin \mathcal{X}^1$. Indeed, if $u_0 \in \mathcal{X}^1$, then since $\mathcal{C}'$ satisfies the properties the lemma, we have that either $(\xi, i \rhd u_0) \in \mathsf{NoUse}(\mathcal{C}')$ if $i < s$, or else there exists $(X, j \overset{?}{\vdash} u_0) \in D(\mathcal{C}')$. Thus we would have that the rule $\textsc{Eq-left-right}(X, \xi)$ would be applicable which contradict the strategy that imposes that the rule Eq-left-right are prioritised over the rule Dest.

By definition of the rule Dest, we know that $\Phi(\mathcal{C}')\sigma \cup \{\xi', s \rhd w\sigma\} = \Phi(\mathcal{C})$ and $\{(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}') \mid X \in S_2\}\sigma = \{(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}) \mid X \in S_2\}$ where $\sigma = \mathsf{mgu}(u_0, v_1)$ and $\mathsf{g}(v_1, \ldots, v_n) \to x_1$ is a fresh instance of $\ell \to r$.
But the definition of $\ell \to r$ implies that $v_1 = \mathsf{f}(x_1, x_2)$, for $\mathsf{f} \in \{\mathsf{senc}, \langle\rangle, \mathsf{sign}\}$; or $v_1 = \mathsf{aenc}(x_1, \mathsf{pk}(x_2))$. Thus, since $u_0 \notin \mathcal{X}^1$, then we have $vars^2(D(\mathcal{C}')) \cap \text{dom}(\sigma) = \emptyset$ when $\mathsf{f} \in \{\mathsf{senc}, \langle\rangle, \mathsf{sign}\}$. Hence the result holds. When $v_1 = \mathsf{aenc}(x_1, \mathsf{pk}(x_2))$, the only way to have $vars^2(D(\mathcal{C}')) \cap \text{dom}(\sigma) \neq \emptyset$ is if $u_0 = \mathsf{aenc}(u_1, y)$ with $y \in vars^2(D(\mathcal{C}'))$. But in such a case, it implies that we have that $y\sigma = \mathsf{pk}(x_2)$. Thus if $x = x_2$ then x satisfies the properties since $x_2 \notin vars^1(\mathcal{C}')$ and $y\sigma = \mathsf{pk}(x_2)$.

3. Let $(\xi, s \rhd x) \in \Phi(\mathcal{C})$ with $x \in \mathcal{X}^1$. Thanks to Property 6, we know that if for all $(X, i \rhd v) \in D(\mathcal{C})$, $X \in S_2(\mathcal{C})$ and $i < s$ implies $v \neq x$, then all term in the frame, x are always used under the constructor pk. But it is not the case for $(\xi, s \rhd x)$. Hence, we deduce that there exists $(X, i \overset{?}{\vdash} v) \in D(\mathcal{C})$ such that $X \in S_2(\mathcal{C})$,$i < s$ and $v = x$.
4. The rule Eq-left-right and Dest do not modify $Er(\mathcal{C}')$. Hence, we only have to look at the new frame element that are added on the frame. But by definition of Dest, the application of $\textsc{Dest}(\xi, \ell \to r, s)$ implies the addition of a new element $(\mathsf{g}(\xi, X_2, \ldots, X_n), s \rhd w)$ where $\mathsf{g} \in \mathcal{F}_d$, $X_2, \ldots X_n$ are fresh, and there exists i, u such that $(\xi, i \rhd u) \in \Phi(\mathcal{C})$. Hence the result holds.
5. Once again, the rule Eq-left-right and Dest no not modify $Er(\mathcal{C}')$ hence $Er(\mathcal{C}') = Er(\mathcal{C})$. Hence, $\mathcal{C}'$ satisfies Property C.2 implies that $\mathcal{C}$ satisfies Property C.2. □

Lemma C.22. *Let (M_1, M_1') be a pair of matrices of constraint system satisfying* $\mathsf{PP1}(s-1)$.

Let (M_2, M_2') be a pair of matrices of constraint systems obtained from (M_1, M_1') by applying Step a of Phase 1 of the strategy with parameters s. Moreover we assume that (M_2, M_2') is obtained after a sequence of application of Dest *or* Eq-left-right. *For all constraint system $\mathcal{C}, \mathcal{C}'$ in (M_2, M_2'),*

1. *for all $(\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C})$, there exists $X \in S_2(\mathcal{C})$ such that for all $\mathcal{C}''$ in M or M', if there exists $(\xi', s \rhd u') \in \mathsf{NoUse}(\mathcal{C}'')$ such that $\mathsf{path}(\xi') = \mathsf{path}(\xi)$ then $\mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C}''))\rfloor_{\Phi(\mathcal{C}'')}\mathsf{acc}^1(\mathcal{C}'') = u'$. Else, by denoting $v' = \mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C}''))\rfloor_{\Phi(\mathcal{C}'')}\mathsf{acc}^1(\mathcal{C}'')$, we have that $Eq(\mathcal{C}'') \vDash v' \overset{?}{\neq} u'$.*
2. *for all $(\xi, i \rhd u) \in \Phi(\mathcal{C}) \setminus \mathsf{NoUse}(\mathcal{C})$, for all $(\xi', i' \rhd u') \in \Phi(\mathcal{C}') \setminus \mathsf{NoUse}(\mathcal{C}')$, if $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ then $\textsc{Dest}(\xi, \ell \to r, s)$ is applicable on $\mathcal{C}$ is equivalent to $\textsc{Dest}(\xi', \ell \to r, s)$ is applicable on $\mathcal{C}'$*

Proof. The proof of this Lemma follows the application of the rule Dest and Eq-left-right in sequence. □

Property C.3. *We say that a pair of matrices of constraint systems (M, M') satisfy* $\mathsf{PP1SaE}(s)$ *if M and M' have the same structure, satisfy* $\mathsf{InvMatrix}(s-1)$ *and* $\mathsf{InvGeneral}$, *and for all constraint system $\mathcal{C}$ in M or M', if $\mathcal{C} \neq \bot$ then $\mathcal{C}$ satisfies the invariants* $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvDest}(s)$, $\mathsf{InvNoUse}(s)$ *and* $\mathsf{InvUntouched}(s)$. *Moreover, for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$,*

— *$X \notin S_2(\mathcal{C})$ implies that $i = s$.*

— *for all $\mathsf{f} \in \mathcal{F}_c$, for all $\xi \in \Pi_n$, $Er(\mathcal{C}) \nVdash X \overset{?}{\neq} \xi$ and $Er(\mathcal{C}) \nVdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$.*

Lemma C.23. *Let (M, M') be a pair of matrices of constraint systems satisfying* $\mathsf{PP1}(s-1)$. *For all pair of matrices of constraint systems (M_1, M_1') obtained from (M, M') at the end of Step a of Phase 1 of the strategy with parameter s, (M_1, M_1') satisfies* $\mathsf{PP1SaE}(s)$.

Proof. We know that all constrain system in M and M' satisfies $\mathsf{InvVarConstraint}(s-1)$. Hence, for all $\mathcal{C} \in M$ (resp. M'), for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, if $i \leq s-1$ implies that $X \in S_2(\mathcal{C})$. Moreover, $\mathcal{C}$ satisfies $\mathsf{InvUntouched}(s-1)$ which implies that if $i > s-1$, $X \in S_2(\mathcal{C})$. Thus we deduce that $X \in S_2(\mathcal{C})$. But during the step a of Phase 1, only the rule Dest add new deducible constraint. Furthermore, Dest is applied with support s. Hence, Dest can only add deducible constraint of the form $Y, s \overset{?}{\vdash} v$. Thus for all $\mathcal{C}$, for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, if $X \notin S_2(\mathcal{C})$ then $i = s$.

Since (M, M') satisfies $\mathsf{PP1}(s-1)$, we already know that already know that M and M' satisfy the invariant $\mathsf{InvMatrix}(s-1)$. Furthermore, Lemma C.21 also indicates that for all constraint system $\mathcal{C}$ in M or M', $\mathcal{C}$ satisfies the invariant $\mathsf{InvGeneral}$, $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvDest}(s-1)$, $\mathsf{InvNoUse}(s-1)$ and $\mathsf{InvUntouched}(s)$. Hence it remains to prove that $\mathcal{C}$ satisfies $\mathsf{InvDest}(s)$ and $\mathsf{InvNoUse}(s)$.

At the end of Step a, we know that the rules Dest and Eq-left-right are not applicable on a constraint system in M or M' for any parameter with support inferior or equal to s.
Invariant $\mathsf{InvNoUse}(s)$*:* Let $(\xi, p \rhd v) \in \Phi(\mathcal{C})$. If $p < s$ then, thanks to $\mathsf{InvNoUse}(s-1)$, the result holds. Else assume that $p = s$ and $v \in \mathcal{X}^1$. But, thanks to Lemma C.21, we have that there exists $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $u = v$ and $i < s$. Thus, since Eq-left-right is not applicable on $\mathcal{C}$, we have that either $(\xi, p \rhd v) \in \mathsf{NoUse}(\mathcal{C})$ or $Eq(\mathcal{C}) \vDash u \overset{?}{\neq} v$. But $u = v$ implies $Eq(\mathcal{C}) \vDash u \overset{?}{\neq} u$ which implies that $C{\downarrow} = \bot$ by normalisation, which is a contradiction with a fact that $\Phi(\mathcal{C})$. Thus we have that $(\xi, p \rhd v) \in \mathsf{NoUse}(\mathcal{C})$ and so $\mathcal{C}$ satisfies $\mathsf{InvNoUse}(s)$.

Invariant $\mathsf{InvDest}(s)$*:* Let $(\xi, p \rhd v) \in \Phi(\mathcal{C})$, $\mathsf{f} \in \mathcal{F}_d$ and $(\xi, p \rhd v) \notin \mathsf{NoUse}(\mathcal{C})$ and $p \leq s$. We do a case analysis on p:

— Case $p = s$: In such a case, we only have to show that either $(\xi', s \rhd v') \in \Phi(\mathcal{C})$ for some ξ' such that $\mathsf{path}(\xi') = \mathsf{f} \cdot \mathsf{path}(\xi)$; or else $ND \vDash \forall \tilde{x}, v \neq u_1 \vee s \overset{?}{\nmid} u_2 \vee \ldots \vee s \overset{?}{\nmid} u_n$ where $\mathsf{f}(u_1, \ldots, u_n) \to w$ is a fresh rewriting rule with $vars^1(u_1, \ldots, u_n, w) = \tilde{x}$.

But we know that DEST is not applicable on $\mathcal{C}$ for any parameter with support s. Hence $\textsc{Dest}(\xi, \mathsf{f}(u_1, \ldots, u_n) \to w, s)$ is not applicable. But since ξ, $\mathsf{f}(u_1, \ldots, u_n) \to w$ and s are valid parameter for $\mathcal{C}$, it implies that $\textsc{Dest}(\xi, \mathsf{f}(u_1, \ldots, u_n) \to w, s)$ was already applied and so the definition of the rule DEST in Figure 7.1 allows us to conclude.

— Case $p < s$: We know that $\mathcal{C}$ satisfies $\mathsf{InvDest}(s-1)$. Hence, we have to show that if for every $p \leq k \leq s-1$, $ND \vDash \forall \tilde{x}, v \neq u_1 \vee s-1 \overset{?}{\nmid} u_2 \vee \ldots \vee s-1 \overset{?}{\nmid} u_n$ where $\mathsf{f}(u_1, \ldots, u_n) \to w$ is a fresh rewriting rule with $vars^1(u_1, \ldots, u_n, w) = \tilde{x}$, then either $(\xi', s \rhd v') \in \Phi(\mathcal{C})$ for some ξ' such that $\mathsf{path}(\xi') = \mathsf{f} \cdot \mathsf{path}(\xi)$; or else $ND \vDash \forall \tilde{x}, v \neq u_1 \vee s \overset{?}{\nmid} u_2 \vee \ldots \vee s \overset{?}{\nmid} u_n$.

But once again, we know that DEST is not applicable on $\mathcal{C}$ for any parameter with support s. Hence $\textsc{Dest}(\xi, \mathsf{f}(u_1, \ldots, u_n) \to w, s)$ is not applicable. But since ξ, $\mathsf{f}(u_1, \ldots, u_n) \to w$ and s are valid parameter for $\mathcal{C}$, it implies that $\textsc{Dest}(\xi, \mathsf{f}(u_1, \ldots, u_n) \to w, s)$ was already applied and so the definition of the rule DEST in Figure 7.1 allows us to conclude. □

C.4.3.3 Invariants of Phase 1, Step *b*

Given a pair of matrices (M, M') such that M (resp. M') has n columns (resp. n'), we say that the k^{th} column of (M, M') is either the k^{th} column of M if $k \leq n$; or else the $(k-n)^{\text{th}}$ column of M' if $k > n$. If $n + n' < k$ then the k^{th} column of (M, M') is not defined.

Moreover, will assume from now on that m is the size of a frame of a constraint system in M or M'.

Property C.4. *We say that a pair of matrices of constraint systems (M, M') satisfy $\mathsf{PP1Sb}(s, k)$ if (M, M') satisfies $\mathsf{PP1SaE}(s)$ and for all $i \leq k$, for all constraint system $\mathcal{C}$ in the i^{th} column of (M_1, M_1'), $\mathcal{C}$ also satisfies $\mathsf{InvVarConstraint}(s)$ and $\mathsf{InvVarFrame}(s)$ (and $\mathsf{InvDedsub}$ when $s = m$)*

Lemma C.24. *Let (M, M') be a pair of matrices of constraint systems satisfying $\mathsf{PP1SaE}(s)$. (M, M') satisfies $\mathsf{PP1Sb}(s, 0)$.*

Proof. Trivial since (M, M') does not have a 0^{th} column. □

Property C.5. *We say that a pair of matrices of constraint systems (M, M') satisfy $\mathsf{PP1SbE}(s, k)$ if M and M' have the same structure, satisfy $\mathsf{InvMatrix}(s-1)$ and $\mathsf{InvGeneral}$, and for all constraint system $\mathcal{C}$ in M or M', if $\mathcal{C} \neq \bot$ then $\mathcal{C}$ satisfies the invariants $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvDest}(s)$, $\mathsf{InvNoUse}(s)$ and $\mathsf{InvUntouched}(s)$. Moreover, for all constraint system $\mathcal{C}$ in the k^{th} column of (M_1, M_1'), for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$,*

— *$X \notin S_2(\mathcal{C})$ implies $u \in \mathcal{X}^1$ and $i = s$.*

— *for all $\mathsf{f} \in \mathcal{F}_c$, for all $\xi \in \Pi_n$, $Er(\mathcal{C}) \nvdash X \overset{?}{\neq} \xi$ and $Er(\mathcal{C}) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$.*

— *if $s = m$ then $\mathcal{C}$ satisfies $\mathsf{InvDedsub}$.*

At last, for all $i \leq k$, for all constraint system $\mathcal{C}$ in the i^{th} column of (M_1, M_1'), $\mathcal{C}$ also satisfies $\mathsf{InvVarConstraint}(s)$ and $\mathsf{InvVarFrame}(s)$ (and $\mathsf{InvDedsub}$ when $s = m$).

Lemma C.25. *Let (M, M') be a pair of matrices of constraint systems satisfying $\mathsf{PP1Sb}(s, k)$. For all pair of matrices of constraint systems (M_1, M_1') obtained at the end of Step b of the first phase with support s and column k on (M, M'), (M_1, M_1') satisfies $\mathsf{PP1SbE}(s, k)$.*

Proof. During Step b of the first phase, the rule DEST and EQ-LEFT-RIGHT are not applied. Furthermore, the rules applied can only be applied with support inferior or equal to s. Hence, thanks to Lemmas C.19, C.15, C.16 and C.17, we can deduce that M_1 and M_1' satisfy $\mathsf{InvMatrix}(s-$

1), and for all $\mathcal{C}$ in M_1 or M_1', $\mathcal{C}$ satisfies $\mathsf{InvGeneral}$, $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvNoUse}(s)$, $\mathsf{InvDest}(s)$, $\mathsf{InvDedsub}$ (when $s = |\Phi(\mathcal{C})|$) and $\mathsf{InvUntouched}(s)$.

At the end of Step b, we cannot applied the rule DED-ST. Hence for all $(\xi, p \rhd v) \in \Phi(\mathcal{C})$, we know that DED-ST$(\xi, \mathsf{f})$ is useless for any $\mathsf{f} \in \mathcal{F}_c$. However, the definition of DED-ST(ξ, f) being useless for all ξ and path implies the invariant $\mathsf{InvDedsub}$. Thus we deduce that $\mathcal{C}$ satisfies the invariant $\mathsf{InvDedsub}$.

Let $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$. Since (M_1, M_1') is obtained at the end of step b, we know that CONS(X, f) is not strongly applicable on $\mathcal{C}$, for all $\mathsf{f} \in \mathcal{F}_c$. Hence it implies that $u \in \mathcal{X}^1$ and either (a) for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$; or (b) for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \nvDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$.

In case (a), since AXIOM(X, ξ) is not strongly applicable on $\mathcal{C}$, for all ξ, we deduce that for all $(\xi, j \rhd v) \in \Phi(\mathcal{C})$, if $j \leq i$ then $Er(\mathcal{C}) \vDash X \overset{?}{\neq} \xi$. But $\mathcal{C}$ satisfies $\mathsf{InvNoUse}(s)$, $\mathsf{InvDest}(s)$, i.e. the rule DEST$(\xi, \ell \to r, s)$ is useless for all ξ and $\ell \to r$. Thus, by Definition 7.9 of the normalisation, we would have that $\mathcal{C}{\downarrow} = \bot$ which is a contradiction with the fact that $D(\mathcal{C}) \neq \emptyset$. Hence this case is impossible

In Case (b), the rule AXIOM can only be applied during Step b if the strong application conditions of the rule are satisfied. But since $u \in \mathcal{X}^1$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \nvDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$, we deduce that the rule AXIOM(X, path) was never applied during step b for any path. Hence, we deduce that for all ξ, $Er \nvDash X \overset{?}{\neq} \xi$. Hence the result holds.

At last, we know that the only rule that add deducible constraints during step b are CONS and DED-ST. But DED-ST is only applied when $i = s$ and it create deducible constraint of the form $X, s \overset{?}{\vdash} u$. On the other hand, thanks to Lemma C.23, we know that for all $\mathcal{C}$ in M or M', for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, if $X \notin S_2$ then $i = s$. But, according to Figure 7.1, by applying CONS(Y, f) for some $\mathsf{f} \in \mathcal{F}_c$ and $Y, i \overset{?}{\vdash} v$, if $Y \in S_2$ (resp. $\notin S_2$) then the rule CONS creates new deducible constraint systems of the form $(Z, i \overset{?}{\vdash} w)$ where Z is in S_2 (resp. not in S_2). Since the only for all $\mathcal{C}$ in M or M', for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, if $X \notin S_2$ then $i = s$, we deduce that the index of any new deducible constraints created by CONS whose second order variables are not in S_2 is necessary s. Thus the result holds. □

C.4.3.4 Invariants of Phase 1, Step c

Property C.6. *We say that a pair of matrices of constraint systems (M, M') satisfy $\mathsf{PP1ScE}(s, k)$ if M and M' have the same structure, satisfy $\mathsf{InvMatrix}(s-1)$ and $\mathsf{InvGeneral}$, and for all constraint system $\mathcal{C}$ in M or M', if $\mathcal{C} \neq \bot$ then $\mathcal{C}$ satisfies the invariants $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvDest}(s)$, $\mathsf{InvNoUse}(s)$ and $\mathsf{InvUntouched}(s)$. Moreover, for all constraint system $\mathcal{C}$ in the k^{th} column of (M_1, M_1'), for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$,*

- *$X \notin S_2(\mathcal{C})$ implies $u \notin \mathcal{X}^1$.*
- *for all $\mathsf{f} \in \mathcal{F}_c$, for all $\xi \in \Pi_n$, $Er \nvDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er \nvDash X \overset{?}{\neq} \xi$*
- *if $s = m$ then $\mathcal{C}$ satisfies $\mathsf{InvDedsub}$.*

At last, for all $i \leq k$, for all constraint system $\mathcal{C}$ in the i^{th} column of (M_1, M_1'), $\mathcal{C}$ also satisfies $\mathsf{InvVarConstraint}(s)$, $\mathsf{InvVarFrame}(s)$ (and $\mathsf{InvDedsub}$ when $s = m$).

Lemma C.26. *Let (M, M') be a pair of matrices of constraint systems satisfying $\mathsf{PP1SbE}(s, k)$. For all pair of matrices of constraint systems (M_1, M_1') obtained at the end of Step c of the first phase with support s and column k on (M, M'), (M_1, M_1') satisfies $\mathsf{PP1ScE}(s, k)$.*

Proof. Let (M_0, M_0') be the pair of matrices of constraint systems, ancestor of (M, M'), obtained at the end of step b. Thanks to Lemma C.25, we know that (M_0, M_0') satisfies $\mathsf{InvMatrix}(s-1)$. Furthermore, we know that for all constraint system $\mathcal{C}$ in the k^{th} column of (M_0, M_0'), $\mathcal{C}$ satisfies

$\mathsf{InvGeneral}$, $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvNoUse}(s)$, $\mathsf{InvDest}(s)$ and $\mathsf{InvUntouched}(s)$. Hence thanks to Lemmas C.19, C.15, C.16 and C.17, we deduce, by a simple induction on the size of the branch between (M_0, M_0') and (M, M'), that (M, M') satisfes $\mathsf{InvMatrix}(s-1)$ and for all constraint system $\mathcal{C}$ in the k^{th} column of (M, M'), $\mathcal{C}$ satisfies $\mathsf{InvGeneral}$, $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvNoUse}(s)$, $\mathsf{InvDest}(s)$ and $\mathsf{InvUntouched}(s)$.

It remains to prove that for all $i \in \{1, \ldots, n\}$, for all $(X, j \overset{?}{\vdash} u) \in D(M_{i,k})$, if $X \notin S_2(\mathcal{C})$ then $u \notin \mathcal{X}^1$. Let's denote $\mathcal{C} = M_{i,k}$. Let $(X, j \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $X \notin S_2(\mathcal{C})$ and $u \in \mathcal{X}^1$. Thus we have that $\mathsf{X}^1(\mathcal{C}) \neq \emptyset$. Thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 10) and Lemma C.10, we know that there exists $(Y, \ell \overset{?}{\vdash} v) \in D(\mathcal{C})$ such that $Y \in S_2(\mathcal{C})$, $\ell < j$ and $u \in vars^1(v)$.

Assume first that $v \in \mathcal{X}^1$ and so $u = v$. Thanks to Lemma C.25 and the fact that the rules applied in step c do not add second order inequation in Er with a variable not in S_2, we deduce that for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \nvDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$. But if there exists $\mathsf{f} \in \mathcal{F}_c$ such that $Er(\mathcal{C}) \vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$, then it would implies that either a rule Axiom or Cons would be applicable on $(Y, \ell \overset{?}{\vdash} v)$; or else $\mathcal{C} = \bot$ by normalisation. Hence, we have that for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \nvDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$. Therefore we have that Eq-right-right(X, Y) is applicable which contradicts the fact that (M, M') was obtained at the end of step c.

Similarly, if $v \notin \mathcal{X}^1$, it implies that either $\mathcal{C} = \bot$ by normalisation or that a rule Axiom or Cons would be applicable on $(Y, \ell \overset{?}{\vdash} v)$, which contradicts our hypothesis.

Thus we deduce that for all $i \in \{1, \ldots, n\}$, for all $(X, j \overset{?}{\vdash} u) \in D(M_{i,k})$, if $X \notin S_2$ then $u \notin \mathcal{X}^1$. □

C.4.3.5 Invariants of Phase 1, end of cycle Step b - c

Property C.7. *We say that a pair of matrices of constraint systems (M, M') satisfy $\mathsf{PP1SbcE}(s, k)$ if M and M' have the same structure, satisfy $\mathsf{InvMatrix}(s-1)$ and $\mathsf{InvGeneral}$, and for all constraint system $\mathcal{C}$ in M or M', if $\mathcal{C} \neq \bot$ then $\mathcal{C}$ satisfies the invariants $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvDest}(s)$, $\mathsf{InvNoUse}(s)$ and $\mathsf{InvUntouched}(s)$. Moreover, for all constraint system $\mathcal{C}$ in the k^{th} column of (M_1, M_1'), for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$,*

- *$X \in S_2(\mathcal{C})$*
- *for all $\mathsf{f} \in \mathcal{F}_c$, for all $\xi \in \Pi_n$, $Er \nvDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er \nvDash X \overset{?}{\neq} \xi$*
- *if $s = m$ then $\mathcal{C}$ satisfies $\mathsf{InvDedsub}$.*

At last, for all $i \leq k$, for all constraint system $\mathcal{C}$ in the i^{th} column of (M_1, M_1'), $\mathcal{C}$ also satisfies $\mathsf{InvVarConstraint}(s)$ and $\mathsf{InvVarFrame}(s)$ (and $\mathsf{InvDedsub}$ when $i = s$).

Lemma C.27. *Let (M, M') be a pair of matrices of constraint systems satisfying $\mathsf{PP1Sb}(s, k)$. For all pair of matrices of constraint systems (M_1, M_1') obtained at the end of the cycle of steps $b+c$ of the first phase with support s and column k on (M, M'), (M_1, M_1') satisfies $\mathsf{PP1SbcE}(s, k)$.*

Proof. (M, M') being obtained at the end of cycle of steps $b+c$ of Phase 1 implies that (M, M') is also obtained at the end of step c. Hence thanks to Lemma C.26, we know that (M, M') satisfies $\mathsf{InvMatrix}(s-1)$ and for all constraint system $\mathcal{C}$ in the k^{th} of (M, M'), we have that $\mathcal{C}$ satisfies the invariants $\mathsf{InvGeneral}$, $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvNoUse}(s)$, $\mathsf{InvDest}(s)$ and $\mathsf{InvUntouched}(s)$. Hence it remains to prove that for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, $X \in S_2(\mathcal{C})$.

Assume that there exists $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $X \notin S_2(\mathcal{C})$. Thus, thanks to Lemma C.26, we have that $u \notin \mathcal{X}^1$. But thanks to Lemma C.25, we know that if no rule of step b is applicable than it would imply that $u \in \mathcal{X}^1$ which is a contradiction. Hence a rule of step b is applicable on (M, M') which contradicts the fact that (M, M') is obtained at the end of the cycle step $b + c$. Hence we have that $X \in S_2(\mathcal{C})$. □

C.4.3.6 Invariant of Phase 1, Step d

Lemma C.28. *Let (M, M') be a pair of matrices of constraint systems satisfying* $\mathsf{PP1SbcE}(s, k)$. *For all pair of matrices of constraint systems (M_1, M_1') obtained at the end of the Step d of the first phase with support s and column k on (M, M'), (M_1, M_1') satisfies* $\mathsf{PP1Sb}(s, k+1)$.

Proof. Let (M_1, M_1') be the pair of matrices of constraint systems, ancestor of (M, M'), obtained at the end of the cycle $b + c$. Thanks to Lemma C.26, we already know that (M_1, M_1') satisfy $\mathsf{InvMatrix}(s-1)$ and $\mathsf{InvGeneral}$. Furthermore, for all constraint system $\mathcal{C}$ in the k^{th} column on (M_1, M_1'), we have that $\mathcal{C}$ satisfies the invariants $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvNoUse}(s)$, $\mathsf{InvDest}(s)$ and $\mathsf{InvUntouched}(s)$. But thanks to Lemmas C.19, C.17, C.16, C.15 and C.14, we have that (M, M') satisfies $\mathsf{InvMatrix}(s-1)$ and $\mathsf{InvGeneral}$. Furthermore, for all constraint system $\mathcal{C}$ in the k^{th} column on (M, M'), we have that $\mathcal{C}$ satisfies the invariants $\mathsf{InvVarFrame}(s-1)$, $\mathsf{InvNoUse}(s)$, $\mathsf{InvDest}(s)$ and $\mathsf{InvUntouched}(s)$.

Using a similar proof as in Lemma C.25, we also show that $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$ and for all $(X, i \rhd x) \in D(\mathcal{C})$, for all $(\xi, j \rhd u) \in \Phi(\mathcal{C})$, for all $\mathsf{f} \in \mathcal{F}_c$, $Er \not\vdash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$ and $Er \not\vdash X \stackrel{?}{\neq} \xi$.

Hence it remains to prove that $\mathcal{C}$ satisfies the invariant $\mathsf{InvVarFrame}(s)$. Let $(\xi, s \rhd v) \in \Phi(\mathcal{C})$ and $Z \in vars^2(\xi)$. Thanks to $\mathcal{C}$ being well formed, we know that there exists $j \leq s$ and a term u such that $(Z, j \stackrel{?}{\vdash} u) \in D(\mathcal{C})$. Furthermore, for all $x \in vars^1(u)$, there exists $(\zeta, k \rhd w) \in \Phi$ such that $k \leq s$ and $x \in vars^1(w)$. But since $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$, we know that $u \in \mathcal{X}^1$ and so $x = u$. But by the property of origination of a constraint system, we deduce that there exists $(X, q \stackrel{?}{\vdash} t) \in D$ with $q < k \leq s$ and $u \in vars^1(t)$. Once again since $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(s)$, we deduce that $t = u$. Moreover, the invariant $\mathsf{InvVarConstraint}(s)$ stipulates that all right hand term of the deducible constraints with index inferior to s are distinct. Hence, we deduce that $(X, q \stackrel{?}{\vdash} t)$ and $(Z, j \stackrel{?}{\vdash} u)$ are the same constraint and so $q = j$. But we proved that $q < k$ and $k \leq s$ which means that $j < s$ and so the result holds. □

C.4.3.7 Invariant of Phase 1, Step e

Lemma C.29. *Let (M, M') be a pair of matrices of constraint systems obtained by following the strategy. Assume that M and M' satisfy the invariant* $\mathsf{InvGeneral}$. *Let $\mathcal{C}$ and $\mathcal{C}'$ be two constraint systems occurring in the same column of M. Assume that $\mathcal{C}$ and $\mathcal{C}'$ satisfy the invariants* $\mathsf{InvVarConstraint}(s)$ *and* $\mathsf{InvUntouched}(s)$ *for some s. We have that there exists a variable renaming $\rho : \mathcal{X}^1 \setminus S_1(\mathcal{C}) \to \mathcal{X}^1 \setminus S_1(\mathcal{C}')$ such that:*

1. $\mathsf{mgu}(Eq(\mathcal{C}))_{|S_1(\mathcal{C})}\rho = \mathsf{mgu}(Eq(\mathcal{C}'))_{|S_1(\mathcal{C}')}$, *and* $D(\mathcal{C})\rho = D(\mathcal{C}')$;
2. $\{(u\rho, u') \mid (\xi, i \rhd u) \in \Phi \wedge (\xi', i' \rhd u') \in \Phi' \wedge \mathsf{path}(\xi) = \mathsf{path}(\xi')\} \subseteq \{(u, u) \mid u \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)\}$.

Proof. First, we define the renaming ρ, and then we show that the two properties are satisfied.

Definition of the renaming ρ. By Lemma 8.1, we know that the matrices M and M' have the same structure, and so the systems $\mathcal{C}$ and $\mathcal{C}'$ have the same shape. Hence, we have:

— $S_2(\mathcal{C}) = S_2(\mathcal{C}')$, and

— $\{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}) \text{ and } X \in S_2(\mathcal{C})\} = \{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}') \text{ and } X \in S_2(\mathcal{C}')\}$.

Since the system $\mathcal{C}$ satisfies the invariants $\mathsf{InvVarConstraint}(s)$ and $\mathsf{InvUntouched}(s)$, we have that $X \in S_2(\mathcal{C})$ for each $(X, i \stackrel{?}{\vdash} u) \in D(\mathcal{C})$, and similarly, we have that $X \in S_2(\mathcal{C}')$ for each $(X, i \stackrel{?}{\vdash} u) \in D(\mathcal{C}')$. This allows us to conclude that $\{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C})\} = \{(X, i) \mid X, i \stackrel{?}{\vdash} u \in D(\mathcal{C}')\}$. Actually, the invariant $\mathsf{InvVarConstraint}(s)$ also tells us that:

— for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $i \leq s$, we have that u is a variable (distinct of the ones introduced by the other constraints); and

— for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}')$ such that $i \leq s$, we have that u is a variable (distinct of the ones introduced by the other constraints).

Hence, this allows us to define a renaming ρ such that $\mathrm{dom}(\rho) = \{x \mid (X, i \overset{?}{\vdash} x) \in D(\mathcal{C}) \wedge i \leq s\}$, and $\rho(x) = X\mathsf{acc}^1(\mathcal{C}')$ where $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$.

Property 1. With such renaming, we trivially have that $D(\mathcal{C})\rho = D(\mathcal{C}')$ but only for the deducibility constraints $(X, i \rhd u)$ with $i \leq s$. Hence, we still have to prove this result for $i > s$. Since the systems $\mathcal{C}$ and $\mathcal{C}'$ occur on the same column of the matrix M, there exists an initial constraint system $\mathcal{C}_0$ that is an ancestor of $\mathcal{C}$ and $\mathcal{C}'$. Moreover, we know that $\mathcal{C}$ and $\mathcal{C}'$ satisfy the invariant $\mathsf{InvUntouched}(s)$. Hence, we deduce that:

— for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ such that $i > s$, we have that $X \in vars^2(D(\mathcal{C}_0))$; and

— for all $(X, i \overset{?}{\vdash} u') \in D(\mathcal{C}')$ such that $i > s$, we have that $X \in vars^2(D(\mathcal{C}_0))$.

Let $\sigma = \mathsf{mgu}(Eq(\mathcal{C}))$ and $\sigma' = \mathsf{mgu}(Eq(\mathcal{C}'))$. Since $\mathcal{C}$ and $\mathcal{C}'$ are normalised, for $i > s$, we deduce that $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, $(X, i \overset{?}{\vdash} u') \in D(\mathcal{C}')$, and $(X, i \overset{?}{\vdash} u_0) \in D(\mathcal{C}_0)$ imply that $u = u_0\sigma$ and $u' = u_0\sigma'$. Let $S_1 \overset{\mathsf{def}}{=} S_1(\mathcal{C}) = S_1(\mathcal{C}') = S_1(\mathcal{C}_0)$. Hence, to conclude the proof of $D(\mathcal{C})\rho = D(\mathcal{C}')$, it remains to show that $\sigma_{|S_1}\rho = \sigma'_{|S_1}$.

By definition of an initial constraint system, we know that for all $x \in S_1$, there exists $(X, k \overset{?}{\vdash} u) \in D(\mathcal{C}_0)$ such that $x \in vars^1(u)$ and $X \in S_2(\mathcal{C}_0)$. Since $\mathcal{C}$ and $\mathcal{C}'$ satisfy the invariant $\mathsf{InvUntouched}(s)$, we have that no rule was applied with support strictly superior to s, and we deduce that for all $(Y, j \overset{?}{\vdash} v) \in D(\mathcal{C}_0)$, for all $y \in vars^1(v)$, $\mathsf{ind}_{\mathcal{C}_0}(y) > s$, we have that $y \notin \mathrm{dom}(\sigma)$ and $y \notin \mathrm{dom}(\sigma')$. Hence, we only focus on variable $x \in S_1$ such that there exists $(X, k \overset{?}{\vdash} u) \in D(\mathcal{C}_0$, $x \in vars^1(u)$ and $k \leq s$. We prove by induction on $k \leq s$ that $X\mathsf{acc}^1(\mathcal{C}_0)\sigma\rho = X\mathsf{acc}^1(\mathcal{C}_0)\sigma'$.

Base case $k = 0$. There is no constraint $X, k \overset{?}{\vdash} u$ with $k = 0$. Hence, the result trivially holds.

Inductive step $k > 0$. Let $(X, k \overset{?}{\vdash} u) \in D(\mathcal{C}_0)$. By Lemma C.11, we know that:

$$\begin{cases} u\sigma = \mathsf{C}\lfloor X\rfloor_{\Phi_0}\mathsf{acc}^1(\mathcal{C}_0)\sigma = \mathsf{C}\lfloor X\Theta\rfloor_{\Phi}\mathsf{acc}^1(\mathcal{C}) \text{ and } \mathsf{param}^{\leq}_{\mathsf{max}}(X\Theta)k \\ u\sigma' = \mathsf{C}\lfloor X\rfloor_{\Phi_0}\mathsf{acc}^1(\mathcal{C}_0)\sigma' = \mathsf{C}\lfloor X\Theta'\rfloor_{\Phi'}\mathsf{acc}^1(\mathcal{C}') \text{ and } \mathsf{param}^{\leq}_{\mathsf{max}}(X\Theta')k \end{cases}$$

where $\Theta = \mathsf{mgu}(Er(\mathcal{C}))$ and $\Theta' = \mathsf{mgu}(Er(\mathcal{C}'))$.

However, $X \in S_2(\mathcal{C}_0)$ implies that $X \in S_2(\mathcal{C}) = S_2(\mathcal{C}')$ and so thanks to M satisfying the invariant $\mathsf{InvGeneral}$ (item 5), we deduce that $\mathsf{C}\lfloor X\Theta\rfloor_{\Phi} = \mathsf{C}\lfloor X\Theta'\rfloor_{\Phi'}$. Furthermore, for all $i \leq k$, for all $w \cdot ax_i \in st(\mathsf{C}\lfloor X\Theta\rfloor_{\Phi})$, we know that there exists u_0 such that $(ax_i, i \rhd u_0) \in \Phi(\mathcal{C}_0)$. Thus by Lemma C.12, $(ax_i, i \rhd u_0\sigma) \in \Phi$ and $(ax_i, i \rhd u_0\sigma') \in \Phi'$. But by definition of a constraint system, for all $y \in vars^1(u_0)$, there exists $(Y, \ell \overset{?}{\vdash} v) \in D(\mathcal{C}_0)$ such that $\ell < k$ and $y \in vars^1(v)$. By our inductive hypothesis, we know that $v\sigma\rho = v\sigma'$ which implies that $y\sigma\rho = y\sigma'$ and so we can deduce that $u_0\sigma\rho = u_0\sigma'$.

But thanks to the definition of a context, for all w, $w \cdot ax_i \in st(\mathsf{C}\lfloor X\Theta\rfloor_{\Phi}) = st(\mathsf{C}\lfloor X\Theta\rfloor_{\Phi'})$ implies that there exists $(\xi, j \rhd v) \in \Phi$ and $(\xi', j' \rhd v') \in \Phi'$ such that $\mathsf{path}(\xi) = \mathsf{path}(\xi') = w \cdot ax_i$. Since $ax_i\mathsf{acc}^1(\mathcal{C})\rho = u_0\sigma\rho = u_0\sigma' = ax_i\mathsf{acc}^1(\mathcal{C}')$, then thanks to Lemma C.13, we can deduce that $v\rho = v'$ and so $(w \cdot ax_i)\mathsf{acc}^1(\mathcal{C})\rho = (w \cdot ax_i)\mathsf{acc}^1(\mathcal{C}')$.

At last, since $\mathsf{param}^{\leq}_{\mathsf{max}}(X\Theta)k$ and $\mathsf{param}^{\leq}_{\mathsf{max}}(X\Theta')k$, then for all $(w \cdot ax_i) \in st(\mathsf{C}\lfloor X\Theta\rfloor_{\Phi})$, we have $i \leq k$. The same holds for $(w \cdot ax_i) \in st(\mathsf{C}\lfloor X\Theta'\rfloor_{\Phi'})$. Hence, since we proved that for all w, for all $i \leq k$, $(w \cdot ax_i)\mathsf{acc}^1(\mathcal{C})\rho = (w \cdot ax_i)\mathsf{acc}^1(\mathcal{C}')$, since $\mathsf{C}\lfloor X\Theta\rfloor_{\Phi} = \mathsf{C}\lfloor X\Theta'\rfloor_{\Phi'}$, and since for

all $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, for all $(X, i \overset{?}{\vdash} x') \in D(\mathcal{C}')$, $i \leq s$ implies $x\rho = x'$, then we can deduce that $\mathsf{C}\lfloor X\Theta \rfloor_\Phi \mathsf{acc}^1(\mathcal{C})\rho = \mathsf{C}\lfloor X\Theta' \rfloor_\Phi \mathsf{acc}^1(\mathcal{C}')$. Thus we conclude that $X\mathsf{acc}^1(\mathcal{C}_0)\sigma\rho = u\sigma\rho = u\sigma' = X\mathsf{acc}^1(\mathcal{C}_0)\sigma'$.

Property 2. Let $(\xi, i \rhd u) \in \Phi$ and $(\xi', i' \rhd u') \in \Phi'$ such that $\mathsf{path}(\xi) = \mathsf{path}(\xi') = w \cdot ax_k$. Since the constraint systems $\mathcal{C}$ and $\mathcal{C}'$ are well-formed, there exist $(ax_k, k \rhd v) \in \Phi$ and $(ax_k, k \rhd v') \in \Phi'$. Thanks to Lemma C.12, we know that there exits v_0 such that $(ax_k, k \rhd v_0) \in \Phi_0$ with $v_0\sigma = v$ and $v_0\sigma' = v'$. Since $\sigma_{|S_1}\rho = \sigma'_{|S_1}$, we deduce that $v_0\sigma\rho = v_0\sigma'$ and so $v\rho = v'$. □

Lemma C.30. *Let (M, M') be a pair of matrices of constraint systems satisfying $\mathsf{PP1}(s)$. For all pair of matrices of constraint systems (M_1, M_1') obtained by applying all the steps of Phase 1 of the strategy with support s, (M_1, M_1') satisfies $\mathsf{PP1}(s+1)$.*

Proof. The step e of the strategy consists of transforming some of the constraint systems into $\bot$. Moreover, the step e is applied only once step d was applied on all columns of (M, M'). Thus, thanks to Lemma C.28, we can already deduce that M and M' satisfies $\mathsf{InvGeneral}$ and for all constraint system $\mathcal{C}$ in (M, M'), $\mathcal{C}$ satisfies the invariants $\mathsf{InvVarConstraint}(s)$, $\mathsf{InvVarFrame}(s)$, $\mathsf{InvNoUse}(s)$, $\mathsf{InvDest}(s)$, $\mathsf{InvUntouched}(s)$, and for all $(X, i \rhd x) \in D(\mathcal{C})$, for all $(\xi, j \rhd u) \in \Phi(\mathcal{C})$, for all $\mathsf{f} \in \mathcal{F}_c$, $Er \nVdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er \nVdash X \overset{?}{\neq} \xi$.

Similarly, we have that (M, M') satisfies the invariant $\mathsf{InvGeneral}$ and $\mathsf{InvMatrix}(s-1)$. Thus it remains to prove that (M, M') satisfies the invariant $\mathsf{InvMatrix}(s)$. But by the definition of the transformation in step e, we can deduce that for all $\mathcal{C}$, for all $\mathcal{C}'$ in the same column of (M, M'), we have $\{\mathsf{path}(\xi), i \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}) \wedge i \leq s\} = \{\mathsf{path}(\xi), i \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}') \wedge i \leq s\}$.

At last, let (M_1, M_1') be the matrices ancestors of (M, M') obtained at the end of Step a of phase 1 with support s. Let $\mathcal{C}$ be a constraint system in (M, M') such that $(\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C})$. Let $\mathcal{C}'$ be a constraint system in (M, M') in the column on $\mathcal{C}$. We proved that there exists $(\xi', s \rhd u') \in \Phi(\mathcal{C}')$ such that $\mathsf{path}(\xi') = \mathsf{path}(\xi)$. Assume that $(\xi', s \rhd u') \notin \Phi(\mathcal{C}')$. Since the frame is not modify during step $b - c - d$, other than applying substitution, we can deduce that there exists $\mathcal{C}_1, \mathcal{C}_1'$ in (M_1, M_1'), $\xi_1, \xi_1' \in \Pi_n$, $u_1, u_1' \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$ such that $\mathcal{C}_1 \to^* \mathcal{C}$, $\mathcal{C}_1' \to^* \mathcal{C}'$, $(\xi_1, s \rhd u_1) \in \mathsf{NoUse}(\mathcal{C}_1)$, $(\xi_1', s \rhd u_1') \in \Phi(\mathcal{C}_1') \setminus \mathsf{NoUse}(\mathcal{C}_1')$ and $\mathsf{path}(\xi_1) = \mathsf{path}(\xi_1')$

Thanks to Lemma C.22, there exists $X \in S_2(\mathcal{C}_1)$ such that $\mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C}_1)) \rfloor_{\Phi(\mathcal{C}_1)} \mathsf{acc}^1(\mathcal{C}_1) = u_1$ and either (a) $(\xi_1', s \rhd u_1') \in \mathsf{NoUse}(\mathcal{C}'')$ and $\mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C}_1')) \rfloor_{\Phi(\mathcal{C}_1')} \mathsf{acc}^1(\mathcal{C}_1') = u_1'$. Or else, (b) by denoting $v_1' = \mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C}_1')) \rfloor_{\Phi(\mathcal{C}_1')} \mathsf{acc}^1(\mathcal{C}_1')$, we have that $Eq(\mathcal{C}_1') \vDash v_1' \overset{?}{\neq} u_1'$. We show that case (b) can not happen.

Let's denote $\sigma = \mathsf{mgu}(Eq(\mathcal{C}))$, $\sigma' = \mathsf{mgu}(Eq(\mathcal{C}'))$, $\theta = \mathsf{mgu}(Er(\mathcal{C}))$ and $\theta' = \mathsf{mgu}(Er(\mathcal{C}'))$. Thanks to Lemma C.11, we deduce that $v_1'\sigma' = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C}')} \mathsf{acc}^1(\mathcal{C}')$ and $u_1\sigma = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C})} \mathsf{acc}^1(\mathcal{C})$. However, since $X \in S_2(\mathcal{C}_1)$ and (M, M') have the same structure, we deduce that $X \in S_2(\mathcal{C})$ and $X \in S_2(\mathcal{C}')$. But (M, M') satisfies the invariant $\mathsf{InvGeneral}$, hence thanks to Property 5 of invariant $\mathsf{InvGeneral}$, we have that $\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C}')} = \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}$.

On the other hand, since $\mathcal{C}$ and $\mathcal{C}'$ satisfies $\mathsf{InvVarConstraint}(s)$ and $\mathsf{InvUntouched}(s)$, then by Lemma C.29, we have that there exists a variable renaming $\rho : \mathcal{X}^1 \setminus S_1(\mathcal{C}) \to \mathcal{X}^1 \setminus S_1(\mathcal{C}')$ such that:

1. $\mathsf{mgu}(Eq(\mathcal{C}))_{|S_1(\mathcal{C})}\rho = \mathsf{mgu}(Eq(\mathcal{C}'))_{|S_1(\mathcal{C}')}$, and $D(\mathcal{C})\rho = D(\mathcal{C}')$;
2. $\{(u\rho, u') \mid (\xi, i \rhd u) \in \Phi \wedge (\xi', i' \rhd u') \in \Phi' \wedge \mathsf{path}(\xi) = \mathsf{path}(\xi')\}$ is include in $\{(u, u) \mid u \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)\}$;

Thus we have that $\mathsf{acc}^1(\mathcal{C})\rho = \mathsf{acc}^1(\mathcal{C}')$ and so $v_1'\sigma' = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C}')} \mathsf{acc}^1(\mathcal{C}') = \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})} \mathsf{acc}^1(\mathcal{C})\rho = u_1\sigma\rho = u\rho = u' = u_1'\sigma'$. Hence, we have that $v_1'\sigma' = u_1'\sigma'$. However, we assume that $Eq(\mathcal{C}_1') \vDash v_1' \overset{?}{\neq} u_1'$ which implies that $Eq(\mathcal{C}') \vDash v_1'\sigma' \overset{?}{\neq} u_1'\sigma'$. But $v_1'\sigma' = u_1'\sigma'$ and by the normalisation, we would have that $\mathcal{C}' = \bot$ which is a contradiction with our hypothesis. Hence, we proved that only case (a) can happen which implies that $(\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C})$ implies $(\xi', s \rhd u') \in \mathsf{NoUse}(\mathcal{C}')$. It allows us to conclude that $\{\mathsf{path}(\xi), i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}) \wedge i \leq s\} = \{\mathsf{path}(\xi), i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}') \wedge i \leq s\}$. □

C.4.3.8 All invariants

Property C.8. *Let (M, M') be a pair of matrices of constraint system. We say that (M, M') satisfies* PP1E *if (M, M') satisfies* PP1(∞) *and for all constraint system $\mathcal{C}$ in M or M', $\mathcal{C}$ also satisfies* InvDedsub.

Lemma C.31. *Let (M, M') be a pair of row matrices of initial constraint systems having the same structure. For all (M'_1, M'_1) obtained at the end of phase 1 of the strategy from (M, M'), (M, M') satisfies* PP1E. *Moreover, for all (M_2, M'_2) such that $(M, M') \to^* (M_2, M'_2) \to^* (M'_1, M'_1)$,*

- *if (M_2, M'_2) is obtained from Step a with support s then (M_2, M'_2) satisfies* PP1Sa(s);
- *if (M_2, M'_2) is obtained at the end of Step a with support s then (M_2, M'_2) satisfies* PP1SaE(s);
- *if (M_2, M'_2) is obtained at the end Step b with support s and column k then (M_2, M'_2) satisfies* PP1SbE(s, k);
- *if (M_2, M'_2) is obtained at the end Step c with support s and column k then (M_2, M'_2) satisfies* PP1ScE(s, k);
- *if (M_2, M'_2) is obtained at the end of the cycle of steps $b + c$ with support s and column k then (M_2, M'_2) satisfies* PP1SbcE(s, k);
- *if (M_2, M'_2) is obtained at the end of Step d with support s then (M_2, M'_2) satisfies* PP1Sb($s, k + 1$).

Proof. Simple induction on the parameter s and k that rely on all the previous lemmas of this subsection. □

Lemma 8.3. *Let (M, M') be a pair of row matrices of initial constraint systems having the same structure. Let (M_1, M'_1) be a pair of matrices of constraint systems obtained by following the strategy on (M, M'). (M_1, M'_1) satisfies* InvGeneral.

Proof. We rely on Lemma C.31 to prove that if (M_1, M'_1) is obtained from Step a with support s, $s \in \mathbb{N}$, then (M_1, M'_1) satisfies PP1Sa(s) and so InvGeneral. For any other step and phase, we rely on Lemma C.19 to conclude. □

C.4.3.9 Invariant of Phase 2, Step a

Lemma C.32. *Let (M, M') be a pair of matrices of constraint systems obtained at the end of Step a of the second phase of the strategy. We have that for all constraint system $\mathcal{C}$ in (M, M'), $vars^1(Eq(\mathcal{C}))$ do not contain universal variable.*

Proof. Thanks to the normalisation, we know that for all constraint system $\mathcal{C}$ in M, the disjunctions of inequations in $Eq(\mathcal{C})$ are of the form $\forall \tilde{y}. \bigvee_i x_i \stackrel{?}{\neq} u_i$ where $\tilde{y}$ is a set of universal variable and x_i are not universal for any i. Furthermore, thanks to the normalisation and more specifically by rule (Nelim1), $x_i \notin \tilde{y}$, for all i and by rule (Nelmin2), for all $y \in \tilde{y}$, there exists i such that $y \in vars^1(u_i)$. Let $x_i \stackrel{?}{\neq} u_i$ and $y \in vars^1(u_i) \cap \tilde{y}$. Since x_i is not a universal variable, there exists $(X, j \stackrel{?}{\vdash} x_i) \in D(\mathcal{C})$. But we assumed that the rules Cons and Axiom are no longer applicable. Thus, we deduce that for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$ and for all $(\xi, k \stackrel{?}{\vdash} u) \in \Phi(\mathcal{C})$, $Er(\mathcal{C}) \vDash X \stackrel{?}{\neq} \xi$. Moreover, we know that $\mathcal{C}$ satisfies the invariant InvDest(∞) hence by the definition of the normalisation, we should have $\mathcal{C} = \bot$ which is a contradiction with our hypothesis on $\mathcal{C}$. Hence $vars^1(u_i) \cap \tilde{y} = \emptyset$ for all $x_i \stackrel{?}{\neq} u_i$. Hence by rule (Nelim2) of the normalisation, we deduce that $Eq(\mathcal{C})$ do not contain universal variable. □

C.4.3.10 Invariant of Phase 2, Step b

Lemma C.33. *Let (M, M') be a pair of matrices of constraint systems at the end of Step b. For all constraint system $\mathcal{C}$ and its association table in M or M', we have that:*

- *for all disjunction $\bigvee_i u_i \overset{?}{\neq} v_i$, if $Eq(\mathcal{C}) = E \wedge \bigvee_i u_i \overset{?}{\neq} v_i$ and $T[\bigvee_i u_i \overset{?}{\neq} v_i] = \bot$, then for all i, $u_i \overset{?}{\neq} v_i$ satisfies one of the following properties:*
  1. *$u_i \in \mathcal{X}^1$ and $v_i \in \mathcal{N}$.*
  2. *$u_i, v_i \in \mathcal{X}^1$, $Er(\mathcal{C}) \not\vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}) \vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{g}$, for all $\mathsf{f}, \mathsf{g} \in \mathcal{F}_c$, where $(X, p \overset{?}{\vdash} u_i), (Y, q \overset{?}{\vdash} v_i) \in D(\mathcal{C})$.*
  3. *$u_i \in \mathcal{X}^1$, $\mathsf{root}(v_i) \in \mathcal{F}_c$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$, where $(X, p \overset{?}{\vdash} u_i) \in D(\mathcal{C})$.*
- *for all disjunction D, if $Eq(\mathcal{C}) = E \wedge D$ and $T[D] = \bigvee_i \xi_i \overset{?}{\neq} \xi'_i$, then either for all i, $st(\xi_i, \xi'_i) \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) = \emptyset$ or else for all i, $\xi_i \overset{?}{\neq} \xi'_i$ satisfies one of the following properties:*
  1. *$\xi_i \in (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X})$*
  2. *$\xi_i, \xi'_i \in \mathcal{X}^2$, for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(\xi_i) \overset{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}) \not\vDash \mathsf{root}(\xi'_i) \overset{?}{\neq} \mathsf{f}$.*
  3. *$\xi_i \in \mathcal{X}^2$, $\mathsf{root}(\xi'_i) \in \mathcal{F}_c$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(\xi_i) \overset{?}{\neq} \mathsf{f}$*

Proof. By definition of Step b oh Phase 2 of the strategy, for all constraint system $\mathcal{C}$ and its association table T, Cons(X, f) and Eq-right-right(X, ξ) is not applicable for all $\mathsf{f} \in \mathcal{F}_c$ and all ξ.

Consider a disjunction $\bigvee_i^n u_i \overset{?}{\neq} v_i$ such that $Eq(\mathcal{C}) = E \wedge \bigvee_i^n u_i \overset{?}{\neq} v_i$ and $T[\bigvee_i^n u_i \overset{?}{\neq} v_i] = \bot$. Let $i \in \{1, \ldots, n\}$. We do a case analysis on u_i and v_i.

- $u_i \in \mathcal{X}^1$ and $v_i \in \mathcal{N}$ (or the reverse): The results holds trivially.
- $u_i \notin \mathcal{X}^1$ and $v_i \in \mathcal{N}$ (or the reverse): We know that $\mathcal{C}$ is normalised. Hence such inequality is necessary reduced either by the rules (Nneq1) or (Nt1). Hence this case is impossible.
- $u_i, v_i \in \mathcal{X}^1$: In such a case, there exists $(X, k \overset{?}{\vdash} u_i) \in D(\mathcal{C})$ and $(Y, \ell \overset{?}{\vdash} v_i) \in D(\mathcal{C})$. Assume w.l.o.g. that $\ell \leq k$. Since Eq-right-right is not applicable for step b, we deduce that the conditions of application of the rule Eq-right-right(X, Y) in Figure 7.2 are not satisfied. Hence we deduce that there exists $\mathsf{f} \in \mathcal{F}_c$ such that $Er \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er \not\vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$ (or the reverse). Assume w.l.o.g. that $Er \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er \not\vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$.

  Since Cons(X, g) is not applicable for Step b, for all $\mathsf{g} \in \mathcal{F}_c$, $Er \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ implies that $Er \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$ for all $\mathsf{g} \in \mathcal{F}_c$. Similarly, $Er \not\vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$ implies that $Er \not\vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{g}$ for all $\mathsf{g} \in \mathcal{F}_c$ (otherwise Cons(Y, g) would be applicable). Hence we deduce that for all $\mathsf{f}, \mathsf{g} \in \mathcal{F}_c$, $Er(\mathcal{C}) \not\vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}) \vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{g}$. Thus the result holds.
- Otherwise, $u_i \in \mathcal{X}^1$ and $\mathsf{root}(v_i) = \mathsf{f} \in \mathcal{F}_c$ (or the reverse): $u_i \in \mathcal{X}^1$ implies that there exist $(X, k \overset{?}{\vdash} u_i) \in D(\mathcal{C})$. Since Cons$(X, \mathsf{g})$ is not applicable for Step b, for all $\mathsf{g} \in \mathcal{F}_c$, we deduce that $st(v_i) \cap \mathcal{N} = \emptyset$, $\mathsf{ind}_{\mathcal{C}}(v_i) \leq k$ and either for all $\mathsf{g} \in \mathcal{F}_c$, $Er \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$; or for all $\mathsf{g} \in \mathcal{F}_c$, $Er \not\vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$.
  $st(v_i) \cap \mathcal{N} = \emptyset$ implies that there exists $\xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$ such that $\xi\mathsf{acc}^1(\mathcal{C}) = v_i$. But Eq-right-right$(X, \xi)$ is not applicable for Step b. Hence, we deduce that $\mathsf{ind}_{\mathcal{C}}(v_i) \leq k$ implies that $Er \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{root}(\xi)$ where $\mathsf{root}(\xi) \in \mathcal{F}_c$. Hence using what we proved

thanks to Cons not being applicable, we deduce that for all $\mathsf{g} \in \mathcal{F}_c$, $Er \vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{g}$. Hence the result holds.

Consider now a disjunction D such that $Eq(\mathcal{C}) = E \wedge D$ for some E, $T[D] = \bigvee_i^n \xi_i \stackrel{?}{\neq} \xi_i'$ and there exists $j \in \{1, \ldots, n\}$ such that $st(\xi_j, \xi_j') \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$. Let $i \in \{1, \ldots, n\}$, we do a case analysis on ξ_i and ξ_i':

— $\xi_i \in (\mathcal{F}_d^* \cdot \mathcal{AX})$ or $\xi_i' \in (\mathcal{F}_d^* \cdot \mathcal{AX})$: The result trivially holds.

— $\xi_i, \xi_i' \in \mathcal{X}^2$: In such a case, since Eq-right-right(ξ_i, ξ_i') is not applicable for Step b, we deduce that there exists $\mathsf{f} \in \mathcal{F}_c$ such that $Er \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$ and $Er \nvDash \mathsf{root}(\xi_i') \stackrel{?}{\neq} \mathsf{f}$ (or the reverse). Assume w.l.o.g. that $Er \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$ and $Er \nvDash \mathsf{root}(\xi_i') \stackrel{?}{\neq} \mathsf{f}$.

Since Cons(ξ_i, g) and Cons(ξ_i', g) are not applicable for Step b for all $\mathsf{g} \in \mathcal{F}_c$, $Er \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$ implies that $Er \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{g}$, for all $\mathsf{g} \in \mathcal{F}_c$. Moreover, $Er \nvDash \mathsf{root}(\xi_i') \stackrel{?}{\neq} \mathsf{f}$ implies that $Er \nvDash \mathsf{root}(\xi_i') \stackrel{?}{\neq} \mathsf{g}$ for all $\mathsf{g} \in \mathcal{F}_c$. Hence the result holds.

— $\xi_i \in \mathcal{X}^2$, $\mathsf{root}(\xi_i') = \mathsf{f} \in \mathcal{F}_c$ (or the reverse): Since Cons(ξ_i, f) is not applicable for Step b, we deduce that either (a) $Er \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$ or (b) $st(\xi) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$ and $\mathsf{paramC}_{\mathsf{max}}^{\xi_i}(\mathcal{C}) \geq \mathsf{paramC}_{\mathsf{max}}^{\xi_i'}(\mathcal{C})$. In case (a), since Cons(ξ_i, g) not applicable for Step b for all $\mathsf{g} \in \mathcal{F}_c$, then $Er \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$ implies for all $\mathsf{g} \in \mathcal{F}_c$, $Er \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{g}$. Hence the result holds. In case (b), since $st(\xi_j, \xi_j') \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$, $st(\xi) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$, $\mathsf{paramC}_{\mathsf{max}}^{\xi_i}(\mathcal{C}) \geq \mathsf{paramC}_{\mathsf{max}}^{\xi_i'}(\mathcal{C})$ and Eq-right-right(ξ_i, ξ_i') not applicable for Step b, then we deduce that $Er(\xi) \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$. But thanks to Cons$(\xi_i, \mathsf{g})$ not applicable for Step b, we conclude that $Er(\xi) \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{g}$ for all $\mathsf{g} \in \mathcal{F}_c$. Hence the result holds.

□

C.5 Proof of soundness

In this section, we will focus on the proof of soundness. However, unlike the proof of completeness, this proof depends heavily on the strategy that has been described in Section 7.4, and on the invariants described in Section C.4. Hence, the proof will depend on the strategy.

C.5.1 Preliminaries

Lemma C.34. *Let Φ be a closed frame and ξ, ξ' be two ground recipes in Π_n with $\mathsf{root}(\xi), \mathsf{root}(\xi') \notin \mathcal{F}_c$ and such that $\mathsf{path}(\xi) = \mathsf{path}(\xi')$. If $\xi\Phi\!\downarrow, \xi'\Phi\!\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$, then we have that $\xi\Phi\!\downarrow = \xi'\Phi\!\downarrow$.*

Proof. We prove this result by induction of the length n of $\mathsf{path}(\xi)$:

Base case $n = 1$: In such a case, we have that $\mathsf{path}(\xi) = \mathsf{path}(\xi') \in \mathcal{AX}$. Hence, we have that $\xi = \xi'$, and so $\xi\Phi\!\downarrow = \xi'\Phi\!\downarrow$.

Inductive step $n > 1$: Since $\mathsf{path}(\xi) = \mathsf{path}(\xi')$, we know that there exists $\mathsf{f} \in \mathcal{F}_d$ and there exist $\xi_1, \ldots, \xi_n, \xi_1', \ldots, \xi_n' \in \Pi_n$ such that $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$ and $\xi' = \mathsf{f}(\xi_1', \ldots, \xi_n')$. By Lemma 6.5, for all $\zeta \in st(\xi) \cup st(\xi')$, we have that $\zeta\Phi\!\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Hence, for all $i = 1, \ldots, n$, for all $\zeta \in st(\xi_i) \cup st(\xi_i')$, we have that $\zeta\Phi\!\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. By definition of path, we have that $\mathsf{path}(\xi_1) = \mathsf{path}(\xi_1')$ (since $\mathsf{path}(\xi) = \mathsf{path}(\xi')$).

Applying our induction hypothesis on (ξ_1, ξ_1'), we obtain that $\xi_1\Phi\!\downarrow = \xi_1'\Phi\!\downarrow$. We have that $\xi\Phi\!\downarrow, \xi'\Phi\!\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Hence, we have that

$$\mathsf{f}(\xi_1\Phi\!\downarrow, \ldots, \xi_n\Phi\!\downarrow) \to \xi\Phi\!\downarrow \quad \text{and} \quad \mathsf{f}(\xi_1'\Phi\!\downarrow, \ldots, \xi_n'\Phi\!\downarrow) \to \xi'\Phi\!\downarrow$$

using the rewriting rule associated to f. This rule is of the form $\mathsf{f}(u_1, \ldots, u_n) \to u$ with $u \in st(u_1)$. Since $\xi_1\Phi\downarrow = \xi_1'\Phi\downarrow$, we easily conclude that $\xi\Phi\downarrow = \xi'\Phi\downarrow$. □

Lemma C.35. *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system obtained by following the strategy and $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. Let $\xi \in \Pi_n$ be a ground recipe conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. For all $\xi' \in st(\xi)$, ξ' conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$.*

Proof. We prove the result by induction on $|\xi|$.

Base case $|\xi| = 1$: In such a case, $\xi \in \mathcal{AX}$. Hence for all $\xi' \in st(\xi)$, $\xi' = \xi$ and so the result trivially holds.

Inductive step $|\xi| > 1$: Otherwise, $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$. We do a case analysis on $\mathsf{C}\lfloor\xi\rfloor_\Phi$.

— Case $|\mathsf{C}\lfloor\xi\rfloor_\Phi| > 1$: In such a case, we have that $\mathsf{C}\lfloor\xi\rfloor_\Phi = \mathsf{f}(\mathsf{C}\lfloor\xi_1\rfloor_\Phi, \ldots, \mathsf{C}\lfloor\xi_n\rfloor_\Phi)$. Moreover, ξ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$ implies that for all $i \in \{1, \ldots, n\}$, ξ_i conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. Hence by inductive hypothesis on ξ_i, for all $i \in \{1, \ldots, n\}$, the result holds.

— Case $|\mathsf{C}\lfloor\xi\rfloor_\Phi| = 1$: Otherwise, since ξ conforms to $\Phi\theta$, we deduce that there exists $(\zeta, i \rhd u) \in \Phi$ such that $(\zeta, i \rhd u) \notin \mathsf{NoUse}\theta$ and $\zeta\theta = \xi$. Thanks to $\mathcal{C}$ being well-formed, (Definition 8.2, item 9), we deduce that $\mathsf{path}(\zeta)$ is closed. Hence there exists $\zeta_1, \ldots, \zeta_n$ such that $\zeta = \mathsf{f}(\zeta_1, \ldots, \zeta_n)$ and $\zeta_i\theta = \xi_i$ for all $i \in \{1, \ldots, n\}$.

But thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 9), we deduce that for all $\zeta' \in st(\zeta)$, $\mathsf{C}\lfloor\zeta'\rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX} \cup \mathcal{X}^2)$ and if $\mathsf{path}(\zeta') \in \mathcal{F}_d^* \cdot \mathcal{AX}$ then there exists j and v such that $(\zeta', j \rhd v) \in \Phi$. Hence for all $i \in \{1, \ldots, n\}$, $\zeta_i\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$ if for all $X \in vars^2(\mathsf{C}\lfloor\zeta'\rfloor_\Phi)$, $X\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$; and for all $\zeta'' \in st(\zeta_i)$, if $(\zeta'', j' \rhd v') \in \Phi$ for some j', v' then $(\zeta'', j' \rhd v') \notin \mathsf{NoUse}\theta$.

Since $(\zeta, i \rhd u) \notin \mathsf{NoUse}$, and relying on Lemma 8.3 (Item 4), we deduce that for all $\zeta'' \in st(\zeta_i)$, if $\mathsf{path}(\zeta'') \in \mathcal{F}_d^* \cdot \mathcal{AX}$ then there exists j and v such that $(\zeta'', j \rhd v) \in \Phi$ and $(\zeta'', j \rhd v) \notin \mathsf{NoUse}$. At last, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies that for all $X \in vars^2(\mathcal{C})$, $X\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. Hence we deduce that for all $i \in \{1, \ldots, n\}$, $\zeta_i\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. We conclude by applying our inductive hypothesis on $\zeta_i\theta$, for all $i \in \{1, \ldots, n\}$. □

Lemma C.36. *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system obtained by following the strategy and $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. Let ξ be a ground recipe in Π_n such that $\xi(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. We have that there exists ξ' a recipe in Π_n such that:*

— *ξ' conforms with $\Phi\theta$ w.r.t $\mathsf{NoUse}\theta$;*

— *$\xi(\Phi\sigma)\downarrow = \xi'(\Phi\sigma)\downarrow$; and*

— *$\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$.*

Proof. We prove this lemma by induction on $|\xi|$.

Base case $|\xi| = 0$: In such a case, the result trivially holds.

Inductive step $|\xi| > 0$: We do a case analysis on $\mathsf{C}\lfloor\xi\rfloor_\Phi$:

— *Case 1:* $\mathsf{C}\lfloor\xi\rfloor_\Phi \in (\mathcal{F}_d^* \cdot \mathcal{AX})$. By definition of $\mathsf{C}\lfloor\xi\rfloor_\Phi$, we know that there exists $(\zeta, i \rhd u) \in \Phi$ such that $\mathsf{path}(\zeta) = \mathsf{path}(\xi)$. Since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and thanks to Lemma 8.3 (item 2), we deduce that $i \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$. Note that $\mathcal{C}$ is a well-formed constraint system and $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, thus by Definition 8.2 (item 5), we have that $(\zeta\theta)(\Phi\sigma)\downarrow = u\sigma$. Moreover, relying on Lemma C.34, we can deduce that $\xi(\Phi\sigma)\downarrow = (\zeta\theta)(\Phi\sigma)\downarrow$.

Case 1.a : $(\zeta, i \rhd u) \in \mathsf{NoUse}$. Let $\Theta = \mathsf{mgu}(Er)$. In such a case, since $\mathcal{C}$ is a well-formed constraint system (Definition 8.2, item 8), we know that there exists $X \in vars^2(\mathcal{C})$ such that $\mathsf{C}\lfloor X\Theta\rfloor_\Phi \mathsf{acc}^1(\mathcal{C}) = u$ and $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\Theta) < i$. Since we proved that $i \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$, we can deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\Theta) < \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$.

But since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we know that $\theta \vDash Er$, $X\theta \in \Pi_n$, $X\theta$ conforms with $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$ and $(X\theta)\Phi\sigma\downarrow = (\zeta\theta)\Phi\sigma\downarrow$. Note that $\xi(\Phi\sigma)\downarrow = (\zeta\theta)(\Phi\sigma)\downarrow$ and so $\xi\Phi\sigma\downarrow = (X\theta)\Phi\sigma\downarrow$.

Furthermore, $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$ also implies that for all $Z \in vars^2(X\Theta)$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z\theta) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z)$. With $\theta \vDash Er$ and $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\Theta) < i$, we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta) < i$. This allows us to conclude for $\xi' = X\theta$.

Case 1.b: $(\zeta, i \rhd u) \notin \mathsf{NoUse}$. In such a case, let $\xi' = \zeta\theta$. Since $\mathcal{C}$ is a well formed constraint system (Definition 8.2, item 3), we know that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta) \leq i$. Since $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$ implies that for all $Z \in vars^2(\zeta)$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z\theta) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z)$, we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta\theta) \leq i$. Since we proved that $i \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$, we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta\theta) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$.
At last, since we assumed that $(\zeta, i \rhd u) \notin \mathsf{NoUse}$ then $(\zeta\theta, i \rhd u) \notin \mathsf{NoUse}\theta$. Hence $\xi' = \zeta\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. Hence, we conclude.

— *Case 2:* $\mathsf{root}(\mathsf{C}\lfloor\xi\rfloor_\Phi) \in \mathcal{F}_c$. By definition of $\mathsf{C}\lfloor\xi\rfloor_\Phi$, there exists $\xi_1, \ldots, \xi_n \in \Pi_n$ such that $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$ and $\mathsf{f} \in \mathcal{F}_c$. But for any $i = 1 \ldots n$, we have $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi_i) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$ and $|\xi_i| < |\xi|$. Thus, by our inductive hypothesis, we can deduce that there exists $\xi'_1, \ldots, \xi'_n \in \Pi_n$ such that for all $i = 1 \ldots n$,

  — ξ'_i conforms with $\Phi\theta$ w.r.t $\mathsf{NoUse}\theta$;

  — $\xi_i(\Phi\sigma)\downarrow = \xi'_i(\Phi\sigma)\downarrow$; and

  — $param(\xi_i) \subseteq \{ax_1, \ldots, ax_j\}$ implies $param(\xi'_i) \subseteq \{ax_1, \ldots, ax_j\}$, for any j.

  Let $\xi' = \mathsf{f}(\xi'_1, \ldots, \xi'_n)$. Since $\mathsf{f} \in \mathcal{F}_c$, we can deduce that:

  — $\mathsf{f}(\xi'_1, \ldots, \xi'_n)$ conforms with $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$;

  — $\xi\Phi\sigma\downarrow = \mathsf{f}(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow = \mathsf{f}(\xi'_1, \ldots, \xi'_n)\Phi\sigma\downarrow = \xi'\Phi\sigma\downarrow$; and

  — $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi') = \max\{\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi'_i) \mid i \in \{1, \ldots, n\}\}$ and so $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi') \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi)$.

— $\mathsf{root}(\mathsf{C}\lfloor\xi\rfloor_\Phi) \in \mathcal{F}_d$. By definition of $\mathsf{C}\lfloor\xi\rfloor_\Phi$, there exists $\xi_1, \ldots, \xi_n \in \Pi_n$ such that $\xi = \mathsf{g}(\xi_1, \ldots, \xi_n)$ and $\mathsf{g} \in \mathcal{F}_d$. As in Case 2, we can apply our inductive hypothesis on each ξ_i. Thus, we will also have that there exists $\xi'_1, \ldots, \xi'_n$ such that:

  — ξ'_i conforms with $\Phi\theta$ w.r.t $\mathsf{NoUse}\theta$;

  — $\xi_i(\Phi\sigma)\downarrow = \xi'_i(\Phi\sigma)\downarrow$; and

  — $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi'_i) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi_i)$.

  Let $\xi' = \mathsf{g}(\xi'_1, \ldots, \xi'_n)$. In order to conclude, we have to show that $\xi' = \mathsf{g}(\xi'_1, \ldots, \xi'_n)$ conforms with $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. We do a case analysis:

  Case 3.a: if $\mathsf{root}(\xi'_1) \in \mathcal{F}_c$, then we have that $\mathsf{g}(\xi'_1, \ldots, \xi'_n) \notin \Pi_n$. But we know that $\mathsf{g}(\xi'_1, \ldots, \xi'_n)\Phi\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ which means that g is reduced by a rewriting rule $\ell \to r$. But all the rewriting rules we consider are defined such that if g is reduced then it implies there exists a substerm ζ of ξ'_1 such that $\zeta\Phi\sigma\downarrow = \mathsf{g}(\xi'_1, \ldots, \xi'_n)\Phi\sigma\downarrow = \mathsf{g}(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow$. Since ξ'_1 conforms with $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, then so does ξ', which allow us to conclude.

  Case 3.b: Otherwise, we deduce that $\xi' \in \Pi_n$. Moreover, if there exists $(\zeta, i \rhd u) \in \Phi$ such that $\mathsf{path}(\zeta) = \mathsf{g} \cdot \mathsf{path}(\xi'_1)$, then we apply the same reasoning than Case 1. Else it implies that $\mathsf{C}\lfloor\xi'\rfloor_\Phi = \mathsf{g}(\mathsf{C}\lfloor\xi'_1\rfloor_\Phi, \ldots, \mathsf{C}\lfloor\xi'_n\rfloor_\Phi)$ and since ξ'_i, $i = 1 \ldots n$ are all conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, we conclude that ξ' conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. □

Lemma C.37. *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system obtained by following the strategy and $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$. Let $s \in \mathbb{N}$. Assume that $\textsc{Dest}(\zeta, \ell \to r, s)$ is useless $\mathcal{C}$ for any $\zeta, \ell \to r$. For all ground recipe $\xi \in \Pi_n$, if ξ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, $\xi\Phi\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $param(\xi) \subseteq \{ax_1, \ldots ax_s\}$ then $\mathsf{C}\lfloor\xi\rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX})$.*

Proof. Let p the position of the smallest subterm of $\mathsf{C}\lfloor\xi\rfloor_\Phi$ such that $\mathsf{root}(\mathsf{C}\lfloor\xi\rfloor_\Phi|_p) \in \mathcal{F}_d$. Hence, we deduce that $\xi|_p = \mathsf{g}(\xi_1, \ldots, \xi_n)$ and $\mathsf{g} \in \mathcal{F}_d$. Moreover, since ξ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, $\xi \in \Pi_n$ and by the minimality of $\mathsf{C}\lfloor\xi\rfloor_\Phi|_p$, we deduce that there exists $(\zeta, i \rhd u) \in \Phi$ such that $\xi_1 = \zeta\theta$ and $(\zeta, i \rhd u) \notin \mathsf{NoUse}$. Since $\mathcal{C}$ is well-formed, we know that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta) \leq i$. Furthermore, since $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$, we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta\theta) \leq i$. Moreover, since $\mathcal{C}$ is obtained

by following the strategy then thanks to Lemma 8.3, $\mathcal{C}$ satisfies $\mathsf{InvGeneral}$. Thus we deduce that $ax_i \in st(\zeta\theta)$ and so $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta\theta) = i$. But $\zeta\theta \in st(\xi)$ and $param(\xi) \subseteq \{ax_1, \dots ax_s\}$ hence $i \leq s$.

Thus thanks $\textsc{Dest}(\zeta, \ell \to r, s)$ being useless on $\mathcal{C}$ with $\ell \to r$ the rewrite rule associated to g, we deduce that :

— either there exists $(\zeta', p' \rhd v') \in \Phi$ for some ξ' such that $\mathsf{path}(\zeta') = \mathsf{g} \cdot \mathsf{path}(\zeta)$ and some p' such that $p' \leq s$. But $\mathsf{path}(\xi|_p) = \mathsf{g} \cdot \mathsf{path}(\xi_1) = \mathsf{g} \cdot \mathsf{path}(\zeta\theta)$. Thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 1), we know that $\mathsf{path}(\zeta)$ is closed hence $\mathsf{path}(\xi|_p) = \mathsf{path}(\zeta')$. This is a contradiction with the fact that $\mathsf{root}(\mathsf{C}\lfloor\xi\rfloor_\Phi|_p) \in \mathcal{F}_d$.

— else $ND \vDash \forall\tilde{x}.u \stackrel{?}{\neq} u_1 \vee s \stackrel{?}{\not=} u_2 \vee \ldots \vee s \stackrel{?}{\not=} u_n$ where $\mathsf{g}(u_1, \dots, u_n) \to w$ is a renaming of $\ell \to r$. But $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies $\sigma \vDash ND$. Moreover, $\xi \in \Pi_n$ and $\xi(\Phi\sigma)\downarrow$ implies, thanks to Lemma 6.5, that $\xi|_p = \mathsf{g}(\xi_1, \dots, \xi_n)(\Phi\sigma)\downarrow$. Hence along with the hypothesis $param(\xi) \subseteq \{ax_1, \dots ax_s\}$, this is a contradiction with $\sigma \vDash ND$.

We conclude that such position p does not exist and so $\mathsf{C}\lfloor\xi\rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX})$. □

C.5.2 Order relation on second order variables

In this subsection, we state some properties regarding the relation $<_\theta$ described in Definition 8.3.

Lemma C.38. *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system. Let (σ, θ) be a pre-solution of $\mathcal{C}$. Let $X, Y \in vars^2(D)$ such that $X <_\theta Y$. We have that $X\theta$ is a strict subterm of $Y\theta$.*

Proof. $X <_\theta Y$ implies that there exist $X_1, \dots, X_n \in vars^2(D)$ such that $X <_\theta X_1 <_\theta \dots <_\theta X_n <_\theta Y$, and if we rename X, Y into X_0, X_{n+1} then we have that for all $i \in \{0, \dots, n\}$, $X_i \in vars^2(\mathsf{C}\lfloor X_{i+1}\theta\rfloor\mathsf{acc}^2(\mathcal{C}))$ and $X_i \neq X_{i+1}$.

Since (σ, θ) is a pre-solution of $\mathcal{C}$, we know that for all $X \in vars^2(\mathcal{C})$, $X\theta$ conforms to the $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. Moreover, for all $i \in \{0, \dots, n\}$, $X_i \in vars^2(\mathsf{C}\lfloor X_{i+1}\theta\rfloor_\Phi\mathsf{acc}^2(\mathcal{C}))$ implies that there exists $(\xi, k \rhd u) \in \Phi$ such that $X_i \in vars^2(\xi)$ and $\mathsf{path}(\xi) \in st(\mathsf{C}\lfloor X_{i+1}\theta\rfloor_\Phi)$. Thanks to $\mathcal{C}$ being well-formed, we know that $\mathsf{path}(\xi) \in \mathcal{F}_d^* \cdot \mathcal{AX}$ and exists hence X_i is a strict subterm of ξ which implies that $X_i\theta$ is a strict subterm of $\xi\theta$. But $X_{i+1}\theta$ is conformed to $\Phi\theta$ w.r.t. NoUse, and thus we have that $\xi\theta \in st(X_{i+1}\theta)$. Thus, we can deduce that $X_i\theta$ is a strict subterm of $X_{i+1}\theta$.

A simple induction on n allows us to conclude that $X_0\theta$ is a strict subterm of $X_{n+1}\theta$ and so $X\theta$ is a strict subterm of $Y\theta$. □

Lemma C.39. *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system. Let (σ, θ) be a pre-solution of $\mathcal{C}$. We have that $<_\theta$ is a strict partial order.*

Proof. Since (σ, θ) is a pre-solution of $\mathcal{C}$, we have that for all $X \in vars^2(D)$, $X\theta$ conforms to $\Phi\theta$ w.r.t. NoUse. By definition, $<_\theta$ is a strict partial order if, and only if:

— $\neg(X <_\theta X)$ (irreflexivity)

— if $X <_\theta Y$ then $\neg(Y <_\theta X)$ (asymmetry)

— if $X <_\theta Y$ and $Y <_\theta Z$ then $X <_\theta Z$ (transitivity)

By Definition 8.3, we already know that $<_\theta$ is closed by transitivity. Assume first that $X <_\theta X$. Thanks to Lemma C.38, we know that $X\theta$ is a strict subterm of $X\theta$ which is impossible. For the same reason, $X <_\theta Y$ and $Y <_\theta X$ would imply that $X\theta$ is a strict subterm of $X\theta$, hence the contradiction. □

Lemma C.40. *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system obtained by following the strategy. Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. Let $X, Y \in vars^2(D)$, we have that $X <_\theta Y$ implies that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$.*

Proof. Thanks to Lemma 8.3, item 1, we know that for all $(\xi, i \rhd u) \in \Phi$, $ax_i \in st(\xi\theta)$. Furthermore, since $\mathcal{C}$ is well formed (Definition 8.2, item 3), we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi) \leq i$. Moreover, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies that for all $Y \in vars^2(\xi)$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\theta) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$. Thus with $ax_i \in st(\xi\theta)$, we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi\theta) = i$.

We have that $X <_\theta Y$, and thus there exist $X_1, \ldots, X_n \in vars^2(D)$ such that $X <_\theta X_1 <_\theta \ldots <_\theta X_n <_\theta Y$ and if we rename X, Y into X_0, X_{n+1} then for all $i \in \{0, \ldots, n\}$, $X_i \in vars^2(\mathsf{C}\lfloor X_{i+1}\theta \rfloor \mathsf{acc}^2(\mathcal{C}))$ and $X_i \neq X_{i+1}$.

Let $i \in \{0, \ldots, n, n+1\}$. We know that there exist u_i and k_i such that $(X_i, k_i \overset{?}{\vdash} u_i) \in D$. Since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we have that $param(X_i\theta) \subseteq \{ax_1, \ldots, ax_{k_i}\}$.

But $X_i \in vars^2(\mathsf{C}\lfloor X_{i+1}\theta \rfloor \mathsf{acc}^2(\mathcal{C}))$ implies that there exists $(\xi, j \rhd v) \in \Phi$ such that $\mathsf{path}(\xi) \in st(\mathsf{C}\lfloor X_{i+1}\theta \rfloor)$ and $X_i \in vars^2(\xi)$. Furthermore, since $X_{i+1}\theta$ conforms with $\Phi\theta$ w.r.t. NoUse, we can deduce that $\xi\theta \in st(X_{i+1}\theta)$. We have seen that $ax_j \in st(\xi\theta)$, we can deduce that $j \leq k_{i+1}$. At last, since $X_i\theta \in st(\xi\theta)$, $param(X_i\theta) \subseteq \{ax_1, \ldots, ax_{k_i}\}$ and $param(\xi\theta) \subseteq \{ax_1, \ldots, ax_j\}$, we can deduce that $k_i \leq j$ which implies $k_i \leq k_{i+1}$ and so $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X_i) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X_{i+1})$. Altogether, this allows us to conclude that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X_0) \leq \ldots \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X_{n+1})$ and so $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$. □

Lemma C.41. *Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ be a well-formed constraint system obtained by following the strategy. Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. Let ξ be a ground recipe in Π_n and $(X, i \overset{?}{\vdash} u) \in D$ such that:*

- *ξ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$ and $param(\xi) \subseteq \{ax_1, \ldots, ax_i\}$.*
- *there exists a position p, such that $\xi' = X\theta_{|p}$, $\xi'\Phi\sigma\downarrow = \xi\Phi\sigma\downarrow$, and $\mathsf{C}_0 = \mathsf{C}_1[\mathsf{C}_2]_p$ where $\mathsf{C}_0 = \mathsf{C}\lfloor X\theta[\xi]_p \rfloor$, $\mathsf{C}_1 = \mathsf{C}\lfloor X\theta \rfloor$ and $\mathsf{C}_2 = \mathsf{C}\lfloor \xi \rfloor$.*
- *for all $Y \in vars^2(\mathsf{C}\lfloor \xi \rfloor \mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Y)$.*

Then there exists θ' such that (σ, θ') is a pre-solution of $\mathcal{C}$ with $\theta' \vDash \mathsf{mgu}(Er)$, $X\theta' = X\theta[\xi]_p$ and for all $Y \in vars^2(D) \setminus \{X\}$, $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi\theta} = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi\theta'}$.

Proof. Since we assume that $\mathcal{C}$ is well-formed (Definition 8.2, item 2), the path of any recipe of a frame element of Φ is closed hence for all ξ, for all θ, $\mathsf{C}\lfloor \xi \rfloor_\Phi = \mathsf{C}\lfloor \xi \rfloor_{\Phi\theta}$. Hence we omit the substitution in the context. Let θ' be a substitution defined as follows:

- for all $Y \in vars^2(D)$ such that $\neg(X <_\theta Y)$, $Y\theta' \overset{\mathsf{def}}{=} Y\theta$
- $X\theta' \overset{\mathsf{def}}{=} X\theta[\xi]_p$
- Otherwise, $Y\theta' \overset{\mathsf{def}}{=} \mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C})\theta'$, where $Y \in vars^2(D)$
- for all $Y \in vars^2(\mathcal{C}) \setminus vars^2(D)$, $Y\theta' = Y\mathsf{mgu}(Er)\theta'$

First, we need to justify that the substitution θ' is well-defined. By Lemma C.39, we know that the relation $<_\theta$ is a strict partial order. Let $Y \in vars^2(D)$ such that $X <_\theta Y$ and so $Y\theta' = \mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C})\theta'$. But for all $Z \in vars^2(\mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, we have $Z <_\theta Y$. Since the relation $<_\theta$ is a strict partial order on $vars^2(D)$, we conclude that $Y\theta'$ is well-defined. At last, for all $Y \in vars^2(\mathcal{C}) \setminus vars^2(D)$, for all $X \in vars^2(Y\mathsf{mgu}(Er))$, $X \in vars^2(D)$. Since θ' is well defined on any variables in $vars^2(D)$, we conclude that θ' is well-defined on all variable of $\mathcal{C}$.

Now, it remains to prove the three expected properties.

Property 1. $X\theta' = X\theta[\xi]_p$: By definition of θ', we know that $X\theta' = X\theta[\xi]_p$ hence the result trivially holds.

Property 2. $\theta' \vDash \mathsf{mgu}(Er)$: By definition, for all $Y \in vars^2(\mathcal{C}) \setminus vars^2(D)$, $Y\theta' = Y\mathsf{mgu}(Er)\theta'$. Hence, we trivially deduce that $\theta' \vDash \mathsf{mgu}(Er)$.

Property 3. for all $Y \in vars^2(D) \setminus \{X\}$, $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi\theta} = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi\theta'}$: For any variable $Y \in vars^2(D)$ such that $\neg(X <_\theta Y)$, we have that $Y\theta' = Y\theta$ which means that $\mathsf{C}\lfloor Y\theta \rfloor_\Phi = \mathsf{C}\lfloor Y\theta' \rfloor_\Phi$. Moreover,

for all $Y \in vars^2(\mathcal{C}) \setminus \{X\}$, if $X <_\theta Y$ then $Y\theta' = \mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C})\theta'$. But since $Y\theta$ is a ground recipe, then $st(\mathsf{C}\lfloor Y\theta \rfloor_\Phi) \cap \mathrm{dom}(\mathsf{acc}^2(\mathcal{C})) \subseteq (\mathcal{F}_d^* \cdot \mathcal{AX})$ and for all $w \in \mathrm{dom}(\mathsf{acc}^2(\mathcal{C})) \cap (\mathcal{F}_d^* \cdot \mathcal{AX})$, we have that $\mathsf{C}\lfloor w\mathsf{acc}^2(\mathcal{C})\theta \rfloor_\Phi = \mathsf{C}\lfloor w\mathsf{acc}^2(\mathcal{C})\theta' \rfloor_\Phi \ (= w)$. Therefore, we have that $\mathsf{C}\lfloor Y\theta' \rfloor_\Phi = \mathsf{C}\lfloor Y\theta \rfloor_\Phi$.

Property 4. (σ, θ') is a pre-solution of $\mathcal{C}$, *i.e.*

— for every $Y \in vars^2(\mathcal{C})$, we have that $Y\theta'$ conforms to the frame $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta$, and

— for every $(Y, j \overset{?}{\vdash} v)$ in D, we have that $(Y\theta')(\Phi\sigma)\downarrow = v\sigma\downarrow$ and $param(Y\theta') \subseteq \{ax_1, \ldots, ax_j\}$.

Thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 7), we know that for all $Y \in vars^2(\mathcal{C})$, $\mathsf{C}\lfloor Y\mathsf{mgu}(Er) \rfloor \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX} \cup \mathcal{X}^2)$ and for all $\zeta \in st(Y\mathsf{mgu}(Er))$, $\mathsf{path}(\zeta) \in \mathcal{F}_d^* \cdot \mathcal{AX}$ implies that there exists j and v such that $(\zeta, j \rhd v) \in \Phi$. Since $\theta \vDash Er$, we deduce that $\mathsf{C}\lfloor Y\theta \rfloor = \mathsf{C}\lfloor Y\mathsf{mgu}(Er) \rfloor_{\{Z \mapsto \mathsf{C}\lfloor Z\theta \rfloor \mid Z \in vars^2(\mathsf{C}\lfloor Y\mathsf{mgu}(Er) \rfloor)\}}$. But by definition, for all $Y \in vars^2(\mathcal{C}) \setminus vars^2(D)$, $Y\theta' = \mathsf{mgu}(Er)\theta'$. Hence, along with $\mathcal{C}$ satisfying $\mathsf{InvGeneral}$ (Definition 8.1, item 3) thanks to Lemma 8.3, we deduce that $Y\theta'$ conforms to $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta'$ if and only if for all $Z \in vars^2(\mathsf{C}\lfloor Y\mathsf{mgu}(Er) \rfloor)$, $Z\theta'$ conforms to $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta'$. Thus it remains to prove that for all $Y \in vars^2(D)$, $Y\theta'$ conforms to $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta'$

Let $Y, j \overset{?}{\vdash} v$ be a deducibility constraint in D. We prove the results by induction on $<_\theta$.

Base case 1: $\neg(X <_\theta Y)$. In such a case, we have that $Y\theta' = Y\theta$. Thus, we have $(Y\theta')\Phi\sigma\downarrow = (Y\theta)\Phi\sigma\downarrow$ and $param(Y\theta') = param(Y\theta) \subseteq \{ax_1, \ldots, ax_j\}$. Furthermore, by definition of $<_\theta$, we have that for all $Z \in vars^2(\mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, $Z <_\theta Y$ and so $\neg(X <_\theta Z)$. Thus, $Z\theta = Z\theta'$. With $Y\theta' = Y\theta$ and $Y\theta$ conforms with $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, we can deduce that $Y\theta'$ also conforms with $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta'$.

Base case 2: $Y = X$. In such a case, we have that $X\theta = X\theta[\xi]_p$. By hypothesis, we know that $\xi(\Phi\sigma)\downarrow = X\theta|_p\Phi\sigma\downarrow$. Thus, we have that $X\theta[\xi]_p\Phi\sigma\downarrow = X\theta\Phi\sigma\downarrow$. Furthermore, we also know that $param(\xi) \subseteq \{ax_1, \ldots, ax_i\}$ and $param(X\theta) \subseteq \{ax_1, \ldots, ax_i\}$. Thus we can conclude that $param(X\theta[\xi]_p) \subseteq \{ax_1, \ldots, ax_i\}$.

By hypothesis, we know that for all $Z \in vars^2(\mathsf{C}\lfloor \xi \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Z)$ which means that $Z\theta' = Z\theta$ and so ξ conforms with $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta$. By definition of $<_\theta$, we have that for all $Z \in vars^2(\mathsf{C}\lfloor X\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, $Z <_\theta X$ which means that $Z\theta' = Z\theta$ and so $X\theta$ conforms with $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta'$. Therefore, since by hypothesis we have that $\mathsf{C}\lfloor X\theta' \rfloor = \mathsf{C}\lfloor X\theta \rfloor [\mathsf{C}\lfloor \xi \rfloor]_p$, we can conclude that $X\theta' = X\theta[\xi]_p$ conforms with $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta'$.

Inductive case $X <_\theta Y$*:* In such a case, $Y\theta' = \mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C})\theta'$. We know by definition of $<_\theta$ that if $Z \in vars^2(\mathsf{C}\lfloor Y\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$ then $Z <_\theta Y$. Hence by Lemma C.40, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$ which implies that there exists a constraint $(Z, \ell \overset{?}{\vdash} r) \in D$ with $\ell \leq j$. Therefore, thanks to our induction hypothesis, we deduce that $(Z\theta')(\phi\sigma)\downarrow = r\sigma\downarrow$ and $param(Z\theta') \subseteq \{ax_1, \ldots, ax_\ell\}$.

Let $w \in st(\mathsf{C}\lfloor Y\theta \rfloor_\Phi) \cap (\mathcal{F}_d^* \cdot \mathcal{AX})$. Hence there exists $(\zeta, \ell \rhd r) \in \Phi$ such that $\zeta = w\mathsf{acc}^2(\mathcal{C})\theta'$. We already show that for all $Z \in vars^2(\zeta)$ with $(Z, \ell' \overset{?}{\vdash} r') \in D$, $Z\theta'(\Phi\sigma)\downarrow = r'\sigma\downarrow$ and $param(Z\theta') \subseteq \{ax_1, \ldots, ax'_\ell\}$. Hence, thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 5), we deduce that $\zeta\theta'(\Phi\sigma)\downarrow = r\downarrow = \zeta\theta(\Phi\sigma)\downarrow$. Moreover, once again thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 3), $param(\zeta\theta') \subseteq \{ax_1, \ldots, ax_\ell\}$ and so $param(w\mathsf{acc}^2(\mathcal{C})\theta') \subseteq \{ax_1, \ldots, ax_\ell\}$. At last, since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and thanks to $\mathcal{C}$ satisfying $\mathsf{InvGeneral}$ (item 1), we deduce that $ax_\ell \in st(\zeta\theta)$ which implies that $ax_\ell \in st(Y\theta)$ and so $\ell \leq j$. Hence, we can conclude that $Y\theta'(\Phi\sigma)\downarrow = Y\theta(\Phi\sigma)\downarrow = v\sigma\downarrow$ and $param(Y\theta') \subseteq \{ax_1, \ldots, ax_j\}$.

Lastly, by definition of θ', we have that $Y\theta' = \mathsf{C}\lfloor Y\theta \rfloor \mathsf{acc}^2(\mathcal{C})\theta'$ thus, since $Y\theta$ conforms with $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, we deduce that $Y\theta'$ conforms with the frame $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta'$. □

C.5.3 Preliminaries for soundness of Phase 1 Step a

In this subsection, we are only looking to prove Lemma 8.6 when the strategy is on the first phase of the first step. Hence, according to Subsection 7.4, the only rules that can be applied during

this phase are DEST and EQ-LEFT-RIGHT. In such a case, thanks to Lemma C.31, the constraint systems or matrices of constraint system we will consider in this subsection all satisfy the invariants $\mathsf{PP1Sa}(s)$ for some s depending on the parameter of the rules DEST and EQ-LEFT-RIGHT.

Let T be a set of terms, and u be a term, we denote by $\mathsf{nb}_{\mathsf{occ}}(u, T)$ the number of occurrences of u in T.

Lemma C.42. *Let M be a matrix of constraint systems obtained during phase 1.a with support s by following the strategy. Let $\mathcal{C}$ be a constraint system in M with Φ its associated frame, and $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. Moreover, let $(\xi, s \rhd u) \in \Phi$ and let $\zeta \in \Pi_n$ such that:*

- *$param(\zeta) \subseteq \{ax_1, \ldots, ax_{s-1}\}$,*
- *$\zeta\Phi\sigma\downarrow = u\sigma$,*
- *for all $Y \in vars^2(\mathcal{C})$, $\xi \notin st(Y\mathsf{mgu}(Er))$, and*
- *for all $\mathsf{g} \in \mathcal{F}_d$, for all $(\xi', i \rhd v) \in \Phi$, $\mathsf{path}(\xi') \neq \mathsf{g} \cdot \mathsf{path}(\xi)$.*

Either $\mathsf{nb}_{\mathsf{occ}}(\xi\theta, \{X\theta \mid X \in vars^2(\mathcal{C})\}) = 0$ or else there exists θ' such that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$ and:

$$\mathsf{nb}_{\mathsf{occ}}(\xi\theta', \{X\theta' \mid X \in vars^2(\mathcal{C})\}) < \mathsf{nb}_{\mathsf{occ}}(\xi\theta', \{X\theta' \mid X \in vars^2(\mathcal{C})\}).$$

Proof. Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, and consider $(X, k \overset{?}{\vdash} v) \in D$ such that $k \geq s$ and $\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta\rfloor)$. We first show that if such $(X, k \overset{?}{\vdash} v)$ does not exists, then $\mathsf{nb}_{\mathsf{occ}}(\xi\theta, \{X\theta \mid X \in vars^2(\mathcal{C})\}) = 0$. By hypothesis, we know that for all $\mathsf{g} \in \mathcal{F}_d$, for all $(\xi', i \rhd v) \in \Phi$, $\mathsf{path}(\xi') \neq \mathsf{g} \cdot \mathsf{path}(\xi)$. moreover, since $\mathcal{C}$ satisfies $\mathsf{PP1Sa}(s)$, we know that for all $(\beta, s \rhd v) \in \Phi(\mathcal{C})$, either $\beta \in \mathcal{AX}$ or there exists $X_2, \ldots, X_n \in \mathcal{X}^2$, $\mathsf{f} \in \mathcal{F}_d$ and $(\beta', p \rhd v) \in \Phi(\mathcal{C})$ such that $\beta = \mathsf{f}(\beta', X_2, \ldots, X_n)$ and $p \leq s$. Hence we deduce that for all $(\beta, s \rhd v) \in \Phi(\mathcal{C})$, if $\xi \in st(\beta)$ then $\xi = \beta$. Furthermore, since for all $Y \in vars^2(\mathcal{C})$, $\xi \notin st(Y\mathsf{mgu}(Er))$, then for all $Y \in vars^2(\mathcal{C})$, $\xi\theta \in st(Y\theta)$ implies that there exists $Z \in vars^2(D)$ such that $\xi\theta \in st(Z\theta)$. We select $X \in vars^2(D)$ such that $X\theta$ is the smallest recipe that contain $\xi\theta$. If X does not exists then $\mathsf{nb}_{\mathsf{occ}}(\xi\theta, \{X\theta \mid X \in vars^2(\mathcal{C})\}) = 0$ else since $X\theta$ conforms with $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$ and for all $(\beta, s \rhd v) \in \Phi(\mathcal{C})$, if $\xi \in st(\beta)$ then $\xi = \beta$, we conclude that $\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta\rfloor)$.

Property 1. We first show for all $\zeta \in \Pi_n$, for all position p in $X\theta$, if $\zeta\Phi\sigma\downarrow = (X\theta)_{|p}\Phi\sigma\downarrow$, then there exists $\zeta' \in st(\zeta)$ and a position p' such that p' is a prefix of p and $X\theta[\zeta]_p(\Phi\sigma)\downarrow = X\theta[\zeta']_{p'}(\Phi\sigma)\downarrow$ and $X\theta[\zeta']_{p'} \in \Pi_n$. We prove this result by induction on the length $|p|$ of p.

Base case $|p| = 0$. In such a case we have that $p = \epsilon$. Let $\zeta' = \zeta$ and $p' = p$. We have that $X\theta[\zeta']_{p'} = \zeta' = \zeta$. Since by hypothesis, we have that $\zeta \in \Pi_n$, the result trivially holds.

Inductive step $|p| > 0$. In such a case, we have that $p = p_1 \cdot r$ for some $r \in \mathbb{N}$ and some p_1 such that $|p_1| < |p|$. Assume that $X\theta_{|p_1} = \mathsf{f}(\xi_1, \ldots, \xi_n)$. We have to distinguish two cases:

1. *$r = 1$, $\mathsf{f} \in \mathcal{F}_d$ and $\mathsf{root}(\zeta) \in \mathcal{F}_c$:* We know that $X\theta(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $X\theta \in \Pi_n$. Thus, by Lemma 6.5, we can deduce that $\mathsf{f}(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$, and thus f has been reduced. But by hypothesis, we also know that $\zeta\Phi\sigma\downarrow = \xi_1\Phi\sigma\downarrow$. If we denote $\zeta = \mathsf{g}(\zeta_1, \ldots, \zeta_m)$, then by definition of our rewrite rule, we have that $\zeta_1\Phi\sigma\downarrow = \mathsf{f}(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow$. Since $\zeta \in \Pi_n$, we have also that $\zeta_1 \in \Pi_n$. Furthermore, we have $\zeta_1\Phi\sigma\downarrow = X\theta_{|p_1}\Phi\sigma\downarrow$. Thanks to our induction hypothesis, we deduce that there exist $\zeta_1' \in st\zeta_1$ and p_1' a prefix of p_1 such $X\theta[\zeta_1]_{p_1}(\Phi\sigma)\downarrow = X\theta[\zeta_1']_{p_1'}(\Phi\sigma)\downarrow$ and $X\theta[\zeta_1']_{p_1'} \in Rr$. Let $\zeta' = \zeta_1'$ and $p' = p_1'$. This allows us to conclude.
2. *Otherwise:* By definition of Π_n, we have that $X\theta[\zeta]_p \in \Pi_n$, and thus the result holds with $\zeta' = \zeta$ and $p' = p$.

Property 2. We show that for all $\zeta \in \Pi_n$ such that $param(\zeta) \subseteq \{ax_1, \ldots, ax_k\}$, for all $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, for all position $p \in \mathcal{P}os(\mathsf{C}\lfloor X\theta\rfloor)$, if $\zeta(\Phi\sigma)\downarrow = (X\theta)_{|p}(\Phi\sigma)\downarrow$, ζ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, for all $Y \in vars^2(\mathsf{C}\lfloor\zeta\rfloor\mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Y)$ and $\mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \mathsf{C}\lfloor X\theta[\zeta]_p\rfloor) < \mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \mathsf{C}\lfloor X\theta\rfloor)$

then there exists θ' such that $(\sigma,\theta') \in \mathsf{Sol}(\mathcal{C})$ and $\mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor Y\theta'\rfloor \mid Y \in vars^2(\mathcal{C})\}) < \mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor Y\theta\rfloor \mid Y \in vars^2(\mathcal{C})\})$.

We prove this property by induction on the length $|p|$ of p.

Base case $|p| = 0$. In such a case, we can apply Lemma C.41. Indeed, ζ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$ and $param(\zeta) \subseteq \{ax_1,\ldots,ax_k\}$. Furthermore we have $\zeta(\Phi\sigma)\downarrow = X\theta(\Phi\sigma)\downarrow$ and for all $Y \in vars^2(\mathsf{C}\lfloor\zeta\rfloor\mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Y)$. Hence, we deduce that there exists θ' such that (σ,θ') is a pre-solution of $\mathcal{C}$ with $X\theta' = \zeta$ and for all $Y \in vars^2(D)\backslash\{X\}$, we have that $\mathsf{C}\lfloor Y\theta\rfloor = \mathsf{C}\lfloor Y\theta'\rfloor$, $\theta' \vDash \mathsf{mgu}(Er)$. By hypothesis, we know that $\mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor\zeta\rfloor\}) < \mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor X\theta\rfloor\})$ and since $\mathsf{C}\lfloor Y\theta\rfloor = \mathsf{C}\lfloor Y\theta'\rfloor$, for all $Y \in vars^2(D)\backslash\{X\}$, we can deduce that

$$\mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor Y\theta'\rfloor \mid Y \in vars^2(D)\}) < \mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor Y\theta\rfloor \mid Y \in vars^2(D)\}).$$

Moreover, by hypothesis on ξ, we know that for all $Y \in vars^2(\mathcal{C})$, $\xi \notin st(Y\mathsf{mgu}(Er))$. Hence, since $\theta \vDash \mathsf{mgu}(Er)$ and $\theta' \vDash \mathsf{mgu}(Er)$, we deduce that:

$$\mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor Y\theta'\rfloor \mid Y \in vars^2(\mathcal{C})\}) < \mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor Y\theta\rfloor \mid Y \in vars^2(\mathcal{C})\}).$$

Furthermore, $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$ implies $\sigma \vDash Eq \wedge ND$. At last, $\mathcal{C}$ satisfies $\mathsf{PP1Sa}(s)$ (item C.2) thus along with $\theta' \vDash \mathsf{mgu}(Er)$, we deduce that $\theta' \vDash Er$ and so the result holds. Hence the result holds.

Inductive step $|p| > 0$. In such a case, we have that $p = p'\cdot r$ with $r \in \mathbb{N}$ and $|p'| < |p|$. Assume that $X\theta_{|p'} = \mathsf{f}(\xi_1,\ldots,\xi_n)$. Since $\zeta(\Phi\sigma)\downarrow = X\theta_{|p}(\Phi\sigma)\downarrow$, we have that $(X\theta_{|p'})[\zeta]_r(\Phi\sigma)\downarrow = X\theta_{|p'}(\Phi\sigma)\downarrow$. By definition of $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$, we know that $\mathsf{f}(\xi_1,\ldots,\xi_n)$ conforms to $\Phi\theta$. Now, we distinguish several cases:

Case (a): $\mathsf{f} \in \mathcal{F}_c$. In such a case, let $\zeta' = (X\theta_{|p'})[\zeta]_r$. Hence we have that $X\theta[\zeta']_{p'} = X\theta[\zeta]_p$ which implies that $X\theta[\zeta']_{p'}(\Phi\sigma)\downarrow = X\theta(\Phi\sigma)\downarrow$ and $X\theta[\zeta']_{p'} \in \Pi_n$.

Moreover, since $\mathsf{f} \in \mathcal{F}_c$, we have that $\mathsf{C}\lfloor\zeta'\rfloor = \mathsf{f}(\mathsf{C}\lfloor\xi_1\rfloor,\ldots,\mathsf{C}\lfloor\xi_{r-1}\rfloor,\mathsf{C}\lfloor\zeta\rfloor,\mathsf{C}\lfloor\xi_{r+1}\rfloor,\ldots,\mathsf{C}\lfloor\xi_n\rfloor)$. But by hypothesis, ζ and $X\theta_{|p'}$ are conforms to $\Phi\theta$ w.r.t. NoUse. Hence, $\xi_1,\ldots,\xi_n$ also conform to $\Phi\theta$ w.r.t. NoUse which implies that ζ' conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$.

Furthermore, we know that $X\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$ and $p \in \mathcal{P}os(\mathsf{C}\lfloor X\theta\rfloor)$ hence $\mathsf{C}\lfloor\xi_1\rfloor,\ldots,\mathsf{C}\lfloor\xi_n\rfloor \in st(\mathsf{C}\lfloor X\theta\rfloor)$. Moreover, by definition of $<_\theta$, we have for all $i \in \{1,\ldots,n\}$, for all $Y \in vars^2(\mathsf{C}\lfloor\xi_i\rfloor\mathsf{acc}^2(\mathcal{C}))$, $Y <_\theta X$. But by Lemma C.39, $<_\theta$ is a strict partial order which means that $Y <_\theta X$ implies $\neg(X <_\theta Y)$. Since by hypothesis, we have that for all $Y \in vars^2(\mathsf{C}\lfloor\zeta\rfloor\mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Y)$, then we can deduce that for all $Y \in vars^2(\mathsf{C}\lfloor\zeta'\rfloor\mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Y)$.

At last, $X\theta[\zeta']_{p'} = X\theta[\zeta]_p$ and $\#\{\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta[\zeta]_p\rfloor)\} < \#\{\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta\rfloor)\}$ trivially implies that $\#\{\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta[\zeta']_{p'}\rfloor)\} < \#\{\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta\rfloor)\}$.

Hence we conclude by applying our inductive hypothesis on θ, ζ' and p'.

Case (b): $\mathsf{f} \in \mathcal{F}_d$, $r \neq 1$. This case is similar to Case (a). Indeed, let $\zeta' = (X\theta_{|p'})[\zeta]_r$. We have $p \in \mathcal{P}os(\mathsf{C}\lfloor X\theta\rfloor)$. Hence $\mathsf{C}\lfloor X\theta_{|p'}\rfloor = \mathsf{f}(\mathsf{C}\lfloor\xi_1\rfloor,\ldots,\mathsf{C}\lfloor\xi_n\rfloor)$ and $\mathsf{C}\lfloor\zeta'\rfloor = \mathsf{f}(\mathsf{C}\lfloor\xi_1\rfloor,\ldots,\mathsf{C}\lfloor\xi_{r-1}\rfloor,\mathsf{C}\lfloor\zeta\rfloor,\mathsf{C}\lfloor\xi_{r+1}\rfloor,\ldots,\mathsf{C}\lfloor\xi_n\rfloor)$. Thus we can apply the same reasoning as Case (a).

Case (c): $\mathsf{f} \in \mathcal{F}_d$, $r = 1$ *and* $|\mathsf{C}\lfloor X\theta_{|p'}[\zeta]_r\rfloor| > 1$. In such a case, we know that $\mathsf{C}\lfloor X\theta_{|p'}[\zeta]_r\rfloor = \mathsf{f}(\mathsf{C}\lfloor\zeta\rfloor,\mathsf{C}\lfloor\xi_2\rfloor,\ldots,\mathsf{C}\lfloor\xi_n\rfloor)$ since $|\mathsf{C}\lfloor X\theta_{|p'}[\zeta]_r\rfloor| > 1$. Thus, similarly to Case (a), we can apply our inductive hypothesis.

Case (d): $\mathsf{f} \in \mathcal{F}_d$, $r = 1$ *and* $|\mathsf{C}\lfloor X\theta_{|p'}[\zeta]_r\rfloor| = 1$. In such a case, we have that there exists $(\gamma, \ell \rhd w) \in \Phi$ such that $\mathsf{path}(\gamma) = \mathsf{f}\cdot\mathsf{path}(\zeta)$. Since ζ conforms to $\Phi\theta$ w.r.t. NoUse, we can denote $\gamma\theta = \mathsf{f}(\zeta,\gamma_2\theta,\ldots,\gamma_n\theta)$. Since $\mathcal{C}$ is well-formed and $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$, we also know that $\gamma\theta(\Phi\sigma)\downarrow = w\sigma$ and $X\theta(\Phi\sigma\downarrow) \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Thus by Lemma 6.5, we can deduce that $X\theta_{|p'}(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Therefore, since we have seen that $\zeta\Phi\sigma\downarrow = \xi_1\Phi\sigma\downarrow$, we can deduce that for all $i \in \{2,\ldots,n\}$, $\xi_i(\Phi\sigma)\downarrow = \gamma_i\theta(\Phi\sigma)\downarrow$. We now distinguish three cases:

— $\ell < s$: In such a case, let $\zeta' = \gamma\theta$. Since $(X, k \overset{?}{\vdash} v) \in D(M_{i,j})$, $k \geq s > \ell$ and $\mathcal{C}$ is well formed, we know that for all $Y \in vars^2(\gamma)$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) < k = \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X)$. Hence by Lemma C.40, $\neg(X <_\theta Y)$.

At last, we know that $\mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \mathsf{C}\lfloor X\theta[\zeta]_p\rfloor) < \mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \mathsf{C}\lfloor X\theta\rfloor)$ hence we deduce that $\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta_{|p'}\rfloor)$. Thus along with $|\mathsf{C}\lfloor\zeta'\rfloor| = 1$, $\mathsf{C}\lfloor\zeta'\rfloor \neq \mathsf{C}\lfloor\xi\rfloor$ (since $s \neq \ell$), we can deduce that $\#\{\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta[\zeta']_{p'}\rfloor)\} < \#\{\mathsf{C}\lfloor\xi\rfloor \in st(\mathsf{C}\lfloor X\theta\rfloor)\}$. Hence, we can apply our inductive hypothesis on ζ', θ and p'.

— $\ell > s$: Such a case is impossible. Indeed, by Property $\mathsf{PP1Sa}(s)$, $\ell > s$ implies that $\gamma \in \mathcal{AX}$ which is in contradiction with $\mathsf{path}(\gamma) = \mathsf{f} \cdot \mathsf{path}(\zeta)$.

— $\ell = s$: Thanks to $\mathcal{C}$ satisfying Property $\mathsf{PP1Sa}(s)$, we know that $\gamma_2, \ldots, \gamma_n \in \mathcal{X}^2$. We prove our result now by induction on $N = \#\{\gamma_i \mid X <_\theta \gamma_i, i \in \{2, \ldots, n\}\}$.

Base case $N = 0$*:* Such a case is similar to the case $\ell < s$. Indeed, by hypothesis, we know that for all $Y \in vars^2(\gamma)$, $\neg(X <_\theta Y)$ (note that the first argument of γ can not be a variable). We can apply our inductive hypothesis on ζ', θ and p'.

Inductive case $N > 0$. We know that there exists $i_0 \in \{2, \ldots, n\}$ such that $X <_\theta \gamma_{i_0}$. Since $\mathcal{C}$ is well-formed, we know that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\gamma_{i_0}) \leq \ell = s$. Moreover, by Lemma C.40, $X <_\theta \gamma_{i_0}$ implies that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\gamma_{i_0})$. Since $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) = k$ and $k \geq s$, we can deduce that $k = s = \ell$ and so there exists v such that $(\gamma_{i_0}, s \overset{?}{\vdash} v) \in D$. Furthermore, we already know that $\xi_{i_0}(\Phi\sigma)\downarrow = \gamma_{i_0}\theta(\Phi\sigma)\downarrow$ and $\mathsf{C}\lfloor\xi_{i_0}\rfloor \in st(\mathsf{C}\lfloor X\theta\rfloor)$ thanks $p \in \mathcal{P}os(\mathsf{C}\lfloor X\theta\rfloor_)$. Hence, for all $Y \in vars^2(\mathsf{C}\lfloor\xi_{i_0}\rfloor\mathsf{acc}^2(\mathcal{C}))$ we have that $Y <_\theta X$, and thus $Y <_\theta \gamma_{i_0}$ (and so $\neg(\gamma_{i_0} <_\theta Y)$). Thus, thanks to Lemma C.41, we know that there exists θ' such that (σ, θ') is a pre-solution of $\mathcal{C}$, $\gamma_{i_0}\theta' = \xi_{i_0}$, $\theta' \vDash \mathsf{mgu}(Er)$ and for all $Y \in vars^2(D)\backslash\{\gamma_{i_0}\}$, $\mathsf{C}\lfloor Y\theta'\rfloor = \mathsf{C}\lfloor Y\theta\rfloor$.

Let $j \in \{2, \ldots, n\}$ such that $i_0 \neq j$. We show that if $X <_{\theta'} \gamma_j$ then $X <_\theta \gamma_j$: $X <_{\theta'} \gamma_j$ implies that there exists $Y_1, \ldots, Y_m$ such that $X <_{\theta'} Y_1 <_{\theta'} \ldots <_{\theta'} Y_m <_{\theta'} \gamma_j$. But we know that for all $Y \in vars^2(\mathcal{C})\backslash\{\gamma_{i_0}\}$, $\mathsf{C}\lfloor Y\theta'\rfloor = \mathsf{C}\lfloor Y\theta\rfloor$. Hence, if for all $q \in \{1, \ldots, m\}$, $Y_q \neq \gamma_{i_0}$, we trivially have that $X <_\theta \gamma_j$. If there exists $q \in \{1, \ldots, m\}$ such that $Y_q = \gamma_{i_0}$, it would imply that $X <_{\theta'} \gamma_{i_0}$. However $X <_{\theta'} \gamma_{i_0}$ is impossible. Indeed, assume that $X <_{\theta'} \gamma_{i_0}$. We have that $\mathsf{C}\lfloor X\theta\rfloor = \mathsf{C}\lfloor X\theta'\rfloor$ and we have seen that $\mathsf{C}\lfloor\zeta_{i_0}\rfloor \in st(\mathsf{C}\lfloor X\theta\rfloor)$. Hence, we have that $\mathsf{C}\lfloor\zeta_{i_0}\rfloor \in st(\mathsf{C}\lfloor X\theta'\rfloor)$, and thus for all $Y \in vars^2(\mathsf{C}\lfloor\xi_{i_0}\rfloor\mathsf{acc}^2(\mathcal{C}))$ we have that $Y <_{\theta'} X$. Since we have assumed that $X <_{\theta'} \gamma_{i_0}$, we have that either (a) $X \in vars^2(\mathsf{C}\lfloor\gamma_{i_0}\theta'\rfloor\mathsf{acc}^2(\mathcal{C}))$, or (b) there exists Z such that $X <_{\theta'} Z$ and $Z \in vars^2(\mathsf{C}\lfloor\gamma_{i_0}\theta'\rfloor\mathsf{acc}^2(\mathcal{C}))$. The case (a) is impossible since this would imply that $X <_{\theta'} X$. The case (b) is impossible too since it would imply that $Z <_{\theta'} X$, and thus $X <_{\theta'} X$. Hence we have that:

$$\{\gamma_i \mid X <_{\theta'} \gamma_i, i \in \{2, \ldots, n\}\} \subset \{\gamma_i \mid X <_\theta \gamma_i, i \in \{2, \ldots, n\}\}.$$

Note that $X <_\theta \gamma_{i_0}$ by hypothesis whereas we have seen that $\neg(X <_{\theta'} \gamma_{i_0})$.

At last, we know that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and (σ, θ') is a pre-solution of $\mathcal{C}$. But thanks to $\mathcal{C}$ satisfying Property $\mathsf{PP1Sa}(s)$, we know that for all $Y, q \overset{?}{\vdash} v$, if $q \geq s$ then $Y \notin vars^2(Er)$. Hence, since $(\gamma_i, s \overset{?}{\vdash} v) \in D$, $\theta' \vDash \mathsf{mgu}(Er)$ and for all $Y \in vars^2(D)\backslash\{\gamma_i\}$, $\mathsf{C}\lfloor Y\theta'\rfloor = \mathsf{C}\lfloor Y\theta\rfloor$, we can conclude that $\theta' \vDash Er$ (we rely on the form of the inequations to conclude on this point). Moreover, since ND only depend on σ, we can conclude that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$ which allows us to apply our inductive hypothesis on $\zeta' = \gamma\theta'$, θ' and p.

Proof of the lemma. Relying on Property 1 and Property 2, we are now able to conclude.

We have $\zeta \in \Pi_n$, $param(\zeta) \subseteq \{ax_1, \ldots, ax_{s-1}\}$, $\zeta\Phi\sigma\downarrow = u\sigma$ where $(\xi, s \rhd u) \in \Phi$ and $X\theta_{|p} = \xi\theta$ where $p \in \mathcal{P}os(\mathsf{C}\lfloor X\theta\rfloor)$. But since $\mathcal{C}$ is well formed and $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we know that $\xi\theta(\Phi\sigma)\downarrow = u\sigma = \zeta(\Phi\sigma)\downarrow$.

Hence by the first property, we can deduce that there exists $\zeta' \in st(\zeta)$, a position p' prefix of p such that $X\theta[\zeta]_p(\Phi\sigma)\downarrow = X\theta[\zeta']_{p'}(\Phi\sigma)\downarrow$ and $X\theta[\zeta']_{p'} \in \Pi_n$. But we already know that

$X\theta[\zeta]_p(\Phi\sigma)\downarrow = X\theta(\Phi\sigma)\downarrow$ so $X\theta[\zeta']_{p'}(\Phi\sigma)\downarrow = X\theta(\Phi\sigma)\downarrow$. Furthermore $\zeta' \in st(\zeta)$, ζ conforms to $\Phi\theta$ w.r.t. NoUse thus we have that ζ' also conforms to $\Phi\theta$ w.r.t. NoUse. At last, $param(\zeta) \subseteq \{ax_1, \ldots, ax_{s-1}\}$ and $\zeta' \in st(\zeta)$ implies that $param(\zeta') \subseteq \{ax_1, \ldots, ax_{s-1}\}$.

Let $Y \in vars^2(\mathsf{C}\lfloor\zeta'\rfloor\mathsf{acc}^2(\mathcal{C}))$ and assume that $X <_\theta Y$. Thanks to Lemma C.40, it implies that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$ and so $s \leq k \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$. Furthermore $Y \in vars^2(\mathsf{C}\lfloor\zeta'\rfloor\mathsf{acc}^2(\mathcal{C}))$ also implies that there exists $(\gamma, i \rhd u) \in \Phi$ such that $\gamma\theta \in st(\zeta')$ and $Y \in vars^2(\gamma)$. But thanks to $\mathcal{C}$ satisfying $\mathsf{InvGeneral}$, $ax_i \in st(\gamma\theta)$. Thus with $param(\zeta') \subseteq \{ax_1, \ldots, ax_{s-1}\}$, we deduce that $i \leq s-1$. On the other hand, $Y \in vars^2(\gamma)$ implies that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) \leq i$. Hence we deduce that $s \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) \leq i \leq s-1$ which is a contradiction. Therefore, for all $Y \in vars^2(\mathsf{C}\lfloor\zeta'\rfloor\mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Y)$.

At last, since $param(\zeta') \subseteq \{ax_1, \ldots, ax_{s-1}\}$ and $ax_s \in st(\xi\theta)$ by the invariant $\mathsf{InvGeneral}$, we can deduce $\xi\theta \notin st(\zeta')$, and thus $\mathsf{C}\lfloor\xi\rfloor \notin st\mathsf{C}\lfloor\zeta'\rfloor$. Furthermore, by hypothesis, for all $(\xi', i \rhd v) \in \Phi$, for all $\mathsf{g} \in \mathcal{F}_d$, we have that $\mathsf{path}(\xi') \neq \mathsf{g} \cdot \mathsf{path}(\xi)$. Since $\xi\theta = X\theta_{|p}$ and p' is a prefix of p, then $\mathsf{C}\lfloor\xi\theta\rfloor = (\mathsf{C}\lfloor X\theta\rfloor)_{|p}$ and we can conclude that $\mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor X\theta[\zeta]_p\rfloor\}) < \mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor X\theta\rfloor\})$.

We can now apply the second property which means that there exists θ' such that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$ and

$$\mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor Y\theta'\rfloor \mid Y \in vars^2(\mathcal{C})\}) < \mathsf{nb}_{\mathsf{occ}}(\mathsf{C}\lfloor\xi\rfloor, \{\mathsf{C}\lfloor Y\theta\rfloor \mid Y \in vars^2(\mathcal{C})\})$$

Once again, since for all $(\xi', i \rhd v) \in \Phi$, for all $\mathsf{g} \in \mathcal{F}_d$, we have that $\mathsf{path}(\xi') \neq \mathsf{g} \cdot \mathsf{path}(\xi)$ and for all $Y \in vars^2(\mathcal{C})$, $Y\theta'$ conforms to $\Phi\theta'$ w.r.t. NoUse, then we can deduce that:

$$\mathsf{nb}_{\mathsf{occ}}(\xi\theta', \{Y\theta' \mid Y \in vars^2(\mathcal{C})\}) < \mathsf{nb}_{\mathsf{occ}}(\xi\theta, \{Y\theta \mid Y \in vars^2(\mathcal{C})\}).$$

□

C.5.4 Soundness

The purpose of this section is to prove the soundness of step e of Phase 1 of the strategy, the soundness of the normalisation and the following lemma.

Lemma 8.6 (soundness)**.** *Let $\mathcal{C}$ be a normalised constraint system obtained by following the strategy and* Rule$(\tilde{p})$ *be a transformation rule applicable on $\mathcal{C}$. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the two resulting constraint systems obtained by applying* Rule$(\tilde{p})$ *on $\mathcal{C}$. We denote by Φ, Φ_1 and Φ_2 the respective frames of $\mathcal{C}$, $\mathcal{C}_1$ and $\mathcal{C}_2$ and we denote by S_1 the set of free variables of $\mathcal{C}$.*

Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. There exist σ', θ', and $i_0 \in \{1, 2\}$ such that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}_{i_0})$, $\sigma = \sigma'|_{vars^1(\mathcal{C})}$ and $\mathsf{Init}(\Phi)\sigma = \mathsf{Init}(\Phi_{i_0})\sigma'$.

We distinguish two cases:

— An application of a rule during Phase 1, Step a. Note that in such a case, only the rules Dest and Eq-left-right are applicable.

— An application of a rule during 1.b, 1.c, 1.d or phase 2. Note that in such a case, only the rules Cons, Axiom, Eq-left-left, Eq-right-right and Ded-st are applicable.

An application during phase 1.a. In such a phase, only the rules Dest and Eq-left-right are applicable. Assume that we are on the cycle with parameter for support equal to s. We prove the result by case analysis on the rules.

Rule Dest$(\xi, l \to r, s)$: Let $\mathsf{g}(u_1, \ldots, u_n) \to u$ be a fresh variant of $l \to r$ and $\tilde{x}$ be the variables that occur in this variant. Note that $\mathsf{g} \in \mathcal{F}_d$. Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. We distinguish two cases.

Case 1: there exist ground recipes $\xi_2, \ldots, \xi_n$ in Π_n such that $\mathsf{g}(\xi\theta, \xi_2, \ldots, \xi_n)\Phi\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $param(\{\xi_2, \ldots, \xi_n\}) \subseteq \{ax_1, \ldots, ax_s\}$. To be more specific, since the rule Dest$(\xi, \ell \to r, s)$ is applicable and the constraint system was obtained by following the strategy, we deduce that $ax_s \in param(\{\xi_2, \ldots, \xi_n\})$ (else σ would not satisfy *ND*).

First, w.l.o.g., we can assume that for any strict subterm ξ'_k of ξ_k with $k \in \{2,\dots,n\}$, we have that $\xi'_k(\Phi\sigma) \neq \xi_k(\Phi\sigma)$ (otherwise, we can simply choose ξ'_k instead of ξ_k). Moreover, by Lemma C.36, we can also assume that $\xi_2,\dots,\xi_n$ conform to the frame $\Phi\theta$. Let $\tau = \mathsf{mgu}(\{(\xi\theta)\phi\sigma\downarrow = u_1,\ \xi_2(\phi\sigma)\downarrow = u_2,\ \dots,\ \xi_n(\phi\sigma)\downarrow = u_n\})$, and $\sigma_1 = \sigma \cup \tau$.

Our goal is to build a substitution θ' such that for all $X \in vars^2(\mathcal{C}_1)$, $X\theta'$ conforms to the frame $\Phi_1\theta'$ (where Φ_1 is the frame associated to $\mathcal{C}_1$). In particular, we have to ensure that there is a unique "key" that is used to decrypt $\xi\theta'$. Actually, we show how to build θ' in order to ensure that $X\theta'$ conforms to $\Phi_1\theta'$ for every $X \in \{Y \mid Y, j \overset{?}{\vdash} r \text{ in } \mathcal{C}_1\}$, we conclude for the remaining variables.

Let $S = \{Y \mid \{Y, j \overset{?}{\vdash} r\} \in \mathcal{C} \text{ and } \mathsf{g}(\xi\theta, \zeta_2, \dots, \zeta_n) \in st(Y\theta) \text{ for some } \zeta_2,\dots,\zeta_n\}$. Notice that for all $Y \in S$, $s \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$ else σ would not satisfy ND. We distinguish two cases:

Case a: $S = \emptyset$. Let θ' be a substitution defined as follows:

— $X_i\theta' = \xi_i$ for $i = 2 \dots n$, and

— $X\theta' = X\theta$ otherwise.

In such a case, it is relatively easy to conclude that $(\sigma_1, \theta') \in \mathsf{Sol}(\mathcal{C}_1)$. In particular, we have that $X\theta'$ conforms to $\Phi_1\theta'$ for all variable X. First $\Phi_1 = \Phi \cup \{\mathsf{g}(\xi, X_2, \dots, X_n), i \rhd w\}$. For every variable $X \notin \{X_2,\dots,X_n\}$, we have $X\theta' = X\theta$, which means that $\Phi\theta = \Phi\theta'$. Since $S = \emptyset$, we easily conclude that $Y\theta'$ conforms to $\Phi_1\theta'$ for all variables in $\{Y \mid Y, j \overset{?}{\vdash} r \in D(\mathcal{C}_1)\} \setminus \{X_2,\dots,X_n\}$. Furthermore, since ξ_i conforms to $\Phi\theta'$ and by the choice of ξ_i, we deduce that ξ_i conforms to $\Phi_1\theta'$. At last, also by the choice of ξ_i, we also deduce that $(X_i\theta')\Phi_1\sigma_1\downarrow = u_i\sigma_1$ which allows us to conclude.

Case b: $S \neq \emptyset$. Otherwise, we chose Y_0 a minimal variable w.r.t. the relation $<_\theta$ and the maximal parameter. Such minimal exists since by Lemma C.39, the relation $<_\theta$ is a strict partial order. We have that $\mathsf{g}(\xi, \zeta_2, \dots, \zeta_n) \in st(Y_0\theta)$ for some recipe $\zeta_2,\dots,\zeta_n$.

If $param(\{\zeta_2,\dots,\zeta_n\}) \not\subseteq \{ax_1,\dots,ax_s\}$ then for each $i \in \{2,\dots,n\}$, we denote by $\zeta_i^0 = \xi_i$, else for each $i \in \{2,\dots,n\}$, we denote by ζ_i^0 a minimal (for the size) subterm of ζ_i such that $\zeta_i(\Phi\sigma)\downarrow = \zeta_i^0(\Phi\sigma)\downarrow$.

Note that in both cases, for all $i \in \{2,\dots,n\}$, $\zeta_i(\Phi\sigma)\downarrow = \zeta_i^0(\Phi\sigma)\downarrow$. Indeed, we know that $\mathsf{g}(\xi\theta, \zeta_2, \dots, \zeta_n) \in st(Y\theta)$, $\mathsf{g}(\xi\theta, \zeta_2, \dots, \zeta_n) \in \Pi_n$ and $Y\theta(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Hence by Lemma 6.5, $\mathsf{g}(\xi, \zeta_2, \dots, \zeta_n)(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Furthermore, we know that $\mathsf{g}(\xi\theta, \xi_2, \dots, \xi_n)\Phi\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Hence since $\mathsf{path}(\mathsf{g}(\xi\theta, \zeta_2, \dots, \zeta_n)) = \mathsf{path}(\mathsf{g}(\xi\theta, \xi_2, \dots, \xi_n))$, then by Lemma C.34, $\zeta_i(\Phi\sigma)\downarrow = \xi_i(\Phi\sigma)\downarrow$, for all $i \in \{2,\dots,n\}$.

We rely on Lemma C.41 to build a substitution θ' that will be conformed to the frame $\Phi_1\theta'$. To achieve this, the idea is to replace any occurrence of $\mathsf{g}(\xi, \dots)$ that occur in $Y\theta$ with $Y \in S$ by $\mathsf{g}(\xi, \zeta_2^0, \dots, \zeta_n^0)$. We prove our result by induction on:

$$m = \mathsf{nb}_{\mathsf{occ}}(\mathsf{g}(\xi\theta, \__2, \dots, \__n), \{Y\theta \mid Y \in S\}) - \mathsf{nb}_{\mathsf{occ}}(\mathsf{g}(\xi\theta, \zeta_2^0, \dots, \zeta_n^0), \{Y\theta \mid Y \in S\})$$

where $\__i$ is used to represent any value.

Base case $m = 0$*:* Let θ' be the substitution defined as follows:

— $X_i\theta' = \zeta_i^0$ for $i = 2, \dots, n$, and

— $X\theta' = X\theta$ otherwise.

We conclude as in the previous case ($S = \emptyset$).

Inductive case $m > 0$*:* Let $Y \in S$ such that there exists $p \in \mathcal{P}os(\mathsf{C}\lfloor Y\theta \rfloor)$ such that $\mathsf{path}(Y\theta_{|p}) = \mathsf{g} \cdot \mathsf{path}(\xi)$ and $Y\theta_{|p} \neq \mathsf{g}(\xi\theta, \zeta_2^0, \dots, \zeta_n^0)$. We first show that such Y and p exists. We know that there is no frame element in Φ whose recipe has a path equal to $\mathsf{g} \cdot \mathsf{path}(\xi)$. Furthermore, thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 9), we deduce that no subterm of a recipe of a frame element in Φ has a path equal to $\mathsf{g} \cdot \mathsf{path}(\xi)$. Hence, we choosing Y minimal w.r.t. $<_\theta$ such that $\mathsf{g}(\xi\theta, \gamma_2, \dots, \gamma_n) \in st(Y\theta)$ for some $\gamma_2 \dots, \gamma_n$ and $\mathsf{g}(\xi\theta, \gamma_2, \dots, \gamma_n) \neq \mathsf{g}(\xi\theta, \zeta_2^0, \dots, \zeta_n^0)$, we

can conclude that there exists $p \in \mathcal{P}os(\mathsf{C}\lfloor Y\theta \rfloor)$ such that $\mathsf{path}(Y\theta_{|p}) = \mathsf{g} \cdot \mathsf{path}(\xi)$ and $Y\theta_{|p} \neq \mathsf{g}(\xi\theta, \zeta_2^0, \ldots, \zeta_n^0)$ (otherwise Y would not be minimal w.r.t. $<_\theta$).

Let $\zeta^0 \stackrel{\mathsf{def}}{=} \mathsf{g}(\xi\theta, \zeta_2^0, \ldots, \zeta_n^0)$.

— Since ζ^0 is a subterm of $Y_0\theta$, we know that ζ^0 conforms to $\Phi\theta$. Furthermore, by definition of each ζ_i^0, $i \in \{1, \ldots, n\}$, we know that $param(\zeta^0) \subseteq \{ax_1, \ldots, ax_s\}$. Moreover, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies $param(Y_0\theta) \subseteq \{ax_1, \ldots, ax_j\}$ hence ζ^0 subterm of $Y_0\theta$ implies that $param(\zeta^0)\{ax_1, \ldots, ax_s\} \subseteq \{ax_1, \ldots, ax_j\}$.

— Since $Y\theta_{|p}$ conforms with $\Phi\theta$ and $\mathsf{path}(Y\theta_{|p}) = g \cdot \mathsf{path}(\xi)$, there exists $\xi_2, \ldots, \xi_n$ such that $Y\theta = \mathsf{g}(\xi\theta, \xi_2, \ldots, \xi_n)$. Furthermore ξ is a recipe of a frame element in Φ, and there is no frame element in Φ having $\mathsf{g} \cdot \mathsf{path}(\xi)$ as a path. Hence, we have that $\mathsf{C}\lfloor Y\theta_{|p} \rfloor_{\Phi\theta} = \mathsf{g}(\mathsf{path}(\xi), \mathsf{C}\lfloor \xi_2 \rfloor_{\Phi\theta}, \ldots, \mathsf{C}\lfloor \xi_n \rfloor_{\Phi\theta})$ and $\mathsf{C}\lfloor \zeta^0 \rfloor_{\Phi\theta} = \mathsf{g}(\mathsf{path}(\xi), \mathsf{C}\lfloor \zeta_2^0 \rfloor_{\Phi\theta}, \ldots, \mathsf{C}\lfloor \zeta_n^0 \rfloor_{\Phi\theta})$. Thus, we deduce that $\mathsf{C}\lfloor Y\theta[\zeta^0]_p \rfloor = \mathsf{C}\lfloor Y\theta \rfloor [\mathsf{C}\lfloor \zeta^0 \rfloor]_p$. We have that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, thus $(Y\theta)(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $(Y_0\theta)(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Thanks to Lemma 6.5, $(Y\theta|_p)(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $\zeta^0(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Since $\mathsf{path}(Y\theta|_p) = \mathsf{path}(\zeta^0)$, by Lemma C.34, we conclude that$(Y\theta|_p)(\Phi\sigma)\downarrow = \zeta^0(\Phi\sigma)\downarrow$.

— Lastly, by definition of $\zeta_2^0, \ldots, \zeta_n^0$, either (a) $\zeta_i^0 = \zeta_i$, for $i \in \{2, \ldots, n\}$ and ζ_i^0 are subterms of $Y_0\theta$; or (b) $param(\mathsf{g}(\xi\theta, \zeta_2, \ldots, \zeta_n)) \not\subseteq \{ax_1, \ldots, ax_s\}$ and $\zeta_i^0 = \xi_i$, for $i \in \{2, \ldots, n\}$.

In case (a), since $Y, Y_0 \in S$ and Y_0 is a minimal variable w.r.t. $<_\theta$ then we deduce that for all $Z \in vars^2(\mathsf{C}\lfloor \zeta^0 \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, $\neg(Y <_\theta Z)$. In case (b), we know that Y_0 is also minimal w.r.t. the maximal parameter. Hence $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y_0) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$. But $param(\mathsf{g}(\xi\theta, \zeta_2, \ldots, \zeta_n)) \not\subseteq \{ax_1, \ldots, ax_s\}$ and $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y_0) > s$ and so $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) > s$. Since $param(\mathsf{g}(\xi\theta, \xi_2, \ldots, \xi_n)) \subseteq \{ax_1, \ldots, ax_s\}$, $\mathcal{C}$ satisfies $\mathsf{InvUntouched}(s)$, $\mathcal{C}$ is a well-formed constraint system (item 3) and by Lemma C.40, then we deduce that for all $Z \in vars^2(\mathsf{C}\lfloor \zeta^0 \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, $\neg(Y <_\theta Z)$.

We satisfy all the conditions required to apply Lemma C.41, Hence, there exists θ' such that (σ, θ') is a pre-solution of $\mathcal{C}$ with $Y\theta' = Y\theta[\zeta^0]_p$ and for all $Z \in vars^2(\mathcal{C}) \setminus \{Y\}$, we have that $\mathsf{C}\lfloor Z\theta \rfloor_{\Phi\theta} = \mathsf{C}\lfloor Z\theta' \rfloor_{\Phi\theta'}$,

Hence, we have that the measure m strictly decreases. Furthermore, since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and using the fact that (σ, θ') is a pre-solution of $\mathcal{C}$, it only remains to prove that $\theta' \models Er$ in order to conclude that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$. Actually, we know that $\theta' \vDash \mathsf{mgu}(Er)$ and thanks to $\mathcal{C}$ satisfying $\mathsf{PP1Sa}(s)$ (item C.2), we have trivially have that $\theta' \vDash Er$ and so $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$. Then, we conclude by relying on our induction hypothesis.

Case 2: for all ground recipes $\xi_2, \ldots, \xi_n$ in Π_n, either $param(\{\xi_2, \ldots, \xi_n\}) \not\subseteq \{ax_1, \ldots, ax_s\}$ or we have that $\mathsf{g}(\xi\theta, \xi_2 \ldots, \xi_n)\Phi\sigma\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Since $\mathsf{g} \in \mathcal{F}_d$, $\mathsf{g}(\xi\theta, \xi_2 \ldots, \xi_n)\Phi\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ means that there exists a substitution τ which maps variable in $\tilde{x}$ to ground constructor terms such that

$$\mathsf{g}(u_1\tau, \ldots, u_n\tau) = \mathsf{g}(\xi\theta(\Phi\sigma)\downarrow, \xi_2(\Phi\sigma)\downarrow, \ldots, \xi_n(\Phi\sigma)\downarrow).$$

This means that $u_1\tau = \xi\theta(\Phi\sigma)\downarrow$, $u_2\tau = \xi_2(\Phi\sigma)\downarrow$, …, and $u_n\tau = \xi_n(\Phi\sigma)\downarrow$. Moreover, since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we have that $(\xi\theta)(\Phi\sigma)\downarrow = v\sigma$. Therefore, we have that:

$$\sigma \models \forall \tilde{x} \cdot [v \neq u_1 \vee s \stackrel{?}{\not\vdash} u_2 \vee \ldots \vee s \stackrel{?}{\not\vdash} u_n]$$

This allows us to conclude that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$.

Rule Eq-left-right(X, ξ): By hypothesis we know that there exists u, v and k such that $(X, k \stackrel{?}{\vdash} u) \in D(\mathcal{C})$, $(\xi, s \rhd v) \in \Phi(\mathcal{C})$ and $k < s$. Furthermore, according to the strategy, we know that the rule Eq-left-right is prioritised over the rule Dest, and that the rule Eq-left-right is strongly applicable on at least on constraint system on the row of the matrix of constraint system. Assume that $\mathcal{C}'$ is such constraint system. By hypothesis, $\mathcal{C}'$ is normalised. It would imply that

there exists $x \in \mathcal{X}^1$ such that $(X, k \overset{?}{\vdash} x) \in D(\mathcal{C}')$ and $(\xi, s \rhd x) \in \Phi(\mathcal{C}')$. Assume now that there exists $(\xi', \ell \rhd w) \in \Phi(\mathcal{C}')$ and $\mathsf{g} \in \mathcal{F}_d$ such that $\mathsf{path}(\xi') = \mathsf{g} \cdot \mathsf{path}(\xi)$, hence it means that an instance of the rule Dest was previously applied on $(\xi, s \rhd x)$. But according to the definition of our rewrite rules and since $\mathcal{C}'$ is normalised, it would imply that x is instantiated by a term different from a variable and so $x \notin \mathcal{X}^1$ which is a contraction. Hence, for all $(\xi', \ell \rhd w) \in \Phi(\mathcal{C}')$, for all $\mathsf{g} \in \mathcal{F}_d$, $\mathsf{path}(\xi') \neq \mathsf{g} \cdot \mathsf{path}(\xi)$. But by Lemma 8.1 and since $\mathcal{C}$ and $\mathcal{C}'$ are on the same row of M, we can deduce that $\mathcal{C}'$ and $\mathcal{C}$ have the same structure and so for all $(\xi', \ell \rhd w) \in \Phi(\mathcal{C})$, for all $\mathsf{g} \in \mathcal{F}_d$, $\mathsf{path}(\xi') \neq \mathsf{g} \cdot \mathsf{path}(\xi)$.

Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. We distinguish two cases:

1. $u\sigma\downarrow = v\sigma\downarrow$. In such a case, we need to transform θ such that the frame element $(\xi, s \rhd v)$ will not be used anymore. Let denote $Nb(\theta) = \mathsf{nb}_{\mathsf{occ}}(\xi\theta, \{Y\theta \mid Y \in vars^2(\mathcal{C})\})$. We show that there exists θ' such that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C}_1)$. We prove this result by induction on $Nb(\theta)$.

   Base case $Nb(\theta) = 0$*:* We know that $\mathcal{C}_1$ is $\mathcal{C}$ where $\mathsf{NoUse}(\mathcal{C}_1) = \mathsf{NoUse}(\mathcal{C}) \cup (\xi, s \rhd v)$ and $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u \overset{?}{=} v$. By hypothesis, we already know that $u\sigma\downarrow = v\sigma\downarrow$, hence $\sigma \vDash Eq(\mathcal{C}_1)$. Hence it remains to prove that for all $Y \in vars^2(\mathcal{C})$, $Y\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C}_1)$. But we already know thanks to $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ that $Y\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})$. And since $Nb(\theta) = 0$, we can conclude that $Y\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C}_1)$ and so $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$.

   Inductive step $Nb(\theta) > 0$*:* Since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and $u\sigma\downarrow = v\sigma\downarrow$, we have that $X\theta(\Phi\sigma)\downarrow = \xi\theta(\Phi\sigma)\downarrow$ and $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$ with $k < s$. Moreover, we proved that for all $(\xi', \ell \rhd w) \in \Phi(\mathcal{C})$, for all $\mathsf{g} \in \mathcal{F}_d$, $\mathsf{path}(\xi') \neq \mathsf{g} \cdot \mathsf{path}(\xi)$. At last, since the rule Dest and Eq-left-right does not add equations in Er and Step a is the first step applied during Phase 1 with parameter s, we deduce that $\xi \notin st(\mathsf{mgu}(Er))$. Hence by Lemma C.42, we can deduce that there exists θ' such that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$ and $Nb(\theta') < Nb(\theta)$. Since $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$ and $u\sigma\downarrow = v\sigma\downarrow$, we have still have that $X\theta'(\Phi\sigma)\downarrow) = \xi\theta'(\Phi\sigma)\downarrow$. Hence, by our inductive hypothesis on θ', we can deduce that there exists θ'' such that $(\sigma, \theta'') \in \mathsf{Sol}(\mathcal{C}_1)$.
2. $u\sigma\downarrow \neq v\sigma\downarrow$. In such a case, it is easy to see that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$. □

An application during phase 1.*b,* 1.*c,* 1.*d or* 2. Rule Cons(X, f) : Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. We distinguish two cases:

1. $\mathsf{root}(X\theta) = \mathsf{f}$. In such a case, there exists $\xi_1, \ldots, \xi_n \in \Pi_n$ such that $X\theta = \mathsf{f}(\xi_1, \ldots, \xi_n)$. Since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and $\mathsf{f} \in \mathcal{F}_c$, we deduce that $param(X\theta) \subseteq \{ax_1, \ldots, ax_i\}$, and

$$(X\theta)(\Phi\sigma)\downarrow = \mathsf{f}(\xi_1(\Phi\sigma)\downarrow, \ldots, \xi_n(\Phi\sigma)\downarrow) = \mathsf{f}(t_1, \ldots, t_n) = t\sigma\downarrow$$

   for some terms $t_1, \ldots, t_n$.
   Let $\theta' = \theta \cup \{X_1 \mapsto \xi_1, \ldots, X_n \mapsto \xi_n\}$ and $\sigma' = \sigma \cup \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$. Since $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$, we trivially have that $\Phi\sigma = \Phi\sigma'$ and thus for every $i \in \{1, \ldots, n\}$, we have that $(X_i\theta')(\Phi\sigma')\downarrow = t_i\downarrow$ and $param(X_i\theta') \subseteq param(X\theta) \subseteq \{ax_1, \ldots, ax_i\}$. Furthermore, $t\sigma' = \mathsf{f}(t_1, \ldots, t_n)$ and $x_i\sigma' = t_i$, for all $i \in \{1, \ldots, n\}$ implies that $t\sigma'\downarrow = \mathsf{f}(x_1, \ldots, x_n)\sigma'\downarrow$. At last, by definition of θ', we also have that $X\theta' = \mathsf{f}(X_1, \ldots, X_n)\theta'$. This allows us to conclude that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}_1)$.
2. $\mathsf{root}(X\theta) \neq \mathsf{f}$. In such a case, we have that $\theta \models Er \wedge \mathsf{root}(X) \neq \mathsf{f}$ and so we can conclude that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$.

Rule Axiom(X, path). Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. We distinguish two cases:

1. $\mathsf{path}(X\theta) = \mathsf{path}$. In such a case, by definition of $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we have that $X\theta$ conforms to $\Phi\theta$ w.r.t. NoUse, and thus $X\theta = \xi\theta$. We have also that $(X\theta)(\Phi\sigma)\downarrow = u\sigma\downarrow$. Lastly, since $\mathcal{C}$ is well-formed, we know that $(\xi\theta)(\Phi\sigma)\downarrow = v\sigma\downarrow$. Altogether, this allows us to deduce that $u\sigma\downarrow = v\sigma\downarrow$. We conclude that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$.

2. $\mathsf{path}(X\theta) \neq \mathsf{path}$. Since $\mathsf{path}(\xi) = \mathsf{path}$, $\mathsf{path}(X\theta) \neq \mathsf{path}$ implies that $X\theta \neq \xi$. Thus, $\theta \vDash Er \wedge X \neq \xi$. We can conclude that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$.

Rule Eq-left-left(ξ_1, ξ_2). Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. We distinguish two cases:

1. $u_1\sigma\downarrow = u_2\sigma\downarrow$. In such a case, it is easy to see that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$.
2. $u_1\sigma\downarrow \neq u_2\sigma\downarrow$. In such a case, it is easy to see that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$.

Thus, in both cases, we easily conclude.

Rule Eq-right-right(X, ξ). Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. We distinguish two cases:

1. $u\sigma\downarrow = v\sigma\downarrow$. In such a case, since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, then for all $Y \in vars^2(\xi)$, we have that $Y\theta$ conforms to the frame $\Phi\theta$ w.r.t. NoUse. But the conditions of the rule Eq-right-right tell us that $\xi \in \mathcal{T}(\mathcal{F}_c, vars^2(\alpha))$ where $\alpha = \{Y \to w \mid (Y, j \overset{?}{\vdash} w) \in D(\mathcal{C}) \wedge j \leq i \wedge Y \in S_2(\mathcal{C})\}$ which means that $\mathsf{C}\lfloor \xi\theta \rfloor_{\Phi\theta} = \xi\{Y \to \mathsf{C}\lfloor Y\theta \rfloor_{\Phi\theta} \mid Y \in vars^2(\xi)\}$. Thus, we deduce that $\xi\theta$ conforms to $\Phi\theta$ too.

   Moreover, the conditions of the rule Eq-right-right also tell us that $v = \xi\alpha$. By $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we deduce that for all $Y \in vars^2(\xi)$, $(Y\theta)\Phi\sigma\downarrow = (Y\alpha)\sigma\downarrow$. Once again, since $\xi \in \mathcal{T}(\mathcal{F}_c, \mathrm{dom}(\alpha))$, we have that $(\xi\theta)\Phi\sigma\downarrow = v\sigma = u\sigma = (X\theta)\Phi\sigma\downarrow$.

   We want to conclude the result by applying Lemma C.41. But in order to do that, we need to prove that for all $Y \in vars^2(\mathsf{C}\lfloor \xi\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Y)$. We prove this property by case analysis on ξ:

   Case $\xi \in vars^2(D)$: In such a case, we denote ξ by Z and so there exists $(Z, j \overset{?}{\vdash} v) \in D$. For all $Y \in vars^2(\mathsf{C}\lfloor \xi\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, we have that $Y <_\theta Z$ by definition of $<_\theta$. Hence, if $\neg(X <_\theta Z)$ then for all $Y \in vars^2(\mathsf{C}\lfloor \xi\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, $\neg(X <_\theta Y$. We apply Lemma C.41 on the deducibility constraint $X, i \overset{?}{\vdash} u$ with the recipe $Z\theta$; otherwise we have $X <_\theta Z$ and so since $<_\theta$ is a strict partial order by Lemma C.39, we deduce that $\neg(Z <_\theta X)$. Thus, we apply Lemma C.41 on the deducible constraint $Z, j \overset{?}{\vdash} v$ with the recipe $X\theta$. Therefore, in both case, we know that there exists θ' such that (σ, θ') is a pre-solution of $\mathcal{C}$ with $X\theta' = Z\theta'$, $\theta' \vDash \mathsf{mgu}(Er)$ and for all $Y \in vars^2(D)\backslash\{X, Z\}$, we have $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi\theta} = \mathsf{C}\lfloor Y\theta' \rfloor_{\Phi\theta'}$. At last, by the condition of Eq-right-right, we know that $\mathsf{root}(X) \overset{?}{\neq} \mathsf{f} \in Er$ is equivalent to $\mathsf{root}(Z) \overset{?}{\neq} \mathsf{f} \in Er$, thus (relying on the form of the inequations in Er) we can deduce that $\theta' \vDash Er$ and so $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C}_1)$.

   Case $\xi \notin vars^2(D)$: As explained in the strategy (Section 7.4), the rule Eq-right-right with such parameter is only applied when the strategy is on the second phase. But, in such a case, since $\mathcal{C}$ satisfies the invariant $\mathsf{InvVarFrame}(\infty)$, we have that $(\star)$:

   $$\text{for all } (\zeta, k \rhd u) \in \Phi, \text{ for all } Z \in vars^2(\zeta), \text{ there exists } j < k \text{ and } z \in \mathcal{X}^1 \text{ such that } (Z, j \overset{?}{\vdash} z) \in D.$$

   Moreover, we proved that $\mathsf{C}\lfloor \xi\theta \rfloor_{\Phi\theta} = \xi\{Y \to \mathsf{C}\lfloor Y\theta \rfloor_{\Phi\theta} \mid Y \in vars^2(\xi)\}$, hence for all $Y \in vars^2(\mathsf{C}\lfloor \xi\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$, there exists $Z \in vars^2(\xi)$ such that $Y \in vars^2(\mathsf{C}\lfloor Z\theta \rfloor_\Phi \mathsf{acc}^2(\mathcal{C}))$. It implies that there exists $(\zeta, k \rhd u) \in \Phi$ such that $Y \in vars^2(\zeta)$ and $\zeta\theta \in st(Z\theta)$. Since $\mathcal{C}$ also satisfies $\mathsf{InvGeneral}$, we know that $ax_k \in st(\zeta\theta)$ and so we have that $k \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z)$. But thanks to $(\star)$, we have that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) < k$, which implies that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) < \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z)$. But by definition of ξ, we have that $Z \in vars^2(\xi)$, and thus we have that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X)$, hence we conclude, thanks to Lemma C.40, that $\neg(X <_\theta Y)$.

   Thus we apply Lemma C.41 on the deducible constraint $X, i \overset{?}{\vdash} u$ with the recipe $\xi\theta$. Therefore, there exists θ' such that (σ, θ') is a pre-solution of $\mathcal{C}$ with $X\theta' = \xi\theta$ and for

all $Y \in vars^2(\mathcal{C})\backslash\{X\}$, we have $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi\theta} = \mathsf{C}\lfloor Y\theta'\rfloor_{\Phi\theta'}$. At last, by the condition of Eq-right-right, we know that $\mathsf{root}(\xi) = \mathsf{f}$ implies $\mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f} \in Er$, thus we can deduce that $\theta' \vDash Er$ and so $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C}_1)$.

2. $u\sigma\downarrow \neq v\sigma\downarrow$. In such a case, it is easy to see that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$.

Rule Ded-st(ξ, f). Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. We distinguish two cases:

1. There exist ground recipes $\xi_1, \ldots, \xi_n$ in Π_n such that $\mathsf{f}(\xi_1, \ldots, \xi_n)(\Phi\sigma)\downarrow = u\sigma\downarrow$. In such a case, we can assume w.l.o.g. (see Lemma C.36) that $\mathsf{f}(\xi_1, \ldots, \xi_n)$ conforms to $\Phi\theta$ w.r.t. NoUse, and thus $\xi_1, \ldots, \xi_n$ conform also to the frame $\Phi\theta$ w.r.t. NoUse. For every $j \in \{1, \ldots, n\}$, let $t_j = \xi_j(\Phi\sigma)\downarrow$. Let $\theta' = \theta \cup \{X_1 \mapsto \xi_1, \ldots, X_n \mapsto \xi_n\}$, and $\sigma' = \sigma \cup \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$. Clearly, we have that $X\theta'$ conforms to $\Phi\theta'$ for every $X \in vars^2(\mathcal{C}_1)$. Since f is a constructor symbol, $\mathsf{f}(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow = u\sigma\downarrow$ implies $\sigma' \vDash u = \mathsf{f}(x_1, \ldots, x_n)$. Moreover, since m is the maximal index that occurs in $\mathcal{C}$, we have that $param(X_i\theta') \subseteq \{ax_1, \ldots, ax_m\}$ and thus $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$.

2. Otherwise, for all ground recipes $\xi_1, \ldots, \xi_n$ in Π_n, we necessarily have that $\mathsf{f}(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow \neq u\sigma\downarrow$. Since f is a constructor symbol, we have that

$$\mathsf{f}(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow = \mathsf{f}(\xi_1\Phi\sigma\downarrow, \ldots, \xi_n\Phi\sigma\downarrow).$$

We can distinguish two cases: either $\mathsf{root}(u\sigma) \neq \mathsf{f}$ or else there exists $i \in \{1 \ldots n\}$, terms $t_1, \ldots, t_n$ such that $u\sigma = \mathsf{f}(t_1, \ldots, t_n)$ and $\xi_i\Phi\sigma\downarrow \neq t_i\downarrow$ for any ground recipe ξ_i. Therefore, we have that:

$$\sigma \vDash \forall \tilde{x} \cdot [u \neq \mathsf{f}(x_1, \ldots, x_n) \vee m \stackrel{?}{\nvdash} x_1 \vee \ldots \vee m \stackrel{?}{\nvdash} x_n].$$

This allows us to conclude that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_2)$. □

Lemma 8.4. *Let $\mathcal{C}$ be a constraint system obtained by following the strategy.* $\mathsf{Sol}(\mathcal{C}) = \mathsf{Sol}(\mathcal{C}\downarrow)$.

Proof. The rules for normalisation presented in Figure 7.3 corresponds to classic transformation on formula of first order logic. Once can easily prove that all the rules in Figure 7.3 preserves the set of solutions. Hence we focus on the two rules presented in Figure 7.4.

Rule (Nname): In such a case, $Eq = Eq' \wedge \forall \tilde{x}.[Eq'' \vee x \stackrel{?}{\neq} a]$, $a \in \mathcal{N}$ and $(X, i \stackrel{?}{\vdash} x) \in D$. Thus $x \notin \tilde{x}$. Moreover, we have that Axiom(X, path) is useless for any path, and Dest$(\xi, \ell \to r, i)$ is useless for any ξ, $\ell \to r$.

Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. We show that $x\sigma \neq a$. Thanks to Lemma C.37, $\mathsf{C}\lfloor X\theta\rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX})$. Moreover, But Axiom$(X, \mathsf{path})$ is useless for any path. Hence either Axiom(X, path) is not applicable or its application results in two constraint systems $\mathcal{C}_1$ and $\mathcal{C}_2$ such that $\mathcal{C}_1$ simplifies to $\bot$ using the rules in Figure 7.3 and $\mathcal{C}_2$ simplifies into $\mathcal{C}$. But if there exists $(\xi, j \rhd v) \in \Phi$ such that $\mathsf{path}(\xi)$, $j \leq i$ and $(\xi, j \rhd u) \notin \mathsf{NoUse}$ then the application of $\mathcal{C}_1$ add the equation $X \stackrel{?}{=} \xi$ in Er, and the equation $x \stackrel{?}{=} v$ in Eq. Hence $\mathcal{C}_1$ simplifies in $\bot$ implies that $X\theta \neq \xi\theta$ or $x\sigma \neq v\sigma$. However, thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 5) and $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we deduce that $X\theta(\Phi\sigma)\downarrow = x\sigma$ and $\xi\theta(\Phi\sigma)\downarrow = v\sigma$. Hence $x\sigma \neq v\sigma$ implies $X\theta \neq \xi\theta$.

Hence Axiom(X, path) is useless for any path implies that for all $(\zeta, j \rhd v) \in \Phi(\mathcal{C})$, if $j \leq i$ and $(\zeta, j \rhd v) \notin \mathsf{NoUse}$ then $X\theta \neq \zeta\theta$. But we know that $\mathcal{C}$ satisfies $\mathsf{InvGeneral}$ thus for all $(\zeta, j \rhd v) \in \Phi(\mathcal{C})$, $ax_j \in st(\zeta\theta)$. Thus, $paramX\theta \subseteq \{ax_1, \ldots, ax_i\}$ and $X\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$ implies that for all $w \in \mathsf{C}\lfloor X\theta\rfloor_\Phi \cap \mathcal{F}_d^* \cdot \mathcal{AX}$, there exists $(\zeta, j \rhd v) \in \Phi(\mathcal{C})$ such that $\mathsf{path}(\zeta) = w$ and $j \leq s$. Since we already proved that in this case, $X\theta \neq \xi\theta$, then we deduce that $\mathsf{C}\lfloor X\theta\rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX})$ and $|\mathsf{C}\lfloor X\theta\rfloor_\Phi| > 1$. Along with $X\theta(\Phi\sigma)\downarrow = x\sigma$, it implies that $|x\sigma| > 1$ and so $x\sigma \neq a$.

Rule (Nnosol): In such a case, we have $(X, i \stackrel{?}{\vdash} u) \in D$, Cons$(X, \mathsf{f})$ is useless for any $\mathsf{f} \in \mathcal{F}_c$, Axiom$(X, \mathsf{path})$ is useless for any path and Dest$(\xi, \ell \to r, i)$ is useless for all $\xi, \ell \to r$.

Let $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$. In the case of rule (Nname), we showed that $\mathsf{C}\lfloor X\theta \rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX})$ and $|\mathsf{C}\lfloor X\theta \rfloor_\Phi| > 1$. But $\textsc{Cons}(X, \mathsf{f})$ is useless for any $\mathsf{f} \in \mathcal{F}_c$. Hence either $\textsc{Cons}(X, \mathsf{f})$ is not applicable or its application results in two constraint systems $\mathcal{C}_1$ and $\mathcal{C}_2$ such that $\mathcal{C}_1$ simplifies to $\bot$ using the rules in Figure 7.3 and $\mathcal{C}_2$ simplifies into $\mathcal{C}$.

Let $\mathsf{f} \in \mathcal{F}_c$. Since $(X, i \overset{?}{\vdash} u) \in D$, then $\textsc{Cons}(X, \mathsf{f})$ is applicable. According to Figure 7.1, its application adds an equation $X \overset{?}{=} \mathsf{f}(X_1, \ldots, X_n)$ in Er and $u \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n)$ where $X_1, \ldots, X_n$ and $x_1, \ldots, x_n$ are fresh variables. Since $X_1, \ldots, X_n$, $x_1, \ldots, x_n$ are fresh, C_1 simplifying to $\bot$ implies that $\mathsf{root}(X\theta) \neq \mathsf{f}$ or $\mathsf{root}(u\sigma) \neq \mathsf{f}$. But $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$ implies that $(X\theta)(\Phi\sigma)\downarrow = u\sigma$. Moreover, along with $\mathsf{C}\lfloor X\theta \rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX})$ and $|\mathsf{C}\lfloor X\theta \rfloor_\Phi| > 1$, C_1 simplifying to $\bot$ implies that $\mathsf{root}(u\sigma) = \mathsf{root}(X\theta)$. Hence we deduce that $\mathsf{root}(X\theta) \neq \mathsf{f}$. Hence, we proved that for all $\mathsf{f} \in \mathcal{F}_c$, $\mathsf{root}(X\theta) \neq \mathsf{f}$ which is a contradiction with $\mathsf{C}\lfloor X\theta \rfloor_\Phi \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX})$ and $|\mathsf{C}\lfloor X\theta \rfloor_\Phi| > 1$. Hence $(\sigma,\theta) \notin \mathsf{Sol}(\mathcal{C})$ and so $\mathsf{Sol}(\mathcal{C}) = \emptyset = \mathsf{Sol}(\bot)$. □

Lemma C.43. *Let (M, M') be a pair of matrices of constraint systems. Let k with the number of column in M and M'. Assume that (M, M') is obtained at the end of Step d of Phase 1 of the strategy with parameter s for the support and k for the index of the column. Let $\mathcal{C}$ be a constraint system in M or M'. If $\mathcal{C}$ is replaced by $\bot$ when applying Step e of Phase 1 of the strategy, then we have that $\mathsf{Sol}(\mathcal{C}) = \emptyset$.*

Proof. Let $\mathcal{C}$ be a constraint system in M or M' such that $\mathcal{C}$ is replaced by $\bot$ when applying Step e of Phase 1 of the strategy. By definition, we know that there is two conditions that trigger the replacement of $\mathcal{C}$ by $\bot$. We prove that if one of the two conditions is satisfied then $\mathsf{Sol}(\mathcal{C}) = \emptyset$.

Condition 1: By definition, we know that there exists a constraint system $\mathcal{C}'$ in the same column as $\mathcal{C}$, a recipe ξ, such that:

— $(\xi, i \rhd u) \in \Phi(\mathcal{C}')$ for some $i \leq s$ and u

— for all $(\xi', j \rhd v) \in \Phi(\mathcal{C})$, $\mathsf{path}(\xi) \neq \mathsf{path}(\xi')$

Let $w \cdot ax_k = \mathsf{path}(\xi)$. We prove by induction on $|w|$ that there exists w' suffix of w, $(\zeta, j \in u) \in \Phi(\mathcal{C})$, $(\zeta', j' \in u') \in \Phi(\mathcal{C}')$ such that:

— $\mathsf{path}(\zeta) = \mathsf{path}(\zeta') = w' \cdot ax_k$, $j \leq s$ and $j' \leq s$.

— there exists $(\zeta'', j'' \rhd u'') \in \Phi(\mathcal{C}')$ such that $\mathsf{path}(\zeta'') = \mathsf{g} \cdot \mathsf{path}(\zeta)$ for some $\mathsf{g} \in \mathcal{F}_d$.

— for all $(\zeta''', j''' \rhd u''') \in \Phi(\mathcal{C})$, $\mathsf{path}(\zeta''') \neq \mathsf{path}(\zeta'')$.

Base case $|w| = 0$: In such a case, we have that $\xi = ax_k$. But $\mathcal{C}$ and $\mathcal{C}'$ are both originated from the same initial constraint system, thus we know that there exists u, u' such that $(ax_k, k \rhd u) \in \Phi(\mathcal{C})$ and $(ax_k, k \rhd u') \in \Phi(\mathcal{C}')$. Hence there is a contradiction with our hypothesis on ξ.

Inductive step $|w| > 0$: Otherwise, there exists w' and $\mathsf{g} \in \mathcal{F}_d$ such that $w = \mathsf{g} \cdot w' \cdot ax_k$. But $\mathcal{C}'$ is a well-formed constraint system, hence by Property 2 of a well formed constraint system, $(\xi, i \rhd u) \in \Phi(\mathcal{C}')$ implies that there exists $(\zeta', j' \rhd v') \in \Phi(\mathcal{C}')$ such that $\mathsf{path}(\zeta') = w' \cdot ax_k$ and $i \leq j'$.

Hence if there exists $(\zeta, j \rhd v) \in \Phi(\mathcal{C})$ such that $\mathsf{path}(\zeta) = \mathsf{path}(\zeta')$ then the result holds with $(\zeta'', j'' \rhd u'') = (\xi, i \rhd u)$. Else, since $|w'| < |w|$, we can apply our inductive hypothesis on $(\zeta', j' \rhd v')$ and so the result also holds.

Main proof for Condition 1: Thanks to Lemma C.31, we know that (M, M') satisfies $\mathsf{PP1Sb}(s, k+1)$. Since k corresponds to the number of column in M and M', we deduce that all constraint systems in M or M' satisfies $\mathsf{InvVarConstraint}(s)$, $\mathsf{InvUntouched}(s)$ and $\mathsf{InvDest}(s)$.

Let $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$. Thanks to $\mathcal{C}$ satisfying $\mathsf{InvDest}(s)$, $(\zeta, j \in u) \in \Phi(\mathcal{C})$ and for all $(\zeta''', j''' \rhd u''') \in \Phi(\mathcal{C})$, $\mathsf{path}(\zeta''') \neq \mathsf{g} \cdot \mathsf{path}(\zeta)$, we deduce that $\sigma \vDash ND(\mathcal{C})$ implies that there is no recipe $\zeta_2, \ldots, \zeta_n \in \Pi_n$ such that $\mathsf{g}(\zeta\theta, \zeta_2, \ldots, \zeta_n)(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta_i) \leq s$ for all $i \in \{2, \ldots, n\}$.

But $(\zeta'', j'' \rhd u'') \in \Phi(\mathcal{C}')$ with $\mathsf{path}(\zeta'') = \mathsf{g} \cdot \mathsf{path}(\zeta)$. Hence there exists $\xi_2, \ldots, \xi_n$ such that $\zeta'' = \mathsf{g}(\zeta', \xi_2, \ldots, \xi_n)$. We will show that $\zeta'\theta(\Phi(\mathcal{C})\sigma)\downarrow = \zeta\theta(\Phi(\mathcal{C})\sigma)\downarrow$ and $\zeta''\theta(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$

with $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi_i\theta) \leq s$, for all $i \in \{2, \ldots, n\}$. Hence it will contradict the fact that $\sigma \vDash ND(\mathcal{C})$ and so it will implies that $\mathsf{Sol}(\mathcal{C}) = \emptyset$.

We know that $\mathcal{C}$ and $\mathcal{C}'$ satisfy $\mathsf{InvVarConstraint}(s)$ and $\mathsf{InvUntouched}(s)$. Moreover, thanks to Lemma C.29, we also know that there exists a variable renaming $\rho : \mathcal{X}^1 \setminus S_1(\mathcal{C}) \to \mathcal{X}^1 \setminus S_1(\mathcal{C}')$ such that:

1. $\mathsf{mgu}(Eq(\mathcal{C}))_{|S_1(\mathcal{C})}\rho = \mathsf{mgu}(Eq(\mathcal{C}'))_{|S_1(\mathcal{C}')}$, and $D(\mathcal{C})\rho = D(\mathcal{C}')$;
2. $\{(u\rho, u') \mid (\xi, i \rhd u) \in \Phi \wedge (\xi', i' \rhd u') \in \Phi' \wedge \mathsf{path}(\xi) = \mathsf{path}(\xi')\}$ is include in $\{(u, u) \mid u \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)\}$;

$(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies that for all $X, k \overset{?}{\vdash} t \in D(\mathcal{C})$, we have $(X\theta)(\Phi(\mathcal{C})\sigma)\downarrow = t\sigma$. Moreover, we know that for all $(ax_\ell, \ell \rhd v) \in \Phi(\mathcal{C})$, $(ax_\ell, \ell \rhd v') \in \Phi(\mathcal{C}')$, $v\rho = v'$ which means that $(X\theta)(\Phi(\mathcal{C})\sigma)\downarrow = t\sigma$ implies $(X\theta)(\Phi(\mathcal{C}')\rho^{-1}\sigma)\downarrow = t'\rho^{-1}\sigma$ with $(X, k \overset{?}{\vdash} t') \in D(\mathcal{C}')$ and $t\rho = t'$.

But, for all $X \in vars^2(\zeta')$, by Property 3 of a well formed constraint system, we know that $\mathsf{param}^{\leq}_{\mathsf{max}}(\zeta')j'$ and so there exists $(X, k \overset{?}{\vdash} t') \in D(\mathcal{C}')$ such that $k \leq j'$. Thus $(X\theta)(\Phi(\mathcal{C}')\rho^{-1}\sigma)\downarrow = t'\rho^{-1}\sigma$. $\mathcal{C}$ being well-formed (item 5) allows us to deduce that $(\zeta'\theta)(\Phi(\mathcal{C}')\rho^{-1}\sigma)\downarrow = u'\rho^{-1}\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Hence, we have that $(\zeta'\theta)(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma\downarrow = (\zeta\theta)(\Phi(\mathcal{C})\sigma)\downarrow$.

Similarly, we have that $(\zeta''\theta)(\Phi(\mathcal{C})\sigma)\downarrow = u''\rho^{-1}\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. It remains to show that for all $i \in \{2, \ldots, n\}$, $param(\xi_i\theta) \subseteq \{ax_1, \ldots, ax_s\}$. Since $\mathcal{C}'$ satisfies the invariant $\mathsf{InvUntouched}(s)$, we know that $j'' \leq s$. But $\mathcal{C}'$ is well-formed (item 3) hence we deduce that $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(\zeta'') \leq j''$. Since $\mathcal{C}$ and $\mathcal{C}'$ have the same shape and satisfy $\mathsf{InvVarConstraint}(s)$, we deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta'') = \mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(\zeta'')$. But for all $X \in vars^2(\zeta'')$, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X)$. Hence, along with $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta'') \leq j''$, it implies that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\zeta''\theta) \leq j'' \leq s$. Since for all $i \in \{2, \ldots, n\}$, $\xi_i \in st(\zeta'')$, we can deduce that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi_i\theta) \leq s$.

Condition 2: By hypothesis, there exists a constraint system $\mathcal{C}'$ in the column of $\mathcal{C}$, $(\xi, i \rhd u) \in \Phi(\mathcal{C})$, $(\xi', i' \rhd u') \in \Phi(\mathcal{C}')$, $\mathsf{f} \in \mathcal{F}_c$ such that

— $\mathsf{path}(\xi) = \mathsf{path}(\xi')$, $i \leq s$ and $i' \leq s$

— $ND(\mathcal{C}) \vDash \forall \tilde{x}.u \neq \mathsf{f}(x_1, \ldots, x_n) \vee s \overset{?}{\not\vdash} x_1 \vee \ldots \vee s \overset{?}{\not\vdash} x_n$ where $\tilde{x} = x_1 \ldots x_n$ are variables.

— there exists $X_1, \ldots, X_n \in vars^2(\mathcal{C}')$ such that $\mathsf{C}\lfloor \mathsf{f}(X_1, \ldots, X_n)\Theta' \rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}') = u'$ and such that $\mathsf{param}^{\leq}_{\mathsf{max}}(\mathsf{f}(X_1, \ldots, X_n)\Theta')s$ where $\Theta' = \mathsf{mgu}(Er(\mathcal{C}'))$.

We prove in Condition 1 that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies that for all $(X, k \overset{?}{\vdash} t') \in D(\mathcal{C}')$, $X\theta(\Phi(\mathcal{C})\sigma)\downarrow = X\theta(\Phi(\mathcal{C}')\rho^{-1}\sigma)\downarrow = t'\rho^{-1}\sigma$. Hence, for all $(\zeta, i \overset{?}{\vdash} v') \in \Phi(\mathcal{C}')$, Property 5 of a well formed constraint systems for $\mathcal{C}'$ implies that $\zeta\theta(\Phi(\mathcal{C}')\rho^{-1}\sigma)\downarrow = v'\rho^{-1}\sigma$ with $\mathsf{param}^{\leq}_{\mathsf{max}}(\zeta\theta)i$.

Let $\Theta = \mathsf{mgu}(Er(\mathcal{C}))$ and $\Theta' = \mathsf{mgu}(Er(\mathcal{C}'))$. By hypothesis, we know that $X_1, \ldots, X_n \in vars^2(\mathcal{C}')$, hence thanks to Property 7 of a well formed constraint system, we have that for all $i \in \{1, \ldots, n\}$, $\mathsf{C}\lfloor X_i\Theta' \rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X} \cup \mathcal{X}^2)$ and for all $\xi \in st(X\Theta')$, $\mathsf{path}(\xi)$ exists implies that there exists j and v such that $(\xi, j \rhd v) \in \Phi(\mathcal{C}')$. Hence, $\mathsf{C}\lfloor \mathsf{f}(X_1, \ldots, X_n)\Theta' \rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}') = u'$ implies that $\mathsf{f}(X_1, \ldots, X_n)\Theta'\theta(\Phi(\mathcal{C}')\rho^{-1}\sigma)\downarrow = u'\rho^{-1}\sigma$. Since $\mathsf{f}(X_1, \ldots, X_n)\Theta'\theta(\Phi(\mathcal{C}')\rho^{-1}\sigma)\downarrow = \mathsf{f}(X_1, \ldots, X_n)\Theta'\theta(\Phi(\mathcal{C})\sigma)\downarrow$ and $u\rho = u'$, we can deduce that $\mathsf{f}(X_1, \ldots, X_n)\Theta'\theta(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma$.

Moreover, we know that $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(\mathsf{f}(X_1, \ldots, X_n)\Theta') \leq s$ which implies that for all $Y \in vars^2(\mathsf{f}(X_1, \ldots, X_n)\Theta')$, $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y) \leq s$. But $\mathcal{C}$ and $\mathcal{C}'$ have the same shape and both satisfy the invariant $\mathsf{InvVarConstraint}(s)$. Hence $\mathsf{param}^{\mathcal{C}'}_{\mathsf{max}}(Y) = \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$. Since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, we have that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) \leq s$ implies $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y\theta) \leq s$. We conclude that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\mathsf{f}(X_1, \ldots, X_n)\Theta'\theta) \leq s$.

Hence, we proved that there exists $\xi_1, \ldots, \xi_n \in \Pi_n$ such that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\xi_i) \leq s$ for all $i \in \{1, \ldots, n\}$, and $\mathsf{f}(\xi_1, \ldots, \xi_n)(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma$. But $ND(\mathcal{C}) \vDash \forall \tilde{x}.u \neq \mathsf{f}(x_1, \ldots, x_n) \vee s \overset{?}{\not\vdash} x_1 \vee \ldots \vee s \overset{?}{\not\vdash} x_n$ where $\tilde{x} = x_1 \ldots x_n$ are variables. Moreover, $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies that $\sigma \vDash ND(\mathcal{C})$ which is a contradiction with the fact that $\mathsf{f}(\xi_1, \ldots, \xi_n)(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma$. Hence, $\mathsf{Sol}(\mathcal{C}) = \emptyset$. □

C.5.5 Link between solutions

Lemma 8.7. *Let $(\mathcal{C}, \mathcal{C}')$ be a pair of normalised constraint systems having the same structure and obtained by following the strategy. We denote by Φ and Φ' their associated frame. We denote by S_1, S_1' their associated set of free variables. Let $\text{RULE}(\tilde{p})$ be a transformation rule applicable on $(\mathcal{C}, \mathcal{C}')$. Let $(\mathcal{C}_1, \mathcal{C}_1')$ and $(\mathcal{C}_2, \mathcal{C}_2')$ the two resulting pairs of constraint systems obtained by applying $\text{RULE}(\tilde{p})$ on $(\mathcal{C}, \mathcal{C}')$, and we denote by Φ_1, Φ_1', Φ_2, and Φ_2' their associated frame.*

Let σ, θ and σ' be three substitutions such that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$, and $\Phi\sigma \sim \Phi'\sigma'$. For all substitution θ',

1. *$(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$ if, and only, if $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$.*
2. *Let $i \in \{1, 2\}$, and σ_i be a substitution such that $\sigma|_{S_1} = \sigma_i|_{S_1}$ and $(\sigma_i, \theta') \in \mathsf{Sol}(\mathcal{C}_i)$. Then, $(\sigma_i', \theta') \in \mathsf{Sol}(\mathcal{C}_i')$ for some substitution σ_i' such that $\sigma'|_{S_1'} = \sigma_i'|_{S_1'}$. Moreover, we have that $\mathsf{Init}(\Phi_i)\sigma_i = \mathsf{Init}(\Phi)\sigma$ and $\mathsf{Init}(\Phi_i')\sigma_i' = \mathsf{Init}(\Phi')\sigma'$.*

Proof. Let σ, θ and σ' be three substitutions such that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$, and $\Phi\sigma \sim \Phi'\sigma$. Let θ' be another substitution. We prove the two properties separately. The variation can be proved in a similar way.

1. We assume that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$, and we show that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$. The other implication can be done in a similar way. Let $\mathcal{C} = (S_1; S_2; \Phi; D; Eq; Er; ND; \mathsf{NoUse})$ and $\mathcal{C}' = (S_1'; S_2'; \Phi'; D'; Eq'; Er'; ND'; \mathsf{NoUse}')$. First, since $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$, we have that $\sigma' \models ND' \wedge Eq'$. Second, since $\mathcal{C}$ and $\mathcal{C}'$ have the same structure, we have that $Er = Er'$, and so $\theta' \models Er'$. Moreover, for any $X, i \overset{?}{\vdash} u' \in D'$, we have that $param(X\theta') \subseteq \{ax_1, \ldots, ax_i\}$ and for any ground recipe ξ in Π_n, ξ conforms to $\Phi\theta'$ w.r.t. $\mathsf{NoUse}\theta'$ if, and only if, ξ conforms to $\Phi'\theta'$ w.r.t. $\mathsf{NoUse}'\theta'$. In order to conclude, it remains to show that $(X\theta')\Phi\sigma'{\downarrow} = u'\sigma'{\downarrow}$ for any $(X, i \overset{?}{\vdash} u') \in D'$.

   Since $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ and $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$, we have $(X\theta)\Phi\sigma{\downarrow} = (X\theta')\Phi\sigma{\downarrow}$ for each constraint $(X, i \overset{?}{\vdash} u) \in D$. Since $\Phi\sigma \sim \Phi'\sigma'$, we have that $(X\theta)\Phi'\sigma'{\downarrow} = (X\theta')\Phi'\sigma'{\downarrow}$ for each constraint $(X, i \overset{?}{\vdash} u') \in D'$ (by relying also on the fact that $\mathcal{C}$ and $\mathcal{C}'$ have the same structure). Moreover, since $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$, we have that $(X\theta)\Phi'\sigma'{\downarrow} = u'\sigma'{\downarrow}$ for each constraint $(X, i \overset{?}{\vdash} u') \in D'$. Altogether, this allows us to obtain that $(X\theta')\Phi'\sigma'{\downarrow} = u'\sigma'{\downarrow}$ for each constraint $(X, i \overset{?}{\vdash} u') \in D'$. This allows us to conclude.

2. Let $i \in \{1, 2\}$ and σ_i be a substitution such that $\sigma = \sigma_i|_{vars^1(\mathcal{C})}$ and $(\sigma_i, \theta') \in \mathsf{Sol}(\mathcal{C}_i)$. First, by inspection of the rules, it is easy to see that $\mathsf{Init}(\Phi_i)\sigma_i = \mathsf{Init}(\Phi)\sigma$ and $\mathsf{Init}(\Phi_i')\sigma_i' = \mathsf{Init}(\Phi')\sigma'$. Since $\mathcal{C}_i$ and $\mathcal{C}_i'$ have the same structure, we have that $Er(\mathcal{C}_i) = Er(\mathcal{C}_i')$, and so $\theta' \models Er(\mathcal{C}_i')$. Moreover, for any $X, j \overset{?}{\vdash} u \in D(\mathcal{C}_i')$, we have that $param(X\theta') \subseteq \{ax_1, \ldots, ax_i\}$ and for any ground recipe ξ in Π_n, ξ conforms to $\Phi_i\theta'$ w.r.t. $\mathsf{NoUse}\theta'$ if, and only if, ξ conforms to $\Phi_i'\theta'$ w.r.t. $\mathsf{NoUse}'\theta'$. In order to conclude, it remains to show that there exists a substitution σ_i' such that $(\sigma_i', \theta') \in \mathsf{Sol}(\mathcal{C}_i')$, *i.e.* such that $(X\theta')\Phi_i\sigma_i'{\downarrow} = u'\sigma_i'{\downarrow}$ for any $(X, j \overset{?}{\vdash} u') \in D_i'$ and $\sigma_i' \models ND_i' \wedge Eq_i'$.

   Thanks to Lemma 8.5, we have that $(\sigma, \theta'|_{vars^2(\mathcal{C})}) \in \mathsf{Sol}(\mathcal{C})$. Thanks to *Item 1*, we know that $(\sigma', \theta'|_{vars^2(\mathcal{C})}) \in \mathsf{Sol}(\mathcal{C}')$. Then, we prove the results by case analysis on the rule $\text{RULE}(\tilde{p})$ (we rely on the notation of Figures 7.1 and 7.2) foccusing on the additional constraints that have been added in $\mathcal{C}_i'$.

   Rule CONS, $i = 1$: Since $(\sigma', \theta'|_{vars^2(\mathcal{C})}) \in \mathsf{Sol}(\mathcal{C}')$ and $\theta' \vDash Er_i'$, we have $(X\theta')\Phi'\sigma'{\downarrow} = t'\sigma'$ and $\mathsf{root}(X\theta') = \mathsf{f}$. Thus, we can deduce that $\mathsf{root}(t'\sigma') = \mathsf{f}$. Let $\sigma_i' = \sigma' \cup \{x_1' \mapsto t_1', \ldots, x_n' \mapsto t_n'\}$ where $t'\sigma' = \mathsf{f}(t_1', \ldots, t_n')$ and so $\sigma_i' \vDash t' \overset{?}{=} \mathsf{f}(x_1', \ldots, x_n')$. Moreover, $(\sigma', \theta'|_{vars^2(\mathcal{C})}) \in$

$\mathsf{Sol}(\mathcal{C}')$ implies that $\sigma' \vDash ND' \wedge Eq'$ which means that $\sigma'_i \vDash ND' \wedge Eq' \wedge t' \stackrel{?}{=} \mathsf{f}(x'_1, \ldots, x'_n)$ and so $\sigma'_i \vDash ND'_i \wedge Eq'_i$. At last, since we already know that $X\theta' = \mathsf{f}(X_1\theta', \ldots, X_n\theta')$ and $(X\theta')\Phi'\sigma'\downarrow = t'\sigma' = \mathsf{f}(x'_1\sigma'_i, \ldots, x'_n\sigma'_i)$, we can deduce that $(X_j\theta')(\Phi'\sigma'_i)\downarrow = x'_j\sigma'_i$ for $j = \{1 \ldots n\}$.

Rule Cons, $i = 2$: We have that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$, thus it remains to prove that $\theta' \vDash \mathsf{root}(X) \neq \mathsf{f}$. We know that $\theta' \vDash Er'_i$ which means that $\theta' \vDash \mathsf{root}(X) \neq \mathsf{f}$.

Rule Axiom, $i = 1$: We already know that $\theta' \vDash Er'_i$ thus $\theta' \vDash X \stackrel{?}{=} \xi$. Thus it remains to prove that $\sigma' \vDash u' \stackrel{?}{=} v'$. We know that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$, and thus we have that $X\theta'(\Phi'\sigma')\downarrow = u'\sigma'$ and $\xi\theta'(\Phi'\sigma')\downarrow = v'\sigma'$. Moreover, thanks to Item 1, we have that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$. This implies that $X\theta'(\Phi\sigma)\downarrow = u\sigma$ and $\xi\theta'(\Phi\sigma)\downarrow = v\sigma$. But, we know that $\Phi\sigma \sim \Phi'\sigma'$ and we have that $\sigma \vDash u \stackrel{?}{=} v$. Thus, we can deduce that $\xi\theta'(\Phi'\sigma')\downarrow = X\theta'(\Phi'\sigma')\downarrow$ and so $\sigma' \vDash u' \stackrel{?}{=} v'$.

Rule Axiom, $i = 2$: We already shown that $\theta' \models Er'_i$ and so $\theta' \models X \stackrel{?}{\neq} \xi$. We have nothing else to prove.

Rule Dest, $i = 1$: Since $(\sigma_i, \theta') \in \mathsf{Sol}(\mathcal{C}_i)$, we can deduce that $\mathsf{f}(\xi, X_2, \ldots, X_n)\theta'(\Phi\sigma)\downarrow = w\sigma_i \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Moreover, we know that $\Phi\sigma \sim \Phi'\sigma'$, thus $\mathsf{f}(\xi, X_2, \ldots, X_n)\theta'(\Phi\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ implies $\mathsf{f}(\xi, X_2, \ldots, X_n)\theta'(\Phi'\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ which means that $\xi\theta'(\Phi'\sigma')\downarrow$ can be reduced by the destructor f. Thus, $\mathsf{f}(u'_1, \ldots, u'_n) \to w'$ being a fresh renaming of $\ell \to r$, we can extend σ' into σ'_i such that $u'_1\sigma'_i = v'\sigma'_i$. Moreover, for each rewriting rule, we have that $vars^1(u'_j) \subseteq vars^1(u'_1)$ for $j = 2 \ldots n$, thus $\mathsf{f}(\xi, X_2, \ldots, X_n)\theta'(\Phi'\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ implies that $X_j\theta'(\Phi'_i\sigma'_i)\downarrow = u'_j\sigma'$ for $j = 2 \ldots n$.

Rule Dest, $i = 2$: The non deducibility constraint added in $\mathcal{C}'_2$ corresponds to the fact that for all $(\xi_1, \ldots, \xi_n) \in \Pi_n$ with parameter included in $\{ax_1, \ldots, ax_i\}$, we have $\xi_1\Phi'\sigma'\downarrow \neq \xi\theta\Phi'\sigma'\downarrow \vee f(\xi_1, \ldots, \xi_n)\Phi'\sigma'\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. But, by hypothesis, we know that $(\sigma_2, \theta') \in \mathsf{Sol}(\mathcal{C}_2)$ and $\sigma = \sigma_2|_{vars^1(\mathcal{C})} = \sigma_2$. Thus, $\sigma_2 \vDash ND_2$ and so for all recipes $(\xi_1, \ldots, \xi_n) \in \Pi_n$ with parameter included in $\{ax_1, \ldots, ax_i\}$, we have $\xi_1\Phi\sigma\downarrow \neq \xi\theta\Phi\sigma\downarrow \vee f(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow \notin \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Lastly, since we have that $\Phi\sigma \sim \Phi'\sigma'$, the result holds.

Rule Eq-left-left: We add an equation $u_1 \stackrel{?}{=} u_2$ (resp. a disequation $u_1 \stackrel{?}{\neq} u_2$). Moreover, we have that $u_j\sigma = \xi_j\theta'(\Phi\sigma)\downarrow$ for $j = 1, 2$. Thanks to Item 1, we know that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$ which means that $u'_j\sigma' = \xi_j\theta'(\Phi'\sigma')\downarrow$, for $j = 1, 2$. Since $\Phi\sigma \sim \Phi'\sigma'$, we have that $u_1\sigma = u_2\sigma$ (resp. $u'_1\sigma' \neq u'_2\sigma'$) and this allows us to conclude.
The case of the rule Eq-left-right can be done in a similar way.

Rule Eq-right-right: We know that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$, and thus we have that $X\theta'(\Phi'\sigma')\downarrow = u'\sigma'$ and for all $Y \in vars^2(\xi)$, $Y\theta'(\Phi'\sigma')\downarrow = w'\sigma'$ where $(Y, k \stackrel{?}{\vdash} w') \in D(\mathcal{C}')$. Hence, by construction of v', we deduce that $\xi\theta'(\Phi'\sigma')\downarrow = v'\sigma'$. Moreover, thanks to Item 1, we have that $(\sigma, \theta') \in \mathsf{Sol}(\mathcal{C})$. Similarly, this implies that $X\theta'(\Phi\sigma)\downarrow = u\sigma$ and $\xi\theta'(\Phi\sigma)\downarrow = v\sigma$. We do a case analysis on i.

Case $i = 1$: In such a case, $\theta' \vDash Er(\mathcal{C}'_i)$ and so $\theta' \vDash X \stackrel{?}{=} \xi$. Thus it remains to prove that $\sigma' \vDash u' \stackrel{?}{=} v'$. But, we know that $\Phi\sigma \sim \Phi'\sigma'$ and we have that $\sigma \vDash u \stackrel{?}{=} v$. Thus, we can deduce that $\xi\theta'(\Phi'\sigma')\downarrow = X\theta'(\Phi'\sigma')\downarrow$ and so $\sigma' \vDash u' \stackrel{?}{=} v'$.
Case $i = 2$: In such a case, we only have to prove that $u'\sigma' \neq v'\sigma'$. We know that $u\sigma \neq v\sigma$ which implies that $X\theta'(\Phi\sigma)\downarrow \neq \xi\theta'(\Phi\sigma)\downarrow$. But we have $\Phi\sigma \sim \Phi'\sigma'$ hence $X\theta'(\Phi\sigma)\downarrow \neq \xi\theta'(\Phi\sigma)\downarrow$ implies $X\theta'(\Phi'\sigma')\downarrow \neq \xi\theta'(\Phi'\sigma')\downarrow$ and so $u'\sigma' \neq v'\sigma'$. This allow us to conclude.

Rule Ded-st : Since $(\sigma_i, \theta') \in \mathsf{Sol}(\mathcal{C}_i)$, we know that $\xi\theta'(\Phi\sigma_i)\downarrow = u\sigma_i$ and depending on the value of i, the constraints added on $\mathcal{C}_i$ indicates whether there exists $\xi_1, \ldots, \xi_n \in \Pi_n$ such that $\mathsf{f}(\xi_1, \ldots, \xi_n)\Phi\sigma\downarrow = \xi\theta'(\Phi\sigma)\downarrow$, or not. But once again, since we have $\Phi\sigma \sim \Phi'\sigma'$ by hypothesis, we can transfer this property on σ'. This allows us to conclude. □

C.6 Link between equivalence symbolic and the final test

C.6.1 Preliminaries

Lemma C.44. *Let M be matrix of constraint systems obtained by following the strategy. Let $\mathcal{C}$ and $\mathcal{C}'$ be two constraint systems from the same column in M. Let $(\sigma,\theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$ and $(\sigma',\theta') \in \mathsf{Sol}(\overline{\mathcal{C}'})$ such that $\sigma_{|S_1(\mathcal{C})} = \sigma'_{|S_1(\mathcal{C}')}$. We have that:*

1. $\mathsf{Init}(\Phi(\mathcal{C}))\sigma = \mathsf{Init}(\Phi(\mathcal{C}'))\sigma'$;
2. *for all $(\xi, i \rhd u) \in \Phi(\mathcal{C})$, for all $(\xi', i' \rhd u') \in \Phi(\mathcal{C}')$, if $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ then $u\sigma = u'\sigma'$;*
3. *for all $X \in S_2(\mathcal{C}) = S_2(\mathcal{C}')$, if $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ and $(X, i \overset{?}{\vdash} u') \in D(\mathcal{C}')$, then $u\sigma = u'\sigma'$.*

Proof. Since $\mathcal{C}$ and $\mathcal{C}'$ are both from the same column of M, we deduce that they have at least one common ancestor. Let $\mathcal{C}_0$ be the constraint system on the row matrix of initial constraint system such that $\mathcal{C}_0 \to^* \mathcal{C}$ and $\mathcal{C}_0 \to^* \mathcal{C}'$.

Property 1: Let $u, u' \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$ such that $(ax_i, i \rhd u) \in \Phi(\mathcal{C})$ and $(ax_i, i \rhd u') \in \Phi(\mathcal{C}')$, for some i. Furthermore let $u_0 \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$ such that $(ax_i, i \rhd u_0) \in \Phi(\mathcal{C}_0)$.

Since $\mathcal{C}_0 \to^* \mathcal{C}$ and $\mathcal{C}_0 \to^* \mathcal{C}'$, we obtain from Lemma C.12 that $u = u_0\Sigma$ and $u' = u_0\Sigma'$ where $\Sigma = \mathsf{mgu}(Eq)$, $\Sigma' = \mathsf{mgu}(Eq')$. But $(\sigma,\theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$ implies that $\sigma \vDash Eq(\mathcal{C})$ and so there exists σ_0 such that $\sigma = \Sigma\sigma_0$. Similarly, there exists σ'_0 such that $\sigma' = \Sigma'\sigma'_0$. Moreover, u_0 is a term in the initial constraint system $\mathcal{C}_0$, hence $vars^1(u_0) \subseteq S_1(\mathcal{C}_0) = S_1(\mathcal{C}) = S_1(\mathcal{C}')$ which also implies that $u_0\sigma_{|S_1(\mathcal{C})} = u_0\sigma$ and $u_0\sigma'_{|S_1(\mathcal{C}')} = u_0\sigma'$. At last, by applying the hypothesis $\sigma_{|S_1(\mathcal{C})} = \sigma'_{|S_1(\mathcal{C}')}$, which leads to $u_0\sigma = u_0\sigma'$. Hence, we have that:

$$u\sigma = u_0\Sigma(\Sigma\sigma_0) = u_0\sigma = u_0\sigma' = u\Sigma'(\Sigma'\sigma'_0) = u'\sigma'.$$

Property 2: Let $(\xi, i \rhd u) \in \Phi(\mathcal{C})$ and $(\xi', i' \rhd u') \in \Phi(\mathcal{C}')$. We know that $\mathcal{C}$ and $\mathcal{C}'$ are well-formed constraint systems. Thanks to Property 5 of a well-formed constraint system and the fact that $(\sigma,\theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$ and $(\sigma',\theta') \in \mathsf{Sol}(\overline{\mathcal{C}'})$, we deduce that $\xi\theta(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma$ and $\xi'\theta'(\Phi(\mathcal{C}')\sigma')\downarrow = u'\sigma'$. We have seen that $\mathsf{Init}(\Phi(\mathcal{C})\sigma = \mathsf{Init}(\Phi(\mathcal{C}'))\sigma'$. We have assumed that $\mathsf{path}(\xi) = \mathsf{path}(\xi')$, and we know that $u\sigma, u'\sigma' \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Hence we can apply Lemma C.34 which leads to $\xi\theta(\Phi(\mathcal{C})\sigma)\downarrow = \xi'\theta'(\Phi(\mathcal{C}')\sigma')\downarrow$ and so $u\sigma = u'\sigma'$.

Property 3: Since $X \in S_2(\mathcal{C}) = S_2(\mathcal{C}')$ and $\mathcal{C}_0 \to^* \mathcal{C}$, $\mathcal{C}_0 \to^* \mathcal{C}'$, then there exists $Y \in S_2(\mathcal{C}_0)$ such that $X \in vars^2(\mathsf{C}\lfloor Y\Theta \rfloor_{\Phi(\mathcal{C})})$ where $\Theta = \mathsf{mgu}(Er(\mathcal{C}))$. Let $\Theta' = \mathsf{mgu}(Er(\mathcal{C}'))$, $\Sigma = \mathsf{mgu}(Eq(\mathcal{C}))$ and $\Sigma' = \mathsf{mgu}(Eq(\mathcal{C}')$. By applying Lemma C.11 on Y, we have that $Y\mathsf{acc}^1(\mathcal{C}_0)\Sigma = \mathsf{C}\lfloor Y\Theta \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C})$.

But $(\sigma,\theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$ implies that $\sigma \models Eq(\mathcal{C})$ and so there exists σ_0 such that $\sigma = \Sigma\sigma_0$, and $\mathsf{acc}^1(\mathcal{C})\sigma = \mathsf{acc}^1(\mathcal{C})\sigma_0$ since $\mathcal{C}$ is normalised. Hence, $Y\mathsf{acc}^1(\mathcal{C}_0)\sigma = \mathsf{C}\lfloor Y\Theta \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C})\sigma_0 = \mathsf{C}\lfloor Y\Theta \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C})\sigma$. Similarly, we have that $Y\mathsf{acc}^1(\mathcal{C}_0)\sigma' = \mathsf{C}\lfloor Y\Theta' \rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}')\sigma'$.

Our inductive hypothesis tells us that $\sigma_{|S_1} = \sigma'_{|S_1}$ which implies that $\mathsf{acc}^1(\mathcal{C}_0)\sigma = \mathsf{acc}^1(\mathcal{C}_0)\sigma'$. Furthermore, M satisfies $\mathsf{InvGeneral}$, hence for all $Z \in S_2(\mathcal{C}) = S_2(\mathcal{C}')$, $\mathsf{C}\lfloor Z\Theta \rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor Z\Theta' \rfloor_{\Phi(\mathcal{C}')}$. Since $Y \in S_2(\mathcal{C}_0)$ then $Y \in S_2(\mathcal{C})$ and so we deduce that $\mathsf{C}\lfloor Y\Theta' \rfloor_{\Phi(\mathcal{C}')} = \mathsf{C}\lfloor Y\Theta \rfloor_{\Phi(\mathcal{C})}$. Hence, we deduce that $\mathsf{C}\lfloor Y\Theta \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C})\sigma = \mathsf{C}\lfloor Y\Theta \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C}')\sigma'$. Since $X \in vars^2(\mathsf{C}\lfloor Y\Theta \rfloor_{\Phi(\mathcal{C})})$, we can conclude that $X\mathsf{acc}^1(\mathcal{C})\sigma = X\mathsf{acc}^1(\mathcal{C}')\sigma'$ and so $u\sigma = u'\sigma'$. □

C.6.2 Step e of the strategy

Lemma C.45. *Let M, M_1 be a matrix of constraint system obtained respectively at the beginning and end of step a of phase 1 with support s such that $M \mapsto M_1$. Let $\mathcal{C}_1$ be a constraint system in M_1. Assume that $\mathcal{C}$ the constraint system in M ancestor of $\mathcal{C}_1$ satisfies $\mathsf{Sol}(\mathcal{C}) = \mathsf{Sol}(\overline{\mathcal{C}})$. Let $(\sigma,\theta) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$. If $(\sigma,\theta) \notin \mathsf{Sol}(\mathcal{C}_1)$ then there exists a constraint system $\mathcal{C}'_1$ in the same column of $\mathcal{C}_1$ such that:*

1. $\mathcal{C} \mapsto^* \mathcal{C}'_1$;

2. $\{(\xi, i) \mid i < s \wedge (\xi, i \rhd u) \in \Phi(\mathcal{C}_1)\} = \{(\xi, i) \mid i < s \wedge (\xi, i \rhd u) \in \Phi(\mathcal{C}'_1)\}$;
3. $\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_1)\} \subseteq \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}'_1)\}$;
4. $\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C}_1)\} = \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \mathsf{NoUse}(C'_1)\} \cap \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_1)\}$;
5. *there exists* $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}'_1)$ *such that* $\sigma_{|S_1(\mathcal{C})} = \sigma'_{|S_1(\mathcal{C})}$.

Proof. We prove this result by induction on the size N of the branch $\mathcal{C} \mapsto \mathcal{C}_1$. According to the strategy, every application of the rule Dest or Eq-left-right implies the application of the same rule with the same parameters on each line of the matrix. Hence, for the induction, we assume that the sequence of applications of a rule Dest or Eq-left-right on each line is applied simultaneously.

Base case $N = 0$: In such a case, $\mathcal{C} = \mathcal{C}_1$, hence $\mathsf{Sol}(\mathcal{C}_1) = \mathsf{Sol}(\overline{\mathcal{C}_1})$. By choosing $\mathcal{C}'_1 = \mathcal{C}_1 = \mathcal{C}$, properties 1, 2 and 3 trivially holds. Furthermore, since $\mathsf{NoUse}(\mathcal{C}'_1) \subseteq \Phi(\mathcal{C}'_1)$, property 4 hold. At last, by hypothesis we have $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$ which implies that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_1)$. With $\mathcal{C}_1 = \mathcal{C}'_1$, we conclude that $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}'_1)$. Hence property 5 holds.

Inductive step $N > 0$: Let $R(\tilde{p})$ be the last rule applied. Let M_2 be the matrix such that $M_2 \to M_1$ (note that the rule applied is necessary an internal rule) and let $\mathcal{C}_2$ be the constraint system in M_2 such that $\mathcal{C}_2 \mapsto \mathcal{C}_1$. Thanks to Lemma 8.5, $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$ implies that $(\sigma', \theta') \in \mathsf{Sol}(\overline{\mathcal{C}_2})$ with $\sigma'_{|S_1} = \sigma_{|S_1}$ and $\theta_{|vars^2\mathcal{C}_2} = \theta'$.

Hence by induction hypothesis, we know that there exists a constraint system $\mathcal{C}'_2$ in the same column of $\mathcal{C}_2$ such that:

1. $\mathcal{C} \mapsto^* \mathcal{C}'_2$
2. $\{(\xi, i) \mid i < s \wedge (\xi, i \rhd u) \in \Phi(\mathcal{C}_2)\} = \{(\xi, i) \mid i < s \wedge (\xi, i \rhd u) \in \Phi(\mathcal{C}'_2)\}$
3. $\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_2)\} \subseteq \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}'_2)\}$
4. $\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C}_2)\} = \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \mathsf{NoUse}(C'_2)\} \cap \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_2)\}$
5. there exists $(\sigma'', \theta'') \in \mathsf{Sol}(\mathcal{C}'_2)$ such that $\sigma''_{|S_1} = \sigma'_{|S_1}$.

Thanks to the description of the strategy (see Section 7.4), we know that applying the rule Dest or Eq-left-right is always follows by the application of the same instance of the rule on each line of the matrix. However, when the parameters of the instance of the rule are not compatible with the constraint systems on a line of the matrix, this line stay untouched. Hence, we do a case analysis on the rule applied and on whether the parameters were compatible or not. Note that Propertiy 3 of the inductive hypothesis on $\mathcal{C}_2$ and $\mathcal{C}'_2$ implies that if the parameters of the rule are compatible for $\mathcal{C}_2$, then they also are compatible for $\mathcal{C}'_2$.

— $R(\tilde{p}) = \text{Eq-left-right}(X_0, \xi_0)$ *where X_0, ξ_0 are not compatible parameters for $\mathcal{C}'_2$:* In such a case, it implies that there is no frame element in $\Phi(\mathcal{C}'_2)$ with $\mathsf{path}(\xi_0)$ as path. Hence, $\mathcal{C}'_2$ remains unchanged and so $\mathcal{C}'_2$ is a constraint system in M_1. However, we know that for all $(\xi, s \rhd u) \in \Phi(\mathcal{C}_2)$, there exists $(\xi', s \rhd u') \in \Phi(\mathcal{C}'_2)$ such that $\mathsf{path}(\xi) = \mathsf{path}(\xi')$, thus we can deduce that X_0, ξ_0 are also not compatible parameters for $\mathcal{C}_2$ which means that $\mathcal{C}_1$ is in fact $\mathcal{C}_2$. Hence, by denoting $\mathcal{C}'_1 = \mathcal{C}'_2$, the result holds.

— $R(\tilde{p}) = \text{Dest}(\xi_0, \ell \mapsto r, s)$ *where $\xi_0, \ell \mapsto r, s$ are not compatible for $\mathcal{C}'_2$:* Similarly to the previous case, it implies that there is no frame element in $\Phi(\mathcal{C}'_2)$ with $\mathsf{path}(\xi_0)$ as path. Hence $\mathcal{C}_2 = \mathcal{C}_1$ and by denoting $\mathcal{C}'_1 = \mathcal{C}'_2$, the result holds.

— $R(\tilde{p}) = \text{Eq-left-right}(X_0, \xi_0)$ *where X_0, ξ_0 are compatible parameters for $\mathcal{C}'_2$ but not for $\mathcal{C}_2$:* In such a case, there exists a frame element $(\xi, s \rhd u) \in \Phi(\mathcal{C}'_2)$ such that $\mathsf{path}(\xi_0) = \mathsf{path}(\xi)$. Furthermore, there is not such frame element in $\mathcal{C}_2$. Since the rule is applied on $\mathcal{C}'_2$ and $(\sigma'', \theta'') \in \mathsf{Sol}(\mathcal{C}'_2)$, then by Lemma 8.6, we can deduce that there exists a constraint system $\mathcal{C}'_1$ in M_1 such that:

  — $\mathcal{C}'_2 \mapsto \mathcal{C}'_1$ which implies $\mathcal{C} \to^* \mathcal{C}'_1$ hence property 1 holds.

— there exist $(\sigma''', \theta''') \in \mathsf{Sol}(\mathcal{C}_1')$ such that $\sigma'''_{|S_1(\mathcal{C})} = \sigma''_{|S_1(\mathcal{C})}$ hence property 5 holds.

Hence it remains to proves Properties 2, 3 and 4. But the rule Eq-left-right does not add new frame elements in the frame, thus properties 2 and 3 are trivially satisfied. At last, by the rule Eq-left-right, we may have:

$$\mathsf{NoUse}(\mathcal{C}_1') = \mathsf{NoUse}(\mathcal{C}_2') \cup \{(\xi, s \rhd u)$$

But, by hypothesis X_0, ξ_0 are not compatible parameters for $\mathcal{C}_2$ then it means that there is no frame element $(\zeta, s \rhd v) \in \Phi(\mathcal{C}_2)$ such that $\mathsf{path}(\zeta) = \mathsf{path}(\xi_0) = \mathsf{path}(\xi)$. Hence, we have that:

$$\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C}_1')\} \cap \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_2)\}$$
$$=$$
$$\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C}_2')\} \cap \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_2)\}$$

Therefore, property 4 holds.

— $R(\tilde{p}) = \textsc{Dest}(\xi_0, \ell \to r, s)$ *where* $\xi_0, \ell \to r, s$ *are compatible parameters for* $\mathcal{C}_2'$ *but not for* $\mathcal{C}_2$*:* Properties 1 and 5 are proved similarly to the previous case. It remains to prove Properties 2, 3 and 4. Since $\xi_0, \ell \to r, s$ are compatible parameters for $\mathcal{C}_2'$, there exist $(\xi, i \rhd u) \in \Phi(\mathcal{C}_2')$ such that $\mathsf{path}(\xi) = \xi_0$ and $i \leq s$.

Let $\mathsf{g} \in \mathcal{F}_d$ be the destructor symbol of $\ell \to r$. This instance of the rule Dest may only add a frame element $(\zeta, s \rhd w)$ where $\mathsf{path}(\zeta) = \mathsf{g} \cdot \mathsf{path}(\xi_0)$. Hence Property 2 holds. Furthermore since $\Phi(\mathcal{C}_2') \subseteq \Phi(\mathcal{C}_1')$, Property 3 also holds. At last, since the rule Dest does not add frame in element in NoUse, we have that $\mathsf{NoUse}(\mathcal{C}_1') = \mathsf{NoUse}(\mathcal{C}_2')$ and so Property 4 holds.

— $R(\tilde{p}) = \textsc{Eq-left-right}(X_0, \xi_0)$ *where* X_0, ξ_0 *are compatible parameters for both* $\mathcal{C}_2$ *and* $\mathcal{C}_2'$*:* Thanks to Lemma 8.6, we can deduce that there exists a constraint system $\mathcal{C}_1'$ in M_1 such that:

— $\mathcal{C}_2' \mapsto \mathcal{C}_1'$ which implies $\mathcal{C} \to^* \mathcal{C}_1'$ hence property 1 holds.

— there exist $(\sigma''', \theta''') \in \mathsf{Sol}(\mathcal{C}_1')$ such that $\sigma'''_{|S_1(\mathcal{C})} = \sigma''_{|S_1(\mathcal{C})}$ hence property 5 holds.

Moreover, since the rule Eq-left-right does not add new frame elements, properties 2 and 3 also holds.

Let $u, u' \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$ such that $(X_0, i \overset{?}{\vdash} u) \in D(\mathcal{C}_1)$ and $(X_0, i \overset{?}{\vdash} u') \in D(\mathcal{C}_1')$. Furthermore, let $(\xi, s \rhd v) \in \Phi(\mathcal{C}_1)$ and $(\xi', s \rhd v') \in \Phi(\mathcal{C}_1')$ such that $\mathsf{path}(\xi) = \mathsf{path}(\xi') = \mathsf{path}(\xi_0)$. We already proved that $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, $(\sigma''', \theta''') \in \mathsf{Sol}(\mathcal{C}_1')$ and $\sigma_{|S_1(\mathcal{C})} = \sigma'''_{|S_1(\mathcal{C})}$. Thus, by Lemma C.44, we can deduce that $u\sigma = u'\sigma'''$ and $v\sigma = v'\sigma'''$. Thus, $\sigma \vDash u \overset{?}{=} v$ is equivalent to $\sigma''' \vDash u' \overset{?}{=} v'$. But by the description of the rule, we have that $(\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C}_1)$ is equivalent to $\sigma \vDash u \overset{?}{=} v$; and $(\xi', s \rhd u') \in \mathsf{NoUse}(\mathcal{C}_1')$ is equivalent to $\sigma''' \vDash u' \overset{?}{=} v'$. Hence we conclude that

$$(\xi, s \rhd v) \in \mathsf{NoUse}(\mathcal{C}_1) \quad \text{is equivalent to} \quad (\xi', s \rhd v') \in \mathsf{NoUse}(\mathcal{C}_1')$$

and so Property 4 holds.

— $R(\tilde{p}) = \textsc{Dest}(\xi_0, \ell \to r, s)$ *where* $\xi_0, \ell \to r, s$ *are compatible parameters for* $\mathcal{C}_2'$ *and* $\mathcal{C}_2$*.* Thanks to Lemma 8.6, we can deduce that there exists a constraint system $\mathcal{C}_1'$ in M_1 such that:

— $\mathcal{C}_2' \mapsto \mathcal{C}_1'$ which implies $\mathcal{C} \to^* \mathcal{C}_1'$ hence property 1 holds.

— there exist $(\sigma''', \theta''') \in \mathsf{Sol}(\mathcal{C}_1')$ such that $\sigma'''_{|S_1(\mathcal{C})} = \sigma''_{|S_1(\mathcal{C})}$ hence property 5 holds.

Let $\mathsf{g} \in \mathcal{F}_d$ be the destructor symbol of $\ell \to r$. This instance of the rule Dest may only add a frame element $(\zeta, s \rhd w)$ where $\mathsf{path}(\zeta) = \mathsf{g} \cdot \mathsf{path}(\xi_0)$. Hence Property 2 holds.

Let $(\xi, j \rhd u) \in \Phi(\mathcal{C}_1)$ and $(\xi', j' \rhd u') \in \Phi(\mathcal{C}_1')$ such that $\mathsf{path}(\xi) = \mathsf{path}(\xi') = \mathsf{path}(\xi_0)$. First of all, thanks to Property 2 of our inductive hypothesis, we deduce that $j = j'$.

Assume now that there exists $(\zeta, s \rhd w) \in \Phi(\mathcal{C}_1)$ such that $\mathsf{path}(\zeta) = \mathsf{g} \cdot \mathsf{path}(\xi_0)$. We show that there necessary exists $(\zeta', s \rhd w') \in \Phi(\mathcal{C}'_1)$ such that $\mathsf{path}(\zeta') = \mathsf{g} \cdot \mathsf{path}(\xi_0)$.

Since $\mathsf{path}(\zeta) = \mathsf{g} \cdot \mathsf{path}(\xi_0)$ then there exists $X_2, \ldots, X_n \in \mathcal{X}^2$ such that $\zeta = \mathsf{g}(\xi, X_2, \ldots, X_n)$. Furthermore, thanks to $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, to the definition of a solution of a constraint system and to Property 5 of a well-formed constraint system, we deduce that $\zeta\theta(\Phi(\mathcal{C}_1)\sigma)\downarrow = w\sigma$.

But $(\sigma''', \theta''') \in \mathsf{Sol}(\mathcal{C}'_1)$ and $\sigma_{|S_1(\mathcal{C})} = \sigma'''_{|S_1(\mathcal{C})}$. Thus, by Lemma C.44, we can deduce that $\mathsf{Init}(\Phi(\mathcal{C}_1))\sigma = \mathsf{Init}(\Phi(\mathcal{C}'_1))\sigma'''$ which implies that $\zeta\theta(\Phi(\mathcal{C}_1)\sigma)\downarrow = \zeta\theta(\Phi(\mathcal{C}'_1)\sigma''')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Furthermore, Lemma C.44 also implies that $u\sigma = u'\sigma'''$. Hence we have that $\xi'\theta'''(\Phi(\mathcal{C}'_1)\sigma''')\downarrow = \xi\theta(\Phi(\mathcal{C}_1)\sigma)\downarrow = \xi\theta(\Phi(\mathcal{C}'_1)\sigma''')\downarrow$. Thus, $\mathsf{g}(\xi'\theta''', X_2\theta, \ldots, X_n\theta)(\Phi(\mathcal{C}'_1)\sigma''')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. With $\sigma''' \vDash ND(\mathcal{C}'_1)$ thanks to $(\sigma''', \theta''') \in \mathsf{Sol}(\mathcal{C}'_1)$, the description of the rule Dest allows us to conclude that there exists $(\zeta', s \rhd w') \in \Phi(\mathcal{C}'_1)$ such that $\mathsf{path}(\zeta') = \mathsf{path}(\zeta)$. Hence Property 3 folds.

Since the rule Dest does not add a frame element in NoUse, we conclude that Property 4 holds. □

Lemma C.46. *Let M, M_1 be a matrix of constraint system obtained respectively at the beginning and end of step a of phase 1 with support s such that $M \mapsto M_1$. Let $\mathcal{C}_1$ be a constraint system in M_1. Assume that $\mathcal{C}$ the constraint system in M ancestor of $\mathcal{C}_1$ satisfies $\mathsf{Sol}(\mathcal{C}) = \mathsf{Sol}(\overline{\mathcal{C}})$.*

Let $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$. Assume that

1. *for all $X \in D(\mathcal{C}_1)$, for all position p of $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_1)}$, if $\mathsf{root}(()\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C}_1)} \in \mathcal{F}_d$, then there is no ground recipe $\xi \in \Pi_n$ such that $\xi(\Phi(\mathcal{C}_1)\sigma)\downarrow = X\theta'|_p(\Phi(\mathcal{C}_1)\sigma)\downarrow$ and $\mathsf{param}_{\mathsf{max}}(\xi') < \mathsf{param}_{\mathsf{max}}(X\theta|_p)$; and*
2. *for all $\xi, \xi' \in st(X\theta \mid X \in D(\mathcal{C}_1))$, $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ implies $\xi = \xi'$.*

There exists a constraint system $\mathcal{C}'_1$ in the same column of $\mathcal{C}_1$ and there exists $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}'_1)$ such that $\sigma_{|S_1(\mathcal{C})} = \sigma'_{|S_1\mathcal{C}}$ and for all $X \in S_2(\mathcal{C}_1)$, $X\theta = X\theta'$

Proof. Our hypothesis on M, M_1 and $\mathcal{C}_1$ allows us to apply Lemma C.45. Hence we have that there exists a constraint system $\mathcal{C}'_1$ in the same column as $\mathcal{C}_1$ such that:

1. $\mathcal{C} \mapsto^* \mathcal{C}'_1$;
2. $\{(\xi, i) \mid i \neq s \wedge (\xi, i \rhd u) \in \Phi(\mathcal{C}_1)\} = \{(\xi, i) \mid i < s \wedge (\xi, i \rhd u) \in \Phi(\mathcal{C}'_1)\}$;
3. $\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_1)\} \subseteq \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}'_1)\}$;
4. $\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \mathsf{NoUse}(\mathcal{C}_1)\} = \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \mathsf{NoUse}(C'_1)\} \cap \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_1)\}$;
5. there exists $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}'_1)$ such that $\sigma_{|S_1(\mathcal{C})} = \sigma'_{|S_1(\mathcal{C})}$.

However, Property 5 is not a sufficient for our result. Hence we will build a new substitution θ'' that satisfies the properties stated in the Lemma.

Since $\mathcal{C}_1$ and $\mathcal{C}'_1$ have the same shape, we have that $S_2(\mathcal{C}_1) = S_2(\mathcal{C}'_1)$. Furthermore, since during Step$a$ of phase 1 with support s, the only added deducible constraint are of the following form: $X, s \overset{?}{\vdash} u$, for some X and u where $X \notin S_2(\mathcal{C}_1)$.

We now show that there exists a second order variable renaming ρ such that:

— $\{(X\rho, i) \mid (X, i \overset{?}{\vdash} u) \in D(\mathcal{C}_1)\} \subseteq \{(X, i) \mid (X, i \overset{?}{\vdash} u) \in D(\mathcal{C}'_1)\}$

— $\{(\xi\rho, i) \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}_1)\} \subseteq \{(\xi, i) \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}'_1)\}$

First of all, since $\mathcal{C}_1$ and $\mathcal{C}'_1$ have the same shape, we deduce that $S_2(\mathcal{C}_1) = S_2(\mathcal{C}'_1)$ and $\{(X, i) \mid (X, i \overset{?}{\vdash} u) \in D(\mathcal{C}_1) \wedge X \in S_2(\mathcal{C}_1)\} = \{(X, i) \mid (X, i \overset{?}{\vdash} u) \in D(\mathcal{C}'_1) \wedge X \in S_2(\mathcal{C}'_1)\}$. Hence, we define ρ on $S_2(\mathcal{C})$ is the identity. Let $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}_1)$ such that $X \notin S_2(\mathcal{C}_1)$. Since M_1 satisfies $\mathsf{PP1Sa}(s)$, we deduce that there exists a unique frame element $(\mathsf{g}(\xi_1, \ldots, \xi_n), j \rhd v) \in \Phi(\mathcal{C}_1)$ and $k \in \{2, \ldots, n\}$ such that $j = s$ and $\xi_k = X$. But we already proved that $\{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}_1)\} \subseteq \{\mathsf{path}(\xi) \mid (\xi, s \rhd u) \in \Phi(\mathcal{C}'_1)\}$. Hence along with Property C.2 of the invariant $\mathsf{PP1Sa}(s)$

on $\mathcal{C}_1'$, we deduce that there exists $(\mathsf{g}(\xi_1', X_2', \ldots, X_n'), s \rhd v') \in \Phi(\mathcal{C}_1')$ and $\mathsf{path}(\xi_1') = \mathsf{path}(\xi_1)$. Thus, we define ρ on X such that $X\rho = X_k'$. Moreover, since $X_k' \notin S_2(\mathcal{C})$, we deduce that there exists u_k' such that $(X_k', s \overset{?}{\vdash} u_k') \in D(\mathcal{C}_1')$. Hence we conclude that $\{(X\rho, i) \mid (X, i \overset{?}{\vdash} u) \in D(\mathcal{C}_1)\} \subseteq \{(X, i) \mid (X, i \overset{?}{\vdash} u) \in D(\mathcal{C}_1')\}$.

We already know that $\{(\xi, i) \mid i \neq s \wedge (\xi, i \rhd u) \in \Phi(\mathcal{C}_1)\} = \{(\xi, i) \mid i < s \wedge (\xi, i \rhd u) \in \Phi(\mathcal{C}_1')\}$. Moreover, for all $(\xi, i \rhd u) \in \Phi(\mathcal{C}_1)$, for all $X \in vars^2(\xi)$, $\mathsf{param}_{\mathsf{max}}^{\mathcal{C}_1}(X) \leq i$ (thanks to $\mathcal{C}_1$ being well-formed. But $\mathcal{C}_1$ satisfies $\mathsf{InvUntouched}(s)$. Hence if $i \neq s$ and $X \in vars^2(\xi)$, then $i < s$ and so $X \in S_2(\mathcal{C})$. Thus, if $i \neq s$ then $\xi\rho = \xi$. Let $(\xi, s \rhd u) \in \Phi(\mathcal{C}_1)$. Since $\mathcal{C}_1$ is well-formed (item 3), $\mathsf{param}_{\mathsf{max}}^{\mathcal{C}_1}(\xi)$ exists and so for all $Z \in vars^2(\xi)$, there exists $(Z, j \overset{?}{\vdash} u) \in D(\mathcal{C}_1)$. Thus by construction of ρ, we deduce that there exists u' such that $(\xi\rho, s \rhd u') \in \Phi(\mathcal{C}_1')$ and so $\{(\xi\rho, i) \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}_1)\} \subseteq \{(\xi, i) \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}_1')\}$.

Hence, for all $X \in vars^2 D(\mathcal{C}_1') \cap \mathrm{img}(\rho)$, we define $X\theta'' = X\rho^{-1}\theta$. It remains to define the variables that are not in $\mathrm{img}(\rho)$.

First of all, for all $Y \in vars^2 D(\mathcal{C}_1)$, for all position p, if $\mathsf{root}(\mathsf{C}\lfloor Y\theta_1 \rfloor_{\Phi(\mathcal{C}_1)}|_p) = \mathsf{g} \in \mathcal{F}_d$ then $\mathsf{path}(Y\theta|_p) \in \mathcal{F}_d^* \cdot \mathcal{AX}$. But since $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, then $Y\theta|_p(\Phi(\mathcal{C}_1)\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Moreover, we know that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}_1')$ with $\sigma_{|S_1(\mathcal{C})} = \sigma'_{|S_1(\mathcal{C})}$. By Lemma C.44, we know that $\mathsf{Init}(\Phi(\mathcal{C}_1))\sigma = \mathsf{Init}(\Phi(\mathcal{C}_1'))\sigma'$ and so $Y\theta|_p(\Phi(\mathcal{C}_1')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$.

Furthermore, the matrix M_1 is obtained at the end of step a of phase 1 of the strategy, then $\mathcal{C}_1'$ satisfies the invariant $\mathsf{InvDest}(s)$. Thus, with $\sigma' \vDash ND(\mathcal{C}_1')$, we can deduce that either (a) there exists a frame element $(\xi, s \rhd u) \in \Phi(\mathcal{C}_1')$ such that $\mathsf{path}(\xi) = \mathsf{path}(Y\theta|_p)$ or else (b) there exist a frame element $(\xi', i \rhd v) \in \Phi(\mathcal{C}_1')$ such that $\mathsf{path}(\xi')$ is a suffix of $\mathsf{path}(Y\theta|_p)$ and $(\xi', i \rhd v) \in \mathsf{NoUse}(\mathcal{C}_1')$.

Case (b): Since $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, we deduce that $Y\theta$ conforms to $\Phi(\mathcal{C}_1)$ w.r.t. $\mathsf{NoUse}(\mathcal{C}_1)$. Hence, we deduce that there is no frame element on $\Phi(\mathcal{C}_1)$ which recipe have $\mathsf{path}(\xi')$ as path. Hence, thanks to property 2 of the well formed constraint system, we can also deduce that there exists a position p' of $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi(\mathcal{C}_1)}$ such that $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi(\mathcal{C}_1)}|_p = \mathsf{path}(\xi')$ and $i = s$. But by Property 8 of a well-formed constraint system, $(\xi', s \rhd v) \in \mathsf{NoUse}(\mathcal{C}_1')$ implies that there exists $Z \in vars^2(\mathcal{C}_1')$ such that $\mathsf{C}\lfloor Z\mathsf{mgu}(Er(\mathcal{C}_1')) \rfloor_{\Phi(\mathcal{C}_1')}\mathsf{acc}^1(\mathcal{C}_1') = v$ and $\mathsf{param}_{\mathsf{max}}^{\mathcal{C}_1'}(Z\mathsf{mgu}(Er(\mathcal{C}_1'))) < s$.

Since we proved that $\mathsf{Init}(\Phi(\mathcal{C}_1))\sigma = \mathsf{Init}(\Phi(\mathcal{C}_1'))\sigma'$, then $Z\theta(\Phi(\mathcal{C}_1)\sigma)\downarrow = \xi'\theta(\Phi(\mathcal{C}_1)\sigma)\downarrow = Y\theta|_p(\Phi(\mathcal{C}_1)\sigma)\downarrow$. Hence we have that $Y\theta|_p(\Phi(\mathcal{C}_1)\sigma)\downarrow = Z\theta(\Phi(\mathcal{C}_1)\sigma)\downarrow$ where $\mathsf{param}_{\mathsf{max}}(Z\theta) < s$ which is a contradiction on the hypothesis on θ. Thus we can deduce that this case is impossible.

Case (a): Let's denote $Y\theta|_p = \mathsf{g}(\zeta_1, \ldots, \zeta_n)$. Thanks to M_1 satisfying invariant $\mathsf{PP1Sa}(s)$, we know that there exists $X_2, \ldots, X_n \in vars^2(D(\mathcal{C}_1'))$ such that $\xi = \mathsf{g}(\xi_1, X_2, \ldots, X_n)$ for some ξ_1. Furthermore, we know, by definition of θ, that all recipe in θ with the same path is equal to $Y\theta|_p$. Hence for all $i \in \{2, \ldots, n\}$, we define $X_i\theta'' = \zeta_i$.

Let $\mathcal{M}$ be the set of all the others variables in $D(\mathcal{C}_1')$ not already defined at this stage. For all $X \in \mathcal{M}$, we define $X\theta'' = \mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C}_1')}\mathsf{acc}^2(\mathcal{C}_1')\theta''$ which exists by following the order $<_{\theta'}$ on $\mathcal{M}$.

Typically, the variables in $\mathcal{M}$ represents the variables that could not be instantiate thanks to θ. Thus, since $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}_1')$, we used θ' to defined those variables. The expression $\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C}_1')}\mathsf{acc}^2(\mathcal{C}_1')\theta''$ represents the fact that the context of $X\theta'$ and $X\theta''$ are the same.

Finally, for all $X \in vars^2(\mathcal{C}_1') \setminus vars^2(D(\mathcal{C}_1'))$, we define $X\theta'' = X\mathsf{mgu}(Er(\mathcal{C}_1'))\theta''$

To verify that $(\sigma', \theta'') \in \mathsf{Sol}(\mathcal{C}_1')$, it remains to prove that $\theta'' \vDash Er(\mathcal{C}_1')$. The others propriety are indeed satisfied by construction of θ''. Thanks to M_1 satisfying invariant $\mathsf{PP1Sa}(s)$ (item C.2), we know that the variable in $D(\mathcal{C}_1')$ do not appear in any inequation in $Er(\mathcal{C}_1')$. Furthermore, since by definition of θ'', θ'' satisfies $\mathsf{mgu}(Er(\mathcal{C}_1'))$, we can deduce that $\theta'' \vDash Er(\mathcal{C}_1')$. □

Lemma C.47. *Let M be a matrix of constraint system obtained at the end of step e of phase 1 of the strategy with s as support. Let $\mathcal{C}$ be a constraint system in M. Let $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$. We have that there exists θ' such that $(\sigma, \theta') \in \mathsf{Sol}(\overline{\mathcal{C}})$ and*

1. *for all $X \in D(\mathcal{C})$, for all position p of $\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C})}$, if $\mathsf{root}(()\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C})|_p} \in \mathcal{F}_d$, then there is no ground recipe $\xi \in \Pi_n$ such that $\xi(\Phi(\mathcal{C})\sigma)\downarrow = X\theta'|_p(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $\mathsf{param}_{\mathsf{max}}(\xi) < \mathsf{param}_{\mathsf{max}}(X\theta'|_p)$; and*

2. for all $\xi, \xi' \in st(X\theta \mid X \in D(\mathcal{C}))$, $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ *implies* $\xi = \xi'$.

Proof. We begin by proving the first property of the result: We show that there exists θ' such that $(\sigma, \theta') \in \mathsf{Sol}(\overline{\mathcal{C}})$ and for all $X \in D(\mathcal{C})$, for all position p of $\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C})}$, if $\mathsf{root}(()\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C})|_p} \in \mathcal{F}_d$, then there is no recipe $\xi \in \Pi_n$ such that $\xi(\Phi(\mathcal{C})\sigma)\downarrow = X\theta'|_p(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $\mathsf{param}_{\mathsf{max}}(\xi) < \mathsf{param}_{\mathsf{max}}(X\theta'|_p)$.

We prove this property by induction on the number of positions p which do not satisfy the property. Let's denote this number $m(\theta)$

Base case $m(\theta) = 0$*:* In such a case, the result trivially holds.

Inductive step $m(\theta) > 0$*:* Otherwise, let $X \in vars^2(D(\mathcal{C}))$, a position p and a ground recipe $\xi \in \Pi_n$ such that $\mathsf{root}(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}|_p) \in \mathcal{F}_d$, $\xi(\Phi(\mathcal{C})\sigma)\downarrow = X\theta|_p(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $\mathsf{param}_{\mathsf{max}}(\xi) < \mathsf{param}_{\mathsf{max}}(X\theta|_p)$.

First of all, thanks to Lemma C.36, we know that there exists $\xi' \in \Pi_n$ such that ξ' conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta$, $\xi(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C})\sigma)\downarrow$ and $\mathsf{param}_{\mathsf{max}}(\xi') \leq \mathsf{param}_{\mathsf{max}}(\xi)$.

Secondly, thanks to Property 1 shown in Lemma C.42, we know that there exists a position p' and $\xi'' \in st(\xi')$, such that p' is a prefix of p and $X\theta[\xi']_p(\Phi(\mathcal{C})\sigma)\downarrow = X\theta[\xi'']_{p'}(\Phi(\mathcal{C})\sigma)\downarrow$ and $X\theta[\xi'']_{p'} \in \Pi_n$.

We want to apply Lemma C.41 for the replacement. We know that $\mathcal{C}$ satisfies the invariants $\mathsf{InvVarFrame}(s)$ and $\mathsf{InvUntouched}(s)$. Thus for all $(\zeta, i \rhd u) \in \Phi(\mathcal{C})$, for all $Z \in vars^2(\zeta)$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Z) < i$. But, for all $Z \in vars^2(\mathsf{C}\lfloor X\theta[\xi'']_{p'} \rfloor_{\Phi(\mathcal{C})})$, either $Z \in vars^2(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})})$ else $Z \in vars^2(\mathsf{C}\lfloor \xi'' \rfloor_{\Phi(\mathcal{C})})$. Since $\xi'' \in st(\xi')$ and $\mathsf{param}_{\mathsf{max}}(\xi') \leq \mathsf{param}_{\mathsf{max}}(\xi) < \mathsf{param}_{\mathsf{max}}(X\theta|_p)$, we can deduce, thanks to Lemma C.40, that $\neg(X < Z)$. Hence we can apply Lemma C.41 which gives us that there exists θ' such that (σ, θ') is a pre-solution of $\mathcal{C}$ with $X\theta' = X\theta[\xi'']_{p'}$, $\theta' \vDash \mathsf{mgu}(Er(\mathcal{C}))$ and for all $Y \in vars^2(D) \setminus \{X\}$, $\mathsf{C}\lfloor Y\theta' \rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor Y\theta \rfloor_{\Phi(\mathcal{C})}$.

However, we know that M is obtained at the end of step e of Phase 1 with support s. Hence thanks to Lemma C.31, M satisfies $\mathsf{PP1Sb}(s, \infty)$ and so $\mathcal{C}$ satisfies: for all $(X, i \rhd x) \in D(\mathcal{C})$, for all $(\xi, j \rhd u) \in \Phi(\mathcal{C})$, for all $\mathsf{f} \in \mathcal{F}_c$, $Er \not\vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$ and $Er \not\vDash X \stackrel{?}{\neq} \xi$. Hence, we can deduce that $\theta' \vDash Er(\mathcal{C})$ and so $(\sigma, \theta') \in \mathsf{Sol}(\overline{\mathcal{C}})$.

At last, since we replace a subterm of $X\theta$ by a recipe of strictly smaller maximal parameter and for all $Y \in vars^2(D) \setminus \{X\}$, $\mathsf{C}\lfloor Y\theta' \rfloor_{\Phi} = \mathsf{C}\lfloor Y\theta \rfloor_{\Phi}$, we can deduce that $m(\theta') < m(\theta)$. Hence we conclude by applying our inductive hypothesis.

We now show the second property of the result. First of all, we know that for all $X \in D(\mathcal{C})$, $X\theta$ conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})$. Hence, if there exists $X, Y \in D(\mathcal{C})$, $\xi \in st(X\theta)$ and $\xi' \in st(Y\theta)$ such that $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ and $\xi \neq \xi'$, it implies that $\mathsf{root}(()\xi) \in \mathcal{F}_d$ and there is no frame element $(\zeta, i \rhd u) \in \Phi(\mathcal{C})$ such that $\mathsf{path}(\xi) = \mathsf{path}(\zeta)$ (otherwise it would contradict the conformity of $X\theta$ to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}\theta$). Hence it implies that there exists p (resp. p') position of $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}$ (resp. $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi(\mathcal{C})}$) such that $X\theta|_p = \xi$ (resp. $Y\theta|_p = \xi'$).

Hence, we do our proof by induction on the number of position that do not satisfies the wanted property: Let $\mathcal{M}(\theta)$ be the set defined such that $\mathcal{M}(\theta) = \{\mathsf{path} \mid X, Y \in vars^2(D(\mathcal{C}))$ and $\xi, \xi' \in st(X\theta, Y\theta)$ and $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ and $\xi \neq \xi'\}$.

Base case $\mathcal{M}(\theta) = \emptyset$*:* In such a case, the result trivially holds.

Inductive step $\mathcal{M}(\theta) \neq \emptyset$*:* Let w be a minimal path in term of size in $\mathcal{M}(\theta)$. Let X_0 be a minimal variable in term of $<_\theta$ of all variables X where there exists a position p of $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}$ such that $\mathsf{path}(X\theta|_p) = w$.

Hence we know that there exists p_0 position of $\mathsf{C}\lfloor X_0\theta \rfloor_{\Phi(\mathcal{C})}$ such that $\mathsf{path}(X_0\theta|_{p_0}) = w$. We will replace any recipe that have w as path by $X_0\theta|_{p_0}$ that we denote ξ_0. Hence, we do a new induction on:

$$m(\theta) = \sum_{\mathsf{path}(\xi)=w} \mathsf{nb}_{\mathsf{occ}}(\xi, \{Y\theta \mid Y \in D(\mathcal{C})\}) - \mathsf{nb}_{\mathsf{occ}}(\xi_0, \{Y\theta \mid Y \in D(\mathcal{C})\})$$

Base case $m = 0$*:* In such a case, it implies that any subterm whose path is equal to w, is in fact ξ_0. Hence it contradicts the fact that $w \in \mathcal{M}$.

Inductive case $m > 0$: Otherwise, we have that there exists $Y \in vars^2(D(\mathcal{C}))$ and p position of $\mathsf{C}\lfloor Y\theta \rfloor_{\Phi(\mathcal{C})}$ such that $\mathsf{path}(Y\theta|_p) = w$ but $Y\theta|_p \neq X\theta|_{p_0}$. We want to apply Lemma C.41 hence we have to verify the application conditions of the lemma. Let $\xi = X\theta|_{p_0}$.

— Since ξ is a subterm of $X\theta$ and $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$, we have that ξ conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta$. Furthermore, thanks to the first property we shown in this lemma, we know that $\mathsf{path}(\xi) = \mathsf{path}(Y\theta|_p)$ implies $\mathsf{param}_{\mathsf{max}}(\xi) = \mathsf{param}_{\mathsf{max}}(Y\theta|_p)$. Hence since $Y\theta|_p \in st(Y\theta)$ and $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$, we deduce that $\mathsf{param}_{\mathsf{max}}(\xi) \leq \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y)$.

— Since $\mathsf{path}(\xi) = \mathsf{path}(Y\theta|_p)$, $\xi(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $Y\theta|_p(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$, then by Lemma C.34, we have that $\xi(\Phi(\mathcal{C})\sigma)\downarrow = Y\theta|_p(\Phi(\mathcal{C})\sigma)\downarrow$. Furthermore $\mathsf{path}(\xi) = \mathsf{path}(Y\theta|_p)$ also implies that $\mathsf{C}\lfloor Y\theta[\xi]_p \rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor Y\theta \rfloor_{\Phi(\mathcal{C})}[\mathsf{C}\lfloor \xi \rfloor_{\Phi(\mathcal{C})}]_p$.

— X_0 was chosen as minimal under $<_\theta$, hence for all $Z \in vars^2(\mathsf{C}\lfloor \xi \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^2(\mathcal{C}))$, if we had $Y <_\theta Z$ then it would imply that $Y <_\theta X_0$ since we have that $Z <_\theta X_0$. This is a contradiction, hence we have that $\neg(Y <_\theta Z)$.

Thus by Lemma C.41, we have that there exists θ' such that (σ, θ') is a pre-solution of $\mathcal{C}$ with $Y\theta' = Y\theta[\xi]_p$, $\theta' \vDash \mathsf{mgu}(Er(\mathcal{C}))$ and for all $Z \in vars^2 D \setminus \{Y\}$, we have that $\mathsf{C}\lfloor Z\theta' \rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor Z\theta \rfloor_{\Phi(\mathcal{C})}$.

But with $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$, we trivially have that $\sigma \vDash Eq(\mathcal{C}_1)$. Moreover, since $\mathcal{C}$ is a constraint system obtained from step e, we have shown that for all $Z \in vars^2(D(\mathcal{C}))$, for all $\mathsf{f} \in \mathcal{F}_c$, there is no inequation in $Er(\mathcal{C})$ of the form $Z \stackrel{?}{\neq} \xi$ or $\mathsf{root}(Z) \stackrel{?}{\neq} \mathsf{f}$ where ξ is a recipe of $\Phi(\mathcal{C})$. Hence we have that $\theta' \vDash Er(\mathcal{C})$ and so $(\sigma, \theta') \in \mathsf{Sol}(\overline{\mathcal{C}})$.

By construction, we have that $m(\theta') < m(\theta)$. Furthermore, the construction of θ', i.e. $Y\theta' = Y\theta[\xi]_p$ and for all $Z \in vars^2 D \setminus \{Y\}$, $\mathsf{C}\lfloor Z\theta' \rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor Z\theta \rfloor_{\Phi(\mathcal{C})}$, imply that X_0 is also a minimal variable in term of $<_{\theta'}$ from all variables $X \in vars^2(D(\mathcal{C}))$ where there exists a position p of $\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C})}$ such that $\mathsf{path}(X\theta'|_p) = w$. Hence we can apply our inductive hypothesis on θ' which conclude the result. □

Definition C.1. *We define a constructor layer equivalence relation between ground recipe of Π_n, denoted $\mathcal{R}_{\mathcal{F}_c}$, as follows: Let $\xi, \xi' \in \Pi_n$ two ground recipes, $\xi \mathcal{R}_{\mathcal{F}_c} \xi'$ if:*

— *$\mathsf{path}(\xi)$, $\mathsf{path}(\xi')$ exist and $\mathsf{path}(\xi) = \mathsf{path}(\xi')$; or*

— *$\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$, $\xi' = \mathsf{g}(\xi'_1, \ldots, \xi'_n)$ for some $\mathsf{g}, \mathsf{f} \in \mathcal{F}_c$, $\mathsf{g} = \mathsf{f}$ and for all $I \in \{1, \ldots, n\}$, $\xi_i \mathcal{R}_{\mathcal{F}_c} \xi'_i$.*

Lemma C.48. *Let Φ a ground frame. Let ξ and ξ' two ground recipe. $\mathsf{C}\lfloor \xi \rfloor_\Phi = \mathsf{C}\lfloor \xi' \rfloor_\Phi$ implies $\xi \mathcal{R}_{\mathcal{F}_c} \xi'$.*

Proof. We prove the result by induction on $|\mathsf{C}\lfloor \xi \rfloor_\Phi|$ but we will also prove in the same time that if $\mathsf{C}\lfloor \xi \rfloor_\Phi = \mathsf{C}\lfloor \xi' \rfloor_\Phi$, and $\mathsf{path}(\xi)$, $\mathsf{path}(\xi')$ exist then $\mathsf{path}(\xi') = \mathsf{path}(\xi)$.

Base case $|\mathsf{C}\lfloor \xi \rfloor_\Phi| = 1$: In such a case, there exists $(\zeta, i \rhd u) \in \Phi$ such that $\mathsf{path}(\zeta) = \mathsf{path}(\xi)$. Since $\mathsf{C}\lfloor \xi' \rfloor_\Phi = \mathsf{C}\lfloor \xi \rfloor_\Phi$, we deduce that $\mathsf{path}(\xi') = \mathsf{path}(\xi)$ and so $\xi \mathcal{R}_{\mathcal{F}_c} \xi'$.

Inductive step $|\mathsf{C}\lfloor \xi \rfloor_\Phi| > 1$: By definition of a context, it implies that $\xi = \mathsf{f}(\xi_1, \ldots, \xi_n)$ and $\mathsf{C}\lfloor \xi \rfloor_\Phi = \mathsf{f}(\mathsf{C}\lfloor \xi_1 \rfloor_\Phi, \ldots, \mathsf{C}\lfloor \xi_n \rfloor_\Phi)$. Similarly, $\xi' = \mathsf{g}(\xi'_1, \ldots, \xi'_n)$ and $\mathsf{C}\lfloor \xi' \rfloor_\Phi = \mathsf{g}(\mathsf{C}\lfloor \xi'_1 \rfloor_\Phi, \ldots, \mathsf{C}\lfloor \xi'_n \rfloor_\Phi)$. Since $\mathsf{C}\lfloor \xi \rfloor_\Phi = \mathsf{C}\lfloor \xi' \rfloor_\Phi$, we deduce that $\mathsf{g} = \mathsf{f}$ and for all $i \in \{1, \ldots, n\}$, $\mathsf{C}\lfloor \xi_i \rfloor_\Phi = \mathsf{C}\lfloor \xi'_i \rfloor_\Phi$. If $\mathsf{f} \in \mathcal{F}_c$ then by induction on ξ_i and ξ'_i, we deduce that for all $i \in \{1, \ldots, n\}$, $\xi_i \mathcal{R}_{\mathcal{F}_c} \xi'_i$. Along with $\mathsf{f} = \mathsf{g}$, we conclude that $\xi \mathcal{R}_{\mathcal{F}_c} \xi'$.

Else we have $\mathsf{f} = \mathsf{g} \in \mathcal{F}_d$. But ξ and ξ' are recipe in Π_n. Hence it implies that $\mathsf{root}(\xi_1) \notin \mathcal{F}_c$ and $\mathsf{root}(\xi'_1) \notin \mathcal{F}_c$. Hence $\mathsf{path}(\xi)$ and $\mathsf{path}(\xi')$ exists and $\mathsf{path}(\xi) = \mathsf{f} \cdot \mathsf{path}(\xi_1)$, $\mathsf{path}(\xi') = \mathsf{f} \cdot \mathsf{path}(\xi'_1)$. But by inductive hypothesis on ξ_1, ξ'_1, we deduce that $\mathsf{path}(\xi_1) = \mathsf{path}(\xi'_1)$. Hence we conclude that $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ and so $\xi \mathcal{R}_{\mathcal{F}_c} \xi'$. □

Lemma C.49. *Let M be a matrix of constraint system obtained during step b to d of phase 1 of the strategy. Let M' be the father of M. Let $\mathcal{C}_1$ a constraint system in M and $\mathcal{C}'_1$ be its father in M'. Let $\mathcal{C}'_2$ be a constraint system in the column of $\mathcal{C}'_1$ in M'. Let $(\sigma'_1, \theta'_1) \in \mathsf{Sol}(\overline{\mathcal{C}'_1})$, $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\mathcal{C}'_2)$ and $(\sigma_1, \theta_1) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$. If*

1. $\sigma_1|_{vars^1(\mathcal{C}'_1)} = \sigma'_1$, $\theta_1|_{vars^2(\mathcal{C}'_1)} = \theta'_1$
2. $\sigma'_1|_{S_1(\mathcal{C}'_1)} = \sigma'_2|_{S_1(\mathcal{C}'_2)}$
3. *for all* $X \in S_2(\mathcal{C}'_1)$, $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$
4. *for all* $X, Y \in S_2(\mathcal{C}'_2)$, *for all* $p \in \mathcal{P}os(\mathsf{C}\lfloor X\theta'_2\rfloor_{\Phi(\mathcal{C}'_2)})$, *for all* $p \in \mathcal{P}os(\mathsf{C}\lfloor Y\theta'_2\rfloor_{\Phi(\mathcal{C}'_2)})$, *if* $\mathsf{path}(X\theta'_2|_p) = \mathsf{path}(X\theta'_2|_{p'})$ *then* $X\theta'_2|_p = X\theta'_2|_{p'}$.

then we have that there a constraint system $\mathcal{C}_2$ *in the column of* $\mathcal{C}_1$ *in* M *and* $(\sigma_2, \theta_2) \in \mathsf{Sol}(\mathcal{C}_2)$ *such that*

1. $\mathcal{C}'_2 \to \mathcal{C}_2$
2. $\sigma_2|_{vars^1(\mathcal{C}'_2)} = \sigma'_2$, $\theta_2|_{vars^2(\mathcal{C}'_2)} = \theta'_2$
3. *for all* $X \in S_2(\mathcal{C}_1)$, $X\theta_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta_2$
4. *for all* $X, Y \in S_2(\mathcal{C}_2)$, *for all* $p \in \mathcal{P}os(\mathsf{C}\lfloor X\theta_2\rfloor_{\Phi(\mathcal{C}'_2)})$, *for all* $p \in \mathcal{P}os(\mathsf{C}\lfloor Y\theta_2\rfloor_{\Phi(\mathcal{C}_2)})$, *if* $\mathsf{path}(X\theta_2|_p) = \mathsf{path}(X\theta_2|_{p'})$ *then* $X\theta_2|_p = X\theta_2|_{p'}$.

Proof. Since M' is the father of M, we do a case analysis on the rule applied on M'.

Case of internal rule not applied on $\mathcal{C}'_2$*:* In such a case, it implies that $\mathcal{C}'_2$ is also a constraint system in the column of $\mathcal{C}_1$ in the matrix M. Hence, by denoting $\mathcal{C}_2 = \mathcal{C}'_2$, and $(\sigma_2, \theta_2) = (\sigma'_2, \theta'_2)$, we trivially have that the two first wanted properties. Hence, it remains to show that $X\theta_1 = X\theta_2$ for all $X \in S_2(\mathcal{C}_1)$. But since $\mathcal{C}_2$ is in M, we know that $\mathcal{C}_2$ and $\mathcal{C}_1$ have the same structure and so $S_2(\mathcal{C}_1) = S_2(\mathcal{C}_2)$. Similarly, we have $S_2(\mathcal{C}'_1) = S_2(\mathcal{C}'_2)$. At last, the rule applied is an internal rule hence we have $S_2(\mathcal{C}_1) = S_2(\mathcal{C}'_1)$. Thus, $\theta_1|_{vars^2(\mathcal{C}'_1)} = \theta'_1$ implies that for all $X \in S_2(\mathcal{C}_1)$, $X\theta_1 = X\theta'_1$. Hence with $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$ and $\theta_2 = \theta'_2$, then we deduce that $X\theta_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta_2$. Moreover, $\theta_2 = \theta'_2$ and hypothesis 4 implies property 4.

Case of internal rule applied on $\mathcal{C}'_2$*:* In such a case, it implies that $\mathcal{C}_1 = \mathcal{C}'_1$ and so $(\sigma_1, \theta_1) = (\sigma'_1, \theta'_1)$. Let $\mathcal{C}_3$ and $\mathcal{C}_4$ be the two constraint system obtained by application of $R(\tilde{p})$ on $\mathcal{C}'_2$. By the definition of an internal rule, we know that both $\mathcal{C}_3$ and $\mathcal{C}_4$ are in the column of $\mathcal{C}_1$ in the matrix M.

Thanks to Lemma 8.6, $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\mathcal{C}'_2)$ implies that there exists $i \in \{3, 4\}$ and $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C}_i)$ such that $\sigma|_{S_1(\mathcal{C}_i)} = \sigma'_2|_{S_1(\mathcal{C}'_2)}$.

Furthermore, the only possible rule applicable in this case are Axiom, Cons, Eq-left-left, Eq-right-right and Ded-st. Since the strategy dictates that Eq-right-right can only be applied internally when ξ is a variable with $\mathsf{param}_{\mathsf{max}}(\xi) < s$ then, by following the proof of Lemma 8.6, we deduce for all $X \in S_2(\mathcal{C}'_2)$, $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}_i)} = \mathsf{C}\lfloor X\theta'_2\rfloor_{\Phi(\mathcal{C}_i)}$. But thanks to Lemma C.48, we deduce that $X\theta \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$.

Let p, p' positions of $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}_i)}$, $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi(\mathcal{C}_i)}$ respectively where $X, Y \in S_2(\mathcal{C})$. Assume that $\mathsf{path}(X\theta|_p) = \mathsf{path}(Y\theta|_{p'})$. But $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}_i)} = \mathsf{C}\lfloor X\theta'_2\rfloor_{\Phi(\mathcal{C}'_1)}$ and $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi(\mathcal{C}_i)} = \mathsf{C}\lfloor Y\theta'_2\rfloor_{\Phi(\mathcal{C}'_1)}$. Hence $\mathsf{path}(X\theta|_p) = \mathsf{path}(Y\theta|_{p'})$ implies that $\mathsf{path}(X\theta'_2|_p) = \mathsf{path}(Y\theta'_2|_{p'})$. By hypothesis 4, we deduce that $X\theta'_2|_p = Y\theta'_2|_{p'}$. Hence, thanks to $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}_i)} = \mathsf{C}\lfloor X\theta'_2\rfloor_{\Phi(\mathcal{C}'_1)}$, $\mathsf{C}\lfloor Y\theta\rfloor_{\Phi(\mathcal{C}_i)} = \mathsf{C}\lfloor Y\theta'_2\rfloor_{\Phi(\mathcal{C}'_1)}$ we deduce that $\mathsf{C}\lfloor X\theta|_p\rfloor_{\Phi(\mathcal{C}_i)} = \mathsf{C}\lfloor Y\theta|_p\rfloor_{\Phi(\mathcal{C}_i)}$. Since $X\theta, Y\theta$ conforms to $\Phi(\mathcal{C}_i)\theta$ w.r.t. $\mathsf{NoUse}\theta$, we deduce that $X\theta|_p = Y\theta|_p$.

Moreover, since $X\theta$ (resp. $X\theta'_2$) conforms with $\Phi(\mathcal{C}'_1\theta)$ (resp. $\Phi(\mathcal{C}_i\theta'_2)$) and $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}_i)} = \mathsf{C}\lfloor X\theta'_2\rfloor_{\Phi(\mathcal{C}_i)}$ Since $S_2(\mathcal{C}'_2) = S_2(\mathcal{C}_i)$ and by hypothesis, $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$, we have that $X\theta_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta$. Hence, the result holds by denoting $\mathcal{C}_2 = \mathcal{C}_i$ and $(\sigma_2, \theta_2) = (\sigma, \theta)$.

Case of external rule: In such a case, the rule $R(\tilde{p})$ is applied on both $\mathcal{C}'_1$ and $\mathcal{C}'_2$. The only possible external rules are Cons, Axiom and Eq-right-right. By definition of an external application of the rule, we also know that if $\mathcal{C}$ is the right (resp. left) son of $\mathcal{C}_1$ then there exists a constraint system $\mathcal{C}'$ in M such that $\mathcal{C}'$ is the right (resp. left) son of $\mathcal{C}'_1$. We do a case analysis on the rule applied.

— Rule $\textsc{Cons}(X, \mathsf{f})$, left son: In such a case, $Er(\mathcal{C}_1) = Er(\mathcal{C}'_1) \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$, $Er(\mathcal{C}_2) = Er(\mathcal{C}'_2) \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$; and $Eq(\mathcal{C}_1) = Eq(\mathcal{C}'_1) \wedge X\mathsf{acc}^1(\mathcal{C}'_1) \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n)$, $Eq(\mathcal{C}_2) = Eq(\mathcal{C}'_2) \wedge X\mathsf{acc}^1(\mathcal{C}'_2) \stackrel{?}{=} \mathsf{f}(y_1, \ldots, y_n)$ where X_i, x_i, y_i are fresh variables for all $i \in \{1, \ldots, n\}$ in $S_2(\mathcal{C}_1) = S_2(\mathcal{C}_2)$. Moreover, $X \in S_2(\mathcal{C}'_1) = S_2(\mathcal{C}'_2)$.

By hypothesis, we know that $(\sigma_1, \theta_1) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, $\theta_1|_{vars^2(\mathcal{C}'_1)}$ and $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{C'_2})$. Hence $\theta_1 \vDash Er(\mathcal{C}_1)$ implies that $\mathsf{root}(X\theta'_1) = \mathsf{f} \in \mathcal{F}_c$. But $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$ thus we deduce that $\mathsf{root}(X\theta'_2) = \mathsf{f}$. We define $\theta_2 = \theta'_2 \cup \{X_1 \mapsto X\theta'_2|_1; \dots X_n \mapsto X\theta'_2|_n\}$. We show that for all $i \in \{1, \dots, n\}$, $X_i\theta_1 \ \mathcal{R}_{\mathcal{F}_c} \ X_i\theta_2$. $\mathsf{root}(X\theta'_2) = \mathsf{root}(X\theta'_1) =\in \mathcal{F}_c$ and $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$ implies by definition of $\mathcal{R}_{\mathcal{F}_c}$ that $X\theta'_1 = \mathsf{f}(\xi_1, \dots, \xi_n)$ and $X\theta'_2 = \mathsf{f}(\zeta_1, \dots, \zeta_n)$ for some $\xi_1, \dots, \xi_n, \zeta_1, \dots, \zeta_n$, and for all $i \in \{1, \dots, n\}$, $\xi_i \ \mathcal{R}_{\mathcal{F}_c} \ \zeta_i$. Since for all $i \in \{1, \dots, n\}$, $X_i\theta_1 = \xi_i$ and $X_i\theta_2 = \zeta_i$, the result holds.

Moreover, since for all $i \in \{1, \dots, n\}$, $\mathsf{C}\lfloor\zeta_i\rfloor_{\Phi(\mathcal{C}_2)}$ is a subterm of $\mathsf{C}\lfloor X\theta'_2\rfloor_{\Phi(\mathcal{C}_2)}$, then hypothesis 4 implies property 4.

It remains to build σ_2. Since $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\mathcal{C}'_2)$, we know that $X\theta'_2(\Phi(\mathcal{C}'_2)\sigma'_2)\downarrow = \mathsf{f}(u_1, \dots, u_n)$ where for all $i \in \{1, \dots, n\}$, $\zeta_i(\Phi(\mathcal{C}'_2)\sigma'_2)\downarrow = u_i$. Since $y_1, \dots, y_n$ are fresh variables, we define $\sigma_2 = \sigma'_2 \cup \{y_1 \mapsto u_1, \dots, y_n \mapsto u_n\}$. Hence $\sigma_2 \vDash X\mathsf{acc}^1(\mathcal{C}'_2) \stackrel{?}{=} \mathsf{f}(y_1, \dots, y_n)$. Hence we conclude that $(\sigma_2, \theta_2) \in \mathsf{Sol}(\overline{\mathcal{C}_2})$.

— Rule Cons(X, f), right son: In such a case, $Er(\mathcal{C}_1) = Er(\mathcal{C}'_1) \wedge \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}_2) = Er(\mathcal{C}'_2) \wedge \mathsf{root}(X) \neq \mathsf{f}$. By hypothesis, we know that $(\sigma_1, \theta_1) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, $\theta_1|_{vars^2(\mathcal{C}'_1)}$ and $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{C'_2})$. Hence $\theta'_1 \vDash Er(\mathcal{C}_1)$ implies that $\mathsf{root}(X\theta'_1) \neq \mathsf{f} \in \mathcal{F}_c$. But $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$ thus we deduce that $\mathsf{root}(X\theta'_2) \neq \mathsf{f}$. Hence the result holds with $(\sigma_2, \theta_2) = (\sigma'_2, \theta'_2)$. Moreover, hypothesis 4 trivially implies property 4.

— Rule Axiom(X, path), left son: $Er(\mathcal{C}_1) = Er(\mathcal{C}'_1) \wedge X \stackrel{?}{=} \xi_1$, $Er(\mathcal{C}_2) = Er(\mathcal{C}'_2) \wedge X \stackrel{?}{=} \xi_2$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}'_1) \wedge X\mathsf{acc}^1(\mathcal{C}'_1) \stackrel{?}{=} \mathsf{path}(\xi_1)\mathsf{acc}^1(\mathcal{C}'_1)$ and $Eq(\mathcal{C}_2) = Eq(\mathcal{C}'_2) \wedge X\mathsf{acc}^1(\mathcal{C}'_2) \stackrel{?}{=} \mathsf{path}(\xi_2)\mathsf{acc}^1(\mathcal{C}'_2)$ where $\mathsf{path}(\xi_1) = \mathsf{path}(\xi_2)$.

By hypothesis, we know that $(\sigma_1, \theta_1) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, $\theta_1|_{vars^2(\mathcal{C}'_1)}$ and $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{C'_2})$. Hence $\theta_1 \vDash Er(\mathcal{C}_1)$ implies $X\theta'_1 = \xi_1\theta'_1$. But $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$ hence since $\mathsf{path}(X\theta'_1$ exists, we deduce that $\mathsf{path}(X\theta'_2)$ exists and $\mathsf{path}(X\theta'_2) = \mathsf{path}(X\theta'_1)$. Moreover, $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{\mathcal{C}'_2})$ also implies that $X\theta'_2$ conforms with $\Phi(\mathcal{C}'_2)\theta'_2$ w.r.t. $\mathsf{NoUse}\theta'_2$. Hence since ξ_2 is a recipe of a frame element of $\Phi(\mathcal{C}'_2)$ such that $\mathsf{path}(\xi_2) = \mathsf{path}(\xi_1) = \mathsf{path}(X\theta'_2)$, we conclude that $X\theta'_2 = \xi_2\theta'_2$. Hence $\theta'_2 \vDash Er(\mathcal{C}_2)$. Moreover, hypothesis 4 trivially implies property 4.

Since $(\sigma'_1, \theta'_1) \in \mathsf{Sol}(\overline{\mathcal{C}'_1})$, $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{\mathcal{C}'_2})$ and $\sigma'_1|_{S_1(\mathcal{C}'_1)} = \sigma'_2|_{S_1\mathcal{C}'_2}$ then by Lemma C.44, we have that $X\mathsf{acc}^1(\mathcal{C}'_2)\sigma'_2 = X\mathsf{acc}^1(\mathcal{C}'_1)\sigma'_1$ and $\mathsf{path}(\xi_1)\mathsf{acc}^1(\mathcal{C}'_1)\sigma'_1 = \mathsf{path}(\xi_2)\mathsf{acc}^1(\mathcal{C}'_2)\sigma'_2$. Hence, σ'_1 satisfies $X\mathsf{acc}^1(\mathcal{C}'_1) \stackrel{?}{=} Y\mathsf{acc}^1(\mathcal{C}'_1)$ implies that $\sigma'_2 \vDash X\mathsf{acc}^1(\mathcal{C}'_2) \stackrel{?}{=} \mathsf{path}(\xi_2)\mathsf{acc}^1(\mathcal{C}'_2)$. Hence $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\mathcal{C}_2)$.

— Rule Axiom(X, path), right son: $Er(\mathcal{C}_1) = Er(\mathcal{C}'_1) \wedge X \stackrel{?}{\neq} \xi_1$, $Er(\mathcal{C}_2) = Er(\mathcal{C}'_2) \wedge X \stackrel{?}{\neq} \xi_2$. By hypothesis, we know that $(\sigma_1, \theta_1) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, $\theta_1|_{vars^2(\mathcal{C}'_1)}$ and $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{\mathcal{C}'_2})$. Hence $\theta_1 \vDash Er(\mathcal{C}_1)$ implies $X\theta'_1 \stackrel{?}{\neq} \xi_1\theta'_1$. Since $X\theta'_1$ conforms with $\Phi(\mathcal{C}'_1)\theta'_1$ w.r.t. $\mathsf{NoUse}(\mathcal{C}'_1)\theta'_1$, it also implies that $\mathsf{path}(X\theta'_1) \stackrel{?}{\neq} \mathsf{path}(\xi_1)$. But $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$ hence $\mathsf{path}(X\theta'_2) = \mathsf{path}(X\theta'_1)$ and so $\mathsf{path}(X\theta'_2) \neq \mathsf{path}(\xi_2)$. Thus, we deduce that $X\theta'_2 \neq \xi_2\theta'_2$ and so the result holds with $(\sigma_2, \theta_2) = (\sigma'_2, \theta'_2)$.

— Rule Eq-right-right(X, ξ), left son: $Er(\mathcal{C}_1) = Er(\mathcal{C}'_1) \wedge X \stackrel{?}{=} Y$, $Er(\mathcal{C}_2) = Er(\mathcal{C}'_2) \wedge X \stackrel{?}{=} Y$, $Eq(\mathcal{C}_1) = Eq(\mathcal{C}'_1) \wedge X\mathsf{acc}^1(\mathcal{C}'_1) \stackrel{?}{=} Y\mathsf{acc}^1(\mathcal{C}'_1)$ and $Eq(\mathcal{C}_2) = Eq(\mathcal{C}'_2) \wedge X\mathsf{acc}^1(\mathcal{C}'_2) \stackrel{?}{=} Y\mathsf{acc}^1(\mathcal{C}'_2)$. By hypothesis, we know that $(\sigma_1, \theta_1) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$, $\theta_1|_{vars^2(\mathcal{C}'_1)}$ and $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{\mathcal{C}'_2})$. Hence $\theta_1 \vDash Er(\mathcal{C}_1)$ implies $X\theta'_1 = Y\theta'_1$. But $X\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ X\theta'_2$ and $Y\theta'_1 \ \mathcal{R}_{\mathcal{F}_c} \ Y\theta'_2$. Hence, we deduce that $X\theta'_2 \ \mathcal{R}_{\mathcal{F}_c} \ Y\theta'_2$. Moreover, thanks to hypothesis 4, $X\theta'_2 \ \mathcal{R}_{\mathcal{F}_c} \ Y\theta'_2$ implies that $X\theta'_2 = Y\theta'_2$ and so $\theta'_2 \vDash Er(\mathcal{C}_2)$.

Since $(\sigma'_1, \theta'_1) \in \mathsf{Sol}(\overline{\mathcal{C}'_1})$, $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{\mathcal{C}'_2})$ and $\sigma'_1|_{S_1(\mathcal{C}'_1)} = \sigma'_2|_{S_1\mathcal{C}'_2}$ then by Lemma C.44, we have that $X\mathsf{acc}^1(\mathcal{C}'_2)\sigma'_2 = X\mathsf{acc}^1(\mathcal{C}'_1)\sigma'_1$ and $Y\mathsf{acc}^1(\mathcal{C}'_1)\sigma'_1 = Y\mathsf{acc}^1(\mathcal{C}'_2)\sigma'_2$. Hence, σ'_1 satisfies $X\mathsf{acc}^1(\mathcal{C}'_1) \stackrel{?}{=} Y\mathsf{acc}^1(\mathcal{C}'_1)$ implies that $\sigma'_2 \vDash X\mathsf{acc}^1(\mathcal{C}'_2) \stackrel{?}{=} Y\mathsf{acc}^1(\mathcal{C}'_2)$. Hence $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\mathcal{C}_2)$. Thus the result holds with $(\sigma_2, \theta_2) = (\sigma'_2, \theta'_2)$.

— Rule Eq-right-right(X, ξ), right son: $Eq(\mathcal{C}_1) = Eq(\mathcal{C}'_1) \wedge X\mathsf{acc}^1(\mathcal{C}'_1) \stackrel{?}{\neq} Y\mathsf{acc}^1(\mathcal{C}'_1)$ and $Eq(\mathcal{C}_2) = Eq(\mathcal{C}'_2) \wedge X\mathsf{acc}^1(\mathcal{C}'_2) \stackrel{?}{=} Y\mathsf{acc}^1(\mathcal{C}'_2)$. Since $(\sigma'_1, \theta'_1) \in \mathsf{Sol}(\overline{\mathcal{C}'_1})$, $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\overline{\mathcal{C}'_2})$ and $\sigma'_1|_{S_1(\mathcal{C}'_1)} = \sigma'_2|_{S_1\mathcal{C}'_2}$ then by Lemma C.44, we have that $X\mathsf{acc}^1(\mathcal{C}'_2)\sigma'_2 = X\mathsf{acc}^1(\mathcal{C}'_1)\sigma'_1$ and $Y\mathsf{acc}^1(\mathcal{C}'_1)\sigma'_1 = Y\mathsf{acc}^1(\mathcal{C}'_2)\sigma'_2$. Hence, σ'_1 satisfies $X\mathsf{acc}^1(\mathcal{C}'_1) \stackrel{?}{\neq} Y\mathsf{acc}^1(\mathcal{C}'_1)$ implies that $\sigma'_2 \vDash X\mathsf{acc}^1(\mathcal{C}'_2) \stackrel{?}{\neq} Y\mathsf{acc}^1(\mathcal{C}'_2)$. Hence $(\sigma'_2, \theta'_2) \in \mathsf{Sol}(\mathcal{C}_2)$. Thus the result holds with $(\sigma_2, \theta_2) = (\sigma'_2, \theta'_2)$. □

Lemma C.50. *Let M be a matrix of constraint system obtained by at step e of phase 1 of the strategy with any support. Let $\mathcal{C}$ be a constraint system in M. We have that $\mathsf{Sol}(\overline{\mathcal{C}}) = \mathsf{Sol}(\mathcal{C})$*

Proof. We prove this result by induction of the support s of the step e. For the purpose of the induction, we assume that step e of phase 1 with support 0 corresponds to the root.

Base case $s = 0$: In such a case, we know that M is a row matrix of initial constraint system. Hence, we trivially have that $\mathsf{Sol}(\mathcal{C}) = \mathsf{Sol}(\overline{\mathcal{C}})$ and so the result holds.

Inductive step $s > 0$: Let $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$. Thanks to Lemma C.47, we know that there exists θ' such that $(\sigma, \theta') \in \mathsf{Sol}(\overline{\mathcal{C}})$ and

1. for all $X \in D(\mathcal{C})$, for all position p of $\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C})}$, if $\mathsf{root}(\mathsf{C}\lfloor X\theta' \rfloor_{\Phi(\mathcal{C})}) \in \mathcal{F}_d$, then there is no recipe $\xi \in \Pi_n$ such that $\xi(\Phi(\mathcal{C})\sigma)\downarrow = X\theta'|_p(\Phi(\mathcal{C})\sigma)\downarrow$ and $\mathsf{param}_{\mathsf{max}}(\xi') < \mathsf{param}_{\mathsf{max}}(X\theta'|_p)$; and
2. for all $\xi, \xi' \in st(X\theta' \mid X \in D(\mathcal{C}))$, $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ implies $\xi = \xi'$.

Since $s > 0$, we also know that there exists a matrix M_1 ancestor of M such that M_1 is obtained at the end of step a of the first phase with support s. Hence, there exists a constraint system $\mathcal{C}_1$ ancestor of $\mathcal{C}$ such that $\mathcal{C}_1$ is in M_1.

By a simple induction on the number of rule applied between M_1 and M, we prove, thanks to Lemma 8.5, that there exists $(\sigma_1, \theta_1) \in \mathsf{Sol}(\overline{\mathcal{C}_1})$ such that $\sigma|_{vars^1(\mathcal{C}_1)} = \sigma_1$ and $\theta|_{vars^2(\mathcal{C}_2)} = \theta_1$.

Let $X \in D(\mathcal{C}_1)$ and p a position of $\mathsf{C}\lfloor X\theta_1 \rfloor_{\Phi(\mathcal{C}_1)}$ such that $\mathsf{root}(\mathsf{C}\lfloor X\theta_1 \rfloor_{\Phi(\mathcal{C}_1)}|_p) \in \mathcal{F}_d$. We show that there exists $Y \in D(\mathcal{C})$ and p' a position of $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}$ such that $\mathsf{root}(\mathsf{C}\lfloor Y\theta \rfloor_{\Phi(\mathcal{C})}|_{p'}) \in \mathcal{F}_d$.

The rule Dest was never applied to obtained M from M_1, thus $\theta|_{vars^2(\mathcal{C}_2)} = \theta_1$ implies that $\mathsf{root}(\mathsf{C}\lfloor X\theta_1 \rfloor_{\Phi(\mathcal{C}_1)}|_p) = \mathsf{root}(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}|_p) \in \mathcal{F}_d$, and so $\mathsf{root}(\mathsf{C}\lfloor X\theta_1 \rfloor_{\Phi(\mathcal{C}_1)}|_p) \in \mathcal{F}_d$ implies $\mathsf{root}(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}|_p) \in \mathcal{F}_d$. But thanks to Property 7 of a well formed constraint system, we know that $\mathsf{C}\lfloor X\mathsf{mgu}(Er(\mathcal{C})) \rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$ and for all $Y \in vars^2(X\mathsf{mgu}(Er(\mathcal{C}))$, $Y \in vars^2(D(\mathcal{C}))$. Hence, $\mathsf{root}(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}|_p) \in \mathcal{F}_d$ implies that there exists $Y \in vars^2(D(\mathcal{C}))$ and a position p' such that $Y\theta|_{p'} = X\theta|_p$ and $\mathsf{root}(\mathsf{C}\lfloor Y\theta \rfloor_{\Phi(\mathcal{C})}|_{p'}) \in \mathcal{F}_d$.

Hence, we deduce that (σ_1, θ_1) satisfies:

1. for all $X \in D(\mathcal{C}_1)$, for all position p of $\mathsf{C}\lfloor X\theta_1 \rfloor_{\Phi(\mathcal{C}_1)}$, if $\mathsf{root}(\mathsf{C}\lfloor X\theta_1 \rfloor_{\Phi(\mathcal{C})}) \in \mathcal{F}_d$, then there is no recipe $\xi \in \Pi_n$ such that $\xi(\Phi(\mathcal{C}_1)\sigma)\downarrow = X\theta_1|_p(\Phi(\mathcal{C}_1)\sigma)\downarrow$ and $\mathsf{param}_{\mathsf{max}}(\xi) < \mathsf{param}_{\mathsf{max}}(X\theta_1|_p)$; and
2. for all $\xi, \xi' \in st(X\theta_1 \mid X \in D(\mathcal{C}_1))$, $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ implies $\xi = \xi'$.

Moreover, let M_2 be the matrix ancestor of M_1 obtained from step e with support $s - 1$. Thanks to our inductive hypothesis, we know that for all constraint system $\mathcal{C}_0$ in M_2, we have $\mathsf{Sol}(\overline{\mathcal{C}_0}) = \mathsf{Sol}(\mathcal{C}_0)$. Hence, we can apply Lemma C.46 on $\mathcal{C}_1$ and (σ_1, θ_1) which implies that there exists a constraint system $\mathcal{C}_2$ in the same column of $\mathcal{C}_1$ and there exists $(\sigma_2, \theta_2) \in \mathsf{Sol}(\mathcal{C}_2)$ such that $\sigma_1|_{S_1(\mathcal{C}_1)} = \sigma_2|_{S_1\mathcal{C}_2}$ and for all $X \in S_2(\mathcal{C}_1)$, $X\theta_1 = X\theta_2$. $X\theta_1 = X\theta_2$ trivially implies that $X\theta_1 \mathcal{R}_{\mathcal{F}_c} X\theta_2$. Moreover, since $X\theta_1 = X\theta_2$ and for all $\xi, \xi' \in st(X\theta_1 \mid X \in D(\mathcal{C}_1))$, $\mathsf{path}(\xi) = \mathsf{path}(\xi')$ implies $\xi = \xi'$, we deduce that for all $X, Y \in S_2(\mathcal{C}_2)$, for all $p \in \mathcal{P}os(\mathsf{C}\lfloor X\theta'_2 \rfloor_{\Phi(\mathcal{C}_2)})$, for all $p \in \mathcal{P}os(\mathsf{C}\lfloor Y\theta'_2 \rfloor_{\Phi(\mathcal{C}'_2)})$, if $\mathsf{path}(X\theta'_2|_p) = \mathsf{path}(X\theta'_2|_{p'})$ then $X\theta'_2|_p = X\theta'_2|_{p'}$.

Once again with a simple induction on the number of rule applied between M_1 and M , we use Lemma C.49 to prove that there exists a constraint system $\mathcal{C}'$ in the column of $\mathcal{C}$ in M and $(\sigma'', \theta'') \in \mathsf{Sol}(\mathcal{C}')$ such that $\sigma''|_{S_1(\mathcal{C}')} = \sigma|_{S_1(\mathcal{C})}$ and for all $X \in S_2(\mathcal{C})$, $X\theta' = X\theta''$.

But thanks to Lemma C.30, we know that the matrix M satisfies the invariant $\mathsf{InvMatrix}()s)$. Thus there exists a renaming ρ of first order variable such that:

— $\{x\rho \mid (X, i \overset{?}{\vdash} x) \in D(\mathcal{C}) \wedge i \leq s\} = \{x \mid (X, i \overset{?}{\vdash} x) \in D(\mathcal{C}') \wedge i \leq s\}$
— $\{u\rho \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}) \wedge i \leq s\} = \{u \mid (\xi, i \rhd u) \in \Phi(\mathcal{C}') \wedge i \leq s\}$
— $ND(\mathcal{C})\rho = ND(\mathcal{C}')$

Moreover, $(\sigma, \theta') \in \mathsf{Sol}(\overline{\mathcal{C}})$, $(\sigma'', \theta'') \in \mathsf{Sol}(\mathcal{C}')$ and $\sigma|_{S_1(\mathcal{C})} = \sigma''|_{S_1(\mathcal{C}')}$. Hence by Lemma C.44, we can deduce that for all $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C}')$, with $i \leq s$, we have that $(X, i \overset{?}{\vdash} x\rho) \in D(\mathcal{C}')$ and $x\sigma'' = x\rho\sigma$. Thus with $ND(\mathcal{C})\rho = ND(\mathcal{C}')$ and $(\sigma'', \theta') \in \mathsf{Sol}(\mathcal{C}')$, we have that $\sigma'' \vDash ND(\mathcal{C}')$ which implies $\sigma'' \vDash ND(\mathcal{C})\rho$. Since for all $x \in vars^1(ND(\mathcal{C}))$, $\mathsf{ind}_{\mathcal{C}}(x) \leq s$, we conclude that $\rho\sigma \vDash ND(\mathcal{C})\rho$ which implies that $\sigma \vDash ND(\mathcal{C})$ and so $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. □

Lemma 8.9. *Let (M, M') be a pair of matrix obtained at the end strategy. For all constraint system $\mathcal{C}$ in M or M', $\mathsf{Sol}(\mathcal{C}) = \mathsf{Sol}(\overline{\mathcal{C}})$.*

Proof. Let $(\sigma, \theta) \in \mathsf{Sol}(\overline{\mathcal{C}})$. Let $\mathcal{C}'$ be the constraint system ancestor of $\mathcal{C}$ such that $\mathcal{C}'$ is on the matrix obtained from the last step e of phase 1. With a simple induction on the number of rule applied from $\mathcal{C}'$ to $\mathcal{C}$, we use Lemma 8.5 to show that there exists $(\sigma', \theta') \in \mathsf{Sol}(\overline{\mathcal{C}'})$ with $\sigma|_{vars^1(\mathcal{C}')} = \sigma'$. But by Lemma C.50, we know that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C}')$. Furthermore, the rules applied on phase 2 of the strategy do not add new non-deducible constraint system hence we can deduce that $\sigma' \vDash ND(\mathcal{C}')$ and $\sigma|_{vars^1(\mathcal{C}')} = \sigma'$ implies that $\sigma \vDash ND(\mathcal{C})$ and so $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. □

C.6.3 Proof of symbolic equivalence on a leaf

Lemma C.51. *Let $\mathcal{C}$ a well formed constraint system in a pair of matrices of constraint systems obtained by application of the strategy on initial pair of matrices of constraint systems. Let σ, σ' two substitution such that $\sigma \vDash Eq$ and $\sigma' \vDash Eq$. If $\sigma_{|S_1} = \sigma'_{|S_1}$, then $\sigma = \sigma'$.*

Proof. The proof of this Lemma is direct from the definition of the rules. Indeed, S_1 correspond to the first order variable in the initial pair of matrices of constraint systems. Furthermore, even if a rule may create fresh first order variable, they are always related to existing variables by an equation on first order terms. □

Lemma 8.8. *Let (M, M') be a pair of matrix obtained at the end strategy. (M, M') is in solved form.*

Proof. Since (M, M') is obtained at the end of the strategy then there exists (M_0, M'_0), (M_1, M'_1) and (M_2, M'_2) such that

$$(M_0, M'_0) \to^* (M_1, M'_1) \to^* (M_2, M'_2) \to^* (M, M')$$

and (M_0, M'_0) is the pair of row matrices of initial constraint system, (M_1, M'_1) is obtained at the end of Phase 1 of the strategy and (M_2, M'_2) is obtained at the end of Step a of Phase 2 of the strategy.

Thanks to Lemma C.31, we know that (M_1, M'_1) satisfies PP1E. Hence for all constraint system in M_1 or M'_1, $\mathcal{C}$ satisfies InvDedsub, InvVarFrame(∞), InvDest(∞), InvNoUse(∞), InvUntouched(∞) and InvVarConstraint(∞). The rule applied on Phase 2 are only Cons, Eq-right-right and Axiom. Hence thanks to Lemma C.16, we deduce that any constraint system in (M, M') satisfies InvNoUse(∞), InvDest(∞), InvVarFrame(∞) and InvDedsub. Moreover by Lemma C.15, we deduce that any constraint system in (M, M') satisfies InvUntouched(∞). At last, by Lemma C.14, we deduce that any constraint system in (M, M') also satisfies InvVarConstraint(∞).

Since (M_1, M'_1) satisfies PP1E, we deduce that (M, M') satisfies InvMatrix(∞) and InvGeneral. Hence thanks to Lemma C.17 and C.19, we deduce that (M, M') also satisfies InvMatrix(∞) and InvGeneral.

Let $\mathcal{C} \neq \bot$ in M or M'. (M, M') being obtained at the end of the strategy implies that the rule Cons and Axiom are not applicable on any constraint system in M, M'. Hence we deduce that either property 2 holds or there exists $X \in D(\mathcal{C})$ such that Axiom(X, path) and Cons(X, f)

are useless for all path, f. But we know that $\mathcal{C}$ satisfies $\mathsf{InvDest}(\infty)$. Hence $\textsc{Dest}(\xi, \ell \to r, s)$ is useless for any $\xi, \ell \to r, s$. This case is impossible since $\mathcal{C}$ is normalised and we assumed that $\mathcal{C} \neq \bot$. Hence $\mathcal{C} \neq \bot$ implies that property 2 holds on $\mathcal{C}$.

We show the properties 1 and 2 of solved pair of matrices on (M, M'). If we assumed that $\mathcal{C}$ and $\mathcal{C}'$ was in the same column. We know that (M, M') satisfies the invariant $\mathsf{InvGeneral}$ and $\mathcal{C}, \mathcal{C}'$ both satisfy $\mathsf{InvVarConstraint}(\infty)$ and $\mathsf{InvUntouched}(\infty)$. Hence by Lemma C.29, we deduce that the properties 1, 2 hold.

It remains to show property 1 of solved constraint system and property 3 of solved pair of matrices. Thanks to Lemma C.32, we know that (M_2, M_2') does not contain universal variables. But the rule Cons, Axiom and Eq-right-right do not add new universal variable. Hence (M, M') does not contain universal variables. (M, M') being at the end of the strategy implies that the rule Axiom, Cons and Eq-right-right are no longer applicable. Consider $\mathcal{C}$ a constraint system in (M, M') and T its association table. We show that all disjunction of inequation are in the association table T.

Let $\bigvee_i u_i \stackrel{?}{\neq} v_i$ such that $Eq(\mathcal{C}) = E \wedge \bigvee_i u_i \stackrel{?}{\neq} v_i$ for some E, and $T[\bigvee_i u_i \neq v_i] = \bot$. Thanks to Lemma C.33, we know that for all i, $u_i \stackrel{?}{\neq} v_i$ satisfies one of the following properties:

1. $u_i \in \mathcal{X}^1$ and $v_i \in \mathcal{N}$: In such a case, there exists $(X, k \stackrel{?}{\vdash} u_i) \in D$. But $\textsc{Axiom}(X, \mathsf{path})$ is useless for any path. Since $\mathcal{C}$ satisfies $\mathsf{InvDest}(\infty)$, $\textsc{Dest}(\xi, \ell \to r, s)$ is useless for any $\xi, \ell \to r, s$. But in such a case, the normalisation rule (Nname) is applicable which is a contradiction with the fact that $\mathcal{C}$ is normalised. Hence this case is impossible.
2. $u_i, v_i \in \mathcal{X}^1$, $Er(\mathcal{C}) \not\vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}) \vDash \mathsf{root}(Y) \stackrel{?}{\neq} \mathsf{g}$, for all $\mathsf{f}, \mathsf{g} \in \mathcal{F}_c$, where $(X, p \stackrel{?}{\vdash} u_i), (Y, q \stackrel{?}{\vdash} v_i) \in D(\mathcal{C})$: In such a case, for all $\mathsf{g} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(Y) \stackrel{?}{\neq} \mathsf{g}$ implies that $\textsc{Cons}(Y, \mathsf{g})$ is useless. But $\textsc{Axiom}(Y, \mathsf{path})$ is not applicable and $Er(\mathcal{C}) \vDash \mathsf{root}(Y) \stackrel{?}{\neq} \mathsf{g}$ implies that $\textsc{Axiom}(Y, \mathsf{path})$ is useless for all path. Since $\mathcal{C}$ satisfies $\mathsf{InvDest}(\infty)$, $\textsc{Dest}(\xi, \ell \to r, s)$ is useless for any $\xi, \ell \to r, s$. But in such a case, the normalisation rule (Nnosol) is applicable which is a contradiction with the fact that $\mathcal{C}$ is normalised. Hence this case is impossible.
3. $u_i \in \mathcal{X}^1$, $\mathsf{root}(v_i) \in \mathcal{F}_c$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$, where $(X, p \stackrel{?}{\vdash} u_i) \in D(\mathcal{C})$: Since for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$, then similarly to the previous case, we prove that this case is impossible.

Since all cases are impossible, we can deduce that $T[\bigvee_i u_i \stackrel{?}{\neq} v_i] \neq \bot$. Hence let $\bigvee_j \xi_j \stackrel{?}{\neq} \xi_j$ such that $T[\bigvee_i u_i \stackrel{?}{\neq} v_i] = \bigvee_i \xi_i \stackrel{?}{\neq} \xi_i$. Once again thanks to Lemma C.33, we know that either for all i, $st(\xi_i, \xi_i') \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) = \emptyset$ or else for all i, $\xi_i \stackrel{?}{\neq} \xi_i'$ satisfies one of the following properties:

1. $\xi_i \in (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X})$
2. $\xi_i, \xi_i' \in \mathcal{X}^2$, for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}) \not\vDash \mathsf{root}(\xi_i') \stackrel{?}{\neq} \mathsf{f}$
3. $\xi_i \in \mathcal{X}^2$, $\mathsf{root}(\xi_i') \in \mathcal{F}_c$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$

We already prove that if for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(X) \neq \mathsf{f}$ for some $X \in vars^2(D)$ then thanks to $\textsc{Axiom}(X, \mathsf{path})$ not being applicable and $\mathcal{C}$ satisfying $\mathsf{InvDest}(\infty)$, the normalisation rule (Nnosol) is applicable which is a contradiction with the fact that $\mathcal{C}$ is normalised. Hence, we deduce that either (a) for all i, $st(\xi_i, \xi_i') \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) = \emptyset$ or else (b) for all i, $st(\xi_i, \xi_i') \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) \stackrel{?}{\neq} \emptyset$. In case (b), we describe in Subsection 7.4 the entry $\bigvee_i u_i \stackrel{?}{\neq} v_i$ should be removed, *i.e.* $T[\bigvee_i u_i \stackrel{?}{\neq} v_i] = \bot$ which is a contradiction with or hypothesis. Hence, we deduce that only case (a) is possible.

To sum up, we proved that for all $\bigvee_i u_i \stackrel{?}{\neq} v_i$, for all E, $Er(\mathcal{C}) = E \wedge \bigvee_i u_i \stackrel{?}{\neq} v_i$ implies that there exists $\bigvee_j \xi_j \stackrel{?}{\neq} \xi_j$ such that $T[\bigvee_i u_i \stackrel{?}{\neq} v_i] = \bigvee_j \xi_j \stackrel{?}{\neq} \xi_j$ and for all j, $st(\xi_j, \xi_j') \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) = \emptyset$.

But thanks to Lemma 8.17, we deduce that $\left(\bigvee_j \beta_j \mathsf{acc}^1(\mathcal{C}) \stackrel{?}{\neq} \beta'_j \mathsf{acc}^1(\mathcal{C})\right)\downarrow = \bigvee_i u_i \stackrel{?}{\neq} v_i$. Since for all j, $st(\xi_j, \xi'_j) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$, we deduce that for all i, $st(u_i, v_i) \cap \mathcal{N} = \emptyset$. Hence property 1 of solved constraint system is satisfied.

Furthermore, Lemma 8.17 also indicates that there exists $\bigvee_i u'_i \stackrel{?}{\neq} v'_i$ such that $T'[\bigvee_i u'_i \stackrel{?}{\neq} v'_i] = \bigvee_j \xi_j \stackrel{?}{\neq} \xi_j$ and so $\left(\bigvee_j \beta_j \mathsf{acc}^1(\mathcal{C}') \stackrel{?}{\neq} \beta'_j \mathsf{acc}^1(\mathcal{C}')\right)\downarrow = \bigvee_i u'_i \stackrel{?}{\neq} v'_i$.

Since $\mathcal{C}$ and $\mathcal{C}'$ both satisfy $\mathsf{InvVarConstraint}(\infty)$, we deduce that for all $(X, i \stackrel{?}{\vdash} u) \in D(\mathcal{C})$, $X \in S_2(\mathcal{C})$ and $u \in \mathcal{X}^1$. Moreover, the variables as right hand term of deducible constraints are all distinct. Since $\mathcal{C}$ and $\mathcal{C}'$ have the same shape, then $(X, i \stackrel{?}{\vdash} x) \in D(\mathcal{C})$ implies that there exists $x' \in\in \mathcal{X}^1$ such that $(X, i \stackrel{?}{\vdash} x') \in D(\mathcal{C}')$. Hence we can define ρ such that for all $x\rho = x'$. Therefore, we have $\mathsf{acc}^1(\mathcal{C})\rho = \mathsf{acc}^1(\mathcal{C}')$ and for all j, $st(\xi_j, \xi'_j) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$. Hence, we deduce that $\bigvee_i u_i\rho \stackrel{?}{\neq} v_i\rho = \bigvee_i u'_i\rho \stackrel{?}{\neq} v'_i\rho$. Thus, we can conclude that $Eq(\mathcal{C})\rho$ restricted to inequation is equal to $Eq(\mathcal{C}')$ restricted to inequation, and so the result holds. □

Lemma 8.10. *Let (M, M') be a pair of matrix obtained at the end strategy. Let $\mathcal{C}$ be a constraint system in M or M' different from $\bot$. There exists $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$.*

Proof. Assume that $\mathcal{C} \neq \bot$. Thanks to Lemma 8.8, we know that (M, M') is in solved form. Hence we deduce that $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(\infty)$ and all right hand terms of deducible constraints are distinct variable. Therefore, for each deducible constraint $(X, i \stackrel{?}{\vdash} x) \in D(\mathcal{C})$, we can define θ on $vars^2(D)$ such that $X\theta \in \mathcal{T}(\mathcal{F}_c, \{ax_1\})$ for all $X \in vars^2(D)$.

We show that for all u, $(ax_1, 1 \rhd u) \notin \mathsf{NoUse}(\mathcal{C})$. (M, M') is in solved form implies that $\mathcal{C}$ is well-formed. Hence, if $(ax_1, 1 \rhd u) \notin \mathsf{NoUse}(\mathcal{C})$ then by Definition 8.2, item 8, there exists $X \in vars^2(\mathcal{C})$ such that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\mathsf{mgu}(Er(\mathcal{C}))) < 1$ which is impossible. Hence $(ax_1, 1 \rhd u) \notin \mathsf{NoUse}(\mathcal{C})$ and so for all $\xi \in \mathcal{T}(\mathcal{F}_c, \{ax_1\})$, for all θ, ξ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$.

Since the set $\mathcal{T}(\mathcal{F}_c, \{ax_1\})$ is infinite, we have an infinite set of pair of substitutions (σ, θ) where for all $(X, i \stackrel{?}{\vdash} x) \in D$, $X\theta(\Phi\sigma)\downarrow = x\sigma$, $param(X\theta) = \{ax_1\} \subseteq \{ax_1, \ldots, ax_i\}$ and $X\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$. We extend each of (σ, θ) by (σ', θ') such that $\theta'|_{vars^2(D)} = \theta$, $\sigma'|_{vars^1(D)} = \sigma$, for all $X \in vars^2(\mathcal{C}) \backslash vars^2(D)$, $X\theta' = X\mathsf{mgu}(Er)\theta$; and for all $x \in vars^1(\mathcal{C}) \setminus vars^1(\mathcal{C})$, $x\sigma' = x\mathsf{mgu}(Eq)\sigma$. Hence obtain an infinite set of pre-solution (σ, θ) of $\mathcal{C}$ such that $\sigma \vDash \mathsf{mgu}(Eq)$ and $\theta \vDash \mathsf{mgu}(Er)$. Moreover, we also know that for all $(X, i \stackrel{?}{\vdash} x) \in D(\mathcal{C})$, for all $\mathsf{f} \in \mathcal{F}_c$, $Er \nvdash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$. At last, we also know that $Er \nvdash X \stackrel{?}{\neq} ax_1$. Hence, we deduce that $\theta \vDash Er$.

It remains to prove that there exists a pre-solution in this infinite set that satisfies the inequations in $Eq(\mathcal{C})$. Since each variable in the inequations are a variable of $vars^1(D)$ and the set of possible value for each of these variable is infinite, then Thanks to [CD94], we deduce that there exists at least one (σ_0, θ_0) of pre-solution such that $\sigma_0 \vDash Eq(\mathcal{C})$ and so $(\sigma_0, \theta_0) \in \mathsf{Sol}(\overline{\mathcal{C}})$. Therefore, thanks to Lemma 8.9, we deduce that $(\sigma_0, \theta_0) \in \mathsf{Sol}(\mathcal{C})$ and so the result holds. □

Definition C.2. *Let $\mathcal{C} = (S_1, S_2, \Phi, D, Eq, Er, ND)$ be a well formed solved constraint system. Let σ be a substitution mapping $vars^1(\mathcal{C})$ to ground messages. We define a new semantics on logic formula built upon elementary formulas using classical connectives. The semantics for the elementary formulas are given below and is extended as expected to general formulas. We have: for all $i \in \mathbb{N}$, for all $u, v \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$,*

- $\sigma \vDash_{\leq i} u \stackrel{?}{=} v$ *if* $\sigma \vDash u \stackrel{?}{=} v$
- $\sigma \vDash_{\leq i} u \stackrel{?}{\neq} v$ *if* $\sigma \vDash u \stackrel{?}{\neq} v$ *or there exists* $x \in vars^1(u) \cup vars^1(v)$ *such that* $\mathsf{ind}_{\mathcal{C}}(x) > i$

Lemma C.52. *Let $\mathcal{C}$ be a well formed solved constraint system on a leaf. Let $n \in \mathbb{N}$. Let (σ, θ) such that:*

— $\sigma \vDash_{\leq n} Eq(\mathcal{C})$

— *for all* $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, $X\theta(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma$ *and* $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta) \leq i$

— *for all* $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, $\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \mathcal{AX}))$ *and for* $(\xi, j \rhd v) \in \Phi(\mathcal{C})$, *if* $\mathsf{path}(\xi) \in st(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})})$ *then* $j \leq i$.

— *for all* $X \in D(\mathcal{C})$, $X\theta$ *conforms to* $\Phi(\mathcal{C})\theta$ *w.r.t.* $\mathsf{NoUse}(\mathcal{C})\theta$

There exists $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C})$ *such that* $\sigma_{x|\mathsf{ind}_{\mathcal{C}}(x) \leq n} = \sigma'_{x|\mathsf{ind}_{\mathcal{C}}(x) \leq n}$

Proof. Since $\mathcal{C}$ is in solved formed, we know that it satisfies the invariants $\mathsf{InvVarConstraint}(\infty)$. Hence, we have that for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, u is a variable. Furthermore, all right hand variables of the deducible constraints are distinct. Thus, for all $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, $\mathsf{ind}_{\mathcal{C}}(x) = i$.

Let $\sigma_0 = \sigma_{x|\mathsf{ind}_{\mathcal{C}}(x) \leq n}$. Let $D_0 = \{(X, i \overset{?}{\vdash} x) \in D(\mathcal{C}) \mid i > n\}$, $\Phi_0 = \mathsf{Init}(\Phi)\sigma_0$ and $Eq_0 = Eq(\mathcal{C})\sigma_0$. D_0, Φ_0 and Eq_0 represent a simplified version of $\mathcal{C}$ where we fixed the value of the variable in $\mathrm{dom}(\sigma_0)$.

Let $(ax_1, 1 \rhd u) \in \Phi_0$. Thanks to the origination property of a constraint system, we know that $vars^1(u) = \emptyset$. Furthermore, since $\mathcal{C}$ is a well formed constraint system, we also have that $(ax_1, 1 \rhd u) \notin \mathsf{NoUse}(\mathcal{C})$. Hence for all $\xi \in \mathcal{T}(\mathcal{F}_c \cup \{ax_1\})$, for all substitution λ, we have $\xi(\Phi_0\lambda)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Thus for all $(X, i \overset{?}{\vdash} x) \in D_0$, x can be instantiate by any recipe $\xi \in \mathcal{T}(\mathcal{F}_c \cup \{ax_1\})$. But the set $\mathcal{T}(\mathcal{F}_c \cup \{ax_1\})$ is an infinite set and for all $x \in vars^1(Eq_0)$, $x \in vars^1(D_0)$. Therefore, thanks to [CD94], we deduce that there exists a substitution (σ_1, θ_1) such that $\mathrm{dom}(\theta_1) = vars^2(D_0)$, $\mathrm{dom}(\sigma_1) = vars^1(D_0)$ and:

— for all $(X, i \overset{?}{\vdash} x) \in D_0$, $X\theta_1 \in \mathcal{T}(\mathcal{F}_c \cup \{ax_1\})$ and $x\sigma_1 = X\theta_1(\Phi_0\sigma_1)\downarrow$

— σ_1 satisfies the inequations of Eq_0.

We define θ' such that:

— for all $X \in vars^2(D(\mathcal{C})) \setminus vars^2(D_0)$, $X\theta' = X\theta$

— for all $X \in vars^2(D_0)$, $X\theta' = X\theta_1$

— for all $X \in vars^2(\mathcal{C}) \setminus vars^2(D(\mathcal{C}))$, $X\theta' = X\mathsf{mgu}(Er(\mathcal{C}))\theta'$.

Furthermore, we define σ' such that:

— $\sigma'|_{x|\mathsf{ind}_{\mathcal{C}}(x) \leq n} = \sigma_0 = \sigma_{x|\mathsf{ind}_{\mathcal{C}}(x) \leq n}$

— $\sigma'|_{x|\mathsf{ind}_{\mathcal{C}}(x) > n} = \sigma_1$

— for all $x \in vars^1(\mathcal{C}) \setminus vars^1(D(\mathcal{C}))$, $x\sigma' = x\mathsf{mgu}(Eq(\mathcal{C}))\sigma'$.

We verify that $(\sigma', \theta') \in \mathsf{Sol}(\overline{\mathcal{C}})$: For all $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, if $i \leq n$ then $X\theta' = X\theta$. But $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X\theta) \leq i$ and $\sigma'|_{x|\mathsf{ind}_{\mathcal{C}}(x) \leq n} = \sigma_0 = \sigma_{x|\mathsf{ind}_{\mathcal{C}}(x) \leq n}$. Thus we have that $X\theta'(\Phi(\mathcal{C})\sigma')\downarrow = X\theta(\Phi(\mathcal{C})\sigma)\downarrow = x\sigma = x\sigma'$.

Furthermore, since for all $(\xi, j \rhd v) \in \Phi(\mathcal{C})$, for all $Y \in vars^2(\xi)$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) < j$, thanks to $\mathcal{C}$ being in solved form and so satisfying the invariant $\mathsf{InvVarFrame}(\infty)$. But for all $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, for all $(\xi, j \rhd v) \in \Phi(\mathcal{C})$, $\mathsf{path}(\xi) \in st(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})})$ implies $j \leq i$ and so for all $Y \in vars^2(\mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^2(()\mathcal{C}))$, $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(Y) < \mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(X)$. Hence we deduce with a simple induction on i that for all $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, if $i \leq n$ then $X\theta$ conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta$ implies that $X\theta'$ conforms to $\Phi(\mathcal{C})\theta'$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta'$.

Moreover, for all $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, if $i > n$ then $X\theta' \in \mathcal{T}(\mathcal{F}_c \cup \{ax_1\})$ and so $X\theta'$ trivially conforms to $\Phi(\mathcal{C})\theta'$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta'$.

We already know that σ_1 satisfies the inequations of Eq_0 where $Eq_0 = Eq(\mathcal{C})\sigma_0$. Hence by definition of σ', we have that $\sigma' \vDash Eq(\mathcal{C})$.

At last, we know that for all $X \in vars^2(\mathcal{C}) \setminus vars^2(D(\mathcal{C}))$, $X\theta' = X\mathsf{mgu}(Er(\mathcal{C}))\theta'$. Furthermore, since $\mathcal{C}$ is in solved form, we have that for all $X \in vars^2(D(\mathcal{C}))$, for all $\mathsf{f} \in \mathcal{F}_c$, for all ξ recipe of $\Phi(\mathcal{C})$, $Er(\mathcal{C}) \not\vdash X \overset{?}{\neq} \xi$ and $Er(\mathcal{C}) \not\vdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$. Hence, we conclude that $\theta' \vDash Er(\mathcal{C})$.

To sum up, we have proved that $(\sigma', \theta') \in \mathsf{Sol}(\overline{\mathcal{C}})$. But since $\mathcal{C}$ is a constraint system on a leaf, then by Lemma 8.9, we know that $\mathsf{Sol}(\mathcal{C}) = \mathsf{Sol}(\overline{\mathcal{C}})$. Hence we conclude that $(\sigma', \theta') \in \mathsf{Sol}(\mathcal{C})$. □

Lemma 8.11. *Let (M, M') be a pair of matrix obtained at the end strategy. Let $\mathcal{C}, \mathcal{C}'$ be two constraint system in the same line in (M, M') ($\mathcal{C}$ and $\mathcal{C}'$ may be contained in the same matrix). If $\mathcal{C} \neq \bot$ and $\mathcal{C}' \neq \bot$ then $\mathcal{C} \approx_s \mathcal{C}'$.*

Proof. We show one side of the equivalence, the other side being done symmetrically. Let $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$. Thanks to Lemma 8.8, we know that (M, M') are in solved form. We will show that there exists σ' such that:

1. (σ', θ) is a pre-solution of $\mathcal{C}'$ with $\sigma' \vDash \mathsf{mgu}(Eq(\mathcal{C}'))$ and $\theta \vDash Er(\mathcal{C}')$;
2. $\sigma' \vDash Eq(\mathcal{C}')$ and for all $\xi, \xi' \in \Pi_n$, if $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C})}, \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$ then
   - $\xi(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C})\sigma)\downarrow$ is equivalent to $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = \xi'(\Phi'(\mathcal{C}')\sigma')\downarrow$
   - $\xi(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ is equivalent to $\xi(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$
3. $\Phi(\mathcal{C})\sigma \sim \Phi(\mathcal{C}')\sigma'$ and $(\sigma', \theta) \in \mathsf{Sol}(\mathcal{C}')$

Property 1: Since (M, M') are in solved form then $\mathcal{C}$ and $\mathcal{C}'$ also have the same structure. Hence, we deduce that $Er(\mathcal{C}) = Er(\mathcal{C}')$. But $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$, thus $\theta \vDash Er(\mathcal{C})$ and so $\theta \vDash Er(\mathcal{C}')$. Moreover, since $\mathcal{C}'$ is normalised, then $\mathsf{mgu}(Eq(\mathcal{C}'))$ exists and $vars^1(\Phi') \cup vars^1(D') = \mathrm{img}(\mathsf{mgu}(Eq')))$. Thus, we will first define σ' on the variables contain in Φ' and D'; and then for any variable $y \in vars^1(\mathcal{C}')$ we will have $y\sigma' = y\ \mathsf{mgu}(Eq')\sigma'$.

We define σ' recursively on the index of minimal constraint of a variable x:

Base case $\mathsf{ind}_{\mathcal{C}'}(x) = 0$ *:* By definition of a constraint system, for all $(X, k \overset{?}{\vdash} u) \in D(\mathcal{C}')$, $k > 0$ which means that for all $x \in vars^1(D')$, $\mathsf{ind}_{\mathcal{C}'}(x) > 0$. Thus, the result trivially holds.

Inductive step $\mathsf{ind}_{\mathcal{C}'}(x) > 1$ *:* Let $(X, k \overset{?}{\vdash} x) \in D(\mathcal{C}')$ such that $k = \mathsf{ind}_{\mathcal{C}'}(x)$. Since $\mathcal{C}$ and $\mathcal{C}'$ have same structure, we deduce that there exists $(X, k \overset{?}{\vdash} y) \in D(\mathcal{C})$ and $param(X\theta) \subseteq \{ax_1, \ldots, ax_k\}$. $\mathcal{C}$ being in solved form indicates that $\mathcal{C}$ satisfies $\mathsf{InvDest}(\infty)$. Hence thanks to Lemma C.37, we have that $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$, which also means that $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$.

$(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies that θ conforms to $\Phi(\mathcal{C})\theta$ w.r.t $\mathsf{NoUse}(\mathcal{C})\theta$. Once again, due to the same structure between $\mathcal{C}$ and $\mathcal{C}'$, we have $\{\xi, i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C})\} = \{\xi, i \mid (\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}')\}$. Thus, θ conforms to $\Phi(\mathcal{C})\theta$ w.r.t $\mathsf{NoUse}(\mathcal{C})\theta$ implies that θ conforms to $\Phi(\mathcal{C}')\theta$ w.r.t $\mathsf{NoUse}(\mathcal{C}')\theta$.

Let $\zeta \in st(()X\theta)$ such that $\mathsf{C}\lfloor\zeta\rfloor_{\Phi(\mathcal{C}')} \in (\mathcal{F}_d^* \cdot \mathcal{AX})$. By definition of a context we know that there exists $(\xi, p \rhd v) \in \Phi(\mathcal{C}')$ such that $\mathsf{path}(\xi) = \mathsf{path}(\zeta)$. Furthermore, since θ conforms to $\Phi(\mathcal{C}')\theta$ w.r.t $\mathsf{NoUse}(\mathcal{C}')\theta$, we have that $\zeta = \xi\theta$. Since $\mathcal{C}'$ is in solved formed, $\mathcal{C}$ satisfies $\mathsf{InvVarFrame}(\infty)$ and so for all $Y \in vars^1\xi$, there exists $(Y, q \overset{?}{\vdash} y) \in D(\mathcal{C}')$ such that $q < p$. But we also know that the right hand term of the deducible constraints are distinct variables. Hence, we have that $\mathsf{ind}_{\mathcal{C}'}(y) = q < p$. Moreover $(\sigma, \theta) \in \mathsf{Sol}(\mathcal{C})$ implies, thanks to (M, M') satisfying $\mathsf{InvGeneral}$, that $param(\xi\theta) \subseteq \{ax_1, \ldots, ax_p\}$ and so $p \leq k$. Thus, we can deduce that $\mathsf{ind}_{\mathcal{C}'}(y) < k$. By applying our inductive hypothesis on y, we know that $(Y\theta)(\Phi(\mathcal{C}')\sigma')\downarrow = y\sigma'$. By Property 5 of a well formed constraint system, we now can deduce $(\xi\theta)(\Phi(\mathcal{C}')\sigma')\downarrow = \zeta(\Phi(\mathcal{C}')\sigma')\downarrow = v\sigma' \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$.

Furthermore, we proved that $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$ which allows us to conclude that $(X\theta)(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and so we define $x\sigma'$ such that : $x\sigma' = (X\theta)(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$.

Property 2: We first prove that for all $n \in \mathbb{N}$, $\sigma' \vDash_{\leq n} Eq(\mathcal{C}')$ implies that for all $\xi, \xi' \in \Pi_n$, if $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')}, \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$, ξ and ξ' conforms to $\Phi(\mathcal{C}')\theta$, and for all $x \in vars^1(\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}')) \cup vars^1(\mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}'))$, $\mathsf{ind}_{\mathcal{C}'}(x) \leq n$, then

— $\xi(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ implies $\xi(\Phi(C)\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$

— $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = \xi'(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ implies $\xi(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$

We prove this result by induction on $(|\xi(\Phi(\mathcal{C}')\sigma')\downarrow|, \mathsf{param}_{\mathsf{max}}(\xi) + \mathsf{param}_{\mathsf{max}}(\xi'))$:

Base case $(|\xi(\Phi(\mathcal{C}')\sigma')\downarrow|, \mathsf{param}_{\mathsf{max}}(\xi) + \mathsf{param}_{\mathsf{max}}(\xi')) = (0, 0)$*:* Such a case is impossible thus the result holds.

Inductive step $(|\xi(\Phi(\mathcal{C}')\sigma')\downarrow|, \mathsf{param}_{\mathsf{max}}(\xi) + \mathsf{param}_{\mathsf{max}}(\xi')) > (0, 0)$ *:* We prove the result by case analysis on the two recipes ξ and ξ' :

— $\mathsf{root}(\xi) = \mathsf{root}(\xi') \in \mathcal{F}_c$: In such a case, assume that $\xi = f(\xi_1, \ldots, \xi_n)$ and $\xi' = f(\xi'_1, \ldots, \xi'_n)$. Since $f \in \mathcal{F}_c$, $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = \xi'(\Phi(\mathcal{C}')\sigma')\downarrow$ implies $\xi_k(\Phi(\mathcal{C}')\sigma')\downarrow = \xi'_k(\Phi(\mathcal{C}')\sigma')\downarrow$, for $k = 1 \ldots n$ and $|\xi_k(\Phi(\mathcal{C}')\sigma')\downarrow| < |\xi(\Phi(\mathcal{C}')\sigma')\downarrow|$, for k $= 1 \ldots n$. At last, since $vars^1(\mathsf{C}\lfloor\xi_k\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}')) \subseteq vars^1(\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}'))$, for $k = 1 \ldots n$, then we can apply our inductive hypothesis on ξ_k and ξ'_k which means that for all $k \in \{1, \ldots, n\}$,

$$\xi_k(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N}) \text{ and } \xi_k(\Phi(\mathcal{C})\sigma)\downarrow = \xi'_k(\Phi(\mathcal{C})\sigma)\downarrow$$

Thus, we deduce that $\xi(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and $\xi(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C})\sigma)\downarrow$.

— $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{F}_d^* \cdot \mathcal{AX}$ and there exists $(\zeta, p \rhd u'_1) \in \mathsf{NoUse}(\mathcal{C}')$ with $\zeta\theta = \xi$: First of all, $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{F}_d^* \cdot \mathcal{AX}$ and ξ conforms to $\Phi(\mathcal{C}')\theta$ implies that there exists $(\zeta, q \rhd u'_1) \in \Phi(\mathcal{C}')$ such that $\zeta\theta = \xi$.

$\mathcal{C}$ and $\mathcal{C}'$ being on the same line of a pair of matrices of constraint systems on the leaves, we deduce that $Er(\mathcal{C}) = Er(\mathcal{C}')$ and there exists $u'_1 \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$ such that $(\zeta, p \rhd u'_1) \in \Phi(\mathcal{C}) \cap \mathsf{NoUse}(\mathcal{C})$. Let's denote $\Theta = \mathsf{mgu}(Er(\mathcal{C}))$, we have $\Theta = \Theta'$.

Since $\mathcal{C}$ is well-formed then by the property 5 of a well-formed constraint system, we deduce that $(\zeta\theta)\Phi(\mathcal{C})\sigma\downarrow = u_1\sigma \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$ and so $\xi(\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c, \mathcal{N})$. Secondly, by the property 8, we also know that there exists $X \in vars^2(\mathcal{C}') = vars^2(\mathcal{C})$ such that

— $\mathsf{C}\lfloor X\Theta'\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c, \mathcal{F}_d^* \cdot \mathcal{AX} \cup \mathcal{X}^2)$

— $\mathsf{C}\lfloor X\Theta'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}') = u'_1$ and $\mathsf{param}_{\mathsf{max}}^{\mathcal{C}'}(X\Theta) < p$

where $\Theta' = \mathsf{mgu}(Er(\mathcal{C}')$.

Furthermore, by hypothesis, we assumed that for all $(Z, q \overset{?}{\vdash} z) \in D(\mathcal{C}')$, $\mathsf{C}\lfloor Z\theta\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$, thus we have that $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$. At last, by Property 5 of a well formed constraint system, we can also conclude that $(X\theta)\Phi(\mathcal{C}')\sigma'\downarrow = u'_1\sigma'\downarrow$.

Furthermore, the equation $\mathsf{C}\lfloor X\Theta'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}') = u'_1$, due to the application of the rule Eq-left-right, implies that $\mathsf{C}\lfloor X\Theta\rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C}) = u_1$. Hence, with the same reasoning, we deduce that $(X\theta)\Phi(\mathcal{C})\sigma\downarrow = u_1\sigma\downarrow$. But $\mathsf{param}_{\mathsf{max}}(X\theta) < p$ therefore we can apply our inductive hypothesis on $(X\theta, \xi')$ which means that $\xi'\Phi(\mathcal{C})\sigma\downarrow = (X\theta)\Phi(\mathcal{C})\sigma\downarrow = u_1\sigma\downarrow = \xi(\Phi(\mathcal{C})\sigma)\downarrow$.

— $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')}, \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{F}_d^* \cdot \mathcal{AX}$: In such a case, we know that there exists $(\zeta, p \rhd u'_1), (\zeta', p' \rhd u'_2) \in \Phi(\mathcal{C}')$ such that $\zeta\theta = \xi$ and $\zeta'\theta = \xi'$. Furthermore, since $\mathcal{C}$ and $\mathcal{C}'$ have the same structure, there exists $u_1, u_2 \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$ such that $(\zeta, p \rhd u_1), (\zeta', p' \rhd u_2) \in \Phi(\mathcal{C})$.

Since $\mathcal{C}$, $\mathcal{C}'$ are well-formed then by the property 5 of a well-formed constraint system, we deduce that $\xi(\Phi(\mathcal{C})\sigma)\downarrow = u_1\sigma$, $\xi'(\Phi(\mathcal{C})\sigma)\downarrow = u_2\sigma$, $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = u'_1\sigma'$ and $\xi'(\Phi(\mathcal{C}')\sigma')\downarrow = u'_2\sigma'$.

But (M, M') is a leaf, then the rule Eq-left-left(ζ, ζ') is already applied on (M, M'). Thus,

— either we have $Eq(\mathcal{C}) \vDash u_1 \overset{?}{=} u_2$ and $Eq(\mathcal{C}') \vDash u'_1 \overset{?}{=} u'_2$: By the normalisation of a constraint system, we deduce that $u_1 = u_2$ and $u'_1 = u'_2$. Thus, we trivially have that $\xi(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C})\sigma)\downarrow$.

— or $Eq(\mathcal{C}) \vDash u_1 \overset{?}{\neq} u_2$ and $Eq(\mathcal{C}') \vDash u_1' \overset{?}{\neq} u_2'$: $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = \xi'(\Phi(\mathcal{C}')\sigma')\downarrow$ implies that $\sigma' \nvDash u_1' \overset{?}{\neq} u_2'$. But we know that for all $x \in vars^1(u_1') \cup vars^1(u_2')$, $\mathsf{ind}_{\mathcal{C}'}(x) \leq n$, thus we have $\sigma' \nvDash u_1 \overset{?}{\neq} u_2$ implies that $\sigma' \nvDash_{\leq n} Eq(\mathcal{C}')$, which is in contradiction with our hypothesis.

— $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{F}_d^* \cdot \mathcal{AX}$ and $\mathsf{root}(\xi') \in \mathcal{F}_c$: By Lemma C.52, we know that there exists $(\sigma'', \theta') \in \mathsf{Sol}(\mathcal{C}')$ such that $\sigma'_{x|\mathsf{ind}_{\mathcal{C}'}(x)\leq n} = \sigma''_{x|\mathsf{ind}_{\mathcal{C}'}(x)\leq n}$.

Since $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{F}_d^* \cdot \mathcal{AX}$, then there exists $(\zeta, p \rhd u) \in \Phi(\mathcal{C})$ and $(\zeta, p \rhd u') \in \Phi(\mathcal{C}')$ such that $\zeta\theta = \xi$. Furthermore, since $\mathcal{C}$, $\mathcal{C}'$ are well-formed then by the property 5 of a well-formed constraint system, we deduce that $\xi(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma$ and $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = u'\sigma'$. Since $(\sigma'', \theta') \in \mathsf{Sol}(\mathcal{C}')$, we also have that $\zeta\theta'(\Phi(\mathcal{C}')\sigma'')\downarrow = u'\sigma''$.

By hypothesis, we know that for all $x \in vars^1(\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}'))$, $\mathsf{ind}_{\mathcal{C}'}(x) \leq n$. Furthermore since $\sigma'_{x|\mathsf{ind}_{\mathcal{C}'}(x)\leq n} = \sigma''_{x|\mathsf{ind}_{\mathcal{C}'}(x)\leq n}$, we can deduce that $u'\sigma' = u'\sigma''$. Thus we have $\zeta\theta'(\Phi(\mathcal{C}')\sigma'')\downarrow = \xi(\Phi(\mathcal{C}')\sigma')\downarrow$.

Similarly, we have that for all $x \in vars^1(\mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}'))$, $\mathsf{ind}_{\mathcal{C}'}(x) \leq n$. Since $\mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$, we can deduce that $\mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}')\sigma' = \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}')\sigma''$. Hence, $(\sigma'', \theta') \in \mathsf{Sol}(\mathcal{C}')$ implies that $\xi'(\Phi(\mathcal{C}')\sigma')\downarrow = \zeta'(\Phi(\mathcal{C}')\sigma'')\downarrow$ where $\zeta' = \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^2(\mathcal{C}')\theta'$.

Thus, $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = \xi'(\Phi(\mathcal{C}')\sigma')\downarrow$ implies that $\zeta\theta'(\Phi(\mathcal{C}')\sigma'')\downarrow = \zeta'(\Phi(\mathcal{C}')\sigma'')\downarrow$. But $(\sigma'', \theta') \in \mathsf{Sol}(\mathcal{C}')$ implies $\sigma'' \vDash ND(\mathcal{C}')$. Furthermore, since $\mathcal{C}'$ satisfies $\mathsf{InvDedsub}\infty$ and since $\mathsf{root}(\xi') \in \mathcal{F}_c$ implies $\mathsf{root}(\zeta') \in \mathcal{F}_c$, we can deduce that there exists $X_1, \ldots, X_n \in vars^2(\mathcal{C}')$ such that $\mathsf{C}\lfloor\mathsf{f}(X_1, \ldots, X_n)\Theta'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}') = u'$, where $\Theta' = \mathsf{mgu}(Er(\mathcal{C}'))$. Furthermore, since $X_1, \ldots, X_n$ was obtained by the application of the rule DED-ST on the frame element $(\zeta, p \rhd u')$, we also have that $\mathsf{C}\lfloor\mathsf{f}(X_1, \ldots, X_n)\Theta\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}') = u$, where $\Theta = \mathsf{mgu}(Er(\mathcal{C})) = \Theta'$.

But thanks to $\mathcal{C}$ being well formed, we know that for all $i \in \{1, \ldots, n\}$, $\mathsf{C}\lfloor X_i\Theta'\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$. Hence we can deduce from Property 5 of a well-formed constraint system that $\mathsf{f}(X_1, \ldots, X_n)\theta(\Phi(\mathcal{C}')\sigma')\downarrow = u'\sigma'$. Similarly, we also have that $\mathsf{f}(X_1, \ldots, X_n)\theta(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma$.

At last, $\mathsf{root}(\xi') \in \mathcal{F}_c$ implies that there exists $\xi_1, \ldots, \xi_n$ such that $\xi' = \mathsf{f}(\xi_1, \ldots, \xi_n)$. Hence, by applying our inductive hypothesis on $(X_i\theta, \xi_i)$ since $|X_i\theta(\Phi(\mathcal{C}')\sigma'\downarrow| < |\xi\Phi(\mathcal{C}')\sigma'\downarrow|$, for $i = 1 \ldots n$. Hence we deduce that $\mathsf{f}(X_1, \ldots, X_n)\theta(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C})\sigma)\downarrow$. Since we already proved that $\mathsf{f}(X_1, \ldots, X_n)\theta(\Phi(\mathcal{C})\sigma)\downarrow = u\sigma = \xi(\Phi(\mathcal{C})\sigma)\downarrow$, we conclude that $\xi(\Phi\sigma)\downarrow = \xi'(\Phi\sigma)\downarrow$.

We continue the proof of Property 2 by proving that for all $n \in \mathbb{N}$, $\sigma' \vDash_{\leq n} Eq(\mathcal{C}')$. We prove this result by induction on n:

Base case $n = 0$*:* In such a case, we know that for all $u \overset{?}{\neq} v$ in $Eq(\mathcal{C}')$, for all $x \in vars^1(u) \cup vars^1(v)$, $\mathsf{ind}_{\mathcal{C}'}(x) > 0$. Moreover, we know that $u, v \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^1)$ thanks to $\mathcal{C}'$ being in solved formed. Thus we can conclude that $\sigma' \vDash_{\leq 0} Eq(\mathcal{C}')$.

Inductive step $n > 0$ *:* Let $u \overset{?}{\neq} v$ in $Eq(\mathcal{C}')$ such that for all $x \in vars^1(u) \cup vars^1(v)$, $\mathsf{ind}_{\mathcal{C}'}(x) \leq n$. But since $\mathcal{C}'$ is in solved form, we know that there is no name inside u and v. Thus, we can define two recipe ξ, ξ' such that $\xi = u\lambda$, $\xi' = v\lambda$ where λ is the substitution $\{x \to X\theta \mid X, p \overset{?}{\vdash} x \in D(\mathcal{C}')\}$ and $\xi\Phi(\mathcal{C}')\sigma'\downarrow = u\sigma'$, $\xi'\Phi(\mathcal{C}')\sigma'\downarrow = v\sigma'$. We know that for all $(X, p \overset{?}{\vdash} x) \in D(\mathcal{C}')$, $\mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$ therefore we can deduce that $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')}, \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$. Furthermore, for all $x \in vars^1(u, v)$, for all $w \in st(\mathsf{C}\lfloor x\lambda\rfloor_{\Phi(\mathcal{C}')}) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}))$, if $(\zeta, q \rhd t) \in \Phi(\mathcal{C}')$ is the frame element such that $w = \mathsf{path}(\zeta)$, then $q \leq \mathsf{ind}_{\mathcal{C}'}(x) \leq n$ and we know that by the property of origination that for all $y \in vars^1(t)$, $\mathsf{ind}_{\mathcal{C}'}(y) < q$. Thus we deduce that for all $y \in vars^1(\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}')) \cup vars^1(\mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')}\mathsf{acc}^1(\mathcal{C}'))$, $\mathsf{ind}_{\mathcal{C}'}(y) < n$.

Assume now that $u\sigma' = v\sigma'$. By our inductive hypothesis, we know that $\sigma' \vDash_{\leq n-1} Eq'$ and from the first result we showed in Property 2, we can deduce that $\xi\Phi(\mathcal{C}')\sigma'\downarrow = \xi'\Phi(\mathcal{C}')\sigma'\downarrow$ implies that $\xi\Phi(\mathcal{C})\sigma\downarrow = \xi'\Phi(\mathcal{C})\sigma\downarrow$. But thanks to (M, M') being in solved form, we know that that there

exists a renaming ρ such that $u\rho \stackrel{?}{\neq} v\rho$ in $Eq(\mathcal{C})$, $\xi\Phi(\mathcal{C})\sigma\downarrow = u\rho\sigma$ and $\xi'\Phi(\mathcal{C})\sigma\downarrow = v\rho\sigma$. Hence, it implies that $\sigma \not\models Eq(\mathcal{C})$ which is incoherent with $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$. Our assumption is contradicted and so $u\sigma' \neq v\sigma'$. Hence the result holds.

By combining the first and second result of Property 2, we prove that $\sigma' \vDash Eq(\mathcal{C}')$ and for all $\xi,\xi' \in \Pi_n$, if $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')}, \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$ and ξ, ξ' conforms to $\Phi(\mathcal{C}')\theta$ then

— $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = \xi'(\Phi(\mathcal{C}')\sigma)\downarrow$ implies $\xi(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C})\sigma)\downarrow$

— $\xi(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ implies $\xi(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$

By hypothesis, we know that $(\sigma,\theta) \in \mathsf{Sol}(\mathcal{C})$ and so $\sigma \vDash Eq(\mathcal{C})$. Thus, we can use the same reasoning as for the first result to prove that: for all $\xi,\xi' \in \Pi_n$, if $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C})}, \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$ and ξ,ξ' conforms to $\Phi(\mathcal{C})\theta$ then

— $\xi(\Phi(\mathcal{C})\sigma)\downarrow = \xi'(\Phi(\mathcal{C})\sigma)\downarrow$ implies $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = \xi'(\Phi(\mathcal{C}')\sigma')\downarrow$

— $\xi(\Phi(\mathcal{C})\sigma)\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ implies $\xi(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$

Thus, we conclude the proof of Property 2.

Property 3: We have to show that $\Phi(\mathcal{C})\sigma \sim \Phi(\mathcal{C}')\sigma'$ and $(\sigma',\theta) \in \mathsf{Sol}(\mathcal{C}')$. From Property 2, we proved the static equivalence for any recipe $\xi,\xi' \in \Pi_n$ such that in $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C})}, \mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C})} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$ and ξ,ξ' conforms to the frame $\Phi(\mathcal{C})\theta$ (and so $\Phi(\mathcal{C}')\theta$).

Thanks to Property 2, we deduce that $\sigma' \vDash Eq(\mathcal{C}')$. Hence we have that $(\sigma',\theta) \in \mathsf{Sol}(\overline{\mathcal{C}'})$. But, thanks to Lemma 8.9, we know that $\mathsf{Sol}(\overline{\mathcal{C}'}) = \mathsf{Sol}(\mathcal{C}')$. Hence, we have that $(\sigma',\theta) \in \mathsf{Sol}(\mathcal{C}')$. Thus, by Lemma C.37, we can deduce that for all $\xi \in \Pi_n$, $\xi(\Phi(\mathcal{C}')\sigma')\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ and ξ conforms to $\Phi(\mathcal{C}')\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C}')\theta$ implies that $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$.

Let $\xi,\xi' \in \Pi_n$. Let $\mathcal{M}(\xi,\xi')$ be the multiset of recipe ζ such that $\zeta \in st(\xi) \cup st(\xi')$ and ζ doesn't conforms to the frame $\Phi(\mathcal{C}')\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C}')\theta$. We prove our result by induction on the natural order on multiset.

Base case $\mathcal{M} = \emptyset$: In such a case, we can deduce that ξ and ξ' conforms to the frame $\Phi(\mathcal{C}')\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C}')\theta$. Thus, thanks to Lemma C.37, $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$ and $\mathsf{C}\lfloor\xi'\rfloor_{\Phi(\mathcal{C}')} \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}))$. Thus we deduce by applying Property 2.

Inductive case $\mathcal{M} \neq \emptyset$: Let $\zeta \in st(\xi) \cup st(\xi')$ the smallest recipe such that ζ doesn't conform to the frame $\Phi(\mathcal{C})\theta$. In such a case, we can deduce that $\mathsf{C}\lfloor\zeta\rfloor_{\Phi(\mathcal{C})} \in (\mathcal{F}_d^* \cdot \mathcal{AX})$ (if not, we would have $\zeta = f(\zeta_1,\ldots,\zeta_n)$ and there exists $i \in \{1\ldots n\}$ such that ζ_i doesn't conforms to the frame $\Phi(\mathcal{C})\theta$ which contradict ζ being the smallest). Since $\mathsf{C}\lfloor\zeta\rfloor_{\Phi(\mathcal{C})} \in (\mathcal{F}_d^* \cdot \mathcal{AX})$, then there exists $(\beta, i \rhd u) \in \Phi(\mathcal{C})$ and $(\beta, i \rhd u') \in \Phi(\mathcal{C}')$ such that $\mathsf{path}(\beta) = \mathsf{path}(\zeta)$. We do a case analysis on $(\beta, i \rhd u)$:

Case $(\beta, i \rhd u) \notin \mathsf{NoUse}(\mathcal{C})$: In such a case, since ζ does not conforms with $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}\theta$, we deduce that $\zeta \neq \beta\theta$. Hence $\zeta = \mathsf{f}(\zeta_1,\ldots,\zeta_n)$ for some $\mathsf{f} \in \mathcal{F}_d$. Moreover, by minimality of ζ, for all $i \in \{1,\ldots,n\}$, we know that ζ_i conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}\theta$. Furthermore, $\mathsf{path}(\beta) = \mathsf{path}(\zeta)$ implies that $\beta = f(\beta_1,\ldots,\beta_n)$ and $\mathsf{path}(\beta_1) = \mathsf{path}(\zeta_1)$. Thus, ζ_1 conforms to $\Phi(\mathcal{C})\theta$ implies that $\zeta_1 = \beta_1\theta$. Furthermore, we know that $\zeta\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ and $(\beta\theta)\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$, then by Lemma 6.5, we deduce that $\zeta_k\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ and $(\beta_k\theta)\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$, for $k = 1\ldots n$. At last, from the rewriting rule we consider and from $\zeta_1 = \beta_1\theta$, we can deduce that $(\beta_k\theta)\Phi(\mathcal{C})\sigma\downarrow = \zeta_k\Phi(\mathcal{C})\sigma\downarrow$. But all of $\beta_k\theta$ and ζ_k conform to $\Phi(\mathcal{C})\theta$, which means by Property 2 that $(\beta_k\theta)\Phi(\mathcal{C}')\sigma'\downarrow = \zeta_k\Phi(\mathcal{C}')\sigma'\downarrow$, for $k = 1\ldots n$ and so $\zeta\Phi(\mathcal{C}')\sigma'\downarrow = (\beta\theta)\Phi(\mathcal{C}')\sigma'\downarrow$.

At last, since ζ is a subterm of ξ or ξ' (w.l.o.g. subterm of ξ), there exists a position p such that $\zeta = \xi_{|p}$. But $\mathcal{M}(\xi[\beta\theta]_p,\xi')$ is strictly smaller than $\mathcal{M}(\xi,\xi')$ since ζ doesn't conforms to $\Phi(\mathcal{C})\theta$ and $\beta\theta$ does. Thus we can apply our inductive hypothesis on $(\xi[\beta\theta]_p,\xi')$ and so:

— $\xi[\beta\theta]_p\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ is equivalent to $\xi[\beta\theta]_p\Phi(\mathcal{C}')\sigma'\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$

— if $\xi[\beta\theta]_p\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ then $\xi[\beta\theta]_p\Phi(\mathcal{C})\sigma\downarrow = \xi'\Phi(\mathcal{C})\sigma\downarrow$ is equivalent to $\xi[\beta\theta]_p\Phi(\mathcal{C}')\sigma'\downarrow = \xi'\Phi(\mathcal{C}')\sigma'\downarrow$.

But $\zeta\Phi(\mathcal{C})\sigma\downarrow = (\beta\theta)\Phi(\mathcal{C})\sigma\downarrow$ and $\zeta\Phi(\mathcal{C}')\sigma'\downarrow = (\beta\theta)\Phi(\mathcal{C}')\sigma'\downarrow$. Hence we deduce that:

— $\xi\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ is equivalent to $\xi\Phi(\mathcal{C}')\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$

— if $\xi\Phi(\mathcal{C})\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ then $\xi\Phi(\mathcal{C})\sigma\downarrow = \xi'\Phi(\mathcal{C})\sigma\downarrow$ is equivalent to $\xi\Phi(\mathcal{C}')\sigma'\downarrow = \xi'\Phi(\mathcal{C}')\sigma'\downarrow$.

Hence the result holds.

Case $(\beta, i \rhd u) \in \mathsf{NoUse}(\mathcal{C})$*:* Thanks to $\mathcal{C}$ being well-formed, we know that there exists $X \in vars^2(\mathcal{C})$ such that $X\theta \in \Pi_n$, $X\theta$ conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta$, and $X\theta(\Phi(\mathcal{C})\sigma)\downarrow = \beta\theta(\Phi(\mathcal{C})\sigma)\downarrow$. Moreover, similarly to the previous case, we can show that $\zeta\Phi(\mathcal{C})\sigma\downarrow = \beta\theta\Phi(\mathcal{C})\sigma\downarrow$ and $\zeta\Phi(\mathcal{C}')\sigma'\downarrow = \beta\theta\Phi(\mathcal{C}')\sigma'\downarrow$. Hence we would want to apply our inductive hypothesis on $\xi[X\theta]_p$ and ξ' where p is the position of ζ in ξ. However, $\xi[X\theta]_p$ is not necessary a recipe in Π_n. Thus we have to transform first this recipe so that we can apply our inductive hypothesis.

Subproperty: We show that for all recipe $\gamma \in \Pi_n$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, for all position p of ξ, if $\gamma \in \Pi_n$, $\gamma\Phi(\mathcal{C})\sigma\downarrow = \xi|_p\Phi(\mathcal{C})\sigma\downarrow$ and $\gamma\Phi(\mathcal{C}')\sigma'\downarrow = \xi|_p\Phi(\mathcal{C}')\sigma'\downarrow$ then that there exists a subterm γ' of γ and a position p' prefix of p such that $\xi[\gamma]_p(\Phi(\mathcal{C})\sigma)\downarrow = \xi[\gamma']_{p'}\Phi(\mathcal{C})\sigma\downarrow$, $\xi[\gamma]_p(\Phi(\mathcal{C}')\sigma')\downarrow = \xi[\gamma']_{p'}\Phi(\mathcal{C}')\sigma'\downarrow$ and $\xi[\zeta']_{p'} \in \Pi_n$. We prove this result by induction on the length $|p|$ of p.
Base case $|p| = 0$. In such a case we have that $p = \epsilon$. In such a case since $\gamma \in \Pi_n$, we deduce that $\xi[\gamma]_p \in \Pi_n$. Hence the result holds.

Inductive step $|p| > 1$*:* In such a case, we have that $p = p_1 \cdot r$ for some $r \in \mathbb{N}$ and some p_1 such that $|p_1| < |p|$. Assume that $\xi_{|p_1} = \mathsf{f}(\xi_1, \ldots, \xi_n)$. We have to distinguish two cases:

1. $r = 1$, $\mathsf{g} \in \mathcal{F}_d$ *and* $\mathsf{root}(\gamma) \in \mathcal{F}_c$*:* Since $\gamma\Phi(\mathcal{C})\sigma\downarrow = \xi|_p\Phi(\mathcal{C})\sigma\downarrow$ and $\xi\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ then by Lemma 6.5, we deduce that g is reduced. We do a case analysis on f:
   — $\mathsf{f} = \mathsf{sdec}$ with $\xi_{|p_1}[\gamma] = \mathsf{sdec}(\mathsf{senc}(\gamma_1, \gamma_2), \xi_2)$ and $\xi_2\Phi(\mathcal{C})\sigma\downarrow = \gamma_2\Phi(\mathcal{C})\sigma\downarrow$. But $\gamma_2 \in st(\gamma)$ and γ_2 conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta$. Moreover, $\xi_2 \in st(\xi)$ hence $\mathcal{M}(\xi, \xi') > \mathcal{M}(\gamma_2, \xi_2)$. Hence by our main inductive hypothesis, we deduce that $\gamma_2\Phi(\mathcal{C}')\sigma'\downarrow = \xi_2\Phi(\mathcal{C}')\sigma'\downarrow$ and so $\xi_{|p_1}[\gamma]\Phi(\mathcal{C}')\sigma'\downarrow = \gamma_1\Phi(\mathcal{C}')\sigma'\downarrow$. Thus, we apply our inductive hypothesis on $\gamma' = \gamma_1$ and p_1. Hence the result holds
   — $\mathsf{f} \in \mathcal{F}_d \setminus \{\mathsf{sdec}\}$: The proof is similar to the case $\mathsf{f} = \mathsf{sdec}$.
2. *Otherwise:* By definition of Π_n, we have that $\xi[X\theta]_p \in \Pi_n$, and thus the result holds with $\zeta' = \zeta$ and $p' = p$.

Main proof: We already know that $X\theta(\Phi(\mathcal{C})\sigma)\downarrow = \beta\theta(\Phi(\mathcal{C})\sigma)\downarrow$. Since $X\theta$ and $\beta\theta$ conforms to $\Phi\theta$ w.r.t. $\mathsf{NoUse}\theta$, we deduce that $X\theta(\Phi(\mathcal{C}')\sigma')\downarrow = \beta\theta(\Phi(\mathcal{C}')\sigma')\downarrow$. Furthermore, we proved that $\zeta\Phi(\mathcal{C})\sigma\downarrow = \beta\theta\Phi(\mathcal{C})\sigma\downarrow$ and $\zeta\Phi(\mathcal{C}')\sigma'\downarrow = \beta\theta\Phi(\mathcal{C}')\sigma'\downarrow$. Hence we deduce that $X\theta(\Phi(\mathcal{C})\sigma)\downarrow = \zeta\Phi(\mathcal{C})\sigma\downarrow$ and $X\theta(\Phi(\mathcal{C}')\sigma')\downarrow = \zeta\Phi(\mathcal{C}')\sigma'\downarrow$. Thanks to Subproperty, we deduce that there exists p' prefix of p and a subterm γ of $X\theta$ such that $\xi(\Phi(\mathcal{C})\sigma)\downarrow = \xi[\gamma]_{p'}\Phi(\mathcal{C})\sigma\downarrow$, $\xi(\Phi(\mathcal{C}')\sigma')\downarrow = \xi[\gamma]_{p'}\Phi(\mathcal{C}')\sigma'\downarrow$ and $\xi[\gamma]_{p'} \in \Pi_n$. But γ is a subterm of $X\theta$ hence is conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta$. Hence, since ζ do not conforms to $\Phi(\mathcal{C})\theta$ w.r.t. $\mathsf{NoUse}(\mathcal{C})\theta$, then $\mathcal{M}(\xi[\gamma]_{p'}, \xi') < \mathcal{M}(\xi, \xi')$. Hence we can apply our inductive hypothesis on $(\xi[\gamma]_{p'}, \xi')$.

$\xi\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ implies $\xi[\gamma]_{p'}\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ and so $\xi[\gamma]_{p'}\Phi(\mathcal{C}')\sigma'\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ which allows us to deduce that $\xi\Phi(\mathcal{C}')\sigma'\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$. Similarly, if $\xi\Phi(\mathcal{C})\sigma\downarrow \in \mathcal{T}(\mathcal{F}_c,\mathcal{N})$ and $\xi\Phi(\mathcal{C})\sigma\downarrow = \xi'\Phi(\mathcal{C})\sigma\downarrow$ then $\xi[\gamma]_{p'}\Phi(\mathcal{C})\sigma\downarrow = \xi'\Phi(\mathcal{C})\sigma\downarrow$. Thus, thanks to our inductive hypothesis, $\xi[\gamma]_{p'}\Phi(\mathcal{C}')\sigma'\downarrow = \xi'\Phi(\mathcal{C}')\sigma'\downarrow$ and so $\xi\Phi(\mathcal{C}')\sigma'\downarrow = \xi'\Phi(\mathcal{C}')\sigma'\downarrow$. The other side of the equivalence can be done symmetrically. Hence the result holds. □

C.7 Proofs of termination

C.7.1 Proofs of termination of each step of Phase 1 of the strategy

In the following section, we will denote $\pi_i(\cdot)$ the function that project the i^{th} of a tuple.

C.7.1.1 Termination of Step a of Phase 1

Lemma 8.12. *Let (M_0, M_0') be a pair of matrices obtained during Step a of Phase* 1 *of the strategy. Let $R(\tilde{p})$ one of the possible instances of* Dest *and* Eq-left-right *applicable on (M_0, M_0'). If we denote (M_1, M_1') the pair of matrices of constraint systems obtained by applying $R(\tilde{p})$ on (M_0, M_0'), then $\mu^m_{1.a}(M_1, M_1') < \mu^m_{1.a}(M_0, M_0')$.*

Proof. Let $\mathcal{C}$ be a constraint system in M_0 or M_1'. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the two constraint systems obtained by application of an instance of Dest or Eq-left-right. We show that:

$$\mu_{1.a}(\mathcal{C}_1) < \mu_{1.a}(\mathcal{C}) \quad \text{and} \quad \mu_{1.a}(\mathcal{C}_2) < \mu_{1.a}(\mathcal{C})$$

We do a case analysis on the rule applied:

Case Eq-left-right(X, ξ)*:* The definition of Eq-left-right implies that there exists $(X, i \rhd u) \in D(\mathcal{C})$ and $(\xi, s \rhd v) \in \Phi(\mathcal{C})$ with $i < s$. Furthermore, we have that $D(\mathcal{C}_1) = D(\mathcal{C})\sigma$ and $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C}_1)\sigma$ where $\sigma = \mathsf{mgu}(u, v)$. But thanks to the property of origination of a constraint system, we know that $vars^1(v) \subseteq vars^1(D(\mathcal{C}))$.

Assume first that σ is different from the identity. In such a case, since $\mathcal{C}_1$ is normalised, we have that $\mathrm{dom}(\sigma) \cap vars^1(D(\mathcal{C}_1)) = \emptyset$ and so $|vars^1(D(\mathcal{C}_1))| < |vars^1(D(\mathcal{C}))|$. Hence we have that $\mu_{1.a}(\mathcal{C}_1) < \mu_{1.a}(\mathcal{C})$.

Assume now that σ is the identity. In such a case, we have that $\pi_1(\mu_{1.a}(\mathcal{C})) = \pi_1(\mu_{1.a}(\mathcal{C}_1))$ and $\pi_2(\mu_{1.a}(\mathcal{C})) = \pi_2(\mu_{1.a}(\mathcal{C}_1)))$. Moreover, the definition of the rule Eq-left-right indicated that $(\xi, s \rhd v) \in \mathsf{NoUse}(\mathcal{C}_1)$ while $(\xi, s \rhd v) \notin \mathsf{NoUse}(\mathcal{C})$. Hence we have that $|\Phi(\mathcal{C}_1)| - |\mathsf{NoUse}(\mathcal{C}_1)| < |\Phi(\mathcal{C})| - |\mathsf{NoUse}(\mathcal{C})|$ and so $\mu_{1.a}(\mathcal{C}_1) < \mu_{1.a}(\mathcal{C})$.

For the constraint system $\mathcal{C}_2$, since no substitution is applied on $\mathcal{C}$, then $\pi_1(\mu_{1.a}(\mathcal{C})) = \pi_1(\mu_{1.a}(\mathcal{C}_2))$ and $\pi_2(\mu_{1.a}(\mathcal{C})) = \pi_2(\mu_{1.a}(\mathcal{C}_2))$. Furthermore, we have that $\mathsf{NoUse}(\mathcal{C}) = \mathsf{NoUse}(\mathcal{C}_2)$ hence $\pi_3(\mu_{1.a}(\mathcal{C})) = \pi_3(\mu_{1.a}(\mathcal{C}_2))$. On the other hand, we have $Eq(\mathcal{C}_2) = Eq(\mathcal{C}) \wedge u \stackrel{?}{\neq} v$. Hence we deduce that $\pi_4(\mu_{1.a}(\mathcal{C}_2) < \pi_4(\mu_{1.a}(\mathcal{C}))$ and so $\mu_{1.a}(\mathcal{C}_2) < \mu_{1.a}(\mathcal{C})$.

Case Dest$(\xi, \ell \to r, s)$*:* By definition there exists i, u such that $(\xi, i \rhd u) \in \Phi(\mathcal{C})$. First of all, we deduce $u \notin \mathcal{X}^1$. Indeed, thanks to Lemma C.31, we know that (M_0, M_0') satisfies Property $\mathsf{PP1Sa}(s)$. Hence if $u \in \mathcal{X}^1$, then either (a) $(\xi, i \rhd u) \in \mathsf{NoUse}(\mathcal{C}')$ if $i < s$, or else (b) there exists $(X, j \stackrel{?}{\vdash} u) \in D(\mathcal{C})$. Case (a) is impossible since the rule Dest$(\xi, \ell \to r, s)$ would not be applicable, and case (b) is also impossible since it would imply that the rule Eq-left-right(X, ξ) is applicable which contradict the strategy that imposes that the rule Eq-left-right are prioritised over the rule Dest.

According to Figure 7.1, $\Phi(\mathcal{C})\sigma \cup \{(\xi', s \rhd w\sigma)\} = \Phi(\mathcal{C}_1)$ and $D(\mathcal{C})\sigma \cup \{X_2, s \stackrel{?}{\vdash} u_2; \ldots; X_n, d \stackrel{?}{\vdash} u_n\} = D(\mathcal{C}_1)$ where $\mathsf{f}(u_1, \ldots, u_n) \to w$ is a fresh renaming of the rewrite rule $\ell \to r$ and $\sigma = \mathsf{mgu}(u_1 \stackrel{?}{=} u)$. But according to the definition of our rewrite rules, $u \notin \mathcal{X}^1$ implies that either $\sigma|_{vars^1(u)}$ is the identity or $\sigma|_{vars^1(u)} = \{x \mapsto \mathsf{pk}(z)\}$ where $x \in vars^1(u)$ and $z \in vars^1(u_1)$. Therefore, along with the fact that $vars^1(u_2, \ldots, u_n) \subseteq vars^1(u_1)$, we deduce that $|vars^1(D(\mathcal{C}_1))| = |vars^1(D(\mathcal{C})|$.

Furthermore, w is a strict subterm of u_1 hence $w\sigma$ is a strict subterm of $u\sigma$. For simplicity, let's denote $\mathcal{F}' = \{\mathsf{aenc}, \mathsf{senc}, \langle\rangle, \mathsf{sign}\}$. For all $(\zeta, p \rhd v) \in \Phi(\mathcal{C})$, since $\sigma|_{vars^1(D)}$ is either identity or $\sigma|_{vars^1(D)} = \{x \to \mathsf{pk}(y)\}$, then have that $|\{t \in st(v\sigma) \mid \mathsf{root}(t) \in \mathcal{F}'\}| = |\{t \in st(v) \mid \mathsf{root}(t) \in \mathcal{F}'\}|$. Furthermore, $w\sigma$ being a strict subterm of $u\sigma$ implies that $|\{t \in st(w\sigma) \mid \mathsf{root}(t) \in \mathcal{F}'\}|$ denoted n_1 is strictly inferior to $|\{t \in st(u\sigma) \mid \mathsf{root}(t) \in \mathcal{F}'\}|$, denoted n_2. At last, Dest$(\xi, \ell \to r, s)$ is useless on $\mathcal{C}_1$ hence there exists a multiset S such that $\pi_2(\mu_{1.a}(\mathcal{C}))) = S \cup \{\{n_2\}\}$ and $\pi_2(\mu_{1.a}(\mathcal{C}_1)) = S \cup \{\{n_1, n_1, n_1, n_1, n_1\}\}$ (there are five rewriting rules available). Thus, $n_1 < n_2$ implies that $\pi_2(\mu_{1.a}(\mathcal{C}_1)) < \pi_2(\mu_{1.a}(\mathcal{C}))$ and so $\mu_{1.a}(\mathcal{C}_1) < \mu_{1.a}(\mathcal{C})$.

For the case of the constraint system $\mathcal{C}_2$, since Dest$(\xi, \ell \to r, s)$ is useless on $\mathcal{C}_2$ and since only non-deducible constraint are added, we trivially have that $|vars^1(D(\mathcal{C}_2))| = vars^1(D(\mathcal{C}))|$ and $\pi_2(\mu_{1.a}(\mathcal{C}_2)) < \pi_2(\mu_{1.a}(\mathcal{C}))$. Hence we have $\mu_{1.a}(\mathcal{C}_2) < \mu_{1.a}(\mathcal{C})$.

We finish the proof by showing that $\mu^m_{1.a}(M_1, M'_1) < \mu^m_{1.a}(M_0, M'_0)$. Each application of DEST or EQ-LEFT-RIGHT are internal rule. Assume that $M_0 = [R_1, \dots R_n]$ and $M'_0 = [R'_1, \dots, R'_n]$ where R_i, R'_i are row matrices. Assume w.l.o.g. that the rule is applied on the first line. By definition of the application of an internal rule, we have $M_1 = [W_1, W_2, R_2, \dots, R_n]$ and $M'_1 = [W'_1, W'_2, R'_2, \dots, R'_n]$ where W_1, W_2 (resp. W'_1, W'_2) are the two row matrices obtained from R_1 (resp. R'_1).

Let $\mathcal{C}$ be constraint system in R_1 (resp. R'_1). Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be the two sons of $\mathcal{C}$ by application of the rule. We know that $\mathcal{C}_1 \in W_1$ (resp. $\mathcal{C}_1 \in W'_1$) and $\mathcal{C}_2 \in W_2$ (resp. $\mathcal{C}_2 \in W'_2$). But since we proved that $\mu_{1.a}(\mathcal{C}_1) < \mu_{1.a}(\mathcal{C})$ and $\mu_{1.a}(\mathcal{C}_2) < \mu_{1.a}(\mathcal{C})$, we deduce that $\mu^m_{1.a}(M_1, M'_1) < \mu^m_{1.a}(M_0, M'_0)$. □

C.7.1.2 Termination of Step b of Phase 1

Lemma 8.13. *Let $\mathcal{C}$ be a well-formed constraint system satisfying the invariant* $\mathsf{InvVarFrame}(s)$. *Let $R(\tilde{p})$ be any instance of any rules except* DEST *and* EQ-LEFT-RIGHT *with support inferior to s. Assume that $R(\tilde{p})$ is strongly applicable on $\mathcal{C}$. Denote $\mathcal{C}_1$ and $\mathcal{C}_2$ the two constraint systems obtained by application of $R(\tilde{p})$ on $\mathcal{C}$. The following property holds :*

$$\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C}) \quad \wedge \quad \mu_{1.b}(\mathcal{C}_2) < \mu_{1.b}(\mathcal{C})$$

Proof. We prove the this lemma by case analysis on the rule $R(\tilde{p})$. We will assume that $\mathcal{C}_1$ (resp. $\mathcal{C}_2$) always corresponds to the son where the guess is positive (resp. negative).

Rule CONS(X, f)*:* We assumed that the rule is strongly applicable on $\mathcal{C}$, hence there exists i, t such that $(X, i \overset{?}{\vdash} t) \in D(\mathcal{C})$ and $t \notin \mathcal{X}^1$ or $t \in \mathcal{X}^1$ and there exists $\mathsf{g} \in \mathcal{F}_c$ such that $\mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$ is in $Er(\mathcal{C})$.

We first focus on $\mathcal{C}_1$. We know that $\mathcal{C}_1$ is normalised. Hence, the rule CONS adds in $D(\mathcal{C}_1)$ the deducible constraints $(X_k, i \overset{?}{\vdash} x_k\sigma)$ where X_k, x_k are fresh, for $k = 1 \dots n$, and $\sigma = \mathsf{mgu}(t, \mathsf{f}(x_1, \dots, x_n))$.

But since X_k are fresh and $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \mathsf{f}(X_1, \dots, X_n)$, we deduce that $(X_k, \mathsf{g}) \notin \pi_1(\mu_{1.b}(\mathcal{C}_1))$, for all g, $k = 1 \dots n$. Furthermore, since for all $(Y, j \overset{?}{\vdash} v) \in D(\mathcal{C})$ other than $(X, i \overset{?}{\vdash} t)$, $(Y, j \overset{?}{\vdash} v\sigma) \in D(\mathcal{C}_1)$, then $(Y, \mathsf{g}) \in \pi_1(\mu_{1.b}(\mathcal{C}_1))$ implies $(Y, \mathsf{g}) \in \pi_1(\mu_{1.b}(\mathcal{C}_1))$. Hence we deduce that $\pi_1(\mu_{1.b}(\mathcal{C}_1)) \leq \pi_1(\mu_{1.b}(\mathcal{C}))$.

If $t \in \mathcal{X}^1$ then we have at least $(X, \mathsf{f}) \in \pi_1(\mathcal{C})$. But since $X \notin vars^2(D(\mathcal{C}_1))$, we deduce that $\pi_1(\mu_{1.b}(\mathcal{C}_1)) < \pi_1(\mu_{1.b}(\mathcal{C}))$. Thus, we conclude that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

Else $t \notin \mathcal{X}^1$: If $\mathsf{root}(t) \neq \mathsf{f}$, we have that $\mathcal{C}_1{\downarrow} = \bot$ hence the result trivially holds. Else $\mathsf{root}(t) = \mathsf{f}$ and so it implies that $\mathsf{root}(t) = \mathsf{f}$ and $\mathrm{dom}(\sigma) = \{x_1, \dots, x_n\}$. Hence $x_k\sigma$ is a strict subterm of t, for $k = 1 \dots n$. Hence we have that $vars^1(D(\mathcal{C}_1)) = vars^1(D(\mathcal{C}))$ and so $\pi_2(\mu_{1.b}(\mathcal{C}_1)) = \pi_2(\mu_{1.b}(\mathcal{C}))$. $t \notin \mathcal{X}^1$ also implies that $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$ and $ND(\mathcal{C}_1) = ND(\mathcal{C})$. Hence, since the applications conditions of DED-ST and EQ-LEFT-LEFT only depends on these two elements, we deduce that $\pi_4(\mu_{1.b}(\mathcal{C}_1)) = \pi_4(\mu_{1.b}(\mathcal{C}))$ and $\pi_3(\mu_{1.b}(\mathcal{C}_1)) = \pi_3(\mu_{1.b}(\mathcal{C}))$. At last, since $\mathrm{dom}(\sigma) = \{x_1, \dots, x_n\}$ and $x_k\sigma$ are strict subterms of t, we can deduce that $\pi_5(\mu_{1.b}(\mathcal{C}_1)) < \pi_5(\mu_{1.b}(\mathcal{C}))$. Thus, we conclude that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

We now focus on $\mathcal{C}_2$. In such a case, the only difference between $\mathcal{C}_2$ and $\mathcal{C}$ is that $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$. Hence we trivially have that $\pi_k(\mu_{1.b}(\mathcal{C}_2)) = \pi_k(\mu_{1.b}(\mathcal{C}))$, for $k = 2 \dots 5$. On the other hand, if $t \in \mathcal{X}^1$ then we have $\pi_1(\mu_{1.b}(\mathcal{C}_2)) < \pi_1(\mu_{1.b}(\mathcal{C}))$ else we have that $\pi_1(\mu_{1.b}(\mathcal{C}_2)) = \pi_1(\mu_{1.b}(\mathcal{C}))$ and $\pi_6(\mu_{1.b}(\mathcal{C}_2)) < \pi_6(\mu_{1.b}(\mathcal{C}))$. Hence, in both cases, we can conclude that $\mu_{1.b}(\mathcal{C}_2) < \mu_{1.b}(\mathcal{C})$.

Rule AXIOM(X, ξ)*:* We assumed that the rule is strongly applicable on $\mathcal{C}$, hence there exists $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$, $(\xi, j \rhd v) \in \Phi(\mathcal{C})$ such that $j \leq i$ and either $u \notin \mathcal{X}^1$ or $t \in \mathcal{X}^1$ and there exists $\mathsf{g} \in \mathcal{F}_c$ such that $Er(\mathcal{C}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$.

We first focus on $\mathcal{C}_1$. We know that $\mathcal{C}_1$ is normalised. Hence, we have that $D(\mathcal{C}_1) = D(\mathcal{C})\sigma \setminus \{X, i \overset{?}{\vdash} u\sigma\}$ where $\sigma = \mathsf{mgu}(u, v)$. Furthermore, we have $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \xi$. Thus, we have that $\pi_1(\mu_{1.b}(\mathcal{C}_1)) \leq \pi_1(\mu_{1.b}(\mathcal{C}))$. We do a case analysis on u and v:

Case $u \in \mathcal{X}^1$: In such a case we have that there exist f such that $(X, \mathsf{f}) \in \pi_1(\mu_{1.b}(\mathcal{C}))$. Since $D(\mathcal{C}_1) = D(\mathcal{C})\sigma \setminus \{X, i \overset{?}{\vdash} u\sigma\}$, we can deduce that $\pi_1(\mu_{1.b}(\mathcal{C}_1)) < \pi_1(\mu_{1.b}(\mathcal{C}))$. Thus we conclude that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

Case $u \notin \mathcal{X}^1$ *and* σ *is the identity:* σ being the identity implies that $u = v$ and so thanks to the origination property of constraint system, we have that $|vars^1(D(\mathcal{C}_1))| = |vars^1(D(\mathcal{C}))|$. Furthermore since $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$, we trivially have that $\pi_4(\mu_{1.b}(\mathcal{C}_1)) = \pi_4(\mu_{1.b}(\mathcal{C}))$ and $\pi_3(\mu_{1.b}(\mathcal{C}_1)) = \pi_3(\mu_{1.b}(\mathcal{C}))$. At last, $u \notin \mathcal{X}^1$ implies that u is either a name or $\mathsf{root}(u) \in \mathcal{F}_c$. Thus $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{X, i \overset{?}{\vdash} u\}$ implies that $\pi_5(\mu_{1.b}(\mathcal{C}_1)) < \pi_5(\mu_{1.b}(\mathcal{C}))$. Thus we conclude that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

Case $u \notin \mathcal{X}^1$ *and* σ *is not the identity:*. By the property of origination of a constraint system, we know that $vars^1(v) \subseteq vars^1(D(\mathcal{C}))$. Hence we have that $|vars^1(D(\mathcal{C})\sigma)| < |vars^1(D(\mathcal{C}))|$. We proved that $D(\mathcal{C}_1) \subseteq D(\mathcal{C})\sigma$, therefore we can deduce that $\pi_2(\mu_{1.b}(\mathcal{C}_1)) < \pi_2(\mu_{1.b}(\mathcal{C}_1))$. Thus we conclude that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

We now focus on $\mathcal{C}_2$. In such a case, the only difference between $\mathcal{C}_2$ and $\mathcal{C}$ is that $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge X \overset{?}{\neq} \xi$. Hence we trivially have that $\pi_k(\mu_{1.b}(\mathcal{C}_2)) = \pi_k(\mu_{1.b}(\mathcal{C}))$, for $k = 1 \dots 6$. Moreover, $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge X \overset{?}{\neq} \xi$ also implies that $\pi_7(\mu_{1.b}(\mathcal{C}_2)) < \pi_7(\mu_{1.b}(\mathcal{C}))$. Thus we conclude that $\mu_{1.b}(\mathcal{C}_2) < \mu_{1.b}(\mathcal{C})$.

Rule Eq-left-left(ξ_1, ξ_2)*:* We assumed that the rule is applicable on $\mathcal{C}$ (for Phase 1), hence there exists $(\xi_1, i_1 \overset{?}{\vdash} u_1) \in \Phi(\mathcal{C})$, $(\xi_2, u_2 \rhd u_2) \in \Phi(\mathcal{C})$.

We first focus on $\mathcal{C}_1$. We know that $\mathcal{C}_1$ is normalised. Hence, we have that $D(\mathcal{C}_1) = D(\mathcal{C})\sigma$ where $\sigma = \mathsf{mgu}(u_1, u_2)$. Thus, we have that $\pi_1(\mu_{1.b}(\mathcal{C}_1)) \leq \pi_1(\mu_{1.b}(\mathcal{C}))$.

By the origination property of a constraint system, we know that $vars^1(u_1, u_2) \subseteq vars^1(D(\mathcal{C}))$. Hence we deduce that $vars^1(D(\mathcal{C}_1)) \subseteq vars^1(D(\mathcal{C}))$. But if σ is not the identity then we have that $|vars^1(D(\mathcal{C}_1))| < |vars^1(D(\mathcal{C}))|$. Thus we deduce that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

Else if σ is the identity, then we have that $vars^1(D(\mathcal{C}_1)) = vars^1(D(\mathcal{C}))$ and so $\pi_2(\mu_{1.b}(\mathcal{C}_1)) \leq \pi_2(\mu_{1.b}(\mathcal{C}))$. Furthermore, σ being the identity also implies that $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$. Thus we deduce that $\pi_3(\mu_{1.b}(\mathcal{C}_1)) = \pi_3(\mu_{1.b}(\mathcal{C}))$. At last, we are focused on the case of the application of the rule Eq-left-left, therefore we trivially have that $\pi_4(\mu_{1.b}(\mathcal{C}_1)) < \pi_4(\mu_{1.b}(\mathcal{C}))$. Thus we conclude that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

Rule Eq-right-right(X, Y)*:* We assumed that the rule is strongly applicable on $\mathcal{C}$ hence there exists $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C})$ and $(Y, j \overset{?}{\vdash} v) \in D(\mathcal{C})$ such that $u = v \in \mathcal{X}^1$. In such a case, we first have that $\mathcal{C}_2 = \bot$ since $Er(\mathcal{C}_2) \vDash u \overset{?}{\neq} u$ yields $\bot$ by normalisation. Thus we deduce that $\mu_{1.b}(\mathcal{C}_2) < \mu_{1.b}(\mathcal{C})$.

We know focus on $\mathcal{C}_1$. We have that $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{Xi \overset{?}{\vdash} u\}$. Hence we deduce that $\pi_k(\mu_{1.b}(\mathcal{C}_1)) \leq \pi_k(\mu_{1.b}(\mathcal{C}))$, for $k = 1, 6, 7$ and that $\pi_8(\mu_{1.b}(\mathcal{C}_1)) < \pi_8(\mu_{1.b}(\mathcal{C}))$. Furthermore, since $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$, we deduce that $\pi_k(\mu_{1.b}(\mathcal{C}_1)) = \pi_k(\mu_{1.b}(\mathcal{C}))$, for $k = 2, 3, 4, 5$. Thus we conclude that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

Rule Ded-st(ξ, f, i)*:* We assumed that the rule is applicable on $\mathcal{C}$ hence there exists $(\xi, i \overset{?}{\vdash} u) \in \Phi(\mathcal{C})$ such that $(\xi, i \overset{?}{\vdash} u) \notin \mathsf{NoUse}(\mathcal{C})$. We know that $\mathcal{C}$ satisfies the invariant $\mathsf{InvVarFrame}(s)$ hence we can deduce that $u \notin \mathcal{X}^1$.

We first focus on $\mathcal{C}_1$: If $\mathsf{root}(u) \neq \mathsf{f}$ then we have that $\mathcal{C}_1 = \bot$ since $\mathcal{C}_1$ is normalised (otherwise, we would have $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge u \overset{?}{=} \mathsf{f}(x_1, \dots, x_n)$ which is reduced into $\bot$ by the normalisation rule (Nins1)). Thus we deduce that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

Else $\mathsf{root}(u) = \mathsf{f}$. In such a case, we have $Er(\mathcal{C}_1) = Er(\mathcal{C})$, $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$ and $D(\mathcal{C}_1) = D(\mathcal{C}) \cup \{X_k, i \overset{?}{\vdash} x_k\sigma\}_{k=1\ldots n}$ such that $x_k\sigma$ is a strict subterm of u and X_k are fresh, for $k = 1 \ldots n$ and $\sigma = \mathsf{mgu}(u \overset{?}{=} \mathsf{f}(x_1, \ldots, x_n))$. X_k being fresh implies that there do not exist $\mathsf{g} \in \mathcal{F}_c$ such that $Er(\mathcal{C}_1) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$. Hence we deduce that $\pi_1(\mu_{1.b}(\mathcal{C}_1)) = \pi_1(\mu_{1.b}(\mathcal{C}))$. Furthermore, thanks to the origination property of a constraint system, we know that $vars^1(u) \subseteq vars^1(D(\mathcal{C}))$, thus thanks to $x_k\sigma$ being strict subterm of u, we deduce that $\pi_2(\mu_{1.b}(\mathcal{C}_1)) \leq \pi_2(\mu_{1.b}(\mathcal{C}))$. At last, we are focused on the case of the application of the rule Ded-st, therefore we trivially have that $\pi_3(\mu_{1.b}(\mathcal{C}_1)) < \pi_3(\mu_{1.b}(\mathcal{C}))$. Thus we conclude that $\mu_{1.b}(\mathcal{C}_1) < \mu_{1.b}(\mathcal{C})$.

We now focus on $\mathcal{C}_2$: In such a case, we have that $Er(\mathcal{C}_2) = Er(\mathcal{C})$, $\Phi(\mathcal{C}_2) = \Phi(\mathcal{C})$ and $D(\mathcal{C}_2) = D(\mathcal{C})$. Hence we trivially have that $\pi_k(\mu_{1.b}(\mathcal{C}_2)) \leq \pi_k(\mu_{1.b}(\mathcal{C}))$, for $k = 1, 2$. At last, we are focused on the case of the application of the rule Ded-st, therefore we trivially have that $\pi_3(\mu_{1.b}(\mathcal{C}_2)) < \pi_3(\mu_{1.b}(\mathcal{C}))$ and so we conclude that $\mu_{1.b}(\mathcal{C}_2) < \mu_{1.b}(\mathcal{C})$. □

C.7.1.3 Termination of Step c of Phase 1

Lemma C.53. *Let (M, M') be a pair of matrices of constraint systems obtained during the Step c of Phase 1 of the the strategy with parameters s and k. Assume w.l.o.g. that the k^{th} column of (M, M') is the k^{th} column of M. Assume that the rule* Cons(X_0, f) *can be applied externally on (M, M') such that $(X_0, j_0 \overset{?}{\vdash} u_0) \in D(M_{i_0,k})$, $i_0 \in \{1, \ldots, n\}$ and*

$$\mathcal{L}^1_{M_{i_0,k}}(X_0, j_0 \overset{?}{\vdash} u_0) = \min\left\{ \mathcal{L}^1_{M_{i,k}}(Z, \ell \overset{?}{\vdash} v) \;\middle|\; \begin{array}{l} i \in \{1, \ldots, n\}, (Z, \ell \overset{?}{\vdash} v) \in D(M_{i,k}), \\ vars(v) \cap \mathsf{X}^1(M_{i,k}) \neq \emptyset, Z \in S_2 \end{array} \right\}$$

For all term u, for all $i \in \{1, \ldots, n\}$, if $(X_0, j_0 \overset{?}{\vdash} u) \in D(M_{i,k})$ and $u \in \mathcal{X}^1$ then $u \notin \mathsf{X}^1(M_{i,k})$.

Proof. For simplicity, we will denote $\mathcal{C} = M_{i_0,k}$ and $\mathcal{C}' = M_{i,k}$. We know that $(X_0, j_0 \overset{?}{\vdash} u_0) \in D(\mathcal{C})$. Let $(X_0, j_0 \overset{?}{\vdash} u) \in D(\mathcal{C}')$ such that $u \in \mathcal{X}^1$. Assume that $u \in \mathsf{X}^1(\mathcal{C}')$.

In such a case, we have that $\mathcal{L}^1_{\mathcal{C}'}(X_0, j_0 \overset{?}{\vdash} u) = (j_0, \epsilon)$. But we know that $(X_0, j_0 \overset{?}{\vdash} u_0) \in D(\mathcal{C})$ and by the minimality of $\mathcal{L}^1_{\mathcal{C}}(X_0, j_0 \overset{?}{\vdash} u_0)$, we deduce that $\mathcal{L}^1_{\mathcal{C}}(X_0, j_0 \overset{?}{\vdash} u_0) \leq (j_0, \epsilon)$. Hence we deduce that $u_0 \in \mathcal{X}^1$ and $u_0 \in \mathsf{X}^1(\mathcal{C})$.

But $u_0 \in \mathsf{X}^1(\mathcal{C})$ also implies that there exists $(Y, j \overset{?}{\vdash} u_0) \in D(\mathcal{C})$ such that $Y \notin S_2$. Moreover, the minimality of $\mathcal{L}^1_{\mathcal{C}}(X_0, j_0 \overset{?}{\vdash} u_0)$ also implies that there no deducible constraint $(Y', j' \overset{?}{\vdash} v) \in D(\mathcal{C})$ such that $u_0 \in vars^1(v)$ and $j' < j_0$. Hence, thanks to $\mathcal{C}$ being well-formed (Definition 8.2, item 10) we deduce that $j_0 < j$. Consider the pair of matrices of constraint system system (M_1, M'_1) ancestor of (M, M') such that (M_1, M'_1) is obtained at the end of Step b with parameters s and k. Thanks to Lemma C.31, we know that (M_1, M'_1) satisfies $\mathsf{PP1SbE}(s, k)$. Let $\mathcal{C}_1$ be the constraint system in M_1 ancestor of $\mathcal{C}$. Since no internal rule are applied other than Eq-right-right during step c, we deduce that $(Y', j' \overset{?}{\vdash} v') \in D(\mathcal{C}_1)$ for some v' and so thanks to Property $\mathsf{PP1SbE}(s, k)$, we deduce that for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}_1) \nvdash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$. Once again thanks to the fact that no internal rule are applied other than Eq-right-right during step c, we can easily show that for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \nvdash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$.

Furthermore, Property $\mathsf{PP1SbE}(s, k)$ indicates that for all $X \in vars^2(D(\mathcal{C}_1))$, for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}_1) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$. Hence, if if there exists $\mathsf{f} \in \mathcal{F}_c$ such that $Er(\mathcal{C}) \vDash \mathsf{root}(X_0) \overset{?}{\neq} \mathsf{f}$, it would implies that there exists $\mathcal{C}''$ such that $\mathcal{C}_1 \to^* \mathcal{C}'' \to^* \mathcal{C}$, $\mathcal{C}''$ is obtained during step c, $(X_0, j_0 \overset{?}{\vdash} u_0) \in D(\mathcal{C}'')$, $(Y, j \overset{?}{\vdash} u_0) \in D(\mathcal{C}'')$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \nvdash \mathsf{root}(X_0) \overset{?}{\neq} \mathsf{f}$. But in such a case, the rule Eq-right-right should have been applied according to the strategy and

so we would have that $(Y, j \overset{?}{\vdash} x_0) \notin D(\mathcal{C})$ which is a contradiction. Hence we conclude that $u \notin \mathsf{X}^1(\mathcal{C}')$. □

Lemma 8.14. *Let (M, M') be a pair of matrices of constraint systems obtained during the Step c of Phase 1 of the strategy with parameter s and k respectively for support and column. Let $R(\tilde{p})$ be the next possible rule applicable according to step c of the strategy and let (M_1, M_1') and (M_2, M_2') be the two pairs of matrices of constraint systems obtained by application of $R(\tilde{p})$ on (M, M') (in the case of $R(\tilde{p})$ being the rule* Eq-right-right*, there is only one pair of constraint system since* Eq-right-right *is applied internally). We have that:*

$$\mu^k_{1.c}(M_1, M_1') < \mu^k_{1.c}(M, M') \quad \wedge \quad \mu^k_{1.c}(M_2, M_2') < \mu^k_{1.c}(M, M')$$

Proof. We prove the result by case analysis on the rule $R(\tilde{p})$:

Rule Eq-right-right(X, Y)*:* Since this rule is applied internally, there exists $\mathcal{C}$ in the k^{th} column of (M, M') such that Eq-right-right(X, Y) is strongly applicable on $\mathcal{C}$. Furthermore, we deduce that $X \notin S_2(\mathcal{C})$ and $Y \in S_2(\mathcal{C})$. Assume that $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ and $(Y, j \overset{?}{\vdash} x) \in D(\mathcal{C})$. According to the strong application condition of Eq-right-right(X, Y) in case of internal rule for phase 1, we have that $j < i$.

By normalisation, we have that $\mathcal{C}_2 = \bot$ and so we trivially have that $\mu_{1.c}(\mathcal{C}_2) < \mu_{1.c}(\mathcal{C})$.

We focus on $\mathcal{C}_1$: Since $Y \in S_2(\mathcal{C})$ and $x \in \mathcal{X}^1$, we can deduce that $\mathcal{L}^1_{\mathcal{C}}(Y, j \overset{?}{\vdash} x) = (j, 0)$. But $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{X, i \overset{?}{\vdash} x\}$.

If there is no $(Z, \ell \overset{?}{\vdash} x) \in D(\mathcal{C})$ such that $Z \neq X$ and $Z \notin S_2(\mathcal{C})$, we have that $x \notin \mathsf{X}^1(\mathcal{C}_1)$. Hence, we can deduce that $\mathcal{L}^1_{\mathcal{C}_1}(Y, j \overset{?}{\vdash} x)$ is not defined and so $\pi_1(\mu_{1.c}(\mathcal{C}_1)) < \pi_1(\mu_{1.c}(\mathcal{C}))$. Thus we deduce that $\mu_{1.c}(\mathcal{C}_1) < \mu_{1.c}(\mathcal{C})$

Else there exists $(Z, \ell \overset{?}{\vdash} x) \in D(\mathcal{C})$ such that $Z \neq X$ and $Z \notin S_2(\mathcal{C})$. Thus $\mathsf{X}^1(\mathcal{C}_1) = \mathsf{X}^1(\mathcal{C})$. Thus, with $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{X, i \overset{?}{\vdash} x\}$, we deduce that $\pi_1(\mu_{1.c}(\mathcal{C}_1)) = \pi_1(\mu_{1.c}(\mathcal{C}))$. Furthermore, we $(X, Z) \in \pi_2(\mu_{1.c}(\mathcal{C}))$ while $(X, Z) \notin \pi_2(\mu_{1.c}(\mathcal{C}_1))$. Hence we easily deduce that $\pi_2(\mu_{1.c}(\mathcal{C}_1)) < \pi_2(\mu_{1.c}(\mathcal{C}))$ and so that $\mu_{1.c}(\mathcal{C}_1) < \mu_{1.c}(\mathcal{C})$.

Since we proved that $\mu_{1.c}(\mathcal{C}_1) < \mu_{1.c}(\mathcal{C})$ and $\mu_{1.c}(\mathcal{C}_2) < \mu_{1.c}(\mathcal{C})$ and since the rule is applied internally, we deduce that $\mu^k_{1.c}(M_1, M_1') < \mu^k_{1.c}(M, M')$.

Rue Cons(X, f)*:* The rule is applied externally. Hence we will show that for all constraint system $\mathcal{C}$ in the k^{th} column, we have that $\mu_{1.c}(\mathcal{C}_1) \leq \mu_{1.c}(\mathcal{C})$ and $\mu_{1.c}(\mathcal{C}_2) \leq \mu_{1.c}(\mathcal{C})$ where $\mathcal{C}_1$ and $\mathcal{C}_2$ are the constraint system obtained by applying the rule on $\mathcal{C}$. Furthermore, by the definition of the strategy, we know that there exists $\mathcal{C}_0$ such that $\mathcal{L}^1_{\mathcal{C}_0}(X, i_0 \overset{?}{\vdash} u_0)$ is minimal. Thus we will show that in the case of $\mathcal{C}_0$, $\mu_{1.c}(\mathcal{C}_1) < \mu_{1.c}(\mathcal{C}_0)$ and $\mu_{1.c}(\mathcal{C}_2) < \mu_{1.c}(\mathcal{C}_0)$, where $\mathcal{C}_1$ and $\mathcal{C}_2$ are the constraint system obtained by applying the rule on $\mathcal{C}_0$.

We first focus on $\mathcal{C}_1$: Thanks to Lemma C.53, we know that for all $\mathcal{C}$ in the k^{th} column, for all $i \in \mathbb{N}$, for all term u, $(X, i_0 \overset{?}{\vdash} u) \in D(\mathcal{C})$ and $u \in \mathcal{X}^1$ implies $u \notin \mathsf{X}^1(\mathcal{C})$.

Case $u \in \mathcal{X}^1$: By normalisation, $D(\mathcal{C}_1) = (D(\mathcal{C})\{X, i_0 \overset{?}{\vdash} u\})\sigma \cup \{X_j, i_0 \overset{?}{\vdash} x_j\}_{j=1\ldots n}$ where $\sigma = \{u \to \mathsf{f}(x_1, \ldots, x_n)\}$ and x_j are fresh variables and $u \notin \mathsf{X}^1(\mathcal{C})$. Hence we deduce that for all $(Y, j \overset{?}{\vdash} v) \in D(\mathcal{C}_1)$, for all p, if $v|_p \in \mathsf{X}^1(\mathcal{C}_1)$ then $v|_p \in \mathcal{X}^1$, $v|_p = v|_p\sigma$ and $(Y, j \overset{?}{\vdash} v) \in D(\mathcal{C})$. Thus we deduce that $v|_p \in \mathsf{X}^1(\mathcal{C})$ and so $\pi_1(\mu_{1.c}(\mathcal{C}_1)) = \pi_1(\mu_{1.c}(\mathcal{C}))$. Since $X_j \in S_2(\mathcal{C}_1)$, for $j = 1 \ldots n$, we deduce that $\pi_2(\mu_{1.c}(\mathcal{C}_1)) = \pi_2(\mu_{1.c}(\mathcal{C}))$. At last, $x_j \notin \mathsf{X}^1(\mathcal{C}_1)$, for $j = 1 \ldots n$ also implies that $\pi_i(\mu_{1.c}(\mathcal{C}_1)) = \pi_i(\mu_{1.c}(\mathcal{C}))$, for $i = 3, 4$ and so we deduce that $\mu_{1.c}(\mathcal{C}_1) = \mu_{1.c}(\mathcal{C})$.

Case $u \notin \mathcal{X}^1$ and $vars^1(u) \cap \mathsf{X}^1(\mathcal{C}) = \emptyset$: In such a case, we have that $D(\mathcal{C}_1) = (D(\mathcal{C}) \setminus \{X, i_0 \overset{?}{\vdash} u\}) \cup \{X_j, i_0 \overset{?}{\vdash} x_j\sigma\}_{j=1\ldots n}$ where $x_j\sigma$ is a strict subterm of u, for $j = 1 \ldots n$ and

$\sigma = \mathsf{mgu}(u \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n))$. Thus, $vars^1(u) \cap \mathsf{X}^1(\mathcal{C}) = \emptyset$ implies that $vars^1(x_j\sigma) \cap \mathsf{X}^1(\mathcal{C}) = \emptyset$ for $j = 1 \ldots n$. Therefore, we have that $\mathcal{L}^1_{\mathcal{C}_1}(X_j, i_0 \stackrel{?}{\vdash} x_j\sigma)$ does not exist, for $j = 1 \ldots n$ which implies that $\pi_j(\mu_{1.c}(\mathcal{C}_1)) = \pi_j(\mu_{1.c}(\mathcal{C}))$ for $j = 1, 3, 4$. At last, since $X_j \in S_2(\mathcal{C}_1)$, for $j = 1 \ldots n$, we deduce that $\pi_2(\mu_{1.c}(\mathcal{C}_1)) = \pi_2(\mu_{1.c}(\mathcal{C}))$ and so we conclude that $\mu_{1.c}(\mathcal{C}_1) = \mu_{1.c}(\mathcal{C})$.

Case $u \notin \mathcal{X}^1$ and $vars^1(u) \cap \mathsf{X}^1(\mathcal{C}) \neq \emptyset$: In such a case, we still have that $D(\mathcal{C}_1) = (D(\mathcal{C}) \setminus \{X, i_0 \stackrel{?}{\vdash} u\}) \cup \{X_j, i_0 \stackrel{?}{\vdash} x_j\sigma\}_{j=1\ldots n}$ where $\sigma = \mathsf{mgu}(u, \mathsf{f}(x_1, \ldots, x_n))$. Since $u \notin \mathcal{X}^1$, we can deduce that for all $j \in \{1, \ldots, n\}$, for all p, if $x_j\sigma|_p \in \mathsf{X}^1(\mathcal{C}_1)$ then $u|_{j \cdot p} = x_j\sigma|_p \in \mathsf{X}^1(\mathcal{C})$. With the fact that $\mathsf{ind}_{\mathcal{C}}(x) = \mathsf{ind}_{\mathcal{C}_1}(x)$ for all $x \in vars^1(D(\mathcal{C}))$, we can deduce that $(\mathsf{ind}_{\mathcal{C}_1}(x_j\sigma|_p), p) < (\mathsf{ind}_{\mathcal{C}}(x_j\sigma|_p), j \cdot p)$, for all $j \in \{1, \ldots, n\}$. Thus we can deduce that $\pi_1(\mu_{1.c}(\mathcal{C}_1)) < \pi_1(\mu_{1.c}(\mathcal{C}))$ and so $\mu_{1.c}(\mathcal{C}_1) < \mu_{1.c}(\mathcal{C})$.

Note that by the application condition on the rule $\textsc{Cons}(X, \mathsf{f})$ for step c, we can deduce that $u_0 \notin \mathcal{X}^1$ and $vars^1(u_0) \cap \mathsf{X}^1(\mathcal{C}_0) \neq \emptyset$. Thus we have $\mu_{1.c}(\mathcal{C}_1) < \mu_{1.c}(\mathcal{C}_0)$

We now focus on $\mathcal{C}_2$: In such a case we have that $D(\mathcal{C}_2) = D(\mathcal{C})$. Hence we trivially have that $\pi_i(\mu_{1.c}(\mathcal{C}_2)) = \pi_i(\mu_{1.c}(\mathcal{C}))$, for $i = 1, 2$. On there other hand, since $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$, we have that $\pi_3(\mu_{1.c}(\mathcal{C}_2)) \leq \pi_3(\mu_{1.c}(\mathcal{C}))$ which allows us to conclude that $\mu_{1.c}(\mathcal{C}_2) \leq \mu_{1.c}(\mathcal{C})$.

Note that by the application condition on the rule $\textsc{Cons}(X, \mathsf{f})$ for step c, we can deduce that $u_0 \notin \mathcal{X}^1$ and $vars^1(u_0) \cap \mathsf{X}^1(\mathcal{C}_0) \neq \emptyset$. Thus we have $\pi_3(\mu_{1.c}(\mathcal{C}_2)) < \pi_3(\mu_{1.c}(\mathcal{C}_0))$ which allows us to conclude that $\mu_{1.c}(\mathcal{C}_2) < \mu_{1.c}(\mathcal{C}_0)$.

Rue $\textsc{Axiom}(X, \mathsf{path})$*:* The rule is applied externally. Hence, similarly to the case of the rule Cons, we will show that for all constraint system $\mathcal{C}$ in the k^{th} column, we have that $\mu_{1.c}(\mathcal{C}_1) \leq \mu_{1.c}(\mathcal{C})$ and $\mu_{1.c}(\mathcal{C}_2) \leq \mu_{1.c}(\mathcal{C})$ where $\mathcal{C}_1$ and $\mathcal{C}_2$ are the constraint system obtained by applying the rule on $\mathcal{C}$. Furthermore, we will show that in the case of $\mathcal{C}_0$, $\mu_{1.c}(\mathcal{C}_1) < \mu_{1.c}(\mathcal{C}_0)$ and $\mu_{1.c}(\mathcal{C}_2) < \mu_{1.c}(\mathcal{C}_0)$, where $\mathcal{C}_1$ and $\mathcal{C}_2$ are the constraint system obtained by applying the rule on $\mathcal{C}_0$.

By definition of the rule, we know that there exists $(X, i \stackrel{?}{\vdash} u) \in D(\mathcal{C})$ and $(\xi, j \stackrel{?}{\vdash} v) \in \Phi(\mathcal{C})$ such that $j \leq i$.

We first focus on $\mathcal{C}_1$: By definition of the rule, we have that $D(\mathcal{C}_1) = D(\mathcal{C})\sigma \setminus \{X, i \stackrel{?}{\vdash} u\sigma\}$ where $\sigma = \mathsf{mgu}(u, v)$. Furthermore, we have that $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})\sigma$.

Let $x \in vars^1(v)$. By the property of origination of a constraint system, we deduce that $\mathsf{ind}_{\mathcal{C}}(x) < i$. Let $(Y, \ell \stackrel{?}{\vdash} t) \in D(\mathcal{C})$ such that $x \in vars^1 t$ and $\mathsf{ind}_{\mathcal{C}}(x) = \ell$. By the minimality of $\mathcal{L}^1_{\mathcal{C}_0}(X, i \stackrel{?}{\vdash} u_0)$, we deduce that $x \notin \mathsf{X}^1(\mathcal{C})$.

Let $x \in \mathsf{X}^1(\mathcal{C}) \cap vars^1(u)$. If $x \in \mathrm{img}(\sigma)$ then we have that $x \in vars^1(v\sigma)$. But by the property of origination of a constraint system, we deduce that there exists $(Y, \ell \stackrel{?}{\vdash} t) \in D(\mathcal{C}_1)$ such that $\ell < j \leq i$ and $x \in vars^1(t)$. Moreover, $x \in \mathsf{X}^1(\mathcal{C})$ implies that for all $(Z, m \stackrel{?}{\vdash} w) \in D(\mathcal{C})$, $x \in vars^1(w)$ implies that $\mathcal{L}^1_{\mathcal{C}}(Z, m \stackrel{?}{\vdash} w)$ exists. But by minimality of $\mathcal{L}^1_{\mathcal{C}_0}(X, i \stackrel{?}{\vdash} u_0)$, we deduce that $i \leq m$. Hence, we deduce that $\mathsf{ind}_{\mathcal{C}}(x) = i$. Since we already show that $\mathsf{ind}_{\mathcal{C}_1}(x) \leq \ell < j \leq i$, we deduce that $\mathsf{ind}_{\mathcal{C}_1}(x) < \mathsf{ind}_{\mathcal{C}}(x)$.

Let $(Y, \ell \stackrel{?}{\vdash} t\sigma) \in D(\mathcal{C}_1)$, let p a position such that $t\sigma|_p \in \mathsf{X}^1(\mathcal{C}_1)$. We distinguish two cases:

— *Case 1, $t|_p = t\sigma|_p$ and $t|_p \notin vars^1(\sigma)$:* In such a case, it implies that $t|_p \in \mathsf{X}^1(\mathcal{C})$. Furthermore, we deduce that $(\mathsf{ind}_{\mathcal{C}}(t|_p) = \mathsf{ind}_{\mathcal{C}_1}(t|_p)$. Hence we have that $\{\{(\mathsf{ind}_{\mathcal{C}_1}(t|_p), p') \mid (Z, \ell' \stackrel{?}{\vdash} t') \in D(\mathcal{C}_1) \wedge t'|_{p'} = t|_p\}\} = \{\{(\mathsf{ind}_{\mathcal{C}}(t|_p), p') \mid (Z, \ell' \stackrel{?}{\vdash} t') \in D(\mathcal{C}) \wedge t'|_{p'} = t|_p\}\}$

— *Case 2, $t\sigma|_p \in \mathrm{img}(\sigma)$:* We have shown that in such a case, $\mathsf{ind}_{\mathcal{C}_1}(t\sigma|_p) < \mathsf{ind}_{\mathcal{C}}(t\sigma|_p)$. Since $t\sigma|_p \in vars^1(u)$, we deduce that $\{\{(\mathsf{ind}_{\mathcal{C}_1}(t\sigma|_p), p') \mid (Z, \ell' \stackrel{?}{\vdash} t') \in D(\mathcal{C}_1) \wedge t'|_{p'} = t\sigma|_p\}\} < \{\{(\mathsf{ind}_{\mathcal{C}}(t\sigma|_p), p') \mid (Z, \ell' \stackrel{?}{\vdash} t') \in D(\mathcal{C}) \wedge t'|_{p'} = t\sigma|_p\}\}$.

Hence we deduce that $\pi_1(\mu_{1.c}(\mathcal{C}_1)) \leq \pi_1(\mu_{1.c}(\mathcal{C}))$ and if $vars^1(u) \cap \mathsf{X}^1(\mathcal{C}) \neq \emptyset$ then $\pi_1(\mu_{1.c}(\mathcal{C}_1)) < \pi_1(\mu_{1.c}(\mathcal{C}))$.

At last, since $D(\mathcal{C}_1) = D(\mathcal{C})\sigma \setminus \{X, i \overset{?}{\vdash} u\sigma\}$ and $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \xi$, we deduce that $\pi_i(\mu_{1.c}(\mathcal{C}_1)) \leq \pi_i(\mu_{1.c}(\mathcal{C}))$, for $i = 2, 3, 4$. Thus we conclude that $\mu_{1.c}(\mathcal{C}_1) \leq \mu_{1.c}(\mathcal{C})$.

Note that $vars^1(u_0) \cap \mathsf{X}^1(\mathcal{C}_0) \neq \emptyset$. Hence we have that $\pi_1(\mu_{1.c}(\mathcal{C}_1)) < \pi_1(\mu_{1.c}(\mathcal{C}_0))$ and so $\mu_{1.c}(\mathcal{C}_1) < \mu_{1.c}(\mathcal{C}_0)$.

We now focus on $\mathcal{C}_2$: In such a case, we have that $D(\mathcal{C}_2) = D(\mathcal{C})$ and $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge X \overset{?}{\neq} \xi$. Hence we trivially have that $\pi_i(\mu_{1.c}(\mathcal{C}_2)) = \pi_i(\mu_{1.c}(\mathcal{C}))$, for $i = 1, 2, 3$. Furthermore, we also have that $\pi_4(\mu_{1.c}(\mathcal{C}_2)) = \pi_4(\mu_{1.c}(\mathcal{C}))$ if $vars^1(u) \cap \mathsf{X}^1(\mathcal{C}) = \emptyset$, else $\pi_4(\mu_{1.c}(\mathcal{C}_2)) < \pi_4(\mu_{1.c}(\mathcal{C}))$.

We conclude that $\mu_{1.c}(\mathcal{C}_2) \leq \mu_{1.c}(\mathcal{C})$ and since $vars^1(u_0) \cap \mathsf{X}^1(\mathcal{C}_0) \neq \emptyset$, we also conclude that $\mu_{1.c}(\mathcal{C}_2) < \mu_{1.c}(\mathcal{C}_0)$. □

C.7.1.4 Termination of the cycle of steps b and c of Phase 1

Lemma C.54. *Let $\mathcal{C}$ be a well-formed constraint system. Let $R(\tilde{p})$ be an instance of one of the following rule :* Cons, Axiom *or* Eq-right-right. *Let $\mathcal{C}_1, \mathcal{C}_2$ be the two constraint systems obtained by application of $R(\tilde{p})$ on $\mathcal{C}$. For all $i \in \{1, 2\}$, for all $u \in \mathcal{T}(\mathcal{F}_c, \mathcal{N} \cup \mathcal{X}^1)$, if $vars^1(u) \subseteq vars^1(D(\mathcal{C}))$ then $\mathsf{ind}_{\mathcal{C}_i}(u\sigma_i) \leq \mathsf{ind}_{\mathcal{C}}(u)$ where $\sigma_i = \mathsf{mgu}(Eq(\mathcal{C}_i))$.*

Proof. We do a case analysis on the rule applied: First of all, according to the definition of the rule in Figure 7.1 and 7.2, we have that $\mathsf{mgu}(Eq(\mathcal{C}_2)) = \mathsf{mgu}(Eq(\mathcal{C}))$ and $D(\mathcal{C}_2) = D(\mathcal{C})$ for all the rules considered. Hence we have that $u\sigma_2 = u$ and $\mathsf{ind}_{\mathcal{C}_2}(u) = \mathsf{ind}_{\mathcal{C}}(u)$. Thus the result holds.

Rule Cons(X, f)*:* Let $(X, i \overset{?}{\vdash} t) \in D(\mathcal{C})$. According to Figure 7.1, we have that $D(\mathcal{C}_1) = (D(\mathcal{C})\sigma \setminus \{X, i \overset{?}{\vdash} t\sigma\}) \cup \{X_k, i \overset{?}{\vdash} x_k\}_{k=1..n}$, where x_k, X_k are fresh variables for $k = 1 \dots n$ and $\sigma = \mathsf{mgu}(t, \mathsf{f}(x_1, \dots, x_n))$.

If $t \notin \mathcal{X}^1$ then it implies that $\mathrm{dom}(\sigma) = \{x_1, \dots, x_n\}$, $D(\mathcal{C})\sigma = D(\mathcal{C})$ and $vars^1(D(\mathcal{C})) = vars^1(D(\mathcal{C}_1))$. Hence we deduce that $u\sigma_1 = u$ and $\mathsf{ind}_{\mathcal{C}_1}(u) = \mathsf{ind}_{\mathcal{C}}(u)$ and so $\mathsf{ind}_{\mathcal{C}_1}(u\sigma_1) = \mathsf{ind}_{\mathcal{C}}(u)$.

Else $t \in \mathcal{X}^1$. In such a case, $\sigma = \{t \mapsto \mathsf{f}(x_1, \dots, x_n)\}$. For all $x \in vars^1(D(C_1)) \setminus \{x_1, \dots, x_n\}$, $\mathsf{ind}_{\mathcal{C}_1}(x) = \mathsf{ind}_{\mathcal{C}}(x)$. Assume now that $\mathsf{ind}_{\mathcal{C}}(t) < i$ thus there exists $(Y, j \overset{?}{\vdash} v) \in D(\mathcal{C})$ such that $t \in vars^1(v)$ and $\mathsf{ind}_{\mathcal{C}}(t) = j$. But it implies that $(Y, j \overset{?}{\vdash} v\sigma) \in D(\mathcal{C}_1)$. On the other hand, if $\mathsf{ind}_{\mathcal{C}}(t) = i$ we know that $\{X_k, i \overset{?}{\vdash} x_k\}_{k=1..n} \subseteq D(\mathcal{C}_1)$. Hence we deduce that for all $k \in \{1, \dots, n\}$, $\mathsf{ind}_{\mathcal{C}_1}(x_k) = \mathsf{ind}_{\mathcal{C}}(t)$.

Since $\mathsf{ind}_{\mathcal{C}}(u) = \max\{\mathsf{ind}_{\mathcal{C}}(x) \mid x \in vars^1(u)\}$, we deduce that $\mathsf{ind}_{\mathcal{C}}(u) = \mathsf{ind}_{\mathcal{C}_1}(u\sigma_1)$. Thus the result holds.

Rule Axiom(X, path)*:* Let $(X, i \overset{?}{\vdash} t) \in D(\mathcal{C})$ and $(\xi, j \rhd v) \in \Phi(\mathcal{C})$ such that $\mathsf{path}(\xi) = \mathsf{path}$. According to Figure 7.1, we have that $D(\mathcal{C}_1) = (D(\mathcal{C})\sigma \setminus \{X, i \overset{?}{\vdash} t\sigma\})$ were $\sigma = \mathsf{mgu}(t, v)$.

For all $x \notin vars^1(t, v)$, we have that $x\sigma = x$ and since $D(\mathcal{C}_1) = (D(\mathcal{C})\sigma \setminus \{X, i \overset{?}{\vdash} t\sigma\})$, we can deduce that $\mathsf{ind}_{\mathcal{C}_1}(x) = \mathsf{ind}_{\mathcal{C}}(x)$.

Let $x \in \mathrm{dom}(\sigma)$. We show that for all $y \in vars^1(x\sigma)$, $\mathsf{ind}_{\mathcal{C}_1}(y) \leq \mathsf{ind}_{\mathcal{C}}(x)$. If $\mathsf{ind}_{\mathcal{C}}(x) = \ell$ then it implies that there exists $(Y, \ell \overset{?}{\vdash} w) \in D(\mathcal{C})$ such that $x \in vars^1(w)$. If $X \neq Y$ then we have that $(Y, \ell \overset{?}{\vdash} w\sigma) \in D(\mathcal{C}_1)$ and since $y \in vars^1(w\sigma)$, we can deduce that $\mathsf{ind}_{\mathcal{C}_1}(y) \leq \mathsf{ind}_{\mathcal{C}}(x)$. Else if $X = Y$ then it implies that $x \in vars^1(t)$. But $\Phi(\mathcal{C}_1) = \Phi(\mathcal{C})$ which implies that $(\xi, j \rhd v\sigma) \in \Phi(\mathcal{C}_1)$. Thus, by the origination property of a constraint system, we can deduce that $y \in vars^1(v\sigma)$ implies that there exists $(Z, p \overset{?}{\vdash} w') \in D(\mathcal{C}_1)$ such that $p < j$ and $y \in vars^1(w')$. Hence we deduce that $\mathsf{ind}_{\mathcal{C}_1}(y) \leq p < j \leq \mathsf{ind}_{\mathcal{C}}(x)$. Hence the result holds.

Let $x \in vars^1(\text{img}(\sigma))$. If $\mathsf{ind}_{\mathcal{C}}(x) < i$ then since $D(\mathcal{C}_1) = (D(\mathcal{C})\sigma \setminus \{X, i \overset{?}{\vdash} t\sigma\})$, we deduce that $\mathsf{ind}_{\mathcal{C}_1}(x) \leq \mathsf{ind}_{\mathcal{C}}(x)$. Else $\mathsf{ind}_{\mathcal{C}}(x) = i$. But since $x \in vars^1(v\sigma)$, the by the properties of origination, we deduce that $\mathsf{ind}_{\mathcal{C}_1}(x) < \mathsf{ind}_{\mathcal{C}}(x)$.

We proved that for all $x \in vars^1(u)$, we have that $\mathsf{ind}_{\mathcal{C}_1}(x\sigma) \leq \mathsf{ind}_{\mathcal{C}}(x)$. Thus we conclude that $\mathsf{ind}_{\mathcal{C}_1}(u\sigma) \leq \mathsf{ind}_{\mathcal{C}}(u)$.

Rule EQ-RIGHT-RIGHT(X, Y)*:* Let $(X, i_1 \overset{?}{\vdash} v_1) \in D(\mathcal{C})$ and $(Y, i_2 \overset{?}{\vdash} v_2) \in D(\mathcal{C})$ such that $i_1 \geq i_2$. According to Figure 7.2, we have that $D(\mathcal{C}_1) = (D(\mathcal{C})\sigma \setminus \{X, i_1 \overset{?}{\vdash} v_1\sigma\})$ were $\sigma = \mathsf{mgu}(v_1, v_2)$.

For all $x \notin vars^1(v_1, v_2)$, we have that $x\sigma = x$ thus we can deduce that $\mathsf{ind}_{\mathcal{C}_1}(x) = \mathsf{ind}_{\mathcal{C}}(x)$.

Let $x \in \text{dom}(\sigma)$. Let $y \in vars^1(x\sigma)$. If $\mathsf{ind}_{\mathcal{C}}(x) = \ell$ then it implies that there exists $(Z, \ell \overset{?}{\vdash} w) \in D(\mathcal{C})$ such that $x \in vars^1(w)$. If $X \neq Z$ then we have that $(Z, \ell \overset{?}{\vdash} w\sigma) \in D(\mathcal{C}_1)$ and since $y \in vars^1(w\sigma)$, we can deduce that $\mathsf{ind}_{\mathcal{C}_1}(y) \leq \mathsf{ind}_{\mathcal{C}}(x)$. Else if $X = Z$ then it implies that $x \in vars^1(v_1)$. But $(Y, i_2 \overset{?}{\vdash} v_2\sigma) \in D(\mathcal{C}_1)$ and $y \in vars^1(v_2\sigma)$. Hence we have that $\mathsf{ind}_{\mathcal{C}_1}(y) \leq i_2 \leq i_1 = \mathsf{ind}_{\mathcal{C}}(x)$.

Let $x \in vars^1(\text{img}(\sigma))$. If $\mathsf{ind}_{\mathcal{C}}(x) < i_1$ then since $D(\mathcal{C}_1) = (D(\mathcal{C})\sigma \setminus \{X, i_1 \overset{?}{\vdash} v_1\sigma\})$, we deduce that $\mathsf{ind}_{\mathcal{C}_1}(x) \leq \mathsf{ind}_{\mathcal{C}}(x)$. Else $\mathsf{ind}_{\mathcal{C}}(x) = i_1$. But since $x \in vars^1(v_2\sigma)$ and $(Y, i_2 \overset{?}{\vdash} v_2\sigma) \in D(\mathcal{C}_1)$, we deduce that $\mathsf{ind}_{\mathcal{C}_1}(x) \leq i_2 \leq i_1 = \mathsf{ind}_{\mathcal{C}}(x)$.

We proved that for all $x \in vars^1(u)$, we have that $\mathsf{ind}_{\mathcal{C}_1}(x\sigma) \leq \mathsf{ind}_{\mathcal{C}}(x)$. Thus we conclude that $\mathsf{ind}_{\mathcal{C}_1}(u\sigma) \leq \mathsf{ind}_{\mathcal{C}}(u)$. □

Lemma 8.15. *Let (M, M') be a pair of matrices of constraint systems obtained at the end of the Step c of Phase 1 of the strategy with parameter s and k respectively for support and column. Let (M_1, M_1') be a pair of matrices of constraint systems obtained by application on (M, M') of Steps b and c with the same parameters. $\mu^k_{1.b+c}(M_1, M_1') < \mu^k_{1.b+c}(M, M')$.*

Proof. Since (M, M') and (M_1, M_1') are both obtained at two consecutive end of step c of phase 1, there exists (M_2, M_2') such that $(M, M') \to^* (M_2, M_2') \to^* (M_1, M_1')$ and (M_2, M_2') is obtained at the end of Step b of Phase 1. The proof of this result is divided in two parts. We first show that $\mu^k_{1.b+c}(M_2, M_2') \leq \mu^k_{1.b+c}(M, M')$. Secondly we show that $\mu^k_{1.b+c}(M_1, M_1') < \mu^k_{1.b+c}(M_2, M_2')$.

First part, $\mu^k_{1.b+c}(M_2, M_2') \leq \mu^k_{1.b+c}(M, M')$: All the rules applied during Step b are internal rules. Furthermore, we know that (M, M') was obtained at the end of step c which means that we already apply at leaf one Step b before obtaining (M, M'). Hence, the rules DED-ST and EQ-LEFT-LEFT are useless on (M, M') for the support s and the column k. Thus the only rules that are applied between (M, M') and (M_2, M_2') are CONS, EQ-RIGHT-RIGHT and AXIOM.

Consider two pair of matrices (M_3, M_3') and (M_4, M_4') such that $(M, M') \to^* (M_4, M_4') \to (M_3, M_3') \to^* (M_2, M_2')$ and assume that $R(\tilde{p})$ is the rule applied on (M_4, M_4'). Let $\mathcal{C}$ be a constraint system in the k^{th} column of (M_4, M_4'). If the rule $R(\tilde{p})$ was not applied on $\mathcal{C}$ then $\mathcal{C}$ is in the k^{th} column on (M_3, M_3'). If the rule $R(\tilde{p})$ was applied on $\mathcal{C}$ then there exists $\mathcal{C}_1$ and $\mathcal{C}_2$ such that $\mathcal{C}_1$ and $\mathcal{C}_2$ both in the k^{th} column of (M_3, M_3') and they are the two constraint systems obtained by applying $R(\tilde{p})$ on $\mathcal{C}$.

According to Figure 7.1 and 7.2, and the normalisation rules 7.3, for all $i \in \{1, 2\}$, for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}_i)$, there exists $(Y, i \overset{?}{\vdash} v) \in D(\mathcal{C})$ and $v' \in st(v)$ such that $v'\mathsf{mgu}(Eq(\mathcal{C}_i)) = u$. But thanks to Lemma C.54, $\mathsf{ind}_{\mathcal{C}_i}(v'\mathsf{mgu}(Eq(\mathcal{C}_i))) \leq \mathsf{ind}_{\mathcal{C}}(v')$ and so $\mathsf{ind}_{\mathcal{C}_i}(u) \leq \mathsf{ind}_{\mathcal{C}}(v')$. Since $v' \in st(v)$, we deduce that $\mathsf{ind}_{\mathcal{C}_i}(u) \leq \mathsf{ind}_{\mathcal{C}}(v)$. This allows us to deduce that $\max\{\mathsf{ind}_{\mathcal{C}_i}(u) \mid X \notin S_2(\mathcal{C}_i), (X, j \overset{?}{\vdash} u) \in D(\mathcal{C}_i)\} \leq \max\{\mathsf{ind}_{\mathcal{C}}(u) \mid X \notin S_2(\mathcal{C}), (X, j \overset{?}{\vdash} u) \in D(\mathcal{C})\}$. Hence we deduce that $\mu^k_{1.b+c}(M_3, M_3') \leq \mu^k_{1.b+c}(M_4, M_4')$. With a simple induction, we can conclude that $\mu^k_{1.b+c}(M_2, M_2') \leq \mu^k_{1.b+c}(M, M')$.

Second part, $\mu^k_{1.b+c}(M_1, M_1') < \mu^k_{1.b+c}(M_2, M_2')$: Let $\mathcal{C}_1$ be a constraint system in the k^{th} column of (M_1, M_1'). Let $\mathcal{C}_2$ be the constraint system ancestor of $\mathcal{C}_1$ such that $\mathcal{C}_2$ is in (M_2, M_2').

Thanks to Lemma C.31, we know that (M_1, M_1') satisfies $\mathsf{PP1ScE}(s,k)$, and (M_2, M_2') satisfies $\mathsf{PP1SbE}(s,k)$. Thus we deduce that for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}_1)$, if $X \notin S_2(\mathcal{C}_1)$, then $u \notin \mathcal{X}^1$ and $i = s$. Furthermore, we deduce that for all $(X, i \overset{?}{\vdash} u) \in D(\mathcal{C}_2)$, if $X \notin S_2(\mathcal{C}_2)$, then $u \in \mathcal{X}^1$ and $i = s$. But the rules applied during step c either remove internal deducible constraint by Eq-right-right or instantiate them by Axiom or Cons. Thus, for all $(X, s \overset{?}{\vdash} t_1) \in D(\mathcal{C}_1)$, if $X \notin S_2(\mathcal{C}_1)$, then there exists t_2 such that $(X, s \overset{?}{\vdash} t_2) \in D(\mathcal{C}_2)$.

Let $\mathcal{C}_3$ be a constraint system such that $\mathcal{C}_2 \to^* \mathcal{C}_3 \to^* \mathcal{C}_1$. Let $(X, s \overset{?}{\vdash} t_3) \in D(\mathcal{C}_3)$. Thanks to Lemma C.54, we know that $\mathsf{ind}_{\mathcal{C}_1}(t_1) \leq \mathsf{ind}_{\mathcal{C}_3}(t_3) \leq \mathsf{ind}_{\mathcal{C}_2}(t_2)$. We will show that that there exists $\mathcal{C}_4$ and $\mathcal{C}_3$ such that $\mathcal{C}_2 \to^* \mathcal{C}_3 \to \mathcal{C}_4 \to^* \mathcal{C}_1$, $(X, s \overset{?}{\vdash} t_4) \in D(\mathcal{C}_4)$, $(X, s \overset{?}{\vdash} t_3) \in D(\mathcal{C}_3)$ and $\mathsf{ind}_{\mathcal{C}_4}(t_4) < \mathsf{ind}_{\mathcal{C}_3}(t_3)$, which will imply that $\mathsf{ind}_{\mathcal{C}_1}(t_1) < \mathsf{ind}_{\mathcal{C}_2}(t_2)$.

Let $\mathcal{C}_3$ and $\mathcal{C}_4$ be the constraint systems such that $(X, s \overset{?}{\vdash} t_3) \in D(\mathcal{C}_3)$, $(X, s \overset{?}{\vdash} t_4) \in D(\mathcal{C}_4)$, $t_2 = t_3$, $t_3 \neq t_4$ and $\mathcal{C}_2 \to^* \mathcal{C}_3 \to \mathcal{C}_4 \to^* \mathcal{C}_1$. Those two constraints systems exist since we know that $t_1 \neq t_2$. Let $\tilde{R()}$ be the rule applied on $\mathcal{C}_3$ to obtained $\mathcal{C}_4$. Since $t_2 = t_3$, we deduce that $t_3 \in \mathcal{X}^1$ and so $t_3 \in \mathsf{X}^1(\mathcal{C}_3)$. Thanks to Lemma C.53 and $t_3 \in \mathsf{X}^1(\mathcal{C}_3)$, we deduce that $\tilde{R()}$ is not an instance of Cons. Furthermore, since Eq-right-right is applied internally we also deduce that $\tilde{R()}$ is not an instance of Eq-right-right. Hence we conclude that $\tilde{R()}$ is an instance of Axiom.

Assume that $R(\tilde{p}) = \textsc{Axiom}(X_0, \mathsf{path})$. By definition, of the rule, there exists $(X_0, i_0 \overset{?}{\vdash} u_0) \in D(\mathcal{C}_3)$ and $(\xi, j_0 \rhd v_0) \in \Phi(\mathcal{C}_3)$ such that $j_0 \leq i_0$. Furthermore, we have $D(\mathcal{C}_4) = D(\mathcal{C}_3)\sigma \setminus \{X_0, i_0 \overset{?}{\vdash} u_0\sigma\}$, $t = v\sigma$ and $\Phi(\mathcal{C}_4) = \Phi(\mathcal{C}_3)\sigma$ where $\sigma = \mathsf{mgu}(u_0, v_0)$. Since by hypothesis $(X, s \overset{?}{\vdash} t_4) \in D(\mathcal{C}_4)$ and $t_4 \neq t_3$, we have that $t_3 \in \mathrm{dom}(\sigma)$ and $t_3 \in \mathit{vars}^1(u_0, v_0)$. But the rule Axiom is applied on the deducible constraint minimal for $\mathcal{L}^1()$. Hence we deduce that $t_3 \in \mathit{vars}^1(u_0)$ and $\mathsf{ind}_{\mathcal{C}_3}(t_3) = i_0$. But for all $x \in \mathit{vars}^1(t_3\sigma)$, we have that $x \in \mathit{vars}^1(v_0\sigma)$ and $\Phi(\mathcal{C}_4) = \Phi(\mathcal{C}_3)\sigma$. Thus, by the property of origination of a constraint system, we know that there exists $(Y, p \overset{?}{\vdash} w) \in D(\mathcal{C}_4)$ such that $x \in \mathit{vars}^1(w)$ and $p < j_0 \leq i_0$. Thus, we have that $\mathsf{ind}_{\mathcal{C}_4}(x) < i_0$ and so for all $x \in \mathit{vars}^1(t_3\sigma))$, $\mathsf{ind}_{\mathcal{C}_4}(x) < \mathsf{ind}_{\mathcal{C}_3}(t_3)$. Since $t_3\sigma = t_4$, we conclude that $\mathsf{ind}_{\mathcal{C}_4}(t_4) < \mathsf{ind}_{\mathcal{C}_3}(t_3)$.

We have shown that for all $(X, s \overset{?}{\vdash} u) \in D(\mathcal{C}_1)$, if $X \notin S_2(\mathcal{C}_1)$ there exists v such that $(X, s \overset{?}{\vdash} v) \in D(\mathcal{C}_2)$ and $\mathsf{ind}_{\mathcal{C}_1}(u) < \mathsf{ind}_{\mathcal{C}_2}(v)$. Thus we can conclude that $\mu^k_{1.b+c}(M_1, M_1') < \mu^k_{1.b+c}(M_2, M_2')$. □

C.7.1.5 Termination of Step d of Phase 1

Lemma 8.16. *Let (M, M') be a pair of matrices of constraint systems obtained during Step d of Phase 1 of the strategy with parameter s and k respectively for support and column. Furthermore, let $R(\tilde{p})$ be the next possible rule applicable according to step d of the strategy and let (M_1, M_1') and (M_2, M_2') be the two pairs of matrices of constraint systems obtained by application of $R(\tilde{p})$ on (M, M'). We have that:*

$$\mu^k_{1.d}(M_1, M_1') < \mu^k_{1.d}(M, M') \quad \wedge \quad \mu^k_{1.d}(M_2, M_2') < \mu^k_{1.d}(M, M')$$

Proof. Assume w.l.o.g. that the k^{th} column of (M, M') is the k^{th} column of M. We know that the only rules that can be applied in Step d are Cons, Eq-right-right and Axiom. Furthermore, the external application of those rules implies that $M_{i,k} \to M_{1i,k}$ and $M_{i,k} \to M_{2i,k}$, for all $i \in \{1, \ldots, n\}$. But thanks to Lemma C.14, we know that if $M_{i,k}$ satisfies $\mathsf{InvVarConstraint}(s)$ then $M_{1i,k}$ and $M_{2i,k}$ also satisfy $\mathsf{InvVarConstraint}(s)$ or are equal to $\bot$. Thus, we deduce that $\pi_1(\mu^k_{1.d}(M_1, M_1')) \leq \pi_1(\mu^k_{1.d}(M, M'))$ and $\pi_1(\mu^k_{1.d}(M_2, M_2')) \leq \pi_1(\mu^k_{1.d}(M, M'))$.

Let i_0 be the index of the line such that for all $i \leq i_0$, $M_{i_0,k} = \bot$ or $M_{i,k}$ satisfies the invariant $\mathsf{InvVarConstraint}(s)$. Let $R(\tilde{p})$ be an instance of a rule that is strongly applicable on $M_{j_0,k}$. We do

a case analysis on j_0.

Case $j_0 \leq i_0+1$ and $R(\tilde{p})$ strongly applicable on $M_{i_0+1,k}$: In such a case, by applying Lemma 8.13, we obtained that $\mu_{1.b}(M_{1i_0+1,k}) < \mu_{1.b}(M_{i_0+1,k})$ and $\mu_{1.b}(M_{2i_0+1,k}) < \mu_{1.b}(M_{i_0+1,k})$. It implies that $\pi_2(\mu^k_{1.d}(M_1, M'_1)) < \pi_2(\mu^k_{1.d}(M, M'))$ and $\pi_2(\mu^k_{1.d}(M_2, M'_2)) < \pi_2(\mu^k_{1.d}(M, M'))$. Hence the result holds.

Case $j_0 \leq i_0 + 1$ and $R(\tilde{p})$ not strongly applicable on $M_{i_0+1,k}$: Since $R(\tilde{p})$ is strongly applicable on $M_{j_0,k}$ then we have that $M_{j_0,k} \neq \bot$ and $j_0 \neq i_0 + 1$. Hence by definition of i_0, we deduce that $M_{j_0,k}$ satisfies the invariant $\mathsf{InvVarConstraint}(s)$. Hence $R(\tilde{p})$ strongly applicable on $M_{j_0,k}$ implies that:

— $R(\tilde{p}) = \textsc{Cons}(X, \mathsf{f})$ with $(X, i \overset{?}{\vdash} x) \in D(M_{j_0,k})$, $x \in \mathcal{X}^1$ and there exists $\mathsf{g} \in \mathcal{F}_c$ such that $Er(M_{j_0,k}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$. But thanks to Lemma 8.3, (M, M') satisfies $\mathsf{InvGeneral}$ and more specifically Property 6 of the invariant $\mathsf{InvGeneral}$ which means that $Er(M_{i_0+1,k}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$. Thus, we deduce that $R(\tilde{p})$ is also strongly applicable on $M_{i_0+1,k}$ which is a contradiction with our hypothesis.

— $R(\tilde{p}) = \textsc{Axiom}(X, \mathsf{path})$ with $(X, i \overset{?}{\vdash} x) \in D(M_{j_0,k})$, $x \in \mathcal{X}^1$ and there exists $\mathsf{g} \in \mathcal{F}_c$ such that $Er(M_{j_0,k}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$. Once again, thanks to Property 6 of the invariant $\mathsf{InvGeneral}$, we deduce that $Er(M_{i_0+1,k}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$. However $R(\tilde{p})$ is not strongly applicable on $M_{i_0+1,k}$ which means that if there exists a frame element $(\xi, i \overset{?}{\vdash} u) \in \Phi(M_{i_0+1,k})$ such that $\mathsf{path}(\xi) = \mathsf{path}$ then $Er(M_{i_0+1,k}) \vDash X \overset{?}{\neq} \xi$. But thanks to Property 7 and the fact that $\textsc{Axiom}(X, \mathsf{path})$ is strongly applicable on $M_{j_0,k}$, we deduce that there is no frame element $(\xi, i \overset{?}{\vdash} u) \in \Phi(M_{i_0+1,k})$ such that $\mathsf{path}(\xi) = \mathsf{path}$. Thus, we have that $M_{1i_0+1,k} = \bot$ and $M_{2i_0+1,k} = M_{i_0+1,k}$.
Therefore, we deduce that $\pi_1(\mu^k_{1.d}(M_1, M'_1)) < \pi_1(\mu^k_{1.d}(M, M'))$ and $\pi_2(\mu^k_{1.d}(M_2, M'_2)) = \pi_2(\mu^k_{1.d}(M, M'))$. Furthermore, we have that $Er(M_{2j_0,k}) = Er(M_{j_0,k}) \wedge X \overset{?}{\neq} \xi$ where $\mathsf{path}(\xi) = \mathsf{path}$ and $(\xi, i \overset{?}{\vdash} u) \in \Phi(M_{j_0,k})$. Hence we deduce that $\pi_3(\mu^k_{1.d}(M_2, M'_2)) < \pi_3(\mu^k_{1.d}(M, M'))$. Thus we conclude that $\mu^k_{1.d}(M_1, M'_1) < \mu^k_{1.d}(M, M')$ and $\mu^k_{1.d}(M_2, M'_2) < \mu^k_{1.d}(M, M')$.

Case $j_0 > i_0 + 1$: In such a case, it implies that no instance of the rule Cons, Axiom and Eq-right-right can be strongly applied on $M_{i_0+1,k}$ with support inferior of equal to s. Thus, we can first deduce that for all $(X, i \overset{?}{\vdash} u) \in D(M_{i_0+1,k})$, if $i \leq s$ then $u \in \mathcal{X}^1$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(M_{i_0+1,k}) \nvDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$. Indeed, assume that $u \notin \mathcal{X}^1$.

— if there exists $(\xi, j \rhd v) \in \Phi(M_{i_0+1,k})$ such that $j \leq i$ and $Er(M_{i_0+1,k}) \nvDash X \overset{?}{\neq} \xi$, then $\textsc{Axiom}(X, \mathsf{path}(\xi))$ would be strongly applicable on $M_{i_0,k}$ which contradicts our hypothesis

— if there exists $\mathsf{f} \in \mathcal{F}_c$ such that $Er(M_{i_0+1,k}) \nvDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$, then $\textsc{Cons}(X, \mathsf{f})$ would be strongly applicable on $M_{i_0+1,k}$ which contradicts our hypothesis.

— Else we would have that for all $(\xi, j \rhd v) \in \Phi(M_{i_0+1,k})$, $j \leq i$ implies $Er(M_{i_0+1,k}) \vDash X \overset{?}{\neq} \xi$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(M_{i_0+1,k}) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$. But in such a case, the normalisation of the constraint system would implies that $M_{i_0+1,k} = \bot$ witch contradicts the definition of i_0.

We now prove that in fact the case $j_0 > i_0 + 1$ is impossible. Assume that $(X, i \overset{?}{\vdash} x)$ and $(Y, j \overset{?}{\vdash} y)$ in $D(M_{i_0+1,k})$ such that $x = y$. In such a case, since we proved that for all $\mathsf{f} \in \mathcal{F}_c$, $Er(M_{i_0+1,k}) \nvDash$

$\mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er(M_{i_0+1,k}) \not\vdash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{f}$, we can deduce that the rule Eq-right-right(X, Y) is strongly applicable on $M_{i_0+1,k}$ which is a contradiction on our hypothesis. Hence we can deduce that $M_{i_0+1,k}$ satisfies the invariant $\mathsf{InvVarConstraint}(s)$ which is also a contradiction on the definition of i_0. Hence the case $j_0 > i_0 + 1$ is impossible. □

C.7.2 Proofs of results on association tables

Lemma C.55. *Let $\mathcal{C}$ be a well-formed constraint system. Let E be a disjunction of disequation on first order constructive term. Let D be a disjunction of disequation on context of recipe. We have that:*

$$L^1_{\mathcal{C}}(E) = L^1_{\mathcal{C}}(E{\downarrow}) \quad \textit{and} \quad L^2_{\mathcal{C}}(D) = L^2_{\mathcal{C}}(D{\wr})$$

Proof. Direct from the definition of normalisation $\downarrow$ and $\wr$. □

Lemma C.56. *Let E be a disjunction of first order constructive term and let σ a substitution on constructive term. Let D be a disjunction of term in $\mathcal{T}(\mathcal{F}_c, (\mathcal{F}_d^* \cdot \mathcal{AX}) \cup \mathcal{X}^2)$ and let θ be a substitution from $\mathcal{X}^2$ to $\mathcal{T}(\mathcal{F}_c, (\mathcal{F}_d^* \cdot \mathcal{AX}) \cup \mathcal{X}^2)$. We have that:*

$$E\sigma{\downarrow} = (E{\downarrow})\sigma{\downarrow} \quad \textit{and} \quad D\theta{\wr} = (D{\wr})\theta{\wr}$$

Let $\mathcal{C}$ be a constraint system such that for all $\beta \in st(D) \cap ((\mathcal{F}_d^* \cdot \mathcal{AX}) \cup \mathcal{X}^2)$, $\beta \in \mathrm{dom}(\mathsf{acc}^1(\mathcal{C}))$. We have that $(D{\wr})\mathsf{acc}^1(\mathcal{C}){\downarrow} = D\mathsf{acc}^1(\mathcal{C}){\downarrow}$.

Proof. Direct from the definition of normalisation $\downarrow$ and $\wr$. □

Lemma 8.17. *Let (M, M') be a pair of matrices of constraint systems obtained during Step b or Step c of Phase 2 of the strategy. For all constraint systems $\mathcal{C}$, $\mathcal{C}'$ in (M, M') and their respective association tables T, T', for all $\bigvee_i^n x_i \overset{?}{\neq} v_i$, for all $\bigvee_j^m \beta_j \overset{?}{\neq} \beta'_j$, if $T[\bigvee_i^n x_i \overset{?}{\neq} v_i] = \bigvee_j^m \beta_j \overset{?}{\neq} \beta'_j$, then we have that:*

$$\left(\bigvee_j^m \beta_j\mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \beta'_j\mathsf{acc}^1(\mathcal{C})\right){\downarrow} = \bigvee_i^n x_i \overset{?}{\neq} v_i$$

Moreover, if $T[\bigvee_i^n x_i \overset{?}{\neq} v_i] = \bigvee_j^m \beta_j \overset{?}{\neq} \beta'_j$ and for all $k \in \{1, \ldots, m\}$, $st(\beta_j, \beta'_j) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$ then there exists $\bigvee_i^{n'} x'_i \overset{?}{\neq} v'_i$ such that $T'[\bigvee_i^{n'} x'_i \overset{?}{\neq} v'_i] = \bigvee_j^m \beta_j \overset{?}{\neq} \beta'_j$.

Proof. Let (M_0, M'_0) be the pair of matrices of constraint system, ancestor of (M, M'), obtained at the end of Step a of the second phase of the strategy. As mentioned in Section 7.2, there is three different kind of transformations on the association tables. The first one, called default transformation, is applied on each application of any rule. The second one consist of added new entry on the association table after the application of the rule Cons. The last one consisting of removing some entry is applied when the association tables meet some conditions. Hence we will prove the result by case analysis on the three transformations and by induction on the size N of the branch between (M_0, M'_0) and (M, M').

Base case $N = 0$: In such a case, we have $(M, M') = (M_0, M'_0)$. But according to the strategy, the association table of any constraint system, in (M_0, M'_0) are empty. Hence the result trivially holds.

Inductive step $N > 0$: Since $N > 0$, we know that there exists (M_1, M'_1) a pair of matrices of constraint systems such that $(M_1, M'_1) \to (M, M')$. Let $\mathcal{C}$ be two constraint system in (M, M') and let T its respective association table. Let $R(\tilde{p})$ be the rule applied on (M_1, M'_1) to obtain (M, M'). Furthermore, let $\mathcal{C}_1$ be the constraint system in (M_1, M'_1) such that $\mathcal{C}_1 \to \mathcal{C}$ and let T_1 its association table. We do a case analysis on the transformation from T_1 to T.

Let E be a disjunction of inequation on first order term and let D be a disjunction of inequations on context of recipe such that $T[E] = D$. For simplicity purpose, we will denote $\sigma = \mathsf{mgu}(Eq(\mathcal{C}))$, $\theta = \mathsf{mgu}(Er(\mathcal{C}))$ and $\Theta = \{X \mapsto \mathsf{C}\lfloor X\theta \rfloor_{\Phi(\mathcal{C})} \mid X \in vars^2(\mathcal{C})\}$.

— *Default transformation:* By definition of the default transformation, we know that there exists a disjunction on first order term such that $E = E_1\sigma\downarrow$ and $T[E] = T_1[E_1]\Theta\downarrow$. Let's denote $T_1[E_1] = D_1 = \bigvee_i \beta_i \stackrel{?}{\neq} \beta'_i$
By inductive hypothesis on (M_1, M'_1), we know that $D_1\mathsf{acc}^1(\mathcal{C}_1)\downarrow = E_1$ which implies that $D_1\mathsf{acc}^1(\mathcal{C}_1)\downarrow\sigma\downarrow = E_1\sigma\downarrow = E$. Hence, thanks to Lemma C.56, we deduce that $D_1\mathsf{acc}^1(\mathcal{C}_1)\sigma\downarrow = E$. But thanks to Lemma C.11, we know that for all i, for all $X \in vars^2(\beta_i)$, $X\mathsf{acc}^1(\mathcal{C}_1)\sigma = \mathsf{C}\lfloor X\theta\rfloor_{\Phi(\mathcal{C})}\mathsf{acc}^1(\mathcal{C}) = X\Theta\mathsf{acc}^1(\mathcal{C})$. Hence, since for all i, $\beta_i \in \mathcal{T}(\mathcal{F}_c, (\mathcal{F}_d^*\cdot\mathcal{AX})\cup\mathcal{X}^2)$, we deduce that $\beta_i\mathsf{acc}^1(\mathcal{C}_1)\sigma = \beta_i\Theta\mathsf{acc}^1(\mathcal{C})$. Similarly, we have that $\beta'_i\mathsf{acc}^1(\mathcal{C}_1)\sigma = \beta'_i\Theta\mathsf{acc}^1(\mathcal{C})$. Hence we deduce that $D_1\mathsf{acc}^1(\mathcal{C}_1)\sigma = D_1\Theta\mathsf{acc}^1(\mathcal{C})$. But again thanks to Lemma C.56, we prove that $D_1\Theta\mathsf{acc}^1(\mathcal{C})\downarrow = (D_1\Theta\{\!\})\mathsf{acc}^1(\mathcal{C})\downarrow = D\mathsf{acc}^1(\mathcal{C})\downarrow$. Thus we conclude that $D\mathsf{acc}^1(\mathcal{C})\downarrow = D_1\mathsf{acc}^1(\mathcal{C}_1)\sigma\downarrow = E$.

— *Rule* Eq-right-right(X, ξ)*:* In the case of the rule Eq-right-right when the inequation holds, the strategy indicates that we add on T the following entry: $T[x \stackrel{?}{\neq} v] := X \stackrel{?}{\neq} \xi$ where $x = X\mathsf{acc}^1(\mathcal{C})$ and $v = \xi\mathsf{acc}^1(\mathcal{C})$. Hence we deduce that $(X \stackrel{?}{\neq} \xi)\mathsf{acc}^1(\mathcal{C})\downarrow = (x \stackrel{?}{\neq} v)\downarrow$. But $v \in T(\mathcal{F}_c, \mathcal{X}^1)$ which means that $(x \stackrel{?}{\neq} v)\downarrow = (x \stackrel{?}{\neq} v)$ and so the result holds.

— *Removing entry:* The last transformation consists of removing some entry from the association table. Hence the result trivially holds. □

Lemma 8.18. *Let $\mathcal{C}$ be a well-formed constraint system satisfying* $\mathsf{InvVarConstraint}(\infty)$*. Let β, β' be two contexts of recipes such that $\beta, \beta' \in \mathcal{T}(\mathcal{F}_c \cup (\mathcal{F}_d^* \cdot \mathcal{AX}), \mathcal{X}^2)$. We have that $L^1_{\mathcal{C}}(\beta\mathsf{acc}^1(\mathcal{C}) \stackrel{?}{\neq} \beta'\mathsf{acc}^1(\mathcal{C})) \leq L^2_{\mathcal{C}}(\beta \stackrel{?}{\neq} \beta')$. Furthermore, if $\beta \in (\mathcal{F}_d^* \cdot \mathcal{AX})$ or $\beta' \in (\mathcal{F}_d^* \cdot \mathcal{AX})$, then we have that $L^1_{\mathcal{C}}(\beta\mathsf{acc}^1(\mathcal{C}) \stackrel{?}{\neq} \beta'\mathsf{acc}^1(\mathcal{C})) < L^2_{\mathcal{C}}(\beta \stackrel{?}{\neq} \beta')$.*

Proof. We know that $L^2_{\mathcal{C}}(\beta \stackrel{?}{\neq} \beta') = \{\{\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\beta); \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\beta')\}\}$. But we know that $\beta, \beta' \in \mathcal{T}(\mathcal{F}_c, (\mathcal{F}_d^* \cdot \mathcal{AX}) \cup \mathcal{X}^2)$ which means that $\mathsf{param}^{\mathcal{C}}_{\mathsf{max}}(\beta) = \max\{\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\zeta) \mid \zeta \in st(\beta) \cap ((\mathcal{F}_d^* \cdot \mathcal{AX}) \cup \mathcal{X}^2)\}$. Similarly, we have that $\mathsf{ind}_{\mathcal{C}}(\beta\mathsf{acc}^1(\mathcal{C})) = \max\{\mathsf{ind}_{\mathcal{C}}(u) \mid \zeta \in st(\beta) \cap ((\mathcal{F}_d^* \cdot \mathcal{AX}) \cup \mathcal{X}^2) \wedge u = \zeta\mathsf{acc}^1(\mathcal{C})\}$.

Hence, to prove the result, it suffices to show that for all $(\xi, i \rhd u) \in \Phi(\mathcal{C})$, $\mathsf{ind}_{\mathcal{C}}(u) \leq \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\mathsf{path}(\xi))$; and for all $(X, i \stackrel{?}{\vdash} x) \in D(\mathcal{C})$, $\mathsf{ind}_{\mathcal{C}}(x) \leq \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X)$.

— Case $(X, i \stackrel{?}{\vdash} x) \in D(\mathcal{C})$: We know that $\mathcal{C}$ satisfies the invariant $\mathsf{InvVarConstraint}(\infty)$. Hence, $(X, i \stackrel{?}{\vdash} x)$ is the only deducible constraint system that contain x which means that $\mathsf{ind}_{\mathcal{C}}(x) = i$. Furthetmore, by definition of $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X)$, we have that $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X) = i$ and so the result holds.

— Case $(\xi, i \rhd u) \in \Phi(\mathcal{C})$: We know that $\mathsf{path}(\xi) \in \mathcal{F}_d^* \cdot \mathcal{AX}$ hence there exists ax_k such that $\xi = w \cdot ax_k$. Hence by definition, we have that $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\mathsf{path}(\xi)) = k$. Let v such that $(ax_k, k \rhd v) \in \Phi(\mathcal{C})$. Once again thanks to the fact that $\mathcal{C}$ is well-formed, we can deduce that $u \in st(v)$. But thanks to the origination property of constraint system and the invariant $\mathsf{InvVarConstraint}(\infty)$, we have that for all $x \in vars^1(v)$. there exists $(X, \ell \stackrel{?}{\vdash} x) \in D(\mathcal{C})$ such that $\ell < k$. Hence, we can deduce that $\mathsf{ind}_{\mathcal{C}}(u) \leq \mathsf{ind}_{\mathcal{C}}(v) < k = \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\mathsf{path}(\xi))$. Hence the result holds. □

Lemma 8.19. *Let (M, M') be a pair of matrices of constraint systems obtained during Step b or Step c of Phase 2 of the strategy. Let $R(\tilde{p})$ be one of the following external rules:* Axiom, Cons, Eq-right-right. *Let (M_1, M'_1) and (M_2, M'_2) be the results of the application of $R(\tilde{p})$ on (M, M'). We have that:*

$$\mu_{gen}(M_1, M'_1) \leq \mu_{gen}(M, M') \quad \textit{and} \quad \mu_{gen}(M_2, M'_2) \leq \mu_{gen}(M, M')$$

Furthermore, for $i = 1, 2$, if a removal transformation was applied on the association table of (M_i, M_i') then $\mu_{gen}(M_i, M_i') < \mu_{gen}(M, M')$.

Proof. The removal transformation is independent of the rule applied on the pair of matrices. Hence, we will first prove the result without considering the removal transformation (i.e. only default transformation for the rule Cons and Axiom; and the default and addition transformations for the rule Eq-right-right). Then we will show the result when a removal transformation is applied. We do a case analysis on the rule applied on (M, M'):

Case Rule $\textsc{Cons}(X, \mathsf{f})$: Let $\mathcal{C}$ be a constraint system in (M, M') different from $\bot$ and let $\mathcal{C}_1, \mathcal{C}_2$ the two constraint systems obtained by applying the rule $\textsc{Cons}(X, \mathsf{f})$ on $\mathcal{C}$. Hence we have that $\mathcal{C}_1$ and $\mathcal{C}_2$ are respectively in (M_1, M_1') and (M_2, M_2'). We can first notice that the only difference between $\mathcal{C}_2$ and $\mathcal{C}$ is that $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$. Hence, we can deduce that $Eq(\mathcal{C}_2) = Eq(\mathcal{C})$ and so the association tables of $\mathcal{C}$ and $\mathcal{C}_2$ are the same. Thus, we conclude that $\mu_{gen}(M_2, M_2') = \mu_{gen}(M, M')$.

We now focus on (M_1, M_1'): Since $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(\infty)$, we know that there exist $x \in \mathcal{X}^1$ such that $(X, i \stackrel{?}{\vdash} x) \in D(\mathcal{C})$. Furthermore, by definition of the rule, we know that $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \stackrel{?}{=} \mathsf{f}(X_1, \ldots, X_n)$ and $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge x \stackrel{?}{=} \mathsf{f}(x_1, \ldots, x_n)$ where X_k, x_k are fresh variable and $\mathsf{ind}_{\mathcal{C}_1}(x_k) = \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X_k) = i$, for all $k = 1 \ldots n$.

Hence, thanks to the normalisation, we deduce that for all disjunction D_1 of disjunction of first order term, if there exists E_1 such that $Eq(\mathcal{C}_1) = E_1 \wedge D_1$ then there exists D and E such that $Eq(\mathcal{C}) = E \wedge D$ and $D\sigma\downarrow = D_1$ where $\sigma = \{x \mapsto \mathsf{f}(x_1, \ldots, x_n)\}$. But $\mathsf{ind}_{\mathcal{C}}(x) = \mathsf{ind}_{\mathcal{C}_1}(x_k)$, for all $k \in \{1, \ldots, n\}$. Hence we deduce that $L^1_{\mathcal{C}}(D) = L^1_{\mathcal{C}_1}(D\sigma)$. But, thanks to Lemma C.55, we have that $L^1_{\mathcal{C}_1}(D\sigma) = L^1_{\mathcal{C}_1}(D\sigma\downarrow)$ and so $L^1_{\mathcal{C}}(D) = L^1_{\mathcal{C}_1}(D_1)$.

Since we only consider the default transformation on the association table from now, we also have that $T_1[D_1] = \bot$ implies $T[D] = \bot$. Hence, we proved that for all D_1 such that $Eq(\mathcal{C}_1) = E_1 \wedge D_1$ and $T_1[D_1] = \bot$, there exists D such that $Eq(\mathcal{C}) = E \wedge D$, $T[D] = \bot$ and $L^1_{\mathcal{C}}(D) = L^1_{\mathcal{C}_1}(D_1)$. Thus we deduce that $\mu^1_{gen}(M_1, M_1') \leq \mu^1_{gen}(M, M')$.

Furthermore, by definition of the default transformation on the association tables, we have that if $T_1[D_1]$ exists then we know that $T[D]$ also exists and $T_1[D_1] = T[D]\Theta\wr$ where $\Theta = \{X \mapsto \mathsf{f}(X_1, \ldots, X_n)\}$. But once again, we know that $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X) = \mathsf{paramC}^{\mathcal{C}_1}_{\mathsf{max}}(X_k)$, for all $k \in \{1, \ldots, n\}$. Hence we deduce that $L^2_{\mathcal{C}}(T[D]) = L^2_{\mathcal{C}_1}(T[D]\Theta)$. Thanks to Lemma C.55, we have that $L^2_{\mathcal{C}_1}(T[D]\Theta) = L^2_{\mathcal{C}_1}(T[D]\Theta\downarrow)$ and so $L^2_{\mathcal{C}_1}(T_1[D_1]) = L^2_{\mathcal{C}}(T[D])$. Thus we deduce that $\mu^2_{gen}(M_1, M_1') \leq \mu^2_{gen}(M, M')$. Since we proved that $\mu^1_{gen}(M_1, M_1') \leq \mu^1_{gen}(M, M')$, we can conclude that $\mu_{gen}(M_1, M_1') \leq \mu_{gen}(M, M')$.

Case Rule $\textsc{Axiom}(X, \mathsf{path})$: Let $\mathcal{C}$ be a constraint system in (M, M') different from $\bot$ and let $\mathcal{C}_1, \mathcal{C}_2$ the two constraint systems obtained by applying the rule $\textsc{Axiom}(X, \mathsf{path})$ on $\mathcal{C}$. Hence we have that $\mathcal{C}_1$ and $\mathcal{C}_2$ are respectively in (M_1, M_1') and (M_2, M_2'). We can first notice that either $\mathcal{C}_1 = \bot$ and $\mathcal{C}_2 = \mathcal{C}$ or else the only difference between $\mathcal{C}_2$ and $\mathcal{C}$ is that $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge \mathsf{root}(X) \stackrel{?}{\neq} \xi$ for some ξ such that $\mathsf{path}(\xi) = \mathsf{path}$. Hence, we can deduce that $Eq(\mathcal{C}_2) = Eq(\mathcal{C})$ and so the association tables of $\mathcal{C}$ and $\mathcal{C}_2$ are the same. Thus, we conclude that $\mu_{gen}(M_2, M_2') = \mu_{gen}(M, M')$.

We now focus on (M_1, M_1'): Since $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(\infty)$, we know that there exist $x \in \mathcal{X}^1$ such that $(X, i \stackrel{?}{\vdash} x) \in D(\mathcal{C})$. Furthermore, by definition of the rule, we know that there exists $(\xi, j \rhd v) \in \Phi(\mathcal{C})$ such that $\mathsf{path}(\xi) = \mathsf{path}$, $j \leq i$, $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \stackrel{?}{=} \xi$ and $Eq(\mathcal{C}_1) = Eq(\mathcal{C}) \wedge x \stackrel{?}{=} v$.

Hence, thanks to the normalisation, we deduce that for all disjunction D_1 of disjunction of first order term, if there exists E_1 such that $Eq(\mathcal{C}_1) = E_1 \wedge D_1$ then there exists D and E such that $Eq(\mathcal{C}) = E \wedge D$ and $D\sigma\downarrow = D_1$ where $\sigma = \{x \mapsto v\}$. But thanks to $\mathsf{InvVarConstraint}(\infty)$ and the property of origination of a constraint system, we know that for all $y \in vars^1(v)$, there exists $(Y, \ell \stackrel{?}{\vdash} y) \in D(\mathcal{C})$ with $\ell < j$. Hence we have that $\mathsf{ind}_{\mathcal{C}_1}(v) < \mathsf{ind}_{\mathcal{C}}(x)$ which implies that

$L^1_{\mathcal{C}_1}(D\sigma) \leq L^1_{\mathcal{C}}(D)$. But, thanks to Lemma C.55, we have that $L^1_{\mathcal{C}_1}(D\sigma) = L^1_{\mathcal{C}_1}(D\sigma\downarrow)$ and so $L^1_{\mathcal{C}_1}(D_1) \leq L^1_{\mathcal{C}}(D)$.

Since we only consider the default transformation on the association table from now, we also have that $T_1[D_1] = \bot$ implies $T[D] = \bot$. Hence, we proved that for all D_1 such that $Eq(\mathcal{C}_1) = E_1 \wedge D_1$ and $T_1[D_1] = \bot$, there exists D such that $Eq(\mathcal{C}) = E \wedge D$, $T[D] = \bot$ and $L^1_{\mathcal{C}_1}(D_1) \leq L^1_{\mathcal{C}}(D)$. Thus we deduce that $\mu^1_{gen}(M_1, M_1') \leq \mu^1_{gen}(M, M')$.

Furthermore, by definition of the default transformation on the association tables, we have that if $T_1[D_1]$ exists then we know that $T[D]$ also exists and $T_1[D_1] = T[D]\Theta\downarrow$ where $\Theta = \{X \mapsto \mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C})}\}$. But we know $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C})} = \mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C}_1)} = \mathsf{path} \in \mathcal{F}^*_d \cdot \mathcal{AX}$. Assume that $\mathsf{path} = w \cdot ax_k$. Since $\mathcal{C}$ is a well-formed constraint system and thanks to Property 2 of a well-formed constraint system, we know that $k \leq j$. Hence we deduce that $\mathsf{paramC}^{\mathcal{C}_1}_{\mathsf{max}}(\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C})}) = k \leq j$. But since $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$, we have that $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X) = i$ with $j \leq i$. Thus we have that $\mathsf{paramC}^{\mathcal{C}_1}_{\mathsf{max}}(X\Theta) \leq \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X)$ which implies that $L^2_{\mathcal{C}_1}(T[D]\Theta) \leq L^2_{\mathcal{C}}(T[D])$. Thanks to Lemma C.55, we have that $L^2_{\mathcal{C}_1}(T[D]\Theta) = L^2_{\mathcal{C}_1}(T[D]\Theta\downarrow)$ and so $L^2_{\mathcal{C}_1}(T_1[D_1]) \leq L^2_{\mathcal{C}}(T[D])$. Thus we deduce that $\mu^2_{gen}(M_1, M_1') \leq \mu^2_{gen}(M, M')$. Since we proved that $\mu^1_{gen}(M_1, M_1') \leq \mu^1_{gen}(M, M')$, we can conclude that $\mu_{gen}(M_1, M_1') \leq \mu_{gen}(M, M')$.

Case Rule Eq-right-right(X, ξ): We first focus on (M_2, M_2'): Let $\mathcal{C}$ be a constraint system in (M, M') different from $\bot$ and T its association table. Let $\mathcal{C}_2$ the constraint system in (M_2, M_2') with T_2 its association table such that $\mathcal{C} \to \mathcal{C}_2$ by the rule Eq-right-right(X, ξ). According to Figure 7.2 and since $\mathcal{C}$ satisfies the invariant $\mathsf{InvVarConstraint}(\infty)$, we have that $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ with $x \in \mathcal{X}^1$. Furthermore, for all $Y \in vars^2(\xi)$, there exists $(Y, j \overset{?}{\vdash} y) \in D(\mathcal{C})$ with $y \in \mathcal{X}^1$, $j \leq i$ and $\mathsf{ind}_{\mathcal{C}}(y) = j$. At last, we have $Eq(\mathcal{C}_2) = (Eq(\mathcal{C}) \wedge x \overset{?}{\neq} v)\downarrow$ where $v = \xi\mathsf{acc}^1(()\mathcal{C})$. Note that if there exists E' and E'' such that $Eq(\mathcal{C}) = E' \wedge (E'' \vee x \overset{?}{\neq} v)$ then we have that $Eq(\mathcal{C}_2) = E'\downarrow \wedge x \overset{?}{\neq} v$. Furthermore, by definition of the addition transformation, we have that $T_2[x \overset{?}{\neq} v] = X \overset{?}{\neq} \xi$, i.e. $(X \overset{?}{\neq} \xi) \in \mathcal{S}(M_2, M_2')$. Since we proved those properties for any constraint system $\mathcal{C}$ in (M, M'), we can deduce that:

- Case (a): If there exists $\mathcal{C}$ in (M, M') and T its association table, E', E'' such that $Eq(\mathcal{C}) = E' \wedge (E'' \vee x \overset{?}{\neq} v)$ and $T[E'' \vee x \overset{?}{\neq} v] = \bot$ where $x = X\mathsf{acc}^1(\mathcal{C})$ and $v = \xi\mathsf{acc}^1(\mathcal{C})$ then the number of occurrence of $L^1_{\mathcal{C}}(E'' \vee x \overset{?}{\neq} v)$ strictly decrease between $\mu^1_{gen}(M, M')$ and $\mu^1_{gen}(M_2, M_2')$. Thus we have $\mu^1_{gen}(M_2, M_2') < \mu^1_{gen}(M, M')$;
- Case (a'): else $\mu^1_{gen}(M_2, M_2') = \mu^1_{gen}(M, M')$

Assume now that there exists E', E'', D such that $Eq(\mathcal{C}) = E' \wedge (E'' \vee x \overset{?}{\neq} v)$ and $T[E'' \vee x \overset{?}{\neq} v] = D \vee X \overset{?}{\neq} \xi$ with $x = X\mathsf{acc}^1(\mathcal{C})$ and $v = \xi\mathsf{acc}^1(\mathcal{C})$. We already have shown that $Eq(\mathcal{C}_2) = E'\downarrow \wedge x \overset{?}{\neq} v$. Hence by definition of the default transformation, we deduce that $T_2[E'' \vee x \overset{?}{\neq} v] = \bot$. Since we prove this property for any constraint system $\mathcal{C}$ in (M, M'), we can deduce that:

- Case (b): if there exists D such that $(D \vee X \overset{?}{\neq} \xi) \in \mathcal{S}(M, M')$ then $(D \vee X \overset{?}{\neq} \xi) \notin \mathcal{S}(M_2, M_2')$ and $(X \overset{?}{\neq} \xi) \in \mathcal{S}(M_2, M_2')$. But for all D, $L^2_{\mathcal{C}_2}(X \overset{?}{\neq} \xi) \leq L^2_{\mathcal{C}}(D \vee X \overset{?}{\neq} \xi)$. Thus, we have that $\mu^2_{gen}(M_2, M_2') \leq \mu^2_{gen}(M, M')$.
- Case (b'): else $\mathcal{S}(M_2, M_2') = \mathcal{S}(M, M') \cup \{X \overset{?}{\neq} \xi\}$ and so $\mu^2_{gen}(M_2, M_2') = \mu^2_{gen}(M, M') \cup \{\{L^2_{\mathcal{C}_2}(X \overset{?}{\neq} \xi)\}\}$.

If (M, M') satisfies Case (b) then we proved that $\mu^2_{gen}(M_2, M_2') \leq \mu^2_{gen}(M, M')$ and in both cases (a) or (a') we proved that $\mu^1_{gen}(M_2, M_2') \leq \mu^1_{gen}(M, M')$. Hence we conclude that $\mu_{gen}(M_2, M_2') \leq \mu_{gen}(M, M')$.

It remains to show that when (M, M') satisfies Case (b') then the result holds. Case (b') is satisfied when then there is no D such that $(D \vee X \overset{?}{\neq} \xi) \in \mathcal{S}(M, M')$. It implies that for all constraint system $\mathcal{C}$ in (M, M') and T its association table, for all E, $T[E \vee x \overset{?}{\neq} v] = \bot$ with $x = X\mathsf{acc}^1(\mathcal{C})$ and $v = \xi\mathsf{acc}^1(\mathcal{C})$. But according to the application condition of the rule EQ-RIGHT-RIGHT for Step b of Phase 2 of the strategy, it implies that there exists $\mathcal{C}$ in (M, M'), T its association table, E', E'' formulas such that $Eq(\mathcal{C}) = E' \wedge (E'' \vee x \overset{?}{\neq} v)$ and $T[E'' \vee x \overset{?}{\neq} v] = \bot$ with $x = X\mathsf{acc}^1(\mathcal{C})$ and $v = \xi\mathsf{acc}^1(\mathcal{C})$. Thus (M, M') satisfies Case (a). In such a case, we already have shown that the number of occurrence of $L^1_{\mathcal{C}}(E'' \vee x \overset{?}{\neq} v)$ strictly decrease between $\mu^1_{gen}(M, M')$ and $\mu^1_{gen}(M_2, M'_2)$. But $L^1_{\mathcal{C}}(x \overset{?}{\neq} v) \leq L^1_{\mathcal{C}}(E'' \vee x \overset{?}{\neq} v)$ and since all constraint system in (M, M') satisfy $\mathsf{InvVarConstraint}(\infty)$ and $\xi \in \mathcal{T}(\mathcal{F}_c, vars^2(\mathcal{C}))$, we also have that $L^2_{\mathcal{C}}(X \overset{?}{\neq} \xi) = L^1_{\mathcal{C}}(x \overset{?}{\neq} v) \leq L^1_{\mathcal{C}}(E'' \vee x \overset{?}{\neq} i)$. Hence we have that $\mu^1_{gen}(M_2, M'_2) \cup \{\{L^2_{\mathcal{C}}(X \overset{?}{\neq} \xi)\}\} \leq \mu^1_{gen}(M, M')$. Thus, $\mu^1_{gen}(M_2, M'_2) \cup \{\{L^2_{\mathcal{C}}(X \overset{?}{\neq} \xi)\}\} \cup \mu^2_{gen}(M_2, M'_2) \leq \mu^1_{gen}(M, M') \cup \mu^2_{gen}(M, M')$ and so we conclude that $\mu_{gen}(M_2, M'_2) \leq \mu_{gen}(M, M')$.

We now focus on (M_1, M'_1): Let $\mathcal{C}$ be a constraint system in (M, M') different from $\bot$ and T its association table. Let $\mathcal{C}_1$ the constraint system in (M_1, M'_1) with T_1 its association table such that $\mathcal{C} \to \mathcal{C}_1$ by the rule EQ-RIGHT-RIGHT(X, ξ). According to Figure 7.2 and since $\mathcal{C}$ satisfies the invariant $\mathsf{InvVarConstraint}(\infty)$, we have that $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ with $x \in \mathcal{X}^1$. Furthermore, for all $Y \in vars^2(\xi)$, there exists $(Y, j \overset{?}{\vdash} y) \in D(\mathcal{C})$ with $y \in \mathcal{X}^1$, $j \leq i$ and $\mathsf{ind}_{\mathcal{C}}(y) = j$. At last, we have $Eq(\mathcal{C}_1) = (Eq(\mathcal{C}) \wedge x \overset{?}{=} v)\downarrow$ and $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{X, i \overset{?}{\vdash} x\}$ where $v = \xi\mathsf{acc}^1(()\mathcal{C})$.

Hence, thanks to the normalisation, we deduce that for all disjunction D_1 of disjunction of first order term, if there exists E_1 such that $Eq(\mathcal{C}_1) = E_1 \wedge D_1$ then there exists D and E such that $Eq(\mathcal{C}) = E \wedge D$ and $D\sigma\downarrow = D_1$ where $\sigma = \{x \mapsto v\}$. But we proved that for all $Y \in vars^2(\xi)$, there exists $(Y, j \overset{?}{\vdash} y) \in D(\mathcal{C})$ with $y \in \mathcal{X}^1$, $j \leq i$ and $\mathsf{ind}_{\mathcal{C}}(y) = j$. Hence with $\xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$, we deduce that $\mathsf{ind}_{\mathcal{C}}(v) \leq \mathsf{ind}_{\mathcal{C}}(x)$. Since $X \notin vars^2(\xi)$ and $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{X, i \overset{?}{\vdash} x\}$, we also have that $\mathsf{ind}_{\mathcal{C}_1}(v) = \mathsf{ind}_{\mathcal{C}}(v)$ which implies that $L^1_{\mathcal{C}_1}(D\sigma) \leq L^1_{\mathcal{C}}(D)$. But, thanks to Lemma C.55, we have that $L^1_{\mathcal{C}_1}(D\sigma) = L^1_{\mathcal{C}_1}(D\sigma\downarrow)$ and so $L^1_{\mathcal{C}_1}(D_1) \leq L^1_{\mathcal{C}}(D)$.

Since only the default transformation on the association table is applied for (M_1, M'_1), we also have that $T_1[D_1] = \bot$ implies $T[D] = \bot$. Hence, we proved that for all D_1 such that $Eq(\mathcal{C}_1) = E_1 \wedge D_1$ and $T_1[D_1] = \bot$, there exists D such that $Eq(\mathcal{C}) = E \wedge D$, $T[D] = \bot$ and $L^1_{\mathcal{C}_1}(D_1) \leq L^1_{\mathcal{C}}(D)$. Thus we deduce that $\mu^1_{gen}(M_1, M'_1) \leq \mu^1_{gen}(M, M')$.

Furthermore, by definition of the default transformation on the association tables, we have that if $T_1[D_1]$ exists then we know that $T[D]$ also exists and $T_1[D_1] = T[D]\Theta\downarrow$ where $\Theta = \{X \mapsto \mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C})}\}$. But $\xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$ implies that $\mathsf{C}\lfloor\xi\rfloor_{\Phi(\mathcal{C})} = \xi$. Moreover, according to Figure 7.2, for all $Y \in vars^2(\xi)$, $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(Y) \leq \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X)$. Since $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{X, i \overset{?}{\vdash} x\}$ and $X \notin vars^2(\xi)$, we deduce that $\mathsf{paramC}^{\mathcal{C}_1}_{\mathsf{max}}(Y) = \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(Y)$ and so $\mathsf{paramC}^{\mathcal{C}_1}_{\mathsf{max}}(\xi) \leq \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X)$. Thus we have $L^2_{\mathcal{C}_1}(T[D]\Theta) \leq L^2_{\mathcal{C}}(T[D])$. Thanks to Lemma C.55, we have that $L^2_{\mathcal{C}_1}(T[D]\Theta) = L^2_{\mathcal{C}_1}(T[D]\Theta\downarrow)$ and so $L^2_{\mathcal{C}_1}(T_1[D_1]) \leq L^2_{\mathcal{C}}(T[D])$. Hence we deduce that $\mu^2_{gen}(M_1, M'_1) \leq \mu^2_{gen}(M, M')$. Since we proved that $\mu^1_{gen}(M_1, M'_1) \leq \mu^1_{gen}(M, M')$, we can conclude that $\mu_{gen}(M_1, M'_1) \leq \mu_{gen}(M, M')$.

Application of the removal transformations: To simplify the proof, for a pair of matrices (M, M'), we denote $(\tilde{M}, \tilde{M'})$ the pair of matrices obtained after applying all the removal transformation. Similarly, for an association table T in (M, M'), we denote $\tilde{T}$ the association table corresponding of T in $(\tilde{M}, \tilde{M'})$.

Let $\mathcal{C}$ be constraint system in (M, M') and its association table T such that $T[D] = \bigvee_j \beta_j \overset{?}{\neq} \beta'_j$ with $\beta_j \in (\mathcal{F}^*_d \cdot \mathcal{AX})$, for all j and for some D. In such a case, we have $(\bigvee_j \beta_j \overset{?}{\neq} \beta'_j) \in \mathcal{S}(M, M')$

which means that there exists a multiset $\mathcal{M}_2$ such that $\mu^2_{gen}(M, M') = \mathcal{M}_2 \cup \{\{L^2_{\mathcal{C}}(\bigvee_j \beta_j \overset{?}{\neq} \beta'_j)\}\}$. However according to the definition of the removal transformation, the entry is in fact removed, i.e. $\tilde{T}[D] = \bot$.

Since the removal transformation is applied on all association tables in (M, M'), we can deduce that $(\bigvee_j \beta_j \overset{?}{\neq} \beta'_j) \notin \mathcal{S}(\tilde{M}, \tilde{M'})$ and so $\mu^2_{gen}(\tilde{M}, \tilde{M'}) = \mathcal{M}_2$. On the other hand, if we denote $\mathcal{M}_1$ the multiset such that $\mathcal{M}_1 = \{\{L^1_{\mathcal{C}}(D) \mid T[D] = (\bigvee_j \beta_j \overset{?}{\neq} \beta'_j)$, for some D and some $\mathcal{C}$ in (M, M') with T its association table$\}\}$, we also have that $\mu^1_{gen}(\tilde{M}, \tilde{M'}) = \mu^1_{gen}(M, M') \cup \mathcal{M}_1$.

But thanks to Lemma 8.17, we know that for all $\mathcal{C}$ in (M, M') and T its association table, for all D, if $T[D] = (\bigvee_j \beta_j \overset{?}{\neq} \beta'_j)$ then $(\bigvee_j \beta_j \overset{?}{\neq} \beta'_j)\mathsf{acc}^1(\mathcal{C})\downarrow = D$. Hence thanks to Lemma C.55, we deduce that $L^1_{\mathcal{C}}(D) = L^1_{\mathcal{C}}(\bigvee_j \beta_j \mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \beta'_j \mathsf{acc}^1(\mathcal{C}))$. At last, thanks to Lemma 8.18 and $\beta_j \in (\mathcal{F}^*_d \cdot \mathcal{A}\mathcal{X})$ for all j, we deduce that $L^1_{\mathcal{C}}(D) < L^2_{\mathcal{C}}(\bigvee_j \beta_j \overset{?}{\neq} \beta'_j)$. Thus, it allows us to conclude that $\mathcal{M}_2 < \{\{L^2_{\mathcal{C}}(\bigvee_j \beta_j \overset{?}{\neq} \beta'_j)\}\}$ and so $\mu_{gen}(\tilde{M}, \tilde{M'}) < \mu_{gen}(M, M')$. □

C.7.3 Proofs of termination of each step of Phase 2 of the strategy

C.7.3.1 Termination of Step a of Phase 2

Lemma 8.20. *Let (M, M') be a pair of matrices of constraint systems obtained during Step a of Phase 2 of the strategy. Let $R(\tilde{p})$ be one instance of the rule* Axiom *or* Cons *such that $R(\tilde{p})$ is strongly applicable on at least one constraint system in M or M'. At last, let (M_1, M'_1) and (M_2, M'_2) be the two pairs of matrices of constraint systems obtained by application of $R(\tilde{p})$ on (M, M'). We have that:*

$$\mu_{2.a}(M_1, M'_1) < \mu_{2.a}(M, M') \quad \wedge \quad \mu_{2.a}(M_2, M'_2) < \mu_{2.a}(M, M')$$

Proof. Thanks to the normalisation, we know that for all constraint system $\mathcal{C}$ in M, the disjunctions of inequations in $Eq(\mathcal{C})$ are of the form $\forall \tilde{y}. \bigvee_i x_i \overset{?}{\neq} u_i$ where $\tilde{y}$ is a set of universal variable and x_i are not universal for any i. Assume that $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ and a rule is applied on this deducible constraint, i.e. $\textsc{Cons}(X, \mathsf{f})$ or $\textsc{Axiom}(X, \mathsf{path})$. Let's denote $\mathcal{C}_1$ and $\mathcal{C}_2$ the two constraint systems obtained by applying the rule on $\mathcal{C}$, i.e. $\mathcal{C}_1$ is in (M_1, M'_1) and $\mathcal{C}_2$ is in (M_2, M'_2).

We first focus on (M_1, M'_1): In the case of rule Cons, since $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(\infty)$, we deduce that x will be instantiated in $\mathcal{C}_1$ by $\mathsf{f}(x_1, \ldots, x_n)$ where x_k are not universal, for all $k \in \{1, \ldots, n\}$. In the case of rule Axiom, since $\mathcal{C}$ satisfies the invariant $\mathsf{InvNoUse}(\infty)$, we know that $(\xi, j \rhd u) \in \Phi(\mathcal{C})$ with $\mathsf{path}(\xi) = \mathsf{path}(\xi)$ implies that $u \notin \mathcal{X}^1$ (otherwise we would have $(\xi, j \rhd u) \in \mathsf{NoUse}(\mathcal{C})$ and so the rule $\textsc{Axiom}(X, \mathsf{path})$ would not be applicable).

Hence we have shown that x is necessary instantiated by a term different from a variable. We denote σ the substitution that instantiates x. Let's now look at the possible atomic statement that contains x:

1. Case $Eq(\mathcal{C}) = E' \wedge [\forall \tilde{y}.z.E'' \vee x \overset{?}{\neq} v]$ and $z \in vars^1(v)$: In such a case, we have that $Eq(\mathcal{C}_1) = E'\sigma\downarrow \wedge [\forall \tilde{y}.z.E''\sigma \vee x\sigma \overset{?}{\neq} v\sigma]\downarrow$. By hypothesis, we know that $Eq(\mathcal{C})$ is normalised. Hence we have that $v \neq z$ which means that $\mathsf{root}(v) \in \mathcal{F}_c$. Let's denote $v = \mathsf{g}(v_1, \ldots, v_n)$. Since $x \in vars^1(v)$, there exists k and a position p such that $z \in vars^1(v_k)$ and $v_k|_p = z$. This implies that $(k \cdot p) \in \pi_1(\mu_{2.a}(M, M'))$. Furthermore, we proved that $x\sigma \notin \mathcal{X}^1$, thus we can denote $x\sigma = \mathsf{f}(u_1, \ldots, u_m)$.

   If $\mathsf{g} = \mathsf{f}$ then $n = m$ and we have that $(x\sigma \overset{?}{\neq} v\sigma)\downarrow = u_1 \vee v_1 \vee \ldots \vee u_n \vee v_n$. Hence in such a case, we deduce that the same occurrence of z have the position $p \in \pi_1(\mu_{2.a}(M_1, M'_1))$.

   Else $\mathsf{g} \neq \mathsf{f}$. In such a case, we deduce that $E'\sigma\downarrow \wedge [\forall \tilde{y}.z.E''\sigma \vee x\sigma \overset{?}{\neq} v\sigma]\downarrow = E'\sigma\downarrow$. Thus, this specific occurrence of z in $Eq(\mathcal{C})$ is no longer in $Eq(\mathcal{C}_1)$.

2. Case $Eq(\mathcal{C}) = E' \wedge [\forall \tilde{y}.z.E'' \vee x' \stackrel{?}{\neq} v]$, $x' \neq x$ and $z \in vars^1(v)$: We already proved that $x\sigma$ do not contain an universal variable and $x \stackrel{?}{\neq} x'$ which implies that $x' \stackrel{?}{\neq} v\sigma)\downarrow = x' \stackrel{?}{\neq} v\sigma$. Hence if p is a position such that $v\sigma|_p = z$ then we also have that $v|_p = z$ which means that $p \in \pi_1(\mu_{2.a}(M, M'))$ and $p \in \pi_1(\mu_{2.a}(M_1, M_1'))$.
3. Case $Eq(\mathcal{C}) = E' \wedge [\forall \tilde{y}.E'' \vee x' \stackrel{?}{\neq} v]$ and $vars^1(v) \cap \tilde{y} = \emptyset$: Once again, we already proved that $x\sigma$ do not contain an universal variable hence $vars^1(v) \cap \tilde{y} = \emptyset$ implies that $vars^1(v\sigma) \cap \tilde{y} = \emptyset$

We have shown that the position of an occurrence of an universal variable either stay the same or decrease. More specifically, we have shown that in Case 1, the position necessary decrease or the occurrence is no longer in $Eq(\mathcal{C}_1)$. Since Case 1 corresponds to the application condition of the rule Axiom and Cons, we deduce that $\mu_{2.a}(M_1, M_1') < \mu_{2.a}(M, M')$.

We now focus on (M_2, M_2'): By definition of the rule Axiom and Cons, we know that the only difference between $\mathcal{C}$ and $\mathcal{C}_2$ is the second order formula Er, i.e. $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$ in the case of the rule $\textsc{Cons}(X, \mathsf{f})$; and $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge X \stackrel{?}{\neq} \xi$ in the case of the rule $\textsc{Axiom}(X, \mathsf{path})$ with $\mathsf{path}(\xi) = \mathsf{path}$ and $(\xi, j \rhd u) \in \Phi(\mathcal{C})$. Hence we trivially have that $\pi_k(\mu_{2.a}(M_2, M_2')) = \pi_k(\mu_{2.a}(M, M'))$ with $k = 1, 2$ and $\pi_3(\mu_{2.a}(M_2, M_2')) < \pi_3(\mu_{2.a}(M, M'))$ in the case of the rule Axiom; whereas we have $\pi_1(\mu_{2.a}(M_2, M_2')) = \pi_1(\mu_{2.a}(M, M'))$ and $\pi_2(\mu_{2.a}(M_2, M_2')) < \pi_2(\mu_{2.a}(M, M'))$ in the case of the rule Cons. We conclude that $\mu_{2.a}(M_2, M_2') < \mu_{2.a}(M, M')$. □

C.7.3.2 Termination of Step b of Phase 2

Definition C.3. *Let $\mathcal{C}$ be a well-formed constraint system satisfying* $\mathsf{InvVarConstraint}(\infty)$. *We say that $\mathsf{H}^1(x \stackrel{?}{\neq} v)$ exists if $\mathsf{root}(v) \in \mathcal{F}_c$ and either $st(v) \cap \mathcal{N} \neq \emptyset$ or $\mathsf{ind}_{\mathcal{C}}(x) < \mathsf{ind}_{\mathcal{C}}(v)$. Moreover, when $\mathsf{H}^1(x \stackrel{?}{\neq} v)$ exists, $\mathsf{H}^1(x \stackrel{?}{\neq} v) = (h_a, h_b)$ where $h_a = \max\{|p| \mid v|_p \in \mathcal{N}\}$ and $h_b = \max\{v|_p \in \mathcal{X}^1 \wedge \mathsf{ind}_{\mathcal{C}}(x) < \mathsf{ind}_{\mathcal{C}}(v|_p)\}$.*

We define similarly H^2 for context of recipes:

Definition C.4. *Let $\mathcal{C}$ be a well-formed constraint system satisfying* $\mathsf{InvVarConstraint}(\infty)$. *We say that $\mathsf{H}^2(X \stackrel{?}{\neq} \beta)$ exists if $\mathsf{root}(\beta) \in \mathcal{F}_c$ and either $st(\beta) \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) \neq \emptyset$ or $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X) < \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\beta)$. Moreover, when $\mathsf{H}^2(X \stackrel{?}{\neq} \beta)$ exists, $\mathsf{H}^2(X \stackrel{?}{\neq} \beta) = (h_a, h_b)$ where $h_a = \max\{|p| \mid \beta|_p \in (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X})\}$ and $h_b = \max\{\beta|_p \in \mathcal{X}^2 \wedge \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X) < \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\beta|_p)\}$.*

Lemma C.57. *Let $\mathcal{C}$ be a well-formed constraint system satisfying* $\mathsf{InvVarConstraint}(\infty)$. *Let $y \stackrel{?}{\neq} u$ be a inequation on term. Let $\sigma = \{x \mapsto v\}$ such that $x \notin vars^1(v)$, $\mathsf{ind}_{\mathcal{C}}(v) \leq \mathsf{ind}_{\mathcal{C}}(x)$ and $st(v) \cap \mathcal{N} = \emptyset$. If $\mathsf{mgu}(y\sigma \stackrel{?}{\neq} u\sigma)$ exists then $\bigvee_k^n z_k \stackrel{?}{\neq} t_k$ such that $(y \stackrel{?}{\neq} u)\sigma\downarrow = \bigvee_k y_k \stackrel{?}{\neq} t_k$. Moreover,*

- *if $\mathsf{ind}_{\mathcal{C}}(x) < \mathsf{ind}_{\mathcal{C}}(y)$ then*
  - *$(y \stackrel{?}{\neq} u)\sigma\downarrow = (y \stackrel{?}{\neq} u\sigma)$; and*
  - *$\mathsf{H}^1(y \stackrel{?}{\neq} u\sigma)$ exists implies $\mathsf{H}^1(y \stackrel{?}{\neq} u)$ exists and $\mathsf{H}^1(y \stackrel{?}{\neq} u\sigma) = \mathsf{H}^1(y \stackrel{?}{\neq} u)$;*
- *else if $\mathsf{ind}_{\mathcal{C}}(x) = \mathsf{ind}_{\mathcal{C}}(y)$ then for all k,*
  - *if $\mathsf{ind}_{\mathcal{C}}(y_k) > \mathsf{ind}_{\mathcal{C}}(y)$ then $\mathsf{H}^1(y_k \stackrel{?}{\neq} t_k)$ does not exist.*
  - *if $\mathsf{ind}_{\mathcal{C}}(y_k) = \mathsf{ind}_{\mathcal{C}}(y)$ and $\mathsf{H}^1(y_k \stackrel{?}{\neq} t_k)$ exists then $\mathsf{H}^1(y \stackrel{?}{\neq} u)$ exists. Moreover, if $x = y$ and $v \notin \mathcal{X}^1$ then $\mathsf{H}^1(y_k \stackrel{?}{\neq} t_k) < \mathsf{H}^1(y \stackrel{?}{\neq} u)$ else $\mathsf{H}^1(y_k \stackrel{?}{\neq} t_k) = \mathsf{H}^1(y \stackrel{?}{\neq} u)$ and $(y \neq u)\sigma\downarrow = (y\sigma \stackrel{?}{\neq} u\sigma)$.*

Proof. Since $\mathsf{mgu}(y\sigma \stackrel{?}{\neq} u\sigma)$ exists, we deduce that either $x \notin st(y,u)$ or $x \in st(u)$ or else $x = y$. In the first two cases, we deduce that $(y\sigma \stackrel{?}{\neq} u\sigma)\downarrow = (y \stackrel{?}{\neq} u\sigma)$. Moreover, since $\mathsf{ind}_{\mathcal{C}}(v) \leq \mathsf{ind}_{\mathcal{C}}(x)$ and $st(v) \cap \mathcal{N} = \emptyset$, we also deduce that if $\mathsf{H}^1(y \stackrel{?}{\neq} u\sigma)$ exists then so does $\mathsf{H}^1(y \stackrel{?}{\neq} u\sigma)$ and $\mathsf{H}^1(y \stackrel{?}{\neq} u) = \mathsf{H}^1(y \stackrel{?}{\neq} u\sigma)$.

Assume now that $x = y$ and $u = v$. In such a case, by the normalisation rule (Nt1), we deduce that $n = 0$ and so the result trivially holds.

At last, assume that $x = y$ and $u \neq v$. In such a case, since $\mathsf{mgu}(y\sigma \stackrel{?}{\neq} u\sigma)$ exists then the only normalisation rule that can be applied are (Nt1) and (Nsplit). Hence, thanks to $st(v) \cap \mathcal{N} = \emptyset$, we deduce that there exists $\bigvee_k y_k \stackrel{?}{\neq} t_k$ such that $(y \stackrel{?}{\neq} u)\sigma\downarrow = \bigvee_k^n y_k \stackrel{?}{\neq} t_k$ and for all $k \in \{1, \ldots, n\}$:

1. either $y_k \in st(v)$ and $t_k \in st(u)$;
2. or $y_k \in st(u)$ and $t_k \in st(v)$.

Let $k \in \{1 \ldots n\}$. Assume first that $\mathsf{ind}_{\mathcal{C}}(y_k) > \mathsf{ind}_{\mathcal{C}}(y)$. Since $x = y$ and $\mathsf{ind}_{\mathcal{C}}(v) \leq \mathsf{ind}_{\mathcal{C}}(x)$, we deduce that $y_k \notin st(v)$ and so $y_k \in st(u)$ and $t_k \in st(v)$. But $t_k \in st(v)$ and $\mathsf{ind}_{\mathcal{C}}(y_k) > \mathsf{ind}_{\mathcal{C}}(v)$ implies $\mathsf{ind}_{\mathcal{C}}(t_k) < \mathsf{ind}_{\mathcal{C}}(y_k)$. Moreover, by hypothesis, $st(v) \cap \mathcal{N} = \emptyset$ hence $st(t_k) \cap \mathcal{N} = \emptyset$. Thus we deduce that $\mathsf{H}^1(y_k \stackrel{?}{\neq} t_k)$ does not exists. Hence the result holds.

Assume now that $\mathsf{ind}_{\mathcal{C}}(y_k) = \mathsf{ind}_{\mathcal{C}}(y)$ and $\mathsf{H}^1(y_k \stackrel{?}{\neq} t_k)$ exists. Thus, we deduce that $\mathsf{root}(t_k) \in \mathcal{F}_c$ and either $st(t_k) \cap \mathcal{N} \neq \emptyset$ or $\mathsf{ind}_{y_k}(<)\mathsf{ind}_{t_k}()$. But we know that $\mathsf{ind}_{\mathcal{C}}(v) \leq \mathsf{ind}_{\mathcal{C}}(x) = \mathsf{ind}_{\mathcal{C}}(y) = \mathsf{ind}_{\mathcal{C}}(y_k)$ and $st(v) \cap \mathcal{N} = \emptyset$ hence in both cases, we deduce that $y_k \in st(v)$ and $t_k \in st(u)$. Furthermore, $t_k \in st(u)$ and $\mathsf{root}(t_k) \in \mathcal{F}_c$ implies that $\mathsf{root}(u) \in \mathcal{F}_c$. At last, $t_k \in st(u)$, $\mathsf{ind}_{\mathcal{C}}(y_k) = \mathsf{ind}_{\mathcal{C}}(y)$ and either $st(t_k) \cap \mathcal{N} \neq \emptyset$ or $\mathsf{ind}_{y_k}(<)\mathsf{ind}_{t_k}()$ imply that either $st(u) \cap \mathcal{N} \neq \emptyset$ or $\mathsf{ind}_y(<)\mathsf{ind}_u()$. Hence $\mathsf{H}^1(y \stackrel{?}{\neq} u)$ exists.

It remains to check the value of $\mathsf{H}^1(y \stackrel{?}{\neq} u)$. If $v \in \mathcal{X}^1$ then $t_k = u$ and so we deduce that $\mathsf{H}^1(y \stackrel{?}{\neq} u) = \mathsf{H}^1(y_k \stackrel{?}{\neq} u)$. Else $v \notin \mathcal{X}^1$ implies that $\mathsf{root}(v) = \mathsf{root}(u) \in \mathcal{F}_c$. In such a case, there is at leaf one application of the normalisation rule (Nsplit) and so the length of the position of each name or each variable in u strictly decrease. Hence we deduce that $\mathsf{H}^1(y_k \stackrel{?}{\neq} t_k) < \mathsf{H}^1(y \stackrel{?}{\neq} u)$. Thus the result holds. □

Lemma C.58. *Let $\mathcal{C}$ be a well-formed constraint system satisfying* $\mathsf{InvVarConstraint}(\infty)$. *Let $Y \stackrel{?}{\neq} \beta$ be a inequation on context of recipe. Let $\theta = \{X \mapsto \xi\}$ such that $X \notin vars^2(\xi)$, $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\xi) \leq \mathsf{param}^{\mathsf{X}}_{\mathsf{max}}(\mathcal{C})$ and $st(\xi) \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) = \emptyset$. If $\mathsf{mgu}(Y\theta \stackrel{?}{\neq} \beta\theta)$ exists then $\bigvee_k^n Y_k \stackrel{?}{\neq} \zeta_k$ such that $(Y \stackrel{?}{\neq} \beta)\theta\downarrow = \bigvee_k Y_k \stackrel{?}{\neq} \zeta_k$. Moreover,*

- *if $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(x) < \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(y)$ then*
  - *$(Y \stackrel{?}{\neq} \beta)\theta\downarrow = (Y \stackrel{?}{\neq} \beta\theta)$; and*
  - *$\mathsf{H}^2(Y \stackrel{?}{\neq} \beta\theta)$ exists implies $\mathsf{H}^2(Y \stackrel{?}{\neq} \beta)$ exists and $\mathsf{H}^2(Y \stackrel{?}{\neq} \beta\theta) = \mathsf{H}^2(Y \stackrel{?}{\neq} \beta)$;*
- *else if $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(X) = \mathsf{param}^{\mathsf{Y}}_{\mathsf{max}}(\mathcal{C})$ then for all k,*
  - *if $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(Y_k) > \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(Y)$ then $\mathsf{H}^2(Y_k \stackrel{?}{\neq} \zeta_k)$ does not exist.*
  - *if $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(Y_k) = \mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(Y)$ and $\mathsf{H}^2(Y_k \stackrel{?}{\neq} \zeta_k)$ exists then $\mathsf{H}^2(Y \stackrel{?}{\neq} \beta)$ exists. Moreover, if $X = Y$ and $\xi \notin \mathcal{X}^2$ then $\mathsf{H}^2(Y_k \stackrel{?}{\neq} \zeta_k) < \mathsf{H}^2(Y \stackrel{?}{\neq} \beta)$ else $\mathsf{H}^2(Y_k \stackrel{?}{\neq} \zeta_k) = \mathsf{H}^2(Y \stackrel{?}{\neq} \beta)$ and $(Y \neq \beta)\theta\downarrow = (Y\theta \stackrel{?}{\neq} \beta\theta)$.*

Proof. The proof is similar to the proof of Lemma C.57. □

Lemma 8.21. *Let (M, M') be a pair of matrices of constraint systems obtained during Step b of the second phase. Let $R(\tilde{p})$ be an applicable rule and (M_1, M_1') and (M_2, M_2') the two pairs of matrices of constraint systems obtained by application of $R(\tilde{p})$ on (M, M'). We have that:*

$$\mu_{2.b}(M_1, M_1') < \mu_{2.b}(M, M') \quad \textit{and} \quad \mu_{2.b}(M_2, M_2') < \mu_{2.b}(M, M')$$

Proof. Thanks to Lemma 8.19, we know that $\mu_{gen}(M_1, M_1') \leq \mu_{gen}(M, M')$ and $\mu_{gen}(M_2, M_2') \leq \mu_{gen}(M, M')$. Furthermore, this lemma also tells us that for all $i \in \{1, 2\}$, if a removal transformation is applied on (M_i, M_i') then $\mu_{gen}(M_i, M_i') < \mu_{gen}(M, M')$ which implies that $\mu_{2.b}(M_i, M_i') < \mu_{2.b}(M, M')$. Hence we will assume from now on that no removal transformation is applied. We do a case analysis on the rule applied and the pair of matrices we consider:

Case rule $\textsc{Cons}(X, \mathsf{f})$ and (M_2, M_2'): Let $\mathcal{C}$ be a constraint system in (M, M') different from $\bot$ and T its association table. Let $\mathcal{C}_2$ be the constraint system in (M_2, M_2') and T_2 its association table such that $\mathcal{C} \to \mathcal{C}_2$. According to Figure 7.1 and the strategy, there exists $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ and $Er(\mathcal{C}) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$. Furthermore, the only difference between $\mathcal{C}_2$ and $\mathcal{C}$ is that $Er(\mathcal{C}_2) = Er(\mathcal{C}) \wedge \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$.

Hence the default transformation on association table does not modify T which means that $T = T_2$ and so $\pi_k(\mu_{2.b}(M_2, M_2')) = \pi_k(\mu_{2.b}(M, M'))$, for $k = 2, 3$. Furthermore, we also have that for all $k \in \{1, \ldots, n\} \setminus \{i\}$, $\mu^k_{var}(M_2, M_2') = \mu^k_{var}(M, M')$, $\mu^k_{cons}(M_2, M_2') = \mu^k_{cons}(M, M')$ and $\mu^i_{var}(M_2, M_2') = \mu^i_{var}(M, M')$. Moreover, for all $\mathsf{g} \in \mathcal{F}_c$, $Er(\mathcal{C}_2) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$ implies that $Er(\mathcal{C}) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{g}$. Thus we deduce that $\mu^i_{cons}(M_2, M_2') \leq \mu^i_{cons}(M_2, M_2')$ and so $\pi_4(\mu_{2.b}(M_2, M_2')) \leq \pi_4(\mu_{2.b}(M, M'))$. At last, since $Er(\mathcal{C}) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}_2) \vDash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$, we deduce that $\pi_5(\mu_{2.b}(M_2, M_2')) < \pi_5(\mu_{2.b}(M, M'))$ and so we have $\mu_{2.b}(M_2, M_2') < \mu_{2.b}(M, M')$.

Case rule $\textsc{Cons}(X, \mathsf{f})$ and (M_1, M_1'): Let $\mathcal{C}$ be a constraint system in (M, M') different from $\bot$ and T its association table. Let $\mathcal{C}_1$ be the constraint system in (M_1, M_1') and T_1 its association table such that $\mathcal{C} \to \mathcal{C}_1$. According to Figure 7.1 and the strategy, there exists $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ and $Er(\mathcal{C}) \nvdash \mathsf{root}(X) \overset{?}{\neq} \mathsf{f}$. Furthermore, we have that $Er(\mathcal{C}_1) = Er(\mathcal{C}) \wedge X \overset{?}{=} \mathsf{f}(X_1, \ldots, X_m)$, $Eq(\mathcal{C}_1) = (Eq(\mathcal{C}) \wedge x \overset{?}{=} \mathsf{f}(x_1, \ldots, x_m))\downarrow$ and $D(\mathcal{C}_1) = D(\mathcal{C}) \setminus \{X, i \overset{?}{\vdash} x\} \cup \{X_k, i \overset{?}{\vdash} x_k\}_{k=1..m}$ where $X_1, x_1, \ldots, X_m, x_m$ are fresh variables.

Thanks to the normalisation, we deduce that for all disjunction D_1 of first order term, if there exists E_1 such that $Eq(\mathcal{C}_1) = E_1 \wedge D_1$ then there exists D and E such that $Eq(\mathcal{C}) = E \wedge D$ and $D\sigma\downarrow = D_1$ where $\sigma = \{x \mapsto \mathsf{f}(x_1, \ldots, x_m)\}$. But we assumed that only the default transformation on association table is applied and so we have $T_1[D_1] = \bot$ is equivalent to $T[D] = \bot$. Thus we deduce that $\pi_2(\mu_{2.b}(M_1, M_1')) \leq \pi_2(\mu_{2.b}(M, M'))$. Moreover, since $T_1[D_1] = T[D]\{X \mapsto \mathsf{f}(X_1, \ldots, X_m)\}\downarrow$, then and $st(T_1[D_1]) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$ is equivalent to $st(T[D]) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$. Hence we deduce that $\pi_3(\mu_{2.b}(M_1, M_1')) \leq \pi_3(\mu_{2.b}(M, M'))$.

We first show that for all $j > i$, $\mu^j_{cons,1}(M, M')$. Let D_1 be a disjunction on first order terms. Let $y_1 \in \mathcal{X}^1$ and u_1 a term such that $Eq(\mathcal{C}_1) = E_1 \wedge (D_1 \vee y_1 \overset{?}{\neq} u_1)$, $T_1[D_1 \vee y_1 \overset{?}{\neq} u_1] = \bot$, $j = \mathsf{ind}_{\mathcal{C}_1}(y_1) > i$ and $\mathsf{H}^1(y_1 \overset{?}{\neq} u_1)$ exists. Since we only consider the default transformation on association table, we deduce that there exists D and E such that $Eq(\mathcal{C}) = E \wedge D$ such that $D\sigma\downarrow = D_1 \vee y_1 \neq u_1$ where $\sigma = \{x \mapsto \mathsf{f}(x_1, \ldots, x_n)\}$.

But $\mathcal{C}$ and $\mathcal{C}_1$ satisfy $\mathsf{InvVarConstraint}(\infty)$. Thus, thanks to Lemma C.57 and since $\mathsf{ind}_{\mathcal{C}}(x) < \mathsf{ind}_{\mathcal{C}_1}(y_1)$ and $\mathsf{H}^1(y_1 \overset{?}{\neq} u_1)$ exists, we deduce that there exists D', y, u such that $D = D' \vee y \overset{?}{\neq} u$,

$(y \overset{?}{\neq} u)\sigma\downarrow = (y \overset{?}{\neq} u\sigma) = (y_1 \neq u_1)$ and $\mathsf{H}^1(y \overset{?}{\neq} u) = \mathsf{H}^1(y_1 \overset{?}{\neq} u_1)$. At last, $\mathcal{C}$ being normalised and satisfying $\mathsf{InvVarConstraint}(\infty)$ also implies that there exists $(Y, j \overset{?}{\vdash} y_1) \in D(\mathcal{C}_1)$ and so $(Y, j \overset{?}{\vdash} y) \in D(\mathcal{C})$. Thus $Er(\mathcal{C}_1) \vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{root}(u_1)$ implies that $Er(\mathcal{C}_1) \vDash \mathsf{root}(Y) \overset{?}{\neq} \mathsf{root}(u)$. We conclude that $\mathsf{H}^1(y_1 \overset{?}{\neq} u_1) \in \mu^j_{cons,1}(M_1, M_1')$ implies that $\mathsf{H}^1(y \overset{?}{\neq} u) \in \mu^j_{cons,1}(M, M')$ and so $\mu^j_{cons,1}(M_1, M_1') \leq \mu^j_{cons,1}(M, M')$.

Using a similar proof and relying on Lemma C.58, we can show that $\mu^j_{cons,2}(M_1, M_1') \leq \mu^j_{cons,2}(M, M')$ and so $\mu^j_{cons}(M_1, M_1') \leq \mu^{M,M'}_{cons}()$.

We now show that $\mu^i_{cons}(M_1, M_1') < \mu^i_{cons}(M, M')$. Let D_1 be a disjunction on first order terms. Let $y_1 \in \mathcal{X}^1$ and u_1 a term such that $Eq(\mathcal{C}_1) = E_1 \wedge (D_1 \vee y_1 \overset{?}{\neq} u_1)$, $T_1[D_1 \vee y_1 \overset{?}{\neq} u_1] = \bot$, $j = \mathsf{ind}_{\mathcal{C}_1}(y_1) = i$ and $\mathsf{H}^1(y_1 \overset{?}{\neq} u_1)$ exists. Since we only consider the default transformation on association table, we deduce that there exists D and E such that $Eq(\mathcal{C}) = E \wedge D$ such that $D\sigma\downarrow = D_1 \vee y_1 \neq u_1$ where $\sigma = \{x \mapsto \mathsf{f}(x_1, \ldots, x_n)\}$.

Thanks to Lemma C.57 and since $\mathsf{ind}_{\mathcal{C}}(x) = \mathsf{ind}_{\mathcal{C}_1}(y_1)$ and $\mathsf{H}^1(y_1 \overset{?}{\neq} u_1)$ exists, we deduce that there exists D', y, u such that $D = D' \vee y \overset{?}{\neq} u$ and $(y \overset{?}{\neq} u)\sigma\downarrow = \bigvee_k y_k \overset{?}{\neq} u_k$. Moreover, since $\sigma = \{x \mapsto \mathsf{f}(x_1, \ldots, x_n\}$, we deduce that $\mathsf{H}^1(y \overset{?}{\neq} u)$ exists and if $x = y$ then $\mathsf{H}^1(y_1 \overset{?}{\neq} t_1) < \mathsf{H}^1(y \overset{?}{\neq} u)$ else $\mathsf{H}^1(y_1 \overset{?}{\neq} t_1) = \mathsf{H}^1(y \overset{?}{\neq} u)$, $y\sigma = y_1$ and $u\sigma = u_1$.

Thus, we deduce that $\mu^i_{cons,1}(M_1, M_1') \leq \mu^i_{cons,1}(M, M')$ and if there exists one inequation in $\mathcal{C}$ of the form $x \overset{?}{\neq} v$ where $\mathsf{H}^1(x \overset{?}{\neq} v)$ exists then $\mu^i_{cons,1}(M_1, M_1') < \mu^i_{cons,1}(M, M')$.

Similarly, relying on Lemma C.58, we can show that $\mu^i_{cons,2}(M_1, M_1') \leq \mu^i_{cons,2}(M, M')$ and if there exists one inequation in T of the form $X \overset{?}{\neq} \beta$ where $\mathsf{H}^2(X \overset{?}{\neq} \beta)$ exists then $\mu^i_{cons,2}(M_1, M_1') < \mu^i_{cons,2}(M, M')$.

But according to the application condition for $\textsc{Cons}(X, \mathsf{f})$, there exists at least one constraint system $\mathcal{C}$ in (M, M') and T its association table that satisfy one of these two conditions. Hence, we deduce that $\mu^i_{cons}(M_1, M_1') < \mu^i_{cons}(M, M')$.

At last, since we trivially have that $\mu^j_{var}(M_1, M_1') = \mu^j_{var}(M, M')$ for all $j > i$, we conclude that $\mu_{2.b}(M_1, M_1') < \mu_{2.b}(M, M')$.

Case rule $\textsc{Eq-right-right}(X, \xi)$ and (M_2, M_2'): Let $\mathcal{C}$ be a constraint system in (M, M') different from $\bot$ and T its association table. Let $\mathcal{C}_2$ be the constraint system in (M_2, M_2') and T_2 its association table such that $\mathcal{C} \to \mathcal{C}_2$. According to Figure 7.1, there exists $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$. Furthermore, the only difference between $\mathcal{C}_2$ and $\mathcal{C}$ is that $Eq(\mathcal{C}_2) = (Eq(\mathcal{C}) \wedge X\mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \xi\mathsf{acc}^1(\mathcal{C}))\downarrow$. Moreover, $T_2[X\mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \xi\mathsf{acc}^1(\mathcal{C})] = X \overset{?}{\neq} \xi$. But we know that $\xi \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$ thus $st(X \overset{?}{\neq} \xi) \cap (\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) = \emptyset$. Hence we deduce that $\pi_k(\mu_{2.b}(M_2, M_2')) < \pi_k(\mu_{2.b}(M, M'))$, for $k = 1, 2$.

But according to the application condition for $\textsc{Eq-right-right}(X, \xi)$, there exists at least one constraint system $\mathcal{C}$ in (M, M') and T its association table such that $Eq(\mathcal{C}) = E' \wedge (E'' \vee X\mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \xi\mathsf{acc}^1(\mathcal{C}))$ and

— either (a) $T[E'' \vee X\mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \xi\mathsf{acc}^1(\mathcal{C})] = \bot$

— or else (b) $T[E'' \vee X\mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \xi\mathsf{acc}^1(\mathcal{C})] = D \vee X \overset{?}{\neq} \xi$ with $(\mathcal{F}_d^* \cdot \mathcal{A}\mathcal{X}) \cap st(D) \neq \emptyset$.

But $Eq(\mathcal{C}_1) = (Eq(\mathcal{C}) \wedge X\mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \xi\mathsf{acc}^1(\mathcal{C}))\downarrow$ and so by the normalisation rule (Nd), we have $Eq(\mathcal{C}_1) = E' \wedge X\mathsf{acc}^1(\mathcal{C}) \overset{?}{\neq} \xi\mathsf{acc}^1(\mathcal{C})$. Hence in Case (a), we deduce that $\pi_2(\mu_{2.b}(M_2, M_2')) <$

$\pi_2(\mu_{2.b}(M, M'))$ whereas in Case (b), $\pi_3(\mu_{2.b}(M_2, M_2')) < \pi_3(\mu_{2.b}(M, M'))$. Thus in both cases, we deduce that $\mu_{2.b}(M_2, M_2') < \mu_{2.b}(M, M')$.

Case rule Eq-right-right(X, ξ) and (M_1, M_1'): Let $\mathcal{C}$ be a constraint system in (M, M') different from $\bot$ and T its association table. Let $\mathcal{C}_1$ be the constraint system in (M_1, M_1') and T_1 its association table such that $\mathcal{C} \to \mathcal{C}_1$. According to the definition of the rule and since $\mathcal{C}$ satisfies $\mathsf{InvVarConstraint}(\infty)$, there exists $(X, i \overset{?}{\vdash} x) \in D(\mathcal{C})$ such that $\mathsf{paramC}^{\mathcal{C}}_{\mathsf{max}}(\xi) \leq i$, $\xi\mathsf{acc}^1(\mathcal{C}) \in \mathcal{T}(\mathcal{F}_c, \mathcal{X}^1)$ and $\mathsf{ind}_{\mathcal{C}}(\xi\mathsf{acc}^1()) \leq i$. Moreover, $Eq(\mathcal{C}_1) = Eq(\mathcal{C})\sigma\downarrow \wedge x \overset{?}{=} \xi\mathsf{acc}^1(\mathcal{C})$ and $Er(\mathcal{C}_1) = Er(\mathcal{C})\theta\downarrow \wedge X \overset{?}{=} \xi$ where $\sigma = \{x \mapsto \xi\mathsf{acc}^1(\mathcal{C})\}$ and $\theta = \{X \mapsto \xi\}$.

Thanks to the normalisation, we deduce that for all disjunction D_1 of first order term, if there exists E_1 such that $Eq(\mathcal{C}_1) = E_1 \wedge D_1$ then there exists D and E such that $Eq(\mathcal{C}) = E \wedge D$ and $D\sigma\downarrow = D_1$. But we assumed that only the default transformation on association table is applied and so we have $T_1[D_1] = \bot$ is equivalent to $T[D] = \bot$. Thus we deduce that $\pi_2(\mu_{2.b}(M_1, M_1')) \leq \pi_2(\mu_{2.b}(M, M'))$. Moreover, since $T_1[D_1] = T[D]\{X \mapsto \xi\}\downarrow$ and $st(\xi) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$, then $st(T_1[D_1]) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$ is equivalent to $st(T[D]) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$. Hence we deduce that $\pi_3(\mu_{2.b}(M_1, M_1')) \leq \pi_3(\mu_{2.b}(M, M'))$.

Once again, we can rely on Lemmas C.57 and C.58. Hence using a similar proof to the one for the rule Cons, we deduce that $\mu^j_{cons}(M_1, M_1') \leq \mu^j_{cons}(M, M')$ for all $j \geq i$. Since the deducible constraint $(X, i \overset{?}{\vdash} x)$ is removed in $D(\mathcal{C}_1)$, we can also deduce that $\mu^i_{var}(M_1, M_1') < \mu^i_{var}(M, M')$ and so $\mu_{2.b}(M_1, M_1') < \mu_{2.b}(M, M')$. □

C.7.3.3 Termination of Step c of Phase 2

Lemma 8.22. *Let (M, M') be a pair of matrices of constraint systems obtained during Step c of the second phase. Let* Axiom$(\tilde{p})$ *be an applicable rule and (M_1, M_1') and (M_2, M_2') the two pairs of matrices of constraint systems obtained by application of* Axiom$(\tilde{p})$ *on (M, M'). We have that:*

$$\mu_{2.c}(M_1, M_1') < \mu_{2.c}(M, M') \quad \textit{and} \quad \mu_{2.c}(M_2, M_2') < \mu_{2.c}(M, M')$$

Proof. Direct from the definition of the rule Axiom. □

C.7.3.4 Termination of the cycle of steps b and c of Phase 2

Lemma 8.23. *Let (M, M') be a pair of matrices of constraint systems obtained at the end of Step b of the second phase. Assume that there exists (M_1, M_1') obtained at the end of the next Step b of the second phase such that $(M, M') \to^* (M_1, M_1')$. At last, assume that there exists (M_0, M_0') obtained at the end of step c of the second phase such that $(M_1, M_1') \to^* (M_0, M_0')$. In such a case, $\mu_{gen}(M_0, M_0') < \mu_{gen}(M, M')$.*

Proof. Since $(M, M') \to^* (M_1, M_1')$ and both are obtained at the end of Step b, we deduce that there exists (M_2, M_2') obtained at the end of Step c such that $(M, M') \to^* (M_2, M_2') \to^* (M_1, M_1')$. First of all, thanks to Lemma 8.19, if a removal transformation was applied on the association table, we directly deduce that $\mu_{gen}(M_0, M_0') < \mu_{gen}(M, M')$. Hence for the rest of this proof, we will assume that no removal transformation is applied.

Consider $\mathcal{C}$ (resp $\mathcal{C}_1$, $\mathcal{C}_2$) a constraint system in (M, M') (resp. (M_1, M_1'), (M_2, M_2')) and T (resp. T_1, T_2) its association table. Assume that $\mathcal{C} \to^* \mathcal{C}_2 \to^* \mathcal{C}_1$.

Let $\bigvee_i u_i \overset{?}{\neq} v_i$ such that $Eq(\mathcal{C}) = E \wedge \bigvee_i u_i \overset{?}{\neq} v_i$ for some E, and $T[\bigvee_i u_i \neq v_i] = \bot$. Thanks to Lemma C.33, we know that for all i, $u_i \overset{?}{\neq} v_i$ satisfies one of the following properties:

1. $u_i \in \mathcal{X}^1$ and $v_i \in \mathcal{N}$
2. $u_i, v_i \in \mathcal{X}^1$, $Er(\mathcal{C}) \vDash \mathsf{root}(X_i) \overset{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}) \nvDash \mathsf{root}(Y_i) \overset{?}{\neq} \mathsf{g}$, for all $\mathsf{f}, \mathsf{g} \in \mathcal{F}_c$, where $(X_i, p \overset{?}{\vdash} u_i), (Y_i, q \overset{?}{\vdash} v_i) \in D(\mathcal{C})$:

3. $u_i \in \mathcal{X}^1$, $\mathsf{root}(v_i) \in \mathcal{F}_c$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$, where $(X_i, p \stackrel{?}{\vdash} u_i) \in D(\mathcal{C})$

We can assume that $\mathcal{C}_1$ and $\mathcal{C}_2$ are different from $\bot$ otherwise the result trivially holds. However, since in case 1 $v_i \in \mathcal{N}$ and in cases 2,3 we have for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(X_i) \stackrel{?}{\neq} \mathsf{f}$, we deduce that the rules $\textsc{Axiom}(X_i, \mathsf{path})$ are applied during step c, for all i. Hence, since we assume that $\mathcal{C}_1 \neq \bot$, we deduce that either $(\bigvee_i u_i \stackrel{?}{\neq} v_i)\mathsf{mgu}(Eq(\mathcal{C}_2))\downarrow$ is true or else each $u_i \in \mathrm{dom}(Eq(\mathsf{mgu}(\mathcal{C}_2)))$ for all i, *i.e.* each u_i are instantiate by the rule axiom.

However, by applying $\textsc{Axiom}(X_i, \mathsf{path})$, we know that there exists a frame element $(\xi, j \rhd t)$ such that $j \leq i$ and $\mathsf{path}(\xi) = \mathsf{path}$. Moreover, by the origination property of a constraint system, we know that $\mathsf{ind}_{\mathcal{C}}(t) < j \leq i$. Hence thanks to Lemma C.54 and since each u_i are instantiated by a term t_i such that $\mathsf{ind}_{\mathcal{C}}(t_i) < \mathsf{ind}_{\mathcal{C}}(u_i)$, we deduce that $L^1_{\mathcal{C}}(\bigvee_i u_i \neq v_i) < L^1_{\mathcal{C}_2}((\bigvee_i u_i \neq v_i)\mathsf{mgu}(Eq(\mathcal{C}_2)))$. Hence we conclude that $\mu_{gen}(M_2, M_2') < \mu_{gen}(M, M')$ and so $\mu_{gen}(M_0, M_0') < \mu_{gen}(M, M')$.

Let $\bigvee_j \xi_j \stackrel{?}{\neq} \xi_j$ such that $T[\bigvee_i u_i \stackrel{?}{\neq} v_i] = \bigvee_i \xi_i \stackrel{?}{\neq} \xi_i'$. Once again thanks to Lemma C.33, we know that either for all i, $st(\xi_i, \xi_i') \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$ or else for all i, $\xi_i \stackrel{?}{\neq} \xi_i'$ satisfies one of the following properties:

1. $\xi_i \in (\mathcal{F}_d^* \cdot \mathcal{AX})$
2. $\xi_i, \xi_i' \in \mathcal{X}^2$, for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$ and $Er(\mathcal{C}) \nvDash \mathsf{root}(\xi_i') \stackrel{?}{\neq} \mathsf{f}$
3. $\xi_i \in \mathcal{X}^2$, $\mathsf{root}(\xi_i') \in \mathcal{F}_c$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$

Assume first that there exists i such that $st(\xi_i, \xi_i') \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$. Since in cases 2 and3 we have $\xi_i \in \mathcal{X}^2$ and for all $\mathsf{f} \in \mathcal{F}_c$, $Er(\mathcal{C}) \vDash \mathsf{root}(\xi_i) \stackrel{?}{\neq} \mathsf{f}$, we deduce that the rules $\textsc{Axiom}(\xi_i, \mathsf{path})$ are applied during step c, for all i and all path. But we consider that $\mathcal{C}_2 \neq \bot$. Hence thanks to the normalisation rule (Nnosol), we deduce that each ξ_i that satisfied cases 2 and 3 are instantiated in $\mathcal{C}_2$ by the rule $\textsc{Axiom}(\xi_i, \mathsf{path})$. But in such a case, we would have that $T_2[(\bigvee_i u_i \stackrel{?}{\neq} v_i)\mathsf{mgu}(Eq(\mathcal{C}_2))\downarrow] = \bigvee_j \zeta_j \stackrel{?}{\neq} \zeta_j'$ where for all j, $\zeta_j \in (\mathcal{F}_d^* \cdot \mathcal{AX})$. Hence the removal transformation would have been applied which is in contradiction with our hypothesis. Hence we deduce that for all i, $st(\xi_i, \xi_i') \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$.

Thus, can assume from now on that for all constraint system $\mathcal{C}$ in M or M', for all $\bigvee_i u_i \stackrel{?}{\neq} v_i$ in $Eq(\mathcal{C})$, there exists D such that $T[\bigvee_i u_i \stackrel{?}{\neq} v_i] = D$ and $st(D) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) = \emptyset$. (We already corvered all the other cases). Since we assumed that $(M, M') \to^* (M_2, M_2')$, it implies that some instances of the rules Axiom was applied between (M, M') and (M_2, M_2') (otherwise, we would have that the strategy would have stop at (M, M')).

But the according to the application conditions of the rule Axiom, it implies that there exists $\mathcal{C}$ in (M, M') and $\mathsf{f} \in \mathcal{F}_c$ such that $(X, i \stackrel{?}{\vdash} x) \in D(\mathcal{C})$ and $Er(\mathcal{C}) \vDash \mathsf{root}(X) \stackrel{?}{\neq} \mathsf{f}$. Since we assumed that $(M_2, M_2') \to^* (M_1, M_1')$, we can deduce that there exists a disjunction $E \vee x \stackrel{?}{\neq} u$ such that $T[E \vee x \stackrel{?}{\neq} u] = D \vee X \stackrel{?}{\neq} \beta$ (otherwise no rule Cons and Eq-right-right would be applicable on (M_2, M_2')). But by application of the rule $\textsc{Axiom}(X, \mathsf{path})$, then we deduce that $st(T_2[(E \vee x \stackrel{?}{\neq} u)\mathsf{mgu}(Eq(\mathcal{C}_2))\downarrow]) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$. Since the rules Cons and Eq-right-right only instantiate the association tables by context of recipes in $\mathcal{T}(\mathcal{F}_c, \mathcal{X}^2)$, we deduce that $st(T_1[(E \vee x \stackrel{?}{\neq} u)\mathsf{mgu}(Eq(\mathcal{C}_1))\downarrow]) \cap (\mathcal{F}_d^* \cdot \mathcal{AX}) \neq \emptyset$. Since (M_0, M_0') is obtained at the end of Step c and $(M_1, M_1') \to^* (M_0, M_0')$, by using a similar proof as previously, we prove that $\mu_{gen}(M_0, M_0') < \mu_{gen}(M_1, M_1')$ and so $\mu_{gen}(M_0, M_0') < \mu_{gen}(M, M')$. □

Appendix D

ProVerif

D.1 Equivalence proofs

We say that a biprocess P is uniform when $\mathsf{fst}(P) \to Q_1$ implies that $P \to Q$ for some biprocess Q with $\mathsf{fst}(Q) \equiv Q_1$, and symmetrically for $\mathsf{snd}(P) \to Q_2$. For the proof of Lemma 9.3, we rely on [BAF08, Theorem 1].

Lemma 9.3. *Let P_0 be a closed biprocess. Suppose that, for all plain evaluation contexts C, all evaluation contexts C', and all reductions $C[P_0] \to^* P$,*

1. *if $P \equiv C'[\mathsf{out}(N, M).Q \mid \mathsf{in}(N', x).R]$, then $\mathsf{fst}(N) =_\Sigma \mathsf{fst}(N')$ if, and only if, $\mathsf{snd}(N) =_\Sigma \mathsf{snd}(N')$; and*
2. *if $P \equiv C'[\mathsf{let}\ x = D\ \mathsf{in}\ Q\ \mathsf{else}\ R]$, then $\mathsf{fst}(D) \downarrow_\Sigma \mathsf{fail}$ if, and only if, $\mathsf{snd}(D) \downarrow_\Sigma \mathsf{fail}$.*

Then P_0 satisfies observational equivalence.

Proof. We show that P is uniform, then we conclude by [BAF08, Theorem 1]. Let us show that, if $\mathsf{fst}(P) \to P_1'$ then there exists a biprocess P' such that $P \to P'$ and $\mathsf{fst}(P') \equiv P_1'$. The case for $\mathsf{snd}(P) \to P_2'$ is symmetric.

By induction on the derivation of $\mathsf{fst}(P) \to P_1'$, we first show that there exist C, Q, and Q_1' such that $P \equiv C[Q]$, $P_1' \equiv \mathsf{fst}(C)[Q_1']$, and $\mathsf{fst}(Q) \to Q_1'$ using one of the four process rules (Red I/O), (Red Fun 1), (Red Fun 2), or (Red Repl): every step in this derivation trivially commutes with fst, except for structural steps that involve a parallel composition and a restriction, in case $a \in \mathit{fnames}(P)$ but $a \notin \mathit{fnames}(\mathsf{fst}(P))$. In that case, we use a preliminary renaming from a to some fresh $a' \notin \mathit{fnames}(P)$.

For each of these four rules, relying on a hypothesis of Corollary 9.3, we find Q' such that $\mathsf{fst}(Q') = Q_1'$ and $Q \to Q'$ using the corresponding biprocess rule:

(Red I/O): We have $Q = \mathsf{out}(N, M).R \mid \mathsf{in}(N', x).R'$ with $\vdash \mathsf{fst}(N) =_\Sigma \mathsf{fst}(N')$ and $Q_1' = \mathsf{fst}(R) \mid \mathsf{fst}(R')\{\mathsf{fst}(M)/x\}$. For $Q' = R \mid R'\{M/x\}$, we have $\mathsf{fst}(Q') = Q_1'$ and, by hypothesis 1, $\mathsf{snd}(N) =_\Sigma \mathsf{snd}(N')$, hence $Q \to Q'$.

(Red Fun 1): We have $Q = \mathsf{let}\ x = D\ \mathsf{in}\ R\ \mathsf{else}\ R'$ with $\mathsf{fst}(D) \downarrow_\Sigma M_1$ and $Q_1' = \mathsf{fst}(R)\{M_1/x\}$. By hypothesis 2 and Lemma 9.2, $\mathsf{snd}(D) \downarrow_\Sigma M_2$ for some M_2. We take $Q' = R\{\mathsf{diff}[M_1, M_2]/x\}$, so that $\mathsf{fst}(Q') = Q_1'$ and $Q \to Q'$.

(Red Fun 2): We have $Q = \mathsf{let}\ x = D\ \mathsf{in}\ R\ \mathsf{else}\ R'$ with $\mathsf{fst}(D) \downarrow_\Sigma \mathsf{fail}$ and $Q_1' = \mathsf{fst}(R')$. By hypothesis 2, $\mathsf{snd}(D) \downarrow_\Sigma \mathsf{fail}$. We obtain $Q \to Q'$ for $Q' = R'$.

(Red Repl): We have $Q =! R$ and $Q_1' = \mathsf{fst}(R) \mid ! \mathsf{fst}(R)$. We take $Q' = R \mid ! R$, so that $\mathsf{fst}(Q') = Q_1'$ and $Q \to Q'$.

To conclude, we take the biprocess $P' = C[Q']$ and the reduction $P \to P'$. □

D.1.1 Lemmas for modelling the equational theory

The following lemma shows the soundness of $D' \downarrow'_{\Sigma'} (M', \sigma', \phi')$ with respect to $D \downarrow_{\Sigma'} M$. We say that a term evaluation is plain if it does not contain diff. Similarly, we define a plain message and plain may-fail message.

Lemma D.1. *Let σ be a closed substitution.*

Let D be a plain term evaluation. If $D\sigma \downarrow_{\Sigma'} U$, then there exist U', σ_1, ϕ, and σ'_1 such that $D \downarrow' (U', \sigma_1, \phi)$, $U = U'\sigma'_1$, $\sigma = \sigma_1\sigma'_1$ except on fresh variables introduced in the computation of $D \downarrow' (U', \sigma_1, \phi)$, and $\sigma'_1 \vDash \phi$.

Let $D_1, \ldots, D_n$ be plain term evaluations. If for all $i \in \{1, \ldots, n\}$, $D_i\sigma \downarrow_{\Sigma'} M_i$, then there exist $U'_1, \ldots, U'_n$, σ_1, ϕ and σ'_1 such that $(D_1, \ldots, D_n) \downarrow' ((U'_1, \ldots, U'_n), \sigma_1, \phi)$, $U_i = U'_i\sigma'_1$ for all $i \in \{1, \ldots, n\}$, $\sigma = \sigma_1\sigma'_1$ except on fresh variables introduced in the computation of $(D_1, \ldots, D_n) \downarrow' ((U'_1, \ldots, U'_n), \sigma_1, \phi)$, and $\sigma'_1 \vDash \phi$.

Proof. The proof is by mutual induction following the definition of $\downarrow'$.

— Case $D = U'$: Take $\sigma_1 = \emptyset$, $\sigma'_1 = \sigma$ and $\phi = \top$. Since $U = U'\sigma$ and $\sigma'_1 \vDash \top$, we have the result.

— Case $D = \text{eval } \mathsf{h}(D_1, \ldots, D_n)$: Since $\text{eval } \mathsf{h}(D_1\sigma, \ldots, D_n\sigma) \downarrow_{\Sigma'} U$, there exist $\mathsf{h}(V_1, \ldots, V_n) \to V \mid\mid \phi'$ in $\text{def}_{\Sigma'}(\mathsf{h})$ and σ_m such that $D_i\sigma \downarrow_{\Sigma'} V_i\sigma_m$, $U = V\sigma_m$ and $\sigma_m \vDash \phi'$.

By induction hypothesis, there exist U'_i, ϕ'', σ_1, and σ'_1 such that $V_i\sigma_m = U'_i\sigma'_1$ for all $i \in \{1, \ldots, n\}$, $(D_1, \ldots, D_n) \downarrow' ((U'_1, \ldots, U'_n), \sigma_1, \phi'')$, $\sigma = \sigma_1\sigma'_1$ except on fresh variables introduced in the computation of $(D_1, \ldots, D_n) \downarrow' ((U'_1, \ldots, U'_n), \sigma_1, \phi'')$, and $\sigma'_1 \vDash \phi''$.

Let σ_u be the most general unifier of U'_i and V_i for $i \in \{1, \ldots, n\}$. (The substitution σ_u exists since $V_i\sigma_m = U'_i\sigma'_1$.) Then $\text{eval } \mathsf{h}(D_1, \ldots, D_n) \downarrow' (V\sigma_u, \sigma_1\sigma_u, \phi'\sigma_u \wedge \phi''\sigma_u)$. The substitution that maps variables of V_i, V as σ_m and other variables as σ'_1 is a unifier of U'_i and V_i, so there exists σ''_1 such that $\sigma_m = \sigma_u\sigma''_1$ on variables of V_i, V, and $\sigma'_1 = \sigma_u\sigma''_1$ on other variables. With $\sigma_m \vDash \phi'$ and $\sigma'_1 \vDash \phi''$, it also implies that $\sigma''_1 \vDash \phi'\sigma_u \wedge \phi''\sigma_u$.

Then $V\sigma_u\sigma''_1 = V\sigma_m = U$ and $\sigma_1\sigma_u\sigma''_1 = \sigma_1\sigma'_1 = \sigma$ except on fresh variables introduced in the computation of $(D_1, \ldots, D_n) \downarrow' ((U'_1, \ldots, U'_n), \sigma_1, \phi'')$ and variables of $V_1, \ldots, V_n, V$, that is, fresh variables introduced in the computation of $D \downarrow' (V\sigma_u, \sigma_1\sigma_u, \phi'\sigma_u \wedge \phi''\sigma_u)$.

— Case $(D_1, \ldots, D_n)$: We have, for all i in $\{1, \ldots, n\}$, $D_i\sigma \downarrow_{\Sigma'} U_i$.

By induction hypothesis, there exist U'_i, ϕ, σ_1, and σ'_1 such that $U_i = U'_i\sigma'_1$ for all $i \in \{1, \ldots, n-1\}$, $(D_1, \ldots, D_{n-1}) \downarrow' ((U'_1, \ldots, U'_{n-1}), \sigma_1, \phi)$, $\sigma = \sigma_1\sigma'_1$ except on fresh variables introduced in the computation of $(D_1, \ldots, D_{n-1}) \downarrow' ((U'_1, \ldots, U'_{n-1}), \sigma_1, \phi)$, and $\sigma'_1 \vDash \phi$.

Then $D_n\sigma = D_n\sigma_1\sigma'_1$, so $(D_n\sigma_1)\sigma'_1 \downarrow_{\Sigma'} U_n$. So by induction hypothesis, there exist U'_n, ϕ', σ_2, and σ'_2 such that $D_n\sigma_1 \downarrow' (U'_n, \sigma_2, \phi')$, $U_n = M'_n\sigma'_2$, $\sigma'_1 = \sigma_2\sigma'_2$ except on fresh variables introduced in the computation of $D_n\sigma_1 \downarrow' (U'_n, \sigma_2, \phi')$, and $\sigma'_2 \vDash \phi'$. With $\sigma'_1 \vDash \phi$, we deduce that $\sigma_2\sigma'_2 \vDash \phi$ and so $\sigma'_2 \vDash \phi\sigma_2$.

Hence $(D_1, \ldots, D_n) \downarrow' ((U'_1\sigma_2, \ldots, U'_{n-1}\sigma_2, U'_n), \sigma_1\sigma_2, \phi\sigma_2 \wedge \phi')$, $\sigma'_2 \vDash \phi\sigma_2$, $U_i = U'_i\sigma'_1 = (U'_i\sigma_2)\sigma'_2$ for all $i \in \{1, \ldots, n-1\}$, $U_n = U'_n\sigma'_2$, and $\sigma = \sigma_1\sigma'_1 = \sigma_1\sigma_2\sigma'_2$ except on fresh variables introduced in the computation of $(D_1, \ldots, D_n) \downarrow' ((U'_1\sigma_2, \ldots, U'_{n-1}\sigma_2, U'_n), \sigma_1\sigma_2, \phi\sigma_2 \wedge \phi')$. □

For the following lemmas, w represents a variable that is either a message variable or a may-fail message variable.

Lemma D.2. *Let U' a plain ground may-fail message, σ a closed substitution and U a plain may-fail message. Assume that $U' =_\Sigma U\sigma$. We have:*

— *if $U' = \mathsf{fail}$ then* $\text{addeval}(U)\sigma \downarrow_{\Sigma'} U'$

— *if U' is a message and* $\text{nf}_{\mathcal{S},\Sigma}(\{U'\} \cup \{w\sigma \mid w \in \mathit{fvars}(U)\})$ *then* $\text{addeval}(U)\sigma \downarrow_{\Sigma'} U'$.

Proof. Assume first that $U' = \mathsf{fail}$. In such a case, U can be either a may-fail variable or fail. In both cases, $\mathrm{addeval}(U) = U$ and so $\mathrm{addeval}(U)\sigma = U\sigma$. Moreover, $U' = \mathsf{fail}$ and $U' =_\Sigma U\sigma$ imply that $U' = U\sigma$ and so $\mathrm{addeval}(U)\sigma \downarrow_{\Sigma'} U'$.

Assume now that U' is a message. The proof is by induction on U.

— Case $U = \mathsf{fail}$: Impossible

— Case $U = w$: We have $w\sigma =_\Sigma U\sigma =_\Sigma U'$. Since $\mathrm{nf}_{\mathcal{S},\Sigma}(\{w\sigma, U'\})$, $w\sigma = U'$. Moreover, $\mathrm{addeval}(U)\sigma = w\sigma \downarrow_{\Sigma'} w\sigma = U'$.

— Case $U = a$: Since $U' ='_\Sigma U\sigma$ and $\mathrm{nf}_{\mathcal{S},\Sigma}(\{U'\})$, we have $U' = a$ by [BAF08, Lemma 4], so $\mathrm{addeval}(U)\sigma = a \downarrow_{\Sigma'} a = U'$.

— Case $U = f(M_1, \ldots, M_n)$: We have $U' =_\Sigma M\sigma =_\Sigma f(M_1\sigma, \ldots, M_n\sigma)$ and $\mathrm{nf}_{\mathcal{S},\Sigma}(\{U'\} \cup \{w\sigma \mid w \in \mathit{fvars}(M)\})$. By Property S2, there exist $M'_1, \ldots, M'_n$ such that $M_i\sigma =_\Sigma M'_i$ and $\mathrm{nf}_{\mathcal{S},\Sigma}(\{U', M'_1, \ldots, M'_n\} \cup \{w\sigma \mid w \in \mathit{fvars}(U)\})$. By Property S4, there exist $f(N_1, \ldots, N_n) \to N$ in $\mathrm{def}_{\Sigma'}(f)$ and σ' such that $U' = N\sigma'$ and $N_i\sigma' = M'_i$ for all $i \in \{1, \ldots, n\}$. By induction hypothesis, $\mathrm{addeval}(M_i)\sigma \downarrow_{\Sigma'} M'_i = N_i\sigma'$ for all $i \in \{1, \ldots, n\}$. By definition of $\downarrow_{\Sigma'}$, $\mathrm{addeval}(U)\sigma = \mathrm{eval}\, f(\mathrm{addeval}(M_1)\sigma, \ldots, \mathrm{addeval}(M_n)\sigma) \downarrow_{\Sigma'} N\sigma' = U'$. □

The following lemma shows the soundness of the rewrite rules of h in Σ' with respect to these rewrite rules in Σ. When h is a destructor, this is proved using the previous two lemmas, and when h is a constructor, this follows from the definition of "Σ' models Σ".

Lemma D.3. *Let $U_1, \ldots, U_n, U$ ground may-fail messages and let $\mathcal{U}$ the set of messages of U_1, $\ldots, U_n, U$. If $\mathsf{h}(V_1, \ldots, V_n) \to V \mid\mid \phi$ is in $\mathrm{def}_\Sigma(\mathsf{h})$, $U_i =_\Sigma V_i\sigma$ for all $i \in \{1, \ldots, n\}$, $U =_\Sigma V\sigma$, $\sigma \vDash \phi$, and $\mathrm{nf}_{\mathcal{S},\Sigma}(\mathcal{U})$, then there exist $\mathsf{h}(V'_1, \ldots, V'_n) \to V' \mid\mid \phi'$ in $\mathrm{def}_{\Sigma'}(\mathsf{h})$ and σ' such that $U_i = V'_i\sigma'$ for all $i \in \{1, \ldots, n\}$, $U = V'\sigma'$ and $\sigma' \vDash \phi'$.*

Proof. Case 1: h is a constructor in Σ. We do a case analysis on U:

— U is a message: In such a case, by definition of a destructor $U_1, \ldots, U_n$ are also messages and $V = \mathsf{h}(V_1, \ldots, V_n)$. Hence we have $U =_\Sigma \mathsf{h}(V_1, \ldots, V_n)\sigma =_\Sigma \mathsf{h}(U_1, \ldots, U_n)$. The result follows from Property S4.

— $U = \mathsf{fail}$: Otherwise, there exists $i \in \{1, \ldots, n\}$ such that $U_i = \mathsf{fail}$. We know that $\mathsf{h}(u_1, \ldots, u_{i-1}, \mathsf{fail}, u_{i+1}, \ldots, u_n) \to \mathsf{fail}$ is in $\mathrm{def}_{\Sigma'}(\mathsf{h})$. Hence, the result holds with σ' such that $u_j\sigma' = U_j$, for all $j \neq i$.

Case 2: h is a destructor in Σ. By Property S2, there exists σ_0 such that $u\sigma_0 = u\sigma$ for all $u \in \mathit{fvars}(V_1, \ldots, V_n, V)$ and $\mathrm{nf}_{\mathcal{S},\Sigma}(\mathcal{U} \cup \{u\sigma_0 \mid u \in \mathit{fvars}(V_1, \ldots, V_n, V)$ and $u\sigma_0$ is a message$\})$. So $U =_\Sigma V\sigma_0$, $U_i =_\Sigma V_i\sigma_0$ for all $i \in \{1, \ldots, n\}$ and $\sigma_0 \vDash \phi$. By Lemma D.2, $\mathrm{addeval}(V)\sigma_0 \downarrow_{\Sigma'} U$ and $\mathrm{addeval}(V_i)\sigma_0 \downarrow_{\Sigma'} U_i$ for all $i \in \{1, \ldots, n\}$. By Lemma D.1, there exist $V'_1, \ldots, V'_n$, V', σ_1, ϕ and σ' such that $\mathrm{addeval}(V_1, \ldots, V_n, V) \downarrow' ((V'_1, \ldots, V'_n, V'), \sigma_1, \phi')$, $V'_i\sigma' = U_i$ for all $i \in \{1, \ldots, n\}$, $V'\sigma' = U$, $\sigma_0 = \sigma_1\sigma'$ and $\sigma' \vDash \phi'$. Then $\mathsf{h}(V'_1, \ldots, V'_n) \to V' \mid\mid \phi\sigma \wedge \phi'$ is in $\mathrm{def}_{\Sigma'}(\mathsf{h})$, $V'_i\sigma' = U_i$ for all $i \in \{1, \ldots, n\}$, and $V'\sigma' = U$. Moreover, we know that $\sigma_0 \vDash \phi$ hence $\sigma' \vDash \phi\sigma_1$ and so $\sigma' \vDash \phi\sigma_1 \wedge \phi'$. Thus the result holds. □

We define the function removeeval such that $\mathrm{removeeval}(D) = M$ where D is a term evaluation that contains no destructor, and M is the term obtained by removing any eval before the function symbols of D.

Lemma D.4. *Assume that D is a plain term evaluation that contains no destructor. If $D \downarrow' (U, \sigma, \phi)$ then $\phi = \top$ and $\mathrm{removeeval}(D)\sigma =_\Sigma U$.*

Assume that $D_1, \ldots, D_n$ are plain term evaluations that contain no destructor. If we have $(D_1, \ldots, D_n) \downarrow' ((U_1, \ldots, U_n), \sigma, \phi)$ then $\phi = \top$ and $\mathrm{removeeval}(D_i)\sigma =_\Sigma U_i$ for all $i \in \{1, \ldots, n\}$.

Proof. The proof is by mutual induction following the definition of $\downarrow'$.

— Case $D = U$: We have $\sigma = \emptyset$ and $\phi = \top$, so $U\sigma =_\Sigma U$.

— Case $D = \text{eval}\ \mathsf{f}(D_1, \ldots, D_n)$: We have $\text{eval}\ \mathsf{f}(D_1, \ldots, D_n) \downarrow' (V\sigma_u, \sigma'\sigma_u, \phi'\sigma_u \wedge \phi\sigma_u)$ where $(D_1, \ldots, D_n) \downarrow' ((U_1, \ldots, U_n), \sigma', \phi')$, f is a constructor in Σ, $\mathsf{f}(V_1, \ldots, V_n) \to V \mid\mid \phi$ is in $\text{def}_{\Sigma'}(\mathsf{f})$ (with new variables), and σ_u is the most general unifier of (U_1, V_1), …, (U_n, V_n). By definition of $\text{def}_{\Sigma'}(\mathsf{f})$, we know that either $V = \mathsf{fail}$ or V is a message; and $\phi = \top$.

Assume first that V is a message. In such a case, $V_1, \ldots, V_n$ are messages too and by Property S3, $\mathsf{f}(V_1, \ldots, V_n) =_\Sigma V$. By induction hypothesis, $\text{removeeval}(D_i)\sigma' =_\Sigma U_i$ for all $i = 1 \ldots n$ and $\phi' = \top$. Moreover we have $U_i\sigma_u =_\Sigma V_i\sigma_u$. Hence we obtain $\text{removeeval}(\text{eval}\ \mathsf{f}(D_1, \ldots, D_n))\sigma' =_\Sigma \mathsf{f}(\text{removeeval}(D_1)\sigma'\sigma_u, \ldots, \text{removeeval}(D_n)\sigma'\sigma_u) =_\Sigma f(U_1\sigma_u, \ldots, U_n\sigma_u) =_\Sigma f(V_1\sigma_u, \ldots, V_n\sigma_u) =_\Sigma V\sigma_u$.

— Case $(D_1, \ldots, D_n)$: We have $(D_1, \ldots, D_n) \downarrow' ((U_1\sigma', \ldots, U_{n-1}\sigma', U_n), \sigma\sigma', \phi\sigma' \wedge \phi')$ where $(D_1, \ldots, D_{n-1}) \downarrow' ((U_1, \ldots, U_{n-1}), \sigma, \phi)$ and $D_n\sigma \downarrow' (U_n, \sigma', \phi')$. Then by induction hypothesis, $\phi' = \top$, $\phi = \top$, $\text{removeeval}(D_i)\sigma =_\Sigma U_i$ for $i \in \{1, \ldots, n-1\}$ and $\text{removeeval}(D_n\sigma)\sigma' =_\Sigma U_n$. Hence, $\text{removeeval}(D_i)\sigma\sigma' =_\Sigma U_i\sigma'$ for $i \in \{1, \ldots, n-1\}$ and $\text{removeeval}(D_n)\sigma\sigma' = U_n$. □

The following lemma shows a completeness property: we do not lose precision by translating computation in Σ into computations in Σ'. The proof of Lemma D.5 relies on Lemma D.4 for destructor applications.

Lemma D.5. *If* $\mathsf{h}(U_1, \ldots, U_n) \to U \mid\mid \phi$ *is in* $\text{def}_{\Sigma'}(\mathsf{h})$ *then there exists* $\mathsf{h}(U'_1, \ldots, U'_n) \to U' \mid\mid \phi'$ *in* $\text{def}_\Sigma(\mathsf{h})$ *and* σ *such that* $U_i =_\Sigma U'_i\sigma$ *for all* $i \in \{1, \ldots, n\}$, $U =_\Sigma U'\sigma$ *and for all* σ', $\sigma' \vDash \phi$ *implies* $\sigma\sigma' \vDash \phi'$.

Proof. Case 1: h is a constructor in Σ. According to the definition of $\text{def}_{\Sigma'}(\mathsf{h})$, $\phi = \top$ and either $U = \mathsf{fail}$ or U is a message.

— $U = \mathsf{fail}$: By definition of $\text{def}_\Sigma(\mathsf{h})$, we have $U_i = U'_i$, for all $i = 1 \ldots n$, $U = U'$ and $\phi' = \top$.

— U is a message: In such a case, $U_1, \ldots, U_n$ are all messages and by Property S3, we have $\mathsf{h}(U_1, \ldots, U_n) =_\Sigma U$. Let σ be defined by $x_i\sigma = U_i$ for all $i \in \{1, \ldots, n\}$, $U'_i = x_i$ for all $i \in \{1, \ldots, n\}$, $U' = \mathsf{h}(x_1, \ldots, x_n)$ and $\phi' = \top$. We have $\mathsf{h}(U'_1, \ldots, U'_n) \to U' \mid\mid \phi'$ in $\text{def}_\Sigma(\mathsf{h})$ because $\mathsf{h}(x_1, \ldots, x_n) \to \mathsf{h}(x_1, \ldots, x_n) \mid\mid \top$ is in $\text{def}_\Sigma(\mathsf{h})$. We also have $U_i =_\Sigma U'_i\sigma$ for all $i \in \{1, \ldots, n\}$, $U =_\Sigma \mathsf{h}(U_1, \ldots, U_n) =_\Sigma U'\sigma$, and for all σ', $\sigma\sigma' \vDash \top$

Case 2: h is a destructor in Σ. Then there exists $\mathsf{h}(U'_1, \ldots, U'_n) \to U' \mid\mid \phi'$ in $\text{def}_\Sigma(\mathsf{h})$, such that $\text{addeval}(U'_1, \ldots, U'_n, U') \downarrow' ((U_1, \ldots, U_n, U), \sigma, \phi'')$ and $\phi = \phi'\sigma \wedge \phi''$. By definition of the destructors of Σ, we know that $U'_1, \ldots, U'_n, U'$ do not contain any destructor. Hence by Lemma D.4, $U =_\Sigma U'\sigma$, for all $i \in \{1, \ldots, n\}$, $U_i =_\Sigma U'_i\sigma$ and $\phi'' = \top$. Hence, for all σ', $\sigma' \vDash \phi$ implies $\sigma' \vDash \phi'\sigma$ which implies $\sigma\sigma \vDash \phi'$. □

D.1.2 Proof of Lemmas 9.6 and 9.7

Lemma 9.7 is a consequence of the following lemma.

Lemma D.6. *Let* g *a destructor of* Σ. $\text{def}_\Sigma(\mathsf{g})$ *does not satisfy Property P2 if and only if there exist* $U_1, \ldots, U_n$ *ground may-fail messages, two pair of rewrite rules* $\mathsf{g}(V_1, \ldots, V_n) \to V \mid\mid \phi$ *and* $\mathsf{g}(V'_1, \ldots, V'_n) \to V' \mid\mid \phi'$ *in* $\text{def}_{\Sigma'}(\mathsf{g})$, *two substitution* σ, σ' *such that:*

— $U_i = V_i\sigma = V'_i\sigma'$ *for all* $i \in \{1, \ldots, n\}$

— $V\sigma = \mathsf{fail}$ *and* $V'\sigma'$ *is a message;*

— $\sigma \vDash \phi$ *and* $\sigma' \vDash \phi'$

Proof. We start by proving the right implication of the result. $\text{def}_\Sigma(\mathsf{g})$ does not satisfy Property P2 implies that there exists $U_1, \ldots, U_n$ ground may-fail messages such that $\mathsf{g}(U_1, \ldots, U_n) \to \mathsf{fail}$ and $\mathsf{g}(U_1, \ldots, U_n) \to M$ for some message M. Hence there exists $\mathsf{g}(V_1, \ldots, V_n) \to V \mid\mid \phi$ and $\mathsf{g}(V'_1, \ldots, V'_n) \to V' \mid\mid \phi'$ in $\text{def}_\Sigma(\mathsf{g})$ and two substitutions σ, σ' such that:

— $U_i =_\Sigma V_i\sigma =_\Sigma V'_i\sigma'$ for all $i \in \{1, \ldots, n\}$

— $V\sigma = \mathsf{fail}$ and $V'\sigma' = M$;

— $\sigma \vDash \phi$ and $\sigma' \vDash \phi'$

Thanks to Property S2, there exists $U'_1, \ldots, U'_n, M'$ such that $U'_i =_\Sigma U_i$ for all $i \in \{1, \ldots, n\}$, $M' =_\Sigma M$ and if we denote $\mathcal{U}$ is the set of messages of $U'_1, \ldots, U'_n, M'$, then $\mathrm{nf}_{\mathcal{S},\Sigma}(\mathcal{U})$.

By application of Lemma D.3 on the two rewrite rules, $(U'_1, \ldots, U'_n, M')$ and $(U'_1, \ldots, U'_n, \mathsf{fail})$, we have that there exist $\mathsf{g}(V_1^1, \ldots, V_n^1) \to V^1 \mid\mid \phi_1$ and $\mathsf{g}(V_1'^1, \ldots, V_n'^1) \to V'^1 \mid\mid \phi'_1$ in $\mathrm{def}_{\Sigma'}(\mathsf{g})$, two substitutions σ_1, σ'_1 such that:

— $U'_i = V_i^1\sigma_1 = V_i'^1\sigma'_1$ for all $i \in \{1, \ldots, n\}$

— $V_1^\sigma = \mathsf{fail}$ and $V'\sigma'_1 = M'$;

— $\sigma_1 \vDash \phi_1$ and $\sigma'_1 \vDash \phi'_1$

Hence the result holds.

We now prove the left implication of the result. We have $U_1, \ldots, U_n$ ground may-fail messages, $\mathsf{g}(V_1, \ldots, V_n) \to V \mid\mid \phi$ and $\mathsf{g}(V'_1, \ldots, V'_n) \to V' \mid\mid \phi'$ in $\mathrm{def}_{\Sigma'}(\mathsf{g})$ and two substitutions σ, σ' such that: two substitution σ, σ' such that:

— $U_i = V_i\sigma = V'_i\sigma'$ for all $i \in \{1, \ldots, n\}$

— $V\sigma = \mathsf{fail}$ and $V'\sigma'$ is a message;

— $\sigma \vDash \phi$ and $\sigma' \vDash \phi'$

By Lemma D.5, we deduce that there exist $\mathsf{g}(V_1^1, \ldots, V_n^1) \to V^1 \mid\mid \phi_1$ and $\mathsf{g}(V_1'^1, \ldots, V_n'^1) \to V'^1 \mid\mid \phi'_1$ in $\mathrm{def}_\Sigma(\mathsf{g})$, and two substitution σ_1, σ'_1 such that:

— $V_i =_\Sigma V_i^1\sigma_1$ and $V'_i =_\Sigma V_i'^1\sigma'_1$ for all $i \in \{1, \ldots, n\}$. Thus we deduce that $U_i =_\Sigma V_i^1\sigma_1\sigma =_\Sigma V_i'^1\sigma'_1\sigma'$

— $V =_\Sigma V^1\sigma_1$ and $V' =_\Sigma V'^1\sigma'_1$. Thus we deduce $\mathsf{fail} = V^1\sigma_1\sigma$ and $V'^1\sigma'_1\sigma'$ is a message.

— for all σ_0, $\sigma_0 \vDash \phi$ implies $\sigma_1\sigma_0 \vDash \phi_1$; and $\sigma_0 \vDash \phi'$ implies $\sigma'_1\sigma_0 \vDash \phi'_1$. Thus we deduce $\sigma_1\sigma \vDash \phi_1$ and $\sigma'_1\sigma' \vDash \phi'_1$.

We can deduce that $\mathsf{g}(U_1, \ldots, U_n) \to \mathsf{fail}$ and $\mathsf{g}(U_1, \ldots, U_n) \to V'^1\sigma'_1\sigma'$ where $V'^1\sigma'_1\sigma'$ is a message and so we conclude that $\mathrm{def}_\Sigma(\mathsf{g})$ do no satisfy Property P2. □

Lemma 9.6 is a consequence of the following lemma.

Lemma D.7. *Let* g *a destructor of* Σ *or arity* n*. Let* $x_1, \ldots, x_n$ *fresh message variables.* $\mathrm{def}_\Sigma(\mathsf{g})$ *does not satisfy Property P1 if and only if there exists* $(U_1, \ldots, U_n)$*, a set* $\mathcal{I}$ *and a closed substitution* σ *such that:*

— $U_i = \mathsf{fail}$*, for all* $i \notin \mathcal{I}$*; and* $U_i = x_i$*, for all* $i \in \mathcal{I}$

— *for all* $\mathsf{g}(V_1, \ldots, V_n) \to V \mid\mid \phi$ *in* $\mathrm{def}_\Sigma(\mathsf{g})$*, if* σ_u *is the most general unifier of* $(V_1, \ldots, V_n)$ *and* $(U_1, \ldots, U_n)$ *then* $\sigma \vDash \forall \tilde{z}.\big[\bigvee_{i\in\mathcal{I}} x_i \neq_\Sigma V_i\sigma_u \vee \neg\phi\sigma_u\big]$ *where* $\tilde{z}$ *are the variables of* $V_1\sigma_u, \ldots, V_n\sigma_u$*.*

Proof. Assume first that $\mathrm{def}_\Sigma(\mathsf{g})$ does not satisfy Property P2. In such a case, there exists $U'_1, \ldots, U'_n$ ground may-fail messages such that for all $\mathsf{g}(V_1, \ldots, V_n) \to V \mid\mid \phi$ in $\mathrm{def}_\Sigma(\mathsf{g})$, if there exists a substitution σ' such that $V_i\sigma' =_\Sigma U'_i$ for $i = 1 \ldots n$ then $\sigma' \nvDash \phi$.

But since U'_i is ground for all $i \in \{1, \ldots, n\}$ then either $U'_i = \mathsf{fail}$ or U'_i is a message. Let $\mathcal{I}$ be the subset of $\{1, \ldots, n\}$ such that for all $i \in \mathcal{I}$, $U'_i = \mathsf{fail}$ and for all $i \notin \mathcal{I}$, U'_i is a message. From $\mathcal{I}$ we define $U_1, \ldots, U_n$ as expected. At last, we define $\sigma = \{x_i \mapsto U'_i \mid i \in \mathcal{I}\}$

Let $\mathsf{g}(V_1, \ldots, V_n) \to V \mid\mid \phi$ in $\mathrm{def}_\Sigma(\mathsf{g})$ such that σ_u is the most general unifier of $(V_1, \ldots, V_n)$ and $(U_1, \ldots, U_n)$. Thus $V_i\sigma_u = U_i\sigma_u$ for all $i \in \{1, \ldots, n\}$. Assume that there exists σ'' such that $U'_i =_\Sigma V_i\sigma_u\sigma''$ for all $i \in \{1, \ldots, n\}$. Hence by hypothesis we have that $\sigma_u\sigma'' \nvDash \phi$ and so $\sigma'' \nvDash \phi\sigma_u$. Thus we conclude that $\sigma \vDash \forall \tilde{z}.\big[\bigvee_{i\in\mathcal{I}} x_i \neq_\Sigma V_i\sigma_u \vee \neg\phi\sigma_u\big]$ where $\tilde{z}$ are the variables of $V_1\sigma_u, \ldots, V_n\sigma_u$.

The proof of the left implication is similar. □

D.1.3 Simplifications of the formulas

These simplifications are adapted from those for testunif (from [Bla04]).

— Conjunction : *conjunction* transforms clauses of the form $H \wedge \text{formula}(\phi_1 \wedge \phi_2) \rightarrow C$ as follows :

$$H \wedge \text{formula}(\phi_1) \wedge \text{formula}(\phi_2) \rightarrow C$$

— Unification of disequations: *unifydiseq* transforms clauses of the form $H \wedge \text{formula}(\forall \tilde{z}.[\bigvee_i^n p_i \neq p'_i \vee \bigvee_j^m \exists \tilde{y}_j.q_j = q'_j]) \rightarrow C$ as follows. It tries to unify $(p_1, \ldots, p_n)$ and $(p'_1, \ldots, p'_n)$ modulo the equational theory. If this unification fails, then the clause becomes $H \rightarrow C$. Otherwise, *unifydiseq* replaces the clause with

$$H \wedge \bigwedge_{k=1}^{\ell} \text{formula}(\forall \tilde{z}.[\bigvee_{j=1}^{n_k} x_j^k \neq x_j^k \sigma_k \vee \bigvee_{j=1}^{m} \exists \tilde{y}_j.q_j \sigma_k = q'_j \sigma_k]) \rightarrow C$$

where $\sigma_k = \{x_1^k \mapsto x_1^k \sigma_k; \ldots; x_{n_k}^k \mapsto x_{n_k}^k \sigma_k\}$, for all $k = 1 \ldots \ell$ are the most general unifiers of $(p_1, \ldots, p_n)$ and $(p'_1, \ldots, p'_n)$ modulo the equational theory and $x_1^k, \ldots, x_{n_k}^k$ are variables of $(p_1, \ldots, p_n)$ and $(p'_1, \ldots, p'_n)$. In this unification, we assume that the variables of in the domain of σ_k do not occur in its image. This way, we don't have trivial equation $x = x$.

— Remove universal variable : *elimuniversal* transforms clauses of the form $H \wedge \text{formula}(\forall \tilde{z}, y.\phi) \rightarrow C$, where y is not a variable of ϕ, as $H \wedge \text{formula}(\forall \tilde{z}.\phi) \rightarrow C$. *elimuniversal* also transforms clauses of the form $H \wedge \text{formula}(\forall \tilde{z}, y.[y \neq t \vee \phi]) \rightarrow C$ as $H \wedge \text{formula}(\forall \tilde{z}.\phi\{^t/_y\}])$. A symetric rule exists for clauses of the form $H \wedge \text{formula}(\forall \tilde{z}, y.[t \neq y \vee \phi]) \rightarrow C$. This simplification is always applied after *unifydiseq*.

— Unification of equation : *unifyeq* transforms clauses of the form

$$H \wedge \text{formula}(\forall \tilde{z}.[\phi \vee \exists \tilde{y}.p = p']) \rightarrow C$$

as follows. It tries to unify p and p' modulo the equational theory. If this unification fails, then the clause becomes $H \wedge \text{formula}(\forall \tilde{z}.\phi) \rightarrow C$. Otherwise, *unifyeq* replaces the clause with

$$H \wedge \text{formula}(\forall \tilde{z}.[\phi \vee \bigvee_{i=1}^{n} \exists \tilde{y}.(x_1^i, \ldots, x_{k_i}^i) = (x_1^i \sigma_i, \ldots, x_{k_i}^i \sigma_i)]) \rightarrow C$$

where $\sigma_i = \{x_1^i \mapsto x_1^i \sigma_i; \ldots; x_{k_i}^i \mapsto x_{k_i}^i \sigma_i\}$, for all $i = 1 \ldots n$ are the most general unifiers of p_1 and p_2 modulo the equational theory and $x_1^i, \ldots, x_{k_i}^i$ are variables of p_1 and p'_2.

— Remove existential variables : *elimexistential* transforms clauses of the form $H \wedge \text{formula}(\forall \tilde{z}.(\phi \vee \exists \tilde{y}.(x_1, \ldots, x_n) = (p_1, \ldots, p_n))) \rightarrow C$ as follows: if I is the set of index i such that $x_i \in \tilde{y}$, then

$$H \wedge \text{formula}(\forall \tilde{z}.(\phi \vee \exists \tilde{y} \backslash \{x_i\}_{i \in I}.(x_{k_1}, \ldots, x_{k_n}) = (p_{k_1}, \ldots, p_{k_n}))) \rightarrow C$$

where $k_1, \ldots, k_n$ are index between 1 and n that are not in I. A symetric rule exists for formulas of the form $H \wedge \text{formula}(\forall \tilde{z}.(\phi \vee \exists \tilde{y}.(p_1, \ldots, p_n) = (x_1, \ldots, x_n))) \rightarrow C$

— Failed universal variables : *univfailed* transforms any clauses of the form $H \wedge \text{formula}(\forall \tilde{z}. \bigvee_{j=1}^m \exists \tilde{y}_j.(x_1^j, \ldots, x_{n_j}^j) = (q_1^j, \ldots, q_{n_j}^j)) \rightarrow C$ as follows: if I is the set of indexes j such that any variables of $(x_1^j, \ldots, x_{n_j}^j, q_1^j, \ldots, q_{n_j}^j)$ is not in $\tilde{z}$, then

$$H \wedge \text{formula}(\bigvee_{j \in I} \exists \tilde{y}_j.(x_1^j, \ldots, x_{n_j}^j) = (q_1^j, \ldots, q_{n_j}^j)) \rightarrow C$$

— Development : *develop* transforms clauses of the form $H \wedge \text{formula}(\phi_1 \vee \phi_2) \rightarrow C$ into two new clauses : $H \wedge \text{formula}(\phi_1) \rightarrow C$ and $H \wedge \text{formula}(\phi_2) \rightarrow C$.

— Application of substitution : *applysubs* transforms clauses of the form $H \wedge \text{formula}(\exists \tilde{y}.(x_1, \ldots, x_n) = (q_1, \ldots, q_n)) \rightarrow C$ into $H\sigma \rightarrow C\sigma$, where σ is the substitution $\{x_1 \mapsto p_1, \ldots x_n \mapsto p_n\}$.

— Detection of failed formula : *elimformula* removes clauses that contain the hypothesis formula($\bot$).

D.2 Proof of the Automatic Modification

This appendix provides the proofs of the results announced in Section 9.4.

D.2.1 Preliminary Lemmas

Lemma D.8. *Let P, Q be processes. Suppose that, for all substitutions σ closing for P and Q, $P\sigma \approx Q\sigma$. Then, for all contexts C closing for P and Q, $C[P] \approx C[Q]$.*

Proof. This lemma is fairly standard in process calculi. For example, [AG99, Appendix C.3] shows a similar result for the spi calculus. We give a proof for our calculus.

We rely on Definition 9.1, and use a relation $\mathcal{R}$ defined by $P_0 \mathcal{R} P_0'$ if and only if

$$P_0 \approx C'[P_1, \ldots, P_n] \text{ and } P_0' \approx C'[P_1', \ldots, P_n']$$

for some context C' such that no hole of C' is in evaluation position and for $i = 1, \ldots, n$, for all substitutions σ closing for P_i and P_i', $P_i\sigma \approx P_i'\sigma$.

We have that, for contexts C closing for P and Q, $C[P]\mathcal{R}C[Q]$. Indeed, if the hole of C is not in evaluation position, $C[P] \approx C[P]$, $C[Q] \approx C[Q]$, and for all substitutions σ closing for P and Q, $\sigma P \approx \sigma Q$. If the hole of C is in evaluation position, P and Q are closed, so $P \approx Q$ by hypothesis, hence $C[P] \approx C[Q]$, so letting $P_0 = C[P]$ and $P_0' = C' = C[Q]$ (C' is a context with no hole), we have $P_0 \approx C'$, $P_0' \approx C'$, hence $C[P]\mathcal{R}C[Q]$.

We show that $\mathcal{R}$ satisfies the three conditions of Definition 9.1. Moreover $\mathcal{R}$ is symmetric, so we can then conclude that $\mathcal{R} \subseteq \approx$, which implies the desired equivalence.

Condition 3 of Definition 9.1: Suppose $P_0 \mathcal{R} P_0'$, and let C be an evaluation context. $P_0 \mathcal{R} P_0'$ implies that there exists a context C', some processes $P_1, \ldots, P_n, P_1', \ldots, P_n'$ such that $P_0 \approx C'[P_1, \ldots, P_n]$ and $P_0' \approx C'[P_1', \ldots, P_n']$. Hence, we have $C[P_0] \approx C[C'[P_1, \ldots, P_n]]$ and $C[P_0'] \approx C[C'[P_1', \ldots, P_n']]$ by Condition 3 of Definition 9.1. Since no hole of C' is in evaluation position, then no hole of $C[C']$ is also in evaluation position. Hence, we deduce that $C[P_0]\mathcal{R}C[P_0']$.

Condition 2 of Definition 9.1: We first show that, if $C'[P_1, \ldots, P_n] \equiv Q_0$ where C' is any context such that no hole of C' is in evaluation position, then $C'[P_1', \ldots, P_n'] \equiv C''[P_1', \ldots, P_n']$ and $Q_0 = C''[P_1, \ldots, P_n]$ for some context C'' such that no hole of C'' is in evaluation position. The proof is done by induction on the derivation of $C'[P_1, \ldots, P_n] \equiv Q_0$.

Next, we show that, if $C'[P_1, \ldots, P_n] \to Q_0$ where C' is any context such that no hole of C' is in evaluation position, and for $i = 1, \ldots, n$, for all substitutions σ closing for P_i and P_i', $\sigma P_i \approx \sigma P_i'$, then $C'[P_1', \ldots, P_n'] \to Q_0' \approx C''[Q_1', \ldots, Q_{n'}']$ and $Q_0 = C''[Q_1, \ldots, Q_{n'}]$ for some context C'' and processes Q_0', Q_1', $\ldots$, $Q_{n'}'$ such that no hole of C'' is in evaluation position and for $i = 1, \ldots, n'$, for all substitutions σ closing for Q_i and Q_i', $\sigma Q_i \approx \sigma Q_i'$. The proof is done by induction on the derivation of $C'[P_1, \ldots, P_n] \to Q_0$.

— Case (Red I/O): We have $C'[P_1, \ldots, P_n] = \mathsf{out}(N, M).Q \mid \mathsf{in}(N', x).P \to Q_0 = Q \mid P\{^M/_x\}$ with $N =_\Sigma N'$. Since no hole of C' is in evaluation position, we have $Q = C_1[P_i(i \in S)]$, $P = C_2[P_i(i \notin S)]$ for some contexts C_1, C_2 and some subset S of $\{1, \ldots, n\}$, and $C'[P_1', \ldots, P_n'] = \mathsf{out}(N, M).C_1[P_i'(i \in S)] \mid \mathsf{in}(N', x).C_2[P_i(i \notin S)] \to C_1[P_i'(i \in S)] \mid C_2[P_i'(i \notin S)]\{^M/_x\}$. We let Q_i and Q_i' be the processes P_i and P_i' for $i \in S$ and $P_i\{^M/_x\}$ and $P_i'\{^M/_x\}$ for $i \notin S$, such that the corresponding hole of C_1 or C_2 is not in evaluation position. We have for all substitutions σ closing for Q_i and Q_i', $\sigma Q_i \approx \sigma Q_i'$, since that property is preserved by instantiation. We let $C'' = C_1[P_i(i \in S, \text{in evaluation position})][\,] \mid C_2\{^M/_x\}[P_i\{^M/_x\}(i \notin S, \text{in evaluation position})][\,]$ where only the holes not in evaluation position remain. We let $Q_0' = C_1[P_i'(i \in S)] \mid C_2[P_i'(i \notin S)]\{^M/_x\}$. We have $Q_0 = C''[Q_1, \ldots, Q_{n'}]$ and $C'[P_1', \ldots, P_n'] \to Q_0' \approx C''[Q_1', \ldots, Q_{n'}']$.

— Case (Red Fun 1): As in the case (Red I/O), the holes in the *in* branch are instantiated. The holes in the *else* branch disappear. The holes that become in evaluation position are handled as in (Red I/O).

— Case (Red Fun 2): The holes in the *in* branch disappear. The holes that become in evaluation position are handled as in (Red I/O).

— Case (Red Repl): The holes are duplicated.

— Cases (Red Par) and (Red Res): these cases follow immediately from the induction hypothesis.

— Case (Red ≡): we use the property shown above for ≡ and the induction hypothesis.

Suppose that $P_0 \to^* Q_0$ and $P_0 \mathcal{R} P'_0$. We have $P_0 \approx C'[P_1, \ldots, P_n]$ and $P'_0 \approx C'[P'_1, \ldots, P'_n]$ for some context C' and processes $P_1, \ldots, P_n$, $P'_1, \ldots, P'_n$ such that no hole of C' is in evaluation position and for $i = 1, \ldots, n$, for all substitutions σ closing for P_i and P'_i, $\sigma P_i \approx \sigma P'_i$. By Condition 2 of Definition 9.1, $C'[P_1, \ldots, P_n] \to^* Q''_0$ and $Q_0 \approx Q''_0$ for some Q''_0. By the property above, $C'[P'_1, \ldots, P'_n](\to\approx)^* C''[Q'_1, \ldots, Q'_{n'}]$ and $Q''_0 = C''[Q_1, \ldots, Q_{n'}]$ for some context C'' and processes $Q'_1, \ldots, Q'_{n'}$ such that no hole of C'' is in evaluation position and for $i = 1, \ldots, n'$, for all substitutions σ closing for Q_i and Q'_i, $\sigma Q_i \approx \sigma Q'_i$. By Condition 2 of Definition 9.1 again, $P'_0 \to^* Q'_0$ and $C''[Q'_1, \ldots, Q'_{n'}] \approx Q'_0$ for some Q'_0. Hence $Q_0 \approx C''[Q_1, \ldots, Q_{n'}]$ and $Q'_0 \approx C''[Q'_1, \ldots, Q'_{n'}]$, so $Q_0 \mathcal{R} Q'_0$.

Condition 1 of Definition 9.1: Let $P \Downarrow^0 M$ if and only if $P = C[\mathsf{out}(M', N).R]$ for some evaluation context C that does not bind $\mathit{fnames}(M)$ and $M =_\Sigma M'$.

We first notice that, if $C'[P_1, \ldots, P_n] \Downarrow^0 M$ where no hole of C' is in evaluation position, then $C'[P'_1, \ldots, P'_n] \downarrow^0_M$, since the difference between $C'[P_1, \ldots, P_n]$ and $C'[P'_1, \ldots, P'_n]$ is only is non-evaluation positions.

Suppose that $P_0 \Downarrow M$ and $P_0 \mathcal{R} P'_0$. We have $P_0 \approx C'[P_1, \ldots, P_n]$ and $P'_0 \approx C'[P'_1, \ldots, P'_n]$ for some context C' and processes $P_1, \ldots, P_n$, $P'_1, \ldots, P'_n$ such that no hole of C' is in evaluation position and for $i = 1, \ldots, n$, for all substitutions σ closing for P_i and P'_i, $\sigma P_i \approx \sigma P'_i$. By Condition 1 of Definition 9.1, $C'[P_1, \ldots, P_n] \Downarrow M$, that is, $C'[P_1, \ldots, P_n] \to^* \Downarrow^0 M$. By the properties proved regarding Condition 2, $C'[P'_1, \ldots, P'_n](\to\approx)^* C''[Q'_1, \ldots, Q'_{n'}]$ and $C''[Q_1, \ldots, Q_{n'}] \downarrow^0_M$ for some context C'' and processes $Q'_1, \ldots, Q'_{n'}$ such that no hole of C'' is in evaluation position. Hence $C''[Q'_1, \ldots, Q'_{n'}] \Downarrow^0 M$, so by Conditions 1 and 2 of Definition 9.1, $C'[P'_1, \ldots, P'_n] \Downarrow M$. Since $C'[P'_1, \ldots, P'_n] \approx P'_0$, we conclude that $P'_0 \Downarrow M$. □

Lemma D.9. *Let P, Q, R be processes. Let a be a name, x a variable, M a term, and D a term evaluation. We have that:*

1. *If $a \notin \mathit{fnames}(P)$, then for all contexts C closing for P, $C[P] \approx C[\nu a.P]$.*
2. *For all contexts C closing for $!\,P$, $C[!\,P] \approx C[!\,!\,P]$.*
3. *For all contexts C closing for $P\{^M/_x\}$, $C[P\{^M/_x\}] \approx C[\mathsf{let}\ x = M\ \mathsf{in}\ P\ \mathsf{else}\ Q]$.*
4. *For all contexts C closing for P, $C[P] \approx C[\mathsf{let}\ x = \mathsf{fail}\ \mathsf{in}\ Q\ \mathsf{else}\ P]$.*
5. *For all contexts C closing for $P_0 = \mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(P)\ \mathsf{else}\ \mathsf{snd}'(P)$, $C[P_0] \approx C[\mathsf{let}\ x = \mathsf{eval}\ \mathsf{glet}(D)\ \mathsf{in}\ P\{^{\mathsf{eval}\ \mathsf{gletin}(x, D_1, D_2)}/_{\mathsf{diff}'[D_1, D_2]}\}\ \mathsf{else}\ 0]$.*
6. *For all contexts C closing for $P \mid Q$, $C[P \mid Q] \approx C[Q \mid P]$.*
   For all contexts C closing for $(P \mid Q) \mid R$, $C[(P \mid Q) \mid R] \approx C[P \mid (Q \mid R)]$.
7. *If $x \notin \mathit{fvars}(P) \cup \mathit{fvars}(Q)$ then for all contexts C closing for $\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q$, $C[\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q] \approx C[\mathsf{let}\ x = \mathsf{notfail}(D)\ \mathsf{in}\ Q\ \mathsf{else}\ P]$.*

Proof. We first prove this result for processes that do not contain diff. By applying the obtained result twice, once for the component fst and once for the component snd, we obtain the same equivalence for processes that may contain diff. By Lemma D.8, it is enough to show:

1. If $a \notin \mathit{fnames}(P)$, then $P \approx \nu a.P$.
2. $!\,P \approx !\,!\,P$.
3. $P\{^M/_x\} \approx \mathsf{let}\ x = M\ \mathsf{in}\ P\ \mathsf{else}\ Q$.
4. $P \approx \mathsf{let}\ x = \mathsf{fail}\ \mathsf{in}\ Q\ \mathsf{else}\ P$.

5. $\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(P)\ \mathsf{else}\ \mathsf{snd}'(P) \approx$
   $\mathsf{let}\ x = \mathrm{eval}\ \mathsf{glet}(D)\ \mathsf{in}\ P\{^{\mathrm{eval}\ \mathsf{gletin}(x,D_1,D_2)}/_{\mathsf{diff}'[D_1,D_2]}\}\ \mathsf{else}\ 0.$
6. $P \mid Q \approx Q \mid P$ and $(P \mid Q) \mid R \approx P \mid (Q \mid R)$.
7. $\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q \approx \mathsf{let}\ x = \mathsf{equals}(D, \mathsf{fail})\ \mathsf{in}\ Q\ \mathsf{else}\ P$

where P, Q, R are closed processes, M and D are ground terms and term evaluation, except for P in Item 3 in which $\mathit{fvars}(P) \subseteq \{x\}$, for Q in Item 4 in which $\mathit{fvars}(Q) \subseteq \{x\}$, and for P in Item 5 in which $\mathit{fvars}(\mathsf{fst}'(P)) \subseteq \{x\}$ and $\mathit{fvars}(\mathsf{snd}'(P)) = \emptyset$. Item 6 is obvious since $\equiv \subseteq \approx$. Note that in Item 7, since P and Q are closed, $x \notin \mathit{fvars}(P) \cup \mathit{fvars}(Q)$. Let us prove the other cases by relying on Definition 9.1.

For Items 3 and 4, we use a relation $\mathcal{R}$ defined by $P_0 \mathcal{R} P_0'$ if and only if

$$P_0 = C[P_1] \text{ and } P_0' = C[P_1']$$

or

$$P_0 = C[P_1'] \text{ and } P_0' = C[P_1]$$

or

$$P_0 = P_0'$$

for some evaluation context C and processes P_1, P_1' such that $P_1 \mathcal{R}_1 P_1'$ where

— in Item 3, $\mathcal{R}_1$ is defined by $P\{^M/_x\} \mathcal{R}_1 \mathsf{let}\ x = M\ \mathsf{in}\ P\ \mathsf{else}\ Q$ for all P, Q, x, M;

— in Item 4, $\mathcal{R}_1$ is defined by $P \mathcal{R}_1 \mathsf{let}\ x = \mathsf{fail}\ \mathsf{in}\ Q\ \mathsf{else}\ P$ for all P, Q, x.

We show that $\mathcal{R}$ satisfies the three conditions of Definition 9.1. Moreover $\mathcal{R}$ is symmetric, so we can then conclude that $\mathcal{R} \subseteq \approx$, which implies the desired equivalences.

— $\mathcal{R}$ obviously satisfies Condition 3 of Definition 9.1.

— To show Condition 2 of Definition 9.1, we show that, if $P_0 \to Q_0$ and $P_0 \mathcal{R} P_0'$, then there exists Q_0' such that $Q_0 \mathcal{R} Q_0'$ and $P_0' \to^* Q_0'$.
  If $P_0 = C[P_1]$, $P_0' = C[P_1']$, and $P_1 \mathcal{R}_1 P_1'$, then we first reduce P_1' into P_1, transforming $\mathsf{let}\ x = M\ \mathsf{in}\ P\ \mathsf{else}\ Q$ into $P\{M/x\}$ (Item 3) and $\mathsf{let}\ x = \mathsf{fail}\ \mathsf{in}\ Q\ \mathsf{else}\ P$ into P (Item 4). In this case, we have $P_0' = C[P_1'] \to C[P_1] = P_0 \to Q_0$. Taking $Q_0' = Q_0$, we have $Q_0 \mathcal{R} Q_0'$ and $P_0' \to^* Q_0'$.
  If $P_0 = C[P_1']$, $P_0' = C[P_1]$, and $P_1 \mathcal{R}_1 P_1'$, we show the following.
  1. If $C[P_1'] \equiv P'$, then $P' = C'[P_1']$ and $C[P_1] \equiv C'[P_1]$ for some evaluation context C', by induction on the derivation of $C[P_1'] \equiv P'$.
  2. If $C[P_1'] \to P'$, then either $P' = C'[P_1]$ and $C[P_1] \equiv C'[P_1]$, or $P' = C'[P_1']$ and $C[P_1] \to C'[P_1]$, for some evaluation context C'. The proof proceeds by induction of the derivation of $C[P_1'] \to P'$:
     Case (Red I/O) is impossible.
     Case (Red Fun 1) can be applied only in Item 3. In this case, $C[P_1'] = P_1' \to P_1$ and we take $C' = [\,]$.
     Case (Red Fun 2) can be applied only in Item 4. In this case, $C[P_1'] = P_1' \to P_1$ and we take $C' = [\,]$.
     In case (Red Par), we have $C[P_1'] = P \mid R \to Q \mid R = P'$ with $P \to Q$. First case: $R = C''[P_1']$, $C = P \mid C''$. Then $P' = Q \mid C''[P_1']$. Let $C' = Q \mid C''$. Then $P' = C'[P_1']$, and $C[P_1] = P \mid C''[P_1] \to Q \mid C''[P_1] = C'[P_1]$. Second case: $P = C''[P_1']$, $C = C'' \mid R$. Then $C''[P_1'] \to Q$. By induction hypothesis, either $Q = C'''[P_1]$ and $C''[P_1] \equiv C'''[P_1]$, or $Q = C'''[P_1']$ and $C''[P_1] \to C'''[P_1]$, for some evaluation context C'. Let $C' = C''' \mid R$. Either $P' = Q \mid R = C'''[P_1] \mid R = C'[P_1]$ and $C[P_1] = C''[P_1] \mid R \equiv C'''[P_1] \mid R = C'[P_1]$, or $P' = Q \mid R = C'''[P_1'] \mid R = C'[P_1']$ and $C[P_1] = C''[P_1] \mid R \to C'''[P_1] \mid R = C'[P_1]$.
     Case (Red Res) follows by induction hypothesis, similarly to the second case of (Red Par).
     Case (Red $\equiv$) follows using the property above for $\equiv$ and the induction hypothesis.

We have $C[P_1'] = P_0 \to Q_0$, so either $Q_0 = C'[P_1]$ and $C[P_1] \equiv C'[P_1]$, or $Q_0 = C'[P_1']$ and $C[P_1] \to C'[P_1]$, for some evaluation context C'. We let $Q_0' = C'[P_1]$. In the first case, we have $Q_0 = Q_0'$ so $Q_0 \mathcal{R} Q_0'$ and $P_0' = C[P_1] \to C'[P_1] = Q_0'$. In the second case, we have $Q_0 = C'[P_1']$ and $Q_0' = C'[P_1]$ so $Q_0 \mathcal{R} Q_0'$ and $P_0' = C[P_1] \to C'[P_1] = Q_0'$.
If $P_0 = P_0'$, let $Q_0' = Q_0$. We have $Q_0 \mathcal{R} Q_0'$ and $P_0' = P_0 \to Q_0 = Q_0'$.

— To show Condition 1, using Condition 2, it is enough to show that, if $P_0 \mathcal{R} P_0'$ and $P_0 \equiv C'[\mathsf{out}(M', N).R]$ for some evaluation context C' that does not bind $\mathit{fnames}(M)$ and $M =_\Sigma M'$, then $P_0' \Downarrow M$.
If $P_0 = C[P_1]$, $P_0' = C[P_1']$, and $P_1 \mathcal{R}_1 P_1'$, we have $P_0' = C[P_1'] \to C[P_1] = P_0$ hence $P_0' \Downarrow M$.
If $P_0 = C[P_1']$, $P_0' = C[P_1]$, and $P_1 \mathcal{R}_1 P_1'$, we have $P_0 \equiv C'[\mathsf{out}(M', N).R]$. By Property 1 shown above for $\equiv$, $C'[\mathsf{out}(M', N).R] = C''[P_1']$ and $C[P_1] \equiv C''[P_1]$ for some C''. Hence there is an evaluation context C''' that does not bind $\mathit{fnames}(M)$ such that $C'[\mathsf{out}(M', N).R] = C''[P_1'] = C'''[P_1', \mathsf{out}(M', N).R]$, thus $P_0' \equiv C'''[P_1, \mathsf{out}(M', N).R]$ and so $P_0' \Downarrow M$.
If $P_0 = P_0'$, the result is obvious.

For Item 5, we define the relation $\mathcal{R}$ by $P_0 \mathcal{R} P_0'$ if and only if

$P_0 = C[\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(P)\ \mathsf{else}\ \mathsf{snd}'(P)]$ and
$P_0' = C[\mathsf{let}\ x = \mathrm{eval}\ \mathsf{glet}(D)\ \mathsf{in}\ P\{^{\mathrm{eval}\ \mathsf{gletin}(x, D_1, D_2)}/_{\mathsf{diff}'[D_1, D_2]}\}\ \mathsf{else}\ 0]$
for some evaluation context C, variable x, term evaluation D, and process P

or

$$P_0 = \mathsf{fst}'(P)\{^M/_x\} \text{ and } P_0' = P\{^{\mathrm{eval}\ \mathsf{gletin}(M, D_1, D_2)}/_{\mathsf{diff}'[D_1, D_2]}\} \text{ for some } P \text{ and } M$$

or

$$P_0 = \mathsf{snd}'(P) \text{ and } P_0' = P\{^{\mathrm{eval}\ \mathsf{gletin}(\mathsf{c_o}, D_1, D_2)}/_{\mathsf{diff}'[D_1, D_2]}\} \text{ for some } P$$

or the symmetric obtained by swapping P_0 and P_0'. We show $\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(P)\ \mathsf{else}\ \mathsf{snd}'(P) \approx \mathsf{let}\ x = \mathrm{eval}\ \mathsf{glet}(D)\ \mathsf{in}\ P\{^{\mathrm{eval}\ \mathsf{gletin}(x, D_1, D_2)}/_{\mathsf{diff}'[D_1, D_2]}\}\ \mathsf{else}\ 0$, by proving that $\mathcal{R}$ satisfies the three conditions of Definition 9.1, similarly to the proof we have done for Items 3 and 4.

For Item 1, we define the relation $\mathcal{R}$ by $P_0 \mathcal{R} P_0'$ if and only if $\nu a.P_0 \equiv P_0'$ or $\nu a.P_0' \equiv P_0$ for some $a \notin \mathit{fnames}(P_0)$ or $a \notin \mathit{fnames}(P_0')$ respectively. We show $P \approx \nu a.P$, by proving that $\mathcal{R}$ satisfies the three conditions of Definition 9.1.

For Item 2, we define the relation $\mathcal{R}$ by $P_0 \mathcal{R} P_0'$ if and only if

$$P_0 \equiv C[!\,P] \text{ and } P_0' \equiv C[!\,!\,P \mid !\,P \mid \ldots \mid !\,P]$$

or

$$P_0 \equiv C[!\,!\,P \mid !\,P \mid \ldots \mid !\,P] \text{ and } P_0' \equiv C[!\,P]$$

for some evaluation context C and process P. We show $!\,P \approx !\,!\,P$, by proving that $\mathcal{R}$ satisfies the three conditions of Definition 9.1.

The proof follows a strategy similar to the proof of Items 3 and 4. For Items 1 and 2, some details are however more complex, because the structural equivalence may modify $\nu a.P$ and $!\,!\,P \mid !\,P \mid \ldots \mid !\,P$. For instance, to show Condition 2 of Definition 9.1 for Item 1, we need to show that, if $P_0 \to Q_0$ and $P_0 \equiv \nu a.P_0'$, then there exists Q_0' such that $Q_0 \equiv \nu a.Q_0'$ and $P_0' \to Q_0'$. Such a result is fairly standard in process calculi and can be proved by showing a series of lemmas decomposing reduction of $\nu a.P$ and $P \mid Q$, using a labeled semantics.

For Item 7, we define the relation $\mathcal{R}$ by $P_0 \mathcal{R} P_0'$ if and only if

$P_0 = C[\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q]$ and
$P_0' = C[\mathsf{let}\ x = \mathsf{notfail}(D)\ \mathsf{in}\ Q\ \mathsf{else}\ P]$
for some evaluation context C, variable x, term evaluation D, and processes P, Q
such that $x \notin \mathit{fvars}(P) \cup \mathit{fvars}(Q)$

or

$$P_0 = P_0'$$

or the symmetric obtained by swapping P_0 and P_0'. We show $\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ Q \approx \mathsf{let}\ x = \mathsf{notfail}(D)\ \mathsf{in}\ Q\ \mathsf{else}\ P$, by proving that $\mathcal{R}$ satisfies the three conditions of Definition 9.1, similarly to the proof we have done for Items 3 and 4. It relies on the fact that if $D \downarrow_\Sigma M$ then $\mathsf{notfail}(D) \downarrow_\Sigma \mathsf{fail}$, and if $D \downarrow_\Sigma \mathsf{fail}$ then $\mathsf{notfail}(D) \downarrow_\Sigma \mathsf{c_o}$ which is a message.

□

D.2.2 Proofs for the *merge* function

Lemma 9.8. *Let P and P' be two biprocesses. If $merge(P, P') = Q$, then:*

- *for all contexts C closing for P, $C[P] \approx C[\mathsf{fst}'(Q)]$;*
- *for all contexts C closing for P', $C[P'] \approx C[\mathsf{snd}'(Q)]$.*

Proof. Let P_0, P_0' be two biprocesses. We prove the result by induction on P_0 and P_0'. We do a case analysis on the rules of Figure 9.6:

Case (Mnil)*:* Trivial.

Case (Mout)*:* We have $P_0 = \mathsf{out}(M, N).P$ and $P_0' = \mathsf{out}(M', N').P'$. Let C be a context closing for P_0. We show that $C[P_0] \approx C[\mathsf{fst}'(merge(P_0, P_0')]$, that is, $C[\mathsf{out}(M, N).P] \approx C[\mathsf{let}\ x = M\ \mathsf{in}\ \mathsf{let}\ x' = N\ \mathsf{in}\ \mathsf{out}(x, x').\mathsf{fst}'(merge(P, P'))]$. Let $C_1 = C[\mathsf{let}\ x = M\ \mathsf{in}\ \mathsf{let}\ x' = N\ \mathsf{in}\ \mathsf{out}(x, x').[\ ]]$. By induction hypothesis on $merge(P, P')$, $C_1[P] \approx C_1[\mathsf{fst}'(merge(P, P'))]$, so we just have to show that $C[\mathsf{out}(M, N).P] \approx C[\mathsf{let}\ x = M\ \mathsf{in}\ \mathsf{let}\ x' = N\ \mathsf{in}\ \mathsf{out}(x, x').P]$. This follows by two applications of Lemma D.9, Item 3.

By symmetry, for all contexts C closing for P_0', $C[P_0'] \approx C[\mathsf{snd}'(merge(P_0, P_0'))]$.

Case (Min)*:* This case is similar to the case (Mout).

Case (Mpar)*:* We have $P_0 = P_1 \mid \ldots \mid P_n$ and $P_0' = P_1' \mid \ldots \mid P_n'$. Furthermore, there exists a permutation $(i_k)_{k=1..n}$ of $(1, \ldots, n)$ such that $merge(P_0, P_0') = Q_1 \mid \ldots \mid Q_n$ with $Q_k = merge(P_k, P_{i_k}')$, for $k = 1..n$. Let C a context closing for P_0. Let $C_k = C[\mathsf{fst}'(Q_1) \mid \ldots \mid \mathsf{fst}'(Q_{k-1}) \mid [] \mid P_{k+1} \mid \ldots \mid P_n]$. Since C is closing for P_0, we have that C_k is closing for P_k. By induction hypothesis on Q_k, we deduce that $C_k[P_k] \approx C_k[\mathsf{fst}'(Q_k)]$. Moreover, $C_k[\mathsf{fst}'(Q_k)] = C_{k+1}[P_{k+1}]$. Hence, with a simple induction on n, we deduce that $C[P_0] \approx C[\mathsf{fst}'(merge(P_0, P_0'))]$.

Using a similar proof and Lemma D.9, Item 6 to permute the elements of the parallel composition, we obtain that $C[P_0'] \approx C[\mathsf{snd}'(merge(P_0, P_0'))]$, for all contexts C closing for P_0'.

Case (Mres)*:* We have $P_0 = \nu a.P$ and $a \notin fnames(P_0')$. Let C a context closing for $\nu a.P$. By induction hypothesis on $merge(P, P_0')$, $C[\nu a.P] \approx C_1[\nu a.\mathsf{fst}'(merge(P, P_0'))]$. Hence, $C[P_0] \approx C[\mathsf{fst}'(merge(P_0, P_0'))]$.

Let C be a context closing for P_0'. Since $a \notin fnames(P_0')$, by Lemma D.9, Item 1, $C[P_0'] \approx C[\nu a.P_0']$. By induction hypothesis on $merge(P, P_0')$, we have $C[\nu a.P_0'] \approx C[\nu a.\mathsf{snd}'(merge(P, P_0'))]$. Hence we obtain the desired result: $C[P_0'] \approx C[\mathsf{snd}'(merge(P_0, P_0'))]$.

Case (Mrepl1)*:* We have $P_0 =! \nu a_1 \ldots \nu a_n .!\, P$ and $P_0' =! P'$. Thanks to our induction hypothesis on $merge(!\, P, P_0')$, we have that, for all contexts C closing for P_0, $C[P_0] \approx C[\mathsf{fst}'(merge(P_0, P_0'))]$.

Let C be a context closing for P_0'. We have $C[P_0'] = C[!\, P'] \approx C[!\,!\, P']$ by Lemma D.9, Item 2, so $C[P_0'] \approx C[!\, \nu a_1 \ldots \nu a_n .!\, P']$ by n applications of Lemma D.9, Item 1. Let $C_1 = C[!\, \nu a_1 \ldots \nu a_n [\ ]]$. By induction hypothesis on $merge(!\, P, !\, P')$, $C_1[!\, P'] \approx C_1[\mathsf{snd}'(merge(!\, P, !\, P'))]$, so $C[P_0'] \approx C[!\, \nu a_1 \ldots \nu a_n .\mathsf{snd}'(merge(!\, P, !\, P'))] = C[\mathsf{snd}'(merge(P_0, P_0'))]$.

Case (Mrepl2)*:* This case follows immediately by induction hypothesis.

Case (Mlet1)*:* In this case, we have $P_0 = \mathsf{let}\ x = D\ \mathsf{in}\ P_1\ \mathsf{else}\ P_2$ and $P_0' = \mathsf{let}\ x' = D'\ \mathsf{in}\ P_1'\ \mathsf{else}\ P_2'$. Let C be a context closing for P_0. Since y is a fresh variable, we have that $C[P_0] \approx C[\mathsf{let}\ y = D\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ P_2]$. By induction hypothesis on both $Q_1 = merge(P_1\{^y/_x\}, P_1'\{^y/_{x'}\})$ and $Q_2 =$

$merge(P_2, P_2')$, we obtain that $C[\mathsf{let}\ y = D\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ P_2] \approx C[\mathsf{let}\ y = D\ \mathsf{in}\ \mathsf{fst}'(Q_1)\ \mathsf{else}\ P_2]$ $\approx C[\mathsf{let}\ y = D\ \mathsf{in}\ \mathsf{fst}'(Q_1)\ \mathsf{else}\ \mathsf{fst}'(Q_2)]$. Since $\mathsf{fst}'(\mathsf{diff}'[D, D']) = D$, we obtain that $C[P_0] \approx C[\mathsf{fst}'(merge(P_0, P_0'))]$.

By symmetry, we obtain that, for all contexts C closing for P_0', $C[P_0'] \approx C[\mathsf{snd}'(merge(P_0, P_0'))]$.

Case (Mlet2)*:* We have $P_0 = \mathsf{let}\ x = D\ \mathsf{in}\ P_1\ \mathsf{else}\ P_2$ and $P_0' = \mathsf{let}\ x' = D'\ \mathsf{in}\ P_1'\ \mathsf{else}\ P_2'$. Let C be a context closing for P_0. Since y is a fresh variable, we have that $C[P_0] \approx C[\mathsf{let}\ y = D\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ P_2]$. By induction hypothesis on both $Q_1 = merge(P_1\{^y/_x\}, P_2')$ and $Q_2 = merge(P_2, P_1')$, we obtain that $C[\mathsf{let}\ y = D\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ P_2] \approx C[\mathsf{let}\ y = D\ \mathsf{in}\ \mathsf{fst}'(Q_1)\ \mathsf{else}\ P_2]$ $\approx C[\mathsf{let}\ y = D\ \mathsf{in}\ \mathsf{fst}'(Q_1)\ \mathsf{else}\ \mathsf{fst}'(Q_2)]$. Since $\mathsf{fst}'(\mathsf{diff}'[D, \mathsf{notfail}(D')]) = D$, we obtain that $C[P_0] \approx C[\mathsf{fst}'(merge(P_0, P_0'))]$.

Consider C a context closing for P_0'. Thanks to Lemma D.9, Item 7, we have that $C[P_0'] \approx C[\mathsf{let}\ x' = \mathsf{notfail}(D')\ \mathsf{in}\ P_2'\ \mathsf{else}\ P_1']$. By inductive hypothesis on both $Q_1 = merge(P_1\{^y/_x\}, P_2')$ and $Q_2 = merge(P_2, P_1')$ we have $C[\mathsf{let}\ x' = \mathsf{notfail}(D')\ \mathsf{in}\ P_2'\ \mathsf{else}\ P_1'] \approx C[\mathsf{let}\ x' = \mathsf{notfail}(D')\ \mathsf{in}\ \mathsf{snd}'(Q_1)$ $\mathsf{else}\ P_1'] \approx C[\mathsf{let}\ x' = \mathsf{notfail}(D')\ \mathsf{in}\ \mathsf{snd}'(Q_1)\ \mathsf{else}\ \mathsf{snd}'(Q_2)$. Since $\mathsf{snd}'(\mathsf{diff}'[D, \mathsf{notfail}(D')]) = \mathsf{notfail}(D')$, we obtain that $C[P_0'] \approx C[\mathsf{snd}'(merge(P_0, P_0'))]$.

Case (Mlet3)*:* We have $P_0 = \mathsf{let}\ x = D\ \mathsf{in}\ P_1\ \mathsf{else}\ P_2$. Let C be a closing context for P_0. Since $\approx$ is closed under renaming and y is a fresh variable, we have that $C[P_0] \approx C[\mathsf{let}\ y = D\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ P_2]$. By induction hypothesis on $Q = merge(P_1\{^y/_x\}, P_0')$ with the context $C[\mathsf{let}\ y = D\ \mathsf{in}\ [\,]\ \mathsf{else}\ P_2]$, we obtain that $C[P_0] \approx C[\mathsf{let}\ y = D\ \mathsf{in}\ \mathsf{fst}'(Q)\ \mathsf{else}\ P_2]$. Since $C[\mathsf{let}\ y = D\ \mathsf{in}\ \mathsf{fst}'(Q)\ \mathsf{else}\ P_2] = C[\mathsf{fst}'(\mathsf{let}\ y = \mathsf{diff}'[D, \mathsf{c_o}]\ \mathsf{in}\ Q\ \mathsf{else}\ P_2)]$, we obtain $C[P_0] \approx C[\mathsf{fst}'(merge(P_0, P_0'))]$.

Let C be a context closing for P_0'. Since y is a fresh variable and so not a variable of P_0', by Lemma D.9, Item 3, $C[P_0'] \approx C[\mathsf{let}\ y = \mathsf{c_o}\ \mathsf{in}\ P_0'\ \mathsf{else}\ P_2]$. By induction hypothesis on $Q = merge(P\{^y/_x\}, P_0')$ with the context $C[\mathsf{let}\ y = \mathsf{c_o}\ \mathsf{in}\ [\,]\ \mathsf{else}\ P_2]$, we obtain that $C[\mathsf{let}\ y = \mathsf{c_o}\ \mathsf{in}\ P_0'\ \mathsf{else}\ P_2] \approx C[\mathsf{let}\ y = \mathsf{c_o}\ \mathsf{in}\ \mathsf{snd}'(Q)\ \mathsf{else}\ P_2]$. Since $C[\mathsf{let}\ y = \mathsf{c_o}\ \mathsf{in}\ \mathsf{snd}'(Q)\ \mathsf{else}\ P_2] = C[\mathsf{snd}'(\mathsf{let}\ y = \mathsf{diff}'[D, \mathsf{c_o}]\ \mathsf{in}\ Q\ \mathsf{else}\ P_2)]$, we conclude that $C[P_0'] \approx C[\mathsf{snd}'(merge(P_0, P_0'))]$.

Case (Mlet4)*:* We have $P_0 = \mathsf{let}\ x = D\ \mathsf{in}\ P_1\ \mathsf{else}\ P_2$. Let C be a closing context for P_0. Since $\approx$ is closed under renaming and y is a fresh variable, we have that $C[P_0] \approx C[\mathsf{let}\ y = D\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ P_2]$. By induction hypothesis on $Q = merge(P_2, P_0')$ with the context $C[\mathsf{let}\ y = D\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ [\,]]$, we obtain that $C[P_0] \approx C[\mathsf{let}\ y = D\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ \mathsf{fst}'(Q)] = C[\mathsf{fst}'(\mathsf{let}\ y = \mathsf{diff}'[D, \mathsf{fail}]\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ Q)]$, we obtain the desired result: $C[P_0] \approx C[\mathsf{fst}'(merge(P_0, P_0'))]$.

Let C be a context closing for P_0'. Thanks to Lemma D.9, Item 4, we deduce that $C[P_0'] \approx C[\mathsf{let}\ y = \mathsf{fail}\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ P_0']$. By induction hypothesis on $Q = merge(P_2, P_0')$ with the context $C[\mathsf{let}\ y = \mathsf{fail}\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ [\,]]$, we obtain $C[\mathsf{let}\ y = \mathsf{fail}\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ P_0'] \approx C[\mathsf{let}\ y = \mathsf{fail}\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ \mathsf{snd}'(Q)] = C[\mathsf{snd}'(\mathsf{let}\ y = \mathsf{diff}'[D, \mathsf{fail}]\ \mathsf{in}\ P_1\{^y/_x\}\ \mathsf{else}\ Q)]$. So we conclude that $C[P_0'] \approx C[\mathsf{snd}'(merge(P_0, P_0'))]$. □

D.2.3 Proofs for the *simpl* Function

Lemma 9.9. *Let P be a biprocess. For all contexts C closing for P, $C[P] \approx C[simpl(P)]$.*

Proof. Let P_0 be a biprocess. We prove by induction on P_0 that, for all contexts C closing for P_0, $C[P_0] \approx C[simpl[P_0]$. We do a case analysis on the rules of Figure 9.7.

Case (Snil)*:* Trivial.

Case (Sout)*:* We have $P_0 = \mathsf{out}(M, N).P$ and $simpl(P_0) = \mathsf{out}(M, N).simpl(P)$. Let C be a context closing for P_0. Let $C_1 = C[\mathsf{out}(M, N).[\,]]$. We have that $C[P_0] = C_1[P]$ with C_1 closing for P. By induction hypothesis on $simpl(P)$, we can deduce that $C_1[P] \approx C_1[simpl(P))] = C[\mathsf{out}(M, N).simpl(P)] = C[simpl(P_0)]$.

Cases (Sin)*,* (Sres) *and* (Srepl)*:* Proof similar to case (Sout).

Case (Smid)*:* We have $P_0 = P \mid Q$. Let C be a context closing for P_0. Let $C_1 = [\,] \mid Q$ and $C_2 = simpl(P) \mid [\,]$. By induction hypothesis on $simpl(P)$, we obtain that $C_1[P] \approx C_1[simpl(P)]$.

Moreover, $C_1[simpl(P)] = C_2[Q]$. By induction hypothesis on $simpl(Q)$, we have that $C_2[Q] \approx C_2[simpl(Q)]$. With $C_2[simpl(Q)] = C[simpl(P_0)]$, we conclude that $C[P_0] \approx C[simpl(P_0)]$.

Case (Slet)*:* Proof similar to case (Smid).

Case (Smerge)*:* We have $P_0 = \mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ P'$. Let $Q' = merge(simpl(P), simpl(P'))$ and $Q = Q'\{^{\mathrm{eval}\ \mathsf{gletin}(x,D_1,D_2)}/_{\mathsf{diff}'[D_1,D_2]}\}$. Thanks to Lemma D.9, Item 5, we deduce that $C[\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(Q')\ \mathsf{else}\ \mathsf{snd}'(Q')] \approx C[\mathsf{let}\ x = \mathrm{eval}\ \mathsf{glet}(D)\ \mathsf{in}\ Q\ \mathsf{else}\ 0]$.

Furthermore, by induction hypothesis on P and Lemma 9.8 with the context $C_1 = C[\mathsf{let}\ x = D\ \mathsf{in}\ [\,]\ \mathsf{else}\ P']$, we obtain $C_1[\mathsf{fst}'(Q')] \approx C_1[simpl(P)] \approx C_1[P]$. Hence we deduce that $C[\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(Q')\ \mathsf{else}\ P'] \approx C[\mathsf{let}\ x = D\ \mathsf{in}\ P\ \mathsf{else}\ P'] = C[P_0]$. By induction hypothesis on P' and Lemma 9.8 with $C_2 = C[\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(Q')\ \mathsf{else}\ [\,]]$, we obtain similarly $C_2[\mathsf{snd}'(Q')] \approx C_2[simpl(P')] \approx C_2[P']$, that is, $C[\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(Q')\ \mathsf{else}\ \mathsf{snd}'(Q')] \approx C[\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(Q')\ \mathsf{else}\ P']$, so we deduce that $C[P_0] \approx C[\mathsf{let}\ x = D\ \mathsf{in}\ \mathsf{fst}'(Q')\ \mathsf{else}\ \mathsf{snd}'(Q')]]$.

By combining the two equivalences, we conclude that $C[P_0] \approx C[simpl(P_0)]$ □