Proof Search in Lax Logic

Jacob M. Howe

Computing Laboratory
University of Kent, Canterbury, CT2 8NF, UK
J.M.Howe@ukc.ac.uk

Abstract. A Gentzen sequent calculus for Lax Logic is presented, the proofs in which naturally correspond in a 1–1 way with the normal natural deductions for the logic. The propositional fragment of this calculus is then used a basis for another calculus, one which uses a history mechanism in order to give a decision procedure for propositional Lax Logic.

1 Introduction and Background

Proof search can be used with either of two meanings. It can either be used to mean the search for all proofs of a formula (proof enumeration), or to mean the search for a yes/no answer to a query (theorem proving). This paper describes two new sequent calculi for Lax Logic. One calculus is for proof enumeration for quantified Lax Logic, the other calculus is for theorem proving in propositional Lax Logic.

Lax Logic is an intuitionistic modal logic with a single modality ($\circ$, *somehow*). This modality is unusual in that it has properties both of necessity and of possibility. The modality can be thought of as expressing correctness up to a constraint, abstracting away from the detail (hence the choice of name, Lax Logic). A formula $\circ P$ can be read as "for some constraint c, formula P holds under c". The modality is axiomatised by three axioms:

$$\circ R : \ S \supset \circ S$$
$$\circ M : \circ \circ S \supset \circ S$$
$$\circ F : \ (S \supset T) \supset (\circ S \supset \circ T)$$

Lax logic has recently been investigated by Fairtlough, Mendler & Walton ([Men93], [FM94], [FM97], [FMW97], [FW97]) and by Benton, Bierman & de Paiva ([BBdP98]).

Curry ([Cur52]) introduced the logic to illustrate cut-elimination in the presence of modalities. The logic was rediscovered by Mendler, who developed the logic for abstract reasoning about constraints in hardware verification ([Men93]). The timing constraints that need to be satisfied in a circuit can be abstracted away as instances of the modality and reasoned about separately from the logical analysis of the circuit. In [Men93], [FM94], [FM97], the proof theory and semantics of Lax Logic are developed, giving Gentzen calculi, natural deduction calculi and Kripke semantics for the logic as well as giving details of the logic's use as a tool for hardware verification.

Lax Logic has also been observed ([BBdP98]) as the type system for Moggi's computational lambda-calculus (see [Mog89]). In [BBdP98] the correspondence between the natural deduction presentation of Lax Logic (there called computational logic) and the computational lambda calculus is given, along with some proof theoretic results on the logic.

In [FMW97] the ability of Lax Logic to give an abstract expression of constraints is utilised to give a proof theoretic semantics for constraint logic programming languages. The constraints to be satisfied can be abstracted away as modalities and the query can be reasoned about logically. The constraints can then be analysed separately. The logic is used to give proofs of queries. These proofs give the constraints to be satisfied. This approach is an extension to constraint logic programming of the view [MNPS91] of logic programming as backwards proof search in constructive logic. In essence, this approach takes normal natural deduction as the proof theoretic semantic for logic programming. Normal natural deductions are then the proof theoretic semantics for constraint logic programming.

Natural deduction has a pragmatic drawback. In searching backwards for the proof of a formula, it is not always obvious which rule to apply. For instance in

$$\frac{\Gamma \vdash P \supset Q \quad \Gamma \vdash P}{\Gamma \vdash Q} \; (\supset_\varepsilon)$$

it is not obvious from the conclusion that we should apply $(\supset_\varepsilon)$. Even when this rule has been decided upon, what P should be is hard to decide. Cut-free Gentzen sequent calculus systems ([Gen69]) are much better from this point of view. When a principal formula has been chosen, the rules which can be applied to it are restricted. When the rule has also been chosen, the rule application is deterministic. The application of logical rules is directed by the syntax of the principal formula. Structural rules can usually be built into the sequent system. In such a system, when a principal formula has been chosen, the next rule application is determined exactly by the syntax of that formula. All logical rules of the sequent calculus are introduction rules (on the left or on the right).

There are well known translations ([Pra65]) between normal natural deductions and sequent proofs. Therefore we can search for proofs in sequent calculus systems and then translate the resulting proofs to normal natural deductions. The drawback here is that many sequent proofs translate to the same normal natural deduction. Hence when one is trying to enumerate all proofs of a formula, the same proof is found again and again.

This gives one motivation for 'permutation-free' sequent calculi (introduced in [Her95], [Her96] for Intuitionistic Logic). These are sequent calculi system for logics (enabling syntax directed proof search) whose proofs can be translated in a 1–1 way with the normal natural deductions for the logic. Permutation-free calculi have the advantages of a sequent calculus system, whilst reflecting the structure of normal natural deductions.

The first calculus described in this paper, PFLAX, is a proof enumeration calculus for first-order quantified Lax Logic. PFLAX is a permutation-free calculus for Lax Logic –

its proofs naturally correspond in a 1–1 way with the normal natural deductions for the logic. This calculus is a suitable calculus for proof search in relation to constraint logic programming.

For many applications of logics a simple provable/unprovable answer will do. In this case we are interested in the quickest way of getting this answer (and in its correctness). Propositional logics are usually decidable and therefore we are interested in finding decision procedures, in particular we would like quick decision procedures.

The contraction rule is a major obstacle to finding decision procedures for non-classical logics. Duplication of a formula means that on backwards proof search the sequents are becoming more complicated, not less. We have no obvious way of seeing that we should terminate the search. Leaving contraction out usually leaves an incomplete calculus. One can either try and find a calculus that duplicates resources in a more subtle way or study the nature of non-terminating backwards search to see where one can stop the search.

The second calculus, PFLAX^{Hist}, is a theorem proving calculus for propositional Lax Logic. This calculus uses a technique for detecting loops using a history mechanism, building on work of Heuerding *et al* ([HSZ96], [Heu98], [How97]). It uses the propositional fragment of PFLAX as the basis calculus to which a history mechanism is added to give a decision procedure for propositional Lax Logic. This technique is general and maybe applied to many propositional logics. A decision procedure for propositional Lax Logic is interesting in relation to Lax Logic's application in hardware verification. We know of no other decision procedure for propositional Lax Logic.

2 Natural Deduction

The permutation-free calculus developed in this paper is a sequent calculus the proofs in which naturally correspond in a 1–1 way to normal natural deductions. In this section we discuss natural deduction for Lax Logic.

A natural deduction calculus for Lax Logic, taken directly from [BBdP98] (with rules for quantifiers and falsum added), can be seen in Figure 1.

Normal natural deductions are the objects of interest, so we look at the normalisation steps. Again these are taken from [BBdP98], with the extra cases for $\perp$ and $\exists_\varepsilon$ added. As the reduction rules for the intuitionistic connectives are standard, we do not include them here, concentrating instead on those involving the modality.

First the β-reduction:

–

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ P\end{array}}{\circ P}\,(\circ_{\mathcal{I}}) \qquad \begin{array}{c}[P]\\ \vdots\\ \circ Q\end{array}}{\circ Q}\,(\circ_{\varepsilon}) \quad\leadsto\quad \begin{array}{c}\vdots\\ [P]\\ \vdots\\ \circ Q\end{array}$$

Now we give the commuting conversions (or c-reductions) involving the modality:

–

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \circ P\end{array} \quad \begin{array}{c}[P]\\ \vdots\\ \circ Q\end{array}}{\circ Q}\,(\circ_{\varepsilon}) \qquad \begin{array}{c}[Q]\\ \vdots\\ \circ R\end{array}}{\circ R}\,(\circ_{\varepsilon}) \quad\leadsto\quad \dfrac{\begin{array}{c}\vdots\\ \circ P\end{array} \quad \dfrac{\begin{array}{c}[P]\\ \vdots\\ \circ Q\end{array} \quad \begin{array}{c}[Q]\\ \vdots\\ \circ R\end{array}}{\circ R}\,(\circ_{\varepsilon})}{\circ R}\,(\circ_{\varepsilon})$$

–

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ P \vee Q\end{array} \quad \begin{array}{c}[P]\\ \vdots\\ \circ R\end{array} \quad \begin{array}{c}[Q]\\ \vdots\\ \circ R\end{array}}{\circ R}\,(\vee_{\varepsilon}) \qquad \begin{array}{c}[R]\\ \vdots\\ \circ S\end{array}}{\circ S}\,(\circ_{\varepsilon}) \quad\leadsto\quad \dfrac{\begin{array}{c}\vdots\\ P \vee Q\end{array} \quad \dfrac{\begin{array}{c}[P]\\ \vdots\\ \circ R\end{array} \quad \begin{array}{c}[R]\\ \vdots\\ \circ S\end{array}}{\circ S}\,(\circ_{\varepsilon}) \quad \dfrac{\begin{array}{c}[Q]\\ \vdots\\ \circ R\end{array} \quad \begin{array}{c}[R]\\ \vdots\\ \circ S\end{array}}{\circ S}\,(\circ_{\varepsilon})}{\circ S}\,(\vee_{\varepsilon})$$

$$\dfrac{}{\Gamma, P \vdash P}\,(ax) \qquad \dfrac{}{\Gamma \vdash \top}\,(\top) \qquad \dfrac{\Gamma \vdash \bot}{\Gamma \vdash P}\,(\bot)$$

$$\dfrac{\Gamma, P \vdash Q}{\Gamma \vdash P \supset Q}\,(\supset_{\mathcal{I}}) \qquad \dfrac{\Gamma \vdash P \supset Q \quad \Gamma \vdash P}{\Gamma \vdash Q}\,(\supset_{\varepsilon})$$

$$\dfrac{\Gamma \vdash P \quad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}\,(\wedge_{\mathcal{I}}) \qquad \dfrac{\Gamma \vdash P \wedge Q}{\Gamma \vdash P}\,(\wedge_{\varepsilon_1}) \qquad \dfrac{\Gamma \vdash P \wedge Q}{\Gamma \vdash Q}\,(\wedge_{\varepsilon_2})$$

$$\dfrac{\Gamma \vdash P}{\Gamma \vdash P \vee Q}\,(\vee_{\mathcal{I}_1}) \qquad \dfrac{\Gamma \vdash Q}{\Gamma \vdash P \vee Q}\,(\vee_{\mathcal{I}_2})$$

$$\dfrac{\Gamma \vdash P \vee Q \quad \Gamma, P \vdash R \quad \Gamma, Q \vdash R}{\Gamma \vdash R}\,(\vee_{\varepsilon})$$

$$\dfrac{\Gamma \vdash P}{\Gamma \vdash \circ P}\,(\circ_{\mathcal{I}}) \qquad \dfrac{\Gamma \vdash \circ P \quad \Gamma, P \vdash \circ Q}{\Gamma \vdash \circ Q}\,(\circ_{\varepsilon})$$

$$\dfrac{\Gamma \vdash P[u/x]}{\Gamma \vdash \forall x P}\,(\forall_{\mathcal{I}})\dagger \qquad \dfrac{\Gamma \vdash \forall x P}{\Gamma \vdash P[t/x]}\,(\forall_{\varepsilon})$$

$$\dfrac{\Gamma \vdash P[t/x]}{\Gamma \vdash \exists x P}\,(\exists_{\mathcal{I}}) \qquad \dfrac{\Gamma \vdash \exists x P \quad \Gamma, P[u/x] \vdash R}{\Gamma \vdash R}\,(\exists_{\varepsilon})\dagger$$

† u not free in Γ

Fig. 1: Sequent style presentation of natural deduction for Lax Logic

–

$$\frac{\dfrac{\bot}{\circ P}\,(\bot_\varepsilon) \qquad \begin{matrix}[P]\\ \vdots \\ \circ Q\end{matrix}}{\circ Q}\,(\circ_\varepsilon) \qquad \leadsto \qquad \frac{\bot}{\circ Q}\,(\bot_\varepsilon)$$

–

$$\frac{\dfrac{\begin{matrix}\\ \vdots \\ \exists x P\end{matrix} \qquad \begin{matrix}[P[u/x]]\\ \vdots \\ \circ Q\end{matrix}}{\circ Q}\,(\exists_\varepsilon) \qquad \begin{matrix}[Q]\\ \vdots \\ \circ R\end{matrix}}{\circ R}\,(\circ_\varepsilon) \qquad \leadsto \qquad \frac{\begin{matrix}\vdots \\ \exists x P\end{matrix} \qquad \dfrac{\begin{matrix}[P[u/x]]\\ \vdots \\ \circ Q\end{matrix} \qquad \begin{matrix}[Q]\\ \vdots \\ \circ R\end{matrix}}{\circ R}\,(\circ_\varepsilon)}{\circ R}\,(\exists_\varepsilon)$$

Definition 1 *A natural deduction is said to be in* β, c**-normal form** *when no* β*-reductions and no* c*-reductions are applicable.*

We give a presentation of a restricted version of natural deduction for Lax Logic. In this calculus, the only deductions are those that are in β, c-normal form. This calculus has two kinds of 'sequents', differentiated by their consequence relations, $\rhd$ and $\rhd\!\rhd$. Rules are applicable only when the premisses have a certain consequence relation. The conclusions have a fixed consequence relation. Thus those deductions that are valid are of a restricted form. This calculus, which we call NLAX, is given (with the proof terms given in the next section) in Figure 2.

Proposition 1 *The calculus* NLAX *only allows deductions to which no* β*-reductions and no* c*-reductions are applicable. Moreover, it allows all* β, c*-normal deductions.*

3 Term Assignment

In this section we give a term assignment system for NLAX. The term system is not only needed for the proofs given later in the paper, but also extremely useful in many applications of constructive logics. In [Mog89] Moggi gave a λ-calculus, which he called the *computational* λ*-calculus*. As is shown in [BBdP98], this calculus naturally matches Lax Logic. More about the computational λ-calculus and Lax Logic (there called computational logic) can be found in [BBdP98].

We give this term system in a syntax we prefer – an abstract syntax with explicit constructors. We could give a translation of Moggi's terms to ours. Proof terms for unrestricted natural deduction for Lax Logic and a translation of Moggi's terms to ours (and vice versa) can be found in [How98a].

We are interested in the 'real' proofs for Lax Logic – the normal natural deductions. We now restrict the terms that can be built, in order that they match our restricted natural deduction calculus NLAX, giving us proof objects. (That is, no reductions will be applicable at the term level; the term reductions match the β- and c-reductions given earlier). The proof terms come in two syntactic categories, **A** and **N**. V is the category of variables (proofs), U is the category of variables (individuals), and T the category of terms. The extra constructor $an(A)$ matches the (M) rule of NLAX.

A ::=

$$var(V) \mid ap(A, N) \mid fst(A) \mid snd(A) \mid apn(A, T)$$

N ::=

$$\begin{gathered} * \mid efq(A) \mid an(A) \mid \lambda V.N \mid pr(N, N) \mid i(N) \mid j(N) \\ wn(A, V.N, V.N) \mid smhi(N) \mid smhe(A, V.N) \\ \lambda U.N \mid prq(T, N) \mid ee(A, U.V.N) \end{gathered}$$

NLAX together with proof annotations for normal terms can be seen in Figure 2.

$$\frac{}{\Gamma, x : P \rhd var(x) : P}\ (ax) \qquad \frac{\Gamma \rhd A : P}{\Gamma \rhd\!\rhd an(A) : P}\ (M)$$

$$\frac{}{\Gamma \rhd\!\rhd * : \top}\ (\top) \qquad \frac{\Gamma \rhd A : \bot}{\Gamma \rhd\!\rhd efq(A) : P}\ (\bot_{\mathcal{E}})$$

$$\frac{\Gamma, x : P \rhd\!\rhd N : Q}{\Gamma \rhd\!\rhd \lambda x.N : P \supset Q}\ (\supset_{\mathcal{I}}) \qquad \frac{\Gamma \rhd A : P \supset Q \quad \Gamma \rhd\!\rhd N : P}{\Gamma \rhd ap(A, N) : Q}\ (\supset_{\mathcal{E}})$$

$$\frac{\Gamma \rhd\!\rhd N_1 : P \quad \Gamma \rhd\!\rhd N_2 : Q}{\Gamma \rhd\!\rhd pr(N_1, N_2) : P \wedge Q}\ (\wedge_{\mathcal{I}})$$

$$\frac{\Gamma \rhd A : P \wedge Q}{\Gamma \rhd fst(A) : P}\ (\wedge_{\mathcal{E}_1}) \qquad \frac{\Gamma \rhd A : P \wedge Q}{\Gamma \rhd snd(A) : Q}\ (\wedge_{\mathcal{E}_2})$$

$$\frac{\Gamma \rhd\!\rhd N : P}{\Gamma \rhd\!\rhd i(N) : P \vee Q}\ (\vee_{\mathcal{I}_1}) \qquad \frac{\Gamma \rhd\!\rhd N : Q}{\Gamma \rhd\!\rhd j(N) : P \vee Q}\ (\vee_{\mathcal{I}_2})$$

$$\frac{\Gamma \rhd A : P \vee Q \quad \Gamma, x_1 : P \rhd\!\rhd N_1 : R \quad \Gamma, x_2 : Q \rhd\!\rhd N_2 : R}{\Gamma \rhd\!\rhd wn(A, x_1.N_1, x_2.N_2) : R}\ (\vee_{\mathcal{E}})$$

$$\frac{\Gamma \rhd\!\rhd N : P}{\Gamma \rhd\!\rhd smhi(N) : \circ P}\ (\circ_{\mathcal{I}}) \qquad \frac{\Gamma \rhd A : \circ P \quad \Gamma, x : P \rhd\!\rhd N : \circ Q}{\Gamma \rhd\!\rhd smhe(A, x.N) : \circ Q}\ (\circ_{\mathcal{E}})$$

$$\frac{\Gamma \rhd\!\rhd N : P[u/x]}{\Gamma \rhd\!\rhd \lambda u.N : \forall x P}\ (\forall_{\mathcal{I}})\dagger \qquad \frac{\Gamma \rhd A : \forall x P}{\Gamma \rhd apn(A, t) : P[t/x]}\ (\forall_{\mathcal{E}})$$

$$\frac{\Gamma \rhd\!\rhd N : P[t/x]}{\Gamma \rhd\!\rhd prq(t, N) : \exists x P}\ (\exists_{\mathcal{I}}) \qquad \frac{\Gamma \rhd A : \exists x P \quad \Gamma, x : P[u/x] \rhd\!\rhd N : R}{\Gamma \rhd\!\rhd ee(A, u.x.N) : R}\ (\exists_{\mathcal{E}})\dagger$$

† u not free in Γ

Fig. 2: NLAX with proof annotations

4 Sequent Calculus

In this section we present a new Gentzen sequent calculus for Lax Logic, PFLAX. The proofs allowed by PFLAX naturally correspond in a 1–1 way to normal natural deductions for Lax Logic – i.e. the proofs of NLAX. First we remind the reader of the sequent calculus as presented in [FM97] and [BBdP98]. This can be seen in Figure 3.

In fact, our presentation is slightly different from both those cited. The calculus in [BBdP98] has no structural rules, that is, the contexts are sets. [FM97] have both weakening and contraction on both the left and the right, plus exchange. Here the only structural rule we consider (or need) is contraction on the left. The contexts in our presentation are multisets. We leave all discussion of cut until later.

$$\frac{}{\Gamma, P \Rightarrow P}\ (ax) \qquad \frac{\Gamma, P, P \Rightarrow R}{\Gamma, P \Rightarrow R}\ (C)$$

$$\frac{}{\Gamma \Rightarrow \top}\ (\top) \qquad \frac{}{\Gamma, \bot \Rightarrow P}\ (\bot)$$

$$\frac{\Gamma, P \Rightarrow Q}{\Gamma \Rightarrow P \supset Q}\ (\supset_{\mathcal{R}}) \qquad \frac{\Gamma \Rightarrow P \quad \Gamma, Q \Rightarrow R}{\Gamma, P \supset Q \Rightarrow R}\ (\supset_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P \quad \Gamma \Rightarrow Q}{\Gamma \Rightarrow P \wedge Q}\ (\wedge_{\mathcal{R}}) \qquad \frac{\Gamma, P \Rightarrow R}{\Gamma, P \wedge Q \Rightarrow R}\ (\wedge_{\mathcal{L}_1}) \qquad \frac{\Gamma, Q \Rightarrow R}{\Gamma, P \wedge Q \Rightarrow R}\ (\wedge_{\mathcal{L}_2})$$

$$\frac{\Gamma \Rightarrow P}{\Gamma \Rightarrow P \vee Q}\ (\vee_{\mathcal{R}_1}) \qquad \frac{\Gamma \Rightarrow Q}{\Gamma \Rightarrow P \vee Q}\ (\vee_{\mathcal{R}_2}) \qquad \frac{\Gamma, P \Rightarrow R \quad \Gamma, Q \Rightarrow R}{\Gamma, P \vee Q \Rightarrow R}\ (\vee_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P}{\Gamma \Rightarrow \circ P}\ (\circ_{\mathcal{R}}) \qquad \frac{\Gamma, P \Rightarrow \circ R}{\Gamma, \circ P \Rightarrow \circ R}\ (\circ_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P[y/x]}{\Gamma \Rightarrow \forall x P}\ (\forall_{\mathcal{R}})\dagger \qquad \frac{\Gamma, P[t/x] \Rightarrow R}{\Gamma, \forall x P \Rightarrow R}\ (\forall_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P[t/x]}{\Gamma \Rightarrow \exists x P}\ (\exists_{\mathcal{R}}) \qquad \frac{\Gamma, P[y/x] \Rightarrow R}{\Gamma, \exists x P \Rightarrow R}\ (\exists_{\mathcal{L}})\dagger$$

† y not free in Γ

Fig. 3: Sequent Calculus for Lax Logic

We now present a new sequent calculus which we call PFLAX ('permutation-free' Lax Logic). This calculus extends the permutation-free calculus MJ ([Her95],[Her96], where the calculus is called LJT, and [DP98],[DP99]) for Intuitionistic Logic to a calculus for Lax Logic. Like MJ this calculus has two forms of judgment, $\Gamma \Rightarrow R$ and $\Gamma \xrightarrow{Q} R$. The first looks like the usual kind of sequent; however, only right rules and contraction are applicable to this kind of sequent in backwards proof search. By backwards proof search we mean proof search starting from the root. The second kind of sequent has a formula (on the left) in a privileged position called the *stoup* (following [Gir91]). The formula in the stoup is always principal in the conclusion of an inference rule. Left rules are only applicable to stoup sequents. PFLAX is displayed in Figure 4.

$$\frac{}{\Gamma \xrightarrow{P} P}\ (ax) \qquad \frac{\Gamma, P \xrightarrow{P} R}{\Gamma, P \Rightarrow R}\ (C)$$

$$\frac{}{\Gamma \Rightarrow \top}\ (\top) \qquad \frac{}{\Gamma \xrightarrow{\perp} P}\ (\perp)$$

$$\frac{\Gamma, P \Rightarrow Q}{\Gamma \Rightarrow P \supset Q}\ (\supset_{\mathcal{R}}) \qquad \frac{\Gamma \Rightarrow P \quad \Gamma \xrightarrow{Q} R}{\Gamma \xrightarrow{P \supset Q} R}\ (\supset_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P \quad \Gamma \Rightarrow Q}{\Gamma \Rightarrow P \wedge Q}\ (\wedge_{\mathcal{R}}) \qquad \frac{\Gamma \xrightarrow{P} R}{\Gamma \xrightarrow{P \wedge Q} R}\ (\wedge_{\mathcal{L}_1}) \qquad \frac{\Gamma \xrightarrow{Q} R}{\Gamma \xrightarrow{P \wedge Q} R}\ (\wedge_{\mathcal{L}_2})$$

$$\frac{\Gamma \Rightarrow P}{\Gamma \Rightarrow P \vee Q}\ (\vee_{\mathcal{R}_1}) \qquad \frac{\Gamma \Rightarrow Q}{\Gamma \Rightarrow P \vee Q}\ (\vee_{\mathcal{R}_2}) \qquad \frac{\Gamma, P \Rightarrow R \quad \Gamma, Q \Rightarrow R}{\Gamma \xrightarrow{P \vee Q} R}\ (\vee_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P}{\Gamma \Rightarrow \circ P}\ (\circ_{\mathcal{R}}) \qquad \frac{\Gamma, P \Rightarrow \circ R}{\Gamma \xrightarrow{\circ P} \circ R}\ (\circ_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P[y/x]}{\Gamma \Rightarrow \forall x P}\ (\forall_{\mathcal{R}})\dagger \qquad \frac{\Gamma \xrightarrow{P[t/x]} R}{\Gamma \xrightarrow{\forall x P} R}\ (\forall_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow P[t/x]}{\Gamma \Rightarrow \exists x P}\ (\exists_{\mathcal{R}}) \qquad \frac{\Gamma, P[y/x] \Rightarrow R}{\Gamma \xrightarrow{\exists x P} R}\ (\exists_{\mathcal{L}})\dagger$$

† y not free in Γ

Fig. 4: The Sequent Calculus PFLAX

The stoup is a form of focusing: the formula in the stoup is always principal in the premiss unless it is a disjunction or a somehow formula. One might ask why we do not formulate the $(\circ_{\mathcal{L}})$ rule as follows

$$\frac{\Gamma \xrightarrow{P} \circ R}{\Gamma \xrightarrow{\circ P} \circ R}\ (\circ_{\mathcal{L}})$$

To answer this, we point out that the resulting calculus would not then match normal natural deductions in the manner we would like. Also consider proofs of the sequent $\circ\circ(P \wedge Q) \Rightarrow \circ(Q \wedge P)$.

5 Term Assignment for Sequent Calculus

We give a term assignment system for PFLAX; these terms, together with those from section 3, will be used to prove the main results on PFLAX. This we get by extending that given in [Her95], [DP96], [DP98]. The term calculus has two syntactic categories, **M** and **Ms**. V is the category of variables (proofs), U is the category of variables (individuals) and T is the category of terms .

M::=

$$* \mid (V; Ms) \mid \lambda V.M \mid pair(M, M) \mid inl(M) \mid inr(M)$$

$$smhr(M) \mid \lambda U.M \mid pairq(T, M)$$

Ms::=

$$[\,] \mid ae \mid (M :: Ms) \mid p(Ms) \mid q(Ms) \mid when(V.M, V.M)$$
$$smhl(V.M) \mid apq(T, Ms) \mid spl(U.V.M)$$

These terms can easily be typed by PFLAX, as seen in Figure 5.

$$\frac{}{\Gamma \xrightarrow{P} [\,] : P}\ (ax) \qquad \frac{\Gamma, x : P \xrightarrow{P} Ms : R}{\Gamma, x : P \Rightarrow (x; Ms) : R}\ (C)$$

$$\frac{}{\Gamma \Rightarrow * : \top}\ (\top) \qquad \frac{}{\Gamma \xrightarrow{\perp} ae : \perp}\ (\perp)$$

$$\frac{\Gamma, x : P \Rightarrow M : Q}{\Gamma \Rightarrow \lambda x.M : P \supset Q}\ (\supset_{\mathcal{R}}) \qquad \frac{\Gamma \Rightarrow M : P \quad \Gamma \xrightarrow{Q} Ms : R}{\Gamma \xrightarrow{P \supset Q} (M :: Ms) : R}\ (\supset_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow M_1 : P \quad \Gamma \Rightarrow M_2 : Q}{\Gamma \Rightarrow pair(M_1, M_2) : P \wedge Q}\ (\wedge_{\mathcal{R}})$$

$$\frac{\Gamma \xrightarrow{P} Ms : R}{\Gamma \xrightarrow{P \wedge Q} p(Ms) : R}\ (\wedge_{\mathcal{L}_1}) \qquad \frac{\Gamma \xrightarrow{Q} Ms : R}{\Gamma \xrightarrow{P \wedge Q} q(Ms) : R}\ (\wedge_{\mathcal{L}_2})$$

$$\frac{\Gamma \Rightarrow M : P}{\Gamma \Rightarrow inl(M) : P \vee Q}\ (\vee_{\mathcal{R}_1}) \qquad \frac{\Gamma \Rightarrow M : Q}{\Gamma \Rightarrow inr(M) : P \vee Q}\ (\vee_{\mathcal{R}_2})$$

$$\frac{\Gamma, x_1 : P \Rightarrow M_1 : R \quad \Gamma, x_2 : Q \Rightarrow M_2 : R}{\Gamma \xrightarrow{P \vee Q} when(x_1.M_1, x_2.M_2) : R}\ (\vee_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow M : P}{\Gamma \Rightarrow smhr(M) : \circ P}\ (\circ_{\mathcal{R}}) \qquad \frac{\Gamma, x : P \Rightarrow M : \circ R}{\Gamma \xrightarrow{\circ P} smhl(x.M) : \circ R}\ (\circ_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow M : P[u/x]}{\Gamma \Rightarrow \lambda u.M : \forall x P}\ (\forall_{\mathcal{R}})\dagger \qquad \frac{\Gamma \xrightarrow{P[t/x]} Ms : R}{\Gamma \xrightarrow{\forall x P} apq(t, Ms) : R}\ (\forall_{\mathcal{L}})$$

$$\frac{\Gamma \Rightarrow M : P[t/x]}{\Gamma \Rightarrow pairq(T, M) : \exists x P}\ (\exists_{\mathcal{R}}) \qquad \frac{\Gamma, P[u/x] \Rightarrow M : R}{\Gamma \xrightarrow{\exists x P} spl(u.x.M) : R}\ (\exists_{\mathcal{L}})\dagger$$

† y not free in Γ

Fig. 5: The Sequent Calculus PFLAX, with Term Assignment

6 Equivalence of the Calculi

Having presented the calculi for Lax Logic, we now prove that they have the properties we claim. We prove the equivalence of the term calculi and soundness and adequacy for PFLAX. These results prove the desired correspondence.

The full details of these proofs are rather repetitive: therefore we only give the proofs for the $\supset, \circ$ fragment of Lax Logic. The remainder of the calculus is Intuitionistic Logic as presented in [DP96]. The details of the proofs extended to the rest of the calculus can be found there.

We start by giving pairs of functions that define translations between the term assignment systems for natural deduction and sequent calculus.

Sequent Calculus $\rightarrow$ Natural Deduction:

$\theta : \mathbf{M} \rightarrow \mathbf{N}$	$\theta' : \mathbf{A} \times \mathbf{Ms} \rightarrow \mathbf{N}$
$\theta(x; Ms) = \theta'(var(x), Ms)$	$\theta'(A, [\,]) = an(A)$
$\theta(\lambda x.M) = \lambda x.\theta(M)$	$\theta'(A, (M :: Ms)) = \theta'(ap(A, \theta(M)), Ms)$
$\theta(smhr(M)) = smhi(\theta(M))$	$\theta'(A, smhl(x.Ms)) = smhe(A, x.\theta(M))$

Natural Deduction $\rightarrow$ Sequent Calculus:

$\psi : \mathbf{N} \rightarrow \mathbf{M}$	$\psi' : \mathbf{A} \times \mathbf{Ms} \rightarrow \mathbf{M}$
$\psi(an(A)) = \psi'(A, [\,])$	$\psi'(var(x), Ms) = (x; Ms)$
$\psi(\lambda x.N) = \lambda x.\psi(N)$	$\psi'(ap(A, N), Ms) = \psi'(A, (\psi(N) :: Ms))$
$\psi(smhe(A, x.N)) = \psi'(A, smhl(x.\psi(N)))$	
$\psi(smhi(N)) = smhr(\psi(N))$	

We prove two lemmas showing the equivalence of the term calculi, that is, the translations from one system to the other are 1–1.

Lemma 1 **i)** $\psi(\theta(M)) \;=\; M$; **ii)** $\psi(\theta'(A, Ms)) \;=\; \psi'(A, Ms)$

PROOF: The proof is by simultaneous induction on the structure of M and Ms.

1. $(x; Ms)$
$$\begin{aligned} \psi(\theta(x; Ms)) &= \psi(\theta'(var(x), Ms)) && \text{def } \theta \\ &= \psi'(var(x), Ms) && \text{ind ii)} \\ &= (x; Ms) && \text{def } \psi' \end{aligned}$$
2. $\lambda x.M$
$$\begin{aligned} \psi(\theta(\lambda x.M)) &= \psi(\lambda x.\theta(M)) && \text{def } \theta \\ &= \lambda x.\psi(\theta(M)) && \text{def } \psi \\ &= \lambda x.M && \text{ind i)} \end{aligned}$$
3. $smhr(M)$
$$\begin{aligned} \psi(\theta(smhr(M))) &= \psi(smhi(\theta(M))) && \text{def } \theta \\ &= smhr(\psi(\theta(M))) && \text{def } \psi \\ &= smhr(M) && \text{ind i)} \end{aligned}$$
4. $[\,]$
$$\begin{aligned} \psi(\theta'(A, [\,])) &= \psi(an(A)) && \text{def } \theta \\ &= \psi'(A, [\,]) && \text{def } \psi' \end{aligned}$$

5. $(M :: Ms)$
$$\begin{aligned}\psi(\theta'(A,(M :: Ms))) &= \psi(\theta'(ap(A,\theta(M)), Ms)) && \text{def } \theta' \\ &= \psi'(ap(A,\theta(M)), Ms) && \text{ind ii)} \\ &= \psi'(A, (\psi(\theta(M)) :: Ms)) && \text{def } \psi' \\ &= \psi'(A, (M :: Ms)) && \text{ind i)}\end{aligned}$$
6. $smhl(x.M)$
$$\begin{aligned}\psi(\theta'(A, smhl(x.M))) &= \psi(smhe(A, x.\theta(M))) && \text{def } \theta' \\ &= \psi'(A, smhl(x.\psi(\theta(M)))) && \text{def } \psi \\ &= \psi'(A, smhl(x.M)) && \text{ind i)}\end{aligned}$$

■

Lemma 2 **i)** $\theta(\psi(N)) \;=\; N$; **ii)** $\theta(\psi'(A, Ms)) \;=\; \theta'(A, Ms)$

PROOF: By simultaneous induction on the structure of N and A.

1. $an(A)$
$$\begin{aligned}\theta(\psi(an(A))) &= \theta(\psi'(A,[\,])) && \text{def } \psi \\ &= \theta'(A,[\,]) && \text{ind ii)} \\ &= an(A) && \text{def } \theta'\end{aligned}$$
2. $\lambda x.N$
$$\begin{aligned}\theta(\psi(\lambda x.N)) &= \theta(\lambda x.\psi(N)) && \text{def } \psi \\ &= \lambda x.\theta(\psi(N)) && \text{def } \theta \\ &= \lambda x.N && \text{ind i)}\end{aligned}$$
3. $smhi(N)$
$$\begin{aligned}\theta(\psi(smhi(N))) &= \theta(smhr(\psi(N))) && \text{def } \psi \\ &= smhi(\theta(\psi(N))) && \text{def } \theta \\ &= smhi(N) && \text{ind i)}\end{aligned}$$
4. $smhe(A, x.N)$
$$\begin{aligned}\theta(\psi(smhe(A, x.N))) &= \theta(\psi'(A, smhl(x.\psi(N)))) && \text{def } \psi \\ &= \theta'(A, smhl(x.\psi(N))) && \text{ind ii)} \\ &= smhe(A, x.\theta(\psi(N))) && \text{def } \theta' \\ &= smhe(A, x.N) && \text{ind i)}\end{aligned}$$
5. $var(x)$
$$\begin{aligned}\theta(\psi'(var(x), Ms)) &= \theta(x; Ms) && \text{def } \psi' \\ &= \theta'(var(x), Ms) && \text{def } \theta\end{aligned}$$
6. $ap(A, N)$
$$\begin{aligned}\theta(\psi'(ap(A,N), Ms)) &= \theta(\psi'(A, (\psi(N) :: Ms))) && \text{def } \psi' \\ &= \theta'(A, (\psi(N) :: Ms)) && \text{ind ii)} \\ &= \theta'(ap(A, \theta(\psi(N))), Ms) && \text{def } \theta' \\ &= \theta'(ap(A,N), Ms) && \text{ind i)}\end{aligned}$$

■

The following two theorems prove soundness and adequacy theorems. These show that the translations respect provability, that is, no sequent (and hence its associated term) can be proved in one system, but not its translation in the other.

Theorem 1 (SOUNDNESS) *The following rules are admissible:*

$$\frac{\Gamma \Rightarrow M : R}{\Gamma \rhd\!\!\rhd \theta(M) : R}\ i) \qquad \frac{\Gamma \rhd A : P \quad \Gamma \xrightarrow{P} Ms : R}{\Gamma \rhd\!\!\rhd \theta'(A, Ms) : R}\ ii)$$

PROOF: By simultaneous induction on the structure of M and Ms.

1. $(x; Ms)$ We have a derivation ending in:

$$\frac{\Gamma, x : P \xrightarrow{P} Ms : R}{\Gamma, x : P \Rightarrow (x; Ms) : R}\ (C)$$

   So we have:

$$\frac{x : P \rhd var(x) : P \quad \Gamma, x : P \xrightarrow{P} Ms : R}{\Gamma \rhd\!\!\rhd \theta'(var(x), Ms) : R}\ ii)$$

   and we know that $\theta'(var(x), Ms) = \theta(x; Ms)$
2. $\lambda x.M$ We have a derivation ending in

$$\frac{\Gamma, x : P \Rightarrow M : Q}{\Gamma \Rightarrow \lambda x.M : P \supset Q}\ (\supset_{\mathcal{R}})$$

   whence

$$\frac{\dfrac{\Gamma, x : P \Rightarrow M : Q}{\Gamma, x : P \rhd\!\!\rhd \theta(M) : Q}\ i)}{\Gamma \rhd\!\!\rhd \lambda x.\theta(M) : P \supset Q}\ (\supset_{\mathcal{I}})$$

   and we know that $\lambda x.\theta(M) = \theta(\lambda x.M)$
3. $smhr(M)$ We have a derivation ending as follows

$$\frac{\Gamma \Rightarrow M : P}{\Gamma \Rightarrow smhr(M) : \circ P}\ (\circ_{\mathcal{R}})$$

   whence

$$\frac{\dfrac{\Gamma \Rightarrow M : P}{\Gamma \rhd\!\!\rhd \theta(M) : P}\ i)}{\Gamma \rhd\!\!\rhd smhi(\theta(M)) : \circ P}\ (\circ_{\mathcal{I}})$$

   and we know that $smhi(\theta(M)) = \theta(smhr(M))$
4. [ ] We have a deduction and a derivation:

$$\Gamma \rhd A : P \qquad \frac{}{\Gamma \xrightarrow{P} [\,] : P}\ (ax)$$

   From the deduction, we obtain:

$$\frac{\Gamma \rhd A : P}{\Gamma \rhd\!\!\rhd an(A) : P}\ (M)$$

   the required results follows since $an(A) = \theta'(A, [\,])$

5. $(M :: Ms)$ We have a derivation ending in

$$\frac{\Gamma \Rightarrow M : P \quad \Gamma \xrightarrow{Q} Ms : R}{\Gamma \xrightarrow{P \supset Q} (M :: Ms) : R} \; (\supset_{\mathcal{L}})$$

whence

$$\frac{\dfrac{\Gamma \rhd A : P \supset Q \quad \dfrac{\Gamma \Rightarrow M : P}{\Gamma \rhd\!\!\rhd \theta(M) : P}\; i)}{\Gamma \rhd ap(A, \theta(M)) : Q} \; (\supset_{\varepsilon}) \quad \Gamma \xrightarrow{Q} Ms : R}{\Gamma \rhd\!\!\rhd \theta'(ap(A, \theta(M)), Ms) : R} \; ii)$$

and we know that $\theta'(ap(A, \theta(M)), Ms) = \theta'(A, (M :: Ms))$

6. $smhl(x.Ms)$ We have a derivation ending

$$\frac{\Gamma, x : P \Rightarrow M : \circ Q}{\Gamma \xrightarrow{\circ P} smhl(x.M) : \circ Q} \; (\circ_{\mathcal{L}})$$

whence

$$\frac{\Gamma \rhd A : \circ P \quad \dfrac{\Gamma, x : P \Rightarrow M : \circ Q}{\Gamma, x : P \rhd\!\!\rhd \theta(M) : \circ Q}\; i)}{\Gamma \rhd\!\!\rhd smhe(A, x.\theta(M)) : \circ Q} \; (\circ_{\varepsilon})$$

and we know that $smhe(A, x.\theta(M)) = \theta'(A, smhl(x.M))$

■

Theorem 2 (ADEQUACY) *The following rules are admissible:*

$$\frac{\Gamma \rhd\!\!\rhd N : R}{\Gamma \Rightarrow \psi(N) : R} \; i) \qquad \frac{\Gamma \rhd A : P \quad \Gamma \xrightarrow{P} Ms : R}{\Gamma \Rightarrow \psi'(A, Ms) : R} \; ii)$$

PROOF: By simultaneous induction on the structure of A and N.

1. $an(A)$ We have a deduction ending

$$\frac{\Gamma \rhd A : P}{\Gamma \rhd\!\!\rhd an(A) : P} \; (M)$$

hence we have

$$\frac{\Gamma \rhd A : P \quad \Gamma \xrightarrow{P} [\,] : P}{\Gamma \Rightarrow \psi'(A, [\,]) : P} \; ii)$$

and we know that $\psi'(A, [\,]) = \psi(an(A))$

2. $\lambda x.N$ We have a deduction ending

$$\frac{\Gamma, x : P \rhd\!\rhd N : Q}{\Gamma \rhd\!\rhd \lambda x.N : P \supset Q} \ (\supset_{\mathcal{I}})$$

whence

$$\frac{\dfrac{\Gamma, x : P \rhd\!\rhd N : Q}{\Gamma, x : P \Rightarrow \psi(N) : Q}\ i)}{\Gamma \Rightarrow \lambda x.\psi(N) : P \supset Q} \ (\supset_{\mathcal{R}})$$

and we know that $\lambda x.\psi(N) = \psi(\lambda x.N)$

3. $smhe(A, x.N)$ We have a deduction ending in

$$\frac{\Gamma \rhd A : \circ P \quad \Gamma, x : P \rhd\!\rhd N : \circ Q}{\Gamma \rhd\!\rhd smhe(A, x.N) : \circ Q} \ (\circ_{\varepsilon})$$

whence

$$\frac{\Gamma \rhd A : \circ P \quad \dfrac{\dfrac{\Gamma, x : P \rhd\!\rhd N : \circ Q}{\Gamma, x : P \Rightarrow \psi(N) : \circ Q}\ i)}{\Gamma \xrightarrow{\circ P} smhl(x.\psi(N)) : \circ Q}\ (\circ_{\mathcal{L}})}{\Gamma \Rightarrow \psi'(A, smhl(x.\psi(N))) : \circ Q} \ ii)$$

and we know that $\psi'(A, smhl(x.\psi(N))) = \psi(smhe(A, x.N))$

4. $smhi(N)$ We have a deduction ending in

$$\frac{\Gamma \rhd\!\rhd N : P}{\Gamma \rhd\!\rhd smhi(N) : \circ P} \ (\circ_{\mathcal{I}})$$

whence

$$\frac{\dfrac{\Gamma \rhd\!\rhd N : P}{\Gamma \Rightarrow \psi(N) : P}\ i)}{\Gamma \Rightarrow smhr(\psi(N)) : \circ P} \ (\circ_{\mathcal{R}})$$

and we know that $smhr(\psi(N)) = \psi(smhi(N))$

5. $var(x)$ We can extend to

$$\frac{\Gamma, x : P \xrightarrow{P} Ms : R}{\Gamma, x : P \Rightarrow (x; Ms) : R} \ (C)$$

and we know that $(x; Ms) = \psi'(var(x), Ms)$

6. $ap(A, N)$ We have a deduction ending in

$$\frac{\Gamma \rhd A : P \supset Q \quad \Gamma \rhd N : P}{\Gamma \rhd ap(A, N) : Q} \ (\supset_{\varepsilon})$$

whence

$$\frac{\Gamma \rhd A : P \supset Q \qquad \dfrac{\dfrac{\Gamma \rhd\!\rhd N : P}{\Gamma \Rightarrow \psi(N) : P}\ i) \quad \Gamma \xrightarrow{Q} Ms : R}{\Gamma \xrightarrow{P \supset Q} (\psi(N) :: Ms) : R}\ (\supset_{\mathcal{L}})}{\Gamma \Rightarrow \psi'(A, (\psi(N) :: Ms)) : R} \ ii)$$

and we know that $\psi'(A, (\psi(N) :: Ms)) = \psi'(ap(A, N), Ms)$

■

Since the term systems are in 1–1 correspondence (from lemma 1 and lemma 2) and the translation preserve provability (theorem 1 and theorem 2), the 1–1 correspondence between PFLAX and NLAX has been established. This is stated in the following theorem.

Theorem 3 *The normal natural deductions of Lax Logic (the proofs of* NLAX*) are in 1–1 correspondence to the proofs of* PFLAX.

PROOF: Immediate from theorems 1 and 2 and lemmas 1 and 2. ■

Since natural deduction is sound and complete, PFLAX must also be.

Corollary 1 *The calculus* PFLAX *is sound and complete.*

6.1 Cut Elimination

We now discuss cut for PFLAX. In the usual sequent calculus, cut may be formulated as follows:

$$\frac{\Gamma \Rightarrow P \quad \Gamma, P \Rightarrow Q}{\Gamma \Rightarrow Q}\ (cut)$$

In PFLAX, the two judgment forms lead to the following four cut rules:

$$\frac{\Gamma \xrightarrow{Q} P \quad \Gamma \xrightarrow{P} R}{\Gamma \xrightarrow{Q} R}\ (cut_1) \qquad \frac{\Gamma \Rightarrow P \quad \Gamma, P \xrightarrow{Q} R}{\Gamma \xrightarrow{Q} R}\ (cut_2)$$

$$\frac{\Gamma \Rightarrow P \quad \Gamma \xrightarrow{P} R}{\Gamma \Rightarrow R}\ (cut_3) \qquad \frac{\Gamma \Rightarrow P \quad \Gamma, P \Rightarrow R}{\Gamma \Rightarrow R}\ (cut_4)$$

We call PFLAX extended with the four cut rules PFLAX^{cut}. We can give reduction rules for PFLAX^{cut} and prove the weak cut elimination theorem for the logic. We can also prove strong normalisation for the term system associated with the logic, hence strong cut-elimination. As these results are not directly relevant to the work presented in this paper, the details have been omitted. These details and proofs can be found in [How98a], [How98b].

Theorem 4 *The rules* $(cut_1), (cut_2), (cut_3), (cut_4)$ *are admissible in* PFLAX.

Theorem 5 *The cut reduction system for* PFLAX *strongly normalises.*

7 Lax Logic and Constraint Logic Programming

In [FMW97] and [Wal97], quantified Lax Logic is used to give a logical analysis of constraint logic programming. Lax Logic is used to separate the logical analysis of provability and the satisfiability of constraints. Here we summarise their approach.

Constraint logic programs consist of clauses, CLP clauses, which are closed formulae of the form:

$$\forall x_1 ... x_n . S \supset H$$

where H is an atom $A(x_1, ..., x_n)$ and S is a formula according to the following grammar:

S::=

$$\top \mid A \mid S \vee S \mid S \wedge S \mid \exists V.S$$

These clauses can contain constraints. An example of a constraint logic program clause is

$$\forall s.s \geq 5 \supset A(s)$$

Queries (goal formulae) are also formulae of **S**. Queries contain no constraints.

Lax Logic is used to separate the constraints from the logical parts of the programs. This is done by a simple procedure: replace all occurrences of constraints in S by $\top$ and modalise the head. For example:

$$\forall s.s \geq 5 \supset A(s) \quad \text{becomes} \quad \forall s.\top \supset \circ A(s)$$

The constraint can be encoded as a special kind of lambda term.

The result of this abstraction is called a Lax Logic program clause (LLP clause). These have the form:

$$\forall x_1 ... x_n . S \supset \circ H$$

where S and H are as for constraint logic program clauses (except that no constraints are allowed in S). Note the constraint program clauses and Lax Logic program clauses are part of the same logic (quantified Lax Logic) and so programs with LLP clauses and constraint-free CLP clauses can be reasoned about together.

If we want to answer a query Q from a program containing LLP clauses, then we try to prove formula $\circ Q$, meaning that Q is proved up to the satisfaction of some, as yet unspecified, constraints. This is done using the natural deduction calculus given in Figure 6.

For any query, we get one or many proofs from the program by using the LLP calculus. This gives us different solutions up to the satisfaction of constraints. What these constraints are differs for each proof. Using the proof term system for the LLP calculus, together with the lambda term (in a different system) encoding the abstracted constraints, the actual constraints to be satisfied can be calculated and then solved using suitable machinery.

$$\frac{}{\Gamma \Rightarrow \top}\ (\top_{\mathcal{I}}) \qquad \frac{}{\Gamma \Rightarrow \circ\top}\ (\circ\top_{\mathcal{I}})$$

$$\frac{\Gamma \Rightarrow P \quad \Gamma \Rightarrow Q}{\Gamma \Rightarrow P \wedge Q}\ (\wedge_{\mathcal{I}}) \qquad \frac{\Gamma \Rightarrow \circ P \quad \Gamma \Rightarrow \circ Q}{\Gamma \Rightarrow \circ(P \wedge Q)}\ (\circ\wedge_{\mathcal{I}})$$

$$\frac{\Gamma \Rightarrow P}{\Gamma \Rightarrow P \vee Q}\ (\vee_{\mathcal{I}_1}) \qquad \frac{\Gamma \Rightarrow Q}{\Gamma \Rightarrow P \vee Q}\ (\vee_{\mathcal{I}_2})$$

$$\frac{\Gamma \Rightarrow \circ P}{\Gamma \Rightarrow \circ(P \vee Q)}\ (\circ\vee_{\mathcal{I}_1}) \qquad \frac{\Gamma \Rightarrow \circ Q}{\Gamma \Rightarrow \circ(P \vee Q)}\ (\circ\vee_{\mathcal{I}_2})$$

$$\frac{\Gamma \Rightarrow P[t/x]}{\Gamma \Rightarrow \exists x.P}\ (\exists_{\mathcal{I}}) \qquad \frac{\Gamma \Rightarrow \circ P[t/x]}{\Gamma \Rightarrow \circ\exists x.P}\ (\circ\exists_{\mathcal{I}})$$

$$\frac{\Gamma, P \supset \circ A \Rightarrow \circ P}{\Gamma, P \supset \circ A \Rightarrow \circ A}\ (\supset \circ_{\mathcal{E}})$$

$$\frac{\Gamma, P \supset A \Rightarrow P}{\Gamma, P \supset A \Rightarrow A}\ (\supset_{\mathcal{E}_1})$$

$$\frac{\Gamma, P \supset A \Rightarrow \circ P}{\Gamma, P \supset A \Rightarrow \circ A}\ (\supset \circ_{\mathcal{E}_2})$$

Fig. 6: Proof search calculus for LLP

For every query, we are interested in all of the proofs of this query, that is, every normal natural deduction of the query. The LLP calculus generates proof terms, but these need to be translated to normal proof terms.

As discussed in the introduction, permutation-free calculi, such as PFLAX, are particularly well suited for the enumeration of all proofs. Also, these calculi give a proof theoretic justification of the form taken by backchaining calculi used for proof search. PFLAX has an advantage over LLP in that it directly generates exactly all the normal natural deduction proofs for any given query. The drawback in using PFLAX is that, even for the fragment of Lax Logic used for constraint logic programming, it does not allow goal-directed proof search. However, despite there being no obvious correspondence between LLP and PFLAX, we consider PFLAX to be a suitable calculus for proof search in the context of constraint logic programming.

8 Deciding Lax Logic

It is useful and interesting to have a decision procedure for any logic. This section describes a decision procedure for propositional Lax Logic. To the best of our knowledge, no decision procedure for propositional Lax Logic has been presented before.

The calculus presented uses a history mechanism to ensure termination of backwards proof search. These history mechanism were introduced in [HSZ96] (see also [Heu98]). The refined history mechanism used here can be found in [How97] (see also [How96], [How98b]). The history mechanism provides a general method for turning a propo-

$$\frac{}{\Gamma \xrightarrow{P} P; \mathcal{H}}\ (ax) \qquad \frac{\Gamma, P \xrightarrow{P} D; \mathcal{H}}{\Gamma, P \Rightarrow D; \mathcal{H}}\ (C)$$

$$\frac{}{\Gamma \Rightarrow \top; \mathcal{H}}\ (\top) \qquad \frac{}{\Gamma \xrightarrow{\bot} D; \mathcal{H}}\ (\bot)$$

$$\frac{\Gamma, P \Rightarrow Q; \{Q\}}{\Gamma \Rightarrow P \supset Q; \mathcal{H}}\ (\supset_{\mathcal{R}1}) \quad \text{if } P \notin \Gamma$$

$$\frac{\Gamma \Rightarrow Q; (Q, \mathcal{H})}{\Gamma \Rightarrow P \supset Q; \mathcal{H}}\ (\supset_{\mathcal{R}2}) \quad \text{if } P \in \Gamma \text{ and } Q \notin \mathcal{H}$$

$$\frac{\Gamma, P \Rightarrow \bot; \{\bot\}}{\Gamma \Rightarrow \neg P; \mathcal{H}}\ (\neg_{\mathcal{R}1}) \quad \text{if } P \notin \Gamma$$

$$\frac{\Gamma \Rightarrow \bot; (\bot, \mathcal{H})}{\Gamma \Rightarrow \neg P; \mathcal{H}}\ (\neg_{\mathcal{R}2}) \quad \text{if } P \in \Gamma \text{ and } \bot \notin \mathcal{H}$$

$$\frac{\Gamma \Rightarrow P; (P, \mathcal{H}) \quad \Gamma \xrightarrow{Q} D; \mathcal{H}}{\Gamma \xrightarrow{P \supset Q} D : \mathcal{H}}\ (\supset_{\mathcal{L}}) \quad \text{if } P \notin \mathcal{H}$$

$$\frac{\Gamma \Rightarrow P; (P, \mathcal{H})}{\Gamma \xrightarrow{\neg P} D; \mathcal{H}}\ (\neg_{\mathcal{L}}) \quad \text{if } D \notin \mathcal{H}$$

$$\frac{\Gamma \Rightarrow P; (P, \mathcal{H}) \quad \Gamma \Rightarrow Q; (Q, \mathcal{H})}{\Gamma \Rightarrow P \wedge Q; \mathcal{H}}\ (\wedge_{\mathcal{R}}) \quad \text{if } P, Q \notin \mathcal{H}$$

$$\frac{\Gamma \xrightarrow{P} D; \mathcal{H}}{\Gamma \xrightarrow{P \wedge Q} D; \mathcal{H}}\ (\wedge_{\mathcal{L}_1}) \qquad \frac{\Gamma \xrightarrow{Q} D; \mathcal{H}}{\Gamma \xrightarrow{P \wedge Q} D; \mathcal{H}}\ (\wedge_{\mathcal{L}_2})$$

$$\frac{\Gamma \Rightarrow P; (P, \mathcal{H})}{\Gamma \Rightarrow P \vee Q; \mathcal{H}}\ (\vee_{\mathcal{R}_1}) \quad \text{if } P \notin \mathcal{H} \qquad \frac{\Gamma \Rightarrow Q; (Q, \mathcal{H})}{\Gamma \Rightarrow P \vee Q; \mathcal{H}}\ (\vee_{\mathcal{R}_2}) \quad \text{if } Q \notin \mathcal{H}$$

$$\frac{\Gamma, P \Rightarrow D; \{D\} \quad \Gamma, Q \Rightarrow D; \{D\}}{\Gamma \xrightarrow{P \vee Q} D; \mathcal{H}}\ (\vee_{\mathcal{L}}) \quad \text{if } P \notin \Gamma \text{ and } Q \notin \Gamma$$

$$\frac{\Gamma \Rightarrow P; (P, \mathcal{H})}{\Gamma \Rightarrow \circ P; \mathcal{H}}\ (\circ_{\mathcal{R}}) \quad \text{if } P \notin \mathcal{H}$$

$$\frac{\Gamma, P \Rightarrow \circ R; \{\circ R\}}{\Gamma \xrightarrow{\circ P} \circ R; \mathcal{H}}\ (\circ_{\mathcal{L}}) \quad \text{if } P \notin \Gamma$$

D is either an atom, $\bot$, disjunction or a modal formula.
Where the history has been extended we have parenthesised $(P, \mathcal{H})$ for emphasis.

Fig. 7: The calculus PFLAX^{Hist} (Scottish)

sitional sequent calculus into a decision procedure. Notice that we use the calculus PFLAX as the base for the history calculus since it gives a more efficient implementation, but we could have instead used a more usual sequent calculus for Lax Logic as the base.

The decision procedure given here uses a history mechanism. Another approach to deciding propositional logics is by the use of 'contraction-free' sequent calculi, such as the one for propositional Intuitionistic Logic given in [Dyc92], [Hud93]. If such a decision procedure for Lax Logic could be found, we would expect it to be faster than one involving a history mechanism. An investigation of contraction-free calculi for Lax Logic can be found in [AF96]. Unfortunately, this investigation did not succeed in finding a contraction-free calculus. We believe that a contraction-free calculus for Lax Logic cannot be found, as (for arbitrary n) examples requiring an entire formula in a sequent to be contracted n times in a proof can be constructed. Consider, as an example, the sequent

$$B \supset (\circ A \supset C) \supset \circ A, \circ B, \circ A \supset C \Rightarrow C$$

where $\circ A \supset C$ needs to duplicated in its entirety in order to prove the sequent.

9 History Mechanisms

Firstly we discuss the general idea of calculi with history mechanisms, then we give the specifics for the history calculus for PFLAX.

9.1 Deciding Propositional Logics Using History Mechanisms

One approach to finding a decision procedure for a propositional logic is to place conditions on the sequent calculus to ensure termination of search. It is elegant to be able to build the content of these conditions into the sequent calculus itself. This is how we develop the calculus for theorem proving in this section. The technique for doing this is quite general and can be applied to many sequent calculi.

In order to ensure termination of backward proof search, we need to check that the same sequent (modulo number of occurrences of formulae of the same type) does not appear again on a branch, that is, proof search does not loop. We need a mechanical way to detect such loops.

One way to do this is to add a *history* to a sequent. The history is the set of all sequents to have occurred so far on a branch of a proof tree. After each backwards inference the new sequent (without its history) is checked to see whether it is a member of this set. If it is we have looping and backtrack. If not the new history is the extension of the old history by the old sequent (without the history component), and we try to prove the new sequent, and so on. Unfortunately, this method is space inefficient as it requires long lists of sequents to be stored by the computer, and all of this list has to be checked at each stage. When the sequents are stored, far more information than necessary is kept. Efficiency would be improved by cutting down the amount of storage and checking needed to prevent looping.

The basis of the reduced history is the realisation (as in [HSZ96]) that one need only store goal formulae in order to loop-check. For the calculi dealt with in this paper, the context cannot decrease; once a formula is in the context it will be in the context of all sequents above it in the proof tree. We say that the calculus has *increasing context*. For two sequents to be the same they need to have the same context (up to multiple occurrences of formulae). Therefore we may empty the history every time the context is (properly) extended. All we need store in the history are goal formulae. If we have a sequent whose goal is already in the history, then we have the same goal and the same context as another sequent, that is, a loop.

There are two slightly different approaches to doing this. There is the straightforward extension of the calculus described in [HSZ96] (which we call the 'Swiss history'; more on this loop-checking method can be found in [Heu98]). There is also related work on histories for Intuitionistic Logic by Gabbay in [Gab91]. The other approach involves storing slightly more formulae in the history, but which for some calculi detects loops more quickly. This we describe as the 'Scottish history' ([How96], [How97]); it can in many cases be more efficient than the Swiss method. In this paper we give a history calculus for Lax Logic using the Scottish history as we believe this to be the better method for intuitionistic logics ([How97]).

One of the great attractions of this approach is its generality. The history mechanism can be attached to a great number of calculi to give decision procedures. A number of applications can be found in [How98b].

9.2 PFLAX^{Hist}

This section gives a history calculus for propositional Lax Logic. It uses the calculus PFLAX as a base to build the calculus, as this calculus has already reduced the search space to a certain extent. PFLAX has the increasing context required for the application of the history mechanism. However, the more usual formulation could have been used instead. PFLAX^{Hist} can be seen in Figure 7.

We give explicit rules for negation (which are just special cases of the rules for implication) for the sake of completeness of connectives. There are two rules for $(\supset_{\mathcal{R}})$. These correspond to the two cases where the new formula, P, is or is not in the context. As noted above, this is very important for history mechanism. Also notice that the number of formulae in the history is at most equal to the length of the formula we check for provability.

A sequent is matched against the conclusions of right rules until the goal formula is either a propositional variable, falsum, or a disjunction (note that disjunction is not covered in [HSZ96], and requires special treatment). This has been ensured by the restriction on goal formulae given in the calculus. A formula from the context is then picked and matched against the left rules of the calculus. The Scottish calculus keeps a complete record of goal formulae between context extensions. At each of the places where the history might be extended, the new goal is checked against the history. If it is in the history, then there is a loop.

There are other places where the rules are restricted to prevent looping. The left rules have side conditions to ensure that the context is increasing. For the $(\supset_{\mathcal{R}})$ rule (which attempts to extend the context) there are two cases corresponding to when the context is and when it is not extended. Something similar is happening in the left rules. Take $(\vee_{\mathcal{L}})$ as an example. In both premisses of the rule a formula may be added to context. If both contexts really are extended, then we can continue building the proof tree. If one or both contexts are not extended then the sequent, S, with the non-extended context, will be the same as some sequent at a lesser height in the proof tree – there is a loop (which we describe as a trivial loop). This is easy to see: since the context and the goal of S are the same as that of the conclusion, the conclusion is the same as the premiss S.

What does a history sequent say? What, in logical terms, is the meaning of a sequent with a history field? Take, for example, the G3^{Hist} sequent $S = \Gamma \Rightarrow R; \mathcal{H}$. This says that for every proof of S, if $P \in \mathcal{H}$, then no sequent of the form $\Gamma \Rightarrow P; \mathcal{H}'$ appears above S in the proof tree of S.

It is now demonstrated that PFLAX^{Hist} is equivalent to PFLAX, in terms of provability. The equivalence is proved via an intermediate calculus PFLAX^{D}. The calculus PFLAX^{D} is the calculus PFLAX where the rule (C) is restricted so that it is only applicable when the goal formula is an atom, a disjunction, falsum or a modal formula.

Proposition 2 *The calculus* PFLAX *is equivalent to the calculus* PFLAX^{D}. *That is, sequent* $\Gamma \Rightarrow G$ *is provable in* PFLAX *iff* $\Gamma \Rightarrow G$ *is provable in* PFLAX^{D}.

The following lemma is needed in the proof of theorem 6.

Lemma 3 (CONTRACTION) *The following rules are admissible in* PFLAX^{Hist}*:*

$$\frac{\Gamma, P, P \Rightarrow R}{\Gamma, P \Rightarrow R}\ (C) \qquad \frac{\Gamma, P, P \xrightarrow{Q} R}{\Gamma, P \xrightarrow{Q} R}\ (C)$$

PROOF: By simultaneous induction on the heights of derivations of premisses. ■

The equivalence proof below, although long, has a simple structure. An algorithm to turn a PFLAX proof tree into a PFLAX^{Hist} proof tree is described in detail. A simple induction argument shows that the algorithm terminates, proving the result.

Theorem 6 *The calculi* PFLAX *and* PFLAX^{Hist} *are equivalent. That is, sequent* $\Gamma \Rightarrow G$ *is provable in* PFLAX *iff sequent* $\Gamma \Rightarrow G; \{G\}$ *is provable in* PFLAX^{Hist}.

PROOF: From Proposition 2 we know that it is enough to show that PFLAX^{D} is equivalent to PFLAX^{Hist}.

It is trivial that any sequent provable in PFLAX^{Hist} is provable in PFLAX^{D}. (Simply drop the history part of the sequent and use contraction above instances of $(\supset_{\mathcal{R}_2})$). We prove the converse.

Take any proof tree for sequent $\Gamma \Rightarrow G$ in PFLAX^{D}. By definition this proof tree is finite. That is, all branches of the tree end with an occurrence of (ax) or $(\bot)$, with all branches having a finite number of nodes (there is also no infinite branching at any node). Using a proof tree for a sequent $\Gamma \Rightarrow G$ in PFLAX^{D} we construct a proof tree for the sequent $\Gamma \Rightarrow G; \{G\}$ in PFLAX^{Hist}. Essentially we take a PFLAX^{D} proof tree and give a recipe for 'snipping out' the loops: removing the sequents that form the loop. Or, looking at it in another way we shall show that failure due to the history mechanism only occurs when there is a loop.

Take any PFLAX^{D} proof tree with $n > 0$ nodes. We take this proof tree and use the following construction to give a PFLAX^{Hist} proof tree.

The following construction takes a PFLAX^{D} proof tree and builds a PFLAX^{Hist} proof tree from the root up. For simplicity we ignore negation, although this can easily be added. In this construction we use 'hybrid trees'. A hybrid tree is a fragment of PFLAX^{Hist} proof tree with all branches that do not have (ax) or $(\bot)$ leaves ending with PFLAX^{D} proof trees. These PFLAX^{D} proof trees have roots which can be obtained by backwards application of a PFLAX^{D} rule to the top history sequent (ignoring its history). We analyse each case of a topmost history sequent with non-history premiss(es) resulting from application of rule (R) in the sequent tree.

- The root of the PFLAX^{D} tree. We change (non-history) sequent $\Gamma \Rightarrow G$ to history sequent $\Gamma \Rightarrow G; \{G\}$.
- (R) is one of (ax), (C), $(\bot)$, $(\top)$, $(\wedge_{\mathcal{L}_1})$, $(\wedge_{\mathcal{L}_2})$, i.e. a rule which in PFLAX^{Hist} has no side conditions. The premiss(es) are changed by adding the appropriate history. They become the history sequents obtained by applying (backwards) the PFLAX^{Hist} rule to the original conclusion.
  For example, if the situation we are analysing is:

$$\frac{\Gamma \xrightarrow{P} D}{\Gamma \xrightarrow{P \wedge Q} D; \mathcal{H}} \ (\wedge_{\mathcal{L}_1})$$

  Then we change this part of the hybrid tree to:

$$\frac{\Gamma \xrightarrow{P} D; \mathcal{H}}{\Gamma \xrightarrow{P \wedge Q} D; \mathcal{H}} \ (\wedge_{\mathcal{L}_1})$$

  We have an extended PFLAX^{Hist} proof tree fragment with PFLAX^{D} proof tree(s) as premiss(es).
- (R) is $(\supset_{\mathcal{R}})$. If the context is extended, simply add the history as appropriate. If the context is not extended, and the new goal is not in the history, again simply extend the history as appropriate. If the new goal is in the history, there is a loop, and here

the history mechanism prevents looping. If the history mechanism condition is not met, then we know that below the conclusion, the hybrid tree has the form:

$$\dfrac{\Gamma \Rightarrow G}{\Gamma \Rightarrow P \supset G; \mathcal{H}} \ (\supset_{\mathcal{R}_2})$$
$$\vdots$$
$$\Gamma \Rightarrow G; \mathcal{H}'$$

where $G \in \mathcal{H}'$ and $\mathcal{H}' \subseteq \mathcal{H}$. The history is not reset at any point in this fragment. This can easily be seen to contain the loop which is the reason for the side conditions not being met. The new hybrid tree is obtained by removing from the previous hybrid tree all the sequents from, but not including, the sequent $\Gamma \Rightarrow G; \mathcal{H}'$ up to and including the sequent $\Gamma \Rightarrow P \supset G; \mathcal{H}$. We can now apply (backwards) the next backwards inference to the first of these sequents. We now know how to proceed.

– (R) is $(\supset_{\mathcal{L}})$. If the side condition is satisfied, then simply add the histories as appropriate. If the side condition is not satisfied, then we know that below the conclusion the hybrid tree has the form:

$$\dfrac{\Gamma \Rightarrow P \quad \Gamma \xrightarrow{Q} R}{\Gamma \xrightarrow{P \supset Q} R; \mathcal{H}} \ (\supset_{\mathcal{L}})$$
$$\vdots$$
$$\Gamma \Rightarrow P; \mathcal{H}'$$

where $P \in \mathcal{H}'$ and $\mathcal{H}' \subseteq \mathcal{H}$. The new hybrid tree is obtained by removing from the previous hybrid tree all the sequents from, but not including, the sequent $\Gamma \Rightarrow P; \mathcal{H}'$ up to and including the sequent $\Gamma \Rightarrow P$. We now know how to proceed.

– (R) is $(\wedge_{\mathcal{R}})$. If the side conditions are satisfied, then simply add the histories as appropriate. Consider the case when the side conditions are not met. Suppose, without loss of generality, that $P \in \mathcal{H}$. We know that below the conclusion the hybrid tree has the form:

$$\dfrac{\Gamma \Rightarrow P \quad \Gamma \Rightarrow Q}{\Gamma \Rightarrow P \wedge Q; \mathcal{H}} \ (\wedge_{\mathcal{R}})$$
$$\vdots$$
$$\Gamma \Rightarrow P; \mathcal{H}'$$

where $\mathcal{H}' \subseteq \mathcal{H}$. The new hybrid tree is obtained by removing from the previous hybrid tree all the sequents from, but not including, $\Gamma \Rightarrow P; \mathcal{H}'$ up to and including $\Gamma \Rightarrow P$. We now know how to proceed.

– (R) is $(\circ_{\mathcal{R}})$. If the side condition is satisfied, then simply add the appropriate history. If the side condition is not satisfies, then we know that below the conclusion the hybrid tree has form:

$$\dfrac{\Gamma \Rightarrow P}{\Gamma \Rightarrow \circ P; \mathcal{H}} \ (\circ_{\mathcal{R}})$$
$$\vdots$$
$$\Gamma \Rightarrow P; \mathcal{H}'$$

where $P \in \mathcal{H}'$ and $\mathcal{H}' \subseteq \mathcal{H}$. The new hybrid tree is obtained by removing from the previous hybrid tree all sequent from, but not including $\Gamma \Rightarrow P; \mathcal{H}$ up to and including $\Gamma \Rightarrow P$. We then know how to proceed.

– (R) is $(\circ_{\mathcal{L}})$. If the side condition is satisfied, then simply add the appropriate history. If the side condition is not satisfied, then we know that below the conclusion the hybrid tree has form:

$$\dfrac{\Gamma, P \Rightarrow \circ R}{\Gamma \xrightarrow{\circ P} \circ R; \mathcal{H}}\ (\circ_{\mathcal{L}})$$
$$\vdots$$
$$\Gamma \Rightarrow \circ R; \mathcal{H}$$

where $P \in \Gamma$. The new hybrid tree is obtained from the old by removing from the old hybrid tree all sequent from, but not including $\Gamma \Rightarrow \circ R; \mathcal{H}$, up to and including $\Gamma \xrightarrow{\circ P} \circ R; \mathcal{H}$. The premiss can be obtained by contraction. We now know how to proceed.

– (R) is $(\vee_{\mathcal{L}})$. If the side conditions are satisfied, then we simply add then appropriate histories. Suppose that one of the side conditions is not satisfied. Without loss of generality we suppose that $P \in \Gamma$. We know that below the conclusion the hybrid tree has the form:

$$\dfrac{\Gamma, P \Rightarrow D \quad \Gamma, Q \Rightarrow D}{\Gamma \xrightarrow{P \vee Q} D; \mathcal{H}}\ (\vee_{\mathcal{L}})$$
$$\vdots$$
$$\Gamma \Rightarrow D; \mathcal{H}$$

The new hybrid tree is obtained from the old by removing from the old hybrid tree all sequents from, but not including $\Gamma \Rightarrow D; \mathcal{H}$, up to and including $\Gamma \xrightarrow{P \vee Q} D; \mathcal{H}$. The entire subtree above above $\Gamma, Q \Rightarrow D$ is also removed. The premiss is obtained by contraction. We now know how to proceed. If $P, Q \in \Gamma$ the we have a choice as to which branch to follow.

– (R) is $(\vee_{\mathcal{R}_1})$. If the side condition is satisfied, then we simply add the appropriate history. If the side condition is not satisfied, then we know that below the conclusion the hybrid tree has form:

$$\dfrac{\Gamma \Rightarrow P}{\Gamma \Rightarrow P \vee Q; \mathcal{H}}\ (\vee_{\mathcal{R}_1})$$
$$\vdots$$
$$\Gamma \Rightarrow P; \mathcal{H}'$$

where $P \in \mathcal{H}'$ and $\mathcal{H}' \subseteq \mathcal{H}$. The new hybrid tree is obtained from the old hybrid tree by removing all sequents from, but not including, $\Gamma \Rightarrow P; \mathcal{H}'$, up to and including $\Gamma \Rightarrow P$. We now know how to proceed.

– (R) is $(\vee_{\mathcal{R}_2})$. Similar to above.

Given that the number of sequents without a history in a hybrid tree is finite and every step described above strictly decreases the number of sequents without a history, this process is terminating. ■

We have shown that PFLAX^{Hist} is sound and complete. For us to prove that it is a decision procedure, we need to prove that it is also terminating, that is, backwards proof search in the calculus ends in success or failure after a finite number of steps. This is proved in the theorem below.

Theorem 7 *Backwards proof search in the calculus* PFLAX^{Hist} *is terminating.*

PROOF: We associate with every sequent a quintuple of natural numbers. With a sequent without a stoup, $\Gamma \Rightarrow R; \mathcal{H}$, we associate:

$$W = (k - n, k - m, 1, 0, r)$$

With a sequent with a stoup, $\Gamma \xrightarrow{P} R; \mathcal{H}$, we associate:

$$W = (k - n, k - m, 0, s, r)$$

Here, k is the number of elements in the *set* of subformulae of (Γ, R); n is the number of elements in the *set* of elements of Γ; m is the number of elements in $\mathcal{H}$; r is the size of goal formula R and s is the size of the stoup formula P. (Notice that although Γ is a multiset, we count its elements as a set). These quintuples are lexicographically ordered from the left.

By inspection we see that for every inference rule W is lower for the premisses than for the conclusion. Consider as an example, $(\supset_{\mathcal{L}})$:

$$\frac{\Gamma \Rightarrow P; (P, \mathcal{H}) \quad \Gamma \xrightarrow{Q} D; \mathcal{H}}{\Gamma \xrightarrow{P \supset Q} D; \mathcal{H}} \; (\supset_{\mathcal{L}}) \text{ if } P \notin \mathcal{H}$$

The conclusion has $W = (k - n, k - m, 0, s_1 + s_2 + 1, r)$. The left premiss has $W' = (k' - n, k' - (m + 1), 1, 0, s_1)$ (where $k' \leq k$). Therefore $W' < W$. The right premiss has $W'' = (k - n, k - m, 0, s_2, r)$. Therefore $W'' < W$. The weights of both premisses are less than the weight of the conclusion.

Hence backward proof search is terminating. ■

When implementing a theorem prover, knowledge of the invertibility of the inference rules can be useful. This information is given in the following proposition.

Proposition 3 *The following inference rules of* PFLAX^{Hist} *are invertible:* $(\supset_{\mathcal{R}1})$, $(\supset_{\mathcal{R}2})$, $(\neg_{\mathcal{R}1})$, $(\neg_{\mathcal{R}2})$, $(\supset_{\mathcal{L}})$, $(\neg_{\mathcal{L}})$, $(\wedge_{\mathcal{R}})$, $(\vee_{\mathcal{L}})$, $(\circ_{\mathcal{L}})$. *The following inference rules of* PFLAX^{Hist} *are not invertible:* (C), $(\wedge_{\mathcal{L}_1})$, $(\wedge_{\mathcal{L}_2})$, $(\vee_{\mathcal{R}_1})$, $(\vee_{\mathcal{R}_2})$, $(\circ_{\mathcal{R}})$.

10 Conclusion and Future Work

This paper has presented two proof search calculi for Lax Logic. The first, PFLAX, is a sequent calculus for first-order quantified Lax Logic. The proofs allowed by this calculus naturally correspond in a 1–1 to the normal natural deductions for first-order quantified Lax Logic. The calculus is well suited for enumerating, without redundancy, all proofs in the logic. This makes the calculus useful in contexts where proof search is for normal natural deductions, such as in (constraint) logic programming.

The second calculus, PFLAX^{Hist} builds on the propositional fragment of the first calculus to give a decision procedure for propositional Lax Logic. A decision procedure for propositional Lax Logic has not been given before. The calculus adds a history mechanism to the propositional calculus to prevent looping. The technique of adding a history mechanism is general and may be applied to a wide range of sequent calculi for propositional logics. Decision procedures are obviously useful in relation to any application that a propositional logic may have. Propositional Lax Logic has been used in hardware verification and PFLAX^{Hist} could be used in this area.

Acknowledgements I would like to thank Roy Dyckhoff for his helpful advice during many useful and interesting discussions. This work has, in part, been supported by EPSRC grant GR/MO8769.

References

[AF96] A. Avellone and M. Ferrari. Almost Duplication-free Tableau Calculi for Propositional Lax Logics. *Springer Lecture Notes in Artificial Intelligence*, 1071:48–64, 1996. Proceedings of TABLEAUX'96.

[BBdP98] P. N. Benton, G. M. Bierman, and V. de Paiva. Computational Types from a Logical Perspective. *Journal of Functional Programming*, 8(2):177–193, 1998.

[Cur52] H. B. Curry. The Elimination Theorem When Modality is Present. *Journal of Symbolic Logic*, 17(4):249–65, 1952.

[DP96] R. Dyckhoff and L. Pinto. A Permutation-free Sequent Calculus for Intuitionistic Logic. Technical Report CS/96/9, University of St Andrews, 1996.

[DP98] R. Dyckhoff and L. Pinto. Cut Elimination and a Permutation-free Sequent Calculus for Intuitionistic Logic. *Studia Logic*, 60:107–118, 1998.

[DP99] R. Dyckhoff and L. Pinto. Permutability of Proofs in Intuitionistic Sequent Calculi. *Theoretical Computer Science*, 212(1-2):141–155, 1999.

[Dyc92] R. Dyckhoff. Contraction-Free Sequent Calculi for Intuitionistic Logic. *Journal of Symbolic Logic*, 57(3):795–807, 1992.

[FM94] M. Fairtlough and M. Mendler. An Intuitionistic Modal Logic with Applications to the Formal Verification of Hardware. In *Computer Science Logic*, pages 354–68. Springer, 1994.

[FM97] M. Fairtlough and M. Mendler. Propositional Lax Logic. *Information and Computation*, 137(1), 1997.

[FMW97] M. Fairtlough, M. Mendler, and M. Walton. First-order Lax Logic as a Framework for Constraint Logic Programming. Technical Report MIPS-9714, University of Passau, 1997.

[FW97] M. Fairtlough and M. Walton. Quantified Lax Logic. Technical Report CS-97-11, University of Sheffield, 1997.

[Gab91] D. Gabbay. Algorithmic Proof with Diminishing Resources, part 1. *Springer Lecture Notes in Computer Science*, 533:156–173, 1991.

[Gen69] G. Gentzen. *The Collected Papers of Gerhard Gentzen*. North-Holland, Amsterdam, 1969. Edited M. E. Szabo.

[Gir91] J.-Y. Girard. A New Constructive Logic: Classical Logic. *Mathematical Structures in Computer Science*, 1:255–296, 1991.

[Her95] H. Herbelin. A λ-calculus Structure Isomorphic to Gentzen-style Sequent Calculus Structure. In L Pacholski and J Tiuryn, editors, *Proceedings of the 1994 workshop Computer Science Logic*, volume 933 of *Springer Lecture Notes in Computer Science*, pages 61–75, 1995.

[Her96] H. Herbelin. A λ-calculus Structure Isomorphic to Sequent Calculus Structure. Unpublished, 1996.

[Heu98] A. Heuerding. *Sequent Calculi for Proof Search in Some Modal Logics*. PhD thesis, Universität Bern, 1998.

[How96] J. M. Howe. Theorem Proving and Partial Proof Search for Intuitionistic Propositional Logic Using a Permutation-free Calculus with Loop Checking. Technical Report CS/96/12, University of St Andews, 1996.

[How97] J. M. Howe. Two Loop Detection Mechanisms: a Comparison. *Springer Lecture Notes in Artificial Intelligence*, 1227:188–200, 1997. Proceedings of TABLEAUX'97.

[How98a] J. M. Howe. A Permutation-free Calculus for Lax Logic. Technical Report CS/98/1, University of St Andrews, 1998.

[How98b] J. M. Howe. *Proof Search Issues in Some Non-Classical Logics*. PhD thesis, University of St Andrews, 1998. Available as Technical Report CS/99/1 and electronically from `http://www-theory.dcs.st-and.ac.uk/~jacob`.

[HSZ96] A. Heuerding, M. Seyfried, and H. Zimmermann. Efficient Loop-Check for Backward Proof Search in Some Non-classical Propositional Logics. *Springer Lecture Notes in Artificial Intelligence*, 933:61–75, 1996. Proceeding of TABLEAUX'96.

[Hud93] J. Hudelmaier. An O(n log n)-space Decision Procedure for Intuitionistic Propositional Logic. *Journal of Logic and Computation*, 3(1):63–75, 1993.

[Men93] M. Mendler. *A Modal Logic for Handling Behavioural Constraints in Formal Hardware Verification*. PhD thesis, University of Edinburgh, 1993. ECS-LFCS-93-255.

[MNPS91] D. Miller, G. Nadathur, F. Pfenning, and A. Scedrov. Uniform Proofs as a Foundation for Logic Programming. *Annals of Pure and Applied Logic*, 51(1-2):125–157, 1991.

[Mog89] E. Moggi. Computational Lambda-Calculus and Monads. In *Proceedings of Logic in Computer Science '89*, pages 14–23, 1989.

[Pra65] D. Prawitz. *Natural Deduction*, volume 3 of *Stockholm Studies in Philosophy*. Almqvist & Wiksell, Stockholm, 1965.

[Wal97] M. Walton. Abstraction and Constraints: Two Sides of the Same Coin. Technical Report CS-97-18, University of Sheffield, 1997.