

# Execution Time of $\lambda$ -Terms via Denotational Semantics and Intersection Types

DANIEL DE CARVALHO  
Università di Roma Tre

---

The multiset based relational model of linear logic induces a semantics of the type free  $\lambda$ -calculus, which corresponds to a non-idempotent intersection type system, System  $R$ . We prove that, in System  $R$ , the size of the type derivations and the size of the types are closely related to the execution time of  $\lambda$ -terms in a particular environment machine, Krivine's machine.

Categories and Subject Descriptors: F.3.2 [Logics and Meanings of Programs]: Semantics of Programming Languages—*Denotational semantics*; F.4.1 [Mathematical Logic and Formal Languages]: Mathematical Logic—*Lambda calculus and related systems*

Additional Key Words and Phrases: Computational complexity, denotational semantics, intersection types,  $\lambda$ -calculus

---

## 1. INTRODUCTION

This paper presents a work whose aim is to obtain information on execution time of  $\lambda$ -terms by semantic means.

By execution time, we mean the number of steps in a computational model. As in [Ehrhard and Regnier 2006], the computational model considered in this paper will be Krivine's machine, a more realistic model than  $\beta$ -reduction. Indeed, Krivine's machine implements (weak) head linear reduction: in one step, we can do at most one substitution. In this paper, we consider two variants of this machine : the first one (Definition 2.4) computes the head-normal form of any  $\lambda$ -term (if it exists) and the second one (Definition 2.11) computes the normal form of any  $\lambda$ -term (if it exists).

The fundamental idea of denotational semantics is that types should be interpreted as the objects of a category  $\mathbb{C}$  and terms should be interpreted as arrows in  $\mathbb{C}$  in such a way that if a term  $t$  reduces to a term  $t'$ , then they are interpreted by the same arrow. By the Curry-Howard isomorphism, a simply typed  $\lambda$ -term is a proof in intuitionistic logic and the  $\beta$ -reduction of a  $\lambda$ -term corresponds to the cut-elimination of a proof. Now, the intuitionistic fragment of linear logic [Girard 1987] is a refinement of intuitionistic logic. This means that when we have a categorical structure  $(\mathbb{C}, \dots)$  for interpreting intuitionistic linear logic, we can derive a category  $\mathbb{K}$  that is a denotational semantics of intuitionistic logic, and thus a denotational semantics of  $\lambda$ -calculus.

Linear logic has various denotational semantics; one of these is the multiset based relational model in the category **Rel** of sets and relations with the comonad associated to the finite multisets functor (see [Tortora de Falco 2000] for interpretations

of proof-nets and Appendix of [Bucciarelli and Ehrhard 2001] for interpretations of derivations of sequent calculus). In this paper, the category  $\mathbb{K}$  is a category equivalent to the Kleisli category of this comonad. The semantics we obtain is *non-uniform* in the following sense: the interpretation of a function contains information about its behaviour on chimerical arguments (see Example 3.8 for an illustration of this fact). As we want to consider type free  $\lambda$ -calculus, we will consider  $\lambda$ -algebras in  $\mathbb{K}$ . We will describe semantics of  $\lambda$ -terms in these  $\lambda$ -algebras as a logical system, using intersection types.

The intersection types system that we consider (System  $R$ , defined in Subsection 4.1) is a reformulation of that of [Coppo *et al.* 1980]; in particular, it lacks idempotency, as System  $\lambda$  in [Kfoury 2000] and System  $\mathbb{I}$  in [Neergaard and Mairson 2004] and contrary to System  $\mathcal{I}$  of [Kfoury *et al.* 1999]. So, we stress the fact that the semantics of [Coppo *et al.* 1980] can be reconstructed in a natural way from the finite multisets relational model of linear logic using the Kleisli construction.

If  $t'$  is a  $\lambda$ -term obtained by applying some reduction steps to  $t$ , then the semantics  $\llbracket t' \rrbracket$  of  $t'$  is the same as the semantics  $\llbracket t \rrbracket$  of  $t$ , so that from  $\llbracket t \rrbracket$ , it is clearly impossible to determine the number of reduction steps leading from  $t$  to  $t'$ . Nevertheless, if  $v$  and  $u$  are two closed normal  $\lambda$ -terms, we can wonder

- 1) Is it the case that the  $\lambda$ -term  $(v)u$  is (head) normalizable?
- 2) If the answer to the previous question is positive, what is the number of steps leading to the (principal head) normal form?

The main point of the paper is to show that it is possible to answer both questions by only referring to the semantics  $\llbracket v \rrbracket$  and  $\llbracket u \rrbracket$  of  $v$  and  $u$  respectively. The answer to the first question is given in Section 5 (Corollary 5.7) and it is a simple adaptation of well-known results. The answer to the second question is given in Section 6.

The paper [Ronchi Della Rocca 1988] presented a procedure that computes a normal form of any  $\lambda$ -term (if it exists) by finding its principal typing (if it exists). In Section 6, we present some quantitative results about the relation between the types and the computation of the (head) normal form. In particular, we prove that the number of steps of execution of a  $\lambda$ -term in the first machine (the one of Definition 2.4) is the size of the least type derivation of the  $\lambda$ -term in System  $R$  (Theorem 6.11) and we prove a similar result (Theorem 6.18) for the second machine (the one of Definition 2.11). We end by proving truly semantic measures of execution time in Subsection 6.4 and Subsection 6.5.

Note that even if this paper, a revised version of [de Carvalho 2006], concerns the  $\lambda$ -calculus and Krivine's machine, we emphasize connections with proof nets of linear logic. Due to these connections, we conjectured in [de Carvalho 2007] that we could obtain some similar results relating on the one hand the length of cut-elimination of nets with some specific strategy and on the other hand the size of the results of experiments. This specific strategy should be a strategy that mimics the one of Krivine's machine and that extends a strategy defined in [Mascari and Pedicini 1994] for a fragment of linear logic. This work has been done in [de Carvalho, Pagani and Tortora de Falco 2008] by adapting our work for the  $\lambda$ -calculus. But it is still difficult to compare both works, because the syntax of proof nets we considered makes that a cut-elimination step is not as elementary as a reduction step in Krivine's machine.

In conclusion, we believe that this work can be useful for implicit characterizations of complexity classes (in particular, the PTIME class, as in [Baillot and Terui (2004)]) by providing a semantic setting in which quantitative aspects can be studied, while taking some distance with the syntactic details.

In summary, Section 2 presents Krivine's machine, Section 3 the semantics we consider and Section 4 the intersection type system induced by this semantics, namely System  $R$ ; Section 5 gives the answer to question 1) and Section 6 the answer to question 2).

*Notations.* We denote by  $\Lambda$  the set of  $\lambda$ -terms, by  $\mathcal{V}$  the set of variables and, for any  $\lambda$ -term  $t$ , by  $FV(t)$  the set of free variables in  $t$ .

We use Krivine's notation for  $\lambda$ -terms: the  $\lambda$ -term  $v$  applied to the  $\lambda$ -term  $u$  is denoted by  $(v)u$ . We will also denote  $(\dots((v)u_1)u_2\dots)u_k$  by  $(v)u_1\dots u_k$ .

We use the notation  $[]$  for multisets while the notation  $\{\}$  is, as usual, for sets. For any set  $A$ , we denote by  $\mathcal{M}_{\text{fin}}(A)$  the set of finite multisets  $a$  whose support, denoted by  $\text{Supp}(a)$ , is a subset of  $A$ . For any set  $A$ , for any  $n \in \mathbb{N}$ , we denote by  $\mathcal{M}_n(A)$  the set of multisets of cardinality  $n$  whose support is a subset of  $A$ . The pairwise union of multisets given by term-by-term addition of multiplicities is denoted by a  $+$  sign and, following this notation, the generalized union is denoted by a  $\sum$  sign. The neutral element for this operation, the empty multiset, is denoted by  $[]$ .

For any set  $A$ , for any  $n \in \mathbb{N}$ , for any  $a \in \mathcal{M}_n(A)$ , we set

$$\mathfrak{S}(a) = \{(\alpha_1, \dots, \alpha_n) \in A^n / a = [\alpha_1, \dots, \alpha_n]\}.$$

## 2. KRIVINE'S MACHINE

We introduce two variants of a machine presented in [Krivine 2007] that implements call-by-name. More precisely, the original machine performs weak head linear reduction, whereas the machine presented in Subsection 2.2 performs head linear reduction. Subsection 2.3 slightly modifies the latter machine as to compute the  $\beta$ -normal form of any normalizable term.

### 2.1 Execution of States

We begin with the definitions of the set  $\mathcal{E}$  of environments and of the set  $\mathcal{C}$  of closures.

Set  $\mathcal{E} = \bigcup_{p \in \mathbb{N}} \mathcal{E}_p$  and set  $\mathcal{C} = \bigcup_{p \in \mathbb{N}} \mathcal{C}_p$ , where  $\mathcal{E}_p$  and  $\mathcal{C}_p$  are defined by induction on  $p$ :

- If  $p = 0$ , then  $\mathcal{E}_p = \{\emptyset\}$  and  $\mathcal{C}_p = \Lambda \times \{\emptyset\}$ .
- $\mathcal{E}_{p+1}$  is the set of partial maps  $\mathcal{V} \rightarrow \mathcal{C}_p$ , whose domain is finite, and  $\mathcal{C}_{p+1} = \Lambda \times \mathcal{E}_{p+1}$ .

For  $e \in \mathcal{E}$ , we denote by  $\mathbf{d}(e)$  the least integer  $p$  such that  $e \in \mathcal{E}_p$ .

For  $c = (t, e) \in \mathcal{C}$ , we define  $\bar{c} = t[e] \in \Lambda$  by induction on  $\mathbf{d}(e)$ :

- If  $\mathbf{d}(e) = 0$ , then  $t[e] = t$ .
- Assume  $t[e]$  defined for  $\mathbf{d}(e) = d$ . If  $\mathbf{d}(e) = d + 1$ , then  $t[e] = t[\bar{c}_1/x_1, \dots, \bar{c}_m/x_m]$ , with  $\{x_1, \dots, x_m\} = \text{dom}(e)$  and, for  $1 \leq j \leq m$ ,  $e(x_j) = c_j$ .

A *stack* is a finite sequence of closures. If  $c_0$  is a closure and  $\pi = (c_1, \dots, c_q)$  is a stack, then  $c_0.\pi$  will denote the stack  $(c_0, \dots, c_q)$ . We will denote by  $\epsilon$  the empty stack.

A *state* is a non-empty stack. If  $s = (c_0, \dots, c_q)$  is a state, then  $\overline{s}$  will denote the  $\lambda$ -term  $(\overline{c_0})\overline{c_1} \dots \overline{c_q}$ .

DEFINITION 2.1. *We say that a  $\lambda$ -term  $t$  respects the variable convention if any variable is bound at most once in  $t$ .*

*For any closure  $c = (t, e)$ , we define, by induction on  $d(e)$ , what it means for  $c$  to respect the variable convention:*

- if  $d(e) = 0$ , then we say that  $c$  respects the variable convention if, and only if,  $t$  respects the variable convention ;
- if  $c = (t, \{(x_1, c_1), \dots, (x_m, c_m)\})$  with  $m \neq 0$ , then we say that  $c$  respects the variable convention if, and only if,
  - $c_1, \dots, c_m$  respect the variable convention ;
  - and the variables  $x_1, \dots, x_m$  are not bound in  $t$ .

*For any state  $s = (c_0, \dots, c_q)$ , we say that  $s$  respects the variable convention if, and only if,  $c_0, \dots, c_q$  respect the variable convention.*

*We denote by  $\mathbb{S}$  the set of the states that respect the variable convention.*

First, we present the execution of a state (that respects the variable convention). It consists in updating a closure  $(t, e)$  and the stack. If  $t$  is an application  $(v)u$ , then we push the closure  $(u, e)$  on the top of the stack and the current closure is now  $(v, e)$ . If  $t$  is an abstraction, then a closure is popped and a new environment is created. If  $t$  is a variable, then the current closure is now the value of the variable of the environment. The partial map  $s \succ_{\mathbb{S}} s'$  (defined below) defines formally the transition from a state to another state.

DEFINITION 2.2. *We define a partial map from  $\mathbb{S}$  to  $\mathbb{S}$ : for any  $s, s' \in \mathbb{S}$ , the notation  $s \succ_{\mathbb{S}} s'$  will mean that the map assigns  $s'$  to  $s$ . The value of the map at  $s$  is defined as follows:*

- if  $s = (x, e).\pi$  and  $x \in \text{dom}(e)$ , then  $s \succ_{\mathbb{S}} e(x).\pi$ ;
- if  $s = (x, e).\pi$  and  $x \in \mathcal{V}$  and  $x \notin \text{dom}(e)$ , then the function is not defined at  $s$ ;
- if  $s = (\lambda x.u, e).(c.\pi')$ , then  $s \succ_{\mathbb{S}} (u, \{(x, c)\} \cup e).\pi'$ ;
- if  $s = (\lambda x.u, e).\epsilon$ , then the function is not defined at  $s$ ;
- if  $s = ((v)u, e).\pi$ , then  $s \succ_{\mathbb{S}} (v, e).((u, e).\pi)$ .

Note that in the case where the current subterm is an abstraction and the stack is empty, the machine stops: it does not reduce under lambda abstractions. That is why we slightly modify this machine in the following subsection.

## 2.2 A machine computing the principal head normal form

Now, the machine has to reduce under lambda abstractions and, in Subsection 2.3, the machine will have to compute the arguments of the head variable. So, we extend the machine so that it performs the reduction of elements of  $\mathcal{K}$ , where  $\mathcal{K} = \bigcup_{n \in \mathbb{N}} \mathcal{K}_n$  with

- $\mathcal{H}_0 = \mathcal{V}$  and  $\mathcal{K}_0 = \mathbb{S}$  ;
- $\mathcal{H}_{n+1} = \mathcal{V} \cup \{(v)u / v \in \mathcal{H}_n \text{ and } u \in \Lambda \cup \mathcal{K}_n\}$  and  
 $\mathcal{K}_{n+1} = \mathbb{S} \cup \mathcal{H}_n \cup \{\lambda y.k / y \in \mathcal{V} \text{ and } k \in \mathcal{K}_n\}$ .

Set  $\mathcal{H} = \bigcup_{n \in \mathbb{N}} \mathcal{H}_n$ . We have  $\mathcal{K} = \mathbb{S} \cup \mathcal{H} \cup \bigcup_{n \in \mathbb{N}} \{\lambda x.k / x \in \mathcal{V} \text{ and } k \in \mathcal{K}_n\}$ .

REMARK 2.3. *We have*

- $\mathcal{H} = \{(x)t_1 \dots t_p / p \in \mathbb{N}, x \in \mathcal{V}, t_1, \dots, t_p \in \Lambda \cup \mathcal{K}\}$
- *hence any element of  $\mathcal{K}$  can be written as either*

$$\lambda x_1 \dots \lambda x_m.s \text{ with } m \in \mathbb{N}, x_1, \dots, x_m \in \mathcal{V} \text{ and } s \in \mathbb{S}$$

*or*

$$\lambda x_1 \dots \lambda x_m.(x)t_1 \dots t_p \text{ with } m, p \in \mathbb{N}, x_1, \dots, x_m \in \mathcal{V} \text{ and } t_1, \dots, t_p \in \mathcal{K} \cup \Lambda.$$

For any  $k \in \mathcal{K}$ , we denote by  $\mathbf{d}(k)$  the least integer  $p$  such that  $k \in \mathcal{K}_p$ .

We extend the definition of  $\bar{s}$  for  $s \in \mathbb{S}$  to  $\bar{k}$  for  $k \in \mathcal{K}$ . For that, we set  $\bar{t} = t$  if  $t \in \Lambda$ . This definition is by induction on  $\mathbf{d}(k)$ :

- if  $\mathbf{d}(k) = 0$ , then  $k \in \mathbb{S}$  and thus  $\bar{k}$  is already defined;
- if  $k \in \mathcal{H}$ , then there are two cases:
  - if  $k \in \mathcal{V}$ , then  $\bar{k}$  is already defined (it is  $k$ ) ;
  - else,  $k = (v)u$  and we set  $\bar{k} = (\bar{v})\bar{u}$  ;
- if  $k = \lambda x.k_0$ , then  $\bar{k} = \lambda x.\bar{k}_0$ .

DEFINITION 2.4. *We define a partial map from  $\mathcal{K}$  to  $\mathcal{K}$ : for any  $k, k' \in \mathcal{K}$ , the notation  $k \succ_h k'$  will mean that the map assigns  $k'$  to  $k$ . The value of the map at  $k$  is defined, by induction on  $\mathbf{d}(k)$ , as follows:*

- if  $k \in \mathbb{S}$  and  $k \succ_{\mathbb{S}} s'$ , then  $k \succ_h s'$ ;
- if  $k = ((x, e), c_1, \dots, c_q) \in \mathbb{S}$  and  $x \in \mathcal{V}$  and  $x \notin \text{dom}(e)$ , then  $k \succ_h (x)\bar{c}_1 \dots \bar{c}_q$ ;
- if  $k = (\lambda x.u, e).e \in \mathbb{S}$ , then  $k \succ_h \lambda x.((u, e).e)$ ;
- if  $k \in \mathcal{H}$ , then the function is not defined at  $k$ ;
- if  $k = \lambda y.k_0$  and  $k_0 \succ_h k'_0$ , then  $k \succ_h \lambda y.k'_0$ .

A difference with the original machine is that our machine reduces under lambda abstractions.

We denote by  $\succ_h^*$  the reflexive transitive closure of  $\succ_h$ . For any  $k \in \mathcal{K}$ ,  $k$  is said to be a *Krivine normal form* if for any  $k' \in \mathcal{K}$ , we do not have  $k \succ_h k'$ .

DEFINITION 2.5. *For any  $k_0 \in \mathcal{K}$ , we define  $l_h(k_0) \in \mathbb{N} \cup \{\infty\}$  as follows: if there exist  $k_1, \dots, k_n \in \mathcal{K}$  such that  $k_i \succ_h k_{i+1}$  for  $0 \leq i \leq n-1$  and  $k_n$  is a Krivine normal form, then we set  $l_h(k_0) = n$ , else we set  $l_h(k_0) = \infty$ .*

PROPOSITION 2.6. *For any  $s \in \mathbb{S}$ , for any  $k' \in \mathcal{K}$ , if  $s \succ_h^* k'$  and  $k'$  is a Krivine normal form, then  $k'$  is a  $\lambda$ -term in head normal form.*

PROOF. By induction on  $l_h(s)$ .

The base case is trivial, because we never have  $l_h(s) = 0$ .

The inductive step is divided into five cases.

	output	current subterm	environment	stack
		$(\lambda x.(x)x)\lambda y.y$	$\emptyset$	$\epsilon$
1		$\lambda x.(x)x$	$\emptyset$	$(\lambda y.y, \emptyset)$
2		$(x)x$	$\{x \mapsto (\lambda y.y, \emptyset)\}$	$\epsilon$
3		$x$	$\{x \mapsto (\lambda y.y, \emptyset)\}$	$(x, \{x \mapsto (\lambda y.y, \emptyset)\})$
4		$\lambda y.y$	$\emptyset$	$(x, \{x \mapsto (\lambda y.y, \emptyset)\})$
5		$y$	$\{y \mapsto (x, \{x \mapsto (\lambda y.y, \emptyset)\})\}$	$\epsilon$
6		$x$	$\{x \mapsto (\lambda y.y, \emptyset)\}$	$\epsilon$
7		$\lambda y.y$	$\emptyset$	$\epsilon$
8	$\lambda y.$	$y$	$\emptyset$	$\epsilon$
9	$\lambda y.y$			

Fig. 1. Example of computation of the principal head normal form

- If  $s = ((x, e), c_1, \dots, c_q)$ ,  $x \in \mathcal{V}$  and  $x \notin \text{dom}(e)$ , then  $s \succ_h (x)\overline{c_1} \dots \overline{c_q}$ . But  $(x)\overline{c_1} \dots \overline{c_q}$  is a Krivine normal form and  $(x)\overline{c_1} \dots \overline{c_q}$  is a  $\lambda$ -term in head normal form.
- If  $s = (\lambda x.u, e).\epsilon$ , then  $k' = \lambda x.k''$  with  $(u, e).\epsilon \succ_h^* k''$ . Now, by induction hypothesis,  $k''$  is a  $\lambda$ -term in head normal form, hence  $k'$  too is a  $\lambda$ -term in head normal form.
- If  $s = ((x, e), c_1, \dots, c_q)$ ,  $x \in \mathcal{V}$  and  $x \in \text{dom}(e)$ , then  $s \succ_h (e(x), \pi)$ . Now,  $e(x).\pi \succ_h^* k'$ , hence, by induction hypothesis,  $k'$  is a  $\lambda$ -term in head normal form.
- If  $s = (\lambda x.u, e).(c.\pi)$ , then  $s \succ_h (u, \{(x, c)\} \cup e).\pi$ . Now,  $(u, \{(x, c)\} \cup e).\pi \succ_h k'$ , hence, by induction hypothesis,  $k'$  is a  $\lambda$ -term in head normal form.
- If  $s = ((v)u, e).\pi$ , then  $s \succ_h (v, e).((u, e).\pi)$ . Now,  $(v, e).((u, e).\pi) \succ_h^* k'$ , hence, by induction hypothesis,  $k'$  is a  $\lambda$ -term in head normal form.

□

EXAMPLE 2.7. Set  $s = (((\lambda x.(x)x)\lambda y.y, \emptyset), \epsilon)$ . We have  $l_h(s) = 9$ :

$$\begin{aligned}
s &\succ_h ((\lambda x.(x)x, \emptyset), (\lambda y.y, \emptyset)) \\
&\succ_h ((x)x, \{(x, (\lambda y.y, \emptyset))\}).\epsilon \\
&\succ_h ((x, \{(x, (\lambda y.y, \emptyset))\}), (x, \{(x, (\lambda y.y, \emptyset))\})) \\
&\succ_h ((\lambda y.y, \emptyset), (x, \{(x, (\lambda y.y, \emptyset))\})) \\
&\succ_h (y, \{(y, (x, \{(x, (\lambda y.y, \emptyset))\}))\}).\epsilon \\
&\succ_h (x, \{(x, (\lambda y.y, \emptyset))\}).\epsilon \\
&\succ_h (\lambda y.y, \emptyset).\epsilon \\
&\succ_h \lambda y.((y, \emptyset).\epsilon) \\
&\succ_h \lambda y.y
\end{aligned}$$

We present the same computation in a more descriptive way in Figure 1.

LEMMA 2.8. For any  $k, k' \in \mathcal{K}$ , if  $k \succ_h k'$ , then  $\overline{k} \rightarrow_h \overline{k'}$ , where  $\rightarrow_h$  is the reflexive closure of the head reduction.

PROOF. There are two cases.

- If  $k \in \mathbb{S}$ , then there are five cases.

- If  $k = ((x, e), c_1, \dots, c_q)$ ,  $x \in \mathcal{V}$  and  $x \notin \text{dom}(e)$ , then  $\bar{k} = (x)\bar{c}_1 \dots \bar{c}_q$  and  $\bar{k}' = (x)c_1 \dots c_q = (x)\bar{c}_1 \dots \bar{c}_q$ : we have  $\bar{k} = \bar{k}'$ .
- If  $k = (\lambda x.u, e). \epsilon$ , then  $\bar{k} = (\lambda x.u)[e] = \lambda x.u[e]$  (because  $k$  respects the variable convention) and  $\bar{k}' = \lambda x.((u, e), \epsilon) = \lambda x.u[e]$ : we have  $\bar{k} = \bar{k}'$ .
- If  $k = ((x, e), c_1, \dots, c_q)$ ,  $x \in \mathcal{V}$  and  $x \in \text{dom}(e)$ , then  $\bar{k} = \overline{e(x)\bar{c}_1 \dots \bar{c}_q}$  and  $\bar{k}' = \overline{(e(x), (c_1, \dots, c_q))} = \overline{e(x)\bar{c}_1 \dots \bar{c}_q}$ : we have  $\bar{k} = \bar{k}'$ .
- If  $k = ((\lambda x.u, e), c_0, \dots, c_q)$ , then  $\bar{k} = ((\lambda x.u)[e])\bar{c}_0 \dots \bar{c}_q = (\lambda x.u[e])\bar{c}_0 \dots \bar{c}_q$  (since  $k$  respects the variable convention) and  $\bar{k}' = \overline{((u, \{(x, c_0)\} \cup e))\bar{c}_1 \dots \bar{c}_q}$ . Now,  $\bar{k}$  reduces in a single head reduction step to  $\bar{k}'$ .
- If  $k = (((v)u, e), c_1 \dots c_q)$ , then  $\bar{k} = (((v)u)[e])\bar{c}_1 \dots \bar{c}_q = (v[e])u[e]\bar{c}_1 \dots \bar{c}_q$  and  $\bar{k}' = \overline{((v, e), (u, e), c_1, \dots, c_q)} = (v[e])u[e]\bar{c}_1 \dots \bar{c}_q$ : we have  $\bar{k} = \bar{k}'$ .
- Else,  $k = \lambda y.k_0$ ; then  $\bar{k} = \lambda y.\bar{k}_0$  and  $\bar{k}' = \lambda y.\bar{k}'_0 = \lambda y.\bar{k}_0$  with  $k_0 \succ_h k'_0$ : we have  $\bar{k}_0 \rightarrow_h \bar{k}'_0$ , hence  $\bar{k} \rightarrow_h \bar{k}'$ .

□

**THEOREM 2.9.** *For any  $k \in \mathcal{K}$ , if  $l_h(k)$  is finite, then  $\bar{k}$  is head normalizable.*

**PROOF.** By induction on  $l_h(k)$ .

If  $l_h(k) = 0$ , then  $k \in \mathcal{H}$ , hence  $k$  can be written as  $(x)t_1 \dots t_p$  and thus  $\bar{k}$  can be written  $(x)\bar{t}_1 \dots \bar{t}_p$ : it is a head normal form. Else, apply Lemma 2.8. □

For any head normalizable  $\lambda$ -term  $t$ , we denote by  $h(t)$  the number of head reductions of  $t$ .

**THEOREM 2.10.** *For any  $s = (t, e).\pi \in \mathbb{S}$ , if  $\bar{s}$  is head normalizable, then  $l_h(s)$  is finite.*

**PROOF.** We prove, by noetherian induction on  $\mathbb{N} \times \mathbb{N} \times \Lambda$  lexically ordered, that for any  $(h, d, t) \in \mathbb{N} \times \mathbb{N} \times \Lambda$ , for any  $s = (t, e).\pi$  such that  $h(s) = h$  and  $d(s) = d$ , if  $s$  is head-normalizable, then  $l_h(s)$  is finite.

If  $h(\bar{s}) = 0$ ,  $d(e) = 0$  and  $t \in \mathcal{V}$ , then we have  $l_h(s) = 1$ .

Else, there are five cases.

- In the case where  $t \in \text{dom}(e)$ , we have  $s \succ_h e(t).\pi$ . Set  $s' = e(t).\pi$  and  $e(t) = (t', e')$ . We have  $\bar{s} = \bar{s}'$  and  $d(e') < d(e)$ , hence we can apply the induction hypothesis:  $l_h(s')$  is finite and thus  $l_h(s) = l_h(s') + 1$  is finite.
- In the case where  $t \in \mathcal{V}$  and  $t \notin \text{dom}(e)$ , we have  $l_h(s) = 1$ .
- In the case where  $t = (v)u$ , we have  $s \succ_h (v, e).((u, e).\pi)$ . Set  $s' = (v, e).((u, e).\pi)$ . We have  $\bar{s}' = \bar{s}$  and thus we can apply the induction hypothesis:  $l_h(s')$  is finite and thus  $l_h(s) = l_h(s') + 1$  is finite.
- In the case where  $t = \lambda x.u$  and  $\pi = \epsilon$ , we have  $s \succ_h \lambda x.((u, e).\epsilon)$ . Set  $s' = (u, e).\epsilon$ . Since  $s$  respects the variable convention, we have  $\bar{s} = \lambda x.u[e] = \lambda x.\bar{s}'$ . We have  $h(\bar{s}') = h(\bar{s})$ , hence we can apply the induction hypothesis:  $l_h(s')$  is finite and thus  $l_h(s) = l_h(s') + 1$  is finite.
- In the case where  $t = \lambda x.u$  and  $\pi = c.\pi'$ , we have  $s \succ_h (u, \{(x, c)\} \cup e).\pi$ . Set  $s' = (u, \{(x, c)\} \cup e).\pi$ . We have  $h(\bar{s}') < h(\bar{s})$ , hence we can apply the induction hypothesis:  $l_h(s')$  is finite and thus  $l_h(s) = l_h(s') + 1$  is finite.

□

We recall that if a  $\lambda$ -term  $t$  has a head-normal form, then the last term of the terminating head reduction of  $t$  is called *the principal head normal form of  $t$*  (see [Barendregt 1984]). Proposition 2.6, Lemma 2.8 and Theorem 2.10 show that for any head normalizable  $\lambda$ -term  $t$  having  $t'$  as principal head normal form, we have  $(t, \emptyset). \epsilon \succ_h^* t'$  and  $t'$  is a Krivine head normal form.

### 2.3 A machine computing the $\beta$ -normal form

We now slightly modify the machine so as to compute the  $\beta$ -normal form of any normalizable  $\lambda$ -term.

**DEFINITION 2.11.** *We define a partial map from  $\mathcal{K}$  to  $\mathcal{K}$ : for any  $k, k' \in \mathcal{K}$ , the notation  $k \succ_\beta k'$  will mean that the map assigns  $k'$  to  $k$ . The value of the map at  $k$  is defined, by induction on  $d(k)$ , as follows:*

$$k \mapsto \begin{cases} s' & \text{if } k \in \mathbb{S} \text{ and } k \succ_{\mathbb{S}} s' \\ (x)(c_1.\epsilon) \dots (c_q.\epsilon) & \text{if } k = ((x, e), c_1, \dots, c_q) \in \mathbb{S}, x \in \mathcal{V} \text{ and } x \notin \text{dom}(e) \\ \lambda x.((u, e).\epsilon) & \text{if } k = ((\lambda x.u, e).\epsilon) \in \mathbb{S} \\ \text{not defined} & \text{if } k \in \mathcal{V} \\ (v')u & \text{if } k = (v)u \text{ and } v \succ_\beta v' \\ (x)u' & \text{if } k = (x)u \text{ with } x \in \mathcal{V} \text{ and } u \succ_\beta u' \\ \lambda y.k'_0 & \text{if } k = \lambda y.k_0 \text{ and } k_0 \succ_\beta k'_0 \end{cases}$$

Let us compare Definition 2.11 with Definition 2.4. The difference is in the case where the current subterm of a state is a variable and where this variable has no value in the environment: the first machine stops, the second machine continues to compute every argument of the variable.

The function  $l_\beta$  is defined as  $l_h$  (see Definition 2.5), but for this new machine.

For any normalizable  $\lambda$ -term  $t$ , we denote by  $n(t)$  the number of steps leading from  $t$  to its normal form following the leftmost reduction strategy.

**THEOREM 2.12.** *For any  $s = (t, e).\pi \in \mathbb{S}$ , if  $\bar{s}$  is normalizable, then  $l_\beta(s)$  is finite.*

**PROOF.** We prove, by noetherian induction on  $\mathbb{N} \times \mathbb{N} \times \Lambda$  lexicographically ordered, that for any  $(h, d, t) \in \mathbb{N} \times \mathbb{N} \times \Lambda$ , for any  $s = (t, e).\pi$  such that  $h(s) = h$  and  $d(s) = d$ , if  $s$  is head-normalizable, then  $l_h(s)$  is finite.

If  $n(\bar{s}) = 0$ ,  $\bar{s} \in \mathcal{V}$ ,  $d(e) = 0$  and  $t \in \mathcal{V}$ , then we have  $l_\beta(s) = 1$ .

Else, there are five cases.

- In the case where  $t \in \mathcal{V} \cap \text{dom}(e)$ , we have  $s \succ_\beta (e(t), \pi)$ . Set  $s' = (e(t), \pi)$  and  $e(t) = (t', e')$ . We have  $\bar{s} = \bar{s}'$  and  $d(e') < d(e)$ , hence we can apply the induction hypothesis:  $l_\beta(s')$  is finite and thus  $l_\beta(s) = l_\beta(s') + 1$  is finite.
- In the case where  $t \in \mathcal{V}$  and  $t \notin \text{dom}(e)$ , set  $\pi = (c_1, \dots, c_q)$ . For any  $k \in \{1, \dots, q\}$ , we have  $n(\bar{c}_k) \leq n(\bar{s})$  and  $\bar{c}_k < \bar{s}$ , hence we can apply the induction hypothesis on  $c_k$ : for any  $k \in \{1, \dots, q\}$ ,  $l_\beta(c_k)$  is finite, hence  $l_\beta(s) = \sum_{k=1}^q l_\beta(c_k) + 1$  is finite too.
- In the case where  $t = (v)u$ , we have  $s \succ_\beta (v, e).((u, e).\pi)$ . Set  $s' = (v, e).((u, e).\pi)$ . We have  $\bar{s}' = \bar{s}$ , hence we can apply the induction hypothesis:  $l_\beta(s')$  is finite and thus  $l_\beta(s) = l_\beta(s') + 1$  is finite.



- In the case where  $t = \lambda x.u$  and  $\pi = \epsilon$ , we have  $s \succ_\beta \lambda x.((u, e).\epsilon)$ . Set  $s' = (u, e).\epsilon$ . Since  $s$  respects the variable convention, we have  $\bar{s} = \lambda x.u[e] = \lambda x.\bar{s}'$ . We have  $n(\bar{s}') = n(\bar{s})$ , hence we can apply the induction hypothesis:  $l_\beta(s')$  is finite and thus  $l_\beta(s) = l_\beta(s') + 1$  is finite.
- In the case where  $t = \lambda x.u$  and  $\pi = c.\pi'$ , we have  $s \succ_\beta (u, \{(x, c)\} \cup e).\pi$ . Set  $s' = (u, \{(x, c)\} \cup e).\pi$ . We have  $n(\bar{s}') < n(\bar{s})$ , hence we can apply the induction hypothesis:  $l_\beta(s')$  is finite and thus  $l_\beta(s) = l_\beta(s') + 1$  is finite.

□

### 3. A NON-UNIFORM SEMANTICS OF $\lambda$ -CALCULUS

We define here the semantics allowing to measure execution time. We have in mind the following philosophy: the semantics of the untyped  $\lambda$ -calculus come from the semantics of the simply typed  $\lambda$ -calculus and any semantics of linear logic induces a semantics of the simply typed  $\lambda$ -calculus. So, we start from a semantics  $\mathfrak{M}$  of linear logic (Subsection 3.1), then we present the induced semantics  $\Lambda(\mathfrak{M})$  of the simply typed  $\lambda$ -calculus (Subsection 3.2) and lastly the semantics of the untyped  $\lambda$ -calculus that we consider (Subsection 3.3). This semantics is *non-uniform* in the sense that the interpretation of a function contains information about its behaviour on arguments whose value can change during the computation: in Subsection 3.4, we give an example illustrating this point.

The first works tackling the problem of giving a general categorical definition of a denotational semantics of linear logic are those of Lafont [Lafont 1988] and of Seely [Seely 1989]. As for the works of Benton, Bierman, Hyland and de Paiva, [Benton *et al.* 1994], [Bierman 1993] and [Bierman 1995], they led to the following axiomatic: a categorical model of the multiplicative exponential fragment of intuitionistic linear logic (IMELL) is a quadruple  $(\mathcal{C}, \mathcal{L}, c, w)$  such that

- $\mathcal{C} = (\mathbb{C}, \otimes, I, \alpha, \lambda, \rho, \gamma)$  is a closed symmetric monoidal category;
  - $\mathcal{L} = ((T, \mathfrak{m}, \mathfrak{n}), \delta, d)$  is a symmetric monoidal comonad on  $\mathcal{C}$ ;
  - $c$  is a monoidal natural transformation from  $(T, \mathfrak{m}, \mathfrak{n})$  to  $\otimes \circ \Delta_{\mathcal{C}} \circ (T, \mathfrak{m}, \mathfrak{n})$  and  $w$  is a monoidal natural transformation from  $(T, \mathfrak{m}, \mathfrak{n})$  to  $*_{\mathcal{C}}$  such that
    - for any object  $A$  of  $\mathbb{C}$ ,  $((T(A), \delta_A), c_A, w_A)$  is a cocommutative comonoid in  $(\mathbb{C}^{\mathbb{T}}, \otimes^{\mathbb{T}}, (I, \mathfrak{n}), \alpha, \lambda, \rho)$
    - and for any  $f \in \mathbb{C}^{\mathbb{T}}[(T(A), \delta_A), (T(B), \delta_B)]$ ,  $f$  is a comonoid morphism,
- where  $\mathbb{T}$  is the comonad  $(T, \delta, d)$  on  $\mathbb{C}$ ,  $\mathbb{C}^{\mathbb{T}}$  is the category of  $\mathbb{T}$ -coalgebras,  $\Delta_{\mathcal{C}}$  is the diagonal monoidal functor from  $\mathcal{C}$  to  $\mathcal{C} \times \mathcal{C}$  and  $*_{\mathcal{C}}$  is the monoidal functor that sends any arrow to  $id_I$ .

Given a categorical model  $\mathfrak{M} = (\mathcal{C}, \mathcal{L}, c, w)$  of IMELL with  $\mathcal{C} = (\mathbb{C}, \otimes, I, \alpha, \lambda, \rho, \gamma)$  and  $\mathcal{L} = ((T, \mathfrak{m}, \mathfrak{n}), \delta, d)$ , we can define a cartesian closed category  $\Lambda(\mathfrak{M})$  such that

- objects are finite sequences of objects of  $\mathbb{C}$
- and arrows  $(A_1, \dots, A_m) \rightarrow (B_1, \dots, B_p)$  are the sequences  $(f_1, \dots, f_p)$  such that every  $f_k$  is an arrow  $\bigotimes_{j=1}^m T(A_j) \rightarrow B_k$  in  $\mathbb{C}$ .

Hence we can interpret simply typed  $\lambda$ -calculus in the category  $\Lambda(\mathfrak{M})$ . This category is (weakly) equivalent<sup>1</sup> to a full subcategory of  $(T, \delta, d)$ -coalgebras exhibited by Hyland. If the category  $\mathbb{C}$  is cartesian, then the categories  $\Lambda(\mathfrak{M})$  and the Kleisli category of the comonad  $(T, \delta, d)$  are (strongly) equivalent<sup>2</sup>. See [de Carvalho 2007] for a full exposition.

Below, we describe completely the category  $\Lambda(\mathfrak{M})$  (with its composition operation and its identities) only for the particular case that we consider in this paper.

### 3.1 A relational model of linear logic

The category of sets and relations is denoted by **Rel** and its composition operation by  $\circ$ . The functor  $T$  from **Rel** to **Rel** is defined by setting

- for any object  $A$  of **Rel**,  $T(A) = \mathcal{M}_{\text{fin}}(A)$ ;
- and, for any  $f \in \mathbf{Rel}(A, B)$ ,  $T(f) \in \mathbf{Rel}(T(A), T(B))$  defined by

$$T(f) = \{([\alpha_1, \dots, \alpha_n], [\beta_1, \dots, \beta_n]) / n \in \mathbb{N} \text{ and } (\alpha_1, \beta_1), \dots, (\alpha_n, \beta_n) \in f\}.$$

The natural transformation  $d$  from  $T$  to the identity functor of **Rel** is defined by setting  $d_A = \{([\alpha], \alpha) / \alpha \in A\}$  and the natural transformation  $\delta$  from  $T$  to  $T \circ T$  by setting  $\delta_A = \{(a_1 + \dots + a_n, [a_1, \dots, a_n]) / n \in \mathbb{N} \text{ and } a_1, \dots, a_n \in T(A)\}$ . It is easy to show that  $(T, \delta, d)$  is a comonad on **Rel**. It is well-known that this comonad can be provided with a structure  $\mathfrak{M}$  that is a denotational semantics of (I)MELL.

This denotational semantics gives rise to a cartesian closed category  $\Lambda(\mathfrak{M})$ .

### 3.2 Interpreting simply typed $\lambda$ -terms

We give the complete description of the category  $\Lambda(\mathfrak{M})$  induced by the denotational semantics  $\mathfrak{M}$  of (I)MELL evoked in the previous subsection:

- objects are finite sequences of sets;
- arrows  $(A_1, \dots, A_m) \rightarrow (B_1, \dots, B_n)$  are the sequences  $(f_1, \dots, f_n)$  such that every  $f_i$  is a subset of  $(\prod_{j=1}^m \mathcal{M}_{\text{fin}}(A_j)) \times B_i$  with the convention  $(\prod_{j=1}^m \mathcal{M}_{\text{fin}}(A_j)) \times B_i = B_i$  if  $m = 0$ ;
- if  $(f_1, \dots, f_p)$  is an arrow  $(A_1, \dots, A_m) \rightarrow (B_1, \dots, B_p)$  and  $(g_1, \dots, g_q)$  is an arrow  $(B_1, \dots, B_p) \rightarrow (C_1, \dots, C_q)$ , then  $(g_1, \dots, g_q) \circ_{\Lambda(\mathfrak{M})} (f_1, \dots, f_p)$  is the arrow  $(h_1, \dots, h_q) : (A_1, \dots, A_m) \rightarrow (C_1, \dots, C_q)$ , where  $h_l$  is

$$\bigcup_{n_1, \dots, n_p \in \mathbb{N}} \left\{ \begin{array}{l} ((\sum_{k=1}^p \sum_{i=1}^{n_k} a_1^{i,k}, \dots, \sum_{k=1}^p \sum_{i=1}^{n_k} a_m^{i,k}), \gamma) / \\ (\forall j \in \{1, \dots, m\}) (\forall k \in \{1, \dots, p\}) (a_j^{i,k})_{1 \leq i \leq n_k} \in (\mathcal{M}_{\text{fin}}(A_j))^{n_k} \\ \text{and } (\exists \beta_1^1, \dots, \beta_1^{n_1} \in B_1) \dots (\exists \beta_p^1, \dots, \beta_p^{n_p} \in B_p) \\ ((([\beta_1^1, \dots, \beta_1^{n_1}], \dots, [\beta_p^1, \dots, \beta_p^{n_p}]), \gamma) \in g_l \text{ and} \\ (\forall k \in \{1, \dots, p\}) (\forall i \in \{1, \dots, n_k\}) ((a_1^{i,k}, \dots, a_m^{i,k}), \beta_k^i) \in f_k) \end{array} \right\}$$

<sup>1</sup>A category  $\mathbb{C}$  is said to be *weakly equivalent* to a category  $\mathbb{D}$  if there exists a functor  $F : \mathbb{C} \rightarrow \mathbb{D}$  full and faithful such that every object  $D$  of  $\mathbb{D}$  is isomorphic to  $F(C)$  for some object  $C$  of  $\mathbb{C}$ .

<sup>2</sup>A category  $\mathbb{C}$  is said to be *strongly equivalent* to a category  $\mathbb{D}$  if there are functors  $F : \mathbb{C} \rightarrow \mathbb{D}$  and  $G : \mathbb{D} \rightarrow \mathbb{C}$  and natural isomorphisms  $G \circ F \cong id_{\mathbb{C}}$  and  $F \circ G \cong id_{\mathbb{D}}$ .

for  $1 \leq l \leq q$ , with the conventions

$$((a_1, \dots, a_m), \gamma) = \gamma \text{ and } (\prod_{j=1}^m \mathcal{M}_{\text{fin}}(A_j)) \times C_l = C_l \text{ if } m = 0;$$

—the identity of  $(A_1, \dots, A_m)$  is  $(d^1, \dots, d^m)$  with

$$d^j = \{(\underbrace{(\square, \dots, \square)}_{j-1 \text{ times}}, [\alpha], \underbrace{(\square, \dots, \square)}_{m-j \text{ times}}), \alpha) / \alpha \in A_j\}.$$

PROPOSITION 3.1. *The category  $\Lambda(\mathfrak{M})$  has the following cartesian closed structure*

$$(\Lambda(\mathfrak{M}), 1, !, \&, \pi^1, \pi^2, (\cdot, \cdot)_{\mathfrak{M}}, \Rightarrow, \Lambda, ev) :$$

- the terminal object 1 is the empty sequence  $()$ ;
- if  $B^1 = (B_1, \dots, B_p)$  and  $B^2 = (B_{p+1}, \dots, B_{p+q})$  are two sequences of sets, then  $B^1 \& B^2$  is the sequence  $(B_1, \dots, B_{p+q})$ ;
- if  $B^1 = (B_1, \dots, B_p)$  and  $B^2 = (B_{p+1}, \dots, B_{p+q})$  are two sequences of sets, then

$$\pi_{B^1, B^2}^1 = (d^1, \dots, d^p) : B^1 \& B^2 \rightarrow B^1 \text{ in } \Lambda(\mathfrak{M})$$

and

$$\pi_{B^1, B^2}^2 = (d^{p+1}, \dots, d^{p+q}) : B^1 \& B^2 \rightarrow B^2 \text{ in } \Lambda(\mathfrak{M})$$

with

$$d^k = \{(\underbrace{(\square, \dots, \square)}_{k-1 \text{ times}}, [\beta], \underbrace{(\square, \dots, \square)}_{p+q-k \text{ times}}), \beta) / \beta \in B_k\};$$

- if  $f^1 = (f_1, \dots, f_p) : C \rightarrow A^1$  and  $f^2 = (f_{p+1}, \dots, f_{p+q}) : C \rightarrow A^2$  in  $\Lambda(\mathfrak{M})$ , then  $(f^1, f^2)_{\mathfrak{M}} = (f_1, \dots, f_{p+q}) : C \rightarrow A^1 \& A^2$ ;

and

- $(A_1, \dots, A_m) \Rightarrow (C_1, \dots, C_q)$  is defined by induction on  $m$ :
  - $() \Rightarrow (C_1, \dots, C_q) = (C_1, \dots, C_q)$

$$\begin{aligned} &(A_1, \dots, A_{m+1}) \Rightarrow (C_1, \dots, C_q) \\ &= ((A_1, \dots, A_m) \Rightarrow (\mathcal{M}_{\text{fin}}(A_{m+1}) \times C_1), \dots, \\ &\quad (A_1, \dots, A_m) \Rightarrow (\mathcal{M}_{\text{fin}}(A_{m+1}) \times C_q)); \end{aligned}$$

- if  $h = (h_1, \dots, h_q) : (A_1, \dots, A_m) \& (B_1, \dots, B_p) \rightarrow (C_1, \dots, C_q)$ , then

$$\Lambda_{(A_1, \dots, A_m), (C_1, \dots, C_q)}^{(B_1, \dots, B_p)}(h) : (A_1, \dots, A_m) \rightarrow (B_1, \dots, B_p) \Rightarrow (C_1, \dots, C_q)$$

is defined by induction on  $p$ :

- if  $p = 0$ , then  $\Lambda_{(A_1, \dots, A_m), (C_1, \dots, C_q)}^{(B_1, \dots, B_p)}(h) = h$ ;
- if  $p = 1$ , then there are two cases:
  - in the case  $m = 0$ ,  $\Lambda_{(A_1, \dots, A_m), (C_1, \dots, C_q)}^{(B_1, \dots, B_p)}(h) = h$ ;

—in the case  $m \neq 0$ ,

$$\Lambda_{(A_1, \dots, A_m), (C_1, \dots, C_q)}^{(B_1, \dots, B_p)}(h) = (\xi_{\prod_{j=1}^m \mathcal{M}_{fin}(A_j), C_1}^{\mathcal{M}_{fin}(B_1)}(h_1), \dots, \xi_{\prod_{j=1}^m \mathcal{M}_{fin}(A_j), C_q}^{\mathcal{M}_{fin}(B_1)}(h_q)),$$

where

$$\xi_{\prod_{j=1}^m \mathcal{M}_{fin}(A_j), C_l}^{\mathcal{M}_{fin}(B_1)}(h_l) = \{(a, (b, \gamma)) ; (a, b, \gamma) \in h_l\};$$

—if  $p \geq 1$ , then

$$\begin{aligned} & \Lambda_{A, (C_1, \dots, C_q)}^{(B_1, \dots, B_{p+1})}(h) \\ &= \Lambda_{A, (\mathcal{M}_{fin}(B_{p+1}) \times C_1, \dots, \mathcal{M}_{fin}(B_{p+1}) \times C_q)}^{(B_1, \dots, B_p)}(\Lambda_{(A_1, \dots, A_m, B_1, \dots, B_p), (C_1, \dots, C_q)}^{(B_{p+1})}(h)), \end{aligned}$$

where  $A = (A_1, \dots, A_m)$ ;

— $ev_{C,B} : (B \Rightarrow C) \& B \rightarrow C$  is defined by setting

$$ev_{(C_1, \dots, C_q), (B_1, \dots, B_p)} = (ev_{(C_1, \dots, C_q), (B_1, \dots, B_p)}^1, \dots, ev_{(C_1, \dots, C_q), (B_1, \dots, B_p)}^q)$$

where, for  $1 \leq k \leq q$ ,

$$\begin{aligned} & ev_{(C_1, \dots, C_q), (B_1, \dots, B_p)}^k \\ &= \left\{ \left( \underbrace{((\square, \dots, \square), [(b_1, \dots, b_p), \gamma])}_{k-1 \text{ times}}, \underbrace{(\square, \dots, \square, b_1, \dots, b_p), \gamma}_{q-k \text{ times}} \right) / \right. \\ & \quad \left. b_1 \in \mathcal{M}_{fin}(B_1), \dots, b_p \in \mathcal{M}_{fin}(B_p), \gamma \in C_k \right\}. \end{aligned}$$

PROOF. By checking some computations or by applying the theorem that states that if  $\mathfrak{M}$  is a denotational semantics of IMELL, then the "induced" structure

$$(\Lambda(\mathfrak{M}), 1, !, \&, \pi^1, \pi^2, (\cdot, \cdot)_{\mathfrak{M}}, \Rightarrow, \Lambda, ev)$$

is a cartesian closed structure (see [de Carvalho 2007]).  $\square$

### 3.3 Interpreting type free $\lambda$ -terms

First, we recall that if  $f : D \rightarrow C$  and  $g : C \rightarrow D$  are two arrows in a category  $\mathbb{C}$ , then  $f$  is a retraction of  $g$  in  $\mathbb{C}$  means that  $f \circ_{\mathbb{C}} g = id_C$  (see, for instance, [Mac Lane 1998]); it is also said that  $(g, f)$  is a retraction pair.

With the cartesian closed structure on  $\Lambda(\mathfrak{M})$ , we have a semantics of the simply typed  $\lambda$ -calculus (see, for instance, [Lambek and Scott, 1986]). Now, in order to have a semantics of the pure  $\lambda$ -calculus, it is therefore enough to have a *reflexive* object  $U$  of  $\Lambda(\mathfrak{M})$ , that is to say such that

$$(U \Rightarrow U) \triangleleft U,$$

that means that there exist  $s \in \Lambda(\mathfrak{M})[U \Rightarrow U, U]$  and  $r \in \Lambda(\mathfrak{M})[U, U \Rightarrow U]$  such that  $r \circ_{\Lambda(\mathfrak{M})} s$  is the identity on  $U \Rightarrow U$ ; in particular,  $(s, r)$  is a retraction pair. We will use the following lemma for exhibiting such a retraction pair.

LEMMA 3.2. *Let  $h : A \rightarrow B$  be an injection between sets. Consider the arrows  $g : \mathcal{M}_{fin}(A) \rightarrow B$  and  $f : \mathcal{M}_{fin}(B) \rightarrow A$  of the category **Rel** defined by  $g = \{([\alpha], h(\alpha)) / \alpha \in A\}$  and  $f = \{([h(\alpha)], \alpha) / \alpha \in A\}$ . Then  $(g) \in \Lambda(\mathfrak{M})((A), (B))$  and  $(f)$  is a retraction of  $(g)$  in  $\Lambda(\mathfrak{M})$ .*

PROOF. An easy computation shows that we have

$$\begin{aligned} (f) \circ_{\Lambda(\mathfrak{M})} (g) &= (f \circ T(g) \circ \delta_A) \\ &= (d_A). \end{aligned}$$

□

If  $D$  is a set, then  $(D) \Rightarrow (D) = (\mathcal{M}_{\text{fin}}(D) \times D)$ . From now on, we assume that  $D$  is a non-empty set and that  $h$  is an injection from  $\mathcal{M}_{\text{fin}}(D) \times D$  to  $D$ . Set

$$g = \{([\alpha], h(\alpha)) / \alpha \in \mathcal{M}_{\text{fin}}(D) \times D\} : \mathcal{M}_{\text{fin}}(\mathcal{M}_{\text{fin}}(D) \times D) \rightarrow D \text{ in } \mathbf{Rel}$$

and

$$f = \{([h(\alpha)], \alpha) / \alpha \in \mathcal{M}_{\text{fin}}(D) \times D\} : \mathcal{M}_{\text{fin}}(D) \rightarrow \mathcal{M}_{\text{fin}}(D) \times D \text{ in } \mathbf{Rel}.$$

We have

$$((D) \Rightarrow (D)) \triangleleft (D) \text{ in the category } \Lambda(\mathfrak{M})$$

and, more precisely:  $(g) \in \Lambda(\mathfrak{M})((D) \Rightarrow (D), (D))$  and  $(f)$  is a retraction of  $(g)$ .

We can therefore define the interpretation of any  $\lambda$ -term.

DEFINITION 3.3. For any  $\lambda$ -term  $t$  possibly containing constants from  $\mathcal{P}(D)$ , for any  $x_1, \dots, x_m \in \mathcal{V}$  distinct such that  $FV(t) \subseteq \{x_1, \dots, x_m\}$ , we define, by induction on  $t$ ,  $\llbracket t \rrbracket_{x_1, \dots, x_m} \subseteq (\prod_{j=1}^m \mathcal{M}_{\text{fin}}(D)) \times D$ :

- $\llbracket x_j \rrbracket_{x_1, \dots, x_m} = \{(\underbrace{[\ ], \dots, [\ ]}_{j-1 \text{ times}}, [\alpha], \underbrace{[\ ], \dots, [\ ]}_{m-j \text{ times}}), \alpha) / \alpha \in D\}$ ;
- for any  $c \in \mathcal{P}(D)$ ,  $\llbracket c \rrbracket_{x_1, \dots, x_m} = (\prod_{j=1}^m \mathcal{M}_{\text{fin}}(D)) \times c$ ;
- $\llbracket \lambda x. u \rrbracket_{x_1, \dots, x_m} = \{((a_1, \dots, a_m), h(a, \alpha)) / ((a_1, \dots, a_m, a), \alpha) \in \llbracket u \rrbracket_{x_1, \dots, x_m, x}\}$ ;
- the value of  $\llbracket (v)u \rrbracket_{x_1, \dots, x_m}$  is

$$\bigcup_{n \in \mathbb{N}} \bigcup_{\alpha_1, \dots, \alpha_n \in D} \left\{ \begin{aligned} &((\sum_{i=0}^n a_1^i, \dots, \sum_{i=0}^n a_m^i), \alpha) / \\ &((a_1^0, \dots, a_m^0), h([\alpha_1, \dots, \alpha_n], \alpha)) \in \llbracket v \rrbracket_{x_1, \dots, x_m} \\ &\text{and } (\forall i \in \{1, \dots, n\}) ((a_1^i, \dots, a_m^i), \alpha_i) \in \llbracket u \rrbracket_{x_1, \dots, x_m} \end{aligned} \right\};$$

with the conventions  $(\prod_{j=1}^m \mathcal{M}_{\text{fin}}(D)) \times D = D$  and  $((a_1, \dots, a_m), \alpha) = \alpha$  if  $m = 0$ .

Now, we can define the interpretation of any  $\lambda$ -term in any environment.

DEFINITION 3.4. For any  $\rho \in \mathcal{P}(D)^\mathcal{V}$  and for any  $\lambda$ -term  $t$  possibly containing constants from  $\mathcal{P}(D)$  such that  $FV(t) = \{x_1, \dots, x_m\}$ , we set

$$\llbracket t \rrbracket_\rho = \bigcup_{a_1 \in \mathcal{M}_{\text{fin}}(\rho(x_1)), \dots, a_m \in \mathcal{M}_{\text{fin}}(\rho(x_m))} \{\alpha \in D / ((a_1, \dots, a_m), \alpha) \in \llbracket t \rrbracket_{x_1, \dots, x_m}\}.$$

For any  $d_1, d_2 \in \mathcal{P}(D)$ , we set

$$d_1 * d_2 = \bigcup_{a \in \mathcal{M}_{\text{fin}}(d_2)} \{\alpha \in D / h(a, \alpha) \in d_1\}.$$

We have

PROPOSITION 3.5. The triple  $(\mathcal{P}(D), *, \llbracket - \rrbracket_-)$  is a  $\lambda$ -algebra.

PROOF. Apply our Proposition 3.1 and Lemma 3.2, and Theorem 5.5.6 of [Barendregt 1984].  $\square$

But the following proposition, a corollary of Proposition 3.7, states that the triple  $(\mathcal{P}(D), *, \llbracket - \rrbracket_-)$  is *not* a  $\lambda$ -model. We recall (see, for instance, [Barendregt 1984]), that a  $\lambda$ -model is a  $\lambda$ -algebra  $(\mathcal{D}, *, \llbracket - \rrbracket_-)$  such that the following property, expressing the  $\xi$ -rule, holds:

for any  $\rho \in \mathcal{D}^\mathcal{V}$ , for any  $x \in \mathcal{V}$  and for any  $\lambda$ -terms  $t_1$  and  $t_2$ , we have

$$((\forall d \in \mathcal{D}) \llbracket t_1 \rrbracket_{\rho[x:=d]} = \llbracket t_2 \rrbracket_{\rho[x:=d]} \Rightarrow \llbracket \lambda x. t_1 \rrbracket_\rho = \llbracket \lambda x. t_2 \rrbracket_\rho).$$

PROPOSITION 3.6. *The  $\lambda$ -algebra  $(\mathcal{P}(D), *, \llbracket - \rrbracket_-)$  is not a  $\lambda$ -model.*

*In other words, there exist  $\rho \in \mathcal{P}(D)^\mathcal{V}$ ,  $x \in \mathcal{V}$  and two  $\lambda$ -terms  $t_1$  and  $t_2$  such that*

$$((\forall d \in \mathcal{P}(D)) \llbracket t_1 \rrbracket_{\rho[x:=d]} = \llbracket t_2 \rrbracket_{\rho[x:=d]} \text{ and } \llbracket \lambda x. t_1 \rrbracket_\rho \neq \llbracket \lambda x. t_2 \rrbracket_\rho).$$

In particular,  $\llbracket t \rrbracket_\rho$  *can not* be defined by induction on  $t$  (an interpretation by polynomials is nevertheless possible in such a way that the  $\xi$ -rule holds - see [Selinger 2002]).

Before stating Proposition 3.7, we recall that any object  $A$  of any category  $\mathbb{K}$  with a terminal object is said *to have enough points* if for any terminal object  $1$  of  $\mathbb{K}$  and for any  $y, z \in \mathbb{K}(A, A)$ , we have  $((\forall x \in \mathbb{K}(1, A)) y \circ_{\mathbb{K}} x = z \circ_{\mathbb{K}} x \Rightarrow y = z)$ .

Remark: it does not follow necessarily that the same holds for any  $y, z \in \mathbb{K}(A, B)$ .

PROPOSITION 3.7. *Let  $A$  be a non-empty set. Then  $(A)$  does not have enough points in  $\Lambda(\mathfrak{M})$ .*

PROOF. Let  $\alpha \in A$ . Let  $y$  and  $z$  be the arrows  $\mathcal{M}_{\text{fin}}(A) \rightarrow A$  of the category **Rel** defined by  $y = \{([\alpha], \alpha)\}$  and  $z = \{([\alpha, \alpha], \alpha)\}$ . Then  $(y)$  and  $(z)$  are two arrows  $(A) \rightarrow (A)$  of the category  $\Lambda(\mathfrak{M})$ .

We recall that the terminal object in  $\Lambda(\mathfrak{M})$  is the empty sequence  $()$ . Now, for any arrow  $x : () \rightarrow (A)$  of the category  $\Lambda(\mathfrak{M})$ , we have  $(y) \circ_{\Lambda(\mathfrak{M})} x = (z) \circ_{\Lambda(\mathfrak{M})} x$ .  $\square$

This proposition explains *why* Proposition 3.6 holds. A more direct proof of Proposition 3.6 can be obtained by considering the two  $\lambda$ -terms  $t_1 = (y)x$  and  $t_2 = (z)x$  with  $\rho(y) = \{([\alpha], \alpha)\}$  and  $\rho(z) = \{([\alpha, \alpha], \alpha)\}$ .

### 3.4 Non-uniformity

Example 3.8 illustrates the non-uniformity of the semantics. It is based on the following idea.

Consider the program

```

 $\lambda x.$ if  $x$  then 1
      else if  $x$  then 1
      else 0

```

applied to a boolean. The second **then** is never read. A *uniform* semantics would ignore it. It is not the case when the semantics is *non-uniform*.

EXAMPLE 3.8. *Set  $\mathbf{0} = \lambda x. \lambda y. y$  and  $\mathbf{1} = \lambda x. \lambda y. x$ . Assume that  $h$  is the inclusion from  $\mathcal{M}_{\text{fin}}(D) \times D$  to  $D$ .*

*Let  $\gamma \in D$ ; set  $\delta = ([\gamma], ([\gamma], \gamma))$  and  $\beta = ([\gamma], ([\gamma], \gamma))$ . We have*

- $([(\square, ([\delta], \delta))], ([\delta], \delta)) \in \llbracket (x)\mathbf{1} \rrbracket_x$ ;
- and  $([(\square, ([\delta], \delta))], \delta) \in \llbracket (x)\mathbf{10} \rrbracket_x$ .

Hence we have  $\alpha_1 = ((\square, ([\delta], \delta)), (\square, ([\delta], \delta)), \delta) \in \llbracket \lambda x.(x)\mathbf{1}(x)\mathbf{10} \rrbracket$ .

We have

- $([(\square, ([\beta], \beta))], ([\beta], \beta)) \in \llbracket (x)\mathbf{1} \rrbracket_x$ ;
- and  $([(\square, ([\beta], \beta))], \beta) \in \llbracket (x)\mathbf{10} \rrbracket_x$ .

Hence we have  $\alpha_2 = ((\square, ([\beta], \beta)), ([\beta], (\square, \beta)), \beta) \in \llbracket \lambda x.(x)\mathbf{1}(x)\mathbf{10} \rrbracket$ .

In a uniform semantics (as in [Girard 1986]), the point  $\alpha_1$  would appear in the semantics of this  $\lambda$ -term, but not the point  $\alpha_2$ , because  $[(\square, ([\beta], \beta)), ([\beta], (\square, \beta))]$  corresponds to a chimerical argument: the argument is read twice and provides two contradictory values.

#### 4. NON-IDEMPOTENT INTERSECTION TYPES

From now on,  $D = \bigcup_{n \in \mathbb{N}} D_n$ , where  $D_n$  is defined by induction on  $n$ :  $D_0$  is a non-empty set  $A$  that does not contain any pairs and  $D_{n+1} = A \cup (\mathcal{M}_{\text{fin}}(D_n) \times D_n)$ . We have  $D = A \dot{\cup} (\mathcal{M}_{\text{fin}}(D) \times D)$ , where  $\dot{\cup}$  is the disjoint union; the injection  $h$  from  $\mathcal{M}_{\text{fin}}(D) \times D$  to  $D$  will be the inclusion. Hence any element of  $D$  can be written  $a_1 \dots a_m \alpha$ , where  $a_1, \dots, a_m \in \mathcal{M}_{\text{fin}}(D)$ ,  $\alpha \in D$  and  $a_1 \dots a_m \alpha$  is defined by induction on  $m$ :

- $a_1 \dots a_0 \alpha = \alpha$ ;
- $a_1 \dots a_{m+1} \alpha = (a_1 \dots a_m, (a_{m+1}, \alpha))$ .

For any  $\alpha \in D$ , we denote by  $\text{depth}(\alpha)$  the least integer  $n$  such that  $\alpha \in D_n$ .

In the preceding section, we defined the semantics we consider (Definitions 3.3 and 3.4). Now, we want to describe this semantics as a logical system: the elements of  $D$  are viewed as propositional formulas. More precisely, a comma separating a multiset of types and a type is understood as an arrow and a non-empty multiset is understood as the conjunction of its elements (their intersection). Note that this means we are considering a commutative (but not necessarily idempotent) intersection.

##### 4.1 System $R$

A *context*  $\Gamma$  is a function from  $\mathcal{V}$  to  $\mathcal{M}_{\text{fin}}(D)$  such that  $\{x \in \mathcal{V} / \Gamma(x) \neq \square\}$  is finite. If  $x_1, \dots, x_m \in \mathcal{V}$  are distinct and  $a_1, \dots, a_m \in \mathcal{M}_{\text{fin}}(D)$ , then  $x_1 : a_1, \dots, x_m : a_m$  denotes the context defined by  $x \mapsto \begin{cases} a_j & \text{if } x = x_j; \\ \square & \text{else.} \end{cases}$  We denote by  $\Phi$  the set of contexts. We define the following binary operation on  $\Phi$ :

$$\begin{aligned} \Phi \times \Phi &\rightarrow \Phi \\ (\Gamma_1, \Gamma_2) &\mapsto \Gamma_1 + \Gamma_2 : \begin{array}{l} \mathcal{V} \rightarrow \mathcal{M}_{\text{fin}}(D) \\ x \mapsto \Gamma_1(x) + \Gamma_2(x), \end{array} \end{aligned}$$

where the second  $+$  denotes the sum of multisets given by term-by-term addition of multiplicities. Note that this operation is associative and commutative. Typing rules concern judgements of the form  $\Gamma \vdash_R t : \alpha$ , where  $\Gamma \in \Phi$ ,  $t$  is a  $\lambda$ -term and  $\alpha \in D$ .

DEFINITION 4.1. *The typing rules of System R are the following:*

$$\frac{}{x : [\alpha] \vdash_R x : \alpha}$$

$$\frac{\Gamma, x : a \vdash_R v : \alpha}{\Gamma \vdash_R \lambda x.v : (a, \alpha)}$$

$$\frac{\Gamma_0 \vdash_R v : ([\alpha_1, \dots, \alpha_n], \alpha) \quad \Gamma_1 \vdash_R u : \alpha_1, \dots, \Gamma_n \vdash_R u : \alpha_n}{\Gamma_0 + \Gamma_1 + \dots + \Gamma_n \vdash_R (v)u : \alpha} \quad n \in \mathbb{N}$$

The typing rule of the application has  $n + 1$  premisses. In particular, in the case where  $n = 0$ , we obtain the following rule:  $\frac{\Gamma_0 \vdash_R v : ([], \alpha)}{\Gamma_0 \vdash_R (v)u : \alpha}$  for any  $\lambda$ -term  $u$ . So, the empty multiset plays the role of the universal type  $\Omega$ .

The intersection we consider is *not* idempotent in the following sense: if a closed  $\lambda$ -term  $t$  has the type  $a_1 \dots a_m \alpha$  and, for  $1 \leq j \leq m$ ,  $\text{Supp}(a'_j) = \text{Supp}(a_j)$ , it does not follow necessarily that  $t$  has the type  $a'_1 \dots a'_m \alpha$ . For instance, the  $\lambda$ -term  $\lambda z. \lambda x. (z)x$  has types  $([[[\alpha], \alpha]], ([\alpha], \alpha))$  and  $([[[\alpha, \alpha], \alpha]], ([\alpha, \alpha], \alpha))$  but not the type  $([[[\alpha], \alpha]], ([\alpha, \alpha], \alpha))$ . On the contrary, the system presented in [Ronchi Della Rocca 1988] and the System  $\mathcal{D}$  presented in [Krivine 1990] consider an idempotent intersection. System  $\lambda$  of [Kfoury 2000] and System  $\mathbb{I}$  of [Neergaard and Mairson 2004] consider a non-idempotent intersection, but the treatment of weakening is not the same.

Interestingly, System  $R$  can be seen as a reformulation of the system of [Coppo *et al.* 1980]. More precisely, types of System  $R$  correspond to their normalized types.

## 4.2 Relating types and semantics

We prove in this subsection that the semantics of a closed  $\lambda$ -term as defined in Subsection 3.3 is the set of its types in System  $R$ . The following assertions relate more precisely types and semantics of any  $\lambda$ -term.

THEOREM 4.2. *For any  $\lambda$ -term  $t$  such that  $FV(t) \subseteq \{x_1, \dots, x_m\}$ , we have*

$$\llbracket t \rrbracket_{x_1, \dots, x_m} = \{a_1 \dots a_m \alpha \in (\prod_{j=1}^m \mathcal{M}_{fin}(D)) \times D \mid x_1 : a_1, \dots, x_m : a_m \vdash_R t : \alpha\}.$$

PROOF. By induction on  $t$ .  $\square$

COROLLARY 4.3. *For any  $\lambda$ -terms  $t$  and  $t'$  such that  $t =_\beta t'$ , if  $\Gamma \vdash_R t : \alpha$ , then we have  $\Gamma \vdash_R t' : \alpha$ .*

PROOF. By our Proposition 3.1 and Lemma 3.2, and Proposition 5.5.5 of [Barendregt 1984], the following property holds: for any  $\lambda$ -terms  $t$  and  $t'$  such that  $t =_\beta t'$  and such that  $FV(t) \subseteq \{x_1, \dots, x_m\}$ , we have  $\llbracket t \rrbracket_{x_1, \dots, x_m} = \llbracket t' \rrbracket_{x_1, \dots, x_m}$ .  $\square$

THEOREM 4.4. *For any  $\lambda$ -term  $t$  and for any  $\Gamma \in \Phi$ , we have*

$$\begin{aligned} & \{\alpha \in D \mid \Gamma \vdash_R t : \alpha\} \\ & \subseteq \{\alpha \in D \mid (\forall \rho \in \mathcal{P}(D)^\mathcal{V})((\forall x \in \mathcal{V}) \Gamma(x) \in \mathcal{M}_{fin}(\rho(x)) \Rightarrow \alpha \in \llbracket t \rrbracket_\rho)\}. \end{aligned}$$

PROOF. Apply Theorem 4.2.  $\square$



REMARK 4.5. *The reverse inclusion is not true.*

THEOREM 4.6. *For any  $\lambda$ -term  $t$  and for any  $\rho \in \mathcal{P}(D)^\mathcal{V}$ , we have*

$$\llbracket t \rrbracket_\rho = \{ \alpha \in D \mid (\exists \Gamma \in \Phi) ((\forall x \in \mathcal{V}) \Gamma(x) \in \mathcal{M}_{fin}(\rho(x)) \text{ and } \Gamma \vdash_R t : \alpha) \} .$$

PROOF. Apply Theorems 4.2 and 4.4.  $\square$

There is another way to compute the interpretation of  $\lambda$ -terms in this semantics. Indeed, it is well-known that we can translate  $\lambda$ -terms into linear logic nets labelled with the types  $I$ ,  $O$ ,  $?I$  and  $!O$  (as in [Regnier 1992]): this translation is defined by induction on the  $\lambda$ -terms. Now, we can do *experiments* (in the sense of [Girard 1987], that introduced this notion in the framework of coherent semantics for working with proof-nets directly, without sequentializing) to compute the semantics of the net in the multiset based relational model: all the translations corresponding to the encoding  $A \Rightarrow B \equiv ?A^\perp \wp B$  have the same semantics. And this semantics is the same as the semantics defined here.

For a survey of translations of  $\lambda$ -terms in proof nets, see [Guerrini 2004].

### 4.3 An equivalence relation on derivations

Definition 4.8 introduces an equivalence relation on the set of derivations of a given  $\lambda$ -term. This relation, as well as the notion of substitution defined immediately after, will play a role in Subsection 6.5.

DEFINITION 4.7. *For any  $\lambda$ -term  $t$ , for any  $(\Gamma, \alpha) \in \Phi \times D$ , we denote by  $\Delta(t, (\Gamma, \alpha))$  the set of derivations of  $\Gamma \vdash_R t : \alpha$ .*

*For any  $\lambda$ -term  $t$ , we set  $\Delta(t) = \bigcup_{(\Gamma, \alpha) \in \Phi \times D} \Delta(t, (\Gamma, \alpha))$ .*

*For any closed  $\lambda$ -term  $t$ , for any  $\alpha \in D$ , we denote by  $\Delta(t, \alpha)$  the set of derivations of  $\vdash_R t : \alpha$ .*

*For any closed  $\lambda$ -term  $t$ , for any integer  $n$ , for any  $a \in \mathcal{M}_n(D)$ , we set*

$$\Delta(t, a) = \bigcup_{(\alpha_1, \dots, \alpha_n) \in \mathfrak{S}(a)} \{ (\Pi_1, \dots, \Pi_n) \in \Delta(t)^n \mid (\forall i \in \{1, \dots, n\}) \Pi_i \in \Delta(t, \alpha_i) \} .$$

*We set  $\Delta = \bigcup_{t \in \Lambda} \Delta(t)$ .*

DEFINITION 4.8. *Let  $t$  be a  $\lambda$ -term. For any  $\Pi, \Pi' \in \Delta(t)$ , we define, by induction on  $\Pi$ , when  $\Pi \sim \Pi'$  holds:*

—if  $\Pi$  is only a leaf, then  $\Pi \sim \Pi'$  if, and only if,  $\Pi'$  is a leaf too;

—if  $\Pi = \frac{\Pi_0}{\Gamma, x : a \vdash_R v : \alpha}$ , then  $\Pi \sim \Pi'$  if, and only if, there exists  $\Pi'_0 \sim \Pi_0$  such that  $\Pi' = \frac{\Pi'_0}{\Gamma \vdash_R \lambda x.v : (a, \alpha)}$ ;

—if  $\Pi' = \frac{\Pi'_0}{\Gamma', x : a' \vdash_R v : \alpha'}$ ;

—if

$$\Pi = \frac{\frac{\Pi_0}{\Gamma_0 \vdash_R v : ([\alpha_1, \dots, \alpha_n], \alpha)} \quad \frac{\Pi_1 \quad \dots \quad \Pi_n}{\Gamma_1 \vdash_R u : \alpha_1 \quad \dots \quad \Gamma_n \vdash_R u : \alpha_n}}{\Gamma_0 + \Gamma_1 + \dots + \Gamma_n \vdash_R (v)u : \alpha} ,$$

then  $\Pi \sim \Pi'$  if, and only if, there exist  $\Pi_0 \sim \Pi'_0$ ,  $\sigma \in \mathfrak{S}_n$ ,  $\Pi_1 \sim \Pi'_{\sigma(1)}, \dots, \Pi_n \sim \Pi'_{\sigma(n)}$  such that

$$\Pi' = \frac{\frac{\Pi'_0}{\Gamma'_0 \vdash_R v : ([\alpha'_1, \dots, \alpha'_n], \alpha')} \quad \frac{\Pi'_1 \quad \dots \quad \Pi'_n}{\Gamma'_1 \vdash_R u : \alpha'_1 \quad \dots \quad \Gamma'_n \vdash_R u : \alpha'_n}}{\Gamma'_0 + \Gamma'_1 + \dots + \Gamma'_n \vdash_R (v)u : \alpha'}.$$

An equivalence class of derivations of a  $\lambda$ -term  $t$  in System  $R$  can be seen as a *simple resource term of the shape of  $t$*  that does not reduce to 0. *Resource  $\lambda$ -calculus* is defined in [Ehrhard and Regnier 2006] and is similar to resource oriented versions of the  $\lambda$ -calculus previously introduced and studied in [Boudol *et al.* 1999] and [Kfoury 2000]. For a full exposition of a precise relation between this equivalence relation and simple resource terms, see [de Carvalho 2007].

DEFINITION 4.9. A substitution  $\sigma$  is a function from  $D$  to  $D$  such that

$$\text{for any } \alpha, \alpha_1, \dots, \alpha_n \in D, \sigma([\alpha_1, \dots, \alpha_n], \alpha) = ([\sigma(\alpha_1), \dots, \sigma(\alpha_n)], \sigma(\alpha)).$$

We denote by  $\mathcal{S}$  the set of substitutions.

For any  $\sigma \in \mathcal{S}$ , we denote by  $\bar{\sigma}$  the function from  $\mathcal{M}_{fin}(D)$  to  $\mathcal{M}_{fin}(D)$  defined by  $\bar{\sigma}([\alpha_1, \dots, \alpha_n]) = [\sigma(\alpha_1), \dots, \sigma(\alpha_n)]$ .

PROPOSITION 4.10. Let  $\Pi$  be a derivation of  $\Gamma \vdash_R t : \alpha$  and let  $\sigma$  be a substitution. Then there exists a derivation  $\Pi'$  of  $\bar{\sigma} \circ \Gamma \vdash_R t : \sigma(\alpha)$  such that  $\Pi \sim \Pi'$ .

PROOF. By induction on  $t$ .  $\square$

## 5. QUALITATIVE RESULTS

In this section, inspired by [Krivine 1990], we prove Theorem 5.6, which formulates *qualitative* relations between assignable types and normalization properties: it characterizes the (head) normalizable  $\lambda$ -terms by semantic means. We also answer to the following question: if  $v$  and  $u$  are two closed normal  $\lambda$ -terms, is it the case that  $(v)u$  is (head) normalizable? The answer is given only referring to  $\llbracket v \rrbracket$  and  $\llbracket u \rrbracket$  in Corollary 5.7. Quantitative versions of this last result will be proved in Section 6.

DEFINITION 5.1. For any  $n \in \mathbb{N}$ , we define, by induction on  $n$ ,  $D_n^{\text{ex}}$  and  $\overline{D_n^{\text{ex}}}$ :

$$\begin{aligned} - D_0^{\text{ex}} &= \overline{D_0^{\text{ex}}} = A; \\ - D_{n+1}^{\text{ex}} &= A \cup (\mathcal{M}_{fin}(\overline{D_n^{\text{ex}}}) \times D_n^{\text{ex}}) \text{ and } \overline{D_{n+1}^{\text{ex}}} = A \cup ((\mathcal{M}_{fin}(D_n^{\text{ex}}) \setminus \{\llbracket \rrbracket\}) \times \overline{D_n^{\text{ex}}}). \end{aligned}$$

We set

$$\begin{aligned} - D^{\text{ex}} &= \bigcup_{n \in \mathbb{N}} D_n^{\text{ex}}; \\ - \overline{D^{\text{ex}}} &= \bigcup_{n \in \mathbb{N}} \overline{D_n^{\text{ex}}}; \\ - \text{and } \Phi^{\text{ex}} &= \{\Gamma \in \Phi / (\forall x \in \mathcal{V}) \Gamma(x) \in \mathcal{M}_{fin}(\overline{D^{\text{ex}}})\}. \end{aligned}$$

Note that  $D^{\text{ex}}$  is the set of the  $\alpha \in D$  such that  $\llbracket \rrbracket$  has no positive occurrences in  $\alpha$ .

PROPOSITION 5.2. (i) Every head-normalizable  $\lambda$ -term is typable in System  $R$ .

- (ii) For any normalizable  $\lambda$ -term  $t$ , there exists  $(\Gamma, \alpha) \in \Phi^{\text{ex}} \times D^{\text{ex}}$  such that  $\Gamma \vdash_R t : \alpha$ .

PROOF. (i) Let  $t$  be a head-normalizable  $\lambda$ -term. There exists a  $\lambda$ -term of the shape  $(\lambda x_1 \dots \lambda x_k.t)v_1 \dots v_n$  such that  $(\lambda x_1 \dots \lambda x_k.t)v_1 \dots v_n =_\beta x$ . Now,  $x$  is typable. Therefore, by Corollary 4.3, the  $\lambda$ -term  $(\lambda x_1 \dots \lambda x_k.t)v_1 \dots v_n$  is typable. Hence  $\lambda x_1 \dots \lambda x_k.t$  is typable.

- (ii) We prove, by induction on  $t$ , that for any normal  $\lambda$ -term  $t$ , the following properties hold:
- there exists  $(\Gamma, \alpha) \in \Phi^{\text{ex}} \times D^{\text{ex}}$  such that  $\Gamma \vdash_R t : \alpha$ ;
  - if, moreover,  $t$  does not begin with  $\lambda$ , then, for any  $\alpha \in D^{\text{ex}}$ , there exists  $\Gamma \in \Phi^{\text{ex}}$  such that  $\Gamma \vdash_R t : \alpha$ .
- Next, just apply Corollary 4.3.

□

If  $\mathcal{X}_1$  and  $\mathcal{X}_2$  are two sets of  $\lambda$ -terms, then  $\mathcal{X}_1 \rightarrow \mathcal{X}_2$  denotes the set of  $\lambda$ -terms  $v$  such that for any  $u \in \mathcal{X}_1$ , we have  $(v)u \in \mathcal{X}_2$ . A set  $\mathcal{X}$  of  $\lambda$ -terms is said to be *saturated* if for any  $\lambda$ -terms  $t_1, \dots, t_n, u$  and for any  $x \in \mathcal{V}$ , we have

$$((u[t/x])t_1 \dots t_n \in \mathcal{X} \Rightarrow (\lambda x.u)tt_1 \dots t_n \in \mathcal{X}).$$

An interpretation is a map from  $A$  to the set of saturated sets. For any interpretation  $\mathcal{I}$  and for any  $\delta \in D \cup \mathcal{M}_{\text{fin}}(D)$ , we define, by induction on  $\delta$ , a saturated set  $|\delta|_{\mathcal{I}}$ :

- if  $\delta \in A$ , then  $|\delta|_{\mathcal{I}} = \mathcal{I}(\delta)$ ;
- if  $\delta = []$ , then  $|\delta|_{\mathcal{I}}$  is the set of all  $\lambda$ -terms;
- if  $\delta = [\alpha_1, \dots, \alpha_{n+1}]$ , then  $|\delta|_{\mathcal{I}} = \bigcap_{i=1}^{n+1} |\alpha_i|_{\mathcal{I}}$ .
- if  $\delta = (a, \alpha)$ , then  $|\delta|_{\mathcal{I}} = |a|_{\mathcal{I}} \rightarrow |\alpha|_{\mathcal{I}}$ .

LEMMA 5.3. Let  $\mathcal{I}$  be an interpretation and let  $u$  be a  $\lambda$ -term such that  $x_1 : a_1, \dots, x_k : a_k \vdash_R u : \alpha$ . If  $t_1 \in |a_1|_{\mathcal{I}}, \dots, t_k \in |a_k|_{\mathcal{I}}$ , then  $u[t_1/x_1, \dots, t_k/x_k] \in |\alpha|_{\mathcal{I}}$ .

PROOF. By induction on  $u$ . □

LEMMA 5.4. (i) Let  $\mathcal{N}$  be the set of head-normalizable terms. For any  $\gamma \in A$ , we set  $\mathcal{I}(\gamma) = \mathcal{N}$ . Then, for any  $\alpha \in D$ , we have  $\mathcal{V} \subseteq |\alpha|_{\mathcal{I}} \subseteq \mathcal{N}$ .

- (ii) Let  $\mathcal{N}$  be the set of normalizable terms. For any  $\gamma \in A$ , we set  $\mathcal{I}(\gamma) = \mathcal{N}$ . For any  $\alpha \in \overline{D^{\text{ex}}}$  (resp.  $\alpha \in D^{\text{ex}}$ ), we have  $\mathcal{V} \subseteq |\alpha|_{\mathcal{I}}$  (resp.  $|\alpha|_{\mathcal{I}} \subseteq \mathcal{N}$ ).

PROOF. (i) Set  $\mathcal{N}_0 = \{(x)t_1 \dots t_n / x \in \mathcal{V} \text{ and } t_1, \dots, t_n \in \Lambda\}$ . We prove, by induction on  $\alpha$ , that we have  $\mathcal{N}_0 \subseteq |\alpha|_{\mathcal{I}} \subseteq \mathcal{N}$ .

If  $\alpha = (b, \beta)$ , then, by induction hypothesis, we have  $\mathcal{N}_0 \subseteq |\beta|_{\mathcal{I}} \subseteq \mathcal{N}$  and  $\mathcal{N}_0 \subseteq |b|_{\mathcal{I}}$ . Hence we have  $\mathcal{N}_0 \subseteq \Lambda \rightarrow \mathcal{N}_0 \subseteq |\alpha|_{\mathcal{I}}$  and  $|\alpha|_{\mathcal{I}} \subseteq \mathcal{N}_0 \rightarrow \mathcal{N} \subseteq \mathcal{N}$ .

- (ii) Set  $\mathcal{N}_0 = \{(x)t_1 \dots t_n / x \in \mathcal{V} \text{ and } t_1, \dots, t_n \in \mathcal{N}\}$ . We prove, by induction on  $\alpha$ , that
- if  $\alpha \in \overline{D^{\text{ex}}}$ , then we have  $\mathcal{N}_0 \subseteq |\alpha|_{\mathcal{I}}$ ;
  - if  $\alpha \in D^{\text{ex}}$ , then we have  $|\alpha|_{\mathcal{I}} \subseteq \mathcal{N}$ .

Suppose  $\alpha = (b, \beta) \in \mathcal{M}_{\text{fin}}(D) \times D$ .

- If  $\alpha \in \overline{D^{\text{ex}}}$ , then  $b \in \mathcal{M}_{\text{fin}}(D^{\text{ex}})$  and  $\beta \in \overline{D^{\text{ex}}}$ . By induction hypothesis, we have  $|b|_{\mathcal{I}} \subseteq \mathcal{N}$  and  $\mathcal{N}_0 \subseteq |\beta|_{\mathcal{I}}$ . Hence  $\mathcal{N}_0 \subseteq \mathcal{N} \rightarrow \mathcal{N}_0 \subseteq |b|_{\mathcal{I}} \rightarrow |\beta|_{\mathcal{I}} = |\alpha|_{\mathcal{I}}$ .
- If  $\alpha \in D^{\text{ex}}$ , then  $b \in \mathcal{M}_{\text{fin}}(\overline{D^{\text{ex}}})$  and  $\beta \in D^{\text{ex}}$ . By induction hypothesis, we have  $\mathcal{N}_0 \subseteq |b|_{\mathcal{I}}$  and  $|\beta|_{\mathcal{I}} \subseteq \mathcal{N}$ . Hence  $|\alpha|_{\mathcal{I}} = |b|_{\mathcal{I}} \rightarrow |\beta|_{\mathcal{I}} \subseteq \mathcal{N}_0 \rightarrow \mathcal{N} \subseteq \mathcal{N}$  (this last inclusion follows from the fact that for any  $\lambda$ -term  $t$ , for any variable  $x$  that is not free in  $t$ , if  $(t)x$  is normalizable, then  $t$  is normalizable, fact that can be proved by induction on the number of left-reductions of  $(t)x$ ).

□

PROPOSITION 5.5. (i) *Every typable  $\lambda$ -term in System  $R$  is head-normalizable.*  
(ii) *Let  $t \in \Lambda$ ,  $\alpha \in D^{\text{ex}}$  and  $\Gamma \in \Phi^{\text{ex}}$  such that  $\Gamma \vdash_R t : \alpha$ . Then  $t$  is normalizable.*

- PROOF. (i) Let  $\Gamma$  be the context  $x_1 : a_1, \dots, x_k : a_k$ . For any  $\gamma \in A$ , we set  $\mathcal{I}(\gamma) = \mathcal{N}$ , where  $\mathcal{N}$  is the set of head-normalizable terms. By Lemma 5.4 (i), we have  $x_1 \in |a_1|_{\mathcal{I}}, \dots, x_k \in |a_k|_{\mathcal{I}}$ . Hence, by Lemma 5.3, we have  $t = t[x_1/x_1, \dots, x_k/x_k] \in |\alpha|_{\mathcal{I}}$ . Using again Lemma 5.4 (i), we obtain  $|\alpha|_{\mathcal{I}} \subseteq \mathcal{N}$ .
- (ii) Let  $\Gamma$  be the context  $x_1 : a_1, \dots, x_k : a_k$ . For any  $\gamma \in A$ , we set  $\mathcal{I}(\gamma) = \mathcal{N}$ , where  $\mathcal{N}$  is the set of normalizable terms. By Lemma 5.4 (ii), we have  $x_1 \in |a_1|_{\mathcal{I}}, \dots, x_k \in |a_k|_{\mathcal{I}}$ . Hence, by Lemma 5.3, we have  $t = t[x_1/x_1, \dots, x_k/x_k] \in |\alpha|_{\mathcal{I}}$ . Using again Lemma 5.4 (ii), we obtain  $|\alpha|_{\mathcal{I}} \subseteq \mathcal{N}$ .

□

THEOREM 5.6. (i) *For any  $t \in \Lambda$ ,  $t$  is head-normalizable if, and only if,  $t$  is typable in System  $R$ .*  
(ii) *For any  $t \in \Lambda$ ,  $t$  is normalizable if, and only if, there exist  $(\Gamma, \alpha) \in \Gamma^{\text{ex}} \times D^{\text{ex}}$  such that  $\Gamma \vdash_R t : \alpha$ .*

- PROOF. (i) Apply Proposition 5.2 (i) and Proposition 5.5 (i).  
(ii) Apply Proposition 5.2 (ii) and Proposition 5.5 (ii).

□

This theorem is not surprising: although System  $R$  is not considered in [Dezani-Ciancaglini *et al.*], it is quite obvious that its typing power is the same as that of the systems containing  $\Omega$  considered in this paper. We can note here a difference with Systems  $\lambda$  and  $\mathbb{I}$  already mentioned: in those systems, only strongly normalizable terms are typable. Of course, such systems characterizing the strongly normalizable terms, cannot be in correspondence with a denotational semantics of  $\lambda$ -calculus.

COROLLARY 5.7. *Let  $v$  and  $u$  two closed normal terms.*

- (i) *The  $\lambda$ -term  $(v)u$  is head-normalizable if, and only if, there exist  $a \in \mathcal{M}_{\text{fin}}(\llbracket u \rrbracket)$  and  $\alpha \in D$  such that  $(a, \alpha) \in \llbracket v \rrbracket$ .*  
(ii) *The  $\lambda$ -term  $(v)u$  is normalizable if, and only if, there exist  $a \in \mathcal{M}_{\text{fin}}(\llbracket u \rrbracket)$  and  $\alpha \in D^{\text{ex}}$  such that  $(a, \alpha) \in \llbracket v \rrbracket$ .*

## 6. QUANTITATIVE RESULTS

We now turn our attention to the quantitative aspects of reduction. The aim is to give a purely semantic account of execution time. Of course, if  $t'$  is the normal form of  $t$ , we know that  $\llbracket t \rrbracket = \llbracket t' \rrbracket$ , so that from  $\llbracket t \rrbracket$  it is clearly impossible to determine the number of reduction steps from  $t$  to  $t'$ . Nevertheless, if  $v$  and  $u$  are two normal  $\lambda$ -terms, we can wonder what is the number of steps leading from  $(v)u$  to its (principal head) normal form. We prove in this section that we can answer the question by only referring to  $\llbracket v \rrbracket$  and  $\llbracket u \rrbracket$  (Theorem 6.30).

### 6.1 Type Derivations for States

We now extend the type derivations for  $\lambda$ -terms to type derivations for closures (Definition 6.1) and for states (Definition 6.4). We will define also the size  $|\Pi|$  of such derivations  $\Pi$ . Naturally, the size  $|\Pi|$  of a derivation  $\Pi$  of System R is quite simply its size as a tree, i.e. the number of its nodes; moreover, for any  $n \in \mathbb{N}$ , for any  $(\Pi_1, \dots, \Pi_n) \in \Delta^n$ , we set  $|(\Pi_1, \dots, \Pi_n)| = \sum_{i=1}^n |\Pi_i|$ .

**DEFINITION 6.1.** *For any closure  $c = (t, e)$ , for any  $(\Gamma, \alpha) \in \Phi \times D$  (respectively  $(\Gamma, a) \in \Phi \times \mathcal{M}_{fin}(D)$ ), we define, by induction on  $d(e)$ , what is a derivation  $\Pi$  of  $\Gamma \vdash c : \alpha$  (respectively  $\Gamma \vdash c : a$ ) and what is  $|\Pi|$  for such a derivation:*

- A derivation of  $\Gamma \vdash (t, \bigcup_{j=1}^m \{(x_j, c_j)\}) : \alpha$  is a pair  $(\Pi_0, \bigcup_{j=1}^m \{(x_j, \Pi_j)\})$ , where
  - $\Pi_0$  is a derivation of  $\Gamma_0, x_1 : a_1, \dots, x_m : a_m \vdash_R t : \alpha$ ;
  - for any  $j \in \{1, \dots, m\}$ ,  $\Pi_j$  is a derivation of  $\Gamma_j \vdash c_j : a_j$ ;
  - and  $\Gamma = \sum_{j=0}^m \Gamma_j$ .
 If  $\Pi = (\Pi_0, \bigcup_{j=1}^m \{(x_j, \Pi_j)\})$  is a derivation of  $\Gamma \vdash c : \alpha$ , then we set  $|\Pi| = \sum_{j=0}^m |\Pi_j|$ .
- For any integer  $p$ , a derivation of  $\Gamma \vdash c : [\alpha_1, \dots, \alpha_p]$  is a  $p$ -tuple  $(\Pi^1, \dots, \Pi^p)$  such that there exists  $(\Gamma^1, \dots, \Gamma^p) \in \Phi^p$  and
  - for  $1 \leq i \leq p$ ,  $\Pi^i$  is a derivation of  $\Gamma^i \vdash c : \alpha_i$ ;
  - and  $\Gamma = \sum_{i=1}^p \Gamma^i$ .
 If  $\Pi = (\Pi^1, \dots, \Pi^p)$  is a derivation of  $\Gamma \vdash c : a$ , then we set  $|\Pi| = \sum_{i=1}^p |\Pi^i|$ .

Definition 6.1 is not so easy to use directly. This is why we introduce Lemmas 6.2 and 6.3, that will be useful for proving Propositions 6.9 and 6.16.

**LEMMA 6.2.** *Let  $((v)u, e) \in \mathcal{C}$ . For any  $b \in \mathcal{M}_{fin}(D)$ ,  $\Gamma', \Gamma'' \in \Phi$ , if  $\Pi'$  is a derivation of  $\Gamma' \vdash (v, e) : (b, \alpha)$  and  $\Pi''$  is a derivation of  $\Gamma'' \vdash (u, e) : b$ , then there exists a derivation  $\Pi$  of  $\Gamma' + \Gamma'' \vdash ((v)u, e) : \alpha$  such that  $|\Pi| = |\Pi'| + |\Pi''| + 1$ .*

**PROOF.** Set  $e = \bigcup_{j=1}^m \{(x_j, c_j)\}$  and  $\Pi' = (\Pi'_0, \bigcup_{j=1}^m \{(x_j, \Pi'_j)\})$ , where

- $\Pi'_0$  is a derivation of  $\Gamma'_0, x_1 : a'_1, \dots, x_m : a'_m \vdash_R v : (b, \alpha)$ ;
- for  $1 \leq j \leq m$ ,  $\Pi'_j$  is a derivation of  $\Gamma'_j \vdash c_j : a'_j$ ;
- and  $\Gamma' = \sum_{j=0}^m \Gamma'_j$ .

Set  $b = [\beta_1, \dots, \beta_p]$  and  $\Pi'' = (\Pi''^1, \dots, \Pi''^p)$  where, for  $1 \leq k \leq p$ ,  $\Pi''^k$  is a derivation  $(\Pi''^k_0, \bigcup_{j=1}^m \{(x_j, \Pi''^k_j)\})$  of  $\Gamma''^k \vdash (u, e) : \beta_k$  with  $\Gamma'' = \sum_{k=1}^p \Gamma''^k$ . For  $k \in \{1, \dots, p\}$ ,

- $\Pi''_0$  is a derivation of  $\Gamma''_0, x_1 : a''_1, \dots, x_m : a''_m \vdash_R u : \beta_k$ ;
- for  $1 \leq j \leq m$ ,  $\Pi''_j$  is a derivation of  $\Gamma''_j \vdash c_j : a''_j$ ;
- and  $\Gamma''^k = \sum_{j=0}^m \Gamma''_j$ .

For  $j \in \{0, \dots, m\}$ , we set  $\Gamma_j = \Gamma'_j + \sum_{k=1}^p \Gamma''_j^k$  and  $a_j = a'_j + \sum_{k=1}^p a''_j^k$ . There exists a derivation  $\Pi_0$  of  $\Gamma_0, x_1 : a_1, \dots, x_m : a_m \vdash_R (v)u : \alpha$  with  $|\Pi_0| = |\Pi'_0| + \sum_{k=1}^p |\Pi''_0^k| + 1$ . Moreover, for  $j \in \{1, \dots, m\}$ ,  $\Pi_j = \Pi'_j * \Pi''_j^1 * \dots * \Pi''_j^p$ , where  $*$  is the concatenation of finite sequences, is a derivation of  $\Gamma_j \vdash c_j : a_j$ . We have

$$\begin{aligned}
 \sum_{j=0}^m \Gamma_j &= \sum_{j=0}^m (\Gamma'_j + \sum_{k=1}^p \Gamma''_j^k) \\
 &= \sum_{j=0}^m \Gamma'_j + \sum_{j=0}^m \sum_{k=1}^p \Gamma''_j^k \\
 &= \Gamma' + \Gamma''.
 \end{aligned}$$

Hence  $\Pi = (\Pi_0, \bigcup_{j=1}^m \{(x_j, \Pi_j)\})$  is a derivation of  $\Gamma' + \Gamma'' \vdash ((v)u, e) : \alpha$ . We have

$$\begin{aligned}
 |\Pi| &= \sum_{j=0}^m |\Pi_j| \\
 &= |\Pi'_0| + \sum_{k=1}^p |\Pi''_0^k| + 1 + \sum_{j=1}^m (|\Pi'_j| + \sum_{k=1}^p |\Pi''_j^k|) \\
 &= |\Pi'| + \sum_{k=1}^p |\Pi''^k| + 1 \\
 &= |\Pi'| + |\Pi''| + 1.
 \end{aligned}$$

□

LEMMA 6.3. *For any closure  $(u, e)$ , for any derivation  $\Pi'$  of  $\Gamma, x : b \vdash (u, e) : \beta$ , there exists a derivation  $\Pi$  of  $\Gamma \vdash (\lambda x.u, e) : (b, \beta)$  such that  $|\Pi| = |\Pi'| + 1$ .*

PROOF. We set  $e = \bigcup_{j=1}^m \{(x_j, c_j)\}$  and  $\Pi' = (\Pi'_0, \bigcup_{j=1}^m \{(x_j, \Pi'_j)\})$ . We know that  $\Pi'_0$  is a derivation of  $\Gamma, x : b, x_1 : a_1, \dots, x_m : a_m \vdash_R u : \beta$ , hence there exists a derivation  $\Pi_0$  of  $\Gamma, x_1 : a_1, \dots, x_m : a_m \vdash_R \lambda x.u : (b, \beta)$ . We set  $\Pi = (\Pi_0, \bigcup_{j=1}^m \{(x_j, \Pi'_j)\})$ : it is a derivation of  $\Gamma \vdash (\lambda x.u, e) : (b, \beta)$  and we have

$$\begin{aligned}
 |\Pi| &= |\Pi_0| + \sum_{j=1}^m |\Pi'_j| \\
 &= |\Pi'_0| + 1 + \sum_{j=1}^m |\Pi'_j| \\
 &= |\Pi'| + 1.
 \end{aligned}$$

□

DEFINITION 6.4. Let  $s = (c_0, \dots, c_q)$  be a state. A finite sequence  $(\Pi_0, \dots, \Pi_q)$  is said to be a derivation of  $\Gamma \vdash s : \alpha$  if there exist  $b_1, \dots, b_q \in \mathcal{M}_{fin}(D)$ ,  $\Gamma_0, \dots, \Gamma_q \in \Phi$  such that

- $\Pi_0$  is a derivation of  $\Gamma_0 \vdash c_0 : b_1 \dots b_q \alpha$ ;
- for any  $k \in \{1, \dots, q\}$ ,  $\Pi_k$  is a derivation of  $\Gamma_k \vdash c_k : b_k$ ;
- and  $\Gamma = \sum_{k=0}^q \Gamma_k$ .

In this case, we set  $|(\Pi_0, \dots, \Pi_q)| = \sum_{k=0}^q |\Pi_k|$ .

As for derivations of closures, we introduce two lemmas about derivations for states, that will be useful for proving Propositions 6.9 and 6.16.

LEMMA 6.5. Let  $m, j_0 \in \mathbb{N}$  such that  $1 \leq j_0 \leq m$ . Let  $s = (c'_{j_0}, c_1, \dots, c_q) \in \mathbb{S}$ ,  $x_1, \dots, x_m \in \mathcal{V}$ ,  $c'_1, \dots, c'_m \in \mathcal{C}$ . For any  $(\Gamma, \alpha) \in \Phi \times D$ , if  $\Pi'$  is a derivation of  $\Gamma \vdash s : \alpha$ , then there exists a derivation  $\Pi$  of  $\Gamma \vdash ((x_{j_0}, \bigcup_{j=1}^m \{(x_j, c'_j)\}), c_1, \dots, c_q) : \alpha$  such that  $|\Pi| = |\Pi'| + 1$ .

PROOF. We set  $\Pi' = (\Pi'', \Pi_1, \dots, \Pi_q)$  with  $\Pi''$  a derivation of  $\Gamma'' \vdash c'_{j_0} : b_1 \dots b_q \alpha$ . We denote by  $\Pi_0$  the derivation of  $x : [b_1 \dots b_q \alpha] \vdash_R x : b_1 \dots b_q \alpha$ . For any  $j \in \{1, \dots, m\}$ , we set

$$\Pi''_j = \begin{cases} (\Pi'') & \text{if } j = j_0 ; \\ \epsilon & \text{else.} \end{cases}$$

The sequence  $((\Pi_0, \bigcup_{j=1}^m \{(x_j, \Pi''_j)\}), \Pi_1, \dots, \Pi_q)$  is a derivation of

$$\Gamma \vdash ((x_{j_0}, \bigcup_{j=1}^m \{(x_j, c'_j)\}), c_1, \dots, c_q) : \alpha$$

and we have

$$\begin{aligned} |((\Pi_0, \bigcup_{j=1}^m \{(x_j, \Pi''_j)\}), \Pi_1, \dots, \Pi_q)| &= |\Pi_0| + \sum_{j=1}^m |\Pi''_j| + \sum_{k=1}^q |\Pi_k| \\ &= 1 + |\Pi''| + \sum_{k=1}^q |\Pi_k| \\ &= 1 + |\Pi'|. \end{aligned}$$

□

LEMMA 6.6. For any state  $s = ((u, \{(x, c)\} \cup e), c_1, \dots, c_q)$ , for any derivation  $\Pi'$  of  $\Gamma \vdash s : \alpha$ , there exists a derivation  $\Pi$  of  $\Gamma \vdash ((\lambda x.u, e), c, c_1, \dots, c_q) : \alpha$  such that  $|\Pi| = |\Pi'| + 1$ .

PROOF. The environment  $e$  is of the shape  $\bigcup_{j=1}^m \{(x_j, c'_j)\}$  and  $\Pi'$  is of the shape  $((\Pi'_0, \{(x, \Pi'')\} \cup \bigcup_{j=1}^m \{(x_j, \Pi'_j)\}), \Pi'_1, \dots, \Pi'_q)$ . We know that  $\Pi'_0$  is a derivation of  $\Gamma, x_1 : a_1, \dots, x_m : a_m, x : a \vdash_R u : b_1 \dots b_q \alpha$ , hence there exists a derivation  $\Pi_0$  of  $x_1 : a_1, \dots, x_m : a_m \vdash_R \lambda x.u : ab_1 \dots b_q \alpha$  such that  $|\Pi_0| = |\Pi'_0| + 1$ . Set  $\Pi = ((\Pi_0, \{(x_1, \Pi'_1), \dots, (x_m, \Pi'_m)\}), \Pi'', \Pi'_1, \dots, \Pi'_q)$ : it is a derivation of  $\Gamma \vdash$

$((\lambda x.u, e), c, c_1, \dots, c_q) : \alpha$  and we have

$$\begin{aligned} |\Pi| &= |\Pi_0| + \sum_{j=1}^m |\Pi'_j| + |\Pi''| + \sum_{k=1}^q |\Pi'_k| \\ &= |\Pi'_0| + 1 + \sum_{j=1}^m |\Pi'_j| + |\Pi''| + \sum_{k=1}^q |\Pi'_k| \\ &= |\Pi'| + 1. \end{aligned}$$

□

## 6.2 Relating size of derivations and execution time

The aim of this subsection is to prove Theorem 6.11, that gives the exact number of steps leading to the principal head normal form by means of derivations in System  $R$ .

LEMMA 6.7. *Let  $((v)u, e)$  be a closure and let  $(\Gamma, \alpha) \in \Phi \times D$ . For any derivation  $\Pi$  of  $\Gamma \vdash ((v)u, e) : \alpha$ , there exist  $b \in \mathcal{M}_{\text{fin}}(D)$ ,  $\Gamma', \Gamma'' \in \Phi$ , a derivation  $\Pi'$  of  $\Gamma' \vdash (v, e) : (b, \alpha)$  and a derivation  $\Pi''$  of  $\Gamma'' \vdash (u, e) : b$  such that  $\Gamma = \Gamma' + \Gamma''$  and  $|\Pi| = |\Pi'| + |\Pi''| + 1$ .*

PROOF. Set  $e = \{(x_1, c_1), \dots, (x_m, c_m)\}$  and  $\Pi = (\Pi_0, \bigcup_{j=1}^m \{(x_j, \Pi_j)\})$  where

- (i)  $\Pi_0$  is a derivation of  $\Gamma_0, x_1 : a_1, \dots, x_m : a_m \vdash_R (v)u : \alpha$ ,
- (ii) for  $1 \leq j \leq m$ ,  $\Pi_j$  is a derivation of  $\Gamma_j \vdash c_j : a_j$ ,
- (iii)  $\Gamma = \sum_{j=0}^m \Gamma_j$ .

By (i), there exist  $b = [\beta_1, \dots, \beta_p] \in \mathcal{M}_{\text{fin}}(D)$ ,  $\Pi_0^0 \in \Delta(v, (\Gamma_0^0, x_1 : a'_1, \dots, x_m : a'_m, (b, \alpha)))$  and, for  $1 \leq k \leq p$ , a derivation  $\Pi_0^k$  of  $\Gamma_0^k, x_1 : a_1''^k, \dots, x_m : a_m''^k \vdash_R u : \beta_k$  such that

$$\begin{aligned} -\Gamma_0 &= \sum_{k=0}^p \Gamma_0^k, \\ -\text{for } 1 \leq j \leq m, a_j &= a'_j + \sum_{k=1}^p a_j''^k \\ -\text{and } |\Pi_0| &= \sum_{k=0}^p |\Pi_0^k| + 1. \end{aligned}$$

For any  $j \in \{1, \dots, m\}$ , we set  $a_j'' = \sum_{k=1}^p a_j''^k$ . By (ii), for any  $j \in \{1, \dots, m\}$ , there exist  $\Gamma_j', \Gamma_j''^1, \dots, \Gamma_j''^m \in \Phi$ , a derivation  $\Pi_j'$  of  $\Gamma_j' \vdash c_j : a'_j$  and, for any  $k \in \{1, \dots, m\}$ , a derivation  $\Pi_j''^k$  of  $\Gamma_j''^k \vdash c_j : a_j''^k$  such that

$$\begin{aligned} -\Gamma_j &= \Gamma_j' + \sum_{k=1}^m \Gamma_j''^k \\ -\text{and } |\Pi_j| &= |\Pi_j'| + \sum_{k=1}^m |\Pi_j''^k|. \end{aligned}$$

Set

$$\begin{aligned} -\Gamma' &= \Gamma_0^0 + \sum_{j=1}^m \Gamma_j'; \\ -\Gamma'' &= \sum_{k=1}^p (\Gamma_0^k + \sum_{j=1}^m \Gamma_j''^k); \\ -\Pi' &= (\Pi_0^0, \bigcup_{j=1}^m \{(x_j, \Pi_j')\}) \\ -\text{and } \Pi'' &= ((\Pi_0^1, \bigcup_{j=1}^m \{(x_j, \Pi_j''^1)\}), \dots, (\Pi_0^p, \bigcup_{j=1}^m \{(x_j, \Pi_j''^p)\})). \end{aligned}$$



We have

$$\begin{aligned}
\Gamma &= \sum_{j=0}^m \Gamma_j \\
&\quad (\text{by (iii)}) \\
&= \sum_{k=0}^p \Gamma_0^k + \sum_{j=1}^m (\Gamma'_j + \sum_{k=1}^p \Gamma_j''^k) \\
&= \Gamma' + \Gamma''
\end{aligned}$$

and

$$\begin{aligned}
|\Pi| &= \sum_{j=0}^m |\Pi_j| \\
&= \sum_{k=0}^p |\Pi_0^k| + 1 + \sum_{j=1}^m (|\Pi'_j| + \sum_{k=1}^p |\Pi_j''^k|) \\
&= |\Pi'| + |\Pi''| + 1.
\end{aligned}$$

□

**PROPOSITION 6.8.** *Let  $t$  be a head normalizable  $\lambda$ -term. For any  $(\Gamma, \alpha) \in \Phi \times D$ , for any  $\Pi \in \Delta(t, (\Gamma, \alpha))$ , we have  $l_h((t, \emptyset).e) \leq |\Pi|$ .*

**PROOF.** By Theorem 2.10, we can prove, by induction on  $l_h(s)$ , that for any  $s \in \mathbb{S}$  such that  $\bar{s}$  is head normalizable, for any  $(\Gamma, \alpha) \in \Phi \times D$ , for any derivation  $\Pi$  of  $\Gamma \vdash s : \alpha$ , we have  $l_h(s) \leq |\Pi|$ .

The base case is trivial, because we never have  $l_h(s) = 0$ . The inductive step is divided into five cases:

- In the case where  $s = (x, e).\pi$ ,  $x \in \mathcal{V}$  and  $x \notin \text{dom}(e)$ ,  $l_h(s) = 1 \leq |\Pi|$ .
- In the case where  $s = ((x_{j_0}, \bigcup_{j=1}^m \{(x_j, c'_j)\}), c_1, \dots, c_q)$  and  $1 \leq j_0 \leq m$ , we have  $\Pi = (\Pi_0, \dots, \Pi_q)$ , where  $\Pi_0 = (\Pi'_0, \bigcup_{j=1}^m \{(x_j, \Pi'_j)\})$  with
  - $\Pi'_0$  is a derivation of  $\Gamma'_0, x_1 : a_1, \dots, x_m : a_m \vdash_R x_{j_0} : b_1 \dots b_q \alpha$ ,
  - for any  $j \in \{1, \dots, m\}$ ,  $\Pi'_j$  is a derivation of  $\Gamma'_j \vdash c'_j : a_j$ ,
  - $\Gamma_0 = \sum_{j=1}^m \Gamma'_j$ ,
  - for  $1 \leq k \leq q$ ,  $\Pi_k$  is a derivation of  $\Gamma_k \vdash c_k : b_k$
  - and  $\Gamma = \sum_{k=0}^q \Gamma_k$ .
 Hence  $a'_{j_0} = [b_1 \dots b_q \alpha]$ . The sequence  $(\Pi'_{j_0}, \Pi_1, \dots, \Pi_q)$  is a derivation of

$$\Gamma'_{j_0} + \sum_{k=1}^q \Gamma_k \vdash (c'_{j_0}, c_1, \dots, c_q) : \alpha.$$

We have

$$\begin{aligned}
l_h(s) &= l_h(c'_{j_0}, c_1, \dots, c_q) + 1 \\
&\leq |(\Pi'_{j_0}, \Pi_1, \dots, \Pi_q)| + 1 \\
&\quad (\text{by induction hypothesis})
\end{aligned}$$

$$\begin{aligned}
&= |\Pi'_{j_0}| + \sum_{k=1}^q |\Pi_k| + 1 \\
&\leq |\Pi_0| + \sum_{k=1}^q |\Pi_k| \\
&= |\Pi|.
\end{aligned}$$

—In the case where  $s = ((\lambda x.u, \{(x_1, c'_1), \dots, (x_m, c'_m)\}), c', c_1, \dots, c_q)$ , we have  $\Pi = ((\Pi'_0, \Pi''_0), \Pi', \Pi_1, \dots, \Pi_q)$  with

- $\Pi'_0$  is a derivation of  $\Gamma'_0, x_1 : a_1, \dots, x_m : a_m \vdash_R \lambda x.u : b'b_1 \dots b_q \alpha$ ;
- $\Pi''_0 = \bigcup_{j=1}^m \{(x_j, \Pi'_j)\}$  where, for  $1 \leq j \leq m$ ,  $\Pi'_j$  is a derivation of  $\Gamma'_j \vdash c'_j : a_j$ ;
- $\Gamma_0 = \sum_{j=0}^m \Gamma'_j$ ;
- $\Pi'$  is a derivation of  $\Gamma' \vdash b' : c'$ ;
- for  $1 \leq k \leq q$ ,  $\Pi_k$  is a derivation of  $\Gamma_k \vdash b_k : c_k$ .

Hence there exists  $\Pi'' \in \Delta(u, (\Gamma'_0, x_1 : a_1, \dots, x_m : a_m, x : b', b_1 \dots b_q \alpha))$  with  $|\Pi'_0| = |\Pi''| + 1$ . The pair  $(\Pi'', \bigcup_{j=1}^m \{(x_j, \Pi'_j)\} \cup \{\Pi'\})$  is a derivation of

$$\Gamma_0 + \Gamma' \vdash (u, \{(x_1, c'_1), \dots, (x_m, c'_m), (x, c)\}) : b_1 \dots b_q \alpha.$$

Hence  $((\Pi'', \bigcup_{j=1}^m \{(x_j, \Pi'_j)\} \cup \{(x, \Pi')\}), \Pi_1, \dots, \Pi_q)$  is a derivation of

$$\Gamma \vdash ((u, \{(x_1, c'_1), \dots, (x_m, c'_m), (x, c)\}), c_1, \dots, c_q) : \alpha.$$

We have

$$\begin{aligned}
l_h(s) &= l_h((u, \{(x_1, c'_1), \dots, (x_m, c'_m), (x, c)\}), c_1, \dots, c_q) + 1 \\
&\leq |((\Pi'', \{\Pi'_1, \dots, \Pi'_m, \Pi'\}), \Pi_1, \dots, \Pi_q)| + 1 \\
&\quad \text{(by induction hypothesis)} \\
&= |\Pi''| + \sum_{j=1}^m |\Pi'_j| + |\Pi'| + \sum_{k=1}^q |\Pi_k| + 1 \\
&= |\Pi'_0| + |\Pi''_0| + |\Pi'| + \sum_{k=1}^q |\Pi_k| \\
&= |\Pi|.
\end{aligned}$$

—In the case where  $s = (((v)u, e), c_1, \dots, c_q)$ , we have  $\Pi = (\Pi_0, \dots, \Pi_q)$  with

- $\Pi_0$  is a derivation of  $\Gamma_0 \vdash ((v)u, e) : b_1 \dots b_q \alpha$ ;
- for  $1 \leq k \leq q$ ,  $\Pi_k$  is a derivation of  $\Gamma_k \vdash c_k : b_k$ ;
- $\Gamma = \sum_{k=0}^q \Gamma_k$ .

By Lemma 6.7, there exist  $b \in \mathcal{M}_{\text{fin}}(D)$ ,  $\Gamma'_0, \Gamma''_0 \in \Phi$ , a derivation  $\Pi'_0$  of  $\Gamma'_0 \vdash (v, e) : bb_1 \dots b_q \alpha$  and a derivation  $\Pi''_0$  of  $\Gamma''_0 \vdash (u, e) : b$  such that  $\Gamma_0 = \Gamma'_0 + \Gamma''_0$  and  $|\Pi_0| = |\Pi'_0| + |\Pi''_0| + 1$ . The sequence  $(\Pi'_0, \Pi''_0, \Pi_1, \dots, \Pi_q)$  is a derivation of  $\Gamma \vdash ((v, e), (u, e), c_1, \dots, c_q) : \alpha$ . We have

$$\begin{aligned}
l_h(s) &= l_h((v, e), (u, e), c_1, \dots, c_q) + 1 \\
&\leq |(\Pi'_0, \Pi''_0, \Pi_1, \dots, \Pi_q)| + 1 \\
&\quad \text{(by induction hypothesis)}
\end{aligned}$$

$$\begin{aligned}
&= |\Pi'_0| + |\Pi''_0| + \sum_{k=1}^q |\Pi_k| + 1 \\
&= |\Pi_0| + \sum_{k=1}^q |\Pi_k| \\
&= |(\Pi_0, \dots, \Pi_q)| \\
&= |\Pi|.
\end{aligned}$$

- In the case where  $s = (\lambda x.u, \bigcup_{j=1}^m \{(x_j, c'_j)\}).\epsilon$ , we have  $\Pi = (\Pi'_0, \Pi''_0)$  with
- $\Pi'_0$  is a derivation of  $\Gamma'_0, x_1 : a_1, \dots, x_m : a_m \vdash_R \lambda x.u : \alpha$ ;
  - $\Pi''_0 = \bigcup_{j=1}^m \{(x_j, \Pi'_j)\}$  where, for  $1 \leq j \leq m$ ,  $\Pi'_j$  is a derivation of  $\Gamma'_j \vdash c'_j : a_j$ ;
  - $\Gamma = \sum_{j=0}^m \Gamma'_j$ .
- Hence there exists a derivation  $\Pi''$  of  $\Gamma'_0, x_1 : a_1, \dots, x_m : a_m, x : b \vdash_R u : \beta$  such that  $\alpha = (b, \beta)$  and  $|\Pi'_0| = |\Pi''| + 1$ . The pair  $(\Pi'', \Pi'_0)$  is a derivation of

$$\Gamma, x : b \vdash (u, \bigcup_{j=1}^m \{(x_j, c_j)\}).\epsilon : \beta.$$

We have

$$\begin{aligned}
l_h(s) &= l_h((u, \bigcup_{j=1}^m \{(x_j, c_j)\}).\epsilon) + 1 \\
&\leq |(\Pi'', \Pi'_0)| + 1 \\
&= |\Pi''| + \sum_{j=1}^m |\Pi'_j| + 1 \\
&= |\Pi'_0| + \sum_{j=1}^m |\Pi'_j| \\
&= |\Pi|.
\end{aligned}$$

□

**PROPOSITION 6.9.** *Let  $t$  be a head normalizable  $\lambda$ -term. There exist  $(\Gamma, \alpha) \in \Phi \times D$  and  $\Pi \in \Delta(t, (\Gamma, \alpha))$  such that  $l_h((t, \emptyset).\epsilon) = |\Pi|$ .*

**PROOF.** By Theorem 2.10, we can prove, by induction on  $l_h(s)$ , that for any  $s \in \mathbb{S}$  such that  $\bar{s}$  is head normalizable, there exist  $(\Gamma, \alpha)$  and a derivation  $\Pi$  of  $\Gamma \vdash s : \alpha$  such that we have  $l_h(s) = |\Pi|$ .

The base case is trivial, because we never have  $l_h(s) = 0$ . The inductive step is divided into five cases:

- In the case where  $s = ((x, e), c_1, \dots, c_q)$ ,  $x \in \mathcal{V}$  and  $x \notin \text{dom}(e)$ , we have  $l_h(s) = 1$  and there exists a derivation  $\Pi = (\Pi_0, \dots, \Pi_q)$  of  $\Gamma \vdash s : \alpha$ , where  $\Pi_0$  is a derivation of  $x : \underbrace{[\dots] \alpha}_{q \text{ times}} \vdash (x, e) : \underbrace{[\dots] \alpha}_{q \text{ times}}$  with  $|\Pi_0| = 1$  and  $|\Pi_1| = \dots = |\Pi_q| = 0$ .

- In the case where  $s$  is of the shape  $(x, e). \pi$  with  $x \in \text{dom}(e)$ , apply the induction hypothesis and Lemma 6.5.
- In the case where  $s$  is of the shape  $((v)u, e). \pi$ , apply the induction hypothesis and Lemma 6.2.
- In the case where  $s$  is of the shape  $(\lambda x. u), e). \epsilon$ , apply the induction hypothesis Lemma 6.3.
- In the case where  $s$  is of the shape  $((\lambda x. u), e). \pi$  with  $\pi \neq \epsilon$ , apply the induction hypothesis and Lemma 6.6.

□

DEFINITION 6.10. For every  $\mathfrak{D} \in \mathcal{P}(\Delta) \cup \mathcal{P}(\Delta^{<\omega})$ , we set  $|\mathfrak{D}| = \{|\Pi| / \Pi \in \mathfrak{D}\}$ .

THEOREM 6.11. For any  $\lambda$ -term  $t$ , we have  $l_h((t, \emptyset). \epsilon) = \inf |\Delta(t)|$ .

PROOF. We distinguish between two cases.

- The  $\lambda$ -term  $t$  is not head normalizable: by Theorem 5.6 (i),  $\inf |\Delta(t)| = \infty$  and, by Theorem 2.9,  $l_h((t, \emptyset). \epsilon) = \infty$ .
- The  $\lambda$ -term  $t$  is head normalizable: apply Proposition 6.8 and Proposition 6.9.

□

### 6.3 Principal typings and 1-typings

In the preceding subsection, we related  $l_h(t)$  and the size of the derivations of  $t$  for any  $\lambda$ -term  $t$ . Now, we want to relate  $l_\beta(t)$  and the size of the derivations of  $t$ . We will show that if the value of  $l_\beta(t)$  is finite (i.e.  $t$  is normalizable), then  $l_\beta(t)$  is the size of the least derivations of  $t$  with typings that satisfy a particular property and that, otherwise, there is no such derivation. In particular, when  $t$  is normalizable,  $l_\beta(t)$  is the size of the derivations of  $t$  with 1-typings. This notion of 1-typing, defined in Definition 6.13, is a generalization of the notion of *principal typing*.

We recall that a typing  $(\Gamma, \alpha)$  for a  $\lambda$ -term is a principal typing if all other typings for the same  $\lambda$ -term can be derived from  $(\Gamma, \alpha)$  by some set of operations. The work of [Coppo *et al.* 1980] could be adapted in order to show that all normal  $\lambda$ -terms have a principal typing in System  $R$  if  $A$  is infinite: the operations are substitution (see Definition 4.9) and expansion (complicated to define); the only difference with [Coppo *et al.* 1980] is that we should have to consider 0-expansions too and not only  $n$ -expansions for  $n \geq 1$ .

DEFINITION 6.12. The typing rules for deriving principal typings of normal  $\lambda$ -terms are the following:

$$\begin{array}{c}
 \frac{}{x : [\gamma] \vdash_P x : \gamma} \gamma \in A \\
 \frac{\Gamma, x : a \vdash_P t : \alpha}{\Gamma \vdash_P \lambda x. t : (a, \alpha)} \\
 \frac{\Gamma_1 \vdash_P u_1 : \alpha_1 \quad \dots \quad \Gamma_n \vdash_P u_n : \alpha_n}{\sum_{i=1}^n \Gamma_i + \{(x, [[\alpha_1] \dots [\alpha_n] \gamma])\} \vdash_P (x)u_1 \dots u_n : \gamma} (*)
 \end{array}$$

(\*) the atoms in  $\Gamma_j$  are disjoint from those in  $\Gamma_k$  if  $j \neq k$  and  $\gamma \in A$  does not appear in the  $\Gamma_i$

A principal typing of a normalizable  $\lambda$ -term is a principal typing of its normal form.

The reader acquainted with the concept of *experiment* on proof nets in linear logic could notice that a principal typing of a normal  $\lambda$ -term is the same thing as the result of what [Tortora de Falco 2000] calls *an injective 1-experiment* of the proof net obtained by the translation of this  $\lambda$ -term mentioned in Subsection 4.2.

The notion of 1-typing is more general than the notion of principal typing: it is the result of a *1-experiment* (not necessarily injective).

DEFINITION 6.13. *The typing rules for deriving 1-typings of normal  $\lambda$ -terms are the following:*

$$\frac{}{x : [\gamma] \vdash_1 x : \gamma} \gamma \in A$$

$$\frac{\Gamma, x : a \vdash_1 t : \alpha}{\Gamma \vdash_1 \lambda x.t : (a, \alpha)}$$

$$\frac{\Gamma_1 \vdash_1 u_1 : \alpha_1 \quad \dots \quad \Gamma_n \vdash_1 u_n : \alpha_n}{\sum_{i=1}^n \Gamma_i + \{(x, [[\alpha_1] \dots [\alpha_n] \gamma])\} \vdash_1 (x)u_1 \dots u_n : \gamma}$$

A 1-typing of a normalizable  $\lambda$ -term is a 1-typing of its normal form.

Note that if  $t$  is a normalizable  $\lambda$ -term and  $(\Gamma, \alpha)$  is a 1-typing of  $t$ , then  $(\Gamma, \alpha) \in \Phi^{\text{ex}} \times D^{\text{ex}}$ ; more precisely, a typing  $(\Gamma, \alpha)$  of a normalizable  $\lambda$ -term is a 1-typing if, and only if, every multiset in negative occurrence in  $\Gamma$  (resp. in positive occurrence in  $\alpha$ ) is a singleton.

LEMMA 6.14. *Let  $(x, e)$  be a closure and let  $\Gamma \in \Phi$  such that  $\Gamma \in \Phi^{\text{ex}}$ . Assume that there exists a derivation of  $\Gamma \vdash (x, e) : b_1 \dots b_q \alpha$ , with  $x \notin \text{dom}(e)$ , then for any  $k \in \{1, \dots, q\}$ , we have  $b_k \neq []$ .*

PROOF. Let  $\Pi$  be such a derivation. Set  $e = \bigcup_{j=1}^m \{(x_j, c_j)\}$ . We have  $\Pi = (\Pi_0, \bigcup_{j=1}^m \{(x_j, \Pi_j)\})$ , where

- (i)  $\Pi_0$  is a derivation of  $\Gamma_0, x_1 : a_1, \dots, x_m : a_m \vdash_R x : b_1 \dots b_q \alpha$ ;
- (ii) for  $j \in \{1, \dots, m\}$ ,  $\Pi_j$  is a derivation of  $\Gamma_j \vdash c_j : a_j$ ;
- (iii) and  $\Gamma = \sum_{j=0}^m \Gamma_j$ .

By (i), since  $x \notin \text{dom}(e)$ ,  $\Gamma_0(x) = [b_1 \dots b_q \alpha]$ . Hence, by (iii), if there existed  $k \in \{1, \dots, q\}$  such that  $b_k = []$ , then we should have  $\Gamma \notin \Phi^{\text{ex}}$ .  $\square$

PROPOSITION 6.15. *Let  $t$  be a normalizable  $\lambda$ -term. If  $\Pi$  is a derivation of  $\Gamma \vdash_R t : \alpha$  and  $(\Gamma, \alpha) \in \Phi^{\text{ex}} \times D^{\text{ex}}$ , then we have  $l_\beta((t, \emptyset). \epsilon) \leq |\Pi|$ .*

PROOF. By Theorem 2.12, we can prove, by induction on  $l_\beta(s)$ , that for any  $s = (c_0, \dots, c_q) \in \mathbb{S}$  such that  $(\overline{c_0})\overline{c_1} \dots \overline{c_q}$  is normalizable, for any  $(\Gamma, \alpha) \in \Phi \times D$ , if  $\Pi$  is a derivation of  $\Gamma \vdash s : \alpha$  and  $(\Gamma, \alpha) \in \Phi^{\text{ex}} \times D^{\text{ex}}$ , then we have  $l_\beta(s) \leq |\Pi|$ .

In the case where  $s = ((x, e), c_1, \dots, c_q)$  and  $x \notin \text{dom}(e)$ , we apply Lemma 6.14.  $\square$

PROPOSITION 6.16. *Assume that  $t$  is a normalizable  $\lambda$ -term and that  $(\Gamma, \alpha)$  is a 1-typing of  $t$ . Then there exists a derivation  $\Pi$  of  $\Gamma \vdash_R t : \alpha$  such that  $l_\beta((t, \emptyset). \epsilon) = |\Pi|$ .*

PROOF. By Theorem 2.12, we can prove, by induction on  $l_\beta(s)$ , that for any  $s \in \mathbb{S}$  such that  $\bar{s}$  is normalizable and for any 1-typing  $(\Gamma, \alpha)$  of  $\bar{s}$ , there exists a derivation  $\Pi$  of  $\Gamma \vdash s : \alpha$  such that  $l_\beta(s) = |\Pi|$ .

The base case is trivial, because we never have  $l_\beta(s) = 0$ . The inductive step is divided into five cases:

—In the case where  $s = ((x, e), c_1, \dots, c_q)$  and  $x \notin \text{dom}(e)$ ,  $(\Gamma, \alpha)$  is a 1-typing of  $(x)t_1 \dots t_q$ , where  $t_1, \dots, t_q$  are the respective normal forms of  $\bar{c}_1, \dots, \bar{c}_q$ , hence there exist  $\Gamma_1, \dots, \Gamma_q, \alpha_1, \dots, \alpha_q$  such that

— $\Gamma = \sum_{k=1}^q \Gamma_k + \{(x, [[\alpha_1] \dots [\alpha_q] \alpha])\}$

—and  $(\Gamma_1, \alpha_1), \dots, (\Gamma_q, \alpha_q)$  are 1-typings of  $t_1, \dots, t_q$  respectively.

By induction hypothesis, there exist  $q$  derivations  $\Pi_1, \dots, \Pi_q$  of  $\Gamma_1 \vdash_R t_1 : \alpha_1, \dots, \Gamma_q \vdash_R t_q : \alpha_q$  respectively. We denote by  $x_1, \dots, x_m$  the elements of  $\text{dom}(e)$ .

We denote by  $\Pi_0$  the derivation of

$$x : [[\alpha_1] \dots [\alpha_q] \alpha] \vdash_R x : \alpha.$$

Set  $\Pi = ((\Pi_0, \bigcup_{j=1}^m \{(x_j, \epsilon)\}), \Pi_1, \dots, \Pi_q)$ : it is a derivation of  $\Gamma \vdash_R \bar{s} : \alpha$  and we have

$$\begin{aligned} l_\beta(s) &= \sum_{k=1}^q l_\beta(c_k) + 1 \\ &= \sum_{k=1}^q |\Pi_k| + 1 \\ &\quad \text{(by induction hypothesis)} \\ &= |\Pi_0| + \sum_{k=1}^q |\Pi_k| \\ &= |\Pi|. \end{aligned}$$

—In the case where  $s$  is of the shape  $(x, e). \pi$  with  $x \in \text{dom}(e)$ , apply the induction hypothesis and Lemma 6.5.

—In the case where  $s$  is of the shape  $((v)u, e). \pi$ , apply the induction hypothesis and Lemma 6.2.

—In the case where  $s$  is of the shape  $(\lambda x. u, e). \epsilon$ , apply the induction hypothesis and Lemma 6.3.

—In the case where  $s$  is of the shape  $(\lambda x. u, e). \pi$  with  $\pi \neq \epsilon$ , apply the induction hypothesis and Lemma 6.6.

□

DEFINITION 6.17. *For any  $\lambda$ -term  $t$ , we set  $\Delta^{\text{ex}}(t) = \bigcup_{(\Gamma, \alpha) \in \Phi^{\text{ex}} \times D^{\text{ex}}} \Delta(t, (\Gamma, \alpha))$ .*

THEOREM 6.18. *For any  $\lambda$ -term  $t$ , we have  $l_\beta((t, \emptyset). \epsilon) = \inf |\Delta^{\text{ex}}(t)|$ .*

PROOF. We distinguish between two cases.

- The  $\lambda$ -term  $t$  is not normalizable: by Theorem 5.6 (ii),  $\inf |\Delta^{\text{ex}}(t)| = \infty$  and, by Theorem 2.12,  $l_\beta((t, \emptyset).e) = \infty$ .
- The  $\lambda$ -term  $t$  is normalizable: apply Proposition 6.15 and Proposition 6.16.

□

#### 6.4 Relating semantics and execution time

In this subsection, we prove the first truly semantic measure of execution time of this paper by bounding (by purely semantic means, i.e. without considering derivations) the number of steps of the computation of the principal head normal form (Theorem 6.23).

We define the size  $|\alpha|$  of any  $\alpha \in D$  using an auxiliary function  $\text{aux}$ .

DEFINITION 6.19. *For any  $\alpha \in D$ , we define  $|\alpha|$  and  $\text{aux}(\alpha)$  by induction on  $\text{depth}(\alpha)$ ,:*

- if  $\alpha \in A$ , then  $|\alpha| = 1$  and  $\text{aux}(\alpha) = 0$ ;
- if  $\alpha = ([\alpha_1, \dots, \alpha_n], \alpha_0)$ , then
  - $|\alpha| = \sum_{i=1}^n \text{aux}(\alpha_i) + |\alpha_0| + 1$
  - and  $\text{aux}(\alpha) = \sum_{i=1}^n |\alpha_i| + \text{textsfaux}(\alpha_0) + 1$ .

For any  $a = [\alpha_1, \dots, \alpha_n] \in \mathcal{M}_{\text{fin}}(D)$ , we set  $|a| = \sum_{i=1}^n |\alpha_i|$  and  $\text{aux}(a) = \sum_{i=1}^n \text{aux}(\alpha_i)$ .

Notice that for any  $\alpha \in D$ , the size  $|\alpha|$  of  $\alpha$  is the sum of the number of positive occurrences of atoms in  $\alpha$  and of the number of commas separating a multiset of types and a type.

EXAMPLE 6.20. Let  $\gamma \in A$ . Set  $\alpha = ([\gamma], \gamma)$  and  $a = \underbrace{[\alpha, \dots, \alpha]}_{n \text{ times}}$ . We have  $|(a, \alpha)| = 2n + 3$ .

LEMMA 6.21. *For any  $\lambda$ -term  $u$ , if there exists a derivation  $\Pi$  of  $x_1 : a_1, \dots, x_m : a_m \vdash_R u : \alpha$ , then  $|a_1 \dots a_m \alpha| = \text{aux}(a_1 \dots a_m \alpha)$ .*

PROOF. By induction on  $\Pi$ . □

LEMMA 6.22. *Let  $v$  be a normal  $\lambda$ -term and let  $\Pi$  be a derivation of  $x_1 : a_1, \dots, x_m : a_m \vdash_R v : \alpha$ . Then we have  $|\Pi| \leq |a_1 \dots a_m \alpha|$ .*

PROOF. By induction on  $v$ . □

THEOREM 6.23. *Let  $v$  and  $u$  be two closed normal  $\lambda$ -terms. Assume  $(a, \alpha) \in \llbracket v \rrbracket$  and  $\text{Supp}(a) \subseteq \llbracket u \rrbracket$ .*

- (i) *We have  $l_h(((v)u, \emptyset).e) \leq 2|a| + |\alpha| + 2$ .*
- (ii) *If, moreover,  $\alpha \in D^{\text{ex}}$ , then we have*

$$l_\beta(((v)u, \emptyset).e) \leq 2|a| + |\alpha| + 2.$$

PROOF. Set  $a = [\alpha_1, \dots, \alpha_n]$ . There exist a derivation  $\Pi_0$  of  $\vdash_R v : (a, \alpha)$  and  $n$  derivations  $\Pi_1, \dots, \Pi_n$  of  $\vdash_R u : \alpha_1, \dots, \vdash_R u : \alpha_n$  respectively. Hence there exists a derivation  $\Pi$  of  $\vdash_R (v)u : \alpha$  such that  $|\Pi| = \sum_{i=0}^n |\Pi_i| + 1$ .

(i) We have

$$\begin{aligned}
l_h(((v)u, \emptyset).e) &\leq \sum_{i=0}^n |\Pi_i| + 1 \\
&\quad \text{(by Proposition 6.8)} \\
&\leq |(a, \alpha)| + \sum_{i=1}^n |\alpha_i| + 1 \\
&\quad \text{(by Lemma 6.22)} \\
&= \sum_{i=1}^n \text{aux}(\alpha_i) + |\alpha| + 1 + |a| + 1 \\
&= \sum_{i=1}^n |\alpha_i| + |\alpha| + 1 + |a| + 1 \\
&\quad \text{(by Lemma 6.21)} \\
&= 2|a| + |\alpha| + 2.
\end{aligned}$$

(ii) The only difference with the proof of (i) is that we apply Proposition 6.15 instead of Proposition 6.8.

□

### 6.5 The exact number of steps

This subsection is devoted to giving the exact number of steps of computation by purely semantic means. For arbitrary points  $(a, \alpha) \in \llbracket v \rrbracket$  such that  $a \in \mathcal{M}_{\text{fin}}(\llbracket u \rrbracket)$ , it is clearly impossible to obtain an equality in Theorem 6.23, because there exist such points with different sizes.

The only equalities we have by now are Theorem 6.11 and Theorem 6.18, which use the size of the derivations. A first idea is then to look for points  $(a, \alpha) \in \llbracket v \rrbracket$  such that  $a \in \mathcal{M}_{\text{fin}}(\llbracket u \rrbracket)$  with  $|(a, \alpha)|$  equals to the sizes of the derivations used in these theorems. But there are cases in which such points do not exist.

A more subtle way out is nevertheless possible, and here is where the notions of equivalence between derivations and of substitution defined in Subsection 4.3 come into the picture. More precisely, using the notion of substitution, Proposition 6.27 (the only place where we use the non-finiteness of the set  $A$  of atoms through Fact 6.24 and Lemma 6.26) shows how to find, for any  $\beta \in \llbracket t \rrbracket$ , an element  $\alpha \in \llbracket t \rrbracket$  such that  $|\alpha| = \min |\Delta(t, \beta)|$ .

We remind that  $A = D \setminus (\mathcal{M}_{\text{fin}}(D) \times D)$ . The equivalence relation  $\sim$  has been defined in Definition 4.8 and the notion of substitution has been defined in Definition 4.9. We recall that we denote by  $\mathcal{S}$  the set of substitutions.

**FACT 6.24.** *Let  $v$  be a normal  $\lambda$ -term and let  $\Pi$  be a derivation of*

$$x_1 : b_1, \dots, x_m : b_m \vdash_R v : \beta.$$

*There exist  $a_1, \dots, a_m, \alpha$  and a derivation  $\Pi'$  of  $x_1 : a_1, \dots, x_m : a_m \vdash_R v : \alpha$  such that  $\Pi' \sim \Pi$  and  $|\Pi'| + m = |a_1 \dots a_m \alpha|$ . If, moreover,  $A$  is infinite, then we can*



choose  $\Pi'$  in such a way that there exists a substitution  $\sigma$  such that  $\bar{\sigma}(a_1) = b_1, \dots, \bar{\sigma}(a_m) = b_m$  and  $\sigma(\alpha) = \beta$ .

PROOF. By induction on  $v$ .  $\square$

In the case where  $A$  is infinite, the derivation  $\Pi'$  of the lemma is what [Coppo *et al.* 1980] calls a *ground deduction* for  $v$ .

DEFINITION 6.25. For every  $X \in \mathcal{P}(D) \cup \mathcal{P}(\mathcal{M}_{fin}(D))$ , we set  $|X| = \{|\alpha| \mid \alpha \in X\}$ .

LEMMA 6.26. Assume  $A$  is infinite. Let  $t$  be a closed normal  $\lambda$ -term, let  $\beta \in D$  and let  $\Pi \in \Delta(t, \beta)$ . Then we have

$$|\Pi| = \min \{|\alpha| \in D \mid (\exists \Pi' \in \Delta(t, \alpha))(\exists \sigma \in \mathcal{S})(\Pi' \sim \Pi \text{ and } \sigma(\alpha) = \beta)\}.$$

PROOF. Apply Lemma 6.22 and Fact 6.24.  $\square$

PROPOSITION 6.27. Assume  $A$  is infinite. Let  $t$  be a closed normal  $\lambda$ -term and let  $\beta \in \llbracket t \rrbracket$ . We have  $\min |\Delta(t, \beta)| = \min \{|\alpha| \in \llbracket t \rrbracket \mid (\exists \sigma \in \mathcal{S})\sigma(\alpha) = \beta\}$ .

PROOF. Set  $m = \min |\Delta(t, \beta)|$  and  $n = \min \{|\alpha| \in \llbracket t \rrbracket \mid (\exists \sigma \in \mathcal{S})\sigma(\alpha) = \beta\}$ .

First, we prove that  $m \leq n$ . Let  $\alpha \in \llbracket t \rrbracket$  such that we have  $(\exists \sigma \in \mathcal{S})\sigma(\alpha) = \beta$ . By Theorem 4.2,  $\Delta(t, \alpha) \neq \emptyset$ : let  $\Pi' \in \Delta(t, \alpha)$ . By Proposition 4.10, there exists  $\Pi \in \Delta(t, \beta)$  such that  $\Pi \sim \Pi'$ . By Lemma 6.22, we have  $|\Pi'| \leq |\alpha|$ . Hence we obtain  $m \leq |\Pi| = |\Pi'| \leq |\alpha|$ .

Now, we prove the inequality  $n \leq m$ . Let  $\Pi \in \Delta(t, \beta)$ .

$$\begin{aligned} n &= \min \{|\alpha| \in D \mid (\exists \Pi' \in \Delta(t, \alpha))(\exists \sigma \in \mathcal{S})\sigma(\alpha) = \beta\} \\ &\quad (\text{by Theorem 4.2}) \\ &\leq \min \{|\alpha| \in D \mid (\exists \Pi' \in \Delta(t, \alpha))(\exists \sigma \in \mathcal{S})(\Pi' \sim \Pi \text{ and } \sigma(\alpha) = \beta)\} \\ &= |\Pi| \\ &\quad (\text{by Lemma 6.26}). \end{aligned}$$

$\square$

COROLLARY 6.28. Assume  $A$  is infinite. Let  $t$  be a closed normal  $\lambda$ -term and let  $b \in \mathcal{M}_{fin}(\llbracket t \rrbracket)$ . We have  $\min |\Delta(t, b)| = \min \{|\alpha| \in \mathcal{M}_{fin}(\llbracket t \rrbracket) \mid (\exists \sigma \in \mathcal{S})\bar{\sigma}(a) = b\}$ .

The point of Theorem 6.30 is that the number of steps of the computation of the (principal head) normal form of  $(v)u$ , where  $v$  and  $u$  are two closed normal  $\lambda$ -terms, can be determined from  $\llbracket v \rrbracket$  and  $\llbracket u \rrbracket$ .

DEFINITION 6.29. For any  $X, Y \subseteq D$ , we denote by  $\mathcal{U}(X, Y)$  the set

$$\{((a, \alpha), a') \in (X \setminus A) \times \mathcal{M}_{fin}(Y) \mid (\exists \sigma \in \mathcal{S})\bar{\sigma}(a) = \bar{\sigma}(a')\}$$

and by  $\mathcal{U}^{\text{ex}}(X, Y)$  the set

$$\{((a, \alpha), a') \in (X \setminus A) \times \mathcal{M}_{fin}(Y) \mid (\exists \sigma \in \mathcal{S})(\bar{\sigma}(a) = \bar{\sigma}(a') \text{ and } \sigma(\alpha) \in D^{\text{ex}})\}.$$

THEOREM 6.30. Assume  $A$  is infinite. For any two closed normal  $\lambda$ -terms  $u$  and  $v$ , we have

$$(i) \quad l_h(((v)u), \emptyset). \epsilon) = \inf \{|\alpha| + |\alpha'| + 1 \mid ((a, \alpha), a') \in \mathcal{U}(\llbracket v \rrbracket, \llbracket u \rrbracket)\};$$

(ii)  $l_\beta(((v)u, \emptyset). \epsilon) = \inf\{|(a, \alpha)| + |a'| + 1 / ((a, \alpha), a') \in \mathcal{U}^{\text{ex}}(\llbracket v \rrbracket, \llbracket u \rrbracket)\}.$

PROOF. (i) We distinguish between two cases.

—If  $\Delta((v)u) = \emptyset$ , then Theorem 6.11 shows that  $l_h(((v)u, \emptyset). \epsilon) = \infty$  and Theorem 4.2 and Proposition 4.10 show that  $\mathcal{U}(\llbracket v \rrbracket, \llbracket u \rrbracket) = \emptyset$ .

—Else, we have

$$\begin{aligned} & l_h(((v)u, \emptyset). \epsilon) \\ &= \min\{|\Pi| + |\Pi'| + 1 / (\Pi, \Pi') \in \bigcup_{(b, \beta) \in \mathcal{M}_{\text{fin}}(D) \times D} (\Delta(v, (b, \beta)) \times \Delta(u, b))\} \\ & \quad (\text{by Theorem 6.11}) \\ &= \min\{|(a, \alpha)| + |a'| + 1 / ((a, \alpha), a') \in \mathcal{U}(\llbracket v \rrbracket, \llbracket u \rrbracket)\} \\ & \quad (\text{by applying Proposition 6.27 and Corollary 6.28, and by noticing} \\ & \quad \text{that the atoms in } a \text{ can be assumed distinct of those in } a'). \end{aligned}$$

(ii) We distinguish between two cases.

—If  $\Delta^{\text{ex}}((v)u) = \emptyset$ , then Theorem 6.18 shows that  $l_\beta(((v)u, \emptyset). \epsilon) = \infty$  and Theorem 4.2 and Proposition 4.10 show that  $\mathcal{U}^{\text{ex}}(\llbracket v \rrbracket, \llbracket u \rrbracket) = \emptyset$ .

—Else, we have

$$\begin{aligned} & l_\beta(((v)u, \emptyset). \epsilon) \\ &= \min\{|\Pi| + |\Pi'| + 1 / (\Pi, \Pi') \in \bigcup_{(b, \beta) \in \mathcal{M}_{\text{fin}}(D) \times D^{\text{ex}}} (\Delta(v, (b, \beta)) \times \Delta(u, b))\} \\ & \quad (\text{by Theorem 6.18}) \\ &= \min\{|(a, \alpha)| + |a'| + 1 / ((a, \alpha), a') \in \mathcal{U}^{\text{ex}}(\llbracket v \rrbracket, \llbracket u \rrbracket)\} \\ & \quad (\text{by applying Proposition 6.27 and Corollary 6.28, and by noticing} \\ & \quad \text{that the atoms in } a \text{ can be assumed distinct of those in } a'). \end{aligned}$$

□

EXAMPLE 6.31. Set  $v = \lambda x.(x)x$  and  $u = \lambda y.y$ . Let  $\gamma_0, \gamma_1 \in A$ . Set

- $\alpha = \gamma_0$ ;
- $a = [\gamma_0, ([\gamma_0], \gamma_0)]$ ;
- $a' = [([\gamma_1], \gamma_1), ([\gamma_2], \gamma_2)]$ .

Let  $\sigma$  be a substitution such that  $\sigma(\gamma_0) = ([\gamma_0], \gamma_0)$ ,  $\sigma(\gamma_1) = \gamma_0$  and  $\sigma(\gamma_2) = \alpha$ . We have

- $(a, \alpha) \in \llbracket v \rrbracket$ ;
- $\text{Supp}(a') \subseteq \llbracket u \rrbracket$ ;
- $\overline{\sigma}(a) = \overline{\sigma}(a')$ ;
- $|(a, \alpha)| = 4$  and  $|a'| = 4$ .

By Example 2.7, we know that we have  $l_h(((v)u, \emptyset). \epsilon) = 9$ . And we have  $|(a, \alpha)| + |a'| + 1 = 9$ .

The following example shows that the assumption that  $A$  is infinite is necessary.

EXAMPLE 6.32. Let  $n$  be a nonzero integer. Set  $I = \lambda y.y$  and  $v = \lambda x.(x) \underbrace{I \dots I}_{n \text{ times}}$ .

We have

$$\mathcal{U}(\llbracket v \rrbracket, \llbracket I \rrbracket) \subseteq \{((\llbracket [\alpha_1], \alpha_1 \rrbracket) \dots \llbracket [\alpha_n], \alpha_n \rrbracket) \alpha, \alpha), \llbracket [\alpha_0], \alpha_0 \rrbracket) / \alpha_0, \dots, \alpha_n, \alpha \in D\}.$$

Let  $\gamma_0, \dots, \gamma_n, \delta \in A$  distinct. We have

$$(((\llbracket [\gamma_1], \gamma_1 \rrbracket) \dots \llbracket [\gamma_n], \gamma_n \rrbracket) \delta, \delta), \llbracket [\gamma_0], \gamma_0 \rrbracket) \in \mathcal{U}^{\text{ex}}(\llbracket v \rrbracket, \llbracket I \rrbracket).$$

Hence, for any  $\alpha_0, \dots, \alpha_n, \alpha \in D$ , if there exists  $i \in \{0, \dots, n\}$  such that  $\alpha_i \notin A$ , then we have

$$\begin{aligned} |((\llbracket [\alpha_1], \alpha_1 \rrbracket) \dots \llbracket [\alpha_n], \alpha_n \rrbracket) \alpha, \alpha)| + |(\llbracket [\alpha_0], \alpha_0 \rrbracket)| + 1 &> l_\beta(((v)u, \emptyset). \epsilon) \\ &= l_h(((v)u, \emptyset). \epsilon). \end{aligned}$$

On the other hand, if  $\gamma_0, \dots, \gamma_n \in A$ ,  $\alpha \in D$  and there exist  $i, j \in \{0, \dots, n\}$  such that  $i \neq j$  and  $\gamma_i = \gamma_j$ , then

$$(((\llbracket [\gamma_1], \gamma_1 \rrbracket) \dots \llbracket [\gamma_n], \gamma_n \rrbracket) \alpha, \alpha), \llbracket [\gamma_0], \gamma_0 \rrbracket) \notin \mathcal{U}(\llbracket v \rrbracket, \llbracket I \rrbracket).$$

All this shows that if  $\text{Card}(A) = n$ , then we do not have

$$l_h(((v)I, \emptyset). \epsilon) = \inf\{|(a, \alpha)| + |a'| + 1 / ((a, \alpha), a') \in \mathcal{U}(\llbracket v \rrbracket, \llbracket I \rrbracket)\};$$

neither

$$l_\beta(((v)I, \emptyset). \epsilon) = \inf\{|(a, \alpha)| + |a'| + 1 / ((a, \alpha), a') \in \mathcal{U}^{\text{ex}}(\llbracket v \rrbracket, \llbracket I \rrbracket)\}.$$

Note that, as the following example illustrates, the non-idempotency is crucial.

EXAMPLE 6.33. For any integer  $n \geq 1$ , set  $\bar{n} = \lambda f. \lambda x. \underbrace{(f) \dots (f)}_{n \text{ times}} x$  and  $I = \lambda y.y$ .

Let  $\gamma \in A$ . Set  $\alpha = ([\gamma], \gamma)$  and  $a = \underbrace{[\alpha, \dots, \alpha]}_{n \text{ times}}$ . We have  $(a, \alpha) \in \llbracket \bar{n} \rrbracket$  and  $\alpha \in \llbracket I \rrbracket$ .

We have  $l_h(((\bar{n})I, \emptyset). \epsilon) = 4(n+1) = 2n+3+2n+1 = |(a, \alpha)| + |a| + 1$  (see Example 6.20). But with idempotent types (as in System  $\mathcal{D}$ ), for any integers  $p, q \geq 1$ , we would have  $\mathcal{U}(\bar{p}, I) = \mathcal{U}(\bar{q}, I)$  (any Church integer  $\bar{n}$ , for  $n \geq 1$  has type  $((\gamma \rightarrow \gamma) \rightarrow (\gamma \rightarrow \gamma))$  in System  $\mathcal{D}$ ).

*Acknowledgements.* This work is partially the result of discussions with Thomas Ehrhard: I warmly thank him. I also thank Patrick Baillot, Simona Ronchi della Rocca and Kazushige Terui too for stimulating discussions.

## REFERENCES

- Barendregt, H. P. (1984) *The Lambda Calculus. Its Syntax and Semantics*, revised edition. North-Holland.
- Baillot, P. and Terui, K. (2004) Light types for polynomial time computation in lambda-calculus. In *Proceedings of LICS 2004*, IEEE Computer Society Press, 266–275.
- Benton, P. N., Bierman, G. M., de Paiva, V. C. V. and Hyland, J. M. E. (1992) Term assignment for intuitionistic linear logic. Technical Report 262, Computer Laboratory, University of Cambridge.
- Bierman, G. M. (1993) On Intuitionistic Linear Logic. PhD thesis, University of Cambridge.
- Bierman, G. M. (1995) What is a categorical model of intuitionistic linear logic? In *Proceedings of Conference on Typed Lambda Calculi and Applications*, volume 902. Springer-Verlag.

- Boudol, G., Curien, P.-L. and Lavatelli, C. (1999) A semantics for lambda calculi with resources. *Math. Struct. in Comp. Science* **9** (4), 437–482.
- Bucciarelli, A. and Ehrhard, T. (2001) On phase semantics and denotational semantics : the exponentials. *Annals of Pure and Applied Logic* **109** 205–241.
- de Carvalho, D. (2006) Execution time of Lambda-Terms via Non-Uniform Semantics and Intersection Types. *Preprint IML*.
- de Carvalho, D. (2007) Sémantiques de la logique linéaire et temps de calcul. PhD thesis, *Université Aix-Marseille 2*.
- de Carvalho, D., Pagani, M. and Tortora de Falco, L. (2008) A Semantic Measure of the Execution Time in Linear Logic. **RR 6441**, INRIA.
- Coppo, M., Dezani-Ciancaglini, M. and Venneri, B. (1980) Principal type schemes and  $\lambda$ -calculus semantics. In J. P. Seldin and J. R. Hindley (editors), *To H. B. Curry : Essays on Combinatory Logic, Lambda Calculus and Formalism*, 535–560. Academic Press.
- Dezani-Ciancaglini, M., Honsell, F. and Motohama, Y. (2005) Compositional characterisations of  $\lambda$ -terms using intersection types. *Theoretical Computer Science* **340** (3), 459–496.
- Ehrhard, T. and Regnier, L. (2006) Böhm Trees, Krivine’s Machine and the Taylor Expansion of Lambda-Terms. In A. Beckmann, U. Berger, B. Löwe and J. V. Tucker (editors), *Logical Approaches to Computational Barriers, Second Conference on Computability in Europe, CiE 2006, Swansea, UK, June 30-July 5, 2006, Proceedings*, 186–197. Springer-Verlag.
- Girard, J. Y. (1986) The system F of variable types, fifteen years later. *Theoretical Computer Science* **45** (2), 159–192.
- Girard, J. Y. (1987) Linear Logic. *Theoretical Computer Science* **50**, 1–102.
- Guerrini, S. (2004) Proof Nets and the  $\lambda$ -calculus. In T. Ehrhard, J.-Y. Girard, P. Ruet and P. Scott (editors), *Linear Logic in Computer Science*, 65–118, Cambridge University Press.
- Kfoury, K., Mairson, H. G., Turbak, F. A. and Wells, J. B. (1999) Relating Typability and Expressiveness in Finite-Rank Intersection Types Systems (Extended Abstract), *ICFP*, 90–101.
- Kfoury, A. J. (2000) A linearization of the Lambda-calculus and consequences. *Journal of Logic and Computation* **10** (3), 411–436.
- Krivine, J. L. (1990) *Lambda-calcul types et modèles*. Masson.
- Krivine, J. L. (2007) A call-by-name lambda-calculus machine. *Higher Order and Symbolic Computation* **20**, 199–207.
- Lafont, Y. (1988) Logiques, catégories et machines. PhD thesis, *Université Paris 7*.
- Lambek, J. and Scott, P. J. (1986) *Introduction to higher order categorical logic*. Cambridge University Press.
- Mac Lane, S. (1998) *Categories for the Working Mathematician*. Springer-Verlag.
- Mascari, G. F. and Pedicini, M. (1994) Head linear reduction and pure proof net extraction. *Theoretical Computer Science* **135** (1), 111–137.
- Neergaard, P. M. and Mairson, H. G. (2004) Types, potency, and idempotency: why nonlinearity and amnesia make a type system work. In *ICFP ’04: Proceedings of the ninth ACM SIGPLAN international conference on Functional programming*, 138–149, ACM Press.
- Regnier, L. (1992) Lambda-calcul et réseaux. PhD thesis, *Université Paris 7*.
- Ronchi Della Rocca, S. (1988) Principal Type Scheme and Unification for Intersection Type Discipline. *Theoretical Computer Science* **59**, 181–209.
- Seely, R. (1989) Linear logic, \*-autonomous categories and cofree coalgebras. *Contemporary Mathematics* **92**.
- Selinger, P. (2002) The Lambda Calculus is Algebraic. *Journal of Functional Programming* **12** (6), 549–566.
- Tortora de Falco, L. (2000) Réseaux, cohérence et expériences obsessionnelles. PhD thesis, *Université Paris 7*.