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. On the one hand, sized-types based termination (SBT) uses types annotated with size expressions and Girard's reducibility candidates, and applies on systems using constructor matching only. On the other hand, semantic labelling transforms a rewrite system by annotating each function symbol with the semantics of its arguments, and applies to any rewrite system.

First, we introduce a simplified version of SBT for the simply-typed lambda-calculus. Then, we give new proofs of the correctness of SBT using semantic labelling, both in the first and in the higher-order case. As a consequence, we show that SBT can be extended to systems using matching on defined symbols (*e.g.* associative functions).

1 Introduction

Sized types were independently introduced by Hughes, Pareto and Sabry [16] and Giménez [11], and were extended to richer type systems, to rewriting and to richer size annotations by various researchers [21,1,2,5,7].

Sized types are types annotated with size expressions. For instance, if T is the type of binary trees then, for each $a \in \mathbb{N}$, a type T^a is introduced to type the trees of height smaller or equal to a. In the general case, the size is some ordinal related to the interpretation of types in Girard's reducibility candidates [12]. However, as suggested in [5], 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, but this applies to functions defined by using matching on constructor terms only.

At about the same time, semantic labelling was introduced for first-order systems by Zantema [22]. It received a lot of attention in the last years and was recently extended to the higher-order case by Hamana [13].

In contrast with SBT, semantic labelling is not a termination criterion but transforms a system into another one whose termination is 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.

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

Finding a model may of course be difficult. We will see that the notion of size used in SBT provides such a model.

In this paper, we study the relationship between these two methods. In particular, we give a new proof of the correctness of SBT using semantic labelling. This will enable us to extend SBT to systems using matching on defined symbols.

Outline. Section 2 introduces our notations. Section 3 explains what SBT is and Section 4 introduces a simplified version of it. To ease the understanding of the paper, we first present the first-order case which already contains the main ideas, and then consider the higher-order case which requires more knowledge. Hence, in Section 5 (resp. 7), we recall what is semantic labelling in the first (resp. higher) order case and show in Section 6 (resp. 8) that SBT is an instance of it. For lack of space, some proofs are given in the Appendices of [8].

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 usual: $\mathcal{X} \subseteq \mathcal{T}$; if $\mathsf{f} \in \mathcal{F}_n$ and $\boldsymbol{t}$ is a sequence $t_1, \dots, t_n \in \mathcal{T}$ of length $n = |\boldsymbol{t}|$, then $\mathsf{f}(\boldsymbol{t}) \in \mathcal{T}$.

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 $[\![\mathsf{f}(t_1, \dots, t_n)]\!]\mu = \mathsf{f}^{\mathcal{M}}([\![t_1]\!]\mu, \dots, [\![t_n]\!]\mu)$.

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 $\mathrm{Pos}^-(\mathsf{B}) = \emptyset$ for each $\mathsf{B} \in \mathcal{B}$,
- $\mathrm{Pos}^\delta(T \Rightarrow U) = 1 \cdot \mathrm{Pos}^{-\delta}(T) \cup 2 \cdot \mathrm{Pos}^\delta(U)$ 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 usual: $\mathcal{F}_T \subseteq \Lambda_T(\Gamma)$; 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). 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 $p \in \mathrm{Pos}(t)$, $l \to r \in \mathcal{R}$ and 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 *pattern* is a variable or a term of the form $\mathsf{c}\boldsymbol{t}$ with c a constructor symbol and $\boldsymbol{t}$ patterns. A rewrite system is *constructor* if every rule is of the form $\mathsf{f}\boldsymbol{l} \to r$ with $\boldsymbol{l}$ patterns.

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 each 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 lead to non-termination [18].

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 each i, either no base type equivalent to B occurs in T_i, or T_i is of the form $\boldsymbol{U} \Rightarrow \mathsf{C}$ with $\mathsf{C} \simeq_{\mathcal{B}} \mathsf{B}$ and no base type equivalent to B occuring in $\boldsymbol{U}$.

SBT applies to constructor systems only. By using semantic labelling, we will prove that it can also be applied to some non-constructor systems.

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 [12]. 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 by the fixpoint $[\![\mathsf{B}]\!]$ of the monotonic function $F_{\mathsf{B}}(X) = \{v \in \mathcal{SN} \mid \forall \text{ constructor } \mathsf{c} : \boldsymbol{T} \Rightarrow \mathsf{B}, \forall \boldsymbol{t}, \forall i \in \mathrm{Ind}(\mathsf{c}), v \to^* \mathsf{c}\boldsymbol{t} \Rightarrow t_i \in [\![T_i]\!]_{\mathsf{B} \mapsto X}\}$ on the lattice of reducibility candidates that is complete for set inclusion [6]. This fixpoint, defined by induction on the well-founded quasi-ordering $\leq_{\mathcal{B}}$ on base types, can 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 $\mathfrak{a}$ transfinite iterations of F_{B}.

This notion of size, which corresponds to the tree height for patterns, has the following properties: it is well-founded; the size of a pattern is strictly bigger than the size of its subterms; if $t \to t'$ then the size of t' is smaller than (since $\to$ may be non confluent) or equal to the size of t.

SBT 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 its arguments decreases 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 $[\![a]\!]\mu <_{\mathfrak{A}} [\![b]\!]\mu$ for each $\mu : \mathcal{X} \to \mathfrak{A}$.

To denote a size that cannot be expressed in $\mathcal{A}$ (or a size that we do not care about), 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 $\leq_{\overline{\mathcal{A}}} = \leq_{\mathcal{A}} \cup \{(a, \infty) \mid a \in \overline{\mathcal{A}}\}$. Note that 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 $[\![a]\!]\mu$: $[\![\mathsf{B}^a]\!]^\mu = F_{\mathsf{B}}^{[\![a]\!]\mu}(\bot)$.

Hence, we assume that every symbol $\mathsf{f} \in \mathcal{F}$ is given an annotated type $\tau_{\mathsf{f}}^{\mathcal{A}}$ whose size variables, like type variables in ML, are implicitly universally quantified and can be instantiated by any size expression. Hence the typing rule for symbols in Figure 1 allows any size substitution φ to be applied to $\tau_{\mathsf{f}}^{\mathcal{A}}$. Subtyping naturally follows from the interpretation of types and the ordering on $\mathcal{A}$.

Fig. 1. Type system with size annotations

$$\frac{\varphi : \mathcal{X} \to \mathcal{A}}{\Gamma \vdash^s \mathsf{f} : \tau_{\mathsf{f}}^{\mathcal{A}}\varphi} \qquad \frac{(x, T) \in \Gamma}{\Gamma \vdash^s x : T} \qquad \frac{\Gamma, x : T \vdash^s u : U \quad x \notin \Gamma}{\Gamma \vdash^s \lambda x^T u : T \Rightarrow U}$$

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

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

Definition 1. *Given a type T, let T^∞ be the type obtained by annotating every base type with ∞, and $\mathrm{annot}_{\mathsf{B}}^{\alpha}(T)$ be the type obtained by annotating every base type $\mathsf{C} \simeq_{\mathcal{B}} \mathsf{B}$ with α, and every base type $\mathsf{C} \not\simeq_{\mathcal{B}} \mathsf{B}$ with ∞. Conversely, given an annotated type T, let $|T|$ be the type obtained by removing all annotations.*

Note that, in constrast to types, terms are unchanged: in $\lambda x^T u$, $T = T^\infty$.

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 monotonic (resp. anti-monotonic) in its i-th argument. The sets of *positive* and *negative* positions in an annotated type are:

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

To ease the expression of termination conditions, for every defined symbol f, $\tau_{\mathsf{f}}^{\mathcal{A}}$ is assumed to be of the form $\boldsymbol{P} \Rightarrow \mathbf{B}^{\boldsymbol{\alpha}_{\mathsf{f}}} \Rightarrow \mathsf{B}^{\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha}_{\mathsf{f}})}$ with $|\tau_{\mathsf{f}}^{\mathcal{A}}| = \tau_{\mathsf{f}}$, $\mathcal{X}(\boldsymbol{P}) = \emptyset$ and $\mathcal{X}(\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha}_{\mathsf{f}})) \subseteq \{\boldsymbol{\alpha}_{\mathsf{f}}\}$ where $\boldsymbol{\alpha}_{\mathsf{f}}$ are pairwise distinct variables. 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 $|\boldsymbol{l}| = |\mathbf{B}|$.

Moreover, the annotated type of a constructor $\mathsf{c} : T_1 \dots T_n \Rightarrow \mathsf{B}$ is:

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

with $\mathsf{c}^{\mathcal{A}}(\alpha) = \infty$ if c is non-recursive, and $\mathsf{c}^{\mathcal{A}}(\alpha) = \mathsf{s}(\alpha)$ otherwise, where s is a monotonic unary symbol interpreted as the ordinal successor and such that $a <_{\mathcal{A}} \mathsf{s}(a)$ for each a.

Termination criterion. We assume given a well-founded quasi-ordering $\geq_{\mathcal{F}}$ 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 each 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 each θ, if $\boldsymbol{p}\theta \in [\![\boldsymbol{P}]\!]$ and $\boldsymbol{l}\theta \in [\![\mathbf{B}]\!]$ then there is ν such that, for each $(x, T) \in \Gamma$, $x\theta \in [\![T]\!]^{\nu}$ and $[\![\boldsymbol{a}]\!]\nu \leq o_{\mathbf{B}}(\boldsymbol{l}\theta)$;*
- *argument decreasingness: $\Gamma \vdash_{\mathsf{f}\boldsymbol{a}}^{s} r : \mathsf{B}^{\mathsf{f}^{\mathcal{A}}(\boldsymbol{a})}$ where $\vdash_{\mathsf{f}\boldsymbol{a}}$ is defined in Figure 2;*
- *size annotations monotonicity:* $\mathrm{Pos}(\boldsymbol{\alpha}, \mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})) \subseteq \mathrm{Pos}^{+}(\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha}))$.

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 the instantiation of the variables of $\boldsymbol{l}$ is, and thus that any recursive call with arguments of size smaller than $\boldsymbol{a}$ is admissible. The existence of such a valid syntactic representation depends on $\boldsymbol{l}$ and the size annotations of constructors. With the chosen annotations, the condition is not satisfied by some patterns (whose type admits elements of size bigger than ω, Appendix A). This suggests to use a more precise annotation for constructors.

The expressive power of the criterion depends on $\mathcal{A}$. Taking the size algebra $\mathcal{A}$ reduced to the successor symbol s (the decidability of which is proved in [3]) is sufficient to handle 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 $\mathsf{0} : \mathsf{O}$, $\mathsf{s} : \mathsf{O}^{\alpha} \Rightarrow \mathsf{O}^{\mathsf{s}\alpha}$ 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 $\mathsf{0} : \mathsf{N}$ and $\mathsf{s} : \mathsf{N}^{\alpha} \Rightarrow \mathsf{N}^{\mathsf{s}\alpha}$:

Fig. 2. Computability closure

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

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

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

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 $- : \mathsf{N}^{\alpha} \Rightarrow \mathsf{N} \Rightarrow \mathsf{N}^{\alpha}$.

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

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 a max symbol

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

- $\mathsf{0} \in \Sigma_0$ 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:

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

with $\boldsymbol{\alpha}$ distinct variables, $\mathsf{c}_{\mathcal{A}}(\boldsymbol{\alpha}) = \mathsf{0}$ 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 prove that constructors are computable: $\mathsf{c} \in [\![\tau^{\mathcal{A}}_{\mathsf{c}}]\!]^{\mu}$ for each μ.

Moreover, now, both constructors and defined symbols have a type of the form $annot^{\alpha_1}_{\mathsf{B}_1}(T_1) \Rightarrow \dots \Rightarrow annot^{\alpha_n}_{\mathsf{B}_n}(T_n) \Rightarrow \mathsf{B}^{\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})}$ 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 postpone subtyping after typing 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 (see Appendix B).

Moreover, the pattern and monotonicity conditions can always be satisfied by defining, for each symbol $\mathsf{f} :^s \boldsymbol{P} \Rightarrow \mathsf{B}^{\boldsymbol{\alpha}} \Rightarrow U$ and rule $\mathsf{f}\boldsymbol{pl} \to r \in \mathcal{R}$, $\boldsymbol{a}$ as $\sigma(\boldsymbol{l})$ where $\sigma(x) = x$ and $\sigma(\mathsf{c}\boldsymbol{t}) = \mathsf{c}^{\mathcal{A}}(\sigma(\boldsymbol{t}))$, and 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 $x = l_i$,
- $annot^x_{\mathsf{B}_i}(T)$ if $\mathsf{c}\boldsymbol{u}x\boldsymbol{v}$ is a subterm of l_i and $\mathsf{c} : \boldsymbol{U} \Rightarrow T \Rightarrow \boldsymbol{V} \Rightarrow \mathsf{C}$.

Note that, if $\Gamma \vdash t : T$ and t is a non-variable pattern then there is a base type B such that $\Gamma \vdash^i t : \mathsf{B}^{\sigma(t)}$. So, $\sigma(t)$ is the most general size of t.

Theorem 2. *Let $\mathcal{R}$ be a constructor system. The relation $\to_\beta \cup \to_{\mathcal{R}}$ terminates if, for each $\mathsf{f} :^s \boldsymbol{P} \Rightarrow \mathsf{B}^{\boldsymbol{\alpha}} \Rightarrow \mathsf{B}^{\mathsf{f}^{\mathcal{A}}(\boldsymbol{\alpha})}$ 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 $\vdash^i$ [3] (see Appendix B) with function applications restricted as in Figure 2.*

The proof is given in Appendix C. In the following, we 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 [22]. 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: $lab^\mu(x) = x$ and $lab^\mu(\mathsf{f}(t_1, \ldots, t_n)) = \mathsf{f}_{\pi_{\mathsf{f}}([\![t_1]\!]\mu, \ldots, [\![t_n]\!]\mu)}(lab^\mu(t_1), \ldots, lab^\mu(t_n))$.

The fundamental theorem of semantic labelling is then:

Theorem 3 ([22]). *Given a rewrite system $\mathcal{R}$, an ordered $\mathcal{F}$-algebra $(\mathcal{M}, \leq_{\mathcal{M}})$ and a labelling system $(S^{\mathsf{f}}, \leq_{\mathsf{f}}, \pi_{\mathsf{f}})_{\mathsf{f} \in \mathcal{F}}$, the relation $\to_{\mathcal{R}}$ terminates if:*

1. *$\mathcal{M}$ is a quasi-model of $\mathcal{R}$, that is:*
   - *for each rule $l \to r \in \mathcal{R}$ and valuation $\mu : \mathcal{X} \to M$, $[\![l]\!]\mu \geq_{\mathcal{M}} [\![r]\!]\mu$,*
   - *for each $\mathsf{f} \in \mathcal{F}$, $\mathsf{f}^{\mathcal{M}}$ is monotonic;*
2. *for each $\mathsf{f} \in \mathcal{F}$, π_{f} is monotonic;*

3. *the relation* $\to_{lab(\mathcal{R})\cup Decr}$ *terminates where:*
$lab(\mathcal{R}) = \{lab^\mu(l) \to lab^\mu(r) \mid l \to r \in \mathcal{R}, \mu : \mathcal{X} \to M\}$,
$Decr = \{\mathsf{f}_a(x_1,\ldots,x_n) \to \mathsf{f}_b(x_1,\ldots,x_n) \mid \mathsf{f} \in \mathcal{F}, a >_\mathsf{f} b\}$.

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 semantics 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 may have already noticed some similarity between semantic labelling and size annotations. We here render it more explicit by giving a new proof of the correctness of SB-termination using semantic labelling.

In the first-order case, the interpretation of a base type does not require transfinite iteration: all sizes are smaller than ω and $\mathfrak{A} = \mathbb{N}$ [6]. Moreover, by taking $\Gamma(x) = \mathsf{B}^x$ for each x of type B, every term t has a most general size $\sigma(t)$ given by its most general type: $\Gamma \vdash^i t : \mathsf{C}^{\sigma(t)}$. 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 each defined symbol f.

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

- $\mathcal{R}$ *is finitely branching and the set of constructors of each type* B *is finite;*
- *for each defined symbol* f, $\mathsf{f}^\mathcal{A}$ *and* $\zeta^\mathcal{A}_\mathsf{f}$ *are monotonic.*

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 $\mathsf{f}^\mathcal{M}(\boldsymbol{a}) = [\![\mathsf{f}^\mathcal{A}(\boldsymbol{\alpha})]\!]\mu$ where $\boldsymbol{\alpha}\mu = \boldsymbol{a}$.

When $\mathsf{f}^\mathcal{A} = \infty$, we proceed in a way similar to *predictive labelling* [15], 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.

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

- For each 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 each 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 patterns and constructors are interpreted by monotonic 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 monotonic wrt μ and the height of $\boldsymbol{l}$. We cannot have an infinite set of $\boldsymbol{l}$'s of bounded height since, for each 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 $S^{\mathsf{f}} = D^{\mathfrak{A}}_{\mathsf{f}}$ which is well-founded wrt $>_{\mathsf{f}}$, and $\pi_{\mathsf{f}} = \zeta^{\mathfrak{A}}_{\mathsf{f}}$.

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 $l = \mathsf{f}\boldsymbol{p}\boldsymbol{l}$, 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 $\mathsf{f}^{\mathcal{M}}(\mathfrak{a}) = max(\{0\} \cup \{[\![r]\!]\mu \mid \mathsf{f}\boldsymbol{l} \to r \in \mathcal{R}, \mu : \mathcal{X} \to \mathfrak{A}, [\![\boldsymbol{l}]\!]\mu \leq \mathfrak{a}\})$ 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, for each i, $a_i \neq \infty$ ($\boldsymbol{a} \neq \infty$ for short). 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 monotonic. If f is a recursive constructor, then $\mathsf{f}^{\mathcal{M}}(\mathfrak{a}) = sup\{\mathfrak{a}_i \mid i \in \mathrm{Ind}(\mathsf{c})\} + 1$ is monotonic. 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 monotonic since $\mathsf{f}^{\mathcal{A}}$ is monotonic 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 monotonic.
2. If f is a defined symbol, then the function π_{f} is monotonic by assumption. If f is a constructor, then the constant function π_{f} is monotonic too.
3. We now prove that $\to_{lab(\mathcal{R}) \cup Decr}$ is precedence-terminating (PT), *i.e.* there is a well-founded relation $>$ on symbols such that, for each rule $\mathsf{f}\boldsymbol{l} \to r \in lab(\mathcal{R}) \cup Decr$, every symbol occurring in r is strictly smaller than f [19].
   Let $\mathsf{g}_a < \mathsf{f}_b$ if $\mathsf{g} <_{\mathcal{F}} \mathsf{f}$ or $\mathsf{g} \simeq_{\mathcal{F}} \mathsf{f}$ and $a <^{\mathfrak{A}}_{\mathsf{f}} b$. The relation $>$ is well-founded since both $>_{\mathcal{F}}$ and $>^{\mathfrak{A}}_{\mathsf{f}}$ are well-founded.
   Decr is clearly PT wrt $>$. Let now $\mathsf{f}\boldsymbol{l} \to r \in \mathcal{R}$, $\mu : \mathcal{X} \to M$ and $\mathsf{g}\boldsymbol{t}$ be a subterm of r. The label of f is $a = \pi_{\mathsf{f}}([\![\boldsymbol{l}]\!]\mu) = \zeta^{\mathfrak{A}}_{\mathsf{f}}([\![\sigma(\boldsymbol{l})]\!]\mu)$ and the label of g is $b = \zeta^{\mathfrak{A}}_{\mathsf{f}}([\![\sigma(\boldsymbol{m})]\!]\mu)$. By assumption, $(\mathsf{f}, \boldsymbol{l}) >^{\mathcal{A}} (\mathsf{g}, \boldsymbol{m})$. Therefore, $a >^{\mathfrak{A}}_{\mathsf{f}} b$. □

It is interesting to note that we could also have taken $M = \mathcal{A}$, assuming that $<^{\mathcal{A}}_{\mathsf{f}}$ is stable by substitution ($\zeta^{\mathcal{A}}_{\mathsf{f}}(\boldsymbol{a}\theta) <^{\mathcal{A}}_{\mathsf{f}} \zeta^{\mathcal{A}}_{\mathsf{f}}(\boldsymbol{b}\theta)$ whenever $\zeta^{\mathcal{A}}_{\mathsf{f}}(\boldsymbol{a}) <^{\mathcal{A}}_{\mathsf{f}} \zeta^{\mathcal{A}}_{\mathsf{f}}(\boldsymbol{b})$). 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 provides a *finite* representation of the infinite $\mathfrak{A}$-labelled system.

Finally, we see from the proof that the system does not need to be constructor:

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

– *$\mathsf{g}^{\mathcal{A}}$ is monotonic 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 $\zeta^{\mathcal{A}}_{\mathsf{f}}(\sigma(\boldsymbol{m})) <^{\mathcal{A}}_{\mathsf{f}} \zeta^{\mathcal{A}}_{\mathsf{f}}(\sigma(\boldsymbol{l}))$.*

Example: assuming that A is the $\Rightarrow$-type constructor, then the expression $\mathsf{F}nuv$ represents the set of n-ary functions from u to v.

$$\begin{array}{rcl@{\qquad}rcl} +\mathsf{0}y & \to & y & \mathsf{F0}uv & \to & v \\ +(\mathsf{s}x)y & \to & \mathsf{s}(+xy) & \mathsf{F}(\mathsf{s}x)uv & \to & \mathsf{A}u(\mathsf{F}xuv) \\ +(+xy)z & \to & +x(+yz) & \mathsf{F}(+xy)uv & \to & \mathsf{F}xu(\mathsf{F}yuv) \end{array}$$

Take $+^{\mathcal{A}}(x,y) = \zeta_+(x,y) = a = 2x+y+1$, $\mathsf{F}^{\mathcal{A}} = \infty$ and $\zeta_{\mathsf{F}}(x,u,v) = x$. The interpretation of $\mathsf{F}^{\mathcal{M}}$ is well-defined since $x < a$ and $y < a$. The labelled system that we obtain (where $b = 2y+z+1$) is precedence-terminating:

$$\begin{array}{rcl@{\qquad}rcl} +_{y+1}\mathsf{0}y & \to & y & \mathsf{F}_0\mathsf{0}uv & \to & v \\ +_{a+2}(\mathsf{s}x)y & \to & \mathsf{s}(+_a xy) & \mathsf{F}_{x+1}(\mathsf{s}x)uv & \to & \mathsf{A}u(\mathsf{F}_x xuv) \\ +_{2a+z+1}(+_a xy)z & \to & +_{2x+b+1}x(+_b yz) & \mathsf{F}_a(+_a xy)uv & \to & \mathsf{F}_x xu(\mathsf{F}_y yuv) \end{array}$$

7 Higher-order semantic labelling

Semantic labelling was extended by Hamana [13] 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) [17] whose categorical semantics based on binding algebras and $\mathcal{F}$-monoids [10] is studied by the same author and proved complete for termination [14].

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* extend 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 [20].

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 [10] is to interpret variables by natural numbers using De Bruijn levels , 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 each $\rho : \Gamma \to \Delta$, $\alpha_B(\Delta) \circ M_B(\rho) = N_B(\rho) \circ \alpha_B(\Gamma)$.

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 [13] (see Appendix E).

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 [14]. Hence, for each $\mathcal{F}$-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 $\overline{\mathcal{F}}_n = \{\mathsf{f}_a \mid \mathsf{f} \in \mathcal{F}_n, a \in S^{\mathsf{f}}\}$. Note that the set of labelled meta-terms has a structure of $\mathcal{F}$-monoid [13].

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

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

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

Theorem 6 ([13]). *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 each $l \to r : T \in \mathcal{R}$, $\theta : \mathcal{Z} \to I^\emptyset$ and Γ, $!_B^{\mathcal{M}}(\Gamma)(l\theta) \geq_{M_B(\Gamma)} !_B^{\mathcal{M}}(\Gamma)(r\theta)$,*
   - *for each $\mathsf{f} \in \mathcal{F}$, $\mathsf{f}^{\mathcal{M}}$ is monotonic;*
2. *for each $\mathsf{f} \in \mathcal{F}$, π_{f} is monotonic;*
3. *the relation $\to_{lab(\mathcal{R}) \cup Decr}$ terminates, where:*
   $lab(\mathcal{R}) = \{lab_B^\emptyset(\Gamma)(l\theta) \to lab_B^\emptyset(\Gamma)(r\theta) \mid l \to r : B \in \mathcal{R}, \theta : \mathcal{Z} \to I^\emptyset, \Gamma \in \mathbb{E}\}$,
   $Decr = \{\mathsf{f}_a(\ldots, \lambda \boldsymbol{x}_i Z_i(\boldsymbol{x}_i), \ldots) \to \mathsf{f}_b(\ldots, \lambda \boldsymbol{x}_i Z_i(\boldsymbol{x}_i), \ldots) \mid \mathsf{f} \in \mathcal{F}, a >_{\mathsf{f}} b\}$.

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 [13], 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 $\lambda_T^U : (T \Rightarrow U) \Rightarrow Arr(T, U)$ and $@_T^U : Arr(T, U) \Rightarrow T \Rightarrow U$ such that T and U are IDTS base types. Note that only λ_T^U 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:

- $\langle x \rangle_\Gamma = x$,
- $\langle \lambda x^T u \rangle_\Gamma = \lambda_{\langle T \rangle}^{\langle U \rangle}(\lambda x \langle u \rangle_{\Gamma, x:T})$ if $\Gamma, x : T \vdash u : U$,
- for $\mathsf{f} : T_1 \Rightarrow \ldots \Rightarrow T_n \Rightarrow \mathsf{B}$ and $U_i = T_{i+1} \Rightarrow \ldots \Rightarrow T_n \Rightarrow \mathsf{B}$,
  $\langle \mathsf{f} t_1 \ldots t_k \rangle_\Gamma = \lambda_{\langle T_{k+1} \rangle}^{\langle U_{k+1} \rangle}(\lambda x_{k+1} \ldots \lambda_{\langle T_n \rangle}^{\langle U_n \rangle}(\lambda x_n \langle \mathsf{f} \rangle(\langle t_1 \rangle_\Gamma, \ldots, \langle t_k \rangle_\Gamma, x_{k+1}, \ldots, x_n))...)$,
- $\langle tu \rangle_\Gamma = @_{\langle U \rangle}^{\langle V \rangle}(\langle t \rangle_\Gamma, \langle u \rangle_\Gamma)$ 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 $\langle \beta \rangle = \bigcup_{T,U \in \mathbb{B}} \beta_T^U$ where β_T^U is:

$$@_T^U(\lambda_T^U(\lambda x Z(x)), X) \to Z(X)$$

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 (Appendix F).

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 monotonic 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 each function f. We therefore need to restrict $N_{Arr(T,U)}$ to the functions that correspond to (are 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 now check that $sup\{f(n) \mid n \in N_{\mathsf{N}}\} + 1 \in N_{\mathsf{O}}$. 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 $x_0 \in N_{Arr(\Gamma,T)}$, $x_1 \in N_{\Delta(1)}$, $\ldots$, $x_p \in N_{\Delta(p)}$ 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 each $x \in N_T$.

Interpretation of variables and function symbols. As one can expect, variables are interpreted by projections: $\iota_{\Gamma(i)}(\Gamma)(i)(\boldsymbol{x}) = x_i$, λ_T^U by the identity: $(\lambda_T^U)^{\mathcal{M}}(\Gamma)(f) = f$, and $@_T^U$ by the application: $(@_T^U)^{\mathcal{M}}(\Gamma)(f,x)(\boldsymbol{y}) = f(\boldsymbol{y}, x(\boldsymbol{y}))$.

One can check that these functions are valid interpretations indeed, *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 $\mu_B(\Gamma)(t, u_1 \ldots u_n)(\boldsymbol{x}) = t(u_1(\boldsymbol{x}), \ldots, u_n(\boldsymbol{x}))$.

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 each f and Γ, $\mathsf{f}^{\mathcal{M}}(\Gamma)(\boldsymbol{x})(\boldsymbol{y}) = \mathsf{f}^{\mathcal{M}}(\emptyset)(x_1(\boldsymbol{y}), \ldots, x_n(\boldsymbol{y}))$ (Appendix G).

One can see that $(\lambda_T^U)^{\mathcal{M}}$ and $(@_T^U)^{\mathcal{M}}$ satisfy this property. Moreover, for each term $t \in I_T^{\emptyset}(\Gamma)$, we have $!_T^{\mathcal{M}}(x_1 : T_1 \ldots x_n : T_n)(t)(\boldsymbol{a}) = [\![t]\!]\mu$ where $x_i\mu = a_i$ 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 each 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 each 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, with Brouwer's ordinals, we have $\sigma(\mathsf{lim} f) = \mathsf{lim}^{\mathcal{A}} f >_{\mathcal{A}} \sigma(fn) = fn$.

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

Theorem 7. *SB-termination implies termination if constructors are strictly-positive and the conditions of 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 Appendix G. For $\langle \mathcal{R} \rangle$, we use the facts that $!^{\mathcal{M}}_B(\Gamma)(l\theta) \leq_{M_B(\Gamma)} !^{\mathcal{M}}_B(\Gamma)(r\theta)$ if $!^{\mathcal{M}}_B(\Gamma)(l\theta)(\boldsymbol{a}) \leq_{M_B(\emptyset)} !^{\mathcal{M}}_B(\Gamma)(r\theta)(\boldsymbol{a})$ for each $\boldsymbol{a}$, and that $!^{\mathcal{M}}_B(\Gamma)(l\theta)(\boldsymbol{a}) = [\![l]\!]\theta\mu$ where $x_i\mu = a_i$.

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 $lab(\langle \mathcal{R} \rangle)$, let $(\Gamma, \boldsymbol{x}) >^{\mathcal{R}}_{\mathsf{f}} (\Delta, \boldsymbol{y})$ if $\Delta = \Gamma + \Gamma'$ and, for each $\boldsymbol{z}\boldsymbol{z}'$, $\zeta_{\mathsf{f}}(\ldots x_i(\boldsymbol{z}) \ldots) >^{\mathfrak{A}}_{\mathsf{f}} \zeta_{\mathsf{f}}(\ldots y_i(\boldsymbol{z}\boldsymbol{z}') \ldots)$. For dealing with $lab(\langle \beta \rangle)$, let $(\Gamma, \boldsymbol{x}) >^{\beta}_{\mathsf{f}} (\Delta, \boldsymbol{y})$ if $\Gamma = \Delta + T$ and there is e such that, for each i and $\boldsymbol{z}$, $x_i(\boldsymbol{z}, e(\boldsymbol{z})) = y_i(\boldsymbol{z})$. Since $>^{\mathcal{R}}_{\mathsf{f}} \circ >^{\beta}_{\mathsf{f}}$ is included in $>^{\mathcal{R}}_{\mathsf{f}} \cup >^{\beta}_{\mathsf{f}} \circ >^{\mathcal{R}}_{\mathsf{f}}$, the relation $>_{\mathsf{f}} = >^{\mathcal{R}}_{\mathsf{f}} \cup >^{\beta}_{\mathsf{f}}$ is well-founded [9].

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

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 $@^U_T(\lambda^U_T(\lambda x lab_U(\Gamma + T)(u)), lab_T(\Gamma)(t)) \rightarrow lab_U(\Gamma)(u^t_x) \in lab(\langle \beta \rangle)$, a symbol f occuring in u is labelled in $lab_U(\Gamma + T)(u)$ by something like $(\Gamma + T + \Delta, !^{\mathcal{M}}_{\boldsymbol{B}}(\Gamma + T + \Delta)(\boldsymbol{v}))$, and by something like $(\Gamma + \Delta, !^{\mathcal{M}}_{\boldsymbol{B}}(\Gamma + \Delta)(\boldsymbol{v}^t_x))$ in $lab_U(\Gamma)(u^t_x)$. 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^U_T (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 (Appendix F). 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* (we do not need the results on *solid* IDTSs [13]). 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 $a >_{\mathsf{f}} b$.

Conclusion. By studying the relationship between sized-types based termination and semantic labelling, we arrived at a new way to prove the correctness of SBT that enabled us to extend it to non-constructor systems, *i.e.* systems with matching on defined symbols (*e.g.* associative symbols, Appendix D). This work can be carried on in various directions by considering: richer type structures with polymorphic or dependent types, non-strictly positive constructors, or the inference of size annotations to automate SBT.

Acknowledgments. The authors want to thank very much Colin Riba and Andreas Abel for their useful remarks on a previous version of this paper. This work was partly supported by the Bayerisch-Französisches Hochschulzentrum.

References

1. A. Abel. Semi-continuous sized types and termination. *Logical Methods in Computer Science*, 4(2):1–33, 2008.
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 and C. Riba. Combining typing and size constraints for checking the termination of higher-order conditional rewrite systems. In *Proc. of LPAR'06*.
8. F. Blanqui and C. Roux. On the relation between sized-types based termination and semantic labelling (full version). `www-rocq.inria.fr/~blanqui/`, 2009.
9. H. Doornbos and B. von Karger. On the union of well-founded relations. *Logic Journal of the IGPL*, 6(2):195–201, 1998.
10. M. Fiore, G. Plotkin, and D. Turi. Abstract syntax and variable binding. In *Proc. of LICS'99*.
11. E. Giménez. *Un Calcul de Constructions infinies et son application à la vérification de systèmes communiquants*. PhD thesis, ENS Lyon, France, 1996.
12. J.-Y. Girard. *Interprétation fonctionelle et élimination des coupures dans l'arithmetique d'ordre supérieur*. PhD thesis, Université Paris VII, France, 1972.
13. M. Hamana. Higher-order semantic labelling for inductive datatype systems. In *Proc. of PPDP'07*.
14. M. Hamana. Universal algebra for termination of higher-order rewriting. In *Proc. of RTA'05*, LNCS 3467.
15. N. Hirokawa and A. Middeldorp. Predictive labeling. In *Proc. of RTA'06*.
16. J. Hughes, L. Pareto, and A. Sabry. Proving the correctness of reactive systems using sized types. In *Proc. of POPL'96*.
17. J. W. Klop, V. van Oostrom, and F. van Raamsdonk. Combinatory reduction systems. *Theoretical Computer Science*, 121:279–308, 1993.
18. N. P. Mendler. *Inductive Definition in Type Theory*. PhD thesis, Cornell University, United States, 1987.
19. A. Middeldorp, H. Ohsaki, and H. Zantema. Transforming termination by self-labelling. In *Proc. of CADE'96*, LNCS 1104.
20. D. Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. In *Proc. of ELP'89*, LNCS 475.
21. H. Xi. Dependent types for program termination verification. In *Proc. of LICS'01*.
22. H. Zantema. Termination of term rewriting by semantic labelling. *Fundamenta Informaticae*, 24:89–105, 1995.