On the relation between sized-types based termination and semantic labelling

Frédéric Blanqui[1] and Cody Roux[2] (INRIA)

[1] FIT 3-604, Tsinghua University, Haidian District, Beijing 100084, China, frederic.blanqui@inria.fr
[2] LORIA**, Pareo team, Campus Scientifique, BP 239, 54506 Vandoeuvre-lès-Nancy, Cedex, France, cody.roux@loria.fr

Abstract. We investigate the relationship between two independently developed termination techniques for first and higher-order rewrite systems. On the one hand, sized-types based termination uses types annotated with size expressions and Girard's reducibility candidates. On the other hand, semantic labelling transforms a rewrite system by annotating each function symbol with the semantic of its arguments.

First, we introduce a simplified version of sized-types based termination for the simply-typed lambda-calculus. Then, we give new proofs of the correctness of sized-types based termination by using semantic labelling both in the first and in the higher-order case.

As a consequence, we show that size-based termination can be easily extended to non-constructor systems with pattern symbols interpreted by monotone and strictly extensive functions.

1 Introduction

Sized-types were independently introduced by Hughes, Pareto and Sabry [18] and Giménez [13], and were extended to richer type systems, to rewriting or to richer size annotations by various other researchers [26, 1, 2, 5, 8].

Sized-types are types annotated with size expressions. For instance, if S is the type of character strings then, for all $a \in \mathbb{N}$, a type S^a is introduced to type the strings of length smaller or equal to a. For a first-order type like S, the size of a closed term in normal form is the height of its tree representation. In the general case, the size is some ordinal related to the interpretation of types in Girard's reducibility candidates [14]. However, as suggested in [5] and studied in [7], other notions of sizes may be interesting.

These size annotations can then be used to prove the termination of functions by checking that the size of arguments decreases along recursive calls.

At about the same time, semantic labelling was introduced for first-order systems by Zantema [27]. It received a lot of attention in the last years [22, 17, 20, 24, 25] and was recently extended to the higher-order case by Hamana [15].

In contrast with size-based termination, semantic labelling is not a termination criterion but transforms a system into another one whose termination is

** UMR 7503 CNRS-INPL-INRIA-Nancy2-UHP

equivalent and hopefully simpler to prove. The transformation consists in annotating function symbols with the semantics of their arguments in some model of the rewrite system. Finding such a model is of course difficult and makes this method difficult to use in practice.

In this paper, we study the relationship between these two methods. In particular, we give a new proof of the correctness of size-based termination by using semantic labelling. Hence, size-based termination provides an interesting model for semantic labelling, all the more so since not all function symbols need to have a semantics *a priori*.

Outline. We start in Section 2 by introducing our notations. We then explain in Section 3 what is size-based termination and give in Section 4 a simplified yet more powerful version. Then, we recall in Section 5 what is semantic labelling in the first-order case and show in Section 6 that size-based termination is an instance of semantic labelling. We then present in Section 7 what is semantic labelling in the higher-order case and show in Section 8 that size-based termination is an instance of semantic labelling in the higher-order case too.

For lack of space, some proofs and examples are detailed in [9].

2 Preliminaries

First-order terms. A *signature* $\mathcal{F}$ is made of a set $\mathcal{F}_n$ of *function symbols* of *arity* n for each $n \in \mathbb{N}$. Let $\mathcal{F}$ be the set of all function symbols. Given a set $\mathcal{X}$ of *variables*, the set of *first-order terms* $\mathcal{T}(\mathcal{F}, \mathcal{X})$ is defined as follows: $\mathcal{X} \subseteq \mathcal{T}$; if $\mathsf{f} \in \mathcal{F}_n$ and $t_1, \ldots, t_n \in \mathcal{T}$, then $\mathsf{f}(t_1, \ldots, t_n) \in \mathcal{T}$. By $\boldsymbol{t}$, we denote a sequence of terms $t_1, \ldots, t_n$ of length $n = |\boldsymbol{t}|$.

An *$\mathcal{F}$-algebra* $\mathcal{M}$ is given by a set M and, for each symbol $\mathsf{f} \in \mathcal{F}_n$, a function $\mathsf{f}^{\mathcal{M}} : M^n \to M$. Given a valuation $\mu : \mathcal{X} \to M$, the interpretation of a term t is defined as follows: $[\![x]\!]\mu = \mu(x)$ and $[\![\mathsf{f}(t_1, \ldots, t_n)]\!]\mu = \mathsf{f}^{\mathcal{M}}([\![t_1]\!]\mu, \ldots, [\![t_n]\!]\mu)$.

Positions are *words* on $\mathbb{N}$. We denote by ε the *empty* word and by $p \cdot q$ or pq the concatenation of p and q. Given a term t, we denote by $t|_p$ the subterm of t at position p, and by $t[u]_p$ the replacement of this subterm by u. Let $\mathrm{Pos}(\mathsf{f}, t)$ be the set of the positions of the occurrences of f in t.

Higher-order terms. The set of (simple) *types* is $\mathbb{T} = \mathcal{T}(\Sigma)$ where $\Sigma_0 = \mathcal{B}$ is a set of *base types*, $\Sigma_2 = \{\Rightarrow\}$ and $\Sigma_n = \emptyset$ otherwise. The sets of *positive* and *negative positions in a type* are inductively defined as follows:

- $\mathrm{Pos}^+(\mathsf{B}) = \varepsilon$ and $\mathrm{Pos}^-(\mathsf{B}) = \emptyset$,
- $\mathrm{Pos}^\delta(T \Rightarrow U) = 1 \cdot \mathrm{Pos}^{-\delta}(T) \cup 2 \cdot \mathrm{Pos}^\delta(U)$ where $--=+$ and $-+=-$.

Let $\mathcal{X}$ be an infinite set of variables. A typing environment Γ is a map from a finite subset of $\mathcal{X}$ to $\mathbb{T}$. For each type T, we assume given a set $\mathcal{F}_T$ of *function symbols of type* T. The sets $\Lambda_T(\Gamma)$ of *terms of type* T *in* Γ are defined as follows:

- $\mathcal{F}_T \subseteq \Lambda_T$;
- if $(x, T) \in \Gamma$ then $x \in \Lambda_T(\Gamma)$;

– if $t \in \Lambda_U(\Gamma, x : T)$, then $\lambda x^T t \in \Lambda_{T \Rightarrow U}(\Gamma)$;
– if $t \in \Lambda_{U \Rightarrow V}(\Gamma)$ and $u \in \Lambda_U(\Gamma)$, then $tu \in \Lambda_V(\Gamma)$.

Let $\mathcal{F}$ (resp. Λ) be the set of all function symbols (resp. terms). As usual, terms are considered up to renaming of bound variables. Let $\mathcal{X}(t)$ be the set of *free variables* of t.

A *substitution* σ is a map from a finite subset of $\mathcal{X}$ to Λ. We denote by $\binom{u}{x}$ the substitution mapping x to u, and by $t\sigma$ the application of σ to t.

A term t *β-rewrites* to a term u, written $t \to_\beta u$, if there is $p \in \mathrm{Pos}(t)$ such that $t|_p = (\lambda x^T v)w$ and $u = t[v^w_x]_p$.

A *rewrite rule* is a pair of terms $l \to r$ of the same type such that $\mathcal{X}(r) \subseteq \mathcal{X}(l)$. A *rewrite system* is a set $\mathcal{R}$ of rewrite rules. A term t *rewrites* to a term u, written $t \to_{\mathcal{R}} u$, if there is a position $p \in \mathrm{Pos}(t)$, a rule $l \to r \in \mathcal{R}$ and a substitution σ such that $t|_p = l\sigma$ and $u = t[r\sigma]_p$.

Constructor systems. A function symbol f is either a *constructor symbol* if no rule left-hand side is headed by f, or a *defined symbol* otherwise. A term is a *constructor term* if it is a variable or of the form $\mathsf{c}\boldsymbol{t}$ with c a constructor symbol and $\boldsymbol{t}$ constructor terms. A rewrite system is *constructor* if every rule is of the form $\mathsf{f}\boldsymbol{l} \to r$ with $\boldsymbol{l}$ constructor terms.

As usual, we assume that constructors form a valid inductive structure [6], that is, there is a well-founded quasi-ordering $\leq_{\mathcal{B}}$ on $\mathcal{B}$ such that, for all base type B, constructor $\mathsf{c} : \boldsymbol{T} \Rightarrow \mathsf{B}$ and base type C occuring at position p in T_i, either $\mathsf{C} <_{\mathcal{B}} \mathsf{B}$ or $\mathsf{C} \simeq_{\mathcal{B}} \mathsf{B}$ and $p \in \mathrm{Pos}^+(T_i)$. Mendler indeed showed that invalid inductive structures led to non-termination [21].

Given a constructor $\mathsf{c} : \boldsymbol{T} \Rightarrow \mathsf{B}$, let $\mathrm{Ind}(c)$ be the set of integers i such that T_i contains a base type $\mathsf{C} \simeq_{\mathcal{B}} \mathsf{B}$. A constructor c with $\mathrm{Ind}(\mathsf{c}) \neq \emptyset$ is said *recursive*.

A constructor $\mathsf{c} : \boldsymbol{T} \Rightarrow \mathsf{B}$ is *strictly-positive* if, for all i, either no base type equivalent to B occurs in T_i, or T_i is of the form $\boldsymbol{U}_i \Rightarrow \mathsf{C}$ with $\mathsf{C} \simeq_{\mathcal{B}} \mathsf{B}$ and no base type equivalent to B in $\boldsymbol{U}$.

3 Sized-types based termination

We now present a simplified version of the termination criterion introduced in [5], where the first author considers rewrite systems on terms of the Calculus of Algebraic Constructions, a complex type system with polymorphic and dependent types. Here, we restrict our attention to simply-typed λ-terms since there is no extension of semantic labelling to polymorphic and dependent types yet.

This termination criterion is based on the semantics of types in reducibility candidates [14]. An arrow type $T \Rightarrow U$ is interpreted by the set $[\![T \Rightarrow U]\!] = \{v \in \mathcal{T} \mid \forall t \in [\![T]\!], vt \in [\![U]\!]\}$. A base type B is interpreted as the fixpoint $[\![\mathsf{B}]\!]$ on the complete lattice of reducibility candidates of the monotone function $F_{\mathsf{B}}(X) = \{v \in \mathcal{SN} \mid \forall \mathsf{c} : \boldsymbol{T} \Rightarrow \mathsf{B}, \forall \boldsymbol{t}, \forall i \in \mathrm{Ind}(\mathsf{c}), v \to^* \mathsf{c}\boldsymbol{t} \Rightarrow t_i \in [\![T_i]\!]\}$ [6]. This fixpoint can thus be reached by transfinite iteration of F_{B} up to some limit ordinal ω_{B} strictly smaller than the first uncountable ordinal $\mathfrak{A}$. This provides us with the following notion of size: the size of a term $t \in [\![\mathsf{B}]\!]$ is the smallest

ordinal $o_{\mathsf{B}}(t) = \mathfrak{a} < \mathfrak{A}$ such that $t \in F^{\mathfrak{a}}_{\mathsf{B}}(\bot)$, where $\bot$ is the smallest element of the lattice and $F^{\mathfrak{a}}_{\mathsf{B}}$ is the function obtained after $\mathfrak{a}$ iterations of F_{B}.

This notion of size, which corresponds to the tree height for first-order constructor terms, has the following important properties:

- the ordering on sizes is well-founded;
- the size of a constructor term is strictly bigger than the size of its subterms;
- if $t \to t'$ then the size of t' is smaller than or equal to the size of t.

Size-based termination consists then in providing a way to syntactically represent the sizes of terms and, given for each function symbol an annotation describing how the size of its output is related to the sizes of its inputs, check that some measure on the sizes of arguments is decreasing in each recursive call.

Size algebra. Sizes are represented and compared by using a first-order term algebra $\mathcal{A} = \mathcal{T}(\Sigma, \mathcal{X})$ equipped with an ordering $\leq_{\mathcal{A}}$ such that:

- $<_{\mathcal{A}}$ is stable by substitution;
- $(\mathfrak{A}, <_{\mathfrak{A}})$, where $<_{\mathfrak{A}}$ is the usual ordering on ordinals, is a model of $(\mathcal{A}, <_{\mathcal{A}})$:
  - every symbol $\mathsf{h} \in \Sigma_n$ is interpreted by a function $\mathsf{h}^{\mathfrak{A}} : \mathfrak{A}^n \to \mathfrak{A}$;
  - if $a <_{\mathcal{A}} b$ then $[\![a]\!]\mu < [\![a]\!]\mu$ for all $\mu : \mathcal{X} \to \mathfrak{A}$.

To denote a size that cannot be expressed in $\mathcal{A}$ (or a size that we do not care of), Σ is extended with a (biggest) nullary element ∞. Let $\overline{\mathcal{A}}$ be the extended term algebra in which all terms containing ∞ are identified, $<_{\overline{\mathcal{A}}} = <_{\mathcal{A}} \cup \{(a, \infty) \mid a \in \mathcal{A}\}$ and $\leq_{\overline{\mathcal{A}}} = \leq_{\mathcal{A}} \cup \{(a, \infty) \mid a \in \overline{\mathcal{A}}\}$. Such an extension is often used in domain theory but with a least element instead.

Annotated types. The set of base types is now all the expressions B^a such that $\mathsf{B} \in \mathcal{B}$ and $a \in \overline{\mathcal{A}}$. The interpretation of B^∞ (also written B) is $[\![\mathsf{B}]\!]$ and, given $a \in \mathcal{A}$, the interpretation of B^a wrt a size valuation $\mu : \mathcal{X} \to \mathfrak{A}$ is the set of terms in $[\![\mathsf{B}]\!]$ whose size is smaller or equal to $[\![a]\!]\mu$: $[\![\mathsf{B}^a]\!]^\mu = F^{[\![a]\!]\mu}_{\mathsf{B}}(\bot)$. The ordering on $\mathcal{A}$ naturally induces the subtyping relation of Figure 1.

Fig. 1. Type system with size annotations

$$\Gamma \vdash \mathsf{f} :^s \tau^{\mathcal{A}}_{\mathsf{f}}\varphi \qquad \frac{(x,T) \in \Gamma}{\Gamma \vdash^s x : T} \qquad \frac{\Gamma, x : T \vdash^s u : U}{\Gamma \vdash^s \lambda x^T u : T \Rightarrow U}$$

$$\frac{\Gamma \vdash^s t : U \Rightarrow V \quad \Gamma \vdash^s u : U}{\Gamma \vdash^s tu : V} \qquad \frac{\Gamma \vdash^s t : T \quad T \leq T'}{\Gamma \vdash^s t : T'}$$

$$\frac{a \leq_{\overline{\mathcal{A}}} b}{\mathsf{B}^a \leq \mathsf{B}^b} \qquad \frac{T' \leq T \quad U \leq U'}{T \Rightarrow U \leq T' \Rightarrow U'} \qquad \frac{T \leq U \quad U \leq V}{T \leq V}$$

Definition 1. *Given a type T, let T^∞ be the annotated type obtained by annotating every occurrence of a base type in T by ∞, and let $\mathit{annot}^\alpha_{\mathsf{B}}(T)$ be the annotated type obtained by annotating every occurrence of a base type $\mathsf{C} \simeq_{\mathcal{B}} \mathsf{B}$ in T by α, and every occurrence of another base type by ∞. Conversely, given an annotated type T, let $|T|$ be the type obtained by removing all annotations.*

Given a size symbol $\mathsf{h} \in \Sigma$, let $\mathrm{Mon}^+(\mathsf{h})$ (resp. $\mathrm{Mon}^-(\mathsf{h})$) be the sets of integers i such that h is monotone (resp. anti-monotone) in its i-th argument. The sets of *positive* and *negative* positions in a type are then extended as follows:

- $\mathrm{Pos}^-(\mathsf{B}^a) = \emptyset$ and $\mathrm{Pos}^+(\mathsf{B}^a) = \{\varepsilon\} \cup 0 \cdot \mathrm{Pos}^+(a)$,
- $\mathrm{Pos}^\delta(\mathsf{h}(\boldsymbol{a})) = \bigcup\{i \cdot \mathrm{Pos}^{\epsilon\delta}(a_i) \mid i \in \mathrm{Mon}^\epsilon(h), \epsilon \in \{-,+\}\}$.

For every defined symbol f, we assume given an annotated type $\tau_\mathsf{f}^\mathcal{A}$ of the form $\boldsymbol{P} \Rightarrow \mathbf{B}^{\boldsymbol{\alpha}_\mathsf{f}} \Rightarrow \mathsf{B}^{\mathsf{f}^\mathcal{A}(\boldsymbol{\alpha}_\mathsf{f})}$ such that $|\tau_\mathsf{f}^\mathcal{A}| = \tau_\mathsf{f}$, $\mathcal{X}(\boldsymbol{P}) = \emptyset$, $\boldsymbol{\alpha}_\mathsf{f}$ are pairwise distinct variables, and $\mathcal{X}(\mathsf{f}^\mathcal{A}(\boldsymbol{\alpha}_\mathsf{f})) \subseteq \{\boldsymbol{\alpha}_\mathsf{f}\}$.

The arguments of type $\mathbf{B}$ are the ones whose size will be taken into account for proving termination. The arguments of type $\boldsymbol{P}$ are parameters and every rule defining f must be of the form $\mathsf{f}\boldsymbol{pl} \to r$ with $\boldsymbol{p} \in \mathcal{X}$, $|\boldsymbol{p}| = |\boldsymbol{P}|$ and $|\boldsymbol{l}| = |\boldsymbol{B}|$.

The size variables $\boldsymbol{\alpha}_\mathsf{f}$ are implicitly universally quantified and can thus be instantiated by any size expression. This is the reason why the typing rule for function symbols is $\mathsf{f} :^s \tau_\mathsf{f}^\mathcal{A}\varphi$. This is similar to type variables in ML.

For annotating constructor types, we assume that $\mathcal{A}$ has a monotone symbol $\mathsf{s} \in \Sigma_1$ interpreted as the ordinal successor, and that $a <_\mathcal{A} \mathsf{s}(a)$ for all a.

The annotated type of a constructor $\mathsf{c} : T_1 \dots T_n \Rightarrow \mathsf{B}$ is then:

$$\tau_\mathsf{c}^\mathcal{A} = annot_\mathsf{B}^\alpha(T_1) \Rightarrow \dots \Rightarrow annot_\mathsf{B}^\alpha(T_n) \Rightarrow \mathsf{B}^{\mathsf{c}^\mathcal{A}(\alpha)}$$

with $\mathsf{c}^\mathcal{A}(\alpha) = \infty$ if c is non-recursive, and $\mathsf{c}^\mathcal{A}(\alpha) = \mathsf{s}(\alpha)$ otherwise.

Termination criterion. We assume given a well-founded quasi-ordering $>_\mathcal{F}$ on $\mathcal{F}$ and, for each function symbol $\mathsf{f} :^s \boldsymbol{T} \Rightarrow \mathbf{B}^{\boldsymbol{\alpha}_\mathsf{f}} \Rightarrow \mathsf{B}^{\mathsf{f}^\mathcal{A}(\boldsymbol{\alpha}_\mathsf{f})}$ and set $X \in \{\mathcal{A}, \mathfrak{A}\}$, an ordered domain $(D_\mathsf{f}^X, <_\mathsf{f}^X)$ and a function $\zeta_\mathsf{f}^X : X^{|\boldsymbol{\alpha}_\mathsf{f}|} \to D_\mathsf{f}^X$ compatible with $\simeq_\mathcal{F}$ (*i.e.* $|\boldsymbol{\alpha}_\mathsf{f}| = |\boldsymbol{\alpha}_\mathsf{g}|$, $D_\mathsf{f}^X = D_\mathsf{g}^X$, $<_\mathsf{f}^X = <_\mathsf{g}^X$ and $\zeta_\mathsf{f}^X = \zeta_\mathsf{g}^X$ whenever $\mathsf{f} \simeq_\mathcal{F} \mathsf{g}$) and such that $>_\mathsf{f}^\mathfrak{A}$ is well-founded and $\zeta_\mathsf{f}^\mathfrak{A}(\boldsymbol{a}\mu) <_\mathsf{f}^\mathfrak{A} \zeta_\mathsf{f}^\mathfrak{A}(\boldsymbol{b}\mu)$ whenever $\zeta_\mathsf{f}^\mathcal{A}(\boldsymbol{a}) <_\mathsf{f}^\mathcal{A} \zeta_\mathsf{f}^\mathcal{A}(\boldsymbol{b})$ and $\mu : \mathcal{X} \to \mathfrak{A}$.

Usual domains are $\mathfrak{A}^n$ ordered lexicographically, or the multisets on $\mathfrak{A}$ ordered with the multiset extension of $>_\mathfrak{A}$.

Theorem 1 ([5]). *Let $\mathcal{R}$ be a constructor system. The relation $\to_\beta \cup \to_\mathcal{R}$ terminates if, for all defined $\mathsf{f} :^s \boldsymbol{P} \Rightarrow \mathbf{B}^{\boldsymbol{\alpha}} \Rightarrow \mathsf{B}^{\mathsf{f}^\mathcal{A}(\boldsymbol{\alpha})}$ and rule $\mathsf{f}\boldsymbol{pl} \to r \in \mathcal{R}$, there is an environment Γ and a size substitution $\binom{\boldsymbol{a}}{\boldsymbol{\alpha}}$ such that:*

- *pattern condition: for all θ, if $\boldsymbol{p}\theta \in [\![\boldsymbol{P}]\!]$ and $\boldsymbol{l}\theta \in [\![\mathbf{B}]\!]$ then there is ν such that, for all $(x,T) \in \Gamma$, $x\theta \in [\![T]\!]^\nu$ and $\boldsymbol{a}\nu \le o_\mathbf{B}(\boldsymbol{l}\theta)$;*
- *argument decreasingness: $\Gamma \vdash^s_{\mathsf{f}\boldsymbol{a}} r : \mathsf{B}^{\mathsf{f}^\mathcal{A}(\boldsymbol{a})}$ where $\vdash_{\mathsf{f}\boldsymbol{a}}$ is defined in Figure 2;*
- *size annotations monotony: $\mathrm{Pos}(\boldsymbol{\alpha}, \mathsf{f}^\mathcal{A}(\boldsymbol{\alpha})) \subseteq \mathrm{Pos}^+(\mathsf{f}^\mathcal{A}(\boldsymbol{\alpha}))$.*

The termination criterion introduced in [5] is not expressed exactly like this. The pattern condition is replaced by syntactic conditions implying the pattern condition, but the termination proof is explicitly based on the pattern condition.

This condition means that $\boldsymbol{a}$ is a valid representation of the size of $\boldsymbol{l}$, whatever is the instantiation of the variables of $\boldsymbol{l}$, and thus that any recursive call

with arguments of size smaller than $\boldsymbol{a}$ is admissible. That such a valid syntactic representation exists depends on $\boldsymbol{l}$, $\mathcal{A}$ and the size annotations of constructors.

The exact class of left-hand sides satisfying this condition is the subject of another work [7]. Note however that, with the chosen annotations for constructor types, many patterns of depth greater than one do not satisfy the pattern condition. This suggests to use a more precise annotation for constructors.

Fig. 2. Computability closure

$$\frac{\mathsf{g} <_{\mathcal{F}} \mathsf{f}}{\Gamma \vdash^{s}_{\mathsf{f}\boldsymbol{a}} \mathsf{g} : \tau^{\mathcal{A}}_{\mathsf{g}}\psi} \quad + \text{ variable, abstraction, application and subtyping rules of Figure 1}$$

$$\frac{\mathsf{g} \simeq_{\mathcal{F}} \mathsf{f} \quad \mathsf{g} :^{s} \boldsymbol{U} \Rightarrow \mathsf{C}^{\boldsymbol{\beta}} \Rightarrow \mathsf{C}^{\mathsf{g}^{\mathcal{A}}(\boldsymbol{\beta})} \quad \Gamma \vdash^{s}_{\mathsf{f}\boldsymbol{a}} \boldsymbol{u} : \boldsymbol{U} \quad \Gamma \vdash^{s}_{\mathsf{f}\boldsymbol{a}} \boldsymbol{m} : \mathsf{B}^{\boldsymbol{b}} \quad \zeta^{\mathcal{A}}_{\mathsf{f}}(\boldsymbol{b}) <^{\mathcal{A}}_{\mathsf{f}} \zeta^{\mathcal{A}}_{\mathsf{f}}(\boldsymbol{a})}{\Gamma \vdash^{s}_{\mathsf{f}\boldsymbol{a}} \mathsf{g}\boldsymbol{u}\boldsymbol{m} : \mathsf{C}^{\mathsf{g}^{\mathcal{A}}(\boldsymbol{b})}}$$

The expressive power of the criterion depends on $\mathcal{A}$. Taking the size algebra reduced to the successor symbol s (the decidability of which is proved in [3]) is sufficient to prove the termination of every "primitive recursive" function. As an example, consider the recursor $\mathsf{rec}_T : \mathsf{O} \Rightarrow T \Rightarrow (\mathsf{O} \Rightarrow T) \Rightarrow ((\mathsf{N} \Rightarrow \mathsf{O}) \Rightarrow (\mathsf{N} \Rightarrow T) \Rightarrow T) \Rightarrow T$ on the type O of Brouwer's ordinals whose constructors are $0 : \mathsf{O}$, $\mathsf{s} : \mathsf{O}^{\alpha} \Rightarrow \mathsf{O}^{\mathsf{s}\alpha}$ and $\mathsf{lim} : (\mathsf{N} \Rightarrow \mathsf{O}^{\alpha}) \Rightarrow \mathsf{O}^{\mathsf{s}\alpha}$, where N is the type of natural numbers whose constructors are $0 : \mathsf{N}$ and $\mathsf{s} : \mathsf{N}^{\alpha} \Rightarrow \mathsf{N}^{\mathsf{s}\alpha}$:

$$\begin{array}{rcl} \mathsf{rec0}uvw & \to & u \\ \mathsf{rec}(\mathsf{s}x)uvw & \to & vx(\mathsf{rec}xuvw) \\ \mathsf{rec}(\mathsf{lim}f)uvw & \to & wf(\lambda n\mathsf{rec}(fn)uvw) \end{array}$$

For instance, with $f : \mathsf{N} \Rightarrow \mathsf{O}^{\alpha}$, we have $\mathsf{lim}f : \mathsf{O}^{\mathsf{s}\alpha}$, $fn : \mathsf{O}^{\alpha}$ and $\mathsf{s}\alpha >_{\mathcal{A}} \alpha$.

An example of non-simply terminating system satisfying the criterion is the following system defining a division function $/ : \mathsf{N}^{\alpha} \Rightarrow \mathsf{N} \Rightarrow \mathsf{N}^{\alpha}$ by using a subtraction function $- : \mathsf{N}^{\alpha} \Rightarrow \mathsf{N} \Rightarrow \mathsf{N}^{\alpha}$.

$$\begin{array}{rcl} -x0 & \to & x \\ -0x & \to & 0 \\ -(\mathsf{s}x)(\mathsf{s}y) & \to & -xy \end{array} \qquad \begin{array}{rcl} /0x & \to & 0 \\ /(\mathsf{s}x)y & \to & \mathsf{s}(/(-xy)y) \end{array}$$

Indeed, with $x : \mathsf{N}^{x}$, we have $\mathsf{s}x : \mathsf{N}^{\mathsf{s}x}$, $-xy : \mathsf{N}^{x}$ and $\mathsf{s}x >_{\mathcal{A}} x$.

4 Annotating constructor types with MAX

In this section, we simplify the previous termination criterion by annotating constructor types in an algebra made of the following symbols:

- $0 \in \Sigma_0$ interpreted as the ordinal 0;
- $\mathsf{s} \in \Sigma_1$ interpreted as the successor ordinal;
- $\mathsf{max} \in \Sigma_2$ interpreted as the max on ordinals.

For the annotated type of a constructor $\mathsf{c} : T_1 \dots T_n \Rightarrow \mathsf{B}$, we now take:

$$\tau_{\mathsf{c}}^{\mathcal{A}} = annot_{\mathsf{B}}^{\alpha_1}(T_1) \Rightarrow \dots \Rightarrow annot_{\mathsf{B}}^{\alpha_n}(T_n) \Rightarrow \mathsf{B}^{\mathsf{c}_{\mathcal{A}}(\alpha_1,\dots,\alpha_n)}$$

with $\boldsymbol{\alpha}$ distinct variables, $\mathsf{c}_{\mathcal{A}}(\boldsymbol{\alpha}) = 0$ if c is non-recursive, and $\mathsf{c}_{\mathcal{A}}(\boldsymbol{\alpha}) = \mathsf{s}(\mathsf{max}(\alpha_i \mid i \in \mathrm{Ind}(\mathsf{c})))$ otherwise, where $\mathsf{max}(\alpha_1, \dots, \alpha_{k+1}) = \mathsf{max}(\alpha_1, \mathsf{max}(\alpha_2, \dots, \alpha_{k+1}))$ and $\mathsf{max}(\alpha_1) = \alpha_1$.

This does not affect the correctness of Theorem 1 since, in this case too, one can easily prove that constructors are computable: $\mathsf{c} \in [\![\tau_{\mathsf{c}}^{\mathcal{A}}]\!]^{\mu}$ for all μ.

Moreover, now, both constructors and defined symbols have a type of the form $annot_{\mathsf{B}_1}^{\alpha_1}(T_1) \Rightarrow \dots \Rightarrow annot_{\mathsf{B}_n}^{\alpha_n}(T_n) \Rightarrow \mathsf{B}^{\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})}$ with $\boldsymbol{\alpha}$ distinct variables.

This means that a constructor can be applied to any sequence of arguments without having to use subtyping. Indeed, previously, not all constructor applications were possible (take $\mathsf{c}xy$ with $\mathsf{c} : \mathsf{B}^{\alpha} \Rightarrow \mathsf{B}^{\alpha} \Rightarrow \mathsf{b}^{\mathsf{s}\alpha}$, $x : \mathsf{B}^{x}$ and $y : \mathsf{B}^{y}$) and some constructor applications required subtyping (take $\mathsf{c}x(\mathsf{d}x)$ with $\mathsf{c} : \mathsf{B}^{\alpha} \Rightarrow \mathsf{B}^{\alpha} \Rightarrow \mathsf{b}^{\mathsf{s}\alpha}$, $\mathsf{d} : \mathsf{B}^{\alpha} \Rightarrow \mathsf{B}^{\mathsf{s}\alpha}$ and $x : \mathsf{B}^{x}$).

We can therefore get rid of subtyping without losing much expressive power. It follows that every term has a most general type given by a simplified version of the type inference system $\vdash^{i}$ of [3] using unification only [9].

We now prove that the pattern and monotony conditions are always satisfied. First, given $\mathsf{f} :^{s} \boldsymbol{P} \Rightarrow \mathbf{B}^{\boldsymbol{\alpha}} \Rightarrow U$ and $\mathsf{f}\boldsymbol{xl} \to r \in \mathcal{R}$, we define Γ as the set of pairs (x, T) such that $x \in \mathcal{X}(\mathsf{f}\boldsymbol{pl})$ and T is:

- P_i if $x = p_i$,
- B_i^x if $x = l_i$,
- $annot_{\mathsf{B}_i}^{x}(T)$ if $\mathsf{c}\boldsymbol{u}x\boldsymbol{v}$ is a subterm of l_i and $\mathsf{c} : \boldsymbol{U} \Rightarrow T \Rightarrow \boldsymbol{V} \Rightarrow \mathsf{C}$.

Then, we let $\boldsymbol{a} = \sigma(\boldsymbol{l})$ where $\sigma(x) = x$ and $\sigma(\mathsf{c}\boldsymbol{t}) = \mathsf{c}^{\mathcal{A}}(\sigma(\boldsymbol{t}))$. If $\Gamma \vdash t : T$ and t is a non-variable constructor term then there is a base type B such that $\Gamma \vdash^{i} t : \mathsf{B}^{\sigma(t)}$. We can say that $\sigma(t)$ is the most general size of t.

Theorem 2 ([9]). *Let $\mathcal{R}$ be a constructor system. The relation $\to_{\beta} \cup \to_{\mathcal{R}}$ terminates if, for all $\mathsf{f} :^{s} \boldsymbol{P} \Rightarrow \mathbf{B}^{\boldsymbol{\alpha}} \Rightarrow \mathsf{B}^{\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})}$ and rule $\mathsf{f}\boldsymbol{pl} \to r \in \mathcal{R}$, we have:*

- *argument decreasingness: $\Gamma \vdash^{i}_{\mathsf{f}\boldsymbol{a}} r : \mathsf{B}^{a}$ and $a \leq_{\overline{\mathcal{A}}} \mathsf{f}^{\mathcal{A}}(\boldsymbol{a})$ where Γ and $\boldsymbol{a} = \sigma(\boldsymbol{l})$ are defined just before and $\vdash^{i}_{\mathsf{f}\boldsymbol{a}}$ is the type inference system $\vdash^{i}$ [9] with function applications restricted as in Figure 2.*

We will say that $\mathcal{R}$ *SB-terminates* if $\mathcal{R}$ satisfies the conditions of Theorem 2.

5 First-order semantic labelling

Semantic labelling is a transformation technique introduced by Hans Zantema for proving the termination of first-order rewrite systems [27]. It consists in labelling function symbols by using some model of the rewrite system.

Let $\mathcal{F}$ be a first-order signature and $\mathcal{M}$ be an $\mathcal{F}$-algebra equipped with a partial order $\leq_{\mathcal{M}}$. For each $\mathsf{f} \in \mathcal{F}_n$, we assume given a non-empty poset $(S^{\mathsf{f}}, \leq_{\mathsf{f}})$

and a *labelling function* $\pi_{\mathsf{f}} : M^n \to S^{\mathsf{f}}$. Then, let $\overline{\mathcal{F}}$ be the signature such that $\overline{\mathcal{F}}_n = \{\mathsf{f}_a \mid \mathsf{f} \in \mathcal{F}_n, a \in S^{\mathsf{f}}\}$.

The *labelling* of a term wrt a valuation $\mu : \mathcal{X} \to M$ is defined as follows:

- $lab^\mu(x) = x$,
- $lab^\mu(\mathsf{f}(t_1, \ldots, t_n)) = \mathsf{f}_{\pi_{\mathsf{f}}([\![t_1]\!]\mu, \ldots, [\![t_n]\!]\mu)}(lab^\mu(t_1), \ldots, lab^\mu(t_n))$.

The fundamental theorem of semantic labelling is then:

Theorem 3 ([27]). *Given a rewrite system $\mathcal{R}$, an ordered $\mathcal{F}$-algebra $(\mathcal{M}, \leq_{\mathcal{M}})$ and a labelling system $(S^{\mathsf{f}}, \leq_{\mathsf{f}}, \pi_{\mathsf{f}})$, the relation $\to_{\mathcal{R}}$ terminates if:*

1. *$\mathcal{M}$ is a quasi-model of $\mathcal{R}$, that is:*
   - *for all rule $l \to r \in \mathcal{R}$ and valuation $\mu : \mathcal{X} \to M$, $[\![l]\!]\mu \geq_{\mathcal{M}} [\![r]\!]\mu$,*
   - *for all $\mathsf{f} \in \mathcal{F}$, $\mathsf{f}^{\mathcal{M}}$ is monotone;*
2. *for all $\mathsf{f} \in \mathcal{F}$, π_{f} is monotone;*
3. *the relation $\to_{lab(\mathcal{R}) \cup Decr}$ terminates where:*
   $lab(\mathcal{R}) = \{lab^\mu(l) \to lab^\mu(r) \mid l \to r \in \mathcal{R}, \mu : \mathcal{X} \to M\}$,
   $Decr = \{\mathsf{f}_a(x_1, \ldots, x_n) \to \mathsf{f}_b(x_1, \ldots, x_n) \mid \mathsf{f} \in \mathcal{F}, a >_{\mathsf{f}} b\}$.

We will say that $\mathcal{R}$ *SL-terminates* if $\mathcal{R}$ satisfies the conditions of Theorem 3.

For instance, by taking $M = \mathbb{N}$, $\mathsf{0}^{\mathcal{M}} = 0$, $\mathsf{s}^{\mathcal{M}}(x) = x + 1$, $-^{\mathcal{M}}(x, y) = x$ and $/^{\mathcal{M}}(x, y) = x$, and by labelling $-$ and $/$ by the semantic of their first argument, we get the following *infinite* system which is easily proved terminating:

$$\begin{array}{rcll} -_i x\mathsf{0} & \to & x & (i \in \mathbb{N}) \\ -_0 \mathsf{0}x & \to & \mathsf{0} & \\ -_{i+1}(\mathsf{s}x)(\mathsf{s}y) & \to & -_i xy & (i \in \mathbb{N}) \end{array} \qquad \begin{array}{rcll} /_0 \mathsf{0}x & \to & \mathsf{0} & \\ /_{i+1}(\mathsf{s}x)y & \to & \mathsf{s}(/_i(-_i xy)y) & (i \in \mathbb{N}) \end{array}$$

6 First-order case

The reader will have already noticed some similarities between semantic labelling and size annotations. We here render them more explicit by showing that, in the first-order case, SB-termination implies SL-termination (not assuming that SB-termination implies termination).

Let $\mathcal{F}$ be a first-order signature. The set $\mathcal{T}(\mathcal{F}, \mathcal{X})$ of first-order terms can be seen as a strict subset of the set of simply-typed λ-terms over $\mathcal{F}$ by taking $\mathcal{B} = \{o\}$ and $f : o^n \Rightarrow o$ for every $f \in \mathcal{F}_n$.

In this case, the interpretation of a base type does not require transfinite iteration: all size are smaller than ω and $\mathfrak{A} = \mathbb{N}$ [6].

Note also that, by taking $\Gamma(x) = o^x$ for all x, every term t has a most general size $\sigma(t)$ given by its most general type: $\Gamma \vdash^i t : o^{\sigma(t)}$. This function σ extends to all terms the function σ defined in the previous section by taking $\sigma(\mathsf{f}(t_1, \ldots, t_n)) = \mathsf{f}^{\mathcal{A}}(\sigma(t_1), \ldots, \sigma(t_n))$ for all defined symbol f.

We now give a new proof of the correctness of SB-termination.

Theorem 4. *SB-termination implies SL-termination if:*

– *$\mathcal{R}$ is finitely branching and, for all* B*, the set of constructors of type* B *is finite;*
– *for all defined symbol* f*,* $\mathsf{f}^{\mathcal{A}}$ *and* $\zeta_{\mathsf{f}}^{\mathcal{A}}$ *are monotone.*

Proof. For the interpretation domain, we take $M = \mathfrak{A} = \mathbb{N}$ which has a structure of poset with $\leq_{\mathcal{M}} = \leq_{\mathfrak{A}} = \leq_{\mathbb{N}}$.

If $\mathsf{f}^{\mathcal{A}}$ is not the constant function equal to ∞ ($\mathsf{f}^{\mathcal{A}} \neq \infty$ for short), which is the case of constructors, then let $\mathsf{f}^{\mathcal{M}}(\mathfrak{a}) = [\![\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})]\!]\mu$ where $\boldsymbol{\alpha}\mu = \mathfrak{a}$.

When $\mathsf{f}^{\mathcal{A}} = \infty$, we proceed in a way similar to *predictive labelling* [17], a variant of semantic labelling where only the semantics of *usable symbols* need to be given when $\mathcal{M}$ is a $\sqcup$-*algebra* (all *finite* subsets of M have a lub wrt $\leq_{\mathcal{M}}$), which is the case of $\mathbb{N}$. Here, the notions of usable symbols and rules are not necessary and a semantics can be given to all symbols thanks to the strong assumptions of SB-termination.

For all $X \in \{\mathcal{A}, \mathfrak{A}\}$, let $>^X$ be the lexicographic combination of $>_{\mathcal{F}}$ and $>_{\mathsf{f}}^X$ as follows: $(\mathsf{f}, \boldsymbol{x}) >^X (\mathsf{g}, \boldsymbol{y})$ if $\mathsf{f} >_{\mathcal{F}} \mathsf{g}$ or $\mathsf{f} \simeq_{\mathcal{F}} \mathsf{g}$ and $\zeta_{\mathsf{f}}^X(\boldsymbol{x}) >_{\mathsf{f}}^X \zeta_{\mathsf{f}}^X(\boldsymbol{y})$. The relation $>$ is well-founded since the relations $>_{\mathcal{F}}$ and $>_{\mathsf{f}}^X$ are well-founded. We then define $\mathsf{f}^{\mathcal{M}}$ by induction on $>^{\mathfrak{A}}$ by taking $\mathsf{f}^{\mathcal{M}}(\mathfrak{a}) = max(\{0\} \cup \{[\![r]\!]\mu \mid \mathsf{f}\boldsymbol{l} \to r \in \mathcal{R}, \mu : \mathcal{X} \to \mathfrak{A}, [\![\boldsymbol{l}]\!]\mu \leq \mathfrak{a}\})$. This function is well defined since:

– For all subterm $\mathsf{g}\boldsymbol{m}$ in r, $(\mathsf{f}, \sigma(\boldsymbol{l})) >^{\mathcal{A}} (\mathsf{g}, \sigma(\boldsymbol{m}))$. Assume that $\mathsf{f} \simeq_{\mathcal{F}} \mathsf{g}$. Then, $\sigma(\boldsymbol{l}) >_{\mathcal{A}} \sigma(\boldsymbol{m})$. Hence, for all symbol f occuring in $\boldsymbol{l}$ or $\boldsymbol{m}$, $\mathsf{f}^{\mathcal{A}} \neq \infty$. Therefore, $[\![\boldsymbol{l}]\!]\mu = [\![\sigma(\boldsymbol{l})]\!]\mu$, $[\![\boldsymbol{m}]\!]\mu = [\![\sigma(\boldsymbol{m})]\!]\mu$ and $(\mathsf{f}, [\![\boldsymbol{l}]\!]\mu) >^{\mathfrak{A}} (\mathsf{g}, [\![\boldsymbol{m}]\!]\mu)$.
– The set $\{(\mathsf{f}\boldsymbol{l} \to r, \mu) \mid \mathsf{f}\boldsymbol{l} \to r \in \mathcal{R}, [\![\boldsymbol{l}]\!]\mu \leq \mathfrak{a}\}$ is finite. Indeed, since $\boldsymbol{l}$ are constructor terms and constructors are interpreted by monotone and strictly extensive functions (*i.e.* $\mathsf{c}^{\mathcal{A}}(\boldsymbol{\alpha}) \geq_{\mathcal{A}} \mathsf{s}(\mathsf{max}(\alpha_i \mid i \in \mathrm{Ind}(\mathsf{c}))))$, $[\![\boldsymbol{l}]\!]\mu$ is strictly monotone wrt μ and the height of $\boldsymbol{l}$. We cannot have an infinite set of $\boldsymbol{l}$'s of bounded height since, for all base type B, the set of constructors of type B is finite. And we cannot have an infinite set of r's since $\mathcal{R}$ is finitely branching.

We do not label the constructors, *i.e.* we take any singleton set for S^{c} and the unique (constant) function from M^n to S^{c} for π_{c}. For any other symbol f, we take $S^{\mathsf{f}} = D_{\mathsf{f}}^X$ which is well-founded wrt $>_{\mathsf{f}}$, and $\pi_{\mathsf{f}} = \zeta_{\mathsf{f}}^X$.

1. $\mathcal{M}$ is a quasi-model of $\mathcal{R}$:
   – Let $\mathsf{f} :^s \boldsymbol{P} \Rightarrow \mathsf{B}^{\boldsymbol{\alpha}} \Rightarrow \mathsf{B}^{\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})}$, $l \to r \in \mathcal{R}$ with $l = \mathsf{f}\boldsymbol{pl}$, and $\mu : \mathcal{X} \to M$. We have $[\![l]\!]\mu = \mathsf{f}^{\mathcal{M}}(\mathfrak{a})$ where $\mathfrak{a} = [\![\boldsymbol{l}]\!]\mu$. If $\mathsf{f}^{\mathcal{A}} = \infty$, then $\mathsf{f}^{\mathcal{M}}(\mathfrak{a}) = max(\{0\} \cup \{[\![r]\!]\mu \mid \mathsf{f}\boldsymbol{l} \to r \in \mathcal{R}, \mu : \mathcal{X} \to \mathfrak{A}, [\![\boldsymbol{l}]\!]\mu = \mathfrak{a}\})$ and $[\![l]\!]\mu \geq [\![r]\!]\mu$. Assume now that $\mathsf{f}^{\mathcal{A}} \neq \infty$. Since $\Gamma \vdash_{\mathsf{f}\boldsymbol{a}} r :^i \mathsf{B}^a$ and $a \leq_{\overline{\mathcal{A}}} \mathsf{f}^{\mathcal{A}}(\boldsymbol{a})$, we have $\sigma(r) = a \leq_{\overline{\mathcal{A}}} \mathsf{f}^{\mathcal{A}}(\boldsymbol{a}) = \sigma(l)$ where $\boldsymbol{a} = \sigma(\boldsymbol{l})$. By definition of Γ and σ, $\boldsymbol{a} \neq \infty$. Therefore, $\sigma(l) \neq \infty$ and $\sigma(r) \leq_{\mathcal{A}} \sigma(l)$. Hence, $[\![l]\!]\mu = \sigma(l)\mu \leq_{\mathfrak{A}} \sigma(r)\mu = [\![r]\!]\mu$ since $\leq_{\mathfrak{A}}$ is a model of $\leq_{\mathcal{A}}$.
   – If f is a non-recursive constructor, then $\mathsf{f}^{\mathcal{M}}(\mathfrak{a}) = 0$ is monotone. If f is a recursive constructor, then $\mathsf{c}^{\mathcal{M}}(\mathfrak{a}) = sup\{\mathfrak{a}_i \mid i \in \mathrm{Ind}(\mathsf{c})\} + 1$ is monotone. If $\mathsf{f}^{\mathcal{A}} \neq \infty$, then $\mathsf{f}^{\mathcal{M}}(\mathfrak{a}) = [\![\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})]\!]\mu$ where $\boldsymbol{\alpha}\mu = \mathfrak{a}$ is monotone since $\mathsf{f}^{\mathcal{A}}$ is monotone by assumption. Finally, if $\mathsf{f}^{\mathcal{A}} = \infty$, then $\mathsf{f}^{\mathcal{M}}(\mathfrak{a}) = max(\{0\} \cup \{[\![r]\!]\mu \mid \mathsf{f}\boldsymbol{l} \to r \in \mathcal{R}, \mu : \mathcal{X} \to \mathfrak{A}, [\![\boldsymbol{l}]\!]\mu \leq \mathfrak{a}\})$ is monotone.

2. If f is a defined symbol, then the function π_{f} is monotone by assumption. If c is a constructor, then the constant function π_{c} is monotone too.
3. We now prove that $\rightarrow_{lab(\mathcal{R})\cup Decr}$ is precedence-terminating, *i.e.* there is a well-founded relation on symbols $>$ such that, for all rule $\mathsf{f}\boldsymbol{l} \rightarrow r \in lab(\mathcal{R}) \cup Decr$, every symbol occurring in r is strictly smaller than f [22].
   Let $<$ be the lexicographic combination of $<_{\mathcal{F}}$ and $<_{\mathsf{f}}^{\mathfrak{A}}$ as follows: $\mathsf{g}_a < \mathsf{f}_b$ if $\mathsf{g} <_{\mathcal{F}} \mathsf{f}$ or $\mathsf{g} \simeq_{\mathcal{F}} \mathsf{f}$ and $a <_{\mathsf{f}}^{\mathfrak{A}} b$. The relation $>$ is well-founded since both $>_{\mathcal{F}}$ and $>_{\mathsf{f}}^{\mathfrak{A}}$ are well-founded. And $Decr$ is clearly precedence-terminating wrt $>$.
   Let now $\mathsf{f}\boldsymbol{l} \rightarrow r \in \mathcal{R}$, $\mu : \mathcal{X} \rightarrow M$ and $\mathsf{g}\boldsymbol{t}$ be a subterm of r. The label of f is $a = \pi_{\mathsf{f}}([\![\boldsymbol{l}]\!]\mu) = \zeta_{\mathsf{f}}^{X}(\sigma(\boldsymbol{l})\mu)$ and the label of g is $b = \zeta_{\mathsf{f}}^{X}(\sigma(\boldsymbol{m})\mu)$. By assumption, $(\mathsf{f}, \boldsymbol{l}) >^{\mathcal{A}} (\mathsf{g}, \boldsymbol{m})$. Therefore, $a >_{\mathsf{f}}^{\mathfrak{A}} b$. □

It is interesting to note that we could also have taken $M = \mathcal{A}$, assuming that $<_{\mathsf{f}}^{\mathcal{A}}$ is stable by substitution ($\zeta_{\mathsf{f}}^{\mathcal{A}}(\boldsymbol{a}\theta) <_{\mathsf{f}}^{\mathcal{A}} \zeta_{\mathsf{f}}^{\mathcal{A}}(\boldsymbol{b}\theta)$ whenever $\zeta_{\mathsf{f}}^{\mathcal{A}}(\boldsymbol{a}) <_{\mathsf{f}}^{\mathcal{A}} \zeta_{\mathsf{f}}^{\mathcal{A}}(\boldsymbol{b})$). The system labelled with $\mathcal{A}$ is a syntactic approximation of the system labelled with $\mathfrak{A}$. Although less powerful *a priori*, it may be interesting since it may provide a *finite* representation of the infinite $\mathfrak{A}$-labelled system.

For instance, by taking $\mathsf{0}^{\mathcal{M}} = 0$, $\mathsf{s}^{\mathcal{M}}(x) = \mathsf{s}x$, $-^{\mathcal{M}}(x, y) = x$ and $/^{\mathcal{M}}(x, y) = x$, and by labelling $-$ and $/$ by the semantic of their first argument, we get that every $\mathfrak{A}$-labelled rule is an i-instance of one of the following $\mathcal{A}$-labelled rules:

$$\begin{array}{rcll} -_i x\mathsf{0} & \rightarrow & x & (i \in \mathcal{X}) \\ -_0 \mathsf{0}x & \rightarrow & \mathsf{0} & \\ -_{\mathsf{s}i}(\mathsf{s}x)(\mathsf{s}y) & \rightarrow & -_i xy & (i \in \mathcal{X}) \end{array} \qquad \begin{array}{rcll} /_0 \mathsf{0}x & \rightarrow & \mathsf{0} & \\ /_{\mathsf{s}i}(\mathsf{s}x)y & \rightarrow & \mathsf{s}(/_i(-_i xy)y) & (i \in \mathcal{X}) \end{array}$$

Finally, we see from the proof that the system does not need to be constructor if the interpretation of a symbol occuring in a left hand-side is a monotone and strictly extensive. An example is given in [9].

Theorem 5. *Theorem 4 holds for any (non-constructor) system $\mathcal{R}$ such that, for all rule $\mathsf{f}\boldsymbol{l} \rightarrow r \in \mathcal{R}$ with $\mathsf{f}^{\mathcal{A}} = \infty$ and subterm $\mathsf{g}\boldsymbol{m}$ in $\boldsymbol{l}$:*

- *$\mathsf{g}^{\mathcal{A}}$ is monotone and strictly extensive: $\mathsf{g}^{\mathcal{A}}(\boldsymbol{\alpha}) \geq_{\mathcal{A}} \mathsf{s}(\mathsf{max}(\alpha_i \mid i \in \mathrm{Ind}(\mathsf{c})))$,*
- *if $\mathsf{g}^{\mathcal{A}} = \infty$, then $\mathsf{g} <_{\mathcal{F}} \mathsf{f}$ or $\mathsf{g} \simeq_{\mathcal{F}} \mathsf{f}$ and $\zeta_{\mathsf{f}}^{\mathcal{A}}(\sigma(\boldsymbol{m})) <_{\mathsf{f}}^{\mathcal{A}} \zeta_{\mathsf{f}}^{\mathcal{A}}(\sigma(\boldsymbol{l}))$.*

7 Higher-order semantic labelling

Semantic labelling was extended by Hamana [15] to second-order Inductive Data Type Systems (IDTSs) with higher-order pattern-matching [4]. IDTSs are a typed version of Klop's Combinatory Reduction Systems (CRSs) [19] whose categorical semantic based on binding algebras and $\mathcal{F}$-monoids [12] is studied by the same author and proved complete for termination [16].

The fundamental theorem of higher-order semantic labelling can be stated exactly as in the first-order case, but the notion of model is more involved.

CRSs and IDTSs. In CRSs, function symbols have a fixed arity. *Meta-terms* extends terms with the application $Z(t_1, \ldots, t_n)$ of a meta-variable $Z \in \mathcal{Z}$ of arity n to n meta-terms $t_1, \ldots, t_n$.

An assignment θ maps every meta-variable of arity n to a term of the form $\lambda x_1..\lambda x_n t$. Its application to a meta-term t, written, $t\theta$, is defined as follows:

- $x\theta = x$, $(\lambda x t)\theta = \lambda x(t\theta)$ and $\mathsf{f}(t_1,\ldots,t_n)\theta = \mathsf{f}(t_1\theta,\ldots,t_n\theta)$;
- for $\theta(Z) = \lambda x_1..\lambda x_n t$, $(Z(t_1,\ldots,t_n))\theta = t\{x_1 \mapsto t_1\theta,\ldots,x_n \mapsto t_n\theta\}$.

A rule is a pair of *meta-terms* $l \to r$ such that l is a higher-order pattern [23].

In IDTSs, variables, meta-variables and symbols are equipped with types over a discrete category $\mathbb{B}$ of base types. However, Hamana only considers *structural meta-terms* where abstractions only appear as arguments of a function symbol, variables are restricted to base types, meta-variables to first-order types and function symbols to second-order types. But, as already noticed by Hamana, this is sufficient to handle any rewrite system (see Section 8). Let $I_B^{\mathcal{Z}}(\Gamma)$ be the set of structural meta-terms of type B in Γ whose meta-variables are in $\mathcal{Z}$.

Models. The key idea of binding algebras [12] is to interpret variables by natural numbers using De Bruijn levels [10], and to handle bound variables by extending the interpretation to typing environments.

Let $\mathbb{F}$ be the category whose objects are the finite cardinals and whose arrows from n to p are all the functions from n to p. Let $\mathbb{E}$ be the (slice) category of typing environments whose objects are the maps $\Gamma : n \to \mathbb{B}$ and whose arrows from $\Gamma : n \to \mathbb{B}$ to $\Delta : p \to \mathbb{B}$ are the functions $\rho : n \to p$ such that $\Gamma = \Delta \circ \rho$.

Given $\Gamma : n \to \mathbb{B}$, let $\Gamma + B : n+1 \to \mathbb{B}$ be the environment such that $(\Gamma + B)(n) = B$ and $(\Gamma + B)(k) = \Gamma(k)$ if $k < n$.

Let $\mathbb{M}$ be the functor category $(\mathbf{Set}^{\mathbb{E}})^{\mathbb{B}}$. An object of $\mathbb{M}$ (presheaf) is given by a family of sets $M_B(\Gamma)$ for every base type B and environment Γ and, for every base type B and arrow $f : \Gamma \to \Delta$, a function $M_B(f) : M_B(\Gamma) \to M_B(\Delta)$ such that $M_B(id_\Gamma) = id_{M_B(\Gamma)}$ and $M_B(f \circ g) = M_B(f) \circ M_B(g)$. An arrow $\alpha : M \to N$ in $\mathbb{M}$ is a natural transformation, *i.e.* a family of functions $\alpha_B(\Gamma) : M_B(\Gamma) \to N_B(\Gamma)$ such that, for all $\rho : \Gamma \to \Delta$, $\alpha_B(\Gamma) \circ N_B(\rho) = M_B(\rho) \circ \alpha_B(\Delta)$.

Given $M \in \mathbb{M}$, $\Gamma \in \mathbb{E}$ and $\boldsymbol{B} \in \mathbb{B}$, let $up_\Gamma^{\boldsymbol{B}}(M) : M(\Gamma) \to M(\Gamma + \boldsymbol{B})$ be the arrow equal to $M(id_\Gamma + 0_\Delta)$ where 0_Δ is the unique morphism from 0 to Δ.

An $\mathcal{X} + \mathcal{F}$*-algebra* $\mathcal{M}$ is given by a presheaf $M \in \mathbb{M}$, an interpretation of variables $\iota : \mathcal{X} \to \mathcal{M}$ and, for every symbol $\mathsf{f} : (\boldsymbol{B}_1 \Rightarrow B_1) \Rightarrow \ldots \Rightarrow (\boldsymbol{B}_n \Rightarrow B_n) \Rightarrow B$ and environment Γ, an arrow $\mathsf{f}^{\mathcal{M}}(\Gamma) : \prod_{i=1}^n M_{B_i}(\Gamma + \boldsymbol{B}_i) \to M_B(\Gamma)$.

The category $\mathbb{M}$ forms a monoidal category with unit $\mathcal{X}$ and product $\bullet$ such that $(M \bullet N)_B(\Gamma)$ is the set of equivalence classes on the set of pairs $(t, \boldsymbol{u})$ with $t \in M_B(\Delta)$ and $u_i \in N_{\Delta(i)}(\Gamma)$ for some Δ, modulo the equivalence relation $\sim$ such that $(t, \boldsymbol{u}) \sim (t', \boldsymbol{u}')$ if there is $\rho : \Delta \to \Delta'$ for which $t \in M_B(\Delta)$, $t' = M_B(\rho)(t)$ and $u'_{\rho(i)} = u_i$.

To interpret substitutions, M must be an $\mathcal{F}$*-monoid*, *i.e.* a monoid $(M, \mu : M^2 \to M)$ compatible with the structure of $\mathcal{F}$-algebra [15, 9].

The presheaf $I^\emptyset$ equipped with the product $\mu_B(\Gamma)(t, \boldsymbol{u}) = t\{i \mapsto u_i\}$ (simultaneous substitution) is initial in the category of $\mathcal{F}$-monoids [16]. Hence, for all $\mathcal{F}$-monoid $\mathcal{M}$, there is a unique morphism $!^{\mathcal{M}} : I^\emptyset \to M$.

Labelling. As in the first-order case, for each $\mathsf{f} : (\boldsymbol{B}_1 \Rightarrow B_1) \Rightarrow \ldots \Rightarrow (\boldsymbol{B}_n \Rightarrow B_n) \Rightarrow B$, we assume given a non-empty poset $(S^{\mathsf{f}}, \leq_{\mathsf{f}})$ for *labels* and a *labelling*

function $\pi_{\mathsf{f}}(\Gamma) : \prod_{i=1}^{n} M_{B_i}(\Gamma + \boldsymbol{B}_i) \to S^{\mathsf{f}}$. Let $\overline{\mathcal{F}}_n = \{\mathsf{f}_a \mid \mathsf{f} \in \mathcal{F}_n, a \in S^{\mathsf{f}}\}$. The set of labelled meta-terms has a structure of $\mathcal{F}$-monoid [15].

The *labelling* of a meta-term wrt a valuation $\theta : \mathcal{Z} \to I^{\emptyset}$ is defined as follows:

- $lab^{\theta}_{B}(\Gamma)(x) = x$;
- $lab^{\theta}_{B}(\Gamma)(Z(t_1, \ldots, t_n)) = Z(lab^{\theta}_{B}(\Gamma)(t_1), \ldots, lab^{\theta}_{B}(\Gamma)(t_n))$;
- for $\mathsf{f} : (\boldsymbol{B}_1 \Rightarrow B_1) \Rightarrow \ldots \Rightarrow (\boldsymbol{B}_n \Rightarrow B_n) \Rightarrow B$ and $\Gamma_i = \Gamma, \boldsymbol{x}_i : \boldsymbol{B}_i$,
  $lab^{\theta}_{B}(\Gamma)(\mathsf{f}(\lambda \boldsymbol{x}_1 t_1, \ldots, \lambda \boldsymbol{x}_n t_n)) = \mathsf{f}_a(lab^{\theta}_{B_1}(\Gamma_1)(t_1), \ldots, lab^{\theta}_{B_n}(\Gamma_n)(t_n))$
  where $s = \pi_{\mathsf{f}}(!^{\mathcal{M}}_{B_1}(\Gamma_1)(t_1), \ldots, !^{\mathcal{M}}_{B_n}(\Gamma_n)(t_n))$.

We can now state Hamana's theorem for higher-order semantic labelling.

Theorem 6 ([15]). *Given a structural IDTS $\mathcal{R}$, an ordered $\mathcal{F}$-algebra $(\mathcal{M}, \leq_{\mathcal{M}})$ and a labelling system $(S^{\mathsf{f}}, \leq_{\mathsf{f}}, \pi_{\mathsf{f}})_{\mathsf{f} \in \mathcal{F}}$, the relation $\to_{\mathcal{R}}$ terminates if:*

1. *$(\mathcal{M}, \leq_{\mathcal{M}})$ is a quasi-model of $\mathcal{R}$, that is:*
   - *for all $l \to r : T \in \mathcal{R}$, $\theta : \mathcal{Z} \to I^{\emptyset}$ and Γ, $!^{\mathcal{M}}_{B}(\Gamma)(l\theta) \geq_{M_B(\Gamma)} !^{\mathcal{M}}_{B}(\Gamma)(r\theta)$,*
   - *for all $\mathsf{f} \in \mathcal{F}$, $\mathsf{f}^{\mathcal{M}}$ is monotone;*
2. *for all $\mathsf{f} \in \mathcal{F}$, π_{f} is monotone;*
3. *the relation $\to_{lab(\mathcal{R}) \cup Decr}$ terminates, where:*
   $lab(\mathcal{R}) = \{lab_B(\Gamma)(l\theta) \to lab_B(\Gamma)(r\theta) \mid l \to r : B \in \mathcal{R}, \theta : \mathcal{Z} \to I^{\emptyset}, \Gamma \in \mathbb{E}\}$,
   $Decr = \{\mathsf{f}_a(\ldots, \lambda \boldsymbol{x}_i Z_i(\boldsymbol{x}_i), \ldots) \to \mathsf{f}_b(\ldots, \lambda \boldsymbol{x}_i Z_i(\boldsymbol{x}_i), \ldots) \mid \mathsf{f} \in \mathcal{F}, a >_{\mathsf{f}} b\}$.

8 Higher-order case

In order to apply Hamana's higher-order semantic labelling, we first need to translate into a structural IDTS not only the rewrite system $\mathcal{R}$ but also β itself.

Translation to structural IDTS. Following Example 4.1 in [15], the relations β and $\mathcal{R}$ can be encoded in a structural IDTS as follows.

Let the set of *IDTS base types* $\mathbb{B}$ be the set $\mathcal{T}(\Sigma)$ where $\Sigma_0 = \mathcal{B}$ is the set of base types, $\Sigma_2 = \{Arr\}$ and $\Sigma_n = \emptyset$ otherwise. A simple type T can then be translated into an IDTS base type $\langle T \rangle$ by taking $\langle T \Rightarrow U \rangle = Arr(\langle T \rangle, \langle U \rangle)$ and $\langle T \rangle = T$ if $T \in \mathcal{B}$. Then, an environment Γ can be translated into an IDTS environment $\langle \Gamma \rangle$ by taking $\langle \emptyset \rangle = \emptyset$ and $\langle x : T, \Gamma \rangle = x : \langle T \rangle, \langle \Gamma \rangle$. Conversely, let $|T|$ be the simple type such that $\langle |T| \rangle = T$.

Let the set of *IDTS function symbols* be the set $\langle \mathcal{F} \rangle$ made of the symbols $\langle \mathsf{f} \rangle : \langle T_1 \rangle \Rightarrow \ldots \Rightarrow \langle T_n \rangle \Rightarrow \mathsf{B}$ such that $\mathsf{f} : T_1 \Rightarrow \ldots \Rightarrow T_n \Rightarrow \mathsf{B}$, and all the symbols $\lambda^{U}_{T} : (T \Rightarrow U) \Rightarrow Arr(T, U)$ and $@^{U}_{T} : Arr(T, U) \Rightarrow T \Rightarrow U$ such that T and U are IDTS base types. Note that only λ^{U}_{T} has a second order type.

A simply-typed λ-term t such that $\Gamma \vdash t : T$ can then be translated into an IDTS term $\langle t \rangle_{\Gamma}$ such that $\langle \Gamma \rangle \vdash \langle t \rangle_{\Gamma} : \langle T \rangle$ as follows:

- $\langle x \rangle_{\Gamma} = x$,
- $\langle \lambda x^{T} u \rangle_{\Gamma} = \lambda^{\langle U \rangle}_{\langle T \rangle}(\lambda x \langle u \rangle_{\Gamma, x:T})$ if $\Gamma, x : T \vdash u : U$,
- for $\mathsf{f} : T_1 \Rightarrow \ldots \Rightarrow T_n \Rightarrow \mathsf{B}$ and $U_i = T_{i+1} \Rightarrow \ldots \Rightarrow T_n \Rightarrow \mathsf{B}$,
  $\langle \mathsf{f} t_1 \ldots t_k \rangle_{\Gamma} = \lambda^{\langle U_{k+1} \rangle}_{\langle T_{k+1} \rangle}(\lambda x_{k+1} \ldots \lambda^{\langle U_n \rangle}_{\langle T_n \rangle}(\lambda x_n \langle \mathsf{f} \rangle(\langle t_1 \rangle_{\Gamma}, \ldots, \langle t_k \rangle_{\Gamma}, x_{k+1}, \ldots, x_n))\ldots)$,

– $\langle tu \rangle_\Gamma = @^{\langle V \rangle}_{\langle U \rangle}(\langle t \rangle_\Gamma, \langle u \rangle_\Gamma)$ if $\Gamma \vdash t : U \Rightarrow V$.

A rewrite rule $l \to r \in \mathcal{R}$ is then translated into the IDTS rule $\langle l \rangle \to \langle r \rangle$ where the free variables of l are seen as nullary meta-variables, and β-rewriting is translated into the family of IDTS rules $\langle \beta \rangle = \bigcup_{T,U \in \mathbb{B}} \beta^U_T$ where β^U_T is:

$$@^U_T(\lambda^U_T(\lambda x Z(x)), X) \to Z(X)$$

where Z (resp. X) is a meta-variable of type $T \Rightarrow U$ (resp. T). Note that only $\langle \beta \rangle$ uses non-nullary meta-variables.

Then, $\to_\mathcal{R} \cup \to_\beta$ terminates iff $\to_{\langle \mathcal{R} \rangle \cup \langle \beta \rangle}$ terminates [9].

Interpretation domain. We now define the interpretation domain M for interpreting $\langle \beta \rangle \cup \langle \mathcal{R} \rangle$. First, we interpret environments as arrow types:

– $M_T(\Gamma) = N_{Arr(\Gamma,T)}$ where:
$Arr(\emptyset, T) = T$ and $Arr(\Gamma + U, T) = Arr(\Gamma, Arr(U, T))$.

As explained at the beginning of Section 3, to every base type $\mathsf{B} \in \mathcal{B}$ corresponds a limit ordinal $\omega_\mathsf{B} < \mathfrak{A}$ that is the number of transfinite iterations of the monotone function F_B that is necessary to build the interpretation of B.

So, a first idea is to take $N_\mathsf{B} = \omega_\mathsf{B}$ and the set of functions from N_T to N_U for $N_{Arr(T,U)}$. But taking all functions creates some problems. Consider for instance the constructor $\mathsf{lim} : (\mathsf{N} \Rightarrow \mathsf{O}) \Rightarrow \mathsf{O}$. We expect $\mathsf{lim}^\mathcal{M}(\emptyset)(f) = sup\{f(n) \mid n \in N_\mathsf{N}\} + 1$ to be a valid interpretation, but $sup\{f(n) \mid n \in N_\mathsf{N}\} + 1$ is not in N_O for all function f. We therefore need to restrict $N_{Arr(T,U)}$ to the functions that correspond to (is realized by) some λ-term.

Hence, let $N_T = \{x \mid \exists t \in \mathcal{T},\ t \vdash_T x\}$ where $\vdash_T$ is defined as follows:

– $t \vdash_\mathsf{B} \mathfrak{a} \in \omega_\mathsf{B}$ if $t \in [\![\mathsf{B}]\!]$ and $o_\mathsf{B}(t) \geq \mathfrak{a}$,
– $v \vdash_{Arr(T,U)} f : N_T \to N_U$ if $v \in [\![|T| \Rightarrow |U|]\!]$ and $vt \vdash_U f(x)$ whenever $t \vdash_T x$.

Then, we can easily check that $sup\{f(n) \mid n \in N_\mathsf{N}\} + 1 \in N_\mathsf{O}$ now. Indeed, if there are v and t such that $v \vdash_{Arr(\mathsf{N},\mathsf{O})} f$ and $t \vdash_\mathsf{N} n$, then $vt \vdash_\mathsf{O} f(n)$ and $\mathsf{lim}(v) \vdash_\mathsf{O} sup\{f(n) \mid n \in N_\mathsf{N}\} + 1 \in N_\mathsf{O}$.

The action of M on $\mathbb{E}$-morphisms is defined as follows. Given $f : \Gamma \to \Delta$ with $\Gamma : n \to \mathbb{B}$ and $\Delta : p \to \mathbb{B}$, let $M_T(f) : M_T(\Gamma) \to M_T(\Delta)$ be the function mapping $x_0 \in N_{Arr(\Gamma,T)}$, $x_1 \in N_{\Delta(1)}$, …, $x_p \in N_{\Delta(p)}$ to $x_0(x_{f(1)}, \ldots, x_{f(n)})$.

Finally, the sets $M_B(\Gamma)$ and N_T are ordered as follows:

– $x \leq_{M_B(\Gamma)} y$ if $x \leq_{N_{Arr(\Gamma,B)}} y$ where:
  - $x \leq_{N_\mathsf{B}} y$ if $x \leq y$,
  - $f \leq_{N_{Arr(T,U)}} g$ if $f(x) \leq_{N_U} g(x)$ for all $x \in N_T$.

Interpretation of variables and function symbols. As one can expect, variables are interpreted by projections, λ^U_T by the identity, etc.:

– $\iota_{\Gamma(i)}(\Gamma)(i)(\boldsymbol{x}) = x_i$,

– $(\lambda_T^U)^{\mathcal{M}}(\Gamma)(f) = f$,
– $(@_T^U)^{\mathcal{M}}(\Gamma)(f,x)(\boldsymbol{y}) = f(\boldsymbol{y}, x(\boldsymbol{y}))$.

And one can easily check that these functions are valid interpretations, *i.e.* $\iota_{\Gamma(i)}(\Gamma)(i)(\boldsymbol{x}) \in N_{\Gamma(i)}$ and $(@_T^U)^{\mathcal{M}}(\Gamma)(f,x)(\boldsymbol{y}) \in N_U$.

Moreover, we have $(@_T^U)^{\mathcal{M}}(\Gamma)(f,x)(\boldsymbol{x}) = \mu_U(\Gamma)(f, \boldsymbol{p}x)$ where $p_i = \iota_{\Gamma(i)}(\Gamma)(i)$ and μ is the monoidal product $\mu_B(\Gamma)(t, u_1 \dots u_n)(\boldsymbol{x}) = t(u_1(\boldsymbol{x}), \dots, u_n(\boldsymbol{x}))$.

We can then verify that $\langle\beta\rangle$ is valid if (M,μ) is an $\mathcal{F}$-monoid, and that (M,μ) is an $\mathcal{F}$-monoid if, for all f and Γ, $\mathsf{f}^{\mathcal{M}}(\Gamma)(\boldsymbol{x})(\boldsymbol{y}) = \mathsf{f}^{\mathcal{M}}(\emptyset)(x_1(\boldsymbol{y}), \dots, x_n(\boldsymbol{y}))$ [9].

One can easily check that $(\lambda_T^U)^{\mathcal{M}}$ and $(@_T^U)^{\mathcal{M}}$ satisfy this property. Moreover, for all term $t \in I_T^{\emptyset}(\Gamma)$, we have $!_T^{\mathcal{M}}(\Gamma)(t)(\boldsymbol{a}) = [\![t]\!]\mu$ where $i\mu = a_i$ and:

$$[\![x]\!]\mu = \mu(x) \quad [\![@_T^U(v,t)]\!]\mu = [\![v]\!]\mu([\![t]\!]\mu) \quad [\![\lambda_T^U(\lambda x u)]\!]\mu = a \mapsto [\![u]\!]\mu_x^a$$
$$[\![\mathsf{f}(\boldsymbol{t})]\!]\mu = \mathsf{f}^{\mathcal{M}}(\emptyset)([\![\boldsymbol{t}]\!]\mu) \quad [\![Z(\boldsymbol{t})]\!]\mu = \mu(Z)([\![\boldsymbol{t}]\!]\mu)$$

Higher-order size algebra. In the first-order case, the interpretation of the function symbols f such that $\mathsf{f}^{\mathcal{A}}$ is not the constant function equal to ∞ (which includes constructors) is $\mathsf{f}^{\mathcal{M}}(\boldsymbol{a}) = [\![\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})]\!]\mu$ where $\boldsymbol{\alpha}\mu = \boldsymbol{a}$. To be able to do the same thing in the higher-order case, we need the size algebra $\mathcal{A}$ to be a typed higher-order algebra interpreted in the sets N_T.

Hence, now, we assume that size expressions are simply-typed λ-terms over a typed signature Σ, and that every function symbol $\mathsf{f} : \tau_{\mathsf{f}}$ is interpreted by ∞ or a size expression $\mathsf{f}^{\mathcal{A}} : \tau_{\mathsf{f}}$. We then let $\sigma : \mathcal{T} \to \overline{\mathcal{A}}$ be the function that replaces in a term every symbol f by $\mathsf{f}^{\mathcal{A}}$, all the terms containing ∞ being identified. Hence, for all term t containing no symbol f such that $\mathsf{f}^{\mathcal{A}} = \infty$, we have $[\![t]\!]\mu = [\![\sigma(t)]\!]\mu$. Finally, we define $<_{\mathcal{A}}$ as the relation such that $a <_{\mathcal{A}} b$ if, for all μ, $[\![a]\!]\mu <_{\mathfrak{A}} [\![b]\!]\mu$.

For instance, for a strictly-positive constructor $\mathsf{c} : \boldsymbol{T} \Rightarrow \mathsf{B}$ with $T_i = \boldsymbol{U}_i \Rightarrow \mathsf{B}_i$, we can assume that there is a symbol $\mathsf{c}^{\mathcal{A}} \in \Sigma$ interpreted by the function $\mathsf{c}^{\mathfrak{A}}(\boldsymbol{x}) = sup\{x_i \boldsymbol{y}_i \mid i \in \mathrm{Ind}(\mathsf{c}), \boldsymbol{y}_i \in N_{\langle \boldsymbol{U}_i \rangle}\} + 1$. Hence, in Brouwer's ordinals, we have $\sigma(\mathsf{lim} f) = \mathsf{lim}^{\mathcal{A}} f >_{\mathcal{A}} \sigma(fn) = fn$.

Hence, using such an higher-order size algebra, we can conclude:

Theorem 7. *SB-termination implies SL-termination if constructors are strictly-positive and the conditions of the Theorems 4 and 5 are satisfied.*

Proof. The proof is similar to the first-order case (Theorem 4). We only point out the main differences.

We first check that $\mathcal{M}$ is a quasi-model. The case of $\langle\beta\rangle$ is detailed in [9]. For $\langle\mathcal{R}\rangle$, we use the facts that $!_B^{\mathcal{M}}(\Gamma)(l\theta) \leq_{M_B(\Gamma)} !_B^{\mathcal{M}}(\Gamma)(r\theta)$ if $!_B^{\mathcal{M}}(\Gamma)(l\theta)(\boldsymbol{a}) \leq_{M_B(\emptyset)} !_B^{\mathcal{M}}(\Gamma)(r\theta)(\boldsymbol{a})$ for all $\boldsymbol{a}$, and that $!_B^{\mathcal{M}}(\Gamma)(l\theta)(\boldsymbol{a}) = [\![l]\!]\theta\mu$ where $i\mu = a_i$.

We do not label applications and abstractions. And for a defined symbol $\mathsf{f} : \boldsymbol{B} \Rightarrow B$, we take $S^{\mathsf{f}} = \coprod_{\Gamma} \prod_{i=1}^{n} M_{B_i}(\Gamma)$ and $\pi_{\mathsf{f}}(\Gamma)(\boldsymbol{x}) = (\Gamma, \boldsymbol{x})$.

We now define a well-founded relation on S^{f} that we will use for proving some higher-order version of precedence-termination. For dealing with $lab(\langle\mathcal{R}\rangle)$, we define $(\Gamma, \boldsymbol{x}) >_{\mathsf{f}}^{\mathcal{R}} (\Delta, \boldsymbol{y})$ if $\Delta = \Gamma + \Gamma'$ and, for all $\boldsymbol{z}\boldsymbol{z}'$, $\zeta_{\mathsf{f}}(\dots x_i(\boldsymbol{z}) \dots) >_{\mathsf{f}}^{\mathfrak{A}} \zeta_{\mathsf{f}}(\dots y_i(\boldsymbol{z}\boldsymbol{z}') \dots)$. For dealing with $lab(\langle\beta\rangle)$, we define $(\Gamma, \boldsymbol{x}) >_{\mathsf{f}}^{\beta} (\Delta, \boldsymbol{y})$ if $\Gamma =$

$\Delta+T$ and there is e such that, for all i and $\boldsymbol{z}$, $x_i(\boldsymbol{z}, e(\boldsymbol{z})) = y_i(\boldsymbol{z})$. Since $>_{\mathsf{f}}^{\mathcal{R}} \circ >_{\mathsf{f}}^{\beta}$ is included in $>_{\mathsf{f}}^{\mathcal{R}} \cup >_{\mathsf{f}}^{\beta} \circ >_{\mathsf{f}}^{\mathcal{R}}$, the relation $>_{\mathsf{f}} = >_{\mathsf{f}}^{\mathcal{R}} \cup >_{\mathsf{f}}^{\beta}$ is well-founded [11].

One can easily check that the functions π_{f} and $\mathsf{f}^{\mathcal{M}}$ are monotone.

We are now left to prove that $\rightarrow_{lab(\langle\beta\rangle)\cup lab(\langle\mathcal{R}\rangle)\cup Decr}$ terminates. First, remark that $\rightarrow_{lab(\langle\beta\rangle)}$ is included in $\rightarrow_{Decr}^{*}\rightarrow_{\langle\beta\rangle}$. Indeed, given $@_T^U(\lambda_T^U(\lambda x lab_U(\Gamma+T)(u)), lab_T(\Gamma)(t)) \rightarrow lab_U(\Gamma)(u_x^t) \in lab(\langle\beta\rangle)$, a symbol f occuring in u is labelled in $lab_U(\Gamma+T)(u)$ by something like $(\Gamma+T+\Delta, !_{\boldsymbol{B}}^{\mathcal{M}}(\Gamma+T+\Delta)(\boldsymbol{v}))$, and by something like $(\Gamma+\Delta, !_{\boldsymbol{B}}^{\mathcal{M}}(\Gamma+\Delta)(\boldsymbol{v}_x^t))$ in $lab_U(\Gamma)(u_x^t)$. Hence, the relation $\rightarrow_{lab(\langle\beta\rangle)\cup lab(\langle\mathcal{R}\rangle)\cup Decr}$ terminates if $\rightarrow_{\langle\beta\rangle\cup lab(\langle\mathcal{R}\rangle)\cup Decr}$ terminates.

By translating back IDTS types to simple types and removing the symbols λ_T^U (function $|\ |$), we get a *β-IDTS* [4] such that $\rightarrow_{\langle\beta\rangle\cup lab(\langle\mathcal{R}\rangle)\cup Decr}$ terminates if $\rightarrow_{|\langle\beta\rangle\cup lab(\langle\mathcal{R}\rangle)\cup Decr|}$ terminates [9]. Moreover, after [4], $\rightarrow_{|\langle\beta\rangle\cup lab(\langle\mathcal{R}\rangle)\cup Decr|}$ terminates if $|lab(\langle\mathcal{R}\rangle)\cup Decr|$ satisfies the *General Schema*. This can be easily checked by using the precedence $>$ on $\overline{\mathcal{F}}$ such that $\mathsf{f}_a > \mathsf{g}_b$ if $\mathsf{f} >_{\mathcal{F}} \mathsf{g}$ or $\mathsf{f} \simeq_{\mathcal{F}} \mathsf{g}$ and $a >_{\mathsf{f}} b$. (We do not need to use Hamana's results on *solid* IDTSs [15].) □

Conclusion. By studying the relationship between size-based termination and semantic labelling, we arrived at a new way to prove the correctness of size-based termination that allowed us to improve size-based termination itself very easily. We also think that size-based termination may provide an interesting framework for using semantic labelling in automated provers but this requires further research. Hence, this work can be carried on in various interesting directions by considering:

- richer type structures with polymorphic or dependent types;
- non-strictly positive constructors handled by using constrained types in [8];
- the inference of size annotations to automate size-based termination.

Acknowledgments. The authors want to thank Colin Riba very much for having pointed out some mistakes in a previous version and the problem with the interpretation of arrow types.

References

1. A. Abel. Termination checking with types. *Theoretical Informatics and Applications*, 38(4):277–319, 2004.
2. G. Barthe, M. J. Frade, E. Giménez, L. Pinto, and T. Uustalu. Type-based termination of recursive definitions. *Mathematical Structures in Computer Science*, 14(1):97–141, 2004.
3. F. Blanqui. Decidability of type-checking in the Calculus of Algebraic Constructions with size annotations. In *Proc. of CSL'05*, LNCS 3634.
4. F. Blanqui. Termination and confluence of higher-order rewrite systems. In *Proc. of RTA'00*, LNCS 1833.
5. F. Blanqui. A type-based termination criterion for dependently-typed higher-order rewrite systems. In *Proc. of RTA'04*, LNCS 3091.
6. F. Blanqui. Definitions by rewriting in the Calculus of Constructions. *Mathematical Structures in Computer Science*, 15(1):37–92, 2005.

7. F. Blanqui. A sized-types based termination criterion for dependently-typed higher-order rule-based programs, 2007. Draft.
8. F. Blanqui and C. Riba. Combining typing and size constraints for checking the termination of higher-order conditional rewrite systems. In *Proc. of LPAR'06*, LNCS 4246.
9. F. Blanqui and C. Roux. On the relation between sized-types based termination and semantic labelling. `http://www.loria.fr/~blanqui/`, 2009. Full version.
10. N. de Bruijn. Lambda-calculus notation with nameless dummies: a tool for automatic formula manipulation with application to the Church-Rosser theorem. *Indagationes Mathematicae*, 34(5):381–392, 1972.
11. H. Doornbos and B. von Karger. On the union of well-founded relations. *Logic Journal of the IGPL*, 6(2):195–201, 1998.
12. M. Fiore, G. Plotkin, and D. Turi. Abstract syntax and variable binding. In *Proc. of LICS'99*.
13. E. Giménez. *Un Calcul de Constructions infinies et son application à la vérification de systèmes communiquants*. PhD thesis, ENS Lyon, France, 1996.
14. J.-Y. Girard. *Interprétation fonctionelle et élimination des coupures dans l'arithmetique d'ordre supérieur*. PhD thesis, Université Paris VII, France, 1972.
15. M. Hamana. Higher-order semantic labelling for inductive datatype systems. In *Proc. of PPDP'07*.
16. M. Hamana. Universal algebra for termination of higher-order rewriting. In *Proc. of RTA'05*, LNCS 3467.
17. N. Hirokawa and A. Middeldorp. Predictive labeling. In *Proc. of RTA'06*, LNCS 4098.
18. J. Hughes, L. Pareto, and A. Sabry. Proving the correctness of reactive systems using sized types. In *Proc. of POPL'96*.
19. J. W. Klop, V. van Oostrom, and F. van Raamsdonk. Combinatory reduction systems: introduction and survey. *Theoretical Computer Science*, 121:279–308, 1993.
20. A. Koprowski and A. Middeldorp. Predictive labeling with dependency pairs using SAT. In *Proc. of CADE'07*, LNCS 4603.
21. N. P. Mendler. *Inductive Definition in Type Theory*. PhD thesis, Cornell University, United States, 1987.
22. A. Middeldorp, H. Ohsaki, and H. Zantema. Transforming termination by self-labelling. In *Proc. of CADE'96*, LNCS 1104.
23. D. Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. In *Proc. of ELP'89*, LNCS 475.
24. C. Sternagel and A. Middeldorp. Root labeling. In *Proc. of RTA'08*, LNCS 5117.
25. R. Thiemann and A. Middeldorp. Innermost termination of rewrite systems by labeling. In *Proc. of WRS'08*, ENTCS 204.
26. H. Xi. Dependent types for program termination verification. In *Proc. of LICS'01*.
27. H. Zantema. Termination of term rewriting by semantic labelling. *Fundamenta Informaticae*, 24:89–105, 1995.

9 Type inference

Let $\mathcal{X}(\Gamma) = \bigcup_{x \in \mathrm{dom}(\Gamma)} \mathcal{X}(x\Gamma)$ be the set of size variables occuring in the types of the variables of $\mathrm{dom}(\Gamma)$.

Fig. 3. Type inference system

$$\frac{(x,T) \in \Gamma}{\Gamma \vdash^i x : T} \qquad \frac{\rho : \mathcal{X}(\tau_{\mathsf{f}}^{\mathcal{A}}) \to \mathcal{X} \setminus \mathcal{X}(\Gamma) \text{ renaming}}{\Gamma \vdash^i \mathsf{f} : \tau_{\mathsf{f}}^{\mathcal{A}}\rho} \qquad \frac{\Gamma, x : T \vdash^i u : U}{\Gamma \vdash^i \lambda x^T u : T \Rightarrow U}$$

$$\frac{\begin{array}{c}\Gamma \vdash^i t : U \Rightarrow V \quad \Gamma \vdash^i u : U' \\ \rho : \mathcal{X}(U') \setminus \mathcal{X}(\Gamma) \to \mathcal{X} \setminus (\mathcal{X}(U) \cup \mathcal{X}(\Gamma)) \text{ renaming} \\ \varphi = mgu(U, U'\rho) \text{ with } \mathcal{X}(\Gamma) \text{ seen as constants}\end{array}}{\Gamma \vdash^i tu : V\varphi}$$

Lemma 1. *The type inference relation of Figure 3 is correct and complete:*

- *If $\Gamma \vdash^i t : T$ then $\Gamma \vdash t : T$.*
- *If $\Gamma \vdash t : T$ then there is T' and φ such that $\Gamma \vdash^i t : T'$ and $T'\varphi = T$.*

Proof. – Correctness: By induction on $\Gamma \vdash^i t : T$, using stability by substitution.
– Completeness: By induction on $\Gamma \vdash t : T$. We only detail the application case. By induction hypothesis, there is T' and φ, and U' and ψ such that $\Gamma \vdash^i t : T'$, $T'\varphi = U \Rightarrow V$, $\Gamma \vdash^i u : U'$ and $U'\psi = U$. It follows that T' is of the form $A \Rightarrow B$ and $U = A\varphi$ and $B\varphi = V$. Hence, there is $\theta = mgu(A, U')$ and φ' such that $\varphi = \theta\varphi'$. Therefore, $\Gamma \vdash^i tu : B\theta$ and there is φ' such that $B\theta\varphi' = V$. □

10 Proof of Theorem 2

Proof. We prove that, for all θ, if $\boldsymbol{p}\theta \in [\![\boldsymbol{P}]\!]$ and $\boldsymbol{l}\theta \in [\![\mathbf{B}]\!]$ then there is ν such that, for all $(x,T) \in \Gamma$, $x\theta \in [\![T]\!]^\nu$ and $\boldsymbol{a}\nu = o_{\mathbf{B}}(\boldsymbol{l}\theta)$.

Let T be a type in which $\mathrm{Pos}(\alpha, T) \subseteq \mathrm{Pos}^+(T)$. Then, $[\![T]\!]^{\mathfrak{a}}_\alpha$ is a monotone function on $\mathfrak{a}$ [5]. Given $t \in [\![T]\!]^{\mathfrak{A}}_\alpha$, let $o_{\lambda\alpha T}(t)$ be the smallest ordinal $\mathfrak{a}$ such that $t \in [\![T]\!]^{\mathfrak{a}}_\alpha$. Note that $o_{\lambda\alpha \mathsf{B}^\alpha} = o_{\mathsf{B}}$.

Let now $(x,T) \in \Gamma$. Since we have an inductive structure, $\mathrm{Pos}(x,T) \subseteq \mathrm{Pos}^+(T)$. One can easily check that $x\theta \in [\![T]\!]^\mu$ where μ is the constant valuation equal to $\mathfrak{A}$. We can thus define $x\nu = o_{\lambda x T}(x\theta)$ and we have $x\theta \in [\![T]\!]^\nu$.

We now prove that $a_i\nu = \sigma(l_i)\nu = o_{\mathbf{B}}(l_i\theta)$ by induction on l_i. If $l_i = x$ and $(x,T) \in \Gamma$ then $\sigma(l_i) = x$, $T = \mathsf{B}_i^x$ and $x\nu = o_{\lambda x \mathsf{B}_i^x}(x\theta) = o_{\mathsf{B}_i}(l_i\theta)$. Assume now that $l_i = \mathsf{c}\boldsymbol{t}$ with $\mathsf{c} : \boldsymbol{T} \Rightarrow \mathsf{C}$. If C is non-recursive, then $\sigma(l_i) = 0$ and $\sigma(l_i)\nu = 0 = o_{\mathsf{B}_i}(l_i\theta)$. Otherwise, $\sigma(l_i) = \mathsf{s}(\mathsf{max}(\sigma(t_{i_1}), \ldots, \sigma(t_{i_k})))$. If T_{i_j} is a base type then, by induction hypothesis, $\sigma(t_{i_j})\nu = o_{T_{i_j}}(t_{i_j}\theta)$. Otherwise, there is $(x,T) \in \Gamma$ such that $t_{i_j} = x$ and $\sigma(t_{i_j})\nu = x\nu = o_{\lambda x T}(x\theta)$. Since $o_{\mathsf{C}}(l_i\theta) = sup\{o_{\lambda\alpha \boldsymbol{T}}(\boldsymbol{t}\theta)\} + 1$, we have $\sigma(l_i)\nu = o_{\mathsf{B}_i}(l_i\theta)$. □

11 $\mathcal{F}$-monoids

To interpret (higher-order) substitutions, a presheaf M must be an *$\mathcal{F}$-monoid*, *i.e.* a monoid $(M, \mu : M^2 \to M)$ compatible with the structure of $\mathcal{F}$-algebra:

- $\mu_B(\Gamma)(\iota_B(\Delta)(i), \boldsymbol{u}) = u_i$;
- $\mu_B(\Gamma)(t, \iota_{\Delta(1)}(\Gamma)(1) \dots \iota_{\Delta(p)}(\Gamma)(p)) = t$;
- for $t \in M_B(\Theta)$, $u_i \in M_{\Theta(i)}(\Delta)$ and $v_i \in M_{\Delta(i)}(\Gamma)$,
  $\mu_B(\Gamma)(\mu_B(\Delta)(t, \boldsymbol{u}), \boldsymbol{v}) = \mu_B(\Gamma)(t, \mu_{\Theta(1)}(\Gamma)(u_1, \boldsymbol{v}) \dots \mu_{\Theta(p)}(\Gamma)(u_p, \boldsymbol{v}))$;
- for $\mathsf{f} : (\boldsymbol{B}_1 \Rightarrow B_1) \Rightarrow \dots \Rightarrow (\boldsymbol{B}_n \Rightarrow B_n) \Rightarrow B$ and $\Gamma_i = \Gamma + \boldsymbol{B}_i$,
  $\mu_B(\Gamma)(\mathsf{f}^{\mathcal{M}}(\Delta)(\boldsymbol{t}), \boldsymbol{u}) = \mathsf{f}^{\mathcal{M}}(\Gamma)(\mu_{B_1}(\Gamma_1)(t_1, \boldsymbol{v}_1), \dots, \mu_{B_n}(\Gamma_n)(t_n, \boldsymbol{v}_n))$
  where $v_{i,j} = up_\Gamma^{\boldsymbol{B}_i}(u_j)$ if $j < |\Delta|$, and $v_{i,j} = |\Gamma| + j - |\Delta|$ otherwise.

In the category of $\mathcal{F}$-monoids, the presheaf of meta-terms $I^{\mathcal{Z}}$ equipped with the product $\mu_B(\Gamma)(t, \boldsymbol{u}) = t\{x_1 \mapsto u_1, \dots, x_n \mapsto u_n\}$ (simultaneous substitution) is free. Hence, given an $\mathcal{F}$-monoid M, any valuation $\phi : \mathcal{Z} \to M$ can be uniquely extended into an $\mathcal{F}$-monoid morphism $\phi^* : I^{\mathcal{Z}} \to M$ such that:

- $\phi^*_B(\Gamma)(x) = \iota_B(\Gamma)(x)$;
- for $Z : \boldsymbol{B} \Rightarrow B$,
  $\phi^*_B(\Gamma)(Z(t_1, \dots, t_n)) = \mu_B(\Gamma)(\phi_B(\boldsymbol{B})(Z), \phi^*_{B_i}(\Gamma)(t_1) \dots \phi^*_{B_n}(\Gamma)(t_n))$;
- for $\mathsf{f} : (\boldsymbol{B}_1 \Rightarrow B_1) \Rightarrow \dots \Rightarrow (\boldsymbol{B}_n \Rightarrow B_n) \Rightarrow B$ and $\Gamma_i = \Gamma, \boldsymbol{x}_i : \boldsymbol{B}_i$,
  $\phi^*_B(\Gamma)(\mathsf{f}(\lambda \boldsymbol{x}_1 t_1, \dots, \lambda \boldsymbol{x}_n t_n)) = (\mathsf{f}^{\mathcal{M}})_B(\Gamma)(\phi^*_{B_1}(\Gamma_1)(t_1), \dots, \phi^*_{B_n}(\Gamma_n)(t_n))$.

Given a labelled term t, let $|t|$ be the term obtained after removing all labels. The presheaf of labelled meta-terms $\overline{I}^{\mathcal{Z}}$ has a structure of $\mathcal{F}$-monoid for each valuation $\theta : \mathcal{Z} \to I^{\emptyset}$ by taking:

- $\mu^\theta_B(\Gamma)(i, \boldsymbol{u}) = u_i$;
- for $Z : \boldsymbol{B} \Rightarrow B$, $\mu^\theta_B(\Gamma)(Z(t_1, \dots, t_n), \boldsymbol{u}) = Z(\mu^\theta_{B_1}(\Gamma)(t_1), \dots, \mu^\theta_{B_n}(\Gamma)(t_n))$;
- for $\mathsf{f} : (\boldsymbol{B}_1 \Rightarrow B_1) \Rightarrow \dots \Rightarrow (\boldsymbol{B}_n \Rightarrow B_n) \Rightarrow B$, $\Gamma_i = \Gamma + \boldsymbol{B}_i$ and $u_i \in \overline{I}^{\theta,\mathcal{Z}}_{\Delta(i)}(\Gamma)$,
  $\mu^\theta_B(\Gamma)(\mathsf{f}_a(\lambda \boldsymbol{x}_1 t_1, \dots, \lambda \boldsymbol{x}_n t_n), \boldsymbol{u}) = \mathsf{f}_b(\mu^\theta_{B_1}(\Gamma_1)(t_1, \boldsymbol{v}_1), \dots, \mu^\theta_{B_n}(\Gamma_n)(t_n, \boldsymbol{v}_n))$
  where $b = \pi^{\mathsf{f}}_B(\Gamma)(!^{\mathcal{M}}_{B_1}(\Gamma_1)(|t_1|\theta), \dots, !^{\mathcal{M}}_{B_n}(\Gamma_n)(|t_n|\theta))$,
  $v_{i,j} = up_\Gamma^{\boldsymbol{B}_i}(u_j)$ if $j < |\Delta|$, and $v_{i,j} = |\Gamma| + j - |\Delta|$ otherwise.

12 Validity of β

Lemma 2. *If (M, μ) is an $\mathcal{F}$-monoid, then $\langle \beta \rangle$ is valid in M.*

Proof. Let l and r be the left and right hand-sides of the rule β^U_T, $\theta : \mathcal{Z} \to I^{\emptyset}$ and Γ. Assume that $\theta(Z) = \lambda x u$ and $\theta(X) = t$. Then, $l\theta = @^U_T(\lambda^U_T(\lambda x u), t)$ and $r\theta = u^t_x$, and $!^{\mathcal{M}}_U(\Gamma)(l\theta) = (@^U_T)^{\mathcal{M}}(\Gamma)(\mathfrak{u}, \mathfrak{t}) = \mu_U(\Gamma)(\mathfrak{u}, \boldsymbol{p}\mathfrak{t})$ and $!^{\mathcal{M}}_U(\Gamma)(r\theta) = !^{\mathcal{M}}_U(\Gamma)(u^t_x)$, where $\mathfrak{u} = !^{\mathcal{M}}_U(\Gamma, x : T)(u)$ and $\mathfrak{t} = !^{\mathcal{M}}_T(\Gamma)(t)$. We now prove by induction on u that, for all Γ, $L = \mu_U(\Gamma)(\mathfrak{u}, \boldsymbol{p}\mathfrak{t})$ is equal to $R = !^{\mathcal{M}}_U(\Gamma)(u^t_x)$.

- $u = x$. Then, $u^t_x = t$, $\mathfrak{u} = p_{n+1}$ and $L = \mathfrak{t} = R$.

– $u = \mathsf{f}(\lambda \boldsymbol{x}_1 u_1, \ldots, \lambda \boldsymbol{x}_p u_p)$ with $\mathsf{f} : (\boldsymbol{B}_1 \Rightarrow B_1) \Rightarrow \ldots \Rightarrow (\boldsymbol{B}_p \Rightarrow B_p) \Rightarrow B$. Then, $u_x^t = \mathsf{f}(\lambda \boldsymbol{x}_1 {u_1}_x^t, \ldots, \lambda \boldsymbol{x}_p {u_p}_x^t)$, $R = \mathsf{f}^{\mathcal{M}}(\Gamma)(\boldsymbol{a})$ where $a_i = !_{B_i}^{\mathcal{M}}(\Gamma + \boldsymbol{B}_i)(u_i^t{}_x)$, and $\mathfrak{u} = \mathsf{f}^{\mathcal{M}}(\Gamma + T)(\boldsymbol{u}^*)$ where $u_i^* = !_{B_i}^{\mathcal{M}}(\Gamma + T + \boldsymbol{B}_i)(u_i)$. Since M is an $\mathcal{F}$-monoid, $L = \mathsf{f}^{\mathcal{M}}(\Gamma)(\boldsymbol{b})$ where $b_i = \mu_{B_i}(\Gamma + T + \boldsymbol{B}_i)(u_i^*, \boldsymbol{v}_i)$. And, by induction hypothesis, we have $b_i = a_i$. □

Lemma 3. *(M, μ) is an $\mathcal{F}$-monoid if* $\mathsf{f}^{\mathcal{M}}(\Gamma)(\boldsymbol{x})(\boldsymbol{y}) = \mathsf{f}^{\mathcal{M}}(\emptyset)(x_1(\boldsymbol{y}), \ldots, x_n(\boldsymbol{y}))$.

Proof. Let $\mathsf{f} : (\boldsymbol{B}_1 \Rightarrow B_1) \Rightarrow \ldots \Rightarrow (\boldsymbol{B}_n \Rightarrow B_n) \Rightarrow B$ and $\Gamma_i = \Gamma + \boldsymbol{B}_i$. We have to prove that $L = \mu_B(\Gamma)(\mathsf{f}^{\mathcal{M}}(\Delta)(\boldsymbol{t}), \boldsymbol{u})$ is equal to $R = \mathsf{f}^{\mathcal{M}}(\Gamma)(\mu_{B_1}(\Gamma_1)(t_1, \boldsymbol{v}_1), \ldots, \mu_{B_n}(\Gamma_n)(t_n, \boldsymbol{v}_n))$, where $v_{i,j} = up_\Gamma^{\boldsymbol{B}_i}(u_j)$ if $j < |\Delta|$, and $v_{i,j} = |\Gamma| + j - |\Delta|$ otherwise.

Let $y_i \in N_{\Gamma(i)}$. We have $L(\boldsymbol{y}) = \mathsf{f}^{\mathcal{M}}(\Delta)(\boldsymbol{t})(\boldsymbol{u}')$ where $u'_j = u_j(\boldsymbol{y})$. Now, by assumption, $L(\boldsymbol{y}) = \mathsf{f}^{\mathcal{M}}(\emptyset)(\boldsymbol{a})$ where $a_i = t_i(\boldsymbol{u}')$, and $R(\boldsymbol{y}) = \mathsf{f}^{\mathcal{M}}(\emptyset)(\boldsymbol{b})$ where $b_i = \mu_{B_1}(\Gamma_1)(t_i, \boldsymbol{v}_i)(\boldsymbol{y}) = t_i(\boldsymbol{v}'_i)$ and $v'_{i,j} = v_{i,j}(\boldsymbol{y})$. Hence, $L(\boldsymbol{y}) = R(\boldsymbol{y})$ since $v'_{i,j} = u_j(\boldsymbol{y}) = u'_j$. □

13 Translation to IDTS and β-IDTS

For the translation $\langle\ \rangle$ from λ-terms to second-order IDTS terms, we have the following properties:

Lemma 4. – *For all t and θ, $\langle t\theta \rangle = \langle t \rangle \langle \theta \rangle$.*
– *If $t \to_{\beta \cup \mathcal{R}} u$ then $\langle t \rangle \to_{\langle \beta \rangle \cup \langle \mathcal{R} \rangle} \langle u \rangle$.*

We now introduce a translation from a *structural* IDTS I having base types in $\mathbb{B}$ and some symbols $\lambda_T^U : (T \Rightarrow U) \Rightarrow Arr(T, U)$ for all $T, U \in \mathbb{B}$, to a *non-structural* IDTS J having base types in $\mathcal{B}$ and no symbol $\lambda_T^U : (T \Rightarrow U) \Rightarrow Arr(T, U)$. The symbols of J are all the symbols symbols $|\mathsf{f}| : |T_1| \Rightarrow \ldots \Rightarrow |T_n| \Rightarrow \mathsf{B}$ such that $\mathsf{f} : T_1 \Rightarrow \ldots \Rightarrow T_n \Rightarrow \mathsf{B}$ is a symbol of I distinct from some λ_T^U. A meta-term in $I_T^{\mathcal{Z}}(\Gamma)$ is then translated into a meta-term in $|I|_{|T|}^{\mathcal{Z}}(|\Gamma|)$ as follows:

– $|x| = x$,
– $|\mathsf{f}(t_1, \ldots, t_n)| = |\mathsf{f}|(|t_1|, \ldots, |t_n|)$,
– $\lambda_T^U(\lambda x u) = \lambda x |u|$,
– $|Z(t_1, \ldots, t_n)| = Z(|t_1|, \ldots, |t_n|)$.

Given a set $\mathcal{S}$ of rules in I, let $|\mathcal{S}|$ be the set of rules $|l| \to |r|$ in $|I|$ such that $l \to r \in \mathcal{S}$.

Lemma 5. – *For all t and θ, $|t\theta| = |t||\theta|$.*
– *If $t \to_{\mathcal{S}} u$ then $|t| \to_{|\mathcal{S}|} |u|$.*

Note that $@_T^U(\lambda_T^U(\lambda x Z(x)), X)$ is translated into $@_T^U(\lambda x Z(x), X)$. Hence, if I has symbols $@_T^U : Arr(T, U) \Rightarrow T \Rightarrow U$ and rules $@_T^U(\lambda_T^U(\lambda x Z(x)), X)$, then $|I|$ is a *β-IDTS* and $\to_{\langle \beta \rangle \cup \mathcal{S}}$ terminates if $|\mathcal{S}|$ satisfies the General Schema [4].

14 Example of non-constructor system

$$\begin{array}{rcl}
+\mathsf{0}y & \to & y \\
+(\mathsf{s}x)y & \to & \mathsf{s}(+xy) \\
+(\mathsf{s}x)y & \to & +x(\mathsf{s}y) \\
+(+xy)z & \to & +x(+yz) \\
\mathsf{F}\mathsf{0}uv & \to & v \\
\mathsf{F}(\mathsf{s}x)uv & \to & \mathsf{A}u(\mathsf{F}xuv) \\
\mathsf{F}(+xy)uv & \to & \mathsf{F}xu(\mathsf{F}yuv)
\end{array}$$

We take:

- $+^{\mathcal{A}}(x,y) = 2x+y+1$ and $\zeta_{+}(x,y) = 2x+y+1$
- $\mathsf{F}^{\mathcal{A}} = \infty$ and $\zeta_{\mathsf{F}}(x,u,v) = x$

The interpretation of $\mathsf{F}^{\mathcal{M}}$ is well-defined since $2x+y+1 \leq a$ implies both $x < a$ and $y < a$, and the labelled system that we obtain is precedence-terminating:

$$\begin{array}{rcl}
+_{y+1}\mathsf{0}y & \to & y \\
+_{2x+y+3}(\mathsf{s}x)y & \to & \mathsf{s}(+_{2x+y+1}) \\
+_{2x+y+3}(\mathsf{s}x)y & \to & +_{2x+y+2}x(\mathsf{s}y) \\
+_{4x+2y+z+3}(+_{2x+y+1}xy)z & \to & +_{2x+2y+z+2}x(+_{2y+z+1}yz) \\
\mathsf{F}_{0}\mathsf{0}uv & \to & v \\
\mathsf{F}_{x+1}(\mathsf{s}x)uv & \to & \mathsf{A}u(\mathsf{F}_{x}xuv) \\
\mathsf{F}_{2x+y+1}(+_{2x+y+1}xy)uv & \to & \mathsf{F}_{x}xu(\mathsf{F}_{y}yuv)
\end{array}$$