Strong Normalization through Intersection Types and Memory

Antonio Bucciarelli Delia Kesner

Univ Paris Diderot, Sorbonne Paris Cité, PPS, CNRS, Paris, France

Daniel Ventura

Univ. Federal de Goiás, Instituto de Informática, Goiânia, Brazil

Abstract

We characterize β-strongly normalizing λ-terms by means of a non-idempotent intersection type system. More precisely, we first define a memory calculus $\mathtt{K}$ together with a non-idempotent intersection type system $\mathcal{K}$, and we show that a $\mathtt{K}$-term t is typable in $\mathcal{K}$ if and only if t is $\mathtt{K}$-strongly normalizing. We then show that β-strong normalization is equivalent to $\mathtt{K}$-strong normalization. We conclude since λ-terms are strictly included in $\mathtt{K}$-terms.

Keywords: Lambda-calculus, memory calculus, strong normalization, intersection types.

1 Introduction

It is well known that the β-strongly normalizing λ-terms can be characterized as those being typable in suitable intersection type (IT) systems. This result dates back to the late 1970s and early 1980s, when intersection types were invented to endow the pure lambda calculus with powerful type-assignment systems [2,12,28,26]. A survey of these results, out of the scope of this paper, can be found for instance in [35,3].

In more recent years, a revisitation of those early results has been driven by the introduction of resource aware semantics of λ-calculi [21,6,15,7] and the corresponding *non-idempotent* intersection types assignment systems. The inhabitation problem for instance, known to be undecidable in an idempotent setting [32], was proved to be decidable for non-idempotent types [8].

Just like their idempotent precursors, these type systems allow for a characterization of strong normalization [5,14] (as well as weak normalization and head normalization [15,9]), but they also grant a substantial improvement: proving that *typable terms are strongly normalizing* becomes much simpler. Let us provide a brief account of this improvement, by highlighting in the way the *quantitative* character

of non-idempotent intersection types versus the *qualitative* flavor of the idempotent ones. The proof of the highlighted statement above, in the non-idempotent case, goes roughly as follows: given a typing derivation for a term t, and willing to prove that t is strongly normalizing, take whatever β-reduct t' of t. The subject reduction lemma, in this case, ensures *not only* that t' is typable *but also* that there exists a typing derivation for t' whose size is smaller than the one of the typing derivation for t we started from. Hence any β-reduction sequence starting from t is finite.

This shrinking of the size of typing derivations along reduction sequences, in sharp contrast to what happens in the idempotent setting, is essentially due to the fact that a type derivation for a term of the shape $(\lambda x.u)v$ may require as many sub-derivations for v as the number of occurrences of x in u[1]. Let us provide a simple example involving the Church numeral $\underline{n} := \lambda y.\lambda x.y(y(...yx)...)$.

Why is the term $u = \lambda x.t(t(...tx)...)$, t being an arbitrarily complex term, "simpler to type" than its β-expanded form $\underline{n}t$? The point is that the typical non-idempotent intersection type[2] that can be assigned to the Church numeral $\underline{n}$ is, in our notation, $[[\sigma] \to \sigma, ..., [\sigma] \to \sigma] \to [\sigma] \to \sigma$, the leftmost multiset containing n copies of $[\sigma] \to \sigma$. Thus, in order to assign a type to $\underline{n}t$, n typing derivations assigning $[\sigma] \to \sigma$ to t must be provided, exactly like in a type derivation for u. At the same time, the outermost application $\underline{n}t$ vanishes with the reduction $\underline{n}t \to_\beta u$, so the typing derivation for u is smaller than that for $\underline{n}t$.

In the idempotent case, on the other hand, a type[3] for $\underline{n}$ is an instance of $\{\{\sigma\} \to \sigma\} \to \{\sigma\} \to \sigma$, and the typing derivations for u may be hugely bigger than those for $\underline{n}t$, the former requiring n sub-derivations for t, the latter just one. That's why, for idempotent intersection type systems, the proof of the result above cannot be *combinatorial*, and is typically based on the *reducibility* argument [31,17,25].

This shift of perspective goes beyond lowering the logical complexity of the proof: the quantitative information provided by typing derivations in the non-idempotent setting unveils interesting relations between typings (static) and reductions (dynamic) of λ-terms. For instance, in [15], a correspondence between the size of a typing derivation for t and the number of steps taken by the Krivine machine to reduce t is presented, and in [5] it is shown how to compute the length of the longest β-reduction sequence starting from any typable strongly normalizing λ-term.

In this paper, we provide a characterization of strongly normalizing λ-terms via a typing system based on non-idempotent intersection types. The structure of the proof is the following:

- We define the K-calculus, reminiscent of Klop's I-calculus [24], where terms are defined by enriching λ-terms with a *memory* operator, β-reduction is split into two different non-erasing reductions, and terms are considered modulo an equivalence relation, reminiscent of Regnier's σ-reduction [29]. In contrast to [24], λ-terms are strictly included in K-terms, which makes our development much easier.
- We introduce the typing system $\mathcal{K}$ for K-terms, based on system $\mathcal{Q}$ for focused intuitionistic logic [18], and we show that a K-term is $\mathcal{K}$-typable if and only if it is

[1] More precisely, it requires exactly as many sub-derivations for v as the number of *typed* occurrences of x in u.

[2] Non-idempotent intersections are denoted by multisets, *e.g.* $[\tau, \tau, \sigma]$ stands for $\tau \wedge \tau \wedge \sigma$.

[3] Idempotent intersections are denoted by sets, *e.g.* $\{\tau, \sigma\}$ stands for $\tau \wedge \sigma$.

K-strongly normalizable. This proof is only based on typing properties of Subject Reduction and Subject Expansion, and does not use any reducibility argument.

- We prove that λ-terms are K-strongly normalizable if and only if they are β-strongly normalizable in the λ-calculus.

Related works: It is only discussed here different approaches to prove strong normalization of λ-calculus by means of intersection types. Several characterizations of strong normalization via *idempotent* intersection types have been presented for the λ-calculus; a survey can be found for example in [3]. To the best of our knowledge, two characterizations of strong normalization via *non-idempotent* intersection types have been presented so far for the λ-calculus, by A. Bernadet and S. Lengrand [5] and by E. De Benedetti and S. Ronchi Della Rocca [14,13], respectively. Non-idempotent intersection is also used in the systems of [22,16], both for the λ-calculus, but characterization of strong normalization is achieved through a relation to an idempotent intersection type system.

In [5], a subtyping relation is introduced to get the subject reduction property, but the system types unnecessary instances of arguments, and turns out to be non relevant, *i.e.* some sort of weakening is allowed. Notions of *optimality* and *principality*[4] of typing derivations are used to derive an exact upper bound for reduction steps. Besides, the "strong normalizing implies typability" property is obtained in [5] through a subject expansion property on a restricted version of the β-reduction, where a memory set is used to trace the free variables of erased terms. In our memory calculus, which is non erasing, both the subject reduction and the subject expansion properties hold unconditionally.

In [14] the typing rule for term variables is weakened, thus the system is non relevant. The characterization of strong normalization, more precisely the "strong normalization implies typability" property, follows from an adaptation of the perpetuality proof in [27]. In [13] another proof of the same property is obtained through an inductive definition of the set of strongly normalizing terms, as done in [19] and in the present work.

In the extended framework of the λ-calculus with explicit substitution, non-idempotent types were also used to characterize strong normalization [5,19]. In [5] the typing system deals with two explicit substitution calculi based on the *structural propagation* paradigm, while in [19] the *substitution at a distance* paradigm is investigated. In all the cases, the normalization property is proved by relying on the postponement of erasing steps, where the explicit substitution operator plays the role of a memory device.

Regarding the *intuitionistic sequent calculus*, Kikuchi [23] refines Valentini's system [33] to yield an idempotent IT system which characterizes strong normalization. These ideas give rise to a non-idempotent intersection system for a computational interpretation of the *focused* intuitionistic calculus [20].

Structure of the paper: Section 2 presents the syntax and semantics of the K-calculus, while Section 3 introduces the non-idempotent typing system $\mathcal{K}$ for K-terms together with its properties. The characterization of β and K-strongly normalizing terms is developed in Section 4. We conclude in Section 5.

[4] The notion of *principal typing* in [5] is different from the usual definition in the literature, *e.g.* [30,37,22].

2 The memory calculus

We are going to characterize the set of strongly normalizing λ-terms by using a memory calculus called K-calculus – reminiscent of Klop's I-calculus [24] – as main technical tool. The point of the memory device in [24] is to obtain a calculus in which strong and weak normalization coincide, which is possible when no information can be lost along reduction sequences [5]. This section introduces the syntax and the operational semantics of the K-calculus, which also uses a memory device to avoid loss of information.

Given a countable infinite set of symbols $x, y, z, \ldots$ we define the set of K-terms by means of the following grammar.

$$t, u, v ::= x \mid \lambda x.t \mid tu \mid t[\![u]\!]$$

The syntactic item $[\![u]\!]$ is called a **memory operator**. Notice that the set of λ-terms is strictly included in the set of K-terms.

The **size** of a term t, written $|t|$, is defined by $|x| := 1$, $|\lambda x.u| := |u| + 1$, $|uv| := |u| + |v| + 1$ and $|t[\![u]\!]| := |t| + |u| + 1$. The notions of **free** and **bound** variables are defined as usual, in particular, $\mathtt{fv}(t[\![u]\!]) := \mathtt{fv}(t) \cup \mathtt{fv}(u)$, $\mathtt{bv}(t[\![u]\!]) := \mathtt{bv}(t) \cup \mathtt{bv}(u)$. We work with the standard notion of **α-conversion** *i.e.* renaming of bound variables. **Substitutions** are (finite) functions from variables to terms. We use the notation $\{x_1/u_1, \ldots, x_n/u_n\}$ $(n \geq 0)$ for a finite substitution ϕ such that $\phi(x_i) = u_i$ for $1 \leq i \leq n$. **Application** of the **substitution** ϕ to the term t, written $t\phi$, may require α-conversions in order to avoid the capture of free variables. Hence we follow the common practice of considering terms up to α-equivalence. However, we feel free to represent α-equivalence classes by any of their members, provided they respect the usual Barendregt's convention [1] stipulating that the sets of free and bound variables of any term are disjoint.

The standard notion of **β-reduction** on λ-terms, written $\to_\beta$, is generated by the closure by contexts of the rewriting rule $(\lambda x.t)u \mapsto_\beta t\{x/u\}$. In other words, the rule β is compatible with the structure of λ-terms. Here is an example of β-reduction:

$$(\lambda x.\lambda y.y)x'y' \to_\beta (\lambda y.y)y' \to_\beta y'$$

We now consider the following equation and rewriting rules on K-terms.

Equation:		**Rules:**	
$t[\![u]\!]v$	$=_\sigma (tv)[\![u]\!]$	$(\lambda x.t)u \mapsto_{\mathtt{neb}} t\{x/u\}$	if $x \in \mathtt{fv}(t)$
		$(\lambda x.t)u \mapsto_{\mathtt{m}} t[\![u]\!]$	if $x \notin \mathtt{fv}(t)$

The names neb and m mean, respectively, non-erasing beta and memory. The reduction relation $\to_{\mathtt{neb},\mathtt{m}}$ is generated by the closure by contexts of the rewriting rules $\mapsto_{\mathtt{neb}}$ and $\mapsto_{\mathtt{m}}$. The relation $\sim_\sigma$, inspired by the σ-equivalence [29] used to identify some permutation of redexes, is the equivalence relation on K-terms generated by

[5] For instance, there is no loss of information in the λ-I-calculus as opposed to the full λ-calculus.

the equation $=_\sigma$ above. Two σ-equivalent terms are undistinguishable in many respects, as for instance the length of reduction sequences starting at them, and the size of their typing derivations.

The $\mathtt{K}$-calculus is given by the set of $\mathtt{K}$-terms and the **reduction relation** $\to_{\mathtt{K}}$ on $\mathtt{K}$-terms, generated by the reduction $\to_{\mathtt{neb},\mathtt{m}}$ modulo the equivalence $\sim_\sigma$. Thus for example

$$(\lambda x.\lambda y.y)x'y' \to_{\mathtt{K}} (\lambda y.y)[\![x']\!]y' \to_{\mathtt{K}} y'[\![x']\!]$$

More precisely,

$$(\lambda x.\lambda y.y)x'y' \to_{\mathtt{m}} (\lambda y.y)[\![x']\!]y' \sim_\sigma (\lambda y.y)y'[\![x']\!] \to_{\mathtt{neb}} y'[\![x']\!]$$

Remark that the term $y'[\![x']\!]$ is not $\mathtt{K}$-reducible anymore, *i.e.* it is a **$\mathtt{K}$-normal form**. Another example is given by the following $\mathtt{K}$-reduction sequence which occurs inside a memory operator

$$y[\![(\lambda w.(\lambda x.x)z)[\![z']\!]x']\!] \to_{\mathtt{K}} y[\![(\lambda w.z)[\![z']\!]x']\!] \to_{\mathtt{K}} y[\![z[\![x']\!][\![z']\!]]\!]$$

Given any reduction relation $\mathcal{R}$, a term t is said to be **$\mathcal{R}$-strongly normalizing**, written $t \in \mathcal{SN}(\mathcal{R})$, iff there is no infinite $\mathcal{R}$-reduction sequence starting at t.

3 The Typing System

In this section we introduce a type system for $\mathtt{K}$-terms, called $\mathcal{K}$, whose intersection types, IT for short, are similar to those in [10,11].

Let $\mathcal{A}$ be a countable infinite set of type variables $\alpha, \beta, \gamma, \ldots$. The sets $\mathcal{T}$ of **strict types**, ranged over by $\sigma, \tau, \ldots$, and $\mathcal{U}$ of **multiset types**, ranged over by $\mathcal{M}, \mathcal{M}', \ldots$, are defined by the following grammars:

$$\sigma, \tau, \rho, \gamma ::= \alpha \mid \mathcal{M} \to \sigma \qquad\qquad \mathcal{M} ::= [\sigma_i]_{i \in I},\ I \text{ finite set}$$

Types are *strict* [6], *i.e.* multiset types do not occur on the right-hand sides of arrows. The empty multiset is written $[\,]$. A multiset type should be read as the intersection of the strict types it contains. For instance, the multiset $[\tau, \tau, \sigma]$ stands for $\tau \wedge \tau \wedge \sigma$, where the symbol $\wedge$ is associative, commutative and non-idempotent. Observe however that the commutativity, associativity and non-idempotency of the intersection symbol is granted by the multiset notation: no further equivalence relation on types is needed.

Type assignments, written Γ, Δ, are functions from variables to multiset types, assigning the empty multiset to all but a finite set of variables. The domain of Γ is given by $\mathtt{dom}(\Gamma) := \{x \mid \Gamma(x) \neq [\,]\}$. The **intersection of type assignments**, written $\Gamma + \Delta$, is defined by $(\Gamma + \Delta)(x) := \Gamma(x) + \Delta(x)$, where $+$ denotes multiset union. Hence, $\mathtt{dom}(\Gamma + \Delta) = \mathtt{dom}(\Gamma) \cup \mathtt{dom}(\Delta)$. An example is $\{x{:}[\sigma], y{:}[\tau]\} + \{x{:}[\sigma'], z{:}[\tau']\} = \{x{:}[\sigma, \sigma'], y{:}[\tau], z{:}[\tau']\}$.

When $\mathtt{dom}(\Gamma)$ and $\mathtt{dom}(\Delta)$ are disjoint we write $\Gamma; \Delta$ instead of $\Gamma + \Delta$. We write $x{:}[\sigma_i]_{i \in I}; \Gamma$, even when $I = \emptyset$, for the assignment $(x{:}[\sigma_i]_{i \in I}; \Gamma)(x) = [\sigma_i]_{i \in I}$ and

[6] The terminology is due to S. van Bakel [34].

$(x{:}[\sigma_i]_{i\in I};\Gamma)(y) = \Gamma(y)$ if $y \neq x$. We write $\Gamma \setminus\setminus x$ for the assignment $(\Gamma \setminus\setminus x)(x) = [\,]$ and $(\Gamma \setminus\setminus x)(y) = \Gamma(y)$ if $y \neq x$.

The type assignment system $\mathcal{K}$ for K-terms is defined in Figure 1. Notice that, in contrast to [4,14] the system is syntax directed.

$$\frac{}{x : [\tau] \vdash x{:}\tau}\ (\mathtt{ax}) \qquad \frac{\Gamma \vdash t{:}\tau}{\Gamma \setminus\setminus x \vdash \lambda x.t{:}\Gamma(x) \to \tau}\ (\to_i) \qquad \frac{\Gamma \vdash t{:}\tau \qquad \Delta \vdash s{:}\sigma}{\Gamma + \Delta \vdash t[\![s]\!]{:}\tau}\ (\mathtt{m})$$

$$\frac{\Gamma \vdash t{:}[\sigma_i]_{i\in I} \to \tau \qquad (\Delta_j \vdash u{:}\sigma_j)_{j\in J}}{\Gamma +_{j\in J} \Delta_j \vdash tu{:}\tau}\ (\to_e)$$

$$\text{where} \quad (I = \emptyset \Rightarrow |J| = 1) \quad \text{and} \quad (I \neq \emptyset \Rightarrow J = I)$$

Fig. 1. The intersection type system for the K-calculus

The typing rule $(\to_e)$ could be specified by means of two different typing rules separating the cases $I = \emptyset$ and $I \neq \emptyset$:

$$\frac{\Gamma \vdash t{:}[\,] \to \tau \qquad \Delta \vdash u{:}\sigma}{\Gamma + \Delta \vdash tu{:}\tau} \qquad \frac{\Gamma \vdash t{:}[\sigma_i]_{i\in I} \to \tau \quad I \neq \emptyset \qquad (\Delta_i \vdash u{:}\sigma_i)_{i\in I}}{\Gamma +_{i\in I} \Delta_i \vdash tu{:}\tau}$$

Indeed, if $I = \emptyset$ in rule $(\to_e)$, then the argument u is erasable, so that we require exactly one typing derivation for u (the *typing witness*), by setting $|J| = 1$ (notice that the type σ is ignored in the final conclusion of the rule); otherwise the argument u is not erasable and several typing derivations for u are required, one for each type of the multiset $[\sigma_i]_{i\in I}$, thus $J = I$. We prefer however to capture both cases in the single rule $(\to_e)$ in order to save some space in our proofs.

A **(typing) derivation** in system $\mathcal{K}$ is a tree Φ obtained by applying the (inductive) rules of the typing system. We write $\Phi \triangleright \Gamma \vdash t{:}\tau$ if there is a derivation Φ in system $\mathcal{K}$ ending in the type judgment $\Gamma \vdash t{:}\tau$. A term t is **typable** in system $\mathcal{K}$ iff there exist a derivation Φ, an assignments Γ and a type τ such that $\Phi \triangleright \Gamma \vdash t{:}\tau$. For any typing derivation tree Φ, we define $\mathtt{sz}(\Phi)$ to be the number of nodes of Φ.

It is worth noticing that the rule $(\to_e)$ makes the difference between an intersection type system characterizing head/weak normalization [18,19] and one characterizing strong normalization. Indeed, in the former, the case $I = \emptyset$ would not impose to type the argument u, and one would obtain typing derivations of the form:

$$\frac{\Gamma \vdash t{:}[\,] \to \tau}{\Gamma \vdash tu{:}\tau}$$

where tu is typed but u is untyped. Conversely, in our system, all the subterms of a typable term have to be typable, thus guaranteeing termination for any reduction strategy.

As an example of typing derivation in $\mathcal{K}$, for $\delta = \lambda x.xx$ one has

$$\Phi_\delta := \dfrac{\dfrac{\overline{x : [[\alpha] \to \beta] \vdash x{:}[\alpha] \to \beta} \qquad \overline{x : [\alpha] \vdash x{:}\alpha}}{x : [[\alpha], [\alpha] \to \beta] \vdash xx{:}\beta}}{\vdash \lambda x.xx{:}[[\alpha], [\alpha] \to \beta] \to \beta}$$

hence

$$\dfrac{\dfrac{\overline{z : [\gamma] \vdash z{:}\gamma}}{z : [\gamma] \vdash \lambda y.z{:}[\,] \to \gamma} \qquad \Phi_\delta \triangleright \vdash \delta{:}[[\alpha], [\alpha] \to \beta] \to \beta}{z : [\gamma] \vdash (\lambda y.z)\delta{:}\gamma}$$

and also

$$\dfrac{\overline{z : [\gamma] \vdash z{:}\gamma} \qquad \Phi_\delta \triangleright \vdash \delta{:}[[\alpha], [\alpha] \to \beta] \to \beta}{z : [\gamma] \vdash z[\![\delta]\!]{:}\gamma}$$

On the other hand, neither $(\lambda y.z)\Omega$ nor $z[\![\Omega]\!]$ are typable in $\mathcal{K}$ for $\Omega = \delta\delta$.

The $\mathcal{K}$-system enjoys relevance (absence of weakening).

Lemma 3.1 (Relevance) *If* $\Phi \triangleright \Gamma \vdash t{:}\tau$ *then* $\mathtt{dom}(\Gamma) = \mathtt{fv}(t)$.

Proof. By induction on the derivation of Φ. □

Moreover, the equivalence relation $\sim_\sigma$ does not alter at all the typing relation.

Lemma 3.2 (Typing Invariance for $\sim_\sigma$) *Let* $t_0 \sim_\sigma t_0'$. *Then* $\Phi \triangleright \Gamma \vdash t_0{:}\tau$ *iff* $\Phi' \triangleright \Gamma \vdash t_0'{:}\tau$. *Moreover,* $\mathtt{sz}(\Phi) = \mathtt{sz}(\Phi')$.

Proof. By induction on the proof of $t_0 \sim_\sigma t_0'$. We only show the base case, the others being straightforward.

If $t_0 = t[\![u]\!]v =_\sigma (tv)[\![u]\!] = t_0'$, then by construction $\Gamma = \Delta + \Pi +_{j\in J} \Gamma_j$ and Φ is of the following form:

$$\dfrac{\dfrac{\Phi_t \triangleright \Delta \vdash t{:}[\sigma_i]_{i\in I} \to \tau \qquad \Phi_u \triangleright \Pi \vdash u{:}\rho}{\Delta + \Pi \vdash t[\![u]\!]{:}[\sigma_i]_{i\in I} \to \tau} \qquad \left(\Phi_v^j \triangleright \Gamma_j \vdash v{:}\sigma_j\right)_{j\in J}}{\Delta + \Pi +_{j\in J} \Gamma_j \vdash t[\![u]\!]v{:}\tau}$$

where $|J| = 1$ if $I = \emptyset$ and $I = J$ otherwise. Moreover, $\mathtt{sz}(\Phi) = \mathtt{sz}(\Phi_t) + \mathtt{sz}(\Phi_u) +_{j\in J} \mathtt{sz}(\Phi_v^j) + 2$. Then,

$$\Phi' := \dfrac{\dfrac{\Phi_t \triangleright \Delta \vdash t{:}[\sigma_i]_{i\in I} \to \tau \qquad (\Phi_v^j \triangleright \Gamma_j \vdash v{:}\sigma_j)_{j\in J}}{\Delta +_{j\in J} \Gamma_j \vdash tv{:}\tau} \qquad \Phi_u \triangleright \Pi \vdash u{:}\rho}{\Delta +_{j\in J} \Gamma_j + \Pi \vdash (tv)[\![u]\!]{:}\tau}$$

where $\mathtt{sz}(\Phi') = \mathtt{sz}(\Phi_t) +_{j\in J} \mathtt{sz}(\Phi_v^j) + \mathtt{sz}(\Phi_u) + 2 = \mathtt{sz}(\Phi)$. □

We are now going to show the essential properties of the typing system $\mathcal{K}$: Subject Reduction (Theorem 3.4) and Subject Expansion (Theorem 3.6), which follow, respectively, from Lemma 3.3 and Lemma 3.5.

Lemma 3.3 (Substitution) *If* $\Phi_t \triangleright x{:}[\rho_i]_{i\in I};\Gamma \vdash t{:}\tau$ *and* $(\Phi^i_u \triangleright \Delta_i \vdash u{:}\rho_i)_{i\in I}$ *then* $\Phi_{t\{x/u\}} \triangleright \Gamma +_{i\in I} \Delta_i \vdash t\{x/u\}{:}\tau$ *where* $\mathtt{sz}(\Phi_{t\{x/u\}}) = \mathtt{sz}(\Phi_t) +_{i\in I} \mathtt{sz}(\Phi^i_u) - |I|$.

Proof. By induction on the structure of t.

- If $t = y \neq x$ then $t\{x/u\} = y$. By relevance one has $x : [\,];\Gamma = \{y : [\tau]\}$ so that $I = \emptyset$ and $\Gamma = \{y : [\tau]\}$ and $\mathtt{sz}(\Phi) = 1$. Therefore, $\mathtt{sz}(\Phi_{y\{x/u\}}) = \mathtt{sz}(\Phi_y)+0-0 = \mathtt{sz}(\Phi_y) +_{i\in I} \mathtt{sz}(\Phi^i_u) - |I|$.
- If $t = x$ then $t\{x/u\} = u$. By construction and relevance one has $x : [\rho_i]_{i\in I};\Gamma = \{x : [\tau]\}$ so that $I = \{m\}$ and $\Gamma = \emptyset$ and $\rho_m = \tau$ and $\mathtt{sz}(\Phi) = 1$. Therefore, for any context Δ_m such that $\Phi^m_u \triangleright \Delta_m \vdash u{:}\rho_m$ the result holds, where $\mathtt{sz}(\Phi_{x\{x/u\}}) = \mathtt{sz}(\Phi_x) + \mathtt{sz}(\Phi^m_u) - 1 = \mathtt{sz}(\Phi^m_u)$.
- If $t = \lambda y.v$ then by α-conversion one can suppose w.l.o.g. that $y \neq x$, $y \notin \mathtt{fv}(u)$, $y \notin \mathtt{dom}(\gamma)$ and $(y \notin \mathtt{dom}(\Delta_i))_{i\in I}$. By construction, $\tau = \mathcal{M} \to \sigma$ and $\Phi_v \triangleright y : \mathcal{M}; x : [\rho_i]_{i\in I};\Gamma \vdash v{:}\sigma$ where $\mathtt{sz}(\Phi_v)+1 = \mathtt{sz}(\Phi_{\lambda y.v})$. By the *i.h.*, $\Phi_{v\{x/u\}} \triangleright (y : \mathcal{M};\Gamma) +_{i\in I} \Delta_i \vdash v\{x/u\}{:}\sigma$ where $\mathtt{sz}(\Phi_{v\{x/u\}}) = \mathtt{sz}(\Phi_v)+_{i\in I}\mathtt{sz}(\Phi^i_u)-|I|$. One has $y \notin \mathtt{dom}(\Delta_i)$ for any $i \in I$ thus $(y : \mathcal{M};\Gamma) +_{i\in I} \Delta_i = y : \mathcal{M};(\Gamma +_{i\in I} \Delta_i)$. The derivation $\Phi_{\lambda y.v\{x/u\}}$ is hence

$$\frac{\Phi_{v\{x/u\}} \triangleright y : \mathcal{M};\Gamma +_{i\in I} \Delta_i \vdash v\{x/u\}{:}\sigma}{\Gamma +_{i\in I} \Delta_i \vdash \lambda y.v\{x/u\}{:}\mathcal{M}\to\sigma}\ (\to_i)$$

  and $\mathtt{sz}(\Phi_{\lambda y.v\{x/u\}}) = \mathtt{sz}(\Phi_{v\{x/u\}}) + 1 = \mathtt{sz}(\Phi_{\lambda y.v}) +_{i\in I} \mathtt{sz}(\Phi^i_u) - |I|$.
- If $t = pv$ then by construction one has $x : [\rho_i]_{i\in I};\Gamma = \Delta +_{j\in J} \Gamma_j$, where $\Phi_p \triangleright \Delta \vdash p{:}[\sigma_k]_{k\in K}\to\tau$ and $(\Phi^j_v \triangleright \Gamma_j \vdash v{:}\sigma_j)_{j\in J}$ and $\mathtt{sz}(\Phi_{pv}) = \mathtt{sz}(\Phi_p)+_{j\in J}\mathtt{sz}(\Phi^j_v)+1$, where either $K = \emptyset$ and $|J| = 1$, or $K = J$. One has $\Delta = x : [\rho_i]_{i\in I_p};\Delta'$ and $(\Gamma_j = x : [\rho_i]_{i\in I_j};\Gamma'_j)_{j\in J}$ where $I = I_p \cup_{j\in J} I_j$, and $I_p, (I_j)_{j\in J}$ can be assumed to be pairwise disjoint sets w.l.o.g. By the *i.h.*, $\Phi_{p\{x/u\}} \triangleright \Delta' +_{i\in I_p} \Delta_i \vdash p\{x/u\}{:}[\sigma_k]_{k\in K}\to\tau$ and $(\Phi^j_{v\{x/u\}} \triangleright \Gamma'_j +_{i\in I_j} \Delta_i \vdash v\{x/u\}{:}\sigma_j)_{j\in J}$ where $\mathtt{sz}(\Phi_{p\{x/u\}}) = \mathtt{sz}(\Phi_p) +_{i\in I_p} \mathtt{sz}(\Phi^i_u) - |I_p|$ and $\mathtt{sz}(\Phi^j_{v\{x/u\}}) = \mathtt{sz}(\Phi^j_v) +_{i\in I_j} \mathtt{sz}(\Phi^i_u) - |I_j|$, for each $j \in J$. Note that $\Delta' +_{j\in J} \Gamma'_j = (x : [\rho_i]_{i\in I};\Gamma) \setminus\setminus x = \Gamma$ and that $+_{i\in I_p}\Delta_i +_{j\in J} (+_{k\in I_j}\Delta_k) = +_{i\in I}\Delta_i$. Therefore, $\Phi_{(pv)\{x/u\}}$ is of the form

$$\frac{\Phi_{p\{x/u\}} \triangleright \Delta' +_{i\in I_p} \Delta_i \vdash p\{x/u\}{:}[\sigma_k]_{k\in K}\to\tau \qquad \left(\Phi^j_{v\{x/u\}} \triangleright \Gamma'_j +_{i\in I_j} \Delta_i \vdash v\{x/u\}{:}\sigma_j\right)_{j\in J}}{\Gamma +_{i\in I} \Delta_i \vdash (pv)\{x/u\}{:}\tau}$$

  where $\mathtt{sz}(\Phi_{(pv)\{x/u\}}) = \mathtt{sz}(\Phi_{p\{x/u\}})+_{j\in J}\mathtt{sz}(\Phi^j_{v\{x/u\}})+1 = \mathtt{sz}(\Phi_p)+_{i\in I_p}\mathtt{sz}(\Phi^i_u)-|I_p| +_{j\in J} \mathtt{sz}(\Phi^j_v) +_{j\in J} (+_{k\in I_j}\mathtt{sz}(\Phi^k_u)) -_{j\in J} |I_j| + 1 = \mathtt{sz}(\Phi_{pv}) +_{i\in I} \mathtt{sz}(\Phi^i_u) - |I|$.
- If $t = p[\![v]\!]$ then by construction $x : [\rho_i]_{i\in I};\Gamma = \Delta + \Pi$ such that $\Phi_p \triangleright \Delta \vdash p{:}\tau$ and $\Phi_v \triangleright \Pi \vdash v{:}\sigma$, where $\mathtt{sz}(\Phi_t) = \mathtt{sz}(\Phi_p) + \mathtt{sz}(\Phi_v) + 1$. One has $\Delta = x : [\rho_i]_{i\in I_p};\Delta'$ and $\Pi = x : [\rho_i]_{i\in I_v};\Pi'$ where $I_p \cup I_v = I$ and $\Delta' + \Pi' = \Gamma$. Suppose w.l.o.g. that I_p and I_v are disjoint. By the *i.h.*, $\Phi_{p\{x/u\}} \triangleright \Delta' +_{i\in I_p} \Delta_i \vdash p\{x/u\}{:}\tau$ and $\Phi_{v\{x/u\}} \triangleright \Pi' +_{i\in I_v} \Delta_i \vdash v\{x/u\}{:}\sigma$ such that $\mathtt{sz}(\Phi_{p\{x/u\}}) = \mathtt{sz}(\Phi_p) +_{i\in I_p} \mathtt{sz}(\Phi^i_u) - |I_p|$ and $\mathtt{sz}(\Phi_{v\{x/u\}}) = \mathtt{sz}(\Phi_v) +_{i\in I_v} \mathtt{sz}(\Phi^i_u) - |I_v|$. Then,

$$\Phi_{p[\![v]\!]\{x/u\}} := \frac{\Phi_{p\{x/u\}} \triangleright \Delta' +_{i\in I_p} \Delta_i \vdash p\{x/u\}{:}\tau \qquad \Phi_{v\{x/u\}} \triangleright \Pi' +_{i\in I_v} \Delta_i \vdash v\{x/u\}{:}\sigma}{\Gamma +_{i\in I} \Delta_i \vdash p\{x/u\}[\![v\{x/u\}]\!]{:}\tau}$$

where $\mathtt{sz}(\Phi_{p[\![v]\!]\{x/u\}}) = \mathtt{sz}(\Phi_{p\{x/u\}}) + \mathtt{sz}(\Phi_{v\{x/u\}}) + 1 = \mathtt{sz}(\Phi_t) +_{i\in I} \mathtt{sz}(\Phi_u^i) - |I|$. □

Theorem 3.4 (Weighted Subject Reduction) *Let $\Phi \triangleright \Gamma \vdash t{:}\tau$. If $t \to_{\mathtt{K}} t'$ then there exists $\Phi' \triangleright \Gamma \vdash t'{:}\tau$ such that $\mathtt{sz}(\Phi) > \mathtt{sz}(\Phi')$.*

Proof. By induction on the reduction relation $\to_{\mathtt{K}}$ using Lemma 3.2 to justify the statement for the equivalence relation $\sim_\sigma$.

- If $t = (\lambda x.v)u$ then by construction $\Gamma = \Delta +_{j\in J} \Gamma_j$ and Φ has the form:

$$\Phi := \dfrac{\dfrac{\Phi_v \triangleright x : [\sigma_i]_{i\in I}; \Delta \vdash v{:}\tau}{\Delta \vdash \lambda x.v{:}[\sigma_i]_{i\in I} \to \tau} \qquad (\Phi_u^j \triangleright \Gamma_j \vdash u{:}\sigma_j)_{j\in J}}{\Delta +_{j\in J} \Gamma_j \vdash (\lambda x.v)s{:}\tau}$$

  Moreover, $\mathtt{sz}(\Phi) = \mathtt{sz}(\Phi_v) +_{j\in J} \mathtt{sz}(\Phi_u^j) + 2$. There are two cases for t'.
  - If $x \in \mathtt{fv}(v)$ then $t' = v\{x/u\}$.
    Suppose $I = \emptyset$ so that $\Gamma = \Delta + \Gamma_u$ and $\Phi_v \triangleright \Delta \vdash v{:}\tau$ where $x \notin \mathtt{dom}(\Delta)$. However, $x \in \mathtt{fv}(v)$ then, by Lemma 3.1, this case is not possible.
    Suppose $I \neq \emptyset$ (*i.e.* $I = J$) so that $\Gamma = \Delta +_{i\in I} \Gamma_i$ where $\Phi_v \triangleright x : [\sigma_i]_{i\in I}; \Delta \vdash v{:}\tau$ and $(\Phi_u^i \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I}$ and $\mathtt{sz}(\Phi) = \mathtt{sz}(\Phi_v) +_{i\in I} \mathtt{sz}(\Phi_u^i) + 2$. By Lemma 3.3, $\Phi_{v\{x/u\}} \triangleright \Gamma \vdash v\{x/u\}{:}\tau$ where $\mathtt{sz}(\Phi_{v\{x/u\}}) = \mathtt{sz}(\Phi_v) +_{i\in I} \mathtt{sz}(\Phi_u^i) - |I| < \mathtt{sz}(\Phi)$. We are done with this case taking $\Phi' := \Phi_{v\{x/u\}}$.
  - If $x \notin \mathtt{fv}(v)$ then $t' = v[\![u]\!]$. Moreover, Lemma 3.1 gives $I = \emptyset$ so that $J = \{j\}$ where $\Gamma = \Delta + \Gamma_j$. Then,

$$\Phi' := \dfrac{\Phi_v \triangleright \Delta \vdash v{:}\tau \qquad \Phi_u^j \triangleright \Gamma_j \vdash u{:}\sigma_j}{\Gamma \vdash v[\![u]\!]{:}\tau}$$

    where $\mathtt{sz}(\Phi') = \mathtt{sz}(\Phi_v) + \mathtt{sz}(\Phi_u^j) + 1 < \mathtt{sz}(\Phi)$.
- All the inductive cases are straightforward.

□

We illustrate the Weighted Subject Reduction property by the following example. Let $\underline{2}$ be the Church numeral $\lambda y.\lambda x.y(y(x))$ and let t be an arbitrary closed term. Then $u_0 = \underline{2}t \to_\beta \lambda x.t(t(x)) = u_1$. Suppose $\Phi_t \triangleright \vdash t{:}[\sigma] \to \sigma$. Then, given the following typing derivation Φ_0 for u_0

$$\Phi_0 := \dfrac{\Phi_{\underline{2}} \triangleright \vdash \underline{2}{:}[[\sigma] \to \sigma, [\sigma] \to \sigma] \to [\sigma] \to \sigma \qquad \Phi_t \triangleright \vdash t{:}[\sigma] \to \sigma \qquad \Phi_t \triangleright \vdash t{:}[\sigma] \to \sigma}{\vdash \underline{2}t{:}[\sigma] \to \sigma}$$

where $\Phi_{\underline{2}}$ is the following typing derivation:

$$\Phi_{\underline{2}} := \dfrac{\dfrac{\dfrac{}{y : [[\sigma]\to\sigma] \vdash y{:}[\sigma]\to\sigma} \quad \dfrac{\dfrac{}{y : [[\sigma]\to\sigma] \vdash y{:}[\sigma]\to\sigma} \quad \dfrac{}{x : [\sigma] \vdash x{:}\sigma}}{y : [[\sigma]\to\sigma], x : [\sigma] \vdash yx{:}\sigma}}{y : [[\sigma]\to\sigma, [\sigma]\to\sigma], x : [\sigma] \vdash y(yx){:}\sigma}}{y : [[\sigma]\to\sigma, [\sigma]\to\sigma] \vdash \lambda x.y(yx){:}[\sigma]\to\sigma}}{\vdash \lambda y.\lambda x.y(yx){:}[[\sigma]\to\sigma, [\sigma]\to\sigma]\to[\sigma]\to\sigma}$$

we can construct the following typing derivation Φ_1 for u_1

$$\Phi_1 := \dfrac{\dfrac{\dfrac{\Phi_t \triangleright \vdash t{:}[\sigma]\to\sigma}{\Phi_t \triangleright \vdash t{:}[\sigma]\to\sigma} \quad \dfrac{\dfrac{\Phi_t \triangleright \vdash t{:}[\sigma]\to\sigma}{\Phi_t \triangleright \vdash t{:}[\sigma]\to\sigma} \quad \dfrac{}{x : [\sigma] \vdash x{:}\sigma}}{x : [\sigma] \vdash tx{:}\sigma}}{x : [\sigma] \vdash t(tx){:}\sigma}}{\vdash \lambda x.t(tx){:}[\sigma]\to\sigma}$$

such that $\mathtt{sz}(\Phi_0) > \mathtt{sz}(\Phi_1)$. Indeed,

$$\mathtt{sz}(\Phi_0) = \mathtt{sz}(\Phi_t)\cdot 2 + 8 > \mathtt{sz}(\Phi_t)\cdot 2 + 4 = \mathtt{sz}(\Phi_1)$$

To obtain Subject Expansion we first need the following property.

Lemma 3.5 (Reverse Substitution) *Let x, t, u be $\mathtt{K}$-terms. If $x \in \mathtt{fv}(t)$ and $\Phi \triangleright \Gamma \vdash t\{x/u\}{:}\tau$ then $\Gamma = \Delta +_{i\in I} \Gamma_i$ where $I \neq \emptyset$ and $\Phi_t \triangleright x : [\sigma_i]_{i\in I}; \Delta \vdash t{:}\tau$ and $(\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I}$.*

Proof. By induction on the structure of t.

- If $t = x$ then $t\{x/u\} = u$ and the result holds, for $\Delta = \emptyset$ and $I = \{m\}$ where $\sigma_m = \tau$. The derivation Φ_t is obtained by an application of rule ($\mathtt{ax}$). Observe that $t = y$ would imply $x \notin \mathtt{fv}(t)$.
- If $t = \lambda y.v$ then $t\{x/u\} = \lambda y.v\{x/u\}$ thus, by construction, $\tau = \mathcal{M}\to\sigma$ where $\Phi \triangleright y : \mathcal{M}; \Gamma \vdash v\{x/u\}{:}\sigma$. By the *i.h.*, $y : \mathcal{M}; \Gamma = \Delta' +_{i\in I} \Gamma_i$ where $\Phi_v \triangleright x : [\sigma_i]_{i\in I}; \Delta' \vdash v{:}\sigma$ and $(\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I}$. By α-conversion one can suppose w.l.o.g. that $y \notin \mathtt{fv}(u)$, then by Lemma 3.1 one has $\Delta' = y : \mathcal{M}; x : [\sigma_i]_{i\in I}; \Delta$ and $(x : [\sigma_i]_{i\in I}; \Delta) +_{i\in I} \Gamma_i = \Gamma$. Then $\Phi_t \triangleright x : [\sigma_i]_{i\in I}; \Delta \vdash \lambda y.v{:}\tau$ by rule ($\to_i$).
- If $t = pv$ then $t\{x/u\} = p\{x/u\}v\{x/u\}$ and by construction $\Gamma = \Pi +_{j\in J} \Pi_j$ and $\Phi_{p\{x/u\}} \triangleright \Pi \vdash p\{x/u\}{:}[\rho_k]_{k\in K}\to\tau$ and $(\Phi^j_{v\{x/u\}} \triangleright \Pi_j \vdash v\{x/u\}{:}\rho_j)_{j\in J}$ where either $K = \emptyset$ and $|J| = 1$ or $K = J$.

  If $x \in \mathtt{fv}(p)$ and $x \in \mathtt{fv}(v)$ then, by the *i.h.* $\Pi = \Delta_p +_{i\in I_p} \Gamma_i$ where $\Phi_p \triangleright x : [\sigma_i]_{i\in I_p}; \Delta_p \vdash p{:}[\rho_k]_{k\in K}\to\tau$ and $(\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I_p}$, and $\Pi_j = \Delta_j +_{i\in I_j} \Gamma_i$ where $\Phi_v \triangleright x : [\sigma_i]_{i\in I_j}; \Delta_j \vdash v{:}\rho_j$ and $(\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I_j}$, for each $j \in J$. Suppose w.l.o.g. that $I_p, (I_j)_{j\in J}$ are pairwise disjoint and let $I = I_p \cup_{j\in J} I_j$. Then, by rule ($\to_e$), $\Phi_t \triangleright x : [\sigma_i]_{i\in I}; \Delta_p +_{j\in J} \Delta_j \vdash pv{:}\tau$. We are done with this case for $\Delta := \Delta_p +_{j\in J} \Delta_j$ since $(x : [\sigma_i]_{i\in I_p}; \Delta_p) +_{j\in J} (x : [\sigma_i]_{i\in I_j}; \Delta_j) = x : [\sigma_i]_{i\in I}; (\Delta_p +_{j\in J} \Delta_j)$ and that

$\Delta_p +_{j\in J} \Delta_j +_{i\in I} \Gamma_i = \Gamma$.

If either $x \notin \mathtt{fv}(v)$ or $x \notin \mathtt{fv}(p)$ then it is analogous to the case above.

- If $t = p[\![v]\!]$ then $t\{x/u\} = p\{x/u\}[\![v\{x/u\}]\!]$ and by construction $\Gamma = \Pi_0 + \Pi_1$ such that $\Phi_{p\{x/u\}} \triangleright \Pi_0 \vdash p\{x/u\}{:}\tau$ and $\Phi_{v\{x/u\}} \triangleright \Pi_1 \vdash v\{x/u\}{:}\sigma$.

  If $x \in \mathtt{fv}(p)$ and $x \in \mathtt{fv}(v)$ then by *i.h.* $\Pi_0 = \Delta_p +_{i\in I_p} \Gamma_i$ where $\Phi_p \triangleright x : [\sigma_i]_{i\in I_p}; \Delta_p \vdash p{:}\tau$ and $(\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I_p}$ and $\Pi_1 = \Delta_v +_{i\in I_v} \Gamma_i$ where $\Phi_v \triangleright x : [\sigma_i]_{i\in I_v}; \Delta_v \vdash v{:}\sigma$ and $(\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I_v}$. Suppose w.l.o.g. that I_p and I_v are disjoint and let $I = I_p \cup I_v$. Then $\Phi_t \triangleright x : [\sigma_i]_{i\in I}; \Delta_p + \Delta_v \vdash p[\![v]\!]{:}\tau$ by the rule $(\mathtt{m})$ and we are done with $\Delta := \Delta_p + \Delta_v$.

  If $x \notin \mathtt{fv}(v)$ then $v\{x/u\} = v$ and by the *i.h.* one has $\Pi_0 = \Delta_p +_{i\in I} \Gamma_i$ such that $\Phi_p \triangleright x : [\sigma_i]_{i\in I}; \Delta_p \vdash p{:}\tau$ and $(\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I}$. Then, by rule $(\mathtt{m})$, $\Phi_t \triangleright x : [\sigma_i]_{i\in I}; \Delta_p + \Pi_1 \vdash p[\![v]\!]{:}\tau$ and we are done with $\Delta := \Delta_p + \Pi_1$.

  If $x \notin \mathtt{fv}(p)$ then it is analogous to the one above.

□

Theorem 3.6 (Subject Expansion) *Let $\Phi' \triangleright \Gamma \vdash t'{:}\tau$. If $t \to_{\mathtt{K}} t'$ then $\Phi \triangleright \Gamma \vdash t{:}\tau$.*

Proof. The proof is by induction on the reduction relation $\to_{\mathtt{K}}$.

- If $t = (\lambda x.v)u \mapsto_{\mathtt{neb}} v\{x/u\} = t'$, where $x \in \mathtt{fv}(v)$, then by Lemma 3.5 $\Gamma = \Delta +_{i\in I} \Gamma_i$ where $I \neq \emptyset$, $\Phi_v \triangleright x : [\sigma_i]_{i\in I}; \Delta \vdash v{:}\tau$ and $(\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I}$. Then

$$\dfrac{\dfrac{\Phi_v \triangleright x : [\sigma_i]_{i\in I}; \Delta \vdash v{:}\tau}{\Delta \vdash \lambda x.v{:}[\sigma_i]_{i\in I} \to \tau} \qquad (\Phi^i_u \triangleright \Gamma_i \vdash u{:}\sigma_i)_{i\in I}}{\Gamma \vdash (\lambda x.v)u{:}\tau} \to_e$$

- If $t = (\lambda x.v)u \mapsto_{\mathtt{m}} v[\![u]\!] = t'$, where $x \notin \mathtt{fv}(v)$ then, by construction, $\Gamma = \Delta + \Gamma_1$ such that $\Phi_v \triangleright \Delta \vdash v{:}\tau$ and $\Phi_u \triangleright \Gamma_1 \vdash u{:}\sigma$. Note that, by Lemma 3.1, $x \notin \mathtt{dom}(\Delta)$. Then,

$$\dfrac{\dfrac{\Phi_v \triangleright \Delta \vdash v{:}\tau}{\Delta \vdash \lambda x.v{:}[\,] \to \tau} \qquad \Phi_u \triangleright \Gamma_1 \vdash u{:}\sigma}{\Gamma \vdash (\lambda x.v)u{:}\tau} \to_e$$

- All the inductive cases are straightforward.

□

4 The Strong Normalization Characterization

In this section we use $\mathcal{K}$-typability to characterize β-strongly normalizing terms, *i.e.* we show that a term is $\mathcal{K}$-typable if and only if it is β-strongly normalizing. The proof does not use any reducibility/computability argument [31,17,25], but goes through the memory operator calculus K that we have introduced in Section 2. More precisely, we first show that K-terms are $\mathcal{K}$-typable if and only if they are K-strongly normalizing. This proof is based on the properties presented in Section 3, namely,

Weighted Subject Reduction and Subject Expansion. We then show that the sets of β and $\mathtt{K}$ strongly normalizing λ-terms are equivalent, a result which is obtained by means of appropriate inductive definitions for both sets.

It is well-known [36] that the set of β-strongly normalizing λ-terms, written $\mathcal{SN}(\beta)$, can be characterized by the following alternative inductive definition:

- If $t_1, \ldots, t_n$ $(n \geq 0) \in \mathcal{ISN}(\beta)$, then $xt_1 \ldots t_n \in \mathcal{ISN}(\beta)$.
- If $t \in \mathcal{ISN}(\beta)$, then $\lambda x.t \in \mathcal{ISN}(\beta)$.
- If $t\{x/u\}t_1 \ldots t_n, u \in \mathcal{ISN}(\beta)$, then $(\lambda x.t)ut_1 \ldots t_n \in \mathcal{ISN}(\beta)$.

Remark that the base case of this inductive definition is given by the first item when $n = 0$. This is indeed a characterization, as expressed by the following Lemma:

Lemma 4.1 ([36]) $\mathcal{SN}(\beta) = \mathcal{ISN}(\beta)$.

In the same spirit, the set of $\mathtt{K}$-strongly normalizing $\mathtt{K}$-terms, written $\mathcal{SN}(\mathtt{K})$, can be characterized by the following alternative definition:

(H) If $t_1, \ldots, t_n$ $(n \geq 0) \in \mathcal{ISN}(\mathtt{K})$, then $xt_1 \ldots t_n \in \mathcal{ISN}(\mathtt{K})$.

(A) If $t \in \mathcal{ISN}(\mathtt{K})$, then $\lambda x.t \in \mathcal{ISN}(\mathtt{K})$.

(I) If $t\{x/u\}t_1 \ldots t_n \in \mathcal{ISN}(\mathtt{K})$ and $x \in \mathtt{fv}(t)$, then $(\lambda x.t)ut_1 \ldots t_n \in \mathcal{ISN}(\mathtt{K})$.

(G) If $t[\![u]\!]t_1 \ldots t_n \in \mathcal{ISN}(\mathtt{K})$ and $x \notin \mathtt{fv}(t)$, then $(\lambda x.t)ut_1 \ldots t_n \in \mathcal{ISN}(\mathtt{K})$.

(S) If $(tt_1 \ldots t_n)[\![u]\!] \in \mathcal{ISN}(\mathtt{K})$ and $n \geq 1$, then $t[\![u]\!]t_1 \ldots t_n \in \mathcal{ISN}(\mathtt{K})$.

(V) If $t, u \in \mathcal{ISN}(\mathtt{K})$, then $t[\![u]\!] \in \mathcal{ISN}(\mathtt{K})$.

As before, the base case of this inductive definition is given by the first item when $n = 0$.

The set $\mathcal{ISN}(\mathtt{K})$ turns out to be equivalent to the set of $\mathtt{K}$-strongly normalizing terms. In order to show that, we write $\eta_{\mathtt{K}}(t)$ to denote the maximal length of a $\mathtt{K}$-reduction sequence starting at t, when t is $\mathtt{K}$-strongly normalizing.

Lemma 4.2 $\mathcal{SN}(\mathtt{K}) = \mathcal{ISN}(\mathtt{K})$.

Proof. If $t \in \mathcal{SN}(\mathtt{K})$, then we easily show that $t \in \mathcal{ISN}(\mathtt{K})$ by induction on the pair $\langle \eta_{\mathtt{K}}(t), |t| \rangle$ using the corresponding lexicographic order.

- The base case of the induction is when $t = x$ (for which $\langle \eta_{\mathtt{K}}(t), |t| \rangle = \langle 0, 1 \rangle$). Then the rule (H) for $n = 0$ gives $x \in \mathcal{ISN}(\mathtt{K})$.
- We reason by cases on the form of t for the inductive steps.
  - $t = \lambda x.u \in \mathcal{SN}(\mathtt{K})$ implies $u \in \mathcal{SN}(\mathtt{K})$. Moreover, $\langle \eta_{\mathtt{K}}(u), |u| \rangle <_{lex} \langle \eta_{\mathtt{K}}(t), |t| \rangle$ so that the *i.h.* gives $u \in \mathcal{ISN}(\mathtt{K})$ and the rule (A) gives $t \in \mathcal{ISN}(\mathtt{K})$.
  - $t = xu_1 \ldots u_n \in \mathcal{SN}(\mathtt{K})$ implies $u_1, \ldots, u_n \in \mathcal{SN}(\mathtt{K})$. Moreover, $\langle \eta_{\mathtt{K}}(u_i), |u_i| \rangle <_{lex} \langle \eta_{\mathtt{K}}(t), |t| \rangle$ so that the *i.h.* gives $u_i \in \mathcal{ISN}(\mathtt{K})$ and the rule (H) gives $t \in \mathcal{ISN}(\mathtt{K})$.
  - $t = (\lambda x.u)vt_1 \ldots t_n \in \mathcal{SN}(\mathtt{K})$ and $x \in \mathtt{fv}(u)$ implies $t' = u\{x/v\}t_1 \ldots t_n \in \mathcal{SN}(\mathtt{K})$. Moreover, $\langle \eta_{\mathtt{K}}(t'), |t'| \rangle <_{lex} \langle \eta_{\mathtt{K}}(t), |t| \rangle$ so that the *i.h.* gives $t' \in \mathcal{ISN}(\mathtt{K})$ and the rule (I) gives $t \in \mathcal{ISN}(\mathtt{K})$.
  - $t = (\lambda x.u)vt_1 \ldots t_n \in \mathcal{SN}(\mathtt{K})$ and $x \notin \mathtt{fv}(u)$ implies $t' = u[\![v]\!]t_1 \ldots t_n \in \mathcal{SN}(\mathtt{K})$. Moreover, $\langle \eta_{\mathtt{K}}(t'), |t'| \rangle <_{lex} \langle \eta_{\mathtt{K}}(t), |t| \rangle$ so that the *i.h.* gives $t' \in \mathcal{ISN}(\mathtt{K})$ and the

rule (G) gives $t \in \mathcal{ISN}(\mathtt{K})$.

- $t = u[\![v]\!] \in \mathcal{SN}(\mathtt{K})$ implies $u, v \in \mathcal{SN}(\mathtt{K})$. Moreover, $\langle \eta_{\mathtt{K}}(u), |u| \rangle <_{lex} \langle \eta_{\mathtt{K}}(t), |t| \rangle$ and $\langle \eta_{\mathtt{K}}(v), |v| \rangle <_{lex} \langle \eta_{\mathtt{K}}(t), |t| \rangle$ so that the *i.h.* gives $u, v \in \mathcal{ISN}(\mathtt{K})$ and the rule (V) gives $t \in \mathcal{ISN}(\mathtt{K})$.
- $t = u[\![v]\!]t_1 \dots t_n \in \mathcal{SN}(\mathtt{K})$, where $n \geq 1$. By definition we have $\eta_{\mathtt{K}}(u[\![v]\!]t_1 \dots t_n) = \eta_{\mathtt{K}}((ut_1 \dots t_n)[\![v]\!])$, so that $ut_1 \dots t_n, v \in \mathcal{SN}(\mathtt{K})$. Moreover $\langle \eta_{\mathtt{K}}(v), |v| \rangle <_{lex} \langle \eta_{\mathtt{K}}(t), |t| \rangle$ and $\langle \eta_{\mathtt{K}}(ut_1 \dots t_n), |ut_1 \dots t_n| \rangle <_{lex} \langle \eta_{\mathtt{K}}(t), |t| \rangle$. By the *i.h.* we get $ut_1 \dots t_n, v \in \mathcal{ISN}(\mathtt{K})$, and thus by rule (V) we get $(ut_1 \dots t_n)[\![v]\!] \in \mathcal{ISN}(\mathtt{K})$, and by rule (S) we obtain $u[\![v]\!]t_1 \dots t_n \in \mathcal{ISN}(\mathtt{K})$.

To show $\mathcal{ISN}(\mathtt{K}) \subseteq \mathcal{SN}(\mathtt{K})$ we reason by induction on the definition of $\mathcal{ISN}(\mathtt{K})$.

(H) If $t = xt_1 \dots t_n \in \mathcal{ISN}(\mathtt{K})$, where $t_1, \dots, t_n \in \mathcal{ISN}(\mathtt{K})$, then the *i.h.* gives $t_1, \dots, t_n \in \mathcal{SN}(\mathtt{K})$ so that the term $xt_1 \dots t_n$ is trivially in $\mathcal{SN}(\mathtt{K})$.

(A) If $t = \lambda x.v \in \mathcal{ISN}(\mathtt{K})$, where $v \in \mathcal{ISN}(\mathtt{K})$, then the *i.h.* gives $v \in \mathcal{SN}(\mathtt{K})$ so that the term $\lambda x.v$ is trivially in $\mathcal{SN}(\mathtt{K})$.

(I) If $t = (\lambda x.v)ut_1 \dots t_n \in \mathcal{ISN}(\mathtt{K})$, where $v\{x/u\}t_1 \dots t_n \in \mathcal{ISN}(\mathtt{K})$ and $x \in \mathtt{fv}(v)$, then the *i.h.* gives $v\{x/u\}t_1 \dots t_n \in \mathcal{SN}(\mathtt{K})$ so that in particular $v, u, t_i \in \mathcal{SN}(\mathtt{K})$. We show that $t \in \mathcal{SN}(\mathtt{K})$ by a second induction on $\eta_{\mathtt{K}}(v) + \eta_{\mathtt{K}}(u) + \Sigma_{i=1\dots n}\eta_{\mathtt{K}}(t_i)$.

Let us see how are all the reducts of t.

If $t \to (\lambda x.v')ut_1 \dots t_n = t'$, where $v \to v'$ or $t \to (\lambda x.v)u't_1 \dots t_n = t'$, where $u \to u'$, or $t \to (\lambda x.v)ut_1 \dots t_i' \dots t_n = t'$, where $t_i \to t_i'$, then $t' \in \mathcal{SN}(\mathtt{K})$ by the second *i.h.*.

If $t \to v\{x/u\}t_1 \dots t_n = t'$, then $t' \in \mathcal{SN}(\mathtt{K})$ as already remarked by the first *i.h.*

Since all reducts of t are in $\mathcal{SN}(\mathtt{K})$, then $t \in \mathcal{SN}(\mathtt{K})$.

(G) If $t = (\lambda x.v)ut_1 \dots t_n \in \mathcal{ISN}(\mathtt{K})$, where $v[\![u]\!]t_1 \dots t_n \in \mathcal{ISN}(\mathtt{K})$ and $x \notin \mathtt{fv}(v)$, then the *i.h.* gives $v[\![u]\!]t_1 \dots t_n \in \mathcal{SN}(\mathtt{K})$, so that in particular $v, u, t_i \in \mathcal{SN}(\mathtt{K})$. We show that $t \in \mathcal{SN}(\mathtt{K})$ by induction on $\eta_{\mathtt{K}}(v) + \eta_{\mathtt{K}}(u) + \Sigma_{i=1\dots n}\eta_{\mathtt{K}}(t_i)$.

As before, let us analyze the reducts of t.

If $t \to (\lambda x.v')ut_1 \dots t_n = t'$, where $v \to v'$ or $t \to (\lambda x.v)u't_1 \dots t_n = t'$, where $u \to u'$, or $t \to (\lambda x.v)ut_1 \dots t_i' \dots t_n = t'$, where $t_i \to t_i'$, then $t' \in \mathcal{SN}(\mathtt{K})$ by the second *i.h.*.

If $t \to v[\![u]\!]t_1 \dots t_n = t'$, then we have remarked already that $t' \in \mathcal{SN}(\mathtt{K})$ by the first *i.h.*

Since all reducts of t are in $\mathcal{SN}(\mathtt{K})$, then $t \in \mathcal{SN}(\mathtt{K})$.

(S) If $v[\![u]\!]t_1 \dots t_n \in \mathcal{ISN}(\mathtt{K})$ where $(vt_1 \dots t_n)[\![u]\!] \in \mathcal{ISN}(\mathtt{K})$ and $n \geq 1$, then the only possibility is $vt_1 \dots t_n, u \in \mathcal{ISN}(\mathtt{K})$. We can then apply the *i.h.* to get $vt_1 \dots t_n, u \in \mathcal{SN}(\mathtt{K})$, so that $(vt_1 \dots t_n)[\![u]\!] \equiv v[\![u]\!]t_1 \dots t_n \in \mathcal{SN}(\mathtt{K})$.

(V) If $t = v[\![u]\!] \in \mathcal{ISN}(\mathtt{K})$, where $v, u \in \mathcal{ISN}(\mathtt{K})$, then the *i.h.* gives $u, v \in \mathcal{SN}(\mathtt{K})$ so that the term $v[\![u]\!]$ is trivially in $\mathcal{SN}(\mathtt{K})$.

□

Since $\mathcal{SN}(\mathtt{K})$ and $\mathcal{ISN}(\mathtt{K})$ are equivalent sets, we can now derive $\mathcal{K}$-typability from K-strong normalization by using this equivalence.

Theorem 4.3 *If $t \in \mathcal{SN}(\mathtt{K})$ then t is $\mathcal{K}$-typable.*

Proof. By induction on the structure of $t \in \mathcal{ISN}(\mathtt{K}) = \mathcal{SN}(\mathtt{K})$.

(H) If $t = xt_1 \cdots t_n$ $(n \geq 0)$ where $t_1, \ldots, t_n \in \mathcal{ISN}(\mathtt{K})$ then, by the *i.h.* $\forall 1 \leq i \leq n, \Gamma_i \vdash t_i{:}\sigma_i$. Let $\tau = [\sigma_1] \to \cdots [\sigma_n] \to \alpha$, for $\alpha \in \mathcal{A}$, and $\Gamma = x : [\tau] + \Gamma_1 + \cdots + \Gamma_n$. Then, $x : [\tau] \vdash x{:}\tau$ by the rule $(\mathtt{ax})$ and, by n applications of the rule $(\to_e)$, $\Gamma \vdash xt_1 \cdots t_n{:}\alpha$.

(A) If $t = \lambda x.v$ where $v \in \mathcal{ISN}(\mathtt{K})$ then, by the *i.h.* $\Gamma \vdash v{:}\tau$ thus, by the rule $(\to_i)$, $\Gamma \setminus\!\setminus x \vdash \lambda x.v{:}\Gamma(x) \to \tau$.

(I) If $t = (\lambda x.v)ut_1 \cdots t_n$ where $x \in \mathtt{fv}(v)$ and $v\{x/u\}t_1 \cdots t_n \in \mathcal{ISN}(\mathtt{K})$ then, by the *i.h.* $\Gamma \vdash v\{x/u\}t_1 \cdots t_n{:}\tau$. Therefore, by Theorem 3.6, $\Gamma \vdash (\lambda x.v)ut_1 \cdots t_n{:}\tau$.

(G) If $t = (\lambda x.v)ut_1 \cdots t_n$ where $x \notin \mathtt{fv}(v)$ and $v[\![u]\!]t_1 \cdots t_n \in \mathcal{ISN}(\mathtt{K})$ then, by the *i.h.* $\Gamma \vdash v[\![u]\!]t_1 \cdots t_n{:}\tau$. Therefore, by Theorem 3.6, $\Gamma \vdash (\lambda x.v)ut_1 \cdots t_n{:}\tau$.

(S) If $t = v[\![u]\!]t_1 \cdots t_n$ where $(vt_1 \cdots t_n)[\![u]\!] \in \mathcal{ISN}(\mathtt{K})$, then by the *i.h.* we have $\Gamma \vdash (vt_1 \cdots t_n)[\![u]\!]{:}\tau$ and, by Lemma 3.2, $\Gamma \vdash v[\![u]\!]t_1 \cdots t_n{:}\tau$.

(V) If $t = v[\![u]\!]$ where $v, u \in \mathcal{ISN}(\mathtt{K})$ then, by the *i.h.* $\Gamma_0 \vdash v{:}\tau$ and $\Gamma_1 \vdash u{:}\sigma$ thus, by the rule $(\mathtt{m})$, $\Gamma_0 + \Gamma_1 \vdash v[\![u]\!]{:}\tau$.

□

The converse can be simply shown by using the Weighted Subject Reduction property. Notice that similar results for *idempotent* intersection type systems cannot be proved in a *combinatorial* way, they are typically based on *reducibility* arguments [31,17,25]. This is exactly the place where the quantitative approach by means of non-idempotent types makes the difference.

Theorem 4.4 *If t is $\mathcal{K}$-typable then $t \in \mathcal{SN}(\mathtt{K})$.*

Proof. Let $\Phi \rhd \Gamma \vdash t{:}\tau$ and suppose that $t \notin \mathcal{SN}(\mathtt{K})$. Then, there exists an infinite reduction sequence $t = t_0 \to t_1 \sim t_2 \to t_3 \sim t_4 \to \cdots \to t_{2n+1} \sim t_{2n+2} \to \cdots$. By Theorem 3.4 and Lemma 3.2 one has that $\forall i \in \mathbb{N}\ \exists \Phi_i \rhd \Gamma \vdash t_i{:}\tau$ such that $\mathtt{sz}(\Phi_{2i}) > \mathtt{sz}(\Phi_{2i+1}) = \mathtt{sz}(\Phi_{2(i+1)})$. Contradiction, since $\mathtt{sz}(\Phi) > 0$, for any Φ. Therefore, $t \in \mathcal{SN}(\mathtt{K})$. □

We have already remarked that the set of λ-terms is included in the set of $\mathtt{K}$-terms. In order to conclude, we need to show that a λ-term t is β-strongly normalizable if and only if the $\mathtt{K}$-term obtained by embedding t in the $\mathtt{K}$-calculus is $\mathtt{K}$-strongly normalizable.

Lemma 4.5 *Let t be a λ-term. If $t \in \mathcal{SN}(\mathtt{K})$, then $t \in \mathcal{SN}(\beta)$.*

Proof. By induction on $\langle \eta_{\mathtt{K}}(t), |t| \rangle$. We only show the interesting case. Let $t = (\lambda x.v)ut_1 \cdots t_n$. By hypothesis $v, u, t_i \in \mathcal{SN}(\mathtt{K})$ so that by the *i.h.* we get $v, u, t_i \in \mathcal{SN}(\beta)$. To prove that $t \in \mathcal{SN}(\beta)$ it is sufficient to show that every β-reduct of t is in $\mathcal{SN}(\beta)$. We proceed by induction on $\eta_\beta(v) + \eta_\beta(u) + \Sigma_{i=1\cdots n}\eta_\beta(t_i)$.

If $t \to_\beta t'$ reduces a subterm v, u, t_i, then the property trivially holds.

Otherwise, if $t \to_\beta t'$ reduces the head redex, there are two cases to consider. If $x \in \mathtt{fv}(v)$, then also $t \to_{\mathtt{K}} v\{x/u\}t_1 \cdots t_n = t'$, so that $t' \in \mathcal{SN}(\beta)$ by the first *i.h.* If $x \notin \mathtt{fv}(v)$, then $t \to_\beta vt_1 \cdots t_n = t'$. However, $t \to_{\mathtt{K}} v[\![u]\!]t_1 \cdots t_n = t' \equiv (vt_1 \cdots t_n)[\![u]\!]$. We have $\eta_{\mathtt{K}}(vt_1 \cdots t_n) \leq \eta_{\mathtt{K}}((vt_1 \cdots t_n)[\![u]\!]) < \eta_{\mathtt{K}}(t)$ so that the first *i.h.* gives $vt_1 \cdots t_n = t' \in \mathcal{SN}(\beta)$ and thus we are done. □

Lemma 4.6 *Let t be a λ-term. If $t \in \mathcal{SN}(\beta)$, then $t \in \mathcal{SN}(\mathtt{K})$.*

Proof. By Lemma 4.1 and Lemma 4.2 it is sufficient to show that $t \in \mathcal{ISN}(\beta)$ implies $t \in \mathcal{ISN}(\mathtt{K})$.

- If $t = xt_1 \dots t_n \in \mathcal{ISN}(\beta)$, where $t_1, \dots, t_n \in \mathcal{ISN}(\beta)$, then the *i.h.* gives $t_1, \dots, t_n \in \mathcal{ISN}(\mathtt{K})$ so that $t \in \mathcal{ISN}(\mathtt{K})$ by rule (H).
- If $t = \lambda x.v \in \mathcal{ISN}(\beta)$, where $v \in \mathcal{ISN}(\beta)$, then the *i.h.* gives $v \in \mathcal{ISN}(\mathtt{K})$ so that $\lambda x.v \in \mathcal{ISN}(\mathtt{K})$ by rule (A).
- If $t = (\lambda x.v)ut_1 \dots t_n \in \mathcal{ISN}(\beta)$, where $v\{x/u\}t_1 \dots t_n, u \in \mathcal{ISN}(\beta)$, then we reason by cases.

  Suppose $x \in \mathtt{fv}(v)$. The *i.h.* gives $v\{x/u\}t_1 \dots t_n \in \mathcal{ISN}(\mathtt{K})$ so that $t \in \mathcal{ISN}(\mathtt{K})$ by rule (I).

  Suppose $x \notin \mathtt{fv}(v)$. The *i.h.* gives $vt_1 \dots t_n, u \in \mathcal{ISN}(\mathtt{K})$ so that $(vt_1 \dots t_n)[\![u]\!] \in \mathcal{ISN}(\mathtt{K})$ by rule (V), and thus $t \in \mathcal{ISN}(\mathtt{K})$ by rule (G).

□

The last two lemmas, and the fact the λ-terms are strictly included in K-terms, allow to conclude:

Corollary 4.7 *Let t be a λ-term. Then t is $\mathcal{K}$-typable if and only if $t \in \mathcal{SN}(\beta)$.*

5 Conclusion

We have presented a characterization of β-strongly normalizing λ-terms via a typing system based on non-idempotent intersection types, through an embedding of the λ-calculus into the memory K-calculus.

As a matter of fact, the same result can be proved more directly by using techniques similar to those presented in [33,3]. This alternative proof goes as follows:

- Prove that typable λ-terms are strongly normalizing, by induction on typing derivations, using the inductive characterization of strong normalization $\mathcal{ISN}(\beta)$, presented in Section 4, and the fact that weighted subject reduction holds for *non-erasing* β-reductions.
- Prove that strongly normalizing λ-terms are typable, by induction on $\mathcal{ISN}(\beta)$, using the fact that subject expansion holds for *non-erasing* β-reductions.

It is possible to adapt this technique to the non-idempotent case. Still, the good point of the memory calculus is its awareness: nothing is lost along reductions, and therefore subject expansion does hold even for the type system characterizing strong normalization, whereas it does not hold for other calculi, in general. This allows for simple, modular proofs of strong normalization for calculi that can be embedded in a memory-like calculus. In this paper, we show the simplest case, namely the one of the pure λ-calculus, but extensions to other frameworks, inspired for instance by classical or linear calculi may naturally be conceived.

References

[1] H. Barendregt. *The Lambda Calculus: Its Syntax and Semantics (revised edition)*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. Elsevier Science, Amsterdan, The Netherlands, 1984.

[2] H. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *Journal Symbolic Logic*, 48(4):931–940, 1983.

[3] H. Barendregt, W. Dekkers, and R. Statman. *Lambda calculus with types*. Perspectives in logic. Cambridge University Press, Cambridge, New York, 2013.

[4] A. Bernadet and S. Lengrand. Complexity of strongly normalising λ-terms via non-idempotent intersection types. In *the 14th International Conference on Foundations of Software Science and Computation Structures* (FOSSACS), volume 6604 of *LNCS*, pages 88–107. Springer-Verlag, 2011.

[5] A. Bernadet and S. Lengrand. Non-idempotent intersection types and strong normalisation. *Logical Methods in Computer Science*, 9(4):3, 2013.

[6] G. Boudol, P.-L. Curien, and C. Lavatelli. A semantics for lambda calculi with resources. *Mathematical Structures in Computer Science*, 9(4):437–482, 1999.

[7] A. Bucciarelli, T. Ehrhard, and G. Manzonetto. Not enough points is enough. In *21st International Workshop on Computer Science Logic* (CSL), volume 4646 of *LNCS*, pages 298–312. Springer-Verlag, 2007.

[8] A. Bucciarelli, D. Kesner, and S. Ronchi Della Rocca. The inhabitation problem for non-idempotent intersection types. In *the 8th Int. Conference on Theoretical Computer Science* (TCS), volume 8705 of *LNCS*, pages 341–354. Springer-Verlag, 2014.

[9] A. Bucciarelli, D. Kesner, and D. Ventura. A combinatorial argument for termination. EBL 2014, XVII Brazilian Logic Conference, Petropolis.

[10] M. Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre Dame, Journal of Formal Logic*, 21:685–693, 1980.

[11] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Principal type schemes and λ-calculus semantics. *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 536–560. Academic Press, 1980.

[12] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Functional characters of solvable terms. *Zeitschrift fur mathematische Logik und Grundlagen der Mathematik*, 27:45–58, 1981.

[13] E. De Benedetti. *Linear logic, type assignment systems and implicit computational complexity*. PhD thesis, Universita' degli Studi di Torino - ENS de Lyon, 2015.

[14] E. De Benedetti and S. Ronchi Della Rocca. Bounding normalization time through intersection types. In *the 6th Workshop on Intersection Types and Related Systems* (ITRS), volume 121 of *EPTCS*, pages 48–57, 2013.

[15] D. de Carvalho. *Sémantiques de la logique linéaire et temps de calcul*. These de doctorat, Université Aix-Marseille II, 2007.

[16] M. Florido and L. Damas. Linearization of the lambda-calculus and its relation with intersection type systems. *Journal Functional Programming* , 14(5):519–546, 2004.

[17] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1989.

[18] D. Kesner and D. Ventura. Quantitative types for intuitionistic calculi. Technical Report hal-00980868, Paris Cité Sorbonne, 2014.

[19] D. Kesner and D. Ventura. Quantitative types for the linear substitution calculus. In *IFIP Theoretical Computer Science (TCS)*, volume 8705 of *LNCS*, pages 296–310. Springer-Verlag, 2014.

[20] D. Kesner and D. Ventura. A resource aware computational interpretation for herbelins syntax. In *the 12th International Colloquium on Theoretical Aspects of Computing* (ICTAC), volume 9399 of *LNCS*, pages 388–403. Springer-Verlag, 2015.

[21] A. Kfoury. A linearization of the lambda-calculus and consequences. Technical report, Boston Universsity, 1996.

[22] A. Kfoury and J. B. Wells. Principality and type inference for intersection types using expansion variables. *Theoretical Computer Science*, 311(1-3):1–70, 2004.

[23] K. Kikuchi. Uniform proofs of normalisation and approximation for intersection types. In *7th Workshop on Intersection Types and Related Systems* (ITRS), 2014.

[24] J. W. Klop. *Combinatory reduction systems.* PhD thesis, Univ. Utrecht, 1980.

[25] J.-L. Krivine. *Lambda-calculus, types and models.* Ellis Horwood, 1993.

[26] D. Leivant. Polymorphic type inference. In *the 10th ACM Symposium on Principles of Programming Languages* (POPL), pages 88–98, 1983.

[27] P. M. Neergard. Theoretical pearls: A bargain for intersection types: a simple strong normalization proof. *Journal of Functional Programming*, 15:669–677, 9 2005.

[28] G. Pottinger. A type assignment for the strongly normalizable λ-terms. *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 561–578. Academic Press, 1980.

[29] L. Regnier. Une équivalence sur les lambda-termes. *Theoretical Computer Science* , 2(126):281–292, 1994.

[30] S. Ronchi Della Rocca. Principal type scheme and unification for intersection type discipline. *Theoretical Computer Science*, 59:1–29, 1988.

[31] W. Tait. Intensional interpretations of functionals of finite type I. *Journal Symbolic Logic*, 32(2):198–212, 1967.

[32] P. Urzyczyn. The emptiness problem for intersection types. *Journal of Symbolic Logic*, 64(3):1195–1215, 1999.

[33] S. Valentini. An elementary proof of strong normalization for intersection types. *Archive of Mathematical Logic*, 40(7):475–488, 2001.

[34] S. van Bakel. Complete restrictions of the intersection type discipline. *Theoretical Computer Science*, 102(1):135–163, 1992.

[35] S. van Bakel. Strict intersection types for the lambda calculus. *ACM Computing Surveys*, page 20, 2011.

[36] F. van Raamsdonk. *Confluence and Normalization for Higher-Order Rewriting.* PhD thesis, Amsterdam University, Netherlands, 1996.

[37] J. B. Wells. The essence of principal typings. In *the 29th International Colloquium on Automata, Languages and Programming* (ICALP), volume 2380 of *LNCS*, pages 913–925. Springer-Verlag, 2002.