Strong Normalization of $\overline{\lambda}\mu\tilde{\mu}$-Calculus with Explicit Substitutions

Emmanuel Polonovski

PPS, CNRS – Université Paris 7
Emmanuel.Polonovski@pps.jussieu.fr

Abstract. The $\overline{\lambda}\mu\tilde{\mu}$-calculus, defined by Curien and Herbelin [7], is a variant of the $\lambda\mu$-calculus that exhibits symmetries such as term/context and call-by-name/call-by-value. Since it is a symmetric, and hence a non-deterministic calculus, usual proof techniques of normalization needs some adjustments to be made to work in this setting. Here we prove the strong normalization (SN) of simply typed $\overline{\lambda}\mu\tilde{\mu}$-calculus with explicit substitutions. For that purpose, we first prove SN of simply typed $\overline{\lambda}\mu\tilde{\mu}$-calculus (by a variant of the reducibility technique from Barbanera and Berardi [2]), then we formalize a proof technique of SN via PSN (preservation of strong normalization), and we prove PSN by the perpetuality technique, as formalized by Bonelli [5].

1 Introduction

1.1 $\overline{\lambda}\mu\tilde{\mu}$-Calculus and Explicit Substitutions

The $\overline{\lambda}\mu\tilde{\mu}$-calculus, defined by Curien and Herbelin [7], is a symmetric variant of Parigot's $\lambda\mu$-calculus [11] that provides a term notation for classical sequent calculus. It exhibits symmetries such as terms/contexts and call-by-name/call-by-value. Its two main reduction rules form a symmetric critical pair, which makes the calculus non-deterministic (non-confluent) and raises difficulties in normalization proofs : a naive definition of reducibility candidates would fall in a symmetric loop of mutual induction.

On the other hand, calculi with explicit substitutions were introduced [1] as a bridge between λ-calculus [6] and concrete implementations of functionnal programming languages. Those calculi intend to refine the evaluation process by proposing reduction rules to deal with the substitution mechanism – a *meta*-operation in the traditionnal λ-calculus. In the study of those calculi, an important task was to establish good properties such as:

- Simulation of β reduction, which says that a term that can be reduced to another in the traditionnal λ-calculus can also be reduced to the same one in the calculus with explicit substitutions.
- Confluence, which says that whatever reduction strategy you choose, you can always find a common reduct.

- Preservation of strong normalization (PSN), which says that if a term is β-strongly normalizing (*i.e.* cannot be infinitely reduced), it is also strongly normalizing with respect to the calculus with explicit substitutions.
- Strong normalization (SN), which says that, with respect to a typing system, every typed term is strongly normalizing in the calculus with explicit substitutions.

It was remarked, at once, that explicit substitutions raises more difficulties in normalization proofs, due to the fact that reductions can now take place in an argument substituted in a term to a variable which is not free in that term. Such reductions produce no trace in the original calculus, because the substitution is bounded to disappear. Therefore we cannot easily infer SN for explicit substitutions from strong normalization of the original calculus.

1.2 The $\overline{\lambda}\mu\tilde{\mu}$-Calculus with Explicit Substitutions: $\overline{\lambda}\mu\tilde{\mu}\mathrm{x}$

Here we work on $\overline{\lambda}\mu\tilde{\mu}\mathtt{x}$, an explicit substitutions version "à la" $\lambda\mathtt{x}$ [4] of the $\overline{\lambda}\mu\tilde{\mu}$-calculus. Its syntax was introduced in [9] and, in the same paper, there was an attempt to prove strong normalization of the deterministic call-by-name fragment directly by the reducibility technique. Unfortunately, the technique did not work so nicely, and the proof of a key lemma (Weakening lemma) turned out to be bugged... We keep this technique for the pure calculus (*i.e.* without explicit substitutions), and, in order to lift it to the *symmetric* calculus, we adjust it like Barbanera and Berardi did for their symmetric λ-calculus [2]. We will see that reducibility sets constructed by fixed point ensure that their definition will not fall in the symmetric infinite loop of terms defined by contexts and *vice versa*.

To prove SN, we formalize a technique initially suggested by Herbelin, which consists in expanding substitutions into pure $\overline{\lambda}\mu\tilde{\mu}$-redexes and to inherit SN of the whole calculus by SN of the pure calculus and by PSN.

Finally, to prove PSN, we use the perpetuality technique, as formalized by Bonelli [5]. The main point of this technique is to exhibit a strategy wich preserves infinite reductions. This together with some material to trace the substitutions backwards, allows us to establish PSN by contradiction.

In the sequel, we will note $\mathcal{SN}_R$ for the set of strongly normalizing terms in the calculus R. We will use $FV(t)$ to denote the set of free variables of t, defined in the usual way.

1.3 Organization

We first present the (simply typed) $\overline{\lambda}\mu\tilde{\mu}$-calculus and we prove SN by the reducibility technique (section 2). In section 3, we use the perpetuality technique to establish PSN. Section 4 formalizes the proof technique of SN *via* PSN, and gives the material to use it for $\overline{\lambda}\mu\tilde{\mu}\mathtt{x}$. Finally, we give the proof of SN of $\overline{\lambda}\mu\tilde{\mu}\mathtt{x}$ in section 4.3.

2 The $\overline{\lambda}\mu\tilde{\mu}$-Calculus and Its Strong Normalization

We first recall the definition of the $\overline{\lambda}\mu\tilde{\mu}$-calculus, then we define reducibility sets and finally we establish strong normalization of the pure calculus.

2.1 Definition

There are three syntactic categories: terms, contexts and commands, respectively noted v, e and c. We take two variable sets: Var is the set of term variables, noted x, y, z etc. ; $Var^{\perp}$ is the set of context variables, noted α, β, etc. We will note t an object, i.e. one of v, e or c. The syntax of the $\overline{\lambda}\mu\tilde{\mu}$-calculus is:

$$\begin{array}{l} c ::= \langle v|e\rangle \\ v ::= x \mid \lambda x.v \mid e \cdot v \mid \mu\alpha.c \\ e ::= \alpha \mid \alpha\lambda.e \mid v \cdot e \mid \tilde{\mu}x.c \end{array}$$

Reduction rules are given below. The rules (μ) and $(\tilde{\mu})$ form a critical pair:

$$\begin{array}{lll} (\beta) & \langle \lambda x.v|v' \cdot e\rangle \to \langle v'|\tilde{\mu}x.\langle v|e\rangle\rangle & \\ (\tilde{\beta}) & \langle e' \cdot v|\alpha\lambda.e\rangle \to \langle \mu\alpha.\langle v|e\rangle|e'\rangle & \\ (\mu) & \langle \mu\alpha.c|e\rangle \to c[e/\alpha] & \\ (\tilde{\mu}) & \langle v|\tilde{\mu}x.c\rangle \to c[v/x] & \\ & & \\ (sv) & \mu\alpha.\langle v|\alpha\rangle \to v & \text{if } \alpha \notin FV(v) \\ (se) & \tilde{\mu}x.\langle x|e\rangle \to e & \text{if } x \notin FV(e) \end{array}$$

Types are usual simple types plus the *minus* type $A - B$ which is the symmetric counterpart of the *arrow* type $A \to B$, its meaning is A and not B. We work here in classical sequent calculus, with a notation to exhibit a formula in a sequent: $\Gamma \vdash A|\Delta$ is the same sequent as $\Gamma \vdash A, \Delta$ but the formula A is exhibited as *active formula*. For further details about this framework and the isomorphism with objects of the $\overline{\lambda}\mu\tilde{\mu}$-calculus, see [7].

Three sequent forms are used to type the syntactic categories: the commands are typed by $(\Gamma \vdash \Delta)$, the terms by $\Gamma \vdash A|\Delta$ and the contexts by $\Gamma|A \vdash \Delta$. Here are the typing rules:

$$\frac{c : (\Gamma, x : A \vdash \Delta)}{\Gamma|\tilde{\mu}x.c : A \vdash \Delta} \qquad \frac{\Gamma \vdash v : A|\Delta \quad \Gamma|e : A \vdash \Delta}{\langle v|e\rangle : (\Gamma \vdash \Delta)} \qquad \frac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha.c : A|\Delta}$$

$$\frac{}{\Gamma|\alpha : A \vdash \Delta, \alpha : A} \qquad \frac{}{\Gamma, x : A \vdash \Delta|x : A}$$

$$\frac{\Gamma|e : B \vdash \alpha : A, \Delta}{\Gamma|\alpha\lambda.e : A - B \vdash \Delta} \qquad \frac{\Gamma, x : A \vdash v : B|\Delta}{\Gamma \vdash \lambda x.v : A \to B|\Delta}$$

$$\frac{\Gamma \vdash v : A|\Delta \quad \Gamma|e : B \vdash \Delta}{\Gamma|v \cdot e : A \to B \vdash \Delta} \qquad \frac{\Gamma \vdash v : B|\Delta \quad \Gamma|e : A \vdash \Delta}{\Gamma \vdash e \cdot v : A - B|\Delta}$$

2.2 Reducibility Sets

We simultaneously define, by induction of the type structure:

– the operators:

$$\begin{array}{ll} \mathit{Lambda}(X_1, X_2) =_{Def} \{\lambda x.v \mid \forall v' \in X_1, e \in X_2 \ \langle v[v'/x] | e\rangle \in [\![\vdash]\!]\} \\ \mathit{Cons}(X_1, X_2) \quad =_{Def} \{v \cdot e \mid v \in X_1 \text{ and } e \in X_2\} \\ \\ \widetilde{\mathit{Lambda}}(X_1, X_2) =_{Def} \{\alpha\lambda.e \mid \forall e' \in X_1, v \in X_2 \ \langle v | e[e'/\alpha]\rangle \in [\![\vdash]\!]\} \\ \widetilde{\mathit{Cons}}(X_1, X_2) \quad =_{Def} \{e \cdot v \mid e \in X_1 \text{ and } v \in X_2\} \\ \\ \mathit{Mu}(X) \qquad\quad =_{Def} \{\mu\alpha.c \mid \forall e \in X \ \ c[e/\alpha] \in [\![\vdash]\!]\} \\ \widetilde{\mathit{Mu}}(X) \qquad\quad =_{Def} \{\tilde{\mu}x.c \mid \forall v \in X \ \ c[v/x] \in [\![\vdash]\!]\} \end{array}$$

Remark 1. Mu and $\widetilde{\mathit{Mu}}$ are decreasing operators: the greater X is, the lesser one can find $\mu\alpha.c$'s (resp. $\tilde{\mu}x.c$'s) that normalize against all e in X.

Then

- if A is atomic

$$\begin{array}{l} \mathit{Neg}_{[\![\vdash A]\!]}(Y) = \mathit{Var} \cup \mathit{Mu}(Y) \\ \mathit{Neg}_{[\![A\vdash]\!]}(X) = \mathit{Var}^{\perp} \cup \widetilde{\mathit{Mu}}(X) \end{array}$$

- if $A = A_1 \to A_2$

$$\begin{array}{l} \mathit{Neg}_{[\![\vdash A]\!]}(Y) = \mathit{Var} \cup \mathit{Mu}(Y) \cup \mathit{Lambda}([\![\vdash A_1]\!], [\![A_2 \vdash]\!]) \\ \mathit{Neg}_{[\![A\vdash]\!]}(X) = \mathit{Var}^{\perp} \cup \widetilde{\mathit{Mu}}(X) \cup \mathit{Cons}([\![\vdash A_1]\!], [\![A_2 \vdash]\!]) \end{array}$$

- if $A = A_1 - A_2$

$$\begin{array}{l} \mathit{Neg}_{[\![\vdash A]\!]}(Y) = \mathit{Var} \cup \mathit{Mu}(Y) \cup \widetilde{\mathit{Cons}}([\![A_1 \vdash]\!], [\![\vdash A_2]\!]) \\ \mathit{Neg}_{[\![A\vdash]\!]}(X) = \mathit{Var}^{\perp} \cup \widetilde{\mathit{Mu}}(X) \cup \widetilde{\mathit{Lambda}}([\![A_1 \vdash]\!], [\![\vdash A_2]\!]) \end{array}$$

Since Mu and $\widetilde{\mathit{Mu}}$ are decreasing operators, Neg is also a decreasing operator. So $\mathit{Neg}_{[\![\vdash A]\!]} \circ \mathit{Neg}_{[\![A\vdash]\!]}$ is an increasing operator, and by Tarski's theorem it has a fixed point X_0 ;

– the reducibility sets:

$$[\![\vdash]\!] = \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$$

and

$$[\![\vdash A]\!] = X_0 \quad \text{and} \quad [\![A \vdash]\!] = \mathit{Neg}_{[\![A\vdash]\!]}(X_0).$$

Proposition 1 (Good definition). *The reducibility sets defined above satisfies*

$$\begin{array}{lll}
\textit{(i)} & Var \subset [\![\vdash A]\!] & \\
\textit{(ii)} & Var^{\perp} \subset [\![A \vdash]\!] & \\
\textit{(iii)} & v \in [\![\vdash A]\!] \iff & \textit{either}\ v = x \\
 & & \quad \textit{or}\ v = e \cdot v' \ \textit{with}\ A = A_1 - A_2, \\
 & & \qquad e \in [\![A_1 \vdash]\!] \ \textit{and}\ v' \in [\![\vdash A_2]\!] \\
 & & \quad \textit{or}\ v = \mu\alpha.c \ \textit{and} \\
 & & \qquad \forall e \in [\![A \vdash]\!] \ \ c[e/\alpha] \in [\![\vdash]\!] \\
 & & \quad \textit{or}\ v = \lambda x.v' \ \textit{with}\ A = A_1 \to A_2 \ \textit{and} \\
 & & \qquad \forall v'' \in [\![\vdash A_1]\!], e \in [\![A_2 \vdash]\!] \ \ \langle v'[v''/x] | e\rangle \in [\![\vdash]\!] \\
\textit{(iv)} & e \in [\![A \vdash]\!] \iff & \textit{either}\ e = \alpha \\
 & & \quad \textit{or}\ e = v \cdot e' \ \textit{with}\ A = A_1 \to A_2, \\
 & & \qquad v \in [\![\vdash A_1]\!] \ \textit{and}\ e' \in [\![A_2 \vdash]\!] \\
 & & \quad \textit{or}\ e = \tilde{\mu}x.c \ \textit{and} \\
 & & \qquad \forall v \in [\![\vdash A]\!] \ \ c[v/x] \in [\![\vdash]\!] \\
 & & \quad \textit{or}\ e = \alpha\lambda.e' \ \textit{with}\ A = A_1 - A_2 \ \textit{and} \\
 & & \qquad \forall e'' \in [\![A_1 \vdash]\!], v \in [\![\vdash A_2]\!] \ \ \langle v | e'[e''/\alpha]\rangle \in [\![\vdash]\!]
\end{array}$$

Proof. From the definition of the reducibility sets, we have $[\![\vdash]\!] = \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ and the points (i) and (ii). We prove the points (iii) and (iv). Due to the symmetry, it suffices to prove (iii).

$$v \in [\![\vdash A]\!] \iff v \in Neg_{[\![\vdash A]\!]} \circ Neg_{[\![A\vdash]\!]}([\![\vdash A]\!]).$$

We then consider the different shapes of A and we inline the corresponding definition of $Neg_{[\![\vdash A]\!]} \circ Neg_{[\![A\vdash]\!]}([\![\vdash A]\!])$.

2.3 Strong Normalization

Here are the two traditionnal lemmas of strong normalization of the reducibility sets (RS) and closure by reduction.

Lemma 1 (SN of RS). *Let A be a type. Then $[\![\vdash A]\!] \subset \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ (1), $[\![A \vdash]\!] \subset \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ (2) and $[\![\vdash]\!] \subset \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ (3).*

Proof. By induction on the structure of A.

1. We consider the different forms of $v \in [\![\vdash A]\!]$:
   - $v = x$: then $v \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$.
   - $v = e \cdot v'$: then $A = A_1 - A_2$ and we conclude by using the induction hypothesis twice.
   - $v = \mu\alpha.c$: by the point (ii) of proposition 1, $\alpha \in [\![A \vdash]\!]$, then, by the point (iii) of proposition 1, $c[\alpha/\alpha] \in [\![\vdash]\!]$, that gives us $c \in [\![\vdash]\!] (= \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}})$. We then have $\mu\alpha.c \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$.
   - $v = \lambda x.v'$, then $A = A_1 \to A_2$: to get $v \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$, we need $v' \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$. By reducibility of $\lambda x.v'$, we have $\forall v'' \in [\![\vdash A_1]\!], e \in [\![A_2 \vdash]\!] \ \ \langle v'[v''/x] | e\rangle \in [\![\vdash]\!] (= \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}})$. By the points (i) and (ii) of proposition 1, we can take x for v'' and α for e, and that gives us $\langle v'[x/x] | \alpha\rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$. We deduce $v' \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ and conclude.

2. The proof for e is similar to the proof for v by symmetry.
3. By definition $[\![\vdash]\!] = \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$.

Lemma 2 (Closure by reduction).

1. $v \in [\![\vdash A]\!],\ v \to v' \implies v' \in [\![\vdash A]\!]$.
2. $e \in [\![A \vdash]\!],\ e \to e' \implies e' \in [\![A \vdash]\!]$.
3. $c \in [\![\vdash]\!],\ c \to c' \implies c' \in [\![\vdash]\!]$.

Proof. By induction on A, considering the different shapes of v, e, and c.

1.1. $v = x$: then no more reduction can occur.
1.2. $v = e_1 \cdot v_1$: we must consider two possible reductions $e_1 \cdot v_1 \to e_2 \cdot v_1$ or $e_1 \cdot v_1 \to e_1 \cdot v_2$. In either case, we conclude by induction hypothesis.
1.3. $v = \mu\alpha.c$: we consider the following two cases.
   – The reduction is $\mu\alpha.c \to \mu\alpha.c'$. By definition of $\mu\alpha.c \in [\![\vdash A]\!]$ we have $\forall e \in [\![A \vdash]\!]\ \ c[e/\alpha] \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$. Then we get $c'[e/\alpha] \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ (always for any $e \in [\![A \vdash]\!]$) and we conclude with the point (iii) of proposition 1.
   – The reduction is $\mu\alpha.\langle v|\alpha\rangle \to v$ with $\alpha \notin FV(v)$. We know by hypothesis that $\mu\alpha.\langle v|\alpha\rangle \in [\![\vdash A]\!]$, then, by the point (iii) of proposition 1, $\forall e \in [\![A \vdash]\!]\ \ \langle v|\alpha\rangle[e/\alpha] \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$, i.e. $\langle v|e\rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$. If v is a variable, then we conclude immediately. if $v = \mu\beta.c$, $\langle \mu\beta.c|e\rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ implies that $c[e/\beta] \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$, which gives us $\mu\beta.c \in [\![\vdash A]\!]$ by the point (iii) of proposition 1. If $v = \lambda x.v'$, $\langle \lambda x.v'|e\rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ gives us, for $e = v_1 \cdot e_1$, $\langle v_1|\tilde{\mu}x.\langle v'|e_1\rangle\rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ then $\langle v'|e_1\rangle[v_1/x] \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ and $\langle v'[v_1/x]|e_1[v_1/x]\rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ and finally, since x is not free in e_1, $\langle v'[v_1/x]|e_1\rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$, which is enough, by the points (iv) and (iii) of proposition 1, to conclude.
1.4. $v = \lambda x.v' : A = A_1 \to A_2$ and the reduction is $\lambda x.v' \to \lambda x.v''$. By the point (iii) of proposition 1, we know that $\forall v''' \in [\![\vdash A_1]\!], e \in [\![A_2 \vdash]\!]\ \ \langle v'[v'''/x]|e\rangle \in [\![\vdash]\!] = \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$, so $\forall v''' \in [\![\vdash A_1]\!], e \in [\![A_2 \vdash]\!]\ \ \langle v''[v'''/x]|e\rangle \in [\![\vdash]\!] = \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$, and we are done.
2.x. Same as 1.x. by symmetry (where x ranges from 1 to 4).
3. $c \in [\![\vdash]\!]$: then $c \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ and $c \to c'$ implies that $c' \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}} = [\![\vdash]\!]$.

Here are now some lemmas to "inductively build" the membership of a RS.

Lemma 3.

$$v \in [\![\vdash A]\!],\ e \in [\![A \vdash]\!] \implies \langle v|e\rangle \in [\![\vdash]\!].$$

Proof. To show that $\langle v|e\rangle \in [\![\vdash]\!]$ is, by definition, to show that $\langle v|e\rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$. We take all possible pairs for v and e and we reason by induction on the strong normalisation of v and e (which we get by lemma 1) and on the length of v and e. We consider all the possible reductions of $\langle v|e\rangle$. If the reduction occurs in v or e, we conclude by induction hypothesis and lemma 2. Else,

- if $v = \mu\alpha.c$, the reduction is $\langle \mu\alpha.c|e\rangle \to c[e/\alpha]$ and we conclude by definition of $\mu\alpha.c \in [\![\vdash A]\!]$,

- if $e = \tilde{\mu}x.c$, we conclude symmetrically to the last point,
- if $v = \lambda x.v'$ and $e = v'' \cdot e'$ (with $A = A_1 \to A_2$), the reduction is $\langle \lambda x.v | v'' \cdot e' \rangle \to \langle v'' | \tilde{\mu}x.\langle v' | e' \rangle \rangle$. We consider the possible reductions of $\langle v'' | \tilde{\mu}x.\langle v' | e' \rangle \rangle$. By reducibility of v and e, we have $v'' \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ and $\langle v'[v''/x] | e' \rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$. Consequently, since the reductions cannot occur infinitely in those terms, we will get to reduce one of the following (where $v'' \to^* v_1$, $\langle v' | e' \rangle \to^* \langle v_2 | e_2 \rangle$):
  - $\langle v_1 | \tilde{\mu}x.\langle x | e_2 \rangle \rangle \to \langle v_1 | e_2 \rangle$: by induction hypothesis, we have $\langle v'' | e' \rangle \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ and $\langle v_1 | e_2 \rangle$ is one of its reducts.
  - $\langle v_1 | \tilde{\mu}x.\langle v_2 | e_2 \rangle \rangle \to \langle v_2[v_1/x] | e_2[v_1/x] \rangle$: this term is also a reduct of $\langle v'[v''/x] | e'[v''/x] \rangle$ which is in $\mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ by reducibility of v, due to the fact that since x is not free in e', hence in e_2, $e_2[v_1/x] = e_2$.
  - $\langle \mu\alpha.c_1 | \tilde{\mu}x.\langle v_2 | e_2 \rangle \rangle \to c_1[\tilde{\mu}x.\langle v_2 | e_2 \rangle / \alpha]$ with $v_1 = \mu\alpha.c_1$. By reducibility of e and by the lemma 2 we have $\mu\alpha.c_1 \in [\![\vdash A_1]\!]$, that gives us, by definition, that $c_1[\tilde{\mu}x.\langle v_2 | e_2 \rangle / \alpha]$ belongs to $[\![\vdash]\!]$ if $\tilde{\mu}x.\langle v_2 | e_2 \rangle$ belongs to $[\![A_1 \vdash]\!]$. And this last condition is satisfied, by definition, if and only if $\forall v_3 \in [\![\vdash A_1]\!]$ we have $\langle v_2[v_3/x] | e_2[v_3/x] \rangle \in [\![\vdash]\!]$, which is a consequence of the reducibility of v (with $e_2[v_3/x] = e_2$, by the same argument as above).
- If $e = \alpha\lambda.e'$ and $v = e'' \cdot v'$, we conclude symmetrically to the last point.
- In all other cases, no reduction can occur.

Lemma 4. *If $v[v'/x] \in [\![\vdash B]\!]$ for all $v' \in [\![\vdash A]\!]$ then $\lambda x.v \in [\![\vdash A \to B]\!]$. If $e[e'/\alpha] \in [\![B \vdash]\!]$ for all $e' \in\in [\![A \vdash]\!]$ then $\alpha\lambda.e \in [\![\vdash A - B]\!]$.*

Proof. By symmetry, we need only to prove one of the implications, let us take the first one. To prove that $\lambda x.v \in [\![\vdash A \to B]\!]$, we need, by the point (iii) of proposition 1, to prove that for all $v' \in [\![\vdash A]\!], e \in [\![B \vdash]\!]$, $\langle v[v'/x] | e \rangle \in [\![\vdash]\!]$. By hypothesis, we have $v[v'/x] \in [\![\vdash B]\!]$. We conclude with the lemma 3.

Here is the adequacy lemma.

Lemma 5 (Adequacy). *Let A be a type and t an object such that $FV(t) \subset X_1 \cup X_2$ ($X_1 \subset Var$ and $X_2 \subset Var^\perp$) and the variables $x_i \in X_1$ are of type B_i and the variables $\alpha_j \in X_2$ are of type C_j. For all set of objects v_i, e_j such that $\forall i\ \ v_i \in [\![\vdash A_i]\!]$ and $\forall j\ \ e_j \in [\![B_j \vdash]\!]$ we have, accordingly to the shape of t,*

1. *if $X_1 : B \vdash v : A | X_2 : C$ then $v[v_1/x_1, ..., v_n/x_n, e_1/\alpha_1, ..., e_m/\alpha_m] \in [\![\vdash A]\!]$*
2. *if $X_1 : B | e : A \vdash X_2 : C$ then $e[v_1/x_1, ..., v_n/x_n, e_1/\alpha_1, ..., e_m/\alpha_m] \in [\![A \vdash]\!]$*
3. *if $c : (X_1 : B \vdash X_2 : C)$ then $c[v_1/x_1, ..., v_n/x_n, e_1/\alpha_1, ..., e_m/\alpha_m] \in [\![\vdash]\!]$*

Remark 2. We note $X_1 : B$ the enumeration $\{x_i : B_i | i \in [1, n]\}$ (the same for $X_2 : C$).

Proof. We note $[//]$ the substitution $[v_1/x_1, ..., v_n/x_n, e_1/\alpha_1, ..., e_m/\alpha_m]$. We reason by induction on the structure of t

- $v = x$: then, by hypothesis, $\exists i,\ A = B_i$. So $v[//] = v_i \in [\![\vdash B_i]\!] = [\![\vdash A]\!]$.

- $v = e \cdot v'$: by induction hypothesis on e and v', and by the point (iii) of proposition 1, we conclude immediately.
- $v = \lambda x.v'$: we then have $A = A' \to A''$. Since we can rename bound variables, we can suppose that $x \notin \{x_1, ..., x_n\}$, which gives us $(\lambda x.v')[//] = \lambda x.(v'[//])$. By induction hypothesis, for all $v'' \in [\![\vdash A']\!]$ we have $v'[v''/x, //] \in [\![\vdash A'']\!]$ and by the lemma 4, we are done.
- $v = \mu\alpha.c$: since we can rename bound variables, we can suppose that $\alpha \notin \{\alpha_1, ..., \alpha_m\}$. Now, by the point (iii) of proposition 1, to prove that $(\mu\alpha.c)[//] = \mu\alpha.(c[//]) \in [\![\vdash A]\!]$ we need only to prove that, for all $e \in [\![A \vdash]\!]$, $c[e/\alpha, //] \in [\![\vdash]\!]$ which is done by induction hypothesis.
- e : the cases for e are similar to those for v by symmetry.
- $c = \langle v|e\rangle$. By induction hypothesis on v and e, and by the lemma 3, we conclude immediately.

We can now establish the main theorem of this section.

Theorem 1. *Every typed $\overline{\lambda}\mu\tilde{\mu}$ object is strongly normalizing.*

Proof. Let t be an object of the $\overline{\lambda}\mu\tilde{\mu}$-calcul typed by Γ and Δ, i.e. such that the conclusion of its typing judgement is either $\Gamma \vdash t : A|\Delta$, or $\Gamma|t : A \vdash \Delta$, or $t : (\Gamma \vdash \Delta)$. Suppose that its free variables are $\{\alpha_1, ..., \alpha_m, x_1, ..., x_n\}$, each one typed $x_i : A_i$ and $\alpha_i : B_i$. By the points (i) and (ii) of proposition 1, we get that for all i, $x_i \in [\![\vdash A_i]\!]$ and $\alpha_i \in [\![B_i \vdash]\!]$. Then, by the lemma 5, $t[x_1/x_1, ..., x_n/x_n, \alpha_1/\alpha_1, ..., \alpha_m/\alpha_m] = t$ is in a reducibility set. By the lemma 1, we get $t \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$.

3 PSN of $\overline{\lambda}\mu\tilde{\mu}$-Calculus with Explicit Substitutions

We first define the $\overline{\lambda}\mu\tilde{\mu}$-calculus with explicit substitutions. Then we show some useful results on the substitution calculus. And finally, we establish the property of preservation of strong normalization.

3.1 Definition

To the three syntactic categories presented in the last section, we add a fourth, regarding explicit substitutions, noted τ. In the sequel, $*$ will stand for either a term or a context variable. The syntax of the $\overline{\lambda}\mu\tilde{\mu}\mathrm{x}$-calculus is:

$$\begin{aligned} c &::= \langle v|e\rangle \mid c\tau \\ v &::= x \mid \lambda x.v \mid e \cdot v \mid \mu\alpha.c \mid v\tau \\ e &::= \alpha \mid \alpha\lambda.e \mid v \cdot e \mid \tilde{\mu}x.c \mid e\tau \\ \tau &::= [x \leftarrow v] \mid [\alpha \leftarrow e] \end{aligned}$$

The source $Dom(\tau)$ of τ is x if $\tau = [x \leftarrow v]$ and α if $\tau = [\alpha \leftarrow e]$. The body $S(\tau)$ of τ is v in the first case and e in the second. We will say that a substitution belongs to $\mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}\mathrm{x}}$ if its substituend itself belongs to $\mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}\mathrm{x}}$.

We extend the typing system by adding a new form of sequent $(\Gamma \vdash \Delta) \Rightarrow (\Gamma' \vdash \Delta')$. Here are the typing rules for explicit substitutions:

$$\frac{\Gamma \vdash v : A|\Delta}{[x \leftarrow v] : (\Gamma, x : A \vdash \Delta) \Rightarrow (\Gamma \vdash \Delta)} \qquad \frac{\Gamma|e : A \vdash \Delta}{[\alpha \leftarrow e] : (\Gamma \vdash \Delta, \alpha : A) \Rightarrow (\Gamma \vdash \Delta)}$$

$$\frac{\Gamma|e : A \vdash \Delta \quad \tau : (\Gamma \vdash \Delta) \Rightarrow (\Gamma' \vdash \Delta')}{\Gamma'|e\tau : A \vdash \Delta'} \quad \frac{\Gamma \vdash v : A|\Delta \quad \tau : (\Gamma \vdash \Delta) \Rightarrow (\Gamma' \vdash \Delta')}{\Gamma' \vdash v\tau : A|\Delta'}$$

$$\frac{c : (\Gamma \vdash \Delta) \quad \tau : (\Gamma \vdash \Delta) \Rightarrow (\Gamma' \vdash \Delta')}{c\tau : (\Gamma' \vdash \Delta')}$$

The reduction rules are the following:

$$\begin{array}{lrl}
(\beta) & \langle \lambda x.v|v' \cdot e\rangle & \to \langle v'|\tilde{\mu}x.\langle v|e\rangle\rangle \\
(\widetilde{\beta}) & \langle e' \cdot v|\alpha\lambda.e\rangle & \to \langle \mu\alpha.\langle v|e\rangle|e'\rangle \\
(mu) & \langle \mu\alpha.c|e\rangle & \to c[\alpha \leftarrow e] \\
(\widetilde{mu}) & \langle v|\tilde{\mu}x.c\rangle & \to c[x \leftarrow v]
\end{array}$$

$$\begin{array}{lrll}
(sv) & \mu\alpha.\langle v|\alpha\rangle & \to v & \text{if } \alpha \notin FV(v) \\
(se) & \tilde{\mu}x.\langle x|e\rangle & \to e & \text{if } x \notin FV(e)
\end{array}$$

$$\begin{array}{lrll}
(c\tau) & \langle v|e\rangle\tau & \to \langle v\tau|e\tau\rangle & \\
(x\tau 1) & x\tau & \to S(\tau) & \text{if } x \in Dom(\tau) \\
(x\tau 2) & x\tau & \to x & \text{if } x \notin Dom(\tau) \\
(\alpha\tau 1) & \alpha\tau & \to S(\tau) & \text{if } \alpha \in Dom(\tau) \\
(\alpha\tau 2) & \alpha\tau & \to \alpha & \text{if } \alpha \notin Dom(\tau) \\
(\cdot\tau) & (v \cdot e)\tau & \to (v\tau) \cdot (e\tau) & \\
(\tilde{\cdot}\tau) & (e \cdot v)\tau & \to (e\tau) \cdot (v\tau) & \\
(\lambda\tau) & (\lambda x.v)\tau & \to \lambda x.(v\tau) & \\
(\widetilde{\lambda}\tau) & (\alpha\lambda.e)\tau & \to \alpha\lambda.(e\tau) & \\
(\mu\tau) & (\mu\alpha.c)\tau & \to \mu\alpha.(c\tau) & \\
(\tilde{\mu}\tau) & (\tilde{\mu}x.c)\tau & \to \tilde{\mu}x.(c\tau) &
\end{array}$$

We reason modulo α-conversion on the bound variable in the rules $(\mu\tau)$, $(\tilde{\mu}\tau)$, $(\lambda\tau)$ and $(\widetilde{\lambda}\tau)$.

3.2 Substitution Calculus

We will note:

- x the set of rules concerning the propagation of substitutions, namely $c\tau$, $x\tau 1$, $x\tau 2$, $\alpha\tau 1$, $\alpha\tau 2$, $\cdot\tau$, $\tilde{\cdot}\tau$, $\lambda\tau$, $\widetilde{\lambda}\tau$, $\mu\tau$ and $\tilde{\mu}\tau$,
- $\neg$x the set of rules not in x, namely those concerning reductions of the original calculus: β, $\widetilde{\beta}$, mu, $\widetilde{mu}$, sv and se.

We present here some usual results on substitution calculi [5].

Lemma 6 (Strong normalization of $\mathtt{x}$). *$\mathtt{x}$ is strongly normalizing and its normal forms are pure objects (i.e. without substitutions).*

Proof. We define the following measure h:

$$\begin{array}{ll} h(*) = 1 & h(\langle v|e\rangle) = h(v)+h(e)+1 \\ h(v\cdot e) = h(v)+h(e)+1 & h(e\cdot v) = h(v)+h(e)+1 \\ h(\lambda x.v) = h(v)+1 & h(\alpha\lambda.e) = h(e)+1 \\ h(\mu\alpha.c) = h(c)+1 & h(\tilde{\mu}x.c) = h(c)+1 \\ h(t[*\leftarrow t']) = h(t)*(h(t')+1) & \end{array}$$

We easily check that each $\mathtt{x}$-reduction strictly decreases h. We prove by contradiction that the normal forms are pure objects: if there is a substitution, we look to the object to which it is applied and we find a reduction to perform.

We will note $\mathtt{x}(t)$ the $\mathtt{x}$-normal form of an object t.

Lemma 7 (Confluence of $\mathtt{x}$). *$\mathtt{x}$ is confluent.*

Proof. All critical pairs have disjoint redexes, which gives us local confluence. By Newman lemma and lemma 6 we get confluence.

Lemma 8 (Substitution). $\mathtt{x}(t[*\leftarrow t']) = \mathtt{x}(t)\{*\leftarrow \mathtt{x}(t')\}$.

Proof. We prove, by induction on the height of t and of the t_i, that

$$\mathtt{x}(t[*_1\leftarrow t_1]...[*_n\leftarrow t_n]) = \mathtt{x}(t)\{*_1\leftarrow \mathtt{x}(t_1)\}...\{*_n\leftarrow \mathtt{x}(t_n)\}.$$

Lemma 9 (Simulation of the $\overline{\lambda}\mu\tilde{\mu}$-calculus). *For all t and u pure objects, if $t \rightarrow_{\overline{\lambda}\mu\tilde{\mu}} u$ then $t \rightarrow^*_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} u$.*

Proof. By induction on the structure of t. The only interesting cases are those in which the reduction occurs at the root.

– $\langle\mu\alpha.c|e\rangle \rightarrow_\mu c\{*\leftarrow e\}$: we have

$$\langle\mu\alpha.c|e\rangle \rightarrow_{mu} c[*\leftarrow e] \rightarrow_{\mathtt{x}} \mathtt{x}(c[*\leftarrow e]) \overset{lemma\ 8}{=} \mathtt{x}(c)\{*\leftarrow \mathtt{x}(e)\}.$$

Since $\langle\mu\alpha.c|e\rangle$ is a pure object, $\mathtt{x}(c)=c$, $\mathtt{x}(e)=e$ and we are done.

– $\langle v|\tilde{\mu}x.c\rangle \rightarrow_\mu c\{*\leftarrow v\}$: this case is similar to the previous by symmetry.

– The other rules are simulated in one step by their homonymes in $\overline{\lambda}\mu\tilde{\mu}\mathtt{x}$.

We say that a reduction is void if it occurs in the body of a substitution $t[*\leftarrow t']$ such that $*\notin \mathtt{x}(t)$. We note it $\overset{v}{\rightarrow}$.

Lemma 10 (Projection).

1. *If $t \rightarrow_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} u$ then $\mathtt{x}(t) \rightarrow^*_{\overline{\lambda}\mu\tilde{\mu}} \mathtt{x}(u)$.*
2. *If $t \rightarrow_{\neg x} u$ is not a void reduction, then $\mathtt{x}(t) \rightarrow^+_{\overline{\lambda}\mu\tilde{\mu}} \mathtt{x}(u)$.*

Proof. We consider three cases:

– the reduction is $t \rightarrow_{\mathtt{x}} u$. Then $\mathtt{x}(t)=\mathtt{x}(u)$.

– the reduction is $t \overset{v}{\rightarrow}_{\neg\mathtt{x}} u$. Then $\mathtt{x}(t)=\mathtt{x}(u)$.

– the reduction is $t \rightarrow_{\neg\mathtt{x}} u$ and is not void. The redex appears in $\mathtt{x}(t)$ and we can reduce it, then obtain $\mathtt{x}(u)$.

3.3 Around Perpetuality

We use the perpetuality technique, formalised by Bonelli [5]. In fact, we use only the first part of the technique, which is enough to prove preservation of strong normalisation. We give some lemmas to extract a void substitution with an infinite derivation inside, and to trace this substitution backwards.

Lemma 11. *Let $t_0 \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} t_1 \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} t_2 \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} ...$ be an infinite reduction. If $\mathtt{x}(t_0) \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$, then there exists an integer k such that for all $i > k$, we have $t_i \overset{v}{\to}_{\overline{\lambda}\mu\tilde{\mu}} t_{i+1}$.*

Proof. Since $\mathtt{x}$ is strongly normalizing, the reduction must be $t_0 \to^*_{\mathtt{x}} t_1 \to_{\neg\mathtt{x}} t_2 \to^*_{\mathtt{x}} t_3 \to_{\neg\mathtt{x}} t_4$... By lemma 10, we have $\mathtt{x}(t_0) \to^*_{\overline{\lambda}\mu\tilde{\mu}} \mathtt{x}(t_1) \to^*_{\overline{\lambda}\mu\tilde{\mu}} \mathtt{x}(t_2) \to^*_{\overline{\lambda}\mu\tilde{\mu}} \mathtt{x}(t_3) \to^*_{\overline{\lambda}\mu\tilde{\mu}} \mathtt{x}(t_4)$... Furthermore, for all even i, if $t_{i+1} \to_{\neg\mathtt{x}} t_{i+2}$ is not a void reduction, then $\mathtt{x}(t_i) \to^+_{\overline{\lambda}\mu\tilde{\mu}} \mathtt{x}(t_{i+2})$. From $\mathtt{x}(t_0) \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$ we deduce that there exists k such that for all even i greater than k we have $t_{i+1} \overset{v}{\to}_{\neg\mathtt{x}} t_{i+2}$. We must now prove that from a certain point, both $\neg\mathtt{x}$ and $\mathtt{x}$ reductions are void. For that, we define the following measure:

$$\begin{array}{llll}
h(*) & = 1 & h(\langle v|e\rangle) & = h(v) + h(e) + 1 \\
h(\mu\alpha.c) & = h(c) + 1 & h(\tilde{\mu}x.c) & = h(c) + 1 \\
h(t[* \leftarrow t']) & = \begin{cases} h(t) * (h(t') + 1) & \text{if } * \in FV(\mathtt{x}(t)) \\ h(t) * 2 & \text{else} \end{cases} & &
\end{array}$$

The last clause guarantees that a void reduction leaves the measure unchanged. We easily satisfies that all other reductions strictly decraese this measure, and we conclude.

The next notion is useful to isolate a void substitution.

Definition 1 (Skeleton). *The skeleton of an object, noted $SK(t)$, is inductively defined as follows:*

$$\begin{array}{llll}
SK(*) & = * & SK(\langle v|e\rangle) & = \langle SK(v)|SK(e)\rangle \\
SK(\mu\alpha.c) & = \mu\alpha.SK(c) & SK(\tilde{\mu}x.c) & = \tilde{\mu}x.SK(c) \\
SK(t[* \leftarrow u]) & = SK(t)[* \leftarrow \bullet] & &
\end{array}$$

We remark that if $t \overset{v}{\to} u$, then $SK(t) = SK(u)$.

The following lemma says that if there is an infinite derivation, then there exists a substitution in which there is an infinite derivation.

Lemma 12. *Let an infinite derivation be $t_0 \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} t_1 \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} t_2 \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} ...$ If $\mathtt{x}(t_0) \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$, then there exists an integer k, an object t, a variable $*$, a context C and an object sequence u_i such that*

$$\begin{array}{rcl}
t_0 \to^*_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} t_k & = & C[t[* \leftarrow u_k]] \\
& \overset{v}{\to}_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} & C[t[* \leftarrow u_{k+1}]] \\
& \overset{v}{\to}_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} & C[t[* \leftarrow u_{k+2}]] \ldots
\end{array}$$

with $u_k \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} u_{k+1} \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} u_{k+2} \to_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}} u_{k+3}$...

Proof. By lemma 11, there exists k such that for all $i > k$, $t_i \rightarrow^{v}_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} t_{i+1}$. Then, we have $SK(t_k) = SK(t_i)$ for all $i \geq k$. The derivation tree of t_k being infinite, by the pigeon hole principle, an infinite derivation must take place in the same substitution of $SK(t_k)$, and we are done.

Lemma 13 (Substitution tracing – 1 step). *Let t and u be two objects such that $t \rightarrow_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} u$ and $u = C[u_1[* \leftarrow u_2]]$. Then*

1. *either $t = C'[u'_1[* \leftarrow u_2]]$,*
2. *or $t = C'[u'_1[* \leftarrow u'_2]]$ with $u_2 \rightarrow u'_2$,*
3. *or u_1 is a command and*
   if $* = \alpha$ then $t = C[\langle \mu\alpha.u_1 | u_2 \rangle]$ else $t = C[\langle u_2 | \tilde{\mu}x.u_1 \rangle]$.

Proof. We reason by induction on t and we consider the following two cases:

- The reduction takes place at the root. First note that if $u_1[* \leftarrow u_2]$ appears in a sub-term of u, which is also a sub-term of t, then for a context C' and $u'_1 = u_1$ the first item holds. This applies also when the rule used to reduce at the root is one of $x\tau$ or $\alpha\tau$. Else if the rule is mu or $\widetilde{mu}$, then the third item holds, else if it is another rule, then the first item holds, in both cases, we use the empty context.
- The reduction is internal.
  - $t = *$. The result holds trivially.
  - $t = \langle v | e \rangle$ with either $v \rightarrow_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} v'$ or $e \rightarrow_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} e'$. We consider the first case, since the second one is similar. We have $u = \langle v' | e \rangle$ and:
    - ⋆ if the sub-term $u_1[* \leftarrow u_2]$ occurs in v', then we use induction hypothesis.
    - ⋆ else the sub-term $u_1[* \leftarrow u_2]$ occurs in e ; then the first item holds.
  - $t = v \cdot e$ or $t = e \cdot v$ with either $v \rightarrow_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} v'$ or $e \rightarrow_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} e'$. We conclude similarly to the previous point.
  - $t = \mu\alpha.c$ or $\tilde{\mu}x.c$ or $\lambda x.v$ or $\alpha\lambda.e$. We use induction hypothesis.
  - $t = t_1[* \leftarrow t_2]$. There are two cases:
    - ⋆ $t_1 \rightarrow_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} t'_1$ and $u = t'_1[* \leftarrow t_2]$. Then if $u_1[* \leftarrow u_2]$ occurs in t'_1 we use induction hypothesis. If it occurs in t_2 the first item holds trivially. Finally, if $u = u_1[* \leftarrow u_2]$ then we take the empty context for C', $u'_1 = t_1$ and the first item holds.
    - ⋆ $t_2 \rightarrow_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} t'_2$ and $u = t_1[* \leftarrow t'_2]$. Then if $u_1[* \leftarrow u_2]$ occurs in t_1 the first item holds trivially. If it occurs in t'_2 we use induction hypothesis. Finally, if $u = u_1[* \leftarrow u_2]$ then we take the empty context for C', $u'_1 = t_1$ and $u'_2 = t_2$ and the second item holds.

This result is naturally extended to many-steps reductions.

Lemma 14 (Substitution tracing). *Let $t_1, ..., t_n$ be objects such that, for all i, $t_i \rightarrow_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} t_{i+1}$ and $t_n = C[u_1[* \leftarrow u_2]]$. Then*

1. *either $* = \alpha$ and there is i such that $t_i = C'[\langle \mu\alpha.u'_1 | u'_2 \rangle]$ with $u_2 \rightarrow^{*}_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} u'_2$,*
2. *or $* = x$ and there is i such that $t_i = C'[\langle u'_2 | \tilde{\mu}x.u'_1 \rangle]$ with $u_2 \rightarrow^{*}_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} u'_2$,*
3. *or $t_1 = C'[u'_1[* \leftarrow u'_2]]$ with $u_2 \rightarrow^{*}_{\overline{\lambda\mu\tilde{\mu}\mathrm{x}}} u'_2$.*

Proof. By induction on the number of reduction steps, using lemma 13.

We formalise the notion of derivation ordering.

Definition 2. *Let ϕ and ψ be two infinite derivations starting form an object t_1. Then ϕ is called smaller than ψ if they reduce the same redexes for the first $n-1$ steps, and the nth redex reduced by ϕ is a strict subterm of the nth redex reduced by ψ.*

Here is the main theorem of this section.

Theorem 2 (PSN). $t \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}} \Rightarrow t \in \mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}}$.

Proof. By contradiction. Suppose that there exists a pure term t which can be infinitely reduced in the $\overline{\lambda}\mu\tilde{\mu}\mathtt{x}$-calculus. We take a minimal derivation of this term. By lemma 12, at a certain point, we can exhibit a infinite derivation in a void substitution. By lemma 14, we can go backwards until we reach the reduction which creates this substitution while keeping the infinite reduction in it. This creation point (chosen by the minimal derivation) is a proper prefix of the reduction point of the infinite derivation inside the future body of the void substitution. This contradicts the minimality of the derivation.

4 PSN Implies SN

4.1 Proof Technique

The technique we present here is very general and can be applied to many calculi with explicit substitutions. The idea of this technique is the following : let t be a typed term with explicit substitutions, with its typing judgement, we build a typed term t' of the pure calculus by expanding the substitutions of t in redexes. We call this expansion *Ateb*. We require the following two properties, which are enough to establish theorem 3.

Property 1 (Preservation of typability). If t is typable in the calculus with explicit substitution, then $Ateb(t)$ is typable in the pure calculus.

Property 2 (Initialization). $Ateb(t)$ reduces to t in 0 or more steps in the calculus with explicit substitutions.

We can now establish the theorem.

Theorem 3. *For all typing system such that all typable terms are strongly normalizing, if there exists a function Ateb from explicit substitution terms to pure terms satisfying properties 1 and 2 then PSN implies SN.*

Proof. For all typed term t of the calculus with explicit substitution, $Ateb(t)$ is a pure typed term (by property 1). By hypothesis of strong normalization of the pure typed calculus, we have $Ateb(t) \in \mathcal{SN}$ (in the present case $\mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}}$). By hypothesis of PSN we obtain that $Ateb(t)$ is in $\mathcal{SN}$ (in the present case $\mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}}$). By property 2, we get $Ateb(t) \rightarrow^* t$, which gives us directly $t \in \mathcal{SN}$ (in the present case $\mathcal{SN}_{\overline{\lambda}\mu\tilde{\mu}\mathtt{x}}$).

4.2 Application to $\overline{\lambda}\mu\tilde{\mu}$

Here is the definition of *Ateb*. It is obvious that for all t, $Ateb(t)$ contains no substitutions. We then check that this function satisfies the two properties we mention above.

Definition 3.

$$\begin{array}{llll}
Ateb(x) & = x & Ateb(\alpha) & = \alpha \\
Ateb(\lambda x.v) & = \lambda x.Ateb(v) & Ateb(\alpha\lambda.e) & = \alpha\lambda.Ateb(e) \\
Ateb(\mu\alpha.c) & = \mu\alpha.Ateb(c) & Ateb(\widetilde{\mu}x.c) & = \widetilde{\mu}x.Ateb(c) \\
Ateb(e \cdot v) & = Ateb(e) \cdot Ateb(v) & Ateb(v \cdot e) & = Ateb(v) \cdot Ateb(e) \\
Ateb(\langle v|e\rangle) & = \langle Ateb(v)|Ateb(e)\rangle & &
\end{array}$$

$$\begin{array}{lll}
Ateb(c[x \leftarrow v]) & = \langle Ateb(v)|\widetilde{\mu}x.Ateb(c)\rangle & \\
Ateb(c[\alpha \leftarrow e]) & = \langle \mu\alpha.Ateb(c)|Ateb(e)\rangle & \\
Ateb(v[x \leftarrow v']) & = \mu\alpha.\langle \lambda x.Ateb(v)|Ateb(v') \cdot \alpha\rangle & \textit{With } \alpha \textit{ fresh variable} \\
Ateb(v[\alpha \leftarrow e]) & = \mu\beta.\langle \mu\alpha.\langle Ateb(v)|\beta\rangle|Ateb(e)\rangle & \textit{With } \beta \textit{ fresh variable} \\
Ateb(e[x \leftarrow v]) & = \widetilde{\mu}y.\langle Ateb(v)|\widetilde{\mu}x.\langle y|Ateb(e)\rangle\rangle & \textit{With } y \textit{ fresh variable} \\
Ateb(e[\alpha \leftarrow e']) & = \widetilde{\mu}x.\langle Ateb(e') \cdot x|\alpha\lambda.Ateb(e)\rangle & \textit{With } x \textit{ fresh variable}
\end{array}$$

Proof. (of property 1) Easy by induction on the proof of the typing judgement of t.

Proof. (of property 2) We proceed by induction on t. Only the cases for substitutions are not easy. By the symmetry of the system, we consider only one half of it.

– We have $Ateb(c[x \leftarrow v]) = \langle Ateb(v)|\widetilde{\mu}x.Ateb(c)\rangle$ and

$$\langle Ateb(v)|\widetilde{\mu}x.Ateb(c)\rangle \to_{\mu} Ateb(c)[x \leftarrow Ateb(v)].$$

– We have $Ateb(v[x \leftarrow v']) = \mu\alpha.\langle \lambda x.Ateb(v)|Ateb(v') \cdot \alpha\rangle$ and

$$\begin{array}{ll}
 & \mu\alpha.\langle \lambda x.Ateb(v)|Ateb(v') \cdot \alpha\rangle \\
\to_{\beta} & \mu\alpha.\langle Ateb(v')|\widetilde{\mu}x.\langle Ateb(v)|\alpha\rangle\rangle \to_{\widetilde{\mu}} \mu\alpha.(\langle Ateb(v)|\alpha\rangle[x \leftarrow Ateb(v')]) \\
\to_{c\tau} & \mu\alpha.\langle Ateb(v)[x \leftarrow Ateb(v')]|\alpha[x \leftarrow Ateb(v')]\rangle \\
\to_{\alpha\tau 2} & \mu\alpha.\langle Ateb(v)[x \leftarrow Ateb(v')]|\alpha\rangle \to_{sv} Ateb(v)[x \leftarrow Ateb(v')].
\end{array}$$

– We have $Ateb(v[\alpha \leftarrow e]) = \mu\beta.\langle \mu\alpha.\langle Ateb(v)|\beta\rangle|Ateb(e)\rangle$ and

$$\begin{array}{ll}
 & \mu\beta.\langle \mu\alpha.\langle Ateb(v)|\beta\rangle|Ateb(e)\rangle \to_{\mu} \mu\beta.(\langle Ateb(v)|\beta\rangle[\alpha \leftarrow Ateb(e)]) \\
\to_{c\tau} & \mu\beta.\langle Ateb(v)[\alpha \leftarrow Ateb(e)]|\beta[\alpha \leftarrow Ateb(e)]\rangle \\
\to_{\alpha\tau 2} & \mu\beta.\langle Ateb(v)[\alpha \leftarrow Ateb(e)]|\beta\rangle \to_{sv} Ateb(v)[\alpha \leftarrow Ateb(e)].
\end{array}$$

In each case, we conclude by induction hypothesis.

We can use Theorem 3.

4.3 Strong Normalization of $\overline{\lambda}\mu\tilde{\mu}$x-Calculus

We collect together our results to prove the main theorem of this work.

Theorem 4. *The typed $\overline{\lambda}\mu\tilde{\mu}\mathtt{x}$-calculus is strongly normalizing.*

Proof. By Theorem 1 (SN for pure calculus), Theorem 2 (PSN) and Theorem 3 (PSN implies SN).

5 Achievements and Perspectives

Using various proof techniques, we have established that the $\overline{\lambda}\mu\tilde{\mu}\mathtt{x}$-calculus is strongly normalizing. For that purpose, we have formalized a proof technique of SN *via* PSN. Let us mention that we have successfully applied this technique, with some adjustments, to prove SN of the $\lambda\upsilon$-calculus (introduced in [3]) for the first time, as far as we know. We also used it to establish that PSN implies SN for the $\lambda\sigma$-calculus [1], for which PSN is known to fail [10], showing that, for this calculus, the only problem of SN is in PSN.

It remains an open problem to build a direct proof, by the reducibility technique, of SN for a symmetric non-deterministic calculus with explicit substitutions. Another direction of work could be to replace substitutions "à la" $\lambda\mathtt{x}$ by substitutions "à la" λ_{ws} [8], which yields, through the addition of explicit weakenings, a more powerful substitution system. It may even help us to find a direct proof of SN. At last, we plan to work on a second order version of $\overline{\lambda}\mu\tilde{\mu}\mathtt{x}$.

References

1. Abadi, M., Cardelli, L., Curien, P.-L., Lévy, J.-J.: Explicit Substitutions. Journal of Functional Programming (1991).
2. Barbanera, F., Berardi, S.: A symmetric lambda-calculus for classical program extraction. Proceedings of TACS'94 (1994), Springer-Verlag LNCS **789**, 495–515.
3. Benaissa, Z.-E.-A., Briaud, D., Lescanne, P., Rouyer-Degli, J.: $\lambda\upsilon$, a calculus of explicit substitutions which preserves strong normalisation. Journal of Functional Programming (1996).
4. Bloo, R., Geuvers, H.: Explicit Substitution: on the Edge of Strong Normalisation. Theoretical Computer Science (1999), **211**, 375–395.
5. Bonelli, E.: Substitutions explicites et réécriture de termes. PhD thesis, Université Paris XI Orsay (2001).
6. Church, A.: The Calculi of Lambda Conversion. Princeton Univ. Press (1941).
7. Curien, P.-L., Herbelin, H.: The duality of computation. Proceedings of ICFP'00 (2000), ACM Press, 233–243.
8. Guillaume, B.: Un calcul de substitution avec étiquettes. PhD thesis, Université de Savoie (1999).
9. Herbelin, H.: Explicit substitutions and reducibility. Journal of Logic and Computation (2001), **11**, 429–449.
10. Mellies, P.-A.: Typed λ-calculi with explicit substitutions may not terminate. Proceedings of TLCA'95 (1995), Springer LNCS, **902**, 328–334.
11. Parigot, M.: $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. Proceedings of LICS'93 (1993), Computer Society Press, 39–46.