Arithmetical proofs of strong normalization results for symmetric lambda calculi

René David, Karim Nour

Fundamenta Informaticae XXI (2001) 1001–1022 1001
IOS Press

Arithmetical proofs of strong normalization results for symmetric λ-calculi

René David & Karim Nour
Laboratoire de Mathématiques
Université de Savoie
73376 Le Bourget du Lac. France
$\{$*david,nour*$\}$*@univ-savoie.fr*

Abstract. We give *arithmetical* proofs of the strong normalization of two symmetric λ-calculi corresponding to classical logic.
The first one is the $\overline{\lambda}\mu\tilde{\mu}$-calculus introduced by Curien & Herbelin. It is derived via the Curry-Howard correspondence from Gentzen's classical sequent calculus LK in order to have a symmetry on one side between "program" and "context" and on other side between "call-by-name" and "call-by-value".
The second one is the symmetric $\lambda\mu$-calculus. It is the $\lambda\mu$-calculus introduced by Parigot in which the reduction rule μ', which is the symmetric of μ, is added.
These results were already known but the previous proofs use candidates of reducibility where the interpretation of a type is defined as the fix point of some increasing operator and thus, are highly non arithmetical.

Keywords: λ-calculus, symmetric calculi, classical logic, strong normalization.

1. Introduction

Since it has been understood that the Curry-Howard correspondence relating proofs and programs can be extended to classical logic (Felleisen [13], Griffin [15]), various systems have been introduced: the λ_c-calculus (Krivine [17]), the λ_{exn}-calculus (de Groote [6]), the $\lambda\mu$-calculus (Parigot [23]), the λ^{Sym}-calculus (Barbanera & Berardi [1]), the λ_Δ-calculus (Rehof & Sorensen [29]), the $\overline{\lambda}\mu\tilde{\mu}$-calculus (Curien & Herbelin [4]), the dual calculus (Wadler [31]), ... Only a few of them have computation rules that correspond to the symmetry of classical logic.

Address for correspondence: René David. Laboratoire de Mathématiques. Campus scientifique. F-73376 Le Bourget du Lac. email : david@univ-savoie.fr

We consider here the $\overline{\lambda}\mu\tilde{\mu}$-calculus and the symmetric $\lambda\mu$-calculus and we give arithmetical proofs of the strong normalization of the simply typed calculi. Though essentially the same proof can be done for the λ^{Sym}-calculus, we do not consider here this calculus since it is somehow different from the previous ones: its main connector is not the arrow but the connectors *or* and *and* and the symmetry of the calculus comes from the de Morgan laws. This proof will appear in Battyanyi's PhD thesis [2] who will also consider the dual calculus. Note that Dougherty & all [12] have shown the strong normalization of this calculus by the reducibility method using the technique of the fixed point construction.

The first proof of strong normalization for a symmetric calculus is the one by Barbanera & Berardi for the λ^{Sym}-calculus. It uses candidates of reducibility but, unlike the usual construction (for example for Girard's system F), the definition of the interpretation of a type needs a rather complex fix-point operation. Yamagata [32] has used the same technic to prove the strong normalization of the symmetric $\lambda\mu$-calculus where the types are those of system F and Parigot, again using the same ideas, has extended Barbanera & Berardi's result to a logic with second order quantification. Polonovsky, using the same technic, has proved in [27] the strong normalization of the $\overline{\lambda}\mu\tilde{\mu}$-reduction. These proofs are highly non arithmetical.

The two proofs that we give are essentially the same but the proof for the $\overline{\lambda}\mu\tilde{\mu}$-calculus is much simpler since some difficult problems that appear in the $\lambda\mu$-calculus do not appear in the $\overline{\lambda}\mu\tilde{\mu}$-calculus. In the $\overline{\lambda}\mu\tilde{\mu}$-calculus, a μ or a λ cannot be created at the root of a term by a reduction but this is not the case for the symmetric $\lambda\mu$-calculus. This is mainly due to the fact that, in the former, there is a right-hand side and a left-hand side whereas, in the latter, this distinction is impossible since a term on the right of an application can go on the left of an application after some reductions.

The idea of the proofs given here comes from the one given by the first author for the simply typed λ-calculus : assuming that a typed term has an infinite reduction, we can define, by looking at some particular steps of this reduction, an infinite sequence of strictly decreasing types. This proof can be found either in [7] (where it appears among many other things) or as a simple unpublished note on the web page of the first author (`www.lama.univ-savoie.fr/~david`).

We also show the strong normalization of the $\mu\tilde{\mu}$-reduction (resp. the $\mu\mu'$-reduction) for the un-typed calculi. The first result was already known and it can be found in [27]. The proof is done (by using candidates of reducibility and a fix point operator) for a typed calculus but, in fact, since the type system is such that every term is typable, the result is valid for every term. It was known that, for the un-typed $\lambda\mu$-calculus, the μ-reduction is strongly normalizing (see [28]) but the strong normalization of the $\mu\mu'$-reduction was an open problem raised long ago by Parigot. Studying this reduction by itself is interesting since a μ (or μ')-reduction can be seen as a way "to put the arguments of the μ where they are used" and it is useful to know that this is terminating.

This paper is an extension of [11]. In particular, section 4 essentially appears there. It is organized as follows. Section 2 gives the syntax of the terms of the $\overline{\lambda}\mu\tilde{\mu}$-calculus and the symmetric $\lambda\mu$-calculus and their reduction rules. Section 3 is devoted to the proof of the normalization results for the $\overline{\lambda}\mu\tilde{\mu}$-calculus and section 4 for the symmetric $\lambda\mu$-calculus. We conclude in section 5 with some remarks and future work.

2. The calculi

2.1. The $\overline{\lambda}\mu\tilde{\mu}$-calculus

2.1.1. The un-typed calculus

There are three kinds of terms, defined by the following grammar, and there are two kinds of variables. In the literature, different authors use different terminology. Here, we will call them either c-terms, or l-terms or r-terms. Similarly, the variables will be called either l-variables (and denoted as $x, y, ...$) or r-variables (and denoted as $\alpha, \beta, ...$).
In the rest of the paper, by term we will mean any of these three kind of terms.

$$\begin{array}{lcccccccc} c & ::= & \langle t_l, t_r \rangle & & & & & & \\ t_l & ::= & x & \mid & \lambda x\, t_l & \mid & \mu\alpha\, c & \mid & t_r.t_l \\ t_r & ::= & \alpha & \mid & \lambda\alpha\, t_r & \mid & \mu x\, c & \mid & t_l.t_r \end{array}$$

Remark 2.1. t_l (resp. t_r) stands of course for the left (resp. right) part of a c-term. At first look, it may be strange that, in the typing rules below, left terms appear in the right part of a sequent and vice-versa. This is just a matter of convention and an other choice could have been done. Except the change of name (done to make easier the analogy between the proofs for $\overline{\lambda}\mu\tilde{\mu}$-calculus and the symmetric $\lambda\mu$-calculus) we have respected the notations of the literature on this calculus.

2.1.2. The typed calculus

The logical part of this calculus is the (classical) sequent calculus which is, intrinsically, symmetric. The types are built from atomic formulas with the connectors $\rightarrow$ and $-$ where the intuitive meaning of $A - B$ is "A and not B". The typing system is a sequent calculus based on judgments of the following form:

$$c : (\Gamma \vdash \triangle) \qquad \Gamma \vdash \boxed{t_l : A}, \triangle \qquad \Gamma, \boxed{t_r : A} \vdash \triangle$$

where Γ (resp. $\triangle$) is a l-context (resp. a r-context), i.e. a set of declarations of the form $x : A$ (resp. $\alpha : A$) where x (resp. α) is a l-variable (resp. a r-variable) and A is a type.

$$\frac{}{\Gamma, x : A \vdash \boxed{x : A}, \triangle} \qquad \frac{}{\Gamma, \boxed{\alpha : A} \vdash \alpha : A, \triangle}$$

$$\frac{\Gamma, x : A \vdash \boxed{t_l : B}, \triangle}{\Gamma \vdash \boxed{\lambda x\, t_l : A \rightarrow B}, \triangle} \qquad \frac{\Gamma \vdash \boxed{t_l : A}, \triangle \quad \Gamma, \boxed{t_r : B} \vdash \triangle}{\Gamma, \boxed{t_l.t_r : A \rightarrow B} \vdash \triangle}$$

$$\frac{\Gamma \vdash \boxed{t_l : A}, \triangle \quad \Gamma, \boxed{t_r : B} \vdash \triangle}{\Gamma \vdash \boxed{t_r.t_l : A - B}, \triangle} \qquad \frac{\Gamma, \boxed{t_r : A} \vdash \alpha : B, \triangle}{\Gamma, \boxed{\lambda\alpha\, t_r : A - B} \vdash \triangle}$$

$$\frac{\Gamma \vdash \boxed{t_l : A}, \triangle \quad \Gamma, \boxed{t_r : A} \vdash \triangle}{\langle t_l, t_r \rangle : (\Gamma \vdash \triangle)}$$

$$\frac{c : (\Gamma \vdash \alpha : A, \triangle)}{\Gamma \vdash \boxed{\mu\alpha\, c : A}, \triangle} \qquad \frac{c : (\Gamma, x : A \vdash \triangle)}{\Gamma, \boxed{\mu x\, c : A} \vdash \triangle}$$

2.1.3. The reduction rules

The cut-elimination procedure (on the logical side) corresponds to the reduction rules (on the terms) given below.

- $\langle \lambda x\, t_l, t'_l.t_r \rangle \triangleright_\lambda \langle t'_l, \mu x\, \langle t_l, t_r \rangle \rangle$
- $\langle t'_r.t_l, \lambda \alpha\, t_r \rangle \triangleright_{\overline{\lambda}} \langle \mu \alpha\, \langle t_l, t_r \rangle, t'_r \rangle$
- $\langle \mu \alpha\, c, t_r \rangle \triangleright_\mu c[\alpha := t_r]$
- $\langle t_l, \mu x\, c \rangle \triangleright_{\tilde{\mu}} c[x := t_l]$
- $\mu \alpha\, \langle t_l, \alpha \rangle \triangleright_{s_l} t_l$ if $\alpha \notin Fv(t_l)$
- $\mu x\, \langle x, t_r \rangle \triangleright_{s_r} t_r$ if $x \notin Fv(t_r)$

Remark 2.2. It is easy to show that the $\mu\tilde{\mu}$-reduction is not confluent. For example $\langle \mu\alpha\, \langle x, \beta \rangle, \mu y\, \langle x, \alpha \rangle \rangle$ reduces both to $\langle x, \beta \rangle$ and to $\langle x, \alpha \rangle$.

Definition 2.1.
- We denote by $\triangleright_l$ the reduction by one of the logical rules i.e. $\triangleright_\lambda$, $\triangleright_{\overline{\lambda}}$, $\triangleright_\mu$ or $\triangleright_{\tilde{\mu}}$.
- We denote by $\triangleright_s$ the reduction by one of the simplification rules i.e. $\triangleright_{s_l}$ or $\triangleright_{s_r}$

2.2. The symmetric $\lambda\mu$-calculus

2.2.1. The un-typed calculus

The set (denoted as $\mathcal{T}$) of $\lambda\mu$-terms or simply terms is defined by the following grammar where $x, y, ...$ are λ-variables and $\alpha, \beta, ...$ are μ-variables:

$$\mathcal{T} ::= x \mid \lambda x \mathcal{T} \mid (\mathcal{T}\ \mathcal{T}) \mid \mu\alpha\mathcal{T} \mid (\alpha\ \mathcal{T})$$

Note that we adopt here a more liberal syntax (also called de Groote's calculus) than in the original calculus since we do not ask that a $\mu\alpha$ is immediately followed by a $(\beta\ M)$ (denoted $[\beta]M$ in Parigot's notation).

2.2.2. The typed calculus

The logical part of this calculus is natural deduction. The types are those of the simply typed $\lambda\mu$-calculus i.e. are built from atomic formulas and the constant symbol $\perp$ with the connector $\to$. As usual $\neg A$ is an abbreviation for $A \to \perp$.

The typing rules are given below where Γ is a context, i.e. a set of declarations of the form $x : A$ and $\alpha : \neg A$ where x is a λ (or intuitionistic) variable, α is a μ (or classical) variable and A is a formula.

$$\frac{}{\Gamma, x : A \vdash x : A}\ ax$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x M : A \to B}\ \to_i \qquad \frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash (M\ N) : B}\ \to_e$$

$$\frac{\Gamma, \alpha : \neg A \vdash M : \perp}{\Gamma \vdash \mu\alpha M : A}\ \perp_e \qquad \frac{\Gamma, \alpha : \neg A \vdash M : A}{\Gamma, \alpha : \neg A \vdash (\alpha\ M) : \perp}\ \perp_i$$

Note that, here, we also have changed Parigot's notation but these typing rules are those of his classical natural deduction. Instead of writing

$$M : (A_1^{x_1}, ..., A_n^{x_n} \vdash B, C_1^{\alpha_1}, ..., C_m^{\alpha_m})$$

we have written

$$x_1 : A_1, ..., x_n : A_n, \alpha_1 : \neg C_1, ..., \alpha_m : \neg C_m \vdash M : B$$

2.2.3. The reduction rules

The cut-elimination procedure (on the logical side) corresponds to the reduction rules (on the terms) given below. Natural deduction is not, intrinsically, symmetric but Parigot has introduced the so called *Free deduction* [22] which is completely symmetric. The $\lambda\mu$-calculus comes from there. To get a confluent calculus he had, in his terminology, to fix the inputs on the left. To keep the symmetry, it is enough to add a new reduction rule (called the μ'-reduction) which is the symmetric rule of the μ-reduction and also corresponds to the elimination of a cut.

- $(\lambda x M\ N) \triangleright_\beta M[x := N]$
- $(\mu\alpha M\ N) \triangleright_\mu \mu\alpha M[\alpha =_r N]$
- $(N\ \mu\alpha M) \triangleright_{\mu'} \mu\alpha M[\alpha =_l N]$
- $(\alpha\ \mu\beta M) \triangleright_\rho M[\beta := \alpha]$
- $\mu\alpha(\alpha\ M) \triangleright_\theta M$ if α is not free in M.

where $M[\alpha =_r N]$ (resp. $M[\alpha =_l N]$) is obtained by replacing each sub-term of M of the form $(\alpha\ U)$ by $(\alpha\ (U\ N))$ (resp. $(\alpha\ (N\ U))$). This substitution is called a μ-substitution (resp. a μ'-substitution).

Remark 2.3. 1. It is shown in [23] that the $\beta\mu$-reduction is confluent but neither $\mu\mu'$ nor $\beta\mu'$ is. For example $(\mu\alpha x\ \mu\beta y)$ reduces both to $\mu\alpha x$ and to $\mu\beta y$. Similarly $(\lambda z x\ \mu\beta y)$ reduces both to x and to $\mu\beta y$.

2. Unlike for a β-substitution where, in $M[x := N]$, the variable x has disappeared it is important to note that, in a μ or μ'-substitution, the variable α has not disappeared. Moreover its type has changed. If the type of N is A and, in M, the type of α is $\neg(A \to B)$ it becomes $\neg B$ in $M[\alpha =_r N]$. If the type of N is $A \to B$ and, in M, the type of α is $\neg A$ it becomes $\neg B$ in $M[\alpha =_l N]$.

3. In section 4, we will *not* consider the rules θ and ρ. The rule θ causes no problem since it is strongly normalizing and it is easy to see that this rule can be postponed. However, unlike for the $\overline{\lambda}\mu\tilde{\mu}$-calculus where all the simplification rules can be postponed, this is not true for the rule ρ and, actually, Battyanyi has shown in [2] that $\mu\mu'\rho$ is *not* strongly normalizing. However he has shown that $\mu\mu'\rho$ (in the untyped case) and $\beta\mu\mu'\rho$ (in the typed case) are *weakly* normalizing.

2.3. Some notations

The following notations will be used for both calculi. It will also be important to note that, in section 3 and 4, we will use the same notations (for example Σ_l, Σ_r) for objects concerning respectively the $\overline{\lambda}\mu\tilde{\mu}$-calculus and the symmetric $\lambda\mu$-calculus. This is done intentionally to show the analogy between the proofs.

Definition 2.2. Let u, v be terms.

1. $cxty(u)$ is the number of symbols occurring in u.
2. We denote by $u \leq v$ (resp. $u < v$) the fact that u is a sub-term (resp. a strict sub-term) of v.
3. A *proper* term is a term that is not a variable.
4. If σ is a substitution and u is a term, we denote by
   - $\sigma + [x := u]$ the substitution σ' such that for $y \neq x$, $\sigma'(y) = \sigma(y)$ and $\sigma'(x) = u$
   - $\sigma[x := u]$ the substitution σ' such that $\sigma'(y) = \sigma(y)[x := u]$.

Definition 2.3. Let A be a type. We denote by $lg(A)$ the number of symbols in A.

In the next sections we will study various reductions. The following notions will correspond to these reductions.

Definition 2.4. Let $\triangleright$ be a notion of reduction.

1. The transitive (resp. reflexive and transitive) closure of $\triangleright$ is denoted by $\triangleright^+$ (resp. $\triangleright^*$). The length (i.e. the number of steps) of the reduction $t \triangleright^* t'$ is denoted by $lg(t \triangleright^* t')$.
2. If t is in SN i.e. t has no infinite reduction, $\eta(t)$ will denote the length of the longest reduction starting from t and $\eta c(t)$ will denote $(\eta(t), cxty(t))$.
3. We denote by $u \prec v$ the fact that $u \leq w$ for some w such that $v \triangleright^* w$ and either $v \triangleright^+ w$ or $u < w$. We denote by $\preceq$ the reflexive closure of $\prec$.

Remark 2.4. - It is easy to check that the relation $\preceq$ is transitive, that $u \preceq v$ iff $u \leq w$ for some w such that $v \triangleright^* w$. We can also prove (but we will not use it) that the relation $\preceq$ is an order on the set SN.

- If $v \in SN$ and $u \prec v$, then $u \in SN$ and $\eta c(u) < \eta c(v)$.

- In the proofs done by induction on some k-uplet of integers, the order we consider is the lexicographic order.

3. Normalization for the $\overline{\lambda}\mu\tilde{\mu}$-calculus

The following lemma will be useful.

Lemma 3.1. Let t be a l-term (resp. a r-term). If $t \in SN$, then $\langle t, \alpha \rangle \in SN$ (resp. $\langle x, t \rangle \in SN$).

Proof By induction on $\eta(t)$. Since $\langle t, \alpha \rangle \notin SN$, $\langle t, \alpha \rangle \triangleright u$ for some u such that $u \notin SN$. If $u = \langle t', \alpha \rangle$ where $t \triangleright t'$ we conclude by the induction hypothesis since $\eta(t') < \eta(t)$. If $t = \mu\beta\, c$ and $u = c[\beta := \alpha] \notin SN$, then $c \notin SN$ and $t \notin SN$. Contradiction. $\square$

3.1. $\triangleright_s$ can be postponed

Definition 3.1. 1. Let $\triangleright_{\mu_0}, \triangleright_{\tilde{\mu}_0}$ be defined as follows:

- $\langle \mu\alpha\, c, t_r \rangle \triangleright_{\mu_0} c[\alpha := t_r]$ if α occurs at most once in c
- $\langle t_l, \mu x\, c \rangle \triangleright_{\tilde{\mu}_0} c[x := t_l]$ if x occurs at most once in c

2. Let $\triangleright_{l_0} = \triangleright_{\mu_0} \cup \triangleright_{\tilde{\mu}_0}$.

Lemma 3.2. If $u \triangleright_s v \triangleright_l w$, then there is t such that $u \triangleright_l t \triangleright^*_s w$ or $u \triangleright_{l_0} t \triangleright_l w$.

Proof By induction on u. $\square$

Lemma 3.3. If $u \triangleright_s v \triangleright_{l_0} w$, then either $u \triangleright_{l_0} w$ or, for some t, $u \triangleright_{l_0} t \triangleright_s w$ or $u \triangleright_{l_0} t \triangleright_{l_0} w$.

Proof By induction on u. $\square$

Lemma 3.4. If $u \triangleright^*_s v \triangleright_{l_0} w$ then, for some t, $u \triangleright^+_{l_0} t \triangleright^*_s w$ and $lg(u \triangleright^*_s v \triangleright_{l_0} w) \leq lg(u \triangleright^+_{l_0} t \triangleright^*_s w)$.

Proof By induction on $lg(u \triangleright^*_s v \triangleright_{l_0} w)$. Use lemma 3.3. $\square$

Lemma 3.5. If $u \triangleright^*_s v \triangleright_l w$ then, for some t, $u \triangleright^+_l t \triangleright^*_s w$.

Proof By induction on $lg(u \triangleright^*_s v \triangleright_l w)$. Use lemmas 3.2 and 3.4. $\square$

Corollary 3.1. $\triangleright_s$ can be postponed.

Proof By lemma 3.5. $\square$

Lemma 3.6. The s-reduction is strongly normalizing.

Proof If $u \triangleright_s v$, then $cxty(u) > cxty(v)$. $\square$

Theorem 3.1. 1. If t is strongly normalizing for the l-reduction, then it is also strongly normalizing for the ls-reduction .

2. If t is strongly normalizing for the $\mu\tilde{\mu}$-reduction, then it is also strongly normalizing for the $\mu\tilde{\mu}s$-reduction.

Proof Use lemmas 3.6 and 3.1. It is easy to check that the lemma 3.1 remains true if we consider only the reduction rules μ and $\tilde{\mu}$. $\square$

3.2. The $\mu\tilde{\mu}$-reduction is strongly normalizing

In this section we consider only the $\mu\tilde{\mu}$-reduction and we restrict the set of terms to the following grammar.

$$
\begin{array}{ccccc}
c & ::= & \langle t_l, t_r \rangle & & \\
t_l & ::= & x & | & \mu\alpha\, c \\
t_r & ::= & \alpha & | & \mu x\, c
\end{array}
$$

It is easy to check that, to prove the strong normalization of the full calculus with the $\mu\tilde{\mu}$-reduction, it is enough to prove the strong normalization of this restricted calculus.

Remember that we are, here, in the un-typed caculus and thus our proof does not use types but the strong normalization of this calculus actually follows from the result of the next section: it is easy to check that, in this restricted calculus, every term is typable by any type, in the context where the free variables are given this type. We have kept this section since the main ideas of the proof of the general case already appear here and this is done in a simpler situation.

The main point of the proof is the following. It is easy to show that if $t \in SN$ but $t[x := t_l] \notin SN$, there is some $\langle x, t_r \rangle \prec t$ such that $t_r[x := t_l] \in SN$ and $\langle t_l, t_r[x := t_l] \rangle \notin SN$. But this is not enough and we need a stronger (and more difficult) version of this: lemma 3.8 ensures that, if $t[\sigma] \in SN$ but $t[\sigma][x := t_l] \notin SN$ then the real cause of non SN is, in some sense, $[x := t_l]$.

Having this result, we show, essentially by induction on $\eta c(t_l) + \eta c(t_r)$, that if $t_l, t_r \in SN$ then $\langle t_l, t_r \rangle \in SN$. The point is that there is, in fact, no deep interactions between t_l and t_r i.e. in a reduct of $\langle t_l, t_r \rangle$ we always know what is coming from t_l and what is coming from t_r. The final result comes then from a trivial induction on the terms.

Definition 3.2.

- We denote by Σ_l (resp. Σ_r) the set of simultaneous substitutions of the form $[x_1 := t_1, ..., x_n := t_n]$ (resp. $[\alpha_1 := t_1, ..., \alpha_n := t_n]$) where $t_1, ..., t_n$ are proper l-terms (r-terms).
- For $s \in \{l, r\}$, if $\sigma = [\xi_1 := t_1, ..., \xi_n := t_n] \in \Sigma_s$, we denote by $dom(\sigma)$ (resp. $Im(\sigma)$) the set $\{\xi_1, ..., \xi_n\}$ (resp. $\{t_1, ..., t_n\}$).

Lemma 3.7. Assume $t_l, t_r \in SN$ and $\langle t_l, t_r \rangle \notin SN$. Then either $t_l = \mu\alpha\, c$ and $c[\alpha := t_r] \notin SN$ or $t_r = \mu x\, c$ and $c[x := t_l] \notin SN$.

Proof By induction on $\eta(t_l) + \eta(t_r)$. Since $\langle t_l, t_r \rangle \notin SN$, $\langle t_l, t_r \rangle \triangleright t$ for some t such that $t \notin SN$. If $t = \langle t'_l, t_r \rangle$ where $t_l \triangleright t'_l$, we conclude by the induction hypothesis since $\eta(t'_l) + \eta(t_r) < \eta(t_l) + \eta(t_r)$. If $t = \langle t_l, t'_r \rangle$ where $t_r \triangleright t'_r$, the proof is similar. If $t_l = \mu\alpha\, c$ and $t = c[\alpha := t_r] \notin SN$ or $t_r = \mu x\, c$ and $t = c[x := t_l] \notin SN$, the result is trivial. □

Lemma 3.8. 1. Let t be a term, t_l a l-term and $\tau \in \Sigma_l$. Assume $t_l \in SN$, x is free in t but not free in $Im(\tau)$. If $t[\tau] \in SN$ but $t[\tau][x := t_l] \notin SN$, there is $\langle x, t_r \rangle \prec t$ and $\tau' \in \Sigma_l$ such that $t_r[\tau'] \in SN$ and $\langle t_l, t_r[\tau'] \rangle \notin SN$.

2. Let t be a term, t_r a r-term and $\sigma \in \Sigma_r$. Assume $t_r \in SN$, α is free in t but not free in $Im(\sigma)$. If $t[\sigma] \in SN$ but $t[\sigma][\alpha := t_r] \notin SN$, there is $\langle t_l, \alpha \rangle \prec t$ and $\sigma' \in \Sigma_r$ such that $t_l[\sigma'] \in SN$ and $\langle t_l[\sigma'], t_r \rangle \notin SN$.

Proof We prove the case (1) (the case (2) is similar). Note that t_l is proper since $t[\tau] \in SN$, $t[\tau][x := t_l] \notin SN$ and x is not free in $Im(\tau)$. Let $Im(\tau) = \{t_1, ..., t_k\}$. Let $\mathcal{U} = \{u \,/\, u$ is proper and $u \preceq t\}$ and $\mathcal{V} = \{v \,/\, v$ is proper and $v \preceq t_i$ for some $i\}$. Define inductively the sets Σ'_l and Σ'_r of substitutions by the following rules:

$\rho \in \Sigma'_l$ iff $\rho = \emptyset$ or $\rho = \rho' + [y := v[\delta]]$ for some l-term $v \in \mathcal{V}$, $\delta \in \Sigma'_r$ and $\rho' \in \Sigma'_l$
$\delta \in \Sigma'_r$ iff $\delta = \emptyset$ or $\delta = \delta' + [\beta := u[\rho]]$ for some r-term $u \in \mathcal{U}$, $\rho \in \Sigma'_l$ and $\delta' \in \Sigma'_r$

Denote by C the conclusion of the lemma, i.e. there is $\langle x, t_r \rangle \prec t$ and $\tau' \in \Sigma_l$ such that $t_r[\tau'] \in SN$ and $\langle t_l, t_r[\tau'] \rangle \notin SN$. We prove something more general.

(1) If $u \in \mathcal{U}$, $\rho \in \Sigma'_l$, $u[\rho] \in SN$ and $u[\rho][x := t_l] \notin SN$, then C holds.
(2) If $v \in \mathcal{V}$, $\delta \in \Sigma'_r$, $v[\delta] \in SN$ and $v[\delta][x := t_l] \notin SN$, then C holds.

The term t is proper since $t[\tau][x := t_l] \notin SN$. Then conclusion C follows from (1) with t and τ.

The properties (1) and (2) are proved by a simultaneous induction on $\eta c(u[\rho])$ (for the first case) and $\eta c(v[\delta])$ (for the second case). We only consider (1), the case (2) is proved in a similar way.

- If u begins with a μ. The result follows from the induction hypothesis.
- If $u = \langle u_l, u_r \rangle$.
  - If $u_r[\rho][x := t_l] \notin SN$: then u_r is proper and the result follows from the induction hypothesis.
  - If $u_l[\rho][x := t_l] \notin SN$ and u_l is proper: the result follows from the induction hypothesis.
  - If $u_l[\rho][x := t_l] \notin SN$ and $u_l = y \in dom(\rho)$. Let $\rho(y) = \mu\beta\, d[\delta]$, then $\mu\beta d[\delta][x := t_l] \notin SN$ and the result follows from the induction hypothesis with $\mu\beta d$ and δ (case (2)) since $\eta c(\mu\beta d[\delta]) < \eta c(u[\rho])$.
  - Otherwise, by lemma 3.7, there are two cases to consider. Note that u_r cannot be a variable because, otherwise, $u[\rho][x := t_l] = \langle u_l[\rho][x := t_l], u_r \rangle$ and thus, by lemma 3.1, $u[\rho][x := t_l]$ would be in SN.

    (1) $u_l[\rho][x := t_l] = \mu\alpha\, c$ and $c[\alpha := u_r[\rho][x := t_l]] \notin SN$.

    - If $u_l = \mu\alpha\, d$, then $d[\alpha := u_r][\rho][x := t_l] \notin SN$ and the result follows from the induction hypothesis with $d[\alpha := u_r]$ and ρ since $\eta(d[\alpha := u_r][\rho]) < \eta(u[\rho])$.

    - If $u_l = y \in dom(\rho)$, let $\rho(y) = \mu\beta\, d[\delta]$, then $d[\delta'][x := t_l] \notin SN$ where $\delta' = \delta + [\beta := u_r[\rho]]$ and the result follows from the induction hypothesis with d and δ' (case(2)).

- If $u_l = x$, then $\langle x, u_r \rangle$ and $\tau' = \rho[x := t_l]$ satisfy the desired conclusion.

(2) $u_r[\rho][x := t_l] = \mu y\, c$ and $c[\alpha := u_l[\rho][x := t_l]] \notin SN$. Then $u_r = \mu y\, d$ and $d[y := u_l][\rho][x := t_l] \notin SN$. The result follows from the induction hypothesis with $d[y := u_l]$ and ρ since $\eta(d[y := u_l][\rho]) < \eta(u[\rho])$. □

Theorem 3.2. The $\mu\tilde{\mu}$-reduction is strongly normalizing.

Proof By induction on the term. It is enough to show that, if $t_l, t_r \in SN$, then $\langle t_l, t_r \rangle \in SN$. We prove something more general: let σ (resp. τ) be in Σ_r (resp. Σ_l) and assume $t_l[\sigma], t_r[\tau] \in SN$. Then $\langle t_l[\sigma], t_r[\tau] \rangle \in SN$. Assume it is not the case and choose some elements such that $t_l[\sigma], t_r[\tau] \in SN$, $\langle t_l[\sigma], t_r[\tau] \rangle \notin SN$ and $(\eta(t_l) + \eta(t_r), cxty(t_l) + cxty(t_r))$ is minimal. By lemma 3.7, either $t_l[\sigma] = \mu\alpha\, c$ and $c[\alpha := t_r[\tau]] \notin SN$ or $t_r[\tau] = \mu x\, c$ and $c[x := t_l[\sigma]] \notin SN$. Look at the second case (the first one is similar). We have $t_r = \mu x\, d$ and $d[\tau] = c$, then $d[\tau][x := t_l[\sigma]] \notin SN$. By lemma 3.8, let $u_r \prec d$ and $\tau' \in \Sigma_l$ be such that $u_r[\tau'] \in SN$, $\langle t_l[\sigma], u_r[\tau'] \notin SN$. This contradicts the minimality of the chosen elements since $\eta c(u_r) < \eta c(t_r)$. □

3.3. The typed $\overline{\lambda}\mu\tilde{\mu}$-calculus is strongly normalizing

In this section, we consider the typed calculus with the l-reduction. By theorem 3.1, this is enough to prove the strong normalization of the full calculus. To simplify notations, we do not write explicitly the type information but, when needed, we denote by $type(t)$ the type of the term t.

The proof is essentially the same as the one of theorem 3.2. It relies on lemma 3.10 for which type considerations are needed: in its proof, some cases cannot be proved "by themselves" and we need an argument using the types. For this reason, its proof is done using the additional fact that we already know that, if $t_l, t_r \in SN$ and the type of t_r is small, then $t[x := t_r]$ also is in SN. Since the proof of lemma 3.11 is done by induction on the type, when we will use lemma 3.10, the additional hypothesis will be available.

Lemma 3.9. Assume $t_l, t_r \in SN$ and $\langle t_l, t_r \rangle \notin SN$. Then either ($t_l = \mu\alpha\, c$ and $c[\alpha := t_r] \notin SN$) or ($t_r = \mu x\, c$ and $c[x := t_l] \notin SN$) or ($t_l = \lambda x u_l$, $t_r = u_l'.u_r$ and $\langle u_l', \mu x \langle u_l, u_r \rangle \rangle \notin SN$) or ($t_r = \lambda\alpha u_r$, $t_l = u_r'.u_l$ and $\langle \mu\alpha \langle u_r, u_l \rangle, u_r' \rangle \notin SN$).

Proof By induction on $\eta(t_l) + \eta(t_r)$. □

Definition 3.3. Let A be a type. We denote $\Sigma_{A,l}$ (resp. $\Sigma_{A,r}$) the set of substitutions of the form $[x_1 := t_1, ..., x_n := t_n]$ (resp. $[\alpha_1 := t_1, ..., \alpha_n := t_n]$) where $t_1, ..., t_n$ are proper l-terms (resp. r-terms) and the type of the x_i (resp. α_i) is A.

Lemma 3.10. Let n be an integer and A be a type such that $lg(A) = n$. Assume H holds where H is: for every $u, v \in SN$ such that $lg(type(v)) < n$, $u[x := v] \in SN$.

1. Let t be a term, t_l a l-term and $\tau \in \Sigma_{A,l}$. Assume $t_l \in SN$ and has type A, x is free in t but not free in $Im(\tau)$. If $t[\tau] \in SN$ but $t[\tau][x := t_l] \notin SN$, there is $\langle x, t_r \rangle \prec t$ and $\tau' \in \Sigma_{A,l}$ such that $t_r[\tau'] \in SN$ and $\langle t_l, t_r[\tau'] \rangle \notin SN$.

2. Let t be a term, t_r a r-term and $\sigma \in \Sigma_{A,r}$. Assume $t_r \in SN$ and has type A, α is free in t but not free in $Im(\sigma)$. If $t[\sigma] \in SN$ but $t[\sigma][\alpha := t_r] \notin SN$, there is $\langle t_l, \alpha\rangle \prec t$ and $\sigma' \in \Sigma_{A,r}$ such that $t_l[\sigma'] \in SN$ and $\langle t_l[\sigma'], t_r\rangle \notin SN$.

Proof We only prove the case (1), the other one is similar. Note that t_l is proper since $t[\tau] \in SN$ and $t[\tau][x := t_l] \notin SN$. Let $Im(\tau) = \{t_1, ..., t_k\}$. Let $\mathcal{U} = \{u$ / u is proper and $u \preceq t\}$ and $\mathcal{V} = \{v$ / v is proper and $v \preceq t_i$ for some $i\}$. Define inductively the sets $\Sigma'_{A,l}$ and $\Sigma'_{A,r}$ of substitutions by the following rules:

$\rho \in \Sigma'_{A,l}$ iff $\rho = \emptyset$ or $\rho = \rho' + [y := v[\delta]]$ for some l-term $v \in \mathcal{V}$, $\delta \in \Sigma'_{A,r}$, $\rho' \in \Sigma'_{A,l}$ and y has type A.
$\delta \in \Sigma'_{A,r}$ iff $\delta = \emptyset$ or $\delta = \delta' + [\beta := u[\rho]]$ for some r-term $u \in \mathcal{U}$, $\rho \in \Sigma'_{A,l}$, $\delta' \in \Sigma'_{A,r}$ and β has type A.

Denote by C the conclusion of the lemma, i.e. there is $\langle x, t_r\rangle \prec t$ and $\tau' \in \Sigma_{A,l}$ such that $t_r[\tau'] \in SN$ and $\langle t_l, t_r[\tau']\rangle \notin SN$. We prove something more general.

(1) If $u \in \mathcal{U}$, $\rho \in \Sigma'_{A,l}$, $u[\rho] \in SN$ and $u[\rho][x := t_l] \notin SN$, then C holds.
(2) If $v \in \mathcal{V}$, $\delta \in \Sigma'_{A,r}$, $v[\delta] \in SN$ and $v[\delta][x := t_l] \notin SN$, then C holds.

Note that, since $t[\tau][x := t_l] \notin SN$, t is proper and thus, C follows from (1) with t and τ. The properties (1) and (2) are proved by a simultaneous induction on $\eta c(u[\rho])$ (for the first case) and $\eta c(v[\delta])$ (for the second case). We only consider (1) since (2) is similar.

The proof is as in lemma 3.8. We only consider the additional cases: $u = \langle u_l, u_r\rangle$, $u_l[\rho][x := t_l] \in SN$, $u_r[\rho][x := t_l] \in SN$, u_r is proper and one of the two following cases occurs.

- $u_l[\rho][x := t_l] = \lambda x v_l$, $u_r[\rho][x := t_l] = v'_l.v_r$ and $\langle v'_l, \mu x\langle v_l, v_r\rangle\rangle \notin SN$. Then, $u_r = w'_l.w_r$, $v'_l = w'_l[\rho][x := t_l]$ and $v_r = w_r[\rho][x := t_l]$. There are three cases to consider.
  - $u_l = \lambda x w_l$ and $w_l[\rho][x := t_l] = v_l$, then the result follows from the induction hypothesis with $\langle w'_l, \mu x\langle w_l, w_r\rangle\rangle$ and ρ since $\eta(\langle w'_l, \mu x\langle w_l, w_r\rangle\rangle[\rho]) < \eta(u[\rho])$.
  - $u_l = y \in dom(\rho)$. Let $\rho(y) = \lambda z w_l[\delta]$, then $a = \langle w'_l[\rho], \mu x\langle w_l[\delta], w_r[\rho]\rangle\rangle$ $[x := t_l] \notin SN$. But,

    - $b = w'_l[\rho][x := t_l], c = w_l[\delta][x := t_l], d = w_r[\rho][x := t_l] \in SN$,
    - $lg(type(b)) < n, lg(type(c)) < n$,
    - $a = \langle x_2, \mu x\langle x_1, d\rangle\rangle[x_1 := c][x_2 := b]$

    and this contradicts the hypothesis (H).
  - $u_l = x$, then $\langle x, u_r\rangle$ and $\tau' = \tau[x := t_l]$ satisfy the desired conclusion.

- $u_l[\rho][x := t_l] = v'_r.v_l$, $u_r[\rho][x := t_l] = \lambda\alpha v_r$ and $\langle \mu\alpha\langle v_l, v_r\rangle, v'_r\rangle \notin SN$. The proof is similar. □

Lemma 3.11. If $t, t_l, t_r \in SN$, then $t[x := t_l], t[\alpha := t_r] \in SN$.

Proof We prove something a bit more general: let A be a type and t a term.
(1) Let $t_1, ..., t_k$ be l-terms and $\tau_1, ..., \tau_k$ be substitutions in $\Sigma_{A,r}$. If, for each i, t_i has type A and $t_i[\tau_i] \in SN$, then $t[x_1 := t_1[\tau_1],\ ...,\ x_k := t_k[\tau_k]] \in SN$.
(2) Let $t_1, ..., t_k$ be r-terms and $\tau_1, ..., \tau_k$ be substitutions in $\Sigma_{A,l}$. If, for each i, t_i has type A and $t_i[\tau_i] \in SN$, then $t[\alpha_1 := t_1[\tau_1],\ ...,\ \alpha_k := t_k[\tau_k]] \in SN$.

We only consider (1) since (2) is similar. This is proved by induction on $(lg(A), \eta(t), cxty(t), \Sigma\ \eta(t_i), \Sigma\ cxty(t_i))$ where, in $\Sigma\ \eta(t_i)$ and $\Sigma\ cxty(t_i)$, we count each occurrence of the substituted variable. For example if $k = 1$ and x_1 has n occurrences, $\Sigma\ \eta(t_i) = n.\eta(t_1)$.

The only no trivial case is $t = \langle u_l, u_r\rangle$. Let $\sigma = [x_1 := t_1[\tau_1],\ ...,\ x_k := t_k[\tau_k]]$. By the induction hypothesis, $u_l[\sigma], u_r[\sigma] \in SN$. By lemma 3.9, there are four cases to consider.

- $u_l[\sigma] = \mu\alpha c$ and $c[\alpha := u_r[\sigma]] \notin SN$.
  - If $u_l = \mu\alpha d$ and $d[\sigma] = c$. Then $d[\alpha := u_r][\sigma] \notin SN$ and, since $\eta(d[\alpha := u_r]) < \eta(t)$, this contradicts the induction hypothesis.
  - If $u_l = x_i$, $t_i = \mu\alpha d$ and $d[\tau_i][\alpha := u_r[\sigma]] \notin SN$. By lemma 3.10, there is $v_l \preceq d$ and $\tau_i' \in \Sigma_{A,r}$ such that $v_l[\tau_i'] \in SN$ and $\langle v_l[\tau_i'], u_r[\sigma]\rangle \notin SN$. Let $t' = \langle y, u_r\rangle$ where y is a fresh variable and $\sigma' = \sigma + [y = v_l[\tau_i']]$. Then $\langle v_l[\tau_i'], u_r[\sigma]\rangle = t'[\sigma']$ and, since $(\eta(v_l), cxty(v_l)) < (\eta(t_i), cxty(t_i))$ we get a contradiction from the induction hypothesis.
- $u_r[\sigma] = \mu x c$ and $c[x := u_l[\sigma]] \notin SN$, then $u_r = \mu x d$, $d[\sigma] = c$ and $d[x := u_l][\sigma] \notin SN$. Since $\eta(d[x := u_l]) < \eta(t)$, this contradicts the induction hypothesis.
- $u_l[\sigma] = \lambda x v_l$, $u_r[\sigma] = v_l'.v_r$ and $\langle v_l', \mu x\langle v_l, v_r\rangle\rangle \notin SN$, then $u_r = w_l'.w_r$, $w_l'[\sigma] = v_l'$ and $w_r[\sigma] = v_r$.
  - If $u_l = \lambda x w_l$ and $w_l[\sigma] = v_l$. Then $\langle w_l', \mu x\langle w_l, w_r\rangle\rangle[\sigma] \notin SN$ and this contradicts the induction hypothesis, since $\eta(\langle w_l', \mu x\langle w_l, w_r\rangle\rangle) < \eta(t)$.
  - If $u_l = x_i$, $t_i = \lambda x w_l$ and $\langle w_l'[\sigma], \mu x\langle w_l[\tau_i],\ w_r[\sigma]\rangle\rangle \notin SN$. Then, $\langle w_l[\tau_i], w_r[\sigma]\rangle = \langle y, u_r[\sigma]\rangle[y := w_l[\tau_i]]$ where y is a fresh variable and thus $\langle w_l[\tau_i], w_r[\sigma]\rangle \in SN$, since $lg(type(w_l[\tau_i])) < lg(A)$.
    Since $\langle w_l'[\sigma], \mu x\langle w_l[\tau_i], w_r[\sigma]\rangle\rangle = \langle z, \mu x\langle w_l[\tau_i], w_r[\sigma]\rangle\rangle[z := w_l'[\sigma]]$ where z is a fresh variable and $lg(type(w_l'[\sigma])) < lg(A)$, this contradicts the induction hypothesis.
- $u_r[\sigma] = \lambda\alpha v_r$, $u_l[\sigma] = v_r'.v_l$ and $\langle \mu\alpha\langle v_l, v_r\rangle, v_r'\rangle \notin SN$. This is proved in the same way. □

Theorem 3.3. Every typed term is in SN.

Proof By induction on the term. It is enough to show that if $t_l, t_r \in SN$, then $\langle t_l, t_r\rangle \in SN$. Since $\langle t_l, t_r\rangle = \langle x, \alpha\rangle[x := t_l][\alpha := t_r]$ where x, α are fresh variables, the result follows from lemma 3.11. □

4. Normalization for the symmetric $\lambda\mu$-calculus

4.1. The $\mu\mu'$-reduction is strongly normalizing

In this section we consider the $\mu\mu'$-reduction, i.e. $M \triangleright M'$ means M' is obtained from M by one step of the $\mu\mu'$-reduction. The proof of theorem 4.1 is essentially the same as the one of theorem 3.2. We first show (cf. lemma 4.2) that a μ or μ'-substitution cannot create a μ and then we show (cf. lemma 4.4) that, if $M[\sigma] \in SN$ but $M[\sigma][\alpha =_r P] \notin SN$, then the real cause of non SN is, in some sense, $[\alpha =_r P]$. The main point is again that, in a reduction of $(M\ N) \in SN$, there is, in fact, no deep interactions between M and N i.e. in a reduct of $(M\ N)$ we always know what is coming from M and what is coming from N.

Definition 4.1.

- The set of simultaneous substitutions of the form $[\alpha_1 =_{s_1} P_1\ ...,\ \alpha_n =_{s_n} P_n]$ where $s_i \in \{l, r\}$ will be denoted by Σ.
- For $s \in \{l, r\}$, the set of simultaneous substitutions of the form $[\alpha_1 =_s P_1\ ...\alpha_n =_s P_n]$ will be denoted by Σ_s.
- If $\sigma = [\alpha_1 =_{s_1} P_1\ ...,\ \alpha_n =_{s_n} P_n]$, we denote by $dom(\sigma)$ (resp. $Im(\sigma)$) the set $\{\alpha_1,\ ...,\ \alpha_n\}$ (resp. $\{P_1,\ ...,\ P_n\}$).
- Let $\sigma \in \Sigma$. We say that $\sigma \in SN$ iff for every $N \in Im(\sigma)$, $N \in SN$.
- If $\overrightarrow{P}$ is a sequence $P_1, ..., P_n$ of terms, $(M\ \overrightarrow{P})$ will denote $(M\ P_1\ ...\ P_n)$.

Lemma 4.1. If $(M\ N) \triangleright^* \mu\alpha P$, then either $M \triangleright^* \mu\alpha M_1$ and $M_1[\alpha =_r N] \triangleright^* P$ or $N \triangleright^* \mu\alpha N_1$ and $N_1[\alpha =_l M] \triangleright^* P$.

Proof By induction on the length of the reduction $(M\ N) \triangleright^* \mu\alpha P$. □

Lemma 4.2. Let M be a term and $\sigma \in \Sigma$. If $M[\sigma] \triangleright^* \mu\alpha P$, then $M \triangleright^* \mu\alpha Q$ for some Q such that $Q[\sigma] \triangleright^* P$.

Proof By induction on M. M cannot be of the form $(\beta\, M')$ or $\lambda x\, M'$. If M begins with a μ, the result is trivial. Otherwise $M = (M_1\ M_2)$ and, by lemma 4.1, either $M_1[\sigma] \triangleright^* \mu\alpha R$ and $R[\alpha =_r M_2[\sigma]] \triangleright^* P$ or $M_2[\sigma] \triangleright^* \mu\alpha R$ and $R[\alpha =_l M_1[\sigma]] \triangleright^* P$. Look at the first case (the other one is similar). By the induction hypothesis $M_1 \triangleright^* \mu\alpha Q$ for some Q such that $Q[\sigma] \triangleright^* R$ and thus $M \triangleright^* \mu\alpha Q[\alpha =_r M_2]$. Since $Q[\alpha =_r M_2][\sigma] = Q[\sigma][\alpha =_r M_2[\sigma]] \triangleright^* R[\alpha =_r M_2[\sigma]] \triangleright^* P$ we are done. □

Lemma 4.3. Assume $M, N \in SN$ and $(M\ N) \notin SN$. Then either $M \triangleright^* \mu\alpha M_1$ and $M_1[\alpha =_r N] \notin SN$ or $N \triangleright^* \mu\beta N_1$ and $N_1[\beta =_l M] \notin SN$.

Proof By induction on $\eta(M) + \eta(N)$. Since $(M\ N) \notin SN$, $(M\ N) \triangleright P$ for some P such that $P \notin SN$. If $P = (M'\ N)$ where $M \triangleright M'$ we conclude by the induction hypothesis since $\eta(M') + \eta(N) < \eta(M) + \eta(N)$. If $P = (M\ N')$ where $N \triangleright N'$ the proof is similar. If $M = \mu\alpha M_1$ and $P = \mu\alpha M_1[\alpha =_r N]$ or $N = \mu\beta N_1$ and $P = \mu\beta N_1[\beta =_l M]$ the result is trivial. □

Lemma 4.4. Let M be a term and $\sigma \in \Sigma_s$. Assume δ is free in M but not free in $Im(\sigma)$. If $M[\sigma] \in SN$ but $M[\sigma][\delta =_s P] \notin SN$, there is $M' \prec M$ and σ' such that $M'[\sigma'] \in SN$ and, if $s = r$, $(M'[\sigma'] \;\; P) \notin SN$ and, if $s = l$, $(P \;\; M'[\sigma']) \notin SN$.

Proof Assume $s = r$ (the other case is similar). Let $Im(\sigma) = \{N_1, \ ..., \ N_k\}$. Assume M, δ, σ, P satisfy the hypothesis. Let $\mathcal{U} = \{U \ / \ U \preceq M\}$ and $\mathcal{V} = \{V \ / \ V \preceq N_i$ for some $i\}$. Define inductively the sets Σ_m and Σ_n of substitutions by the following rules:

$\rho \in \Sigma_m$ iff $\rho = \emptyset$ or $\rho = \rho' + [\beta =_r V[\tau]]$ for some $V \in \mathcal{V}$, $\tau \in \Sigma_n$ and $\rho' \in \Sigma_m$

$\tau \in \Sigma_n$ iff $\tau = \emptyset$ or $\tau = \tau' + [\alpha =_l U[\rho]]$ for some $U \in \mathcal{U}$, $\rho \in \Sigma_m$ and $\tau' \in \Sigma_n$

Denote by C the conclusion of the lemma, i.e. there is $M' \prec M$ and σ' such that $M'[\sigma'] \in SN$, and $(M'[\sigma'] \;\; P) \notin SN$.

We prove something more general.

(1) Let $U \in \mathcal{U}$ and $\rho \in \Sigma_m$. Assume $U[\rho] \in SN$ and $U[\rho][\delta =_r P] \notin SN$. Then, C holds.

(2) Let $V \in \mathcal{V}$ and $\tau \in \Sigma_n$. Assume $V[\tau] \in SN$ and $V[\tau][\delta =_r P] \notin SN$. Then, C holds.

The conclusion C follows from (1) with M and σ. The properties (1) and (2) are proved by a simultaneous induction on $\eta c(U[\rho])$ (for the first case) and $\eta c(V[\tau])$ (for the second case).

Look first at (1)

- if $U = \lambda x U'$ or $U = \mu\alpha U'$: the result follows from the induction hypothesis with U' and ρ.

- if $U = (U_1 \;\; U_2)$: if $U_i[\rho][\delta =_r P] \notin SN$ for $i = 1$ or $i = 2$, the result follows from the induction hypothesis with U_i and ρ. Otherwise, by lemma 4.2 and 4.3, say $U_1 \triangleright^* \mu\alpha U_1'$ and, letting $U' = U_1'[\alpha =_r U_2]$, $U'[\rho][\delta =_r P] \notin SN$ and the result follows from the induction hypothesis with U' and ρ.

- if $U = (\delta \; U_1)$: if $U_1[\rho][\delta =_r P] \in SN$, then $M' = U_1$ and $\sigma' = \rho[\delta =_r P]$ satisfy the desired conclusion. Otherwise, the result follows from the induction hypothesis with U_1 and ρ.

- if $U = (\alpha \; U_1)$: if $\alpha \notin dom(\rho)$ or $U_1[\rho][\delta =_r P] \notin SN$, the result follows from the induction hypothesis with U_1 and ρ. Otherwise, let $\rho(\alpha) = V[\tau]$. If $V[\tau][\delta =_r P] \notin SN$, the result follows from the induction hypothesis with V and τ (with (2)). Otherwise, by lemmas 4.2 and 4.3, there are two cases to consider.

- $U_1 \triangleright^* \mu\alpha_1 U_2$ and $U_2[\rho'][\delta =_r P] \notin SN$ where $\rho' = \rho + [\alpha_1 =_r V[\tau]]$. The result follows from the induction hypothesis with U_2 and ρ'.

- $V \triangleright^* \mu\beta V_1$ and $V_1[\tau'][\delta =_r P] \notin SN$ where $\tau' = \tau + [\beta =_l U_1[\rho]]$. The result follows from the induction hypothesis with V_1 and τ' (with (2)).

The case (2) is proved in the same way. Note that, since δ is not free in the N_i, the case $b = (\delta \; V_1)$ does not appear. □

Theorem 4.1. Every term is in SN.

Proof By induction on the term. It is enough to show that, if $M, N \in SN$, then $(M \;\; N) \in SN$. We prove something more general: let σ (resp. τ) be in Σ_r (resp. Σ_l) and assume $M[\sigma], N[\tau] \in SN$. Then $(M[\sigma] \;\; N[\tau]) \in SN$. Assume it is not the case and choose some elements such that $M[\sigma], N[\tau] \in SN$, $(M[\sigma] \;\; N[\tau]) \notin SN$ and

$(\eta(M)+\eta(N), cxty(M)+cxty(N))$ is minimal. By lemma 4.3, either $M[\sigma]\triangleright^* \mu\delta M_1$ and $M_1[\delta =_r N[\tau]] \notin SN$ or $N[\tau] \triangleright^* \mu\beta N_1$ and $N_1[\beta =_l M[\sigma]] \notin SN$. Look at the first case (the other one is similar). By lemma 4.2, $M \triangleright^* \mu\delta M_2$ for some M_2 such that $M_2[\sigma] \triangleright^* M_1$. Thus, $M_2[\sigma][\delta =_r N[\tau]] \notin SN$. By lemma 4.4 with M_2, σ and $N[\tau]$, let $M' \prec M_2$ and σ' be such that $M'[\sigma'] \in SN$, $(M'[\sigma']\ N[\tau]) \notin SN$. This contradicts the minimality of the chosen elements since $\eta c(M') < \eta c(M)$. □

4.2. The simply typed symmetric $\lambda\mu$-calculus is strongly normalizing

In this section, we consider the simply typed calculus with the $\beta\mu\mu'$-reduction i.e. $M \triangleright M'$ means M' is obtained from M by one step of the $\beta\mu\mu'$-reduction. The strong normalization of the $\beta\mu\mu'$-reduction is proved essentially as in theorem 3.3.

There is, however, a new difficulty : a β-substitution may create a μ, i.e. the fact that $M[x := N] \triangleright^* \mu\alpha P$ does not imply that $M \triangleright^* \mu\alpha Q$. Moreover the μ may come from a complicated interaction between M and N and, in particular, the alternation between M and N can be lost. Let e.g. $M = (M_1\ (x\ (\lambda y_1 \lambda y_2 \mu\alpha M_4)\ M_2\ M_3))$ and $N = \lambda z(z\ N_1)$. Then $M[x := N] \triangleright^* (M_1\ (\mu\alpha M_4'\ M_3)) \triangleright^* \mu\alpha M_4'[\alpha =_r M_3][\alpha =_l M_1]$. To deal with this situation, we need to consider some new kind of $\mu\mu'$-substitutions (see definition 4.2). Lemma 4.10 gives the different ways in which a μ may appear. The difficult case in the proof (when a μ is created and the control between M and N is lost) will be solved by using a typing argument.

To simplify the notations, we do not write explicitly the type information but, when needed, we denote by $type(M)$ the type of the term M.

Lemma 4.5. 1. If $(M\ N) \triangleright^* \lambda x P$, then $M \triangleright^* \lambda y M_1$ and $M_1[y := N] \triangleright^* \lambda x P$.

2. If $(M\ N) \triangleright^* \mu\alpha P$, then either ($M \triangleright^* \lambda y M_1$ and $M_1[y := N] \triangleright^* \mu\alpha P$) or ($M \triangleright^* \mu\alpha M_1$ and $M_1[\alpha =_r N] \triangleright^* P$) or ($N \triangleright^* \mu\alpha N_1$ and $N_1[\alpha =_l M] \triangleright^* P$).

Proof (1) is trivial. (2) is as in lemma 4.1. □

Lemma 4.6. Let $M \in SN$ and $\sigma = [x_1 := N_1, ..., x_k := N_k]$. Assume $M[\sigma] \triangleright^* \lambda y P$. Then, either $M \triangleright^* \lambda y P_1$ and $P_1[\sigma] \triangleright^* P$ or $M \triangleright^* (x_i\ \overrightarrow{Q})$ and $(N_i\ \overrightarrow{Q[\sigma]}) \triangleright^* \lambda y P$.

Proof By induction on $\eta c(M)$. The only non immediate case is $M = (R\ S)$. By lemma 4.5, there is a term R_1 such that $R[\sigma] \triangleright^* \lambda z R_1$ and $R_1[z := S[\sigma]] \triangleright^* \lambda y P$. By the induction hypothesis (since $\eta c(R) < \eta c(M)$), we have two cases to consider.

(1) $R \triangleright^* \lambda z R_2$ and $R_2[\sigma] \triangleright^* R_1$, then $R_2[z := S][\sigma] \triangleright^* \lambda y P$. By the induction hypothesis (since $\eta(R_2[z := S]) < \eta(M)$),

- either $R_2[z := S] \triangleright^* \lambda y P_1$ and $P_1[\sigma] \triangleright^* P$; but then $M \triangleright^* \lambda y P_1$ and we are done.

- or $R_2[z := S] \triangleright^* (x_i\ \overrightarrow{Q})$ and $(N_i\ \overrightarrow{Q[\sigma]}) \triangleright^* \lambda y P$, then $M \triangleright^* (x_i\ \overrightarrow{Q})$ and again we are done.

(2) $R \triangleright^* (x_i\ \overrightarrow{Q})$ and $(N_i\ \overrightarrow{Q[\sigma]}) \triangleright^* \lambda z R_1$. Then $M \triangleright^* (x_i\ \overrightarrow{Q}\ S)$ and the result is trivial. □

Definition 4.2.
- An address is a finite list of symbols in $\{l, r\}$. The empty list is denoted by $[]$ and, if a is an address and $s \in \{l, r\}$, $[s :: a]$ denotes the list obtained by putting s at the beginning of a.
- Let a be an address and M be a term. The sub-term of M at the address a (denoted as M_a) is defined recursively as follows : if $M = (P\ Q)$ and $a = [r :: b]$ (resp. $a = [l :: b]$) then $M_a = Q_b$ (resp. P_b) and undefined otherwise.
- Let M be a term and a be an address such that M_a is defined. Then $M\langle a = N\rangle$ is the term M where the sub-term M_a has been replaced by N.
- Let M, N be some terms and a be an address such that M_a is defined. Then $N[\alpha =_a M]$ is the term N in which each sub-term of the form $(\alpha\ U)$ is replaced by $(\alpha\ M\langle a = U\rangle)$.

Remark 4.1. - Let $N = \lambda x(\alpha\ \lambda y(x\ \mu\beta(\alpha\ y)))$, $M = (M_1\ (M_2\ M_3))$ and $a = [r :: l]$. Then $N[\alpha =_a M] = \lambda x(\alpha\ (M_1\ (\lambda y(x\ \mu\beta(\alpha\ (M_1\ (y\ M_3))))\ M_3)))$.

- Let $M = (P\ ((R\ (x\ T))\ Q))$ and $a = [r :: l :: r :: l]$. Then $N[\alpha =_a M] = N[\alpha =_r T][\alpha =_l R][\alpha =_r Q][\alpha =_r P]$.

- Note that the sub-terms of a term having an address in the sense given above are those for which the path to the root consists only on applications (taking either the left or right son).

- Note that $[\alpha =_{[l]} M]$ is not the same as $[\alpha =_l M]$ but $[\alpha =_l M]$ is the same as $[\alpha =_{[r]} (M\ N)]$ where N does not matter. More generally, the term $N[\alpha =_a M]$ does not depend of M_a.

- Note that $M\langle a = N\rangle$ can be written as $M'[x_a := N]$ where M' is the term M in which M_a has been replaced by the fresh variable x_a and thus (this will be used in the proof of lemma 4.12) if M_a is a variable x, $(\alpha\ U)[\alpha =_a M] = (\alpha\ M_1[y := U[\alpha =_a M]])$ where M_1 is the term M in which the particular occurrence of x at the address a has been replaced by the fresh name y and the other occurrences of x remain unchanged.

Lemma 4.7. Let M be a term and $\sigma = [\alpha_1 =_{a_1} N_1, ..., \alpha_n =_{a_n} N_n]$.

1. If $M[\sigma] \triangleright^* \lambda x P$, then $M \triangleright^* \lambda x Q$ and $Q[\sigma] \triangleright^* P$.
2. If $M[\sigma] \triangleright^* \mu\alpha P$, then $M \triangleright^* \mu\alpha Q$ and $Q[\sigma] \triangleright^* P$.

Proof By induction on M. Use lemma 4.5. □

Lemma 4.8. Assume $M, N \in SN$ and $(M\ \ N) \notin SN$. Then, either ($M \triangleright^* \lambda y P$ and $P[y := N] \notin SN$) or ($M \triangleright^* \mu\alpha P$ and $P[\alpha =_r N] \notin SN$) or ($N \triangleright^* \mu\alpha P$ and $P[\alpha =_l M] \notin SN$).

Proof By induction on $\eta(M) + \eta(N)$. □

Lemma 4.9. If $\Gamma \vdash M : A$ and $M \triangleright^* N$ then $\Gamma \vdash N : A$.

Proof Straightforward. □

Lemma 4.10. Let n be an integer, $M \in SN$, $\sigma = [x_1 := N_1, ..., x_k := N_k]$ where $lg(type(N_i)) = n$ for each i. Assume $M[\sigma] \triangleright^* \mu\alpha P$. Then,

1. either $M \triangleright^* \mu\alpha P_1$ and $P_1[\sigma] \triangleright^* P$
2. or $M \triangleright^* Q$ and, for some i, $N_i \triangleright^* \mu\alpha N_i'$ and $N_i'[\alpha =_a Q[\sigma]] \triangleright^* P$ for some address a in Q such that $Q_a = x_i$.
3. or $M \triangleright^* Q$, $Q_a[\sigma] \triangleright^* \mu\alpha N'$ and $N'[\alpha =_a Q[\sigma]] \triangleright^* P$ for some address a in Q such that $lg(type(Q_a)) < n$.

Proof By induction on $\eta c(M)$. The only non immediate case is $M = (R\ S)$. Since $M[\sigma] \triangleright^* \mu\alpha P$, the application $(R[\sigma]\ S[\sigma])$ must be reduced. Thus there are three cases to consider.

- It is reduced by a μ'-reduction, i.e. there is a term S_1 such that $S[\sigma] \triangleright^* \mu\alpha S_1$ and $S_1[\alpha =_l R[\sigma]] \triangleright^* P$. By the induction hypothesis:
  - either $S \triangleright^* \mu\alpha Q$ and $Q[\sigma] \triangleright^* S_1$, then $M \triangleright^* \mu\alpha Q[\alpha =_l R]$ and $Q[\alpha =_l R][\sigma] \triangleright^* P$.
  - or $S \triangleright^* Q$ and, for some i, $N_i \triangleright^* \mu\alpha N_i'$, $Q_a = x_i$ for some address a in Q and $N_i'[\alpha =_a Q[\sigma]] \triangleright^* S_1$. Then $M \triangleright^* (R\ Q) = Q'$ and letting $b = [r :: a]$ we have $N_i'[\alpha =_b Q'[\sigma]] \triangleright^* P$.
  - or $S \triangleright^* Q$, $Q_a[\sigma] \triangleright^* \mu\alpha N'$ for some address a in Q such that $lg(type(Q_a)) < n$ and $N'[\alpha =_a Q[\sigma]] \triangleright^* S_1$. Then $M \triangleright^* (R\ Q) = Q'$ and letting $b = [r :: a]$ we have $N'[\alpha =_b Q'[\sigma]] \triangleright^* P$ and $lg(type(Q_b')) < n$.
- It is reduced by a μ-reduction. This case is similar to the previous one.
- It is reduced by a β-reduction, i.e. there is a term U such that $R[\sigma] \triangleright^* \lambda y U$ and $U[y := S[\sigma]] \triangleright^* \mu\alpha P$. By lemma 4.6, there are two cases to consider.
  - either $R \triangleright^* \lambda y R_1$ and $R_1[\sigma][y := S[\sigma]] = R_1[y := S][\sigma] \triangleright^* \mu\alpha P$. The result follows from the induction hypothesis since $\eta(R_1[y := S]) < \eta(M)$.
  - or $R \triangleright^* (x_i\ \overrightarrow{R_1})$. Then $Q = (x_i\ \overrightarrow{R_1}\ S)$ and $a = []$ satisfy the desired conclusion since then $lg(type(M)) < n$. $\square$

Definition 4.3. Let A be a type. We denote by Σ_A the set of substitutions of the form $[\alpha_1 =_{a_1} M_1, ..., \alpha_n =_{a_n} M_n]$ where the type of the α_i is $\neg A$.

Remark 4.2. Remember that the type of α is not the same in N and in $N[\alpha =_a M]$. The previous definition may thus be considered as ambiguous. When we consider the term $N[\sigma]$ where $\sigma \in \Sigma_A$, we assume that N (and not $N[\sigma]$) is typed in the context where the α_i have type A. Also note that considering $N[\alpha =_a M]$ implies that the type of M_a is A.

Lemma 4.11. Let n be an integer and A be a type such that $lg(A) = n$. Let N, P be terms and $\tau \in \Sigma_A$. Assume that,

- for every $M, N \in SN$ such that $lg(type(N)) < n$, $M[x := N] \in SN$.
- $N[\tau] \in SN$ but $N[\tau][\delta =_a P] \notin SN$.

- δ is free and has type $\neg A$ in N but δ is not free in $Im(\tau)$.

Then, there is $N' \prec N$ and $\tau' \in \Sigma_A$ such that $N'[\tau'] \in SN$ and $P\langle a = N'[\tau']\rangle \notin SN$.

Proof The proof looks like the one of lemma 4.4. Denote by (H) the first assumption i.e. for every $M, N \in SN$ such that $lg(type(N)) < n$, $M[x := N] \in SN$.

Let $\tau = [\alpha_1 =_{a_1} M_1, ..., \alpha_n =_{a_n} M_n]$, $\mathcal{U} = \{U \;/\; U \preceq N\}$ and $\mathcal{V} = \{V \;/\; V \preceq M_i$ for some $i\}$. Define inductively the sets Σ_m and Σ_n of substitutions by the following rules:

$\rho \in \Sigma_n$ iff $\rho = \emptyset$ or $\rho = \rho' + [\alpha =_a V[\sigma]]$ for some $V \in \mathcal{V}$, $\sigma \in \Sigma_m$, $\rho' \in \Sigma_n$ and α has type $\neg A$.

$\sigma \in \Sigma_m$ iff $\sigma = \emptyset$ or $\sigma = \sigma' + [x := U[\rho]]$ for some $U \in \mathcal{U}$, $\rho \in \Sigma_n$, $\sigma' \in \Sigma_m$ and x has type A.

Denote by C the conclusion of the lemma. We prove something more general.

(1) Let $U \in \mathcal{U}$ and $\rho \in \Sigma_n$. Assume $U[\rho] \in SN$ and $U[\rho][\delta =_a P] \notin SN$. Then, C holds.

(2) Let $V \in \mathcal{V}$ and $\sigma \in \Sigma_m$. Assume $V[\sigma] \in SN$ and $V[\sigma][\delta =_a P] \notin SN$. Then, C holds.

Note that the definitions of the sets Σ_n and Σ_m are not the same as the ones of lemma 4.4. We gather here in Σ_n all the $\mu\mu'$-substitutions getting thus the new substitutions of definition 4.2 and we put in Σ_m only the λ-substitutions.

The conclusion C follows from (1) with N and τ. The properties (1) and (2) are proved by a simultaneous induction on $\eta c(U[\rho])$ (for the first case) and $\eta c(V[\tau])$ (for the second case).

The proof is by case analysis as in lemma 4.4. We only consider the new case for $V[\sigma]$, i.e. when $V = (V_1\ V_2)$ and $V_i[\sigma][\delta =_a P] \in SN$. The other ones are done essentially in the same way as in lemma 4.4.

- Assume first the interaction between V_1 and V_2 is a β-reduction. If $V_1 \triangleright^* \lambda x V_1'$, the result follows from the induction hypothesis with $V_1'[x := V_2][\sigma]$. Otherwise, by lemma 4.6, $V_1 \triangleright^* (x\ \overrightarrow{W})$. Let $\sigma(x) = U[\rho]$. Then $(U[\rho]\ \overrightarrow{W}[\sigma]) \triangleright^* \lambda y Q$ and $Q[y := V_2[\sigma]][\delta =_a P] \notin SN$. But, since the type of x is A, the type of y is less than A and since $Q[\delta =_a P]$ and $V_2[\sigma][\delta =_a P]$ are in SN this contradicts (H).

- Assume next the interaction between V_1 and V_2 is a μ or μ'-reduction. We consider only the case μ (the other one is similar). If $V_1 \triangleright^* \mu\alpha V_1'$, the result follows from the induction hypothesis with $V_1'[\alpha =_r V_2][\sigma]$. Otherwise, by lemma 4.10, there are two cases to consider.

- $V_1 \triangleright^* Q$, $Q_c = x$ for some address c in Q and $x \in dom(\sigma)$, $\sigma(x) = U[\rho]$, $U[\rho] \triangleright^* \mu\alpha U_1$ and $U_1[\alpha =_c Q[\sigma]][\alpha =_r V_2[\sigma]][\delta =_a P] \notin SN$. By lemma 4.7, we have $U \triangleright^* \mu\alpha U_2$ and $U_2[\rho] \triangleright^* U_1$, then $U_2[\rho][\alpha =_c Q[\sigma]][\alpha =_r V_2[\sigma]][\delta =_a P] \notin SN$. Let $V' = (Q\ V_2)$ and $b = l :: c$. The result follows then from the induction hypothesis with $U_2[\rho']$ where $\rho' = \rho + [\alpha =_b V'[\sigma]]$.

- $V_1 \triangleright^* Q$, $Q_c[\sigma][\delta =_a P] \triangleright^* \mu\alpha R$ for some address c in Q such that $lg(type(Q_c)) < n$, $R[\alpha =_c Q[\sigma][\delta =_a P]][\alpha =_r V_2[\sigma][\delta =_a P]] \notin SN$. Let $V' = (Q'\ V_2)$ where Q' is the same as Q but Q_c has been replaced by a fresh variable y and $b = l :: c$. Then $R[\alpha =_b V'[\sigma][\delta =_a P]] \notin SN$. Let R' be such that $R' \prec R$, $R'[\alpha =_b V'[\sigma][\delta =_a P]] \notin SN$ and $\eta c(R')$ is minimal. It is easy to check that $R' = (\alpha\ R'')$, $R''[\alpha =_b V'[\sigma][\delta =_a P]] \in SN$ and $V'[\sigma'][\delta =_a P] \notin SN$ where $\sigma' = \sigma +$

$[y := R''[\alpha =_b V'[\sigma]]]$. If $V'[\sigma][\delta =_a P] \notin SN$, we get the result by the induction hypothesis since $\eta c(V'[\sigma]) < \eta c(V[\sigma])$. Otherwise this contradicts the assumption (H) since $V'[\sigma][\delta =_a P], R''[\alpha =_b V'[\sigma][\delta =_a P]] \in SN$, $V'[\sigma][\delta =_a P][y := R''[\alpha =_b V'[\sigma][\delta =_a P]]] \notin SN$ and the type of y is less than n. $\square$

Lemma 4.12. If $M, N \in SN$, then $M[x := N] \in SN$.

Proof We prove something a bit more general: let A be a type, $M, N_1, ..., N_k$ be terms and $\tau_1, ..., \tau_k$ be substitutions in Σ_A. Assume that, for each i, N_i has type A and $N_i[\tau_i] \in SN$. Then $M[x_1 := N_1[\tau_1],\ ...,\ x_k := N_k[\tau_k]] \in SN$. This is proved by induction on $(lg(A), \eta(M), cxty(M), \Sigma\ \eta(N_i), \Sigma\ cxty(N_i))$ where, in $\Sigma\ \eta(N_i)$ and $\Sigma\ cxty(N_i)$, we count each occurrence of the substituted variable. For example if $k = 1$ and x_1 has n occurrences, $\Sigma\ \eta(N_i) = n.\eta(N_1)$.

If M is $\lambda y M_1$ or $(\alpha\ M_1)$ or $\mu\alpha M_1$ or a variable, the result is trivial. Assume then that $M = (M_1\ M_2)$. Let $\sigma = [x_1 := N_1[\tau_1],\ ...,\ x_k := N_k[\tau_k]]$. By the induction hypothesis, $M_1[\sigma], M_2[\sigma] \in SN$. By lemma 4.8 there are 3 cases to consider.

- $M_1[\sigma] \triangleright^* \lambda y P$ and $P[y := M_2[\sigma]] \notin SN$. By lemma 4.6, there are two cases to consider.

  - $M_1 \triangleright^* \lambda y Q$ and $Q[\sigma] \triangleright^* P$. Then $Q[y := M_2][\sigma] = Q[\sigma][y := M_2[\sigma]] \triangleright^* P[y := M_2[\sigma]]$ and, since $\eta(Q[y := M_2]) < \eta(M)$, this contradicts the induction hypothesis.
  - $M_1 \triangleright^* (x_i\ \overrightarrow{Q})$ and $(N_i\ \overrightarrow{Q[\sigma]}) \triangleright^* \lambda y P$. Then, since the type of N_i is A, $lg(type(y)) < lg(A)$. But $P, M_2[\sigma] \in SN$ and $P[y := M_2[\sigma]] \notin SN$. This contradicts the induction hypothesis.

- $M_1[\sigma] \triangleright^* \mu\alpha P$ and $P[\alpha =_r M_2[\sigma]] \notin SN$. By lemma 4.10, there are three cases to consider.

  - $M_1 \triangleright^* \mu\alpha Q$ and $Q[\sigma] \triangleright^* P$. Then, $Q[\alpha =_r M_2][\sigma] = Q[\sigma][\alpha =_r M_2[\sigma]] \triangleright^* P[\alpha =_r M_2[\sigma]]$ and, since $\eta(Q[\alpha =_r M_2]) < \eta(M)$, this contradicts the induction hypothesis.
  - $M_1 \triangleright^* Q$, $N_i[\tau_i] \triangleright^* \mu\alpha L'$ and $Q_a = x_i$ for some address a in Q such that $L'[\alpha =_a Q[\sigma]] \triangleright^* P$ and thus $L'[\alpha =_b M'[\sigma]] \notin SN$ where $b = (l :: a)$ and $M' = (Q\ M_2)$.

    By lemma 4.2, $N_i \triangleright^* \mu\alpha L$ and $L[\tau_i] \triangleright^* L'$. Thus, $L[\tau_i][\alpha =_b M'[\sigma]] \notin SN$. By lemma 4.11, there is $L_1 \prec L$ and τ' such that $L_1[\tau'] \in SN$ and $M'[\sigma]\langle b = L_1[\tau']\rangle \notin SN$. Let M'' be M' where the variable x_i at the address b has been replaced by the fresh variable y and let $\sigma_1 = \sigma + [y := L_1[\tau']]$. Then $M''[\sigma_1] = M'[\sigma]\langle b = L_1[\tau']\rangle \notin SN$.

    If $M_1 \triangleright^+ Q$ we get a contradiction from the induction hypothesis since $\eta(M'') < \eta(M)$. Otherwise, M'' is the same as M up to the change of name of a variable and σ_1 differs from σ only at the address b. At this address, x_i was substituted in σ by $N_i[\tau_i]$ and in σ_1 by $L_1[\tau']$ but $\eta c(L_1) < \eta c(N_i)$ and thus we get a contradiction from the induction hypothesis.

– $M \triangleright^* Q$, $Q_a[\sigma] \triangleright^* \mu\alpha L$ for some address a in Q such that $lg(type(Q_a)) < lg(A)$ and $L[\alpha =_a Q[\sigma]] \triangleright^* P$. Then, $L[\alpha =_b M'[\sigma]] \notin SN$ where $b = [l :: a]$ and $M' = (Q\ M_2)$.

By lemma 4.11, there is an L' and τ' such that $L'[\tau'] \in SN$ and $M'[\sigma]\langle b = L'[\tau']\rangle \notin SN$. Let M'' be M' where the variable x_i at the address b has been replaced by the fresh variable y. Then $M''[\sigma][y := L'[\tau']] = M'[\sigma]\langle b = L'[\tau']\rangle \notin SN$.

But $\eta(M'') \leq \eta(M)$ and $cxty(M'') < cxty(M)$ since, because of its type, Q_a cannot be a variable and thus, by the induction hypothesis, $M''[\sigma] \in SN$. Since $M''[\sigma][y := L'[\tau']] \notin SN$ and $lg(type(L')) < lg(A)$, this contradicts the induction hypothesis.

- $M_2[\sigma] \triangleright^* \mu\alpha P$ and $P[\alpha =_l M_1[\sigma]] \notin SN$. This case is similar to the previous one. □

Theorem 4.2. Every typed term is in SN.

Proof By induction on the term. It is enough to show that if $M, N \in SN$, then $(M\ N) \in SN$. Since $(M\ N) = (x\ y)[x := M][y := N]$ where x, y are fresh variables, the result follows by applying theorem 4.12 twice and the induction hypothesis. □

5. Remarks and future work

5.1. Why the usual candidates do not work ?

In [26], the proof of the strong normalization of the $\lambda\mu$-calculus is done by using the *usual* (i.e. defined without a fix-point operation) candidates of reducibility. This proof could be easily extended to the symmetric $\lambda\mu$-calculus if we knew the following properties for the un-typed calculus:

1. If N and $(M[x := N]\ \overrightarrow{P})$ are in SN, then so is $(\lambda x M\ N\ \overrightarrow{P})$.
2. If N and $(M[\alpha =_r N]\ \overrightarrow{P})$ are in SN, then so is $(\mu\alpha M\ N\ \overrightarrow{P})$.
3. If $\overrightarrow{P}$ are in SN, then so is $(x\ \overrightarrow{P})$.

These properties are easy to show for the $\beta\mu$-reduction but they were not known for the $\beta\mu\mu'$-reduction.

The third property is true but the properties (1) and (2) are false. The proof of (3) and the counter-examples for (1) and (2) can be found in [10].

5.2. Future work

We believe that our technique, will allow to give explicit bounds for the length of the reductions of a typed term. This is a goal we will try to manage.

References

[1] F. Barbanera and S. Berardi. *A symmetric lambda-calculus for classical program extraction*. In M. Hagiya and J.C. Mitchell, editors, Proceedings of theoretical aspects of computer software, TACS'94. LNCS (789), pp. 495-515. Springer Verlag, 1994.

[2] P. Battyanyi. *Normalization results for the symmetric $\lambda\mu$- calculus.* Private communication. To appear in his PhD thesis.

[3] R. Constable and C. Murthy. *Finding computational content in classical proofs.* In G. Huet and G. Plotkin, editors, Logical Frameworks, pp. 341-362, Cambridge University Press, 1991.

[4] P.L. Curien and H. Herbelin. *The Duality of Computation*. Proc. International Conference on Functional Programming, September 2000, Montral, IEEE, 2000.

[5] P. de Groote. *A CPS-translation of the lambda-mu-calculus*. In S. Tison, editor, 19th International Colloquium on Trees in Algebra and Programming, CAAP'94, volume 787 of Lecture Notes in Computer Science, pp 85-99. Springer, 1994.

[6] P. de Groote. *A simple calculus of exception handling*. In M. Dezani and G. Plotkin, editors, Second International Conference on Typed Lambda Calculi and Applications, TLCA'95, volume 902 of Lecture Notes in Computer Science, pp. 201-215. Springer, 1995.

[7] R. David. *Normalization without reducibility*. Annals of Pure and Applied Logic (107), pp. 121-130, 2001.

[8] R. David and K. Nour. *A short proof of the strong normalization of the simply typed $\lambda\mu$-calculus*. Schedae Informaticae n12, pp. 27-34, 2003.

[9] R. David and K. Nour. *A short proof of the strong normalization of classical natural deduction with disjunction*. The Journal of Symbolic Logic n 68.4, pp. 1277 - 1288, 2003.

[10] R. David and K. Nour. *Why the usual candidates of reducibility do not work for the symetric $\lambda\mu$-calculus*. Electronic Notes in Computer Science vol 140, pp 101-111, 2005.

[11] R. David and K. Nour. *Arithmetical proofs of some strong normalization results for the symmetric $\lambda\mu$-calculus*. TLCA'05, LNCS 3461, pp. 162-178, 2005.

[12] D. Dougherty, S.Ghilezan, P.Lescanne, S. Likavec. *Strong normalization of the classical dual sequent calculus*. LPAR'05 LNCS 3835 p 169-183.

[13] M. Felleisen, D.P. Friedman, E.E. Kohlbecker and B.F. Duba. *A Syntactic Theory of Sequential Control.* Theoretical Computer Science 52, pp. 205-237, 1987.

[14] J.-Y. Girard. *A new constructive logic: classical logic.* MSCS (1), pp. 255-296, 1991.

[15] T.G. Griffin. *A formulae-as-types notion of control.* POPL'90 pp 47-58.

[16] F. Joachimski and R. Matthes. *Short proofs of normalization for the simply-types λ-calculus, permutative conversions and Godel's T.* Archive for Mathematical Logic 42, pp. 59-87, 2003.

[17] J.-L. Krivine. *Classical logic, storage operators and 2nd order lambda-calculus.* Annals of Pure and Applied Logic (68), pp. 53-78, 1994.

[18] C.R. Murthy. *An evaluation semantics for classical proofs.* In Proceedings of the sixth annual IEEE symposium on logic in computer science, pp. 96-107, 1991.

[19] K. Nour. *La valeur d'un entier classique en $\lambda\mu$-calcul.* Archive for Mathematical Logic (36), pp. 461-471, 1997.

[20] K. Nour. *A non-deterministic classical logic (the $\lambda\mu^{++}$-calculus).* Mathematical Logic Quarterly (48), pp. 357 - 366, 2002.

[21] K. Nour and K. Saber. *A semantical proof of the strong normalization theorem of full propositionnal classical natural deduction*. Archive for Mathematical Logic vol 45, pp 357-364, 2006.

[22] M. Parigot. *Free Deduction: An Analysis of "Computations" in Classical Logic*. Proceedings. Lecture Notes in Computer Science, Vol. 592, Springer, pp. 361-380, 1992.

[23] M. Parigot. *$\lambda\mu$-calculus: An algorithm interpretation of classical natural deduction.* Lecture Notes in Artificial Intelligence (624), pp. 190-201. Springer Verlag, 1992.

[24] M. Parigot. *Strong normalization for second order classical natural deduction*. In Proceedings, Eighth Annual IEEE Symposium on Logic in Computer Science, pp. 39-46, Montreal, Canada, 19–23 June 1993. IEEE Computer Society Press.

[25] M. Parigot. *Classical proofs as programs.* In G. Gottlob, A. Leitsch, and D. Mundici, eds., Proc. of 3rd Kurt Godel Colloquium, KGC'93, vol. 713 of Lecture Notes in Computer Science, pp. 263-276. Springer-Verlag, 1993.

[26] M. Parigot. *Proofs of strong normalization for second order classical natural deduction.* Journal of Symbolic Logic, 62 (4), pp. 1461-1479, 1997.

[27] E. Polonovsky. *Substitutions explicites, logique et normalisation*. PhD thesis. Paris 7, 2004.

[28] W. Py. *Confluence en $\lambda\mu$-calcul.* PhD thesis. University of Chambéry, 1998.

[29] N.J. Rehof and M.H. Sorensen. *The λ_Δ-calculus.* In M. Hagiya and J.C. Mitchell, editors, Proceedings of the international symposium on theoretical aspects of computer software, TACS'94, LNCS (789), pp. 516-542. Springer Verlag, 1994.

[30] P. Wadler. *Call-by-value is dual to Call-by-name*. International Conference on Functional Programming. Uppsala, August 2003.

[31] P. Wadler. *Call-by-value is dual to Call-by-name. Re-loaded*. RTA'05, LNCS 3467, pp. 185-203, 2005

[32] Y. Yamagata. *Strong Normalization of Second Order Symmetric Lambda-mu Calculus.* TACS 2001, Lecture Notes in Computer Science 2215, pp. 459-467, 2001.