Towards a Domain Theory for Termination Proofs

Stefan Kahrs*

Abstract

We present a general framework for termination proofs for Higher-Order Rewrite Systems. The method is tailor-made for having simple proofs showing the termination of enriched λ-calculi.

1 Introduction

A semantical method for termination proofs of first-order term rewriting systems has been presented by Zantema at the CTRS'92 workshop [12]. Jaco van de Pol extended this method to higher-order term rewriting systems [8]. However, his extension has certain drawbacks; in particular it is ill-suited to show termination for many enriched λ-calculi.

The reason for many of the technical problems for HRS termination proofs is the meta-level of HRSs. HRS rewrite steps are between $=_{\beta\eta}$-equivalence classes of terms of the simply typed λ-calculus $\lambda^{\rightarrow}$. Any semantic interpretation has to assign the same values to any member of such a class. This means to interpret a function type $\sigma \rightarrow \tau$ as a suitable subset of functions between $[\![\sigma]\!]$ and $[\![\tau]\!]$, making it possible to interpret syntactic application and abstraction as semantic application and abstraction, respectively. The trouble is: for $\lambda^{\rightarrow}$ these objectives are virtually incompatible. The most suitable subset restriction for functions is the restriction to strictly monotonic functions. This would imply that context application preserves termination. However, variable abstraction can introduce non-monotonic functions: constant functions like $\lambda x.c$ are not monotonic.

Robin Gandy approached these problems [3] by interpreting a term $\lambda x.c$ not as the constant function $(x \mapsto c)$, but instead as a function $(x \mapsto c + L(x))$, where L is some monotonic type conversion function, and where $+$ was some appropriate addition on the result type. Gandy does not give criteria how to obtain these $+$ and L operations in general; however, they naturally arise from a categorical semantics as we shall see later.

Gandy's paper is only of limited use when one tries to generalise the approach and apply it to other systems. This is not too surprising as Higher-Order Rewrite Systems — which provide a *syntactical* framework for expressing enriched λ-calculi — are a rather recent invention. Our aim is to devise a *semantical* framework for termination proofs and to link it with HRSs.

2 Preliminaries

An *Abstract Reduction System* (short: ARS) consists of a set A and a binary relation $\rightarrow$ on A. We write $\boldsymbol{A} \models P$ if the ARS $\boldsymbol{A} = (A, \rightarrow)$ has the property P. Given an ARS $\boldsymbol{A} = (A, \rightarrow_A)$, $t \in A$ is a *normal form* if $\neg\exists u \in A.\ t \rightarrow_A u$. An ARS $\boldsymbol{A} = (A, \rightarrow)$ is *strongly normalising*, $\boldsymbol{A} \models \mathrm{SN}$, iff there is no non-empty relation R on A satisfying the equation $R = \ \rightarrow; R$, i.e. iff there are no infinite chains of $\rightarrow$-steps.

We do not have room to introduce the concepts we borrow from category theory to make this paper self-contained. The reader can find the missing definitions in most standard works on category theory, e.g. in [6]. Here is a brief summary defining some of these terms.

*Laboratory for Foundations of Computer Science, University of Edinburgh, Edinburgh, Scotland; email: `smk@dcs.ed.ac.uk`; Tel.: (44)-31-650-5139; Fax: (44)-31-667-7209.

Given a category $\mathcal{A}$ and an object $X \in |\mathcal{A}|$, the *comma* category $\mathcal{A} \downarrow X$ has as objects pairs (Y, f), such that $Y \in |\mathcal{A}|$ and $f \in \mathcal{A}(Y, X)$, and a morphism $m \in (\mathcal{A} \downarrow X)((Y, g), (Z, h))$ is a morphism $m \in \mathcal{A}(Y, Z)$ such that $h \circ m = g$. There is a forgetful functor $U : \mathcal{A} \downarrow X \to \mathcal{A}$ defined as $U(X, f) = X$ and $U(m) = m$. Dually, $X \downarrow \mathcal{A}$ is defined as $(\mathcal{A}^{op} \downarrow X)^{op}$.

A *monad* in a catecory $\mathcal{A}$ is a triple $(\boldsymbol{T}, \mu, \eta)$ where $\boldsymbol{T} : \mathcal{A} \to \mathcal{A}$ is a functor and $\mu : \boldsymbol{T}(\boldsymbol{T}(\_)) \to \boldsymbol{T}(\_)$ and $\eta : \_ \to \boldsymbol{T}(\_)$ are natural transformations satisfying $\mu \circ \eta = id$ and $\mu \circ \mu = \boldsymbol{T}(\mu) \circ \mu$. Given such a monad, the category $\mathcal{A}^{\boldsymbol{T}}$ of $\boldsymbol{T}$-algebras has as objects pairs (X, f), such that $X \in |\mathcal{A}|$, $f \in \mathcal{A}(\boldsymbol{T}(X), X)$, $f \circ \mu = f \circ \boldsymbol{T}(f)$ and $f \circ \eta = id$; a morphism $g \in \mathcal{A}^{\boldsymbol{T}}((X, f), (X', f'))$ is a morphism $g \in \mathcal{A}(X, X')$ such that $g \circ f = f' \circ \boldsymbol{T}(f)$. There is a forgetful functor $U : \mathcal{A}^{\boldsymbol{T}} \to \mathcal{A}$ defined as $U(X, f) = X$ and $U(g) = g$.

Let $\mathcal{A}$ and $\mathcal{B}$ be categories and $U : \mathcal{A} \to \mathcal{B}$ be a functor. A functor $F : \mathcal{B} \to \mathcal{B}$ *can be lifted along* U if there is a functor $F' : \mathcal{A} \to \mathcal{A}$ such that $F \circ U = U \circ F'$. If $\mathcal{A} = \mathcal{B}^{\boldsymbol{T}}$ we omit the "along U" and understand U to be the forgetful functor U of the monad. Similarly for $\mathcal{A} = \mathcal{B} \downarrow X$.

A *monoidal category* is a tuple $(\mathcal{A}, \otimes, I, a, l, r)$ where $\mathcal{A}$ is a category, $\otimes$ is a functor $\mathcal{A} \times \mathcal{A} \to \mathcal{A}$, I is an object in $\mathcal{A}$, and a, l, and r are natural isomorphisms $a : A \otimes (B \otimes C) \cong (A \otimes B) \otimes C$, $l : I \otimes A \cong A$, $r : A \otimes I \cong A$ (natural in A, B and C) such that "all coherence diagrams" commute, i.e. all diagrams only involving the isomorphisms and $\otimes$. A monoidal category is *symmetric* if there is also a natural isomorphism $s : A \otimes B \cong B \otimes A$ maintaining the property that all coherence diagrams commute. A (symmetric) monoidal category is called *closed* if the functor $\_ \otimes A$ has a right adjoint $A \Rightarrow \_$ for any $A \in |\mathcal{A}|$. We write $ap : (A \Rightarrow \_) \otimes A \to \_$ for the co-unit of the adjunction and $cur(f)$ as shorthand for $(id \Rightarrow f) \circ \eta$ where $\eta : \_ \to A \Rightarrow (\_ \otimes A)$ is the unit of the adjunction.

A *monoid* in a monoidal category $(\mathcal{A}, \otimes, I, a, l, r)$ is a triple $(X, \oplus, 0)$, where $X \in |\mathcal{A}|$, $\oplus \in \mathcal{A}(X \otimes X, X)$, and $0 \in \mathcal{A}(I, X)$, such that $\oplus \circ (0 \otimes id) \circ l^{-1} = \oplus \circ (id \otimes 0) \circ r^{-1} = id$ and $\oplus \circ (\oplus \otimes id) = \oplus \circ (id \otimes \oplus) \circ a^{-1}$. The *representation monad* of a monoid is the monad $(X \otimes \_, \mu, \eta)$, where $\mu = (\oplus \otimes id) \circ a^{-1}$ and $\eta = (0 \otimes id) \circ l^{-1}$.

3 Semantics

For proving termination in a semantical setting, we want to interpret types by (partially) well-ordered sets. To make this general enough to cater for typed rewrite systems with non-elementary types like products, coproducts, or function types, we have to develop something like a domain theory for well-orderings.

Definition 1. A *partially well-ordered set* is an ARS $\boldsymbol{A} = (A, >_A)$ such that $\boldsymbol{A} \models \mathrm{SN}$ and that $>_A$ is transitive. Convention: we shall write $\geq_A$ for the reflexive closure of $>_A$ on A. If $\geq_A$ is a total order, we call $\boldsymbol{A}$ a *well-ordered set*.

3.1 The category of well-ordered sets

Definition 2. The category WO is defined as follows:

Objects: $\boldsymbol{A} = (A, >_A) \in |\mathsf{WO}|$ if $\boldsymbol{A}$ is a partially well-ordered set.

Morphisms: a morphism $f : \boldsymbol{A} \to \boldsymbol{B}$ is a function $f : A \to B$ satisfying
$\forall a, a' \in A.\ a >_A a' \Rightarrow f(a) >_B f(b)$.

Composition and identities: as in Set, the category of sets.

We write TWO for the full subcategory of WO that only contains well-ordered sets.

Suppose we have an enriched λ-calculus with non-elementary types such as $\sigma \to \tau$, $\sigma \times \tau$, etc. To find well-ordered structures (objects in WO) for these composite types means to find the corresponding endofunctors on WO. Notice that WO has no terminal object. With a terminal object one can express constant functions (those that factor through the terminal object), but constant functions do not preserve strict orders such as any (non-empty) well-order $>$ in WO.

Definition 3. We define two summation functors on WO, categorical sum $\mathsf{WO} \sqcup \mathsf{WO} \to \mathsf{WO}$ and ordinal sum $\mathsf{WO} + \mathsf{WO} \to \mathsf{WO}$ as follows: let $\boldsymbol{A} = (A, >_A)$ and $\boldsymbol{B} = (B, >_B)$, then

- $\boldsymbol{A} \sqcup \boldsymbol{B} = (A \times \{0\} \cup B \times \{1\}, >_{A \sqcup B})$ such that $(a, 0) >_{A \sqcup B} (a', 0) \iff a >_A a'$ and $(b, 1) >_{A \sqcup B} (b', 1) \iff b >_B b'$. On morphisms, $f \sqcup g$ is $f + g$ from Set.
- $\boldsymbol{A} + \boldsymbol{B}$ is defined just as $\boldsymbol{A} \sqcup \boldsymbol{B}$, except for the order:
$(x, n) >_{A+B} (x', n') \iff (x, n) >_{A \sqcup B} (x', n') \ \vee \ n > n'$

The notation $\boldsymbol{A} \sqcup \boldsymbol{B}$ for the coproduct is motivated by the observation that the order type of $\alpha \sqcup \beta$ (for ordinals α and β) is just the ordinal $\alpha \cup \beta$, i.e. the maximum of the two. It is routine to check that the categorical sum is indeed a coproduct in the sense of category theory.

The ordinal sum is associative (modulo isomorphisms) but not commutative.

Definition 4. We define three multiplication functors, categorical product $\mathsf{WO} \sqcap \mathsf{WO} \to \mathsf{WO}$, ordinal product $\mathsf{WO} * \mathsf{WO} \to \mathsf{WO}$, and symmetric product $\mathsf{WO} \otimes \mathsf{WO} \to \mathsf{WO}$ as follows: let $\boldsymbol{A} = (A, >_A)$ and $\boldsymbol{B} = (B, >_B)$, then

- $\boldsymbol{A} \sqcap \boldsymbol{B} = (A \times B, >_{A \sqcap B})$ such that $(a, b) >_{A \sqcap B} (a', b') \iff a >_A a' \ \wedge \ b >_B b'$. On morphisms, $f \sqcap g$ is $f \times g$ from Set.
- $\boldsymbol{A} * \boldsymbol{B}$ is defined just as $\boldsymbol{A} \sqcap \boldsymbol{B}$, except for the order:
$(a, b) >_{A*B} (a', b') \iff (a, b) >_{A \sqcap B} (a', b') \ \vee \ a >_A a'$
- Similarly, $\boldsymbol{A} \otimes \boldsymbol{B}$ is defined just as $\boldsymbol{A} \sqcap \boldsymbol{B}$, except for the order $>_{A \otimes B}$:
$(a, b) >_{A \otimes B} (a', b') \iff a >_A a' \ \wedge \ b \geq_B b' \ \vee \ a \geq_A a' \ \wedge \ b >_B b'$.

On first view, one might think that the multiplication functors differ only in minor details, but these details are quite significant. Writing bold numbers for the objects corresponding to ordinals in WO we have for example: $\mathbf{3} \sqcap \mathbf{7} \approx \mathbf{3}$, $\mathbf{3} * \mathbf{7} \approx \mathbf{21}$, and $\mathbf{3} \otimes \mathbf{7} \approx \mathbf{9}$. Here, $\boldsymbol{A} \approx \boldsymbol{\alpha}$ is used to mean that there are morphisms $\boldsymbol{\alpha} \xrightarrow{f} \boldsymbol{A} \xrightarrow{g} \boldsymbol{\alpha}$. So we characterise an object $\boldsymbol{A} \in |\mathsf{WO}|$ by the longest chains it includes.

Proposition 5. *$(\mathsf{WO}, \otimes, \mathbf{1})$ is a symmetric monoidal category, where $\mathbf{1} = (\{0\}, \emptyset)$.*

We also have such results for the categorical product and for the ordinal product, but each with one restriction. The categorical product is associative and commutative (modulo isomorphisms), but it lacks a neutral element, essentially because of the Burali-Forti paradox. The ordinal product is associative and has a neutral element, but it cannot be symmetric as there is no isomorphism (monotonic bijection) between $\boldsymbol{\omega} * \mathbf{2}$ and $\mathbf{2} * \boldsymbol{\omega}$.

Proposition 6. *The functor $\_ \otimes A$ has a right adjoint $A \Rightarrow \_$, i.e. WO is monoidal closed.*

Proof. We define the functor $\_ \Rightarrow \_ : \mathsf{WO}^{op} \times \mathsf{WO} \to \mathsf{WO}$ by
$(A, >_A) \Rightarrow (B, >_B) = (A \Rightarrow B, >_{A \Rightarrow B})$ with:

$$\begin{array}{lcl} A \Rightarrow B & = & \{f \in A \to B \mid \forall x, y \in A.\ (x >_A y \Rightarrow f(x) >_B f(y))\} \\ f >_{A \Rightarrow B} g & \iff & \forall x \in A.\ f(x) >_B g(x) \end{array}$$

We have to slightly amend this definition in case A is the empty set: $>_{\emptyset \Rightarrow B} = \emptyset$, the empty relation is SN. If A is non-empty, it has an element $a \in A$, and each infinite chain $f_1 >_{A \Rightarrow B} f_2 >_{A \Rightarrow B} \dots$ can be mapped to an infinite chain $f_1(a) >_B f_2(a) >_B \dots$ in B. On morphisms, we have as usual $(f \Rightarrow g)(h) = g \circ h \circ f$. Checking the adjunction properties is routine. $\square$

Apart from (ordinal) addition and multiplication, we can also define a corresponding (ordinal) exponentiation. This is only defined for totally well-ordered sets.

Definition 7. We define a functor $\mathcal{F}$: TWO $\to$ TWO as follows: let $\boldsymbol{A} = (A, >_A)$, $\boldsymbol{B} = (B, >_B)$, $\boldsymbol{A} \xrightarrow{f} \boldsymbol{B}$ then

- $\mathcal{F}(\boldsymbol{A}) = (\mathcal{F}(A), \succ_A)$ where $\mathcal{F}(A)$ is the set of finite subsets of A, and $M \succ_A N \iff \exists m \in (M \setminus N).\ \forall n \in (N \setminus M).\ m >_A n$.
- $\mathcal{F}(f)(M) = \{f(m) \mid m \in M\}$.

The restriction to TWO is necessary to make $\mathcal{F}$ functorial. All morphisms $f : \boldsymbol{A} \to \boldsymbol{B}$ in TWO are monic, in particular they guarantee that the witness $m \in (M \setminus N)$ for $M \succ_A N$ is mapped to a witness $f(m) \in (f(M) \setminus f(N)) = f(M \setminus N)$ for $f(M) \succ_B f(N)$, which is necessary to make $\mathcal{F}(f)$ monotonic.

$\mathcal{F}(A)$ contains only finite subsets of A, but still the operator $\mathcal{F}$ corresponds in ordinal arithmetic to exponentiation to the power of 2. A general ordinal exponentiation can also be defined:

Definition 8. The functor $\boldsymbol{Fin}$: TWO $\times$ TWO $\to$ TWO is defined as follows. Let $\boldsymbol{A} = (A, >_A)$, $\boldsymbol{B} = (B, >_B)$, $\boldsymbol{C} = (C, >_C)$ and $\boldsymbol{D} = (D, >_D)$, and let $\boldsymbol{A} \xrightarrow{f} \boldsymbol{B}$ and $\boldsymbol{C} \xrightarrow{g} \boldsymbol{D}$. Let $\bot_B$ be the smallest element of $\boldsymbol{B}$ (if it exists).

- $\boldsymbol{Fin}(\boldsymbol{A}, \boldsymbol{B}) = (A \rightsquigarrow B, \succ_{A*B})$ where
  $A \rightsquigarrow B = \{M \in \mathcal{F}(A * B) \mid (a, b) \in M \wedge (a, b') \in M \implies b = b' \ \wedge\ b >_B \bot_B\}$.
- $\boldsymbol{Fin}(f, g) = \mathcal{F}(f * g)$

Notice that for $(B, >_B) = \boldsymbol{B} \in |\mathsf{TWO}|$ a smallest element $\bot_B \in B$ always exists, with the only exception $\boldsymbol{B} = \boldsymbol{0} = (\emptyset, \emptyset)$ for which we have $\boldsymbol{Fin}(\boldsymbol{A}, \boldsymbol{0}) = \boldsymbol{1} = (\{\emptyset\}, \emptyset)$. Because $\boldsymbol{Fin}$ operates in TWO, any morphisms f and g are injective and hence $f * g$ sends finite maps to finite maps. Similarly, f and g cannot map non-$\bot$ elements to $\bot$, because they are monotonic. That $\mathcal{F}(f * g)$ is monotonic follows easily from the fact that we inherit the order from $\mathcal{F}(\boldsymbol{A} * \boldsymbol{B})$.

3.2 Ordinals

The functors described so far allow to construct new (partially) well-ordered sets from given ones. For termination proofs, it is useful to have an arithmetic and logic for well-orderings available. The obvious choice is ordinal arithmetic.

Definition 9. An *ordinal* is a set α such that all its elements are ordinals and

- $\forall \beta \in \alpha.\ \forall \gamma \in \alpha.\ \beta \in \gamma \vee \gamma \in \beta \vee \beta = \gamma$
- $\forall \beta \in \alpha.\ \forall \gamma \in \beta.\ \gamma \in \alpha$

These are the so-called "von Neumann ordinals" [7]. In the following, I identify 0, 1, 2, etc. with their corresponding ordinal, ω is used for the ordinal corresponding to the set of natural numbers.

The functor I called "ordinal addition" is not quite ordinal addition in the usual sense, because we would need to postcompose it with a functor TYPE that maps partially well-ordered sets to their order type (the corresponding ordinal). We can do that by the following principle:

Definition 10. Given an endofunctor F : TWO $\to$ TWO, we define a corresponding function $\lfloor F \rfloor$ on the class of ordinals as follows: for any ordinals α and β, $\lfloor F \rfloor(\alpha)$ is the unique ordinal γ such that there is an isomorphism $F(\boldsymbol{\alpha}) \cong \boldsymbol{\gamma}$.

If no confusion arises, I shall write $F(\alpha)$ instead of $\lfloor F \rfloor(\alpha)$, i.e. if it is clear from the context that α and/or β are ordinals and not objects in TWO. A more traditional notation for $\lfloor \boldsymbol{Fin} \rfloor(\gamma, \beta)$ is $\beta^{(\gamma)}$, i.e. ordinal exponentiation.

There is an intuitive understanding of $\boldsymbol{Fin}(\gamma,\beta)$. Each element in the carrier set can be seen as the (unique) representation of an ordinal to base β, such that the domain of the finite map gives the exponents, and the range the coefficients. This also explains the exclusion of bottom ($= 0$) coefficients, and the use of lexicographic ordering, because "exponents matter more than coefficients".

Definition 11. Given an ordinal α, we define a map $\_\#\_ : \boldsymbol{Fin}(\alpha,\omega) \times \boldsymbol{Fin}(\alpha,\omega) \to \boldsymbol{Fin}(\alpha,\omega)$ as follows:

$$\begin{array}{rl} M\#N = & \{(a, m+n) \mid (a,m) \in M \wedge (a,n) \in N\} \cup \\ & \{(a,m) \in M \mid \neg\exists n.(a,n) \in N\} \cup \{(a,n) \in N \mid \neg\exists m.(a,m) \in M\} \end{array}$$

Intuitively, $\#$ does the following: it takes the representation of two ordinals to the base ω (the so-called "Cantorian normal form") and then adds the coefficients pointwise. Since addition is closed on natural numbers (and the coefficients happen to be natural numbers here), the result $M\#N$ is still in $\boldsymbol{Fin}(\alpha,\omega)$, i.e. we do not have to worry about carries. It is easy to see that $\oplus$ is associative, commutative, and has a neutral element (the empty map). This form of addition (the function $\lfloor\#\rfloor$ on ordinals) is the so-called "natural sum" of Hessenberg, see [4, 5].

Proposition 12. *For any ordinal α, $\#$ is a morphism in* TWO*:*

$$\boldsymbol{Fin}(\alpha,\omega) \otimes \boldsymbol{Fin}(\alpha,\omega) \xrightarrow{\#} \boldsymbol{Fin}(\alpha,\omega).$$

Proof. Since $\#$ is commutative, we only have to show: $M \succ_{\alpha*\omega} M' \Rightarrow M\#N \succ_{\alpha*\omega} M'\#N$. There has to be a largest $a \in \alpha$ such that there is an element $p = (a,m) \in M$ but $p \notin N$. This a is also the largest element at which $M\#N$ and $M'\#N$ differ. $M\#N$ contains the pair $(a, m+n_a)$ and $M'\#N$ contains either no pair $(a, \_)$ or a pair $(a, m'+n_a)$ with $m \ni m'$, $\omega \ni n_a \supseteq 0$ and thus $m + n_a \ni m' + n_a$. Because the order in $\succ_{\alpha*\omega}$ is lexicographic, (a,m) is greater than all (a',k) for $a \ni a'$ and is therefore larger than all elements in $(M'\#N) \setminus (M\#N)$. □

Definition 13. Let α be an ordinal. It is called a *limit ordinal* iff $\forall\beta \in \alpha. \beta{+}1 \in \alpha$. It is is called *indecomposable* iff $\forall\beta \in \alpha.\ \beta{+}\alpha = \alpha$. It is called an *epsilon ordinal* iff $\forall\beta \in \alpha.\ \beta \ni 1 \Rightarrow \beta^{(\alpha)} = \alpha$.

The relevance of indecomposable[1] ordinals to this paper is that they can be understood as algebras for ordinal addition (and also natural sum), i.e. their elements are closed under ordinal addition. Indecomposable ordinals (greater than 0) are exactly those of the form $\omega^{(\gamma)}$ for some ordinal γ. This property is very closely related to the mentioned properties of $\#$. Indecomposability of α is sufficient *and* necessary for the existence of a morphism from $\alpha \otimes \alpha$ to α, for the latter case see lemma 15 in [2].

Proposition 14. *Let α be an indecomposable ordinal. Then $(\boldsymbol{\alpha}, \#, 0)$ is a monoid in* WO*, where $\#$ is natural sum and $0 : \mathbf{1} \to \boldsymbol{\alpha}$ is the function that maps the element of 1 to 0.*

3.3 Type-casting Ordinals

For concrete termination proofs it is useful to have "type-conversion functions" that can translate values of any type into ordinals and back. In particular, if we have erasing rewrite rules, i.e. rules in which some variables only occur on the left-hand side, then the presence of type-conversion functions in the semantics makes it easier to give a semantic interpretation for the symbols. This will become quite clear in our application example.

Definition 15. Let $\mathcal{A}$ be a category and $X \in |\mathcal{A}|$. We define the category $\mathcal{A} \downdownarrows X$ as follows: $\mathcal{A} \downdownarrows X$ is the comma-category $X \downarrow \mathcal{A} \downarrow X$. Thus, an object in $\mathcal{A} \downdownarrows X$ is an object Z in $\mathcal{A}$, accompanied by two distinguished morphisms $X \xrightarrow{z} Z$ (encoding) and $Z \xrightarrow{\bar{z}} X$ (decoding), such that $\bar{z} \circ z = id_X$. A morphism from $X \xrightarrow{z} Z \xrightarrow{\bar{z}} X$ to $X \xrightarrow{y} Y \xrightarrow{\bar{y}} X$ in $\mathcal{A} \downdownarrows X$ is a morphism $f \in \mathcal{A}(Z,Y)$ such that $f \circ z = y$ and $\bar{y} \circ f = \bar{z}$.

[1] The name is taken from [9]; in [2] they are called "additive principal"; Hessenberg [4] called them "Hauptzahlen", main numbers.

$$
\begin{tikzcd}
X \arrow[r, "r^{-1}"] \arrow[d, "cur(\oplus)"'] \arrow[dr, phantom, "(i)"] & X \otimes I \arrow[r, "id \otimes 0"] \arrow[d, "cur(\oplus) \otimes id"'] \arrow[dr, phantom, "(ii)"] & X \otimes X \arrow[r, "\oplus"] \arrow[d, "cur(\oplus) \otimes id"'] \arrow[dr, phantom, "(iii)"] & X \\
X \Rightarrow X \arrow[r, "r^{-1}"'] & (X \Rightarrow X) \otimes I \arrow[r, "id \otimes 0"'] & (X \Rightarrow X) \otimes X \arrow[r, "ap"'] & X \arrow[u, "id"']
\end{tikzcd}
$$

Figure 1: Theorem 19, case 3

To be precise, the category $X \downarrow \mathcal{A} \downarrow X$ should be written as either $(X \downarrow \mathcal{A}) \downarrow (X \xrightarrow{id} X)$ or as $(X \xrightarrow{id} X) \downarrow (\mathcal{A} \downarrow X)$, but both constructions result in identical categories.

In particular, we are interested in categories like $\mathsf{WO} \downdownarrows \boldsymbol{\alpha}$ where α is an ordinal. In this case, the maps z and $\bar{z}$ convert ordinals less than α into elements of Z and back; the required equation $\bar{z} \circ z = id$ is another way of saying that decoding is the inverse of encoding. It is not required that the converse is true; in particular, an object $\boldsymbol{Z}$ may not be totally ordered although $\boldsymbol{\alpha}$ is.

Proposition 16. *By an abuse of notation, we write X to denote the object $X \xrightarrow{id} X \xrightarrow{id} X$ in $\mathcal{A} \downdownarrows X$. We have:*

1. *X is a null object in $\mathcal{A} \downdownarrows X$, i.e. it is initial and terminal.*
2. *For any two objects $A, B \in |\mathcal{A} \downdownarrows X|$ there is a morphism $A \xrightarrow{0} B$ uniquely defined by $0 = A \to X \to B$.*

Definition 17. We call an endofunctor $F : \mathcal{A} \to \mathcal{A}$ *retractable at* $X \in |\mathcal{A}|$ iff there is an object $X \xrightarrow{F} F(X) \xrightarrow{\bar{F}} X$ in $\mathcal{A} \downdownarrows X$.

Proposition 18. *If F is retractable at X then F can be lifted to $\mathcal{A} \downdownarrows X$. $F' : \mathcal{A} \downdownarrows X \to \mathcal{A} \downdownarrows X$ maps each object $X \xrightarrow{z} Z \xrightarrow{\bar{z}} X$ to $X \xrightarrow{F} F(X) \xrightarrow{F(z)} F(Z) \xrightarrow{F(\bar{z})} F(X) \xrightarrow{\bar{F}} X$. On morphisms, we just define $F'(f) = F(f)$.*

We can extend the notion of retractablity to bifunctors in an obvious way.

Theorem 19. *Let $\mathcal{A}$ be a monoidal closed category with binary products and coproducts.*

1. *Categorical product $\sqcap$ and coproduct $\sqcup$ are retractable at any X:* $X \xrightarrow{\langle id, id \rangle} X \sqcap X \xrightarrow{\pi_1} \boldsymbol{\alpha}$ *is a retraction for the product; the coproduct is dual.*
2. *If $(X, \oplus, 0)$ is a monoid in $\mathcal{A}$ then the tensor product $\otimes$ is retractable at X.* $X \cong I \otimes X \xrightarrow{0 \otimes id} X \otimes X \xrightarrow{\oplus} X$ *is a retraction by the coherence properties of a monoid.*
3. *The exponential $\Rightarrow$ (right adjoint of $\otimes$) is retractable at monoids:* $X \xrightarrow{cur(\oplus)} X \Rightarrow X \cong (X \Rightarrow X) \otimes I \xrightarrow{id \otimes 0} X \Rightarrow X \otimes X \xrightarrow{ap} X$ *is a retraction; ap is the co-unit of the adjunction and $cur(\oplus)$ is the curried form of $\oplus : X \otimes X \to X$, also given by the adjunction.*

Proof. Cases 1 and 2 are immediate. For case 3, see figure 1: (i) commutes by naturality of r^{-1}, one of the natural isomorphisms of the monoidal structure of $\mathcal{A}$, (ii) commutes by functorality of $\otimes$, and (iii) is the co-unit equation for ap. The upper line, $\oplus \circ (id \otimes 0) \circ r^{-1}$, is the identity because $(X, \oplus, 0)$ is a monoid. $\square$

Taking $\mathcal{A} = \mathsf{WO}$ it follows that $\otimes$ and $\Rightarrow$ are retractable at indecomposable ordinals, $\oplus$ being the natural sum. Ordinal addition, multiplication and exponentiation are typically not retractable, except in trivial cases.

3.4 α-addition everywhere

Type conversions are in general not quite good enough to deal with erasing rewrite rules: they allows us to take an element a of type σ and an element b of type τ, map both to some ordinal, add them and map them to any type we wish. But we do *not* get the following: $\sigma(\bar{\sigma}(a)\#\bar{\tau}(b)) \geq_\sigma a$. This is needed to handle erasing rules which are also collapsing. The solution to this problem is to find some addition operation that directly operates on σ.

Consider the representation monad $\boldsymbol{M} = (\boldsymbol{\alpha} \otimes \_, \mu, \eta)$ of the monoid $(\boldsymbol{\alpha}, \#, 0)$. $(\boldsymbol{A}, \boxplus)$ being an $\boldsymbol{M}$-algebra means in particular that (i) $\boxplus \circ \eta = id$, which is the same as saying that $0 \boxplus x = x$, and (ii) $\boxplus \circ \mu = \boxplus \circ (id \otimes \boxplus)$, which is the same as $(m\#n) \boxplus x = m \boxplus (n \boxplus x)$. Especially the unit-property is useful: in $\boldsymbol{\alpha} \otimes \boldsymbol{A}$ there are α-chains if $\boldsymbol{A}$ is non-empty, e.g. $(0, x) < (1, x) < \ldots$. The presence of a monotonic map $\boxplus$ means that there are also α-chains in $\boldsymbol{A}$; because of the retraction property $0 \boxplus x = x$ we have α-chains starting from any element x of A: $x = 0 \boxplus x < 1 \boxplus x < \ldots$; this mirrors the behaviour of rewrite systems with collapsing rules.

Proposition 20. *Let $\mathcal{A}$ be a monoidal category and $(X, \oplus, 0)$ be a monoid in $\mathcal{A}$ and let $\boldsymbol{T}$ be the representation monad of this monoid. Then $(X, \oplus) \in |\mathcal{A}^{\boldsymbol{T}}|$.*

Although the observation in proposition 20 is rather trivial, it is important for the whole method. We can interpret "atomic" types by $\boldsymbol{\alpha}$ and leave the interpretation of composite types to functors on $\mathsf{WO}^{\boldsymbol{M}}$, supporting collapsing rules of composite types.

To maintain the existence of a $\boxplus$-operation with nice algebraic properties we have to make the functors we are interested in operate on $\mathsf{WO}^{\boldsymbol{M}}$.

Definition 21. Let $\mathcal{A}$ be a category and $\boldsymbol{T} = (T, \mu, \eta)$ a monad on $\mathcal{A}$. A functor $F : \mathcal{A} \to \mathcal{A}$ is called *$\boldsymbol{T}$-distributive* if there is a natural transformation $\delta : T(F(U(\_))) \to F(T(U(\_)))$ such that $F(\boxplus_A) \circ \delta_A \circ \eta = id$ and $F(\boxplus_A) \circ \delta_A \circ \mu = F(\boxplus_A) \circ \delta_A \circ T(F(\boxplus_A) \circ \delta_A)$. Here, $U : \mathcal{A}^{\boldsymbol{T}} \to \mathcal{A}$ is the forgetful functor of the monad and $\boxplus_A : T(A) \to A$ is the algebra morphism on A.

For $\boldsymbol{T}$-distributive functors, we get a new $\boxplus'$ operation on $F(A)$ as $\boxplus' = F(\boxplus_A) \circ \delta$.

Proposition 22. *If F is $\boldsymbol{T}$-distributive then it can be lifted to $\mathcal{A}^{\boldsymbol{T}}$.*

We can now check whether the retractable functors we have so far, i.e. product, coproduct, symmetric multiplication, and arrow, are $\boldsymbol{M}$-distributive or not.

Lemma 23. *Let $\mathcal{A}$ be a category with binary products $\_ \sqcap \_$. Let $\boldsymbol{T} = (T, \mu, \eta)$ be a monad on $\mathcal{A}$. Then $\_ \sqcap X$ is $\boldsymbol{T}$-distributive if $(X, \boxplus) \in |\mathcal{A}^{\boldsymbol{T}}|$.*

Proof. We can define δ as $\langle T(\pi_1), \boxplus \circ T(\pi_2) \rangle$. The new addition is $\boxplus' = \langle \boxplus \circ T\pi_1, \boxplus \circ T\pi_2 \rangle$.

Unit property: $\boxplus' \circ \eta = \langle \boxplus \circ T\pi_1, \boxplus \circ T\pi_2 \rangle \circ \eta = \langle \boxplus \circ T\pi_1 \circ \eta, \boxplus \circ T\pi_2 \circ \eta \rangle = \langle \boxplus \circ \eta \circ \pi_1, \boxplus \circ \eta \circ \pi_2 \rangle = \langle \pi_1, \pi_2 \rangle = id$. Associativity: $\boxplus' \circ T\boxplus' = \langle \boxplus \circ T\pi_1, \boxplus \circ T\pi_2 \rangle \circ T\boxplus' = \langle \boxplus \circ T\pi_1 \circ T\boxplus', \boxplus \circ T\pi_2 \circ T\boxplus' \rangle = \langle \boxplus \circ T(\pi_1 \circ \boxplus'), \boxplus \circ T(\pi_2 \circ \boxplus') \rangle = \langle \boxplus \circ T(\boxplus \circ T\pi_1), \boxplus \circ T(\boxplus \circ T\pi_2) \rangle = \langle \boxplus \circ T(\boxplus) \circ TT\pi_1, \boxplus \circ T(\boxplus) \circ TT\pi_2 \rangle = \langle \boxplus \circ \mu \circ TT\pi_1, \boxplus \circ \mu \circ TT\pi_2 \rangle = \langle \boxplus \circ T\pi_1 \circ \mu, \boxplus \circ T\pi_2 \circ \mu \rangle = \langle \boxplus \circ T\pi_1, \boxplus \circ T\pi_2 \rangle \circ \mu = \boxplus' \circ \mu$. □

Lemma 24. *Let $\mathcal{A}$ be a symmetric monoidal closed category with binary coproducts $\_ \sqcup \_$. Let X be a monoid in $\mathcal{A}$ and $\boldsymbol{T}$ be its representation monad. If $(Z, \boxplus) \in |\mathcal{A}^{\boldsymbol{T}}|$ then the functor $\_ \sqcup Z$ is $\boldsymbol{T}$-distributive.*

Proof. Because $\mathcal{A}$ is symmetric monoidal closed, the functor $X \otimes \_$ has a right adjoint which implies that it preserves colimits. Hence we have a natural isomorphism $\iota : X \otimes (\_ \sqcup Z) \cong (X \otimes \_) \sqcup (X \otimes Z)$ given by $\iota = ap \circ s \circ (id \otimes [cur(i_1 \circ s), cur(i_2 \circ s)])$ (where i_1 and i_2 are the coproduct injections and s is the symmetry isomorphism) and we get $\delta = (id \sqcup \boxplus) \circ \iota$. Checking the equations for δ is routine. □

Lemma 25. *Let $\mathcal{A}$ be a monoidal category, X a monoid in $\mathcal{A}$ and $\boldsymbol{T}$ the representation monad of X. Then the functor $\_\otimes Y$ is $\boldsymbol{T}$-distributive.*

Lemma 26. *Let $\mathcal{A}$ be a monoidal closed category, X a monoid in $\mathcal{A}$, $\boldsymbol{T}$ the representation monad of X, and $(Y, \boxplus) \in |\mathcal{A}^{\boldsymbol{T}}|$. Then the functor $Y \Rightarrow \_$ is $\boldsymbol{T}$-distributive.*

Proof. We can define $\delta = cur((id \otimes ap) \circ a^{-1})$. The new addition is $\boxplus' = (id \Rightarrow \boxplus) \circ \delta = (id \Rightarrow \boxplus) \circ cur((id \otimes ap) \circ a^{-1}) = cur(\boxplus \circ (id \otimes ap) \circ a^{-1})$. We write μ and η as abbreviations for the multiplication and unit of the monad.

We have: $a^{-1} \circ (\eta \otimes id) = a^{-1} \circ ((0 \otimes id) \otimes id) \circ (l^{-1} \otimes id) = (0 \otimes id) \circ a^{-1} \circ (l^{-1} \otimes id) = (0 \otimes id) \circ l^{-1} = \eta$. This gives us the unit property: $\boxplus' \circ \eta = cur(\boxplus \circ (id \otimes ap) \circ a^{-1}) \circ \eta = cur(\boxplus \circ (id \otimes ap) \circ a^{-1} \circ (\eta \otimes id)) = cur(\boxplus \circ (id \otimes ap) \circ \eta) = cur(\boxplus \circ \eta \circ ap) = cur(ap) = id$.

For associativity we have $\boxplus' \circ \mu = cur(\boxplus \circ (id \otimes ap) \circ a^{-1}) \circ \mu = cur(\boxplus \circ (id \otimes ap) \circ a^{-1} \circ (\mu \otimes id))$ and on the other side of the equation $\boxplus' \circ (id \otimes \boxplus') = cur(\boxplus \circ (id \otimes ap) \circ a^{-1}) \circ (id \otimes \boxplus') = cur(\boxplus \circ (id \otimes ap) \circ a^{-1} \circ ((id \otimes \boxplus') \otimes id))$.

We can now show that both sides are equal (leaving out the "cur"), starting from the left: $\boxplus \circ (id \otimes ap) \circ a^{-1} \circ (\mu \otimes id) = \boxplus \circ (id \otimes ap) \circ a^{-1} \circ (((\oplus \otimes id) \circ a) \otimes id) = \boxplus \circ (id \otimes ap) \circ a^{-1} \circ ((\oplus \otimes id) \otimes id) \circ (a \otimes id) = \boxplus \circ (id \otimes ap) \circ (\oplus \otimes id) \circ a^{-1} \circ (a \otimes id) = \boxplus \circ (\oplus \otimes id) \circ (id \otimes ap) \circ a \circ (id \otimes a^{-1})) \circ a^{-1} = \boxplus \circ (\oplus \otimes id) \circ a \circ (id \otimes (id \otimes ap)) \circ (id \otimes a^{-1})) \circ a^{-1} = \boxplus \circ (id \otimes \boxplus) \circ (id \otimes (id \otimes ap)) \circ (id \otimes a^{-1})) \circ a^{-1} = \boxplus \circ (id \otimes (\boxplus \circ (id \otimes ap) \circ a^{-1})) \circ a^{-1} = \boxplus \circ (id \otimes (ap \circ cur(\boxplus \circ (id \otimes ap) \circ a^{-1}) \otimes id)) \circ a^{-1} = \boxplus \circ (id \otimes (ap \circ \boxplus' \otimes id)) \circ a^{-1} = \boxplus \circ (id \otimes ap) \circ (id \otimes (\boxplus' \otimes id)) \circ a^{-1} = \boxplus \circ (id \otimes ap) \circ a^{-1} \circ ((id \otimes \boxplus') \otimes id)$ □

Theorem 27. *Let $\mathcal{A}$ be a symmetric monoidal closed category, X a monoid in $\mathcal{A}$, and $\boldsymbol{T}$ the representation monad of X. Then the functor $\_ \odot \_$ can be lifted to $\mathcal{A}^{\boldsymbol{T}}$, where $\odot$ ranges over $\sqcup$, $\sqcap$, $\otimes$, $\Rightarrow$.*

Proof. Follows immediately from proposition 22 and lemmas 23, 24, 25, and 26. □

Now it would be nice if we could combine the construction of $\mathsf{WO}^{\boldsymbol{M}}$ and $\mathsf{WO} \downdownarrows \boldsymbol{\alpha}$, i.e. lift the functors to $\mathsf{WO}^{\boldsymbol{M}} \downdownarrows (\boldsymbol{\alpha}, \#)$.

Proposition 28. *Let $\mathcal{A}$ be a category and $\boldsymbol{T} = (T, \mu, \eta)$ be a monad on $\mathcal{A}$. An endofunctor $F : \mathcal{A} \to \mathcal{A}$ can be lifted to $\mathcal{A}^{\boldsymbol{T}} \downdownarrows (X, \oplus)$ if F is $\boldsymbol{T}$-distributive, F is retractable at X with retraction $X \xrightarrow{F} F(X) \xrightarrow{\bar{F}} X$, and the retraction maps are morphisms in $\mathcal{A}^{\boldsymbol{T}}$.*

We do not have the space to show that the given retractions (at monoids) for the functors $\_ \sqcap \_$, $\_ \sqcup \_$, $\_ \otimes \_$, and $\_ \Rightarrow \_$ are indeed morphisms in $\mathcal{A}^{\boldsymbol{T}}$, for any symmetric monoidal closed $\mathcal{A}$ with representation monad $\boldsymbol{T}$ — this result should not be too surprising since they are entirely built out of coherence maps.

Putting these results together, we have the following recipe for interpreting types by partially well-ordered sets:

Theorem 29. *Let α be an indecomposable ordinal. For any object $\boldsymbol{A} \in \mathsf{WO}$, which we can build from applying the functors $\sqcap$, $\sqcup$, $\otimes$, and $\Rightarrow$ in arbitrary order to $\boldsymbol{\alpha}$, we have:*

1. *There are morphisms $\boldsymbol{\alpha} \xrightarrow{a} \boldsymbol{A} \xrightarrow{\bar{a}} \boldsymbol{\alpha}$ such that $\bar{a} \circ a = id_\alpha$.*
2. *For any other object $\boldsymbol{B}$ built this way, we have a morphism $\boldsymbol{A} \xrightarrow{0} \boldsymbol{B}$.*
3. *We have a morphism $\boldsymbol{\alpha} \otimes \boldsymbol{A} \xrightarrow{\boxplus} \boldsymbol{A}$ such that $0 \boxplus x = x$ and $m \boxplus (n \boxplus x) = (m \oplus n) \boxplus x$.*
4. *The conversions $\boldsymbol{A} \xrightarrow{0} \boldsymbol{B}$ preserve addition, i.e. $0(n \boxplus x) = n \boxplus 0(x)$.*

Proof. This is just a summary of some of the results above. □

Moreover, we can extend this result to other endofunctors on WO, provided they are retractable at $\boldsymbol{\alpha}$, they are $\boldsymbol{M}$-distributive and their retraction maps are $\mathsf{WO}^{\boldsymbol{M}}$-homomorphisms.

4 Syntax

The previous section presented a semantic domain for the interpretation of rewrite systems that supports termination proofs. We still have to provide a connection between the syntax (Higher-Order Rewrite systems) and this semantics. Since we are not concerned here with implementability issues, we can choose Wolfram's generalisation of HRSs (see chapter 4 in [11]) as syntactic domain. HRSs are based on simply typed λ-calculus $\lambda^{\rightarrow}$. Its terms can be seen as either equivalence classes of λ-terms, the equivalence relation being $=_{\beta\eta}$, or as long β-normal forms, i.e. as canonical representatives of those classes.

The problem with HRSs is the lack of "nice" interpretations of $\lambda^{\rightarrow}$ in WO, simply because WO is not a CCC. Such an interpretation should assign the same values to $\beta\eta$-convertible terms, because HRSs rewrite modulo $\beta\eta$-conversion. Moreover, it should also interpret function types as sets of monotonic functions; this is necessary to lift termination of rewriting to its congruence closure, i.e. to rewriting on subterms. These objectives are conflicting for $\lambda^{\rightarrow}$: mapping convertible λ-terms to the same values means to interpret syntactic λ-abstraction by semantic λ-abstraction; but $\lambda^{\rightarrow}$ contains constant functions (like $\lambda x.c$) the semantic equivalent of which are not monotonic. The approach of van de Pol [8] tries to solve this problem by weakening the second objective and allowing certain non-monotonic functions in function types.

4.1 Term, Types, and Their Interpretations

Instead of allowing non-monotonic functions in the semantic domain, we sacrifice the other mentioned objective and allow β-equivalent terms to have different semantic interpretations.

To interpret types and terms in WO (or $\mathsf{WO}^M \downdownarrows \boldsymbol{\alpha}$), we shall give some functions from sets of types or terms into $|\mathsf{WO}|$ or $\bigcup|\mathsf{WO}|$, respectively. The notation $\bigcup|\mathsf{WO}|$ is shorthand for $\bigcup\{s \mid (s,>) \in |\mathsf{WO}|\}$; similarly for $\mathsf{WO}^M \downdownarrows \boldsymbol{\alpha}$, we suppress the application of the forgetful functor $U : \mathsf{WO}^M \downdownarrows \boldsymbol{\alpha} \rightarrow \mathsf{Set}$. Since the domain of the mentioned functions is always a set, their graph is a set as well and so we shall not worry about foundational issues.

Definition 30. Given a set of *base types* $\mathcal{B}$ we define the set of *types over* $\mathcal{B}$, $\mathit{Typ}(\mathcal{B})$, as the smallest set of words over the alphabet $\{\rightarrow, (,)\} \uplus \mathcal{B}$ satisfying:

1. $\sigma \in \mathcal{B} \Rightarrow \sigma \in \mathit{Typ}(\mathcal{B})$
2. $\alpha, \beta \in \mathit{Typ} \Rightarrow (\alpha \rightarrow \beta) \in \mathit{Typ}(\mathcal{B})$

$\mathit{Typ}(\mathcal{B})$ comprises the types of $\lambda^{\rightarrow}$. As usual, we drop many parentheses and take $\rightarrow$ to be right-associative. Having only one type constructor for non-base types reflects the meta-level we are dealing with, i.e. $\lambda^{\rightarrow}$. This does not prevent us from giving base types an internal structure reflecting the type structure we want on the object-level.

Definition 31. Let $\mathcal{B}$ be a set of base types, let $b : \mathcal{B} \rightarrow |\mathsf{WO}|$ a function mapping base types to objects in WO. We define a map $[\![\_]\!]_b : \mathit{Typ}(\mathcal{B}) \rightarrow |\mathsf{WO}|$ as follows:

$$\begin{aligned} [\![\sigma \rightarrow \tau]\!]_b &= [\![\sigma]\!]_b \Rightarrow [\![\tau]\!]_b \\ [\![\tau]\!]_b &= b(\tau), \text{ if } \tau \in \mathcal{B} \end{aligned}$$

Here, "$\Rightarrow$" is the functor from proposition 6.

Analogously, we derive from a function $b : \mathcal{B} \rightarrow |\mathsf{WO}^M \downdownarrows \boldsymbol{\alpha}|$ a function for all types $[\![\_]\!]_b : \mathit{Typ}(\mathcal{B}) \rightarrow |\mathsf{WO}^M \downdownarrows \boldsymbol{\alpha}|$, provided α is an indecomposable ordinal, because we can lift $\_ \Rightarrow \_$ at monoids, see theorem 29. In other words, the interpretation $[\![\sigma]\!]_b$ of a type σ in $\mathsf{WO}^M \downdownarrows \boldsymbol{\alpha}$ is an object $\boldsymbol{\alpha} \xrightarrow{\sigma} ((S, >_\sigma), \boxplus_\sigma) \xrightarrow{\bar{\sigma}} \boldsymbol{\alpha}$ in $\mathsf{WO}^M \downdownarrows \boldsymbol{\alpha}$. We shall write $\bar{\sigma}$, $\boxplus_\sigma$, and $>_\sigma$ to refer to the corresponding components of $[\![\sigma]\!]_b$.

Definition 32. An HRS-signature is a tuple $(\mathcal{B}, \mathcal{S}, \mathcal{C})$, where $\mathcal{B}$ is a set, $\mathcal{S}$ a set of *symbols*, and $\mathcal{C} : \mathcal{S} \rightarrow \mathit{Typ}(\mathcal{B})$ is a function assigning types to symbols.

$$\frac{x \notin \Gamma}{\Gamma \cup \{x : \sigma\} \vdash x : \sigma} \qquad \frac{}{\Gamma \vdash c : \mathcal{C}(c)}$$

$$\frac{\Gamma \vdash t : \sigma \to \tau \quad \Gamma \vdash u : \sigma}{\Gamma \vdash (t\ u) : \tau} \qquad \frac{\Gamma \cup \{x : \sigma\} \vdash t : \tau \quad x \notin \Gamma}{\Gamma \vdash (\lambda x : \sigma . t) : \sigma \to \tau}$$

Figure 2: The type system $\lambda^{\to}$

Independently from particular signatures, we assume the existence of a countably infinite set of *variables*, called $\mathcal{V}$. In the following, we shall usually suppress the signature $\Sigma = (\mathcal{B}, \mathcal{S}, \mathcal{C})$ and the interpretation of base types b, i.e. we assume a fixed Σ and b unless otherwise stated.

Definition 33. A *preterm* is a λ-term with variables taken from $\mathcal{V}$ and constants taken from $\mathcal{S}$. We require abstractions to be in Church-style [1], i.e. an abstraction has the form $(\lambda x : \sigma.t)$, where x is a variable, $\sigma \in \mathit{Typ}(\mathcal{B})$, and t is a preterm. We write $\Lambda\Sigma$ for the set of all preterms. Given a preterm t, we write $t{\downarrow}$ for the β-normal form of t if it exists.

Preterms are just untyped λ-terms with type annotations for abstractions. They may or may not be well-typed in some type system.

Definition 34. A *context* is a finite set $\Gamma \subset \mathcal{V} \times \mathit{Typ}(\mathcal{B})$ such that $(x, \tau) \in \Gamma \wedge (x, \sigma) \in \Gamma \Rightarrow \sigma = \tau$. Convention: we write $x : \tau$ instead of (x, τ) for elements of a context. We write $x \notin \Gamma$ as shorthand for $\neg\exists\sigma.(x, \sigma) \in \Gamma$.

Definition 35. A *judgement* is a triple (Γ, t, τ), written $\Gamma \vdash t : \tau$, where Γ is a context, $t \in \Lambda\Sigma$, and $\tau \in \mathit{Typ}(\mathcal{B})$. Given a judgement $J = \Gamma \vdash t : \tau$, we write $J{\downarrow}$ for the judgement $\Gamma \vdash t{\downarrow} : \tau$ (which exists if $t{\downarrow}$ exists).

Definition 36. The *type theory* $\lambda^{\to}$ is the smallest set of judgements that can be derived from the rules in figure 2.

Proposition 37. *Let $J \in \lambda^{\to}$. Then $J{\downarrow}$ exists and $J{\downarrow} \in \lambda^{\to}$.*

Proposition 38. *Let $\Gamma \vdash t : \tau$ and $\Gamma \vdash t : \tau'$ be derivable judgements in $\lambda^{\to}$. Then $\tau = \tau'$.*

Propositions 37 and 38 are well-known properties of $\lambda^{\to}$-Church [1].

Definition 39. A *symbol interpretation* is a function $\varrho : \mathcal{S} \to \bigcup |\mathsf{WO}^M \Downarrow \boldsymbol{\alpha}|$ such that $\varrho(s) \in [\![\mathcal{C}(s)]\!]_b$. A *variable interpretation* is a function $\upsilon : \mathcal{V} \to \bigcup |\mathsf{WO}^M \Downarrow \boldsymbol{\alpha}|$. If υ is a variable interpretation and $x \in \mathcal{V}$ then we write $\upsilon[x \mapsto y]$ for a variable interpretation υ' with $\upsilon'(x) = y$ and $\upsilon'(x') = \upsilon(x')$ if $x \neq x'$. A variable interpretation υ is *consistent* w.r.t. Γ if $\forall (x, \tau) \in \Gamma.\ \upsilon(x) \in [\![\tau]\!]_b$.

An *interpretation* is a pair (ϱ, υ) of a symbol interpretation ϱ and a variable interpretation υ. An interpretation (ϱ, υ) is called *consistent* (w.r.t. Γ) if υ is.

Definition 40. Let $\rho = (\varrho, \upsilon)$ be an interpretation.We define a partial function $[\![\_]\!]_\rho : \lambda^{\to} \to \bigcup |\mathsf{WO}^M \Downarrow \boldsymbol{\alpha}|$ with domain $\{\Gamma \vdash t : \tau \mid \rho \text{ consistent with } \Gamma\}$ by the following equations:

$$\begin{array}{rcl}
[\![\Gamma \vdash c : \tau]\!]_\rho & = & \varrho(c) \\
[\![\Gamma \vdash x : \tau]\!]_\rho & = & \upsilon(x) \\
[\![\Gamma \vdash (f\ a) : \tau]\!]_\rho & = & [\![\Gamma \vdash f : \sigma \to \tau]\!]_\rho([\![\Gamma \vdash a : \sigma]\!]_\rho) \\
 & & \text{where } (\Gamma \vdash a : \sigma) \in \lambda^{\to} \\
[\![\Gamma \vdash (\lambda x : \sigma.\, t) : \tau]\!]_\rho & = & (z \mapsto \overline{\sigma}(z) \boxplus_\tau [\![\Gamma \cup \{x : \sigma\} \vdash t : \tau]\!]_{\rho[x \mapsto z]}) \\
 & & \text{where } z \in [\![\sigma]\!]_b
\end{array}$$

To see that this definition is well-formed observe the following properties of semantic interpretation of types and judgements:

Theorem 41. *1.* $[\![\Gamma \vdash t : \tau]\!]_\rho \in [\![\tau]\!]_b$

2. Let $x \in FV(t)$, $z, z' \in [\![\sigma]\!]_b$ *and* $z >_\sigma z'$. *If* $\Gamma \vdash t : \tau$ *and if* $\rho[x \mapsto z]$ *is consistent w.r.t.* Γ *then*

$$[\![\Gamma \vdash t : \tau]\!]_{\rho[x \mapsto z]} >_\tau [\![\Gamma \vdash t : \tau]\!]_{\rho[x \mapsto z']}$$

Proof. By induction over the term structure. For variables and constants the result follows immediately from the assumptions about ρ.

Applications: for some σ, $\Gamma \vdash a : \sigma$ is a derivable judgement, because the domain of $[\![\_]\!]_\rho$ only contains derivable judgements and it does contain $\Gamma \vdash (f\ a) : \tau$. Moreover, proposition 38 claims that σ is unique.

Abstractions: the side-condition $z \in [\![\sigma]\!]_b$ ensures that the new interpretation $\rho[x \mapsto z]$ is consistent w.r.t. the larger context $\Gamma \cup \{x : s\}$. This allows us to apply the induction hypothesis to $\Gamma \cup \{x : \sigma\} \vdash t : \tau$. It remains to show that the constructed function is of the right form, e.g. monotonic. It is clear that it has the right domain and codomain, because $\bar{\sigma}$ maps σ-elements to α and $\boxplus_\tau$ is a function from $\alpha \times \tau$ to τ. For checking monotonicity we use the abbreviation $V(z)$ for $[\![\Gamma \cup \{x : \sigma\} \vdash t : \tau]\!]_{\rho[x \mapsto z]}$. The variable x either occurs free in t or not. If it does, then for $z >_\sigma z'$ we get by induction hypothesis $V(z) >_\tau V(z')$, if it does not $V(z) = V(z')$; thus in both cases $V(z) \geq_\tau V(z')$. The function $\bar{\sigma}$ is monotonic, hence $\bar{\sigma}(z) \ni \bar{\sigma}(z')$ and we get $(\bar{\sigma}(z), V(z)) >_{\boldsymbol{\alpha} \otimes \tau} (\bar{\sigma}(z'), V(z'))$. Since $\boxplus_\tau : \boldsymbol{\alpha} \otimes \tau \to \tau$ is monotonic we get the required result. □

The chosen interpretation for λ-abstraction may look a bit peculiar, because (as advertised) it does not have the property that β-convertible terms have equal interpretations. Therefore, β-reduction is only of limited use for the meta-level of rewriting.

Definition 42. A *presubstitution* is a function $\theta : \mathcal{V} \to \Lambda\Sigma$. Given $t \in \Lambda\Sigma$, we write t^θ for the preterm we get by replacing all free variables in t by their image under θ, avoiding name capture by α-conversion. A *substitution* is a triple (θ, Γ, Δ), written $\theta : \Gamma \to \Delta$, if θ is a presubstitution and Γ and Δ are contexts such that $\forall (x : \tau) \in \Gamma.\ \Delta \vdash \theta(x) : \tau$.

Proposition 43. *Let* $\Gamma \vdash t : \tau$ *and* $\theta : \Gamma \to \Delta$. *Then* $\Delta \vdash t^\theta : \tau$.

This is the standard substitution lemma for $\lambda^\to$, generalised to substitutions that replace all free variables at once. It motivates the following definition:

Definition 44. Let $J = \Gamma \vdash t : \tau$ and $\vartheta = \theta : \Gamma \to \Delta$. We write J^ϑ for the judgement $\Delta \vdash t^\theta : \tau$.

Definition 45. Let $\rho = (\varrho, \upsilon)$ be an interpretation consistent w.r.t. Δ and let $\vartheta = \theta : \Gamma \to \Delta$ be a substitution. We define another interpretation $\rho \circ \vartheta$ as $(\varrho, \upsilon \circ \vartheta)$ where the variable interpretation $\upsilon \circ \vartheta$ is given by:

$$(\upsilon \circ \vartheta)(x) = \begin{cases} [\![\Delta \vdash \theta(x) : \tau]\!]_\rho, & \text{if } (x, \tau) \in \Gamma \\ \upsilon(x), & \text{otherwise} \end{cases}$$

It is easy to see that $\rho \circ \theta$ is consistent w.r.t. Δ.

Proposition 46. *Let* $J = (\Gamma \vdash t : \tau) \in \lambda^\to$, $\vartheta = \theta : \Gamma \to \Delta$, *and* ρ *be an interpretation consistent w.r.t.* Δ. *Then* $[\![J^\vartheta]\!]_\rho = [\![J]\!]_{\rho \circ \vartheta}$.

Proposition 46 is a typical argument often used in semantic interpretations of the λ-calculus; it does quite happily work with rather non-standard interpretations of λ-abstractions as in our case.

Lemma 47. *Let* $(\Gamma \vdash t : \tau) \in \lambda^\to$ *and* $t \to_\beta t'$. *Let* ρ *be an interpretation consistent with* Γ. *Then* $[\![\Gamma \vdash t : \tau]\!]_\rho \geq_\tau [\![\Gamma \vdash t' : \tau]\!]_\rho$.

Proof. By induction over the term structure of t.

Base case: suppose t is a β-redex $(\lambda x.u)\ a$ and t' its contractum $u[a/x]$; we write $\hat{a}$ as shorthand for $[\![\Gamma \vdash a : \sigma]\!]_\rho$. Using proposition 46 and the algebraic properties of $\boxplus$ we get $[\![\Gamma \vdash t : \tau]\!]_\rho = [\![\Gamma \vdash (\lambda x.u)\ a : \tau]\!]_\rho = \bar{\sigma}(\hat{a}) \boxplus_\tau [\![\Gamma \cup \{x : \sigma\} \vdash u : \tau]\!]_{\rho[x \mapsto \hat{a}]} = \bar{\sigma}(\hat{a}) \boxplus_\tau [\![\Gamma \vdash t' : \tau]\!]_\rho \geq_\tau 0 \boxplus_\tau [\![\Gamma \vdash t' : \tau]\!]_\rho = [\![\Gamma \vdash t' : \tau]\!]_\rho$.

The induction steps are trivial, using the first part of theorem 41. ◻

4.2 Rules and Their Interpretations

The definition of HRS varies a bit in the literature. The following is another slight variation of the definitions of van de Pol or Wolfram [8, 11].

Definition 48. An *HRS-rule* is a tuple (Γ, l, r, τ) such that Γ is a context, $\tau \in \mathcal{B}$, and $\Gamma \vdash l : \tau$ and $\Gamma \vdash r : \tau$ are in $\lambda^{\rightarrow}$. Notation: we write rules as $\Gamma \vdash l \rightarrow r : \tau$.

The condition that τ is a base type does not restrict the expressive power as we can always η-expand rules by adding fresh variables to the context. The reason for using a context rather than λ-abstractions is the interpretation we have chosen for abstractions.

Definition 49. An HRS is a pair (Σ, R) where Σ is a signature and R a set of rules over Σ.

An HRS is associated with an ARS. The elements of this ARS are (derivable) judgements in β-normal form and the relation is given by the following notion of rule application.

Definition 50. Let (Σ, R) be an HRS. For a given a judgement $(\Gamma \vdash C : \sigma) \in \lambda^{\rightarrow}$ a *rule application* is pair $(\vartheta_l, \vartheta_r)$ of substitutions, $\vartheta_l = \theta_l : \Gamma \rightarrow \Delta$ and $\vartheta_r = \theta_r : \Gamma \rightarrow \Delta$, such that for all $(x : \sigma) \in \Gamma$ either $(*)$ $\theta_l(x) = \theta_r(x)$, or $(**)$ there is a rule $(E \vdash l \rightarrow r : \tau) \in R$ with $E = \{y_1 : \sigma_1, \ldots, y_n : \sigma_n\}$, $\sigma = \sigma_1 \rightarrow \cdots \rightarrow \sigma_n \rightarrow \tau$, and $\theta_l(x) = \lambda y_1 : \sigma_1. \cdots \lambda y_n : \sigma_n.l$ and $\theta_r(x) = \lambda y_1 : \sigma_1. \cdots \lambda y_n : \sigma_n.r$. (Notation: we shall abbreviate this as $\theta_l(x) = \boxed{l}$ and $\theta_r(x) = \boxed{r}$.) A rule application is called *proper* if for at least one $(x : \sigma) \in \Gamma$ we have $x \in \mathrm{FV}(C{\downarrow})$ and $(**)$.

We define a relation $\rightarrow_R$ on β-normal forms of judgements in $\lambda^{\rightarrow}$ as follows: given a judgement $J = (\Gamma \vdash C : \sigma) \in \lambda^{\rightarrow}$ and a proper rule application $(\vartheta_l, \vartheta_r)$ then $J^{\vartheta_l}{\downarrow} \rightarrow_R J^{\vartheta_r}{\downarrow}$.

The above notion gives a more or less canonical definition of HRS reduction; it is slightly more general than the definitions in the literature [11, 8] as it supports reduction with more than one rule at a time. The reason for requiring properness is the following proposition:

Proposition 51. *Let J be judgement in $\lambda^{\rightarrow}$ and $(\vartheta_l, \vartheta_r)$ be a rule application which is not proper. Then $J^{\vartheta_l}{\downarrow} = J^{\vartheta_r}{\downarrow}$.*

Therefore we need properness to give $\rightarrow_R$ a chance to be terminating. Any approach attempting to reason about termination of HRS reduction has to make similar restrictions in the definition of its reduction relation $\rightarrow_R$.

It is often convenient to assume that a rule application only instantiates one rule at one particular position in a term. We can define this as follows:

Definition 52. A rule application $(\vartheta_l, \vartheta_r)$ for a judgement J is called *linear* if $\vartheta_l(y) = \vartheta_r(y)$ for all but one y from the context of J, and if $\vartheta_l(x) = \boxed{l}$ and $\vartheta_r(x) = \boxed{r}$ then x occurs at most once in $J{\downarrow}$.

Lemma 53. *Let $J \rightarrow_R J'$. Then there are judgements $J_1, \ldots, J_n \in \lambda^{\rightarrow}$ such that $J = J_1 \rightarrow_R J_2 \rightarrow_R \cdots \rightarrow_R J_n = J'$ where each $J_i \rightarrow_R J_{i+1}$ by a linear rule application.*

Definition 54. Let ϱ be a symbol interpretation. A rule $\Gamma \vdash l \rightarrow r : \tau$ is called *ϱ-decreasing* if for all substitutions $\theta : \Gamma \rightarrow \Delta$ and all variable interpretations υ that are consistent with Δ it is true that $[\![\Delta \vdash l^\theta{\downarrow} : \tau]\!]_{(\varrho,\upsilon)} >_\tau [\![\Delta \vdash r^\theta{\downarrow} : \tau]\!]_{(\varrho,\upsilon)}$.

Theorem 55. *Let (Σ, R) be a HRS. If ϱ is a symbol interpretation such that all rules in R are ϱ-decreasing then* $(\boldsymbol{\lambda}^{\rightarrow}, \rightarrow_{\boldsymbol{R}}) \models \mathrm{SN}$.

Proof. We simply prove that $J \rightarrow_R J'$ implies $[\![J]\!]_\rho >_\tau [\![J']\!]_\rho$ where $J = B \vdash t : \tau$ and $\rho = (\varrho, \upsilon)$ for some variable interpretation υ consistent with B. By lemma 53 is is sufficient to consider the case in which $J \rightarrow_R J'$ by a linear rule application.

Suppose we get $J \rightarrow_R J'$ by applying the proper rule application $(\vartheta_l, \vartheta_r)$ to the judgement $C = E \vdash c : \sigma$. We can assume w.l.o.g. that $c = c{\downarrow}$. We show $[\![E \vdash c : \sigma]\!]_{\rho\circ\vartheta_l} >_\sigma [\![E \vdash c : \sigma]\!]_{\rho\circ\vartheta_r}$ by induction on the term structure of c.

If c is a symbol then we have a contradiction because the rule application cannot be proper.

If c is a variable then by properness $\theta_l(c) = \boxed{l}$ and $\theta_r(c) = \boxed{r}$. By assumption, the rule $\{y_1 : \sigma_1, \ldots, y_n : \sigma_n\} \vdash l \rightarrow r : \tau$ is ϱ-decreasing and from monotonicity of $n \boxplus \_$ for fixed n we conclude $[\![\theta_l(c)]\!]_\rho >_\sigma [\![\theta_r(c)]\!]_\rho$.

If c is a λ-abstraction $\lambda x : \tau.t$ then we define a new rule application $(\vartheta'_l, \vartheta'_r)$ for the judgement $E \cup \{x : \tau\} \vdash t : \sigma'$ which is as the old one except $\vartheta'_l(x) = x = \vartheta'_r(x)$. Obviously this rule application is still proper. By induction hypothesis we have $[\![E \cup \{x : \tau\} \vdash t : \sigma']\!]_{\rho\circ\vartheta'_l} >_{\sigma'} [\![E \cup \{x : \tau\} \vdash t : \sigma']\!]_{\rho\circ\vartheta'_r}$. Using again monotonicity of $n \boxplus \_$ we get the result for c.

If c is an application $t\,t_1 \cdots t_n$ $(n \geq 1)$ such that t is not an application, then t is either a constant or a variable since we assumed c to be in β-normal form. If c is either a constant or a variable that is not instantiated to a rule by the rule application, then $t^{\theta_l} = t^{\theta_r}$ and the rule application is still proper for at least one of the judgements $E \vdash t_i : \sigma_i$. We can apply either the induction hypothesis or proposition 51 to the t_i. By monotonicity of $[\![t^{\theta_l}]\!]_\rho$ the result follows. If t is instantiated to a rule $\Gamma \vdash l \rightarrow r : \tau$ with $\Gamma = \{y_1 : \sigma_1, \ldots, y_k : \sigma_k\}$ then we have (if $k = n$) $c^{\theta_l}{\downarrow} = (t\,t_1 \cdots t_n)^{\theta_l}{\downarrow} = l^{\theta'_l}{\downarrow}$ where $\theta'_l : \Gamma \rightarrow \Delta$ is a substitution with $\theta'_l(y_i) = t_i^{\theta_l}{\downarrow}$. The same argument applies to $c^{\theta_r}{\downarrow}$, giving us another substitution $\theta'_r : \Gamma \rightarrow \Delta$. Since we assumed the rule application to be linear, it is not proper for the judgements $J_i = E \vdash t_i : \sigma_i$. Thus $t_i^{\theta_l} = t_i^{\theta_r}$ by proposition 51 and we have $\theta'_l = \theta'_r$. We conclude $[\![l^{\theta'_l}{\downarrow}]\!]_\rho >_\tau [\![r^{\theta'_l}{\downarrow}]\!]_\rho = [\![r^{\theta'_r}{\downarrow}]\!]_\rho$ because the rule is ϱ-decreasing. If $k \neq n$, i.e. if the number (k) of arguments of the rule differs from the number (n) of arguments of t then $n < k$ because τ is a base type. In this case, the abstractions $\lambda y_{n+1} : \sigma_{n+1}. \cdots \lambda y_k : \sigma_k$ remain and we have to show that $[\![\lambda y_{n+1}. \cdots \lambda y_k. l^{\theta'_l}{\downarrow}]\!]_\rho >_{\sigma_{n+1} \rightarrow \cdots \rightarrow \sigma_k \rightarrow \tau} [\![\lambda y_{n+1}. \cdots \lambda y_k. r^{\theta'_l}{\downarrow}]\!]_\rho$. Again we can use the ϱ-decreasing argument and get the result from the monotonicity of $\boxplus$. $\square$

5 Applying the method

We can apply theorem 55 to show that a given HRS is terminating. For this we need the following ingredients:

- an interpretation b for any base type
- a symbol interpretation ϱ consistent w.r.t. b and
- a proof that each rule is ϱ-decreasing.

Suppose we want to define an enriched λ-calculus $\lambda^{\rightarrow,\times,+}$ with products and coproducts over some set of elementary types, see for example [10]. The first problem we have is that this is not quite an HRS, because we have product and coproduct types that can carry function types. The solution is to consider all types of this calculus to be base types and build $\lambda^{\rightarrow}$ on top it. We get a function b from base types to objects in $\mathsf{WO}^M \Downarrow \boldsymbol{\alpha}$ by mapping each elementary type to $\boldsymbol{\alpha}$ (any indecomposable limit ordinal will do) and each type constructor to some functor; in this case we can take the corresponding functors from theorem 29. The rewrite rules for β- and η-reduction of this calculus are shown in figure 3. Each rule is only a schema for infinitely many rules of the same shape for each combination of base types, so we have to look for a corresponding schematic

$$
\begin{array}{rcl}
\texttt{AP}\ (\texttt{LAM}\ f)\ a & \to & f\ a \\
\texttt{LAM}\ (\texttt{AP}\ f) & \to & f \\
\texttt{FST}\ (\texttt{PAIR}\ x\ y) & \to & x \\
\texttt{SND}\ (\texttt{PAIR}\ x\ y) & \to & y \\
\texttt{PAIR}\ (\texttt{FST}\ p)\ (\texttt{SND}\ p) & \to & p \\
\texttt{CASE}\ (\texttt{INL}\ x)\ f\ g & \to & f\ x \\
\texttt{CASE}\ (\texttt{INR}\ x)\ f\ g & \to & g\ x \\
\texttt{CASE}\ c\ (\lambda x:\sigma.\ f\ (\texttt{INL}\ x))\ (\lambda y:\tau.\ f\ (\texttt{INR}\ y)) & \to & f\ c
\end{array}
$$

Figure 3: Rules for $\lambda^{\to,\times,+}$

interpretation of the schematic symbols. The easiest case are the products as they are purely first-order:

$$
\begin{array}{rcl}
\varrho(\texttt{PAIR}_{\sigma,\tau})\ x\ y & = & ((1 \oplus \bar{\tau}(y)) \boxplus_\sigma x, (1 \oplus \bar{\sigma}(x)) \boxplus_\tau y) \\
\varrho(\texttt{FST}_{\sigma,\tau})\ (x,y) & = & x \\
\varrho(\texttt{SND}_{\sigma,\tau})\ (x,y) & = & y
\end{array}
$$

The three rules involving products are clearly ϱ-decreasing. The only problem was to define $\varrho(\texttt{PAIR})$ in such a way that it is a monotonic function in $\sigma \Rightarrow \tau \Rightarrow (\sigma \sqcap \tau)$.

This was pretty simple but also very typical: to get larger values on the left-hand sides the symbols have to make some "$1 \boxplus \_$" noise and to deal with erasing rules they have to garbage-collect the erased term using the addition operator.

For function types we do essentially the same thing, but now some meta-level β-reduction can take place:

$$
\begin{array}{rcl}
\varrho(\texttt{LAM}_{\sigma,\tau})\ f & = & 1 \boxplus_{\sigma\Rightarrow\tau} f \\
\varrho(\texttt{AP}_{\sigma,\tau})\ f\ a & = & f(a)
\end{array}
$$

The η-rule is trivial, the β-rule is only a little bit trickier: $[\![\texttt{AP}\ (\texttt{LAM}\ f^\theta)\ a^\theta]\!] = [\![\texttt{LAM}\ f^\theta]\!]([\![a^\theta]\!]) = (1 \boxplus_{\sigma\Rightarrow\tau} [\![f^\theta]\!])([\![a^\theta]\!]) >_\tau [\![f^\theta]\!]([\![a^\theta]\!]) = [\![(f^\theta\ a^\theta)]\!] \geq_\tau [\![(f^\theta\ a^\theta)\downarrow]\!]$. The last step used lemma 47.

Finding the interpretation for the coproduct type is similarly simple, but again we have to be careful to make $\varrho(\texttt{CASE})$ monotonic by collecting the garbage:

$$
\begin{array}{rcl}
\varrho(\texttt{INL}_{\sigma,\tau})\ x & = & i_1(x) \\
\varrho(\texttt{INR}_{\sigma,\tau})\ y & = & i_2(x) \\
\varrho(\texttt{CASE}_{\sigma,\tau,\nu})\ (i_1(x))\ f\ g & = & 1 \boxplus_\nu \overline{\tau \Rightarrow \nu}(g) \boxplus_\nu f(x) \\
\varrho(\texttt{CASE}_{\sigma,\tau,\nu})\ (i_2(y))\ f\ g & = & 1 \boxplus_\nu \overline{\sigma \Rightarrow \nu}(f) \boxplus_\nu g(y)
\end{array}
$$

Showing that the case-selection rules are ϱ-decreasing is straightforward (as for the function type), but the last rule is a bit more problematic as we have meta-level β-reduction on both sides of the rule. The variable f is second-order, i.e. we reach the β-normal form of $(f^\theta\ c^\theta)$ in a single β-step. For an arbitrary substitution $\theta(f) = \lambda x : \sigma + \tau.\ t$ we get as interpretation of the left-hand side of rule 8: if $[\![\theta(c)]\!] = i_1(c')$ then $1 \boxplus_\nu \overline{\tau \Rightarrow \nu}(g') \boxplus_\nu \overline{\sigma}(c') \boxplus_\nu [\![t[\texttt{INL}\ y/x]]\!]_{y\mapsto c'} \geq_\nu 1 \boxplus_\nu [\![t[\texttt{INL}\ y/x]]\!]_{y\mapsto c'} >_\nu [\![t[\texttt{INL}\ y/x]]\!]_{y\mapsto c'} = [\![t[c^\theta/x]]\!] = [\![(f\ c)^\theta\downarrow]\!]$. Here we used proposition 46 to compose an interpretation with a substitution. The case of the right injection is dual.

Gandy's paper [3] also considers the same example but he cannot directly show with his method the termination of the η-rule for the coproduct. Van de Pol [8] can deal with coproducts but not with internalised function types on the object-level.

6 Conclusion and further work

We have presented another semantic approach to termination proofs for higher-order term rewriting systems. The application of the method is fairly simple and one does not have to understand why the method works to apply it to an example. In particular, we have a number of criteria that allow us to deal with parametric types: any functor that is "retractable" at certain ordinals and "$\boldsymbol{M}$-distributive" is a candidate for the interpretation of parametric types.

Another generalisation of previous work by Gandy [3] and van de Pol [8] is to consider arbitrary ordinals rather than just the natural numbers. This is significant because several algebraic properties of natural sum do not generalise beyond natural numbers.

The method as presented here does not apply to dependent types, e.g. to show termination of the calculus of constructions. The main (and only) difficulty is that functors only correpond to parametric types, but not to dependent types. Thus one has to use a more sophisticated semantic construct than functors to mirror the dependency of the syntax.

Acknowledgements

People who helped developing this paper by providing feedback, advice, etc. include: Randall Dougherty, Marcello Fiore, Philippa Gardner, Jaco van de Pol, John Power, Alex Simpson and Judith Underwood. The figures were drawn with Paul Taylor's diagram package.

The research reported here was supported by SERC grant GR/J07303.

References

[1] Hendrik P. Barendregt. Lambda calculi with types. In *Handbook of Logic in Computer Science, Vol.2*, pages 117–309. Oxford Science Publications, 1992.

[2] M. C. F. Ferreira and H. Zantema. Total termination of term rewriting. In *Rewriting Techniques and Applications*, pages 213–227, 1993. LNCS 690.

[3] R.O. Gandy. Proofs of strong normalization. In J.P. Seldin and J.R. Hindley, editors, *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 457–477. Academic Press, 1980.

[4] Gerhard Hessenberg. Grundbegriffe der Mengenlehre. *Abhandlungen der Fries'schen Schule*, pages 479–706, 1906.

[5] Ernst Jacobsthal. Über den Aufbau der transfiniten Arithmetik. *Mathematische Annalen*, 67:130–144, 1909.

[6] Saunders MacLane. *Categories for the Working Mathematician*. Springer, 1971.

[7] John von Neumann. Zur Einführung der transfiniten Zahlen. *Acta litterarum ac scientarum*, 1:199–208, 1923.

[8] Jaco van de Pol. Termination proofs for higher-order rewrite systems. In *Higher-Order Algebra, Logic, and Term Rewriting*, pages 305–325, 1993. LNCS 816.

[9] Michael D. Potter. *Sets: An Introduction*. Oxford University Press, 1990.

[10] Simon Thompson. *Type Theory and Functional Programming*. Addison-Wesley, 1991.

[11] D.A. Wolfram. *The Clausal Theory of Types*. Cambridge University Press, 1993.

[12] Hans Zantema. Termination of term rewriting by interpretation. In *CTRS*, pages 155–167, 1992. LNCS 656.