Meta-Theoretical Properties of lambda_phi: A Left-Linear Variant of lambda_sigma

César Muñoz

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET AUTOMATIQUE

Meta-theoretical properties of λ_ϕ: A left-linear variant of λ_σ

César A. Muñoz H.

N ° 3107
Février 1997

THÈME 2

ISSN 0249-6399

Meta-theoretical properties of λ_ϕ: A left-linear variant of λ_σ

César A. Muñoz H. *

Thème 2 — Génie logiciel
et calcul symbolique

Projet Coq

Rapport de recherche n ° 3107 — Février 1997 — 22 pages

Abstract: In this paper we consider explicit substitutions calculi that allow open terms. In particular, we propose a variant of the λ_σ-calculus, that we call λ_ϕ. For this calculus and its simply-typed version, we study its meta-theoretical properties. The λ_ϕ-calculus enjoys the same general characteristics as λ_σ, i.e. a simple and finitary first-order presentation, confluent on terms with meta-variables, with a composition operator and with simultaneous substitutions. However, λ_ϕ does not have the non-left-linear surjective pairing rule of λ_σ which raises technical problems in some frameworks.

(Résumé : tsvp)

*Cesar.Munoz@inria.fr

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
Téléphone : (33 1) 39 63 55 11 – Télécopie : (33 1) 39 63 53 30

Propriétés méta-théoriques de λ_ϕ: Une variante linéaire à gauche de λ_σ

Résumé : Dans cet article, on s'intéresse aux calculs avec substitutions explicites qui admettent des termes avec des méta-variables. En particulier, on propose une variante du λ_σ-calcul, qu'on appelle λ_ϕ. Pour la version pure et pour la version simplement typée de ce calcul on étudie les propriétés méta-théoriques. Le calcul λ_ϕ satisfait les mêmes propriétés générales de λ_σ, c'est-à-dire:

- une présentation simple, finitaire et de premier ordre,
- confluent pour les termes avec des méta-variables
- avec des opérateurs de composition et de substitutions simultanées.

Mais λ_ϕ ne comporte pas la règle non-linéaire de λ_σ qui pose des problèmes techniques dans certaines applications.

1 Introduction

There are several versions of explicit substitutions calculi (see for example [Les94] for a overview of some of them, but also among others [KR95], [BR95], [LRD95], [Kes96], [Muñ96], [FKP96], ...). All these calculi implement λ-calculus by means of a lazy mechanism of reduction of substitutions. In order to consider meta-variables, most of them have a strong drawback: non-confluence on terms with meta-variables. Confluence and weak normalization are sufficient to decide equivalence of terms. Hence, these two properties seem to be desirable in any extension of λ-calculus with meta-variables and explicit substitutions.

The λ_σ-calculus[1] is one of the most popular explicit substitutions calculus. It is a first-order rewriting system with two sorts of expressions: terms and substitutions. In this calculus, free and bound variables are represented by de Bruijn's indices, and hence, λ-terms correspond to ground λ_σ-terms without substitutions. Confluence and termination properties of λ_σ for ground terms are shown in [ACCL91]. The λ_σ-calculus is not confluent on general open terms ([CHL96]), however [Río93] has proved that it is confluent on semi-open terms, i.e. expressions with meta-variables of terms but not meta-variables of substitutions.

Others calculi that are confluent on terms with meta-variables are $\lambda_{\Uparrow}$ ([CHL96]), $\lambda_{\mathcal{S}_e}$ ([KR95]) and λ_ζ ([Muñ96]). In particular, $\lambda_{\Uparrow}$ is a confluent first-order rewriting system. Compared with $\lambda_{\Uparrow}$, λ_σ has less rewriting rules and it is compatible with the η-rule. But it has also some advantages whit respect to $\lambda_{\mathcal{S}_e}$ and λ_ζ: λ_σ is a finitary first-order system and it allows composition and simultaneous substitutions.

In λ_σ the composition operator was introduced to solve a critical pair, and so, to gain local confluence. Composition of substitutions introduces simultaneous substitutions that happens to be useful for several purposes. For example, the modeling of closures of an abstract machine ([HMP95]) or the pruning of search space in unifications algorithms ([DHK95]). Also, these features improve the substitution mechanism by allowing parallel substitutions of variables. But also, composition of substitutions and simultaneous substitutions are responsible of the following non-left-linear rule: $\mathbf{1}[S] \cdot (\uparrow \circ S) \xrightarrow{\text{(SCons)}} S$. Informally, if we interpret S as a list, $\mathbf{1}$ as the head function and $\uparrow$ as the tail function, then this rule correspond to the surjective-pairing rule.

The (SCons) rule, $\mathbf{1}[S] \cdot \uparrow \circ S \longrightarrow S$, is impractical for many reasons. In fact, [Muñ97] has shown that λ_σ may loses the subject reduction property in a dependent types system due to the (SCons) rule. But also, independently, Nadathur has remarked in [Nad96] that this non-left-linear rule is difficult to handle in implementations. He has remarked that in λ_σ with semi-open terms this rule is admissible when the following scheme of rule is maintained: $\mathbf{1}[\uparrow^n] \cdot \uparrow^{n+1} \xrightarrow{\text{(SCons)}} \uparrow^n$, where $\uparrow^n$ is a notation for "$\overbrace{\uparrow \circ \ldots \circ \uparrow}^{n\text{-times}}$".

Following this idea, we propose an explicit substitutions calculus that keep the same spirit of λ_σ, i.e. a simple and finitary first-order presentation, confluent on terms with meta-variables, with a composition operator of substitutions and with simultaneous substitutions. But in contrast to λ_σ, the new calculus has not the (SCons) rule and it is extensible to higher-order typed systems.

The rest of the paper is organized as follows. In Section 2 we present the λ_ϕ-calculus. Confluence and normalization properties of the calculus of substitutions are proved in Sections 3 and 4. In Section 5 we show that λ_ϕ is confluent on λ-terms with meta-variables. Finally, in Section 6 we study the simply typed version of λ_ϕ. In particular we prove subject reduction and weak normalization properties.

2 λ_ϕ-Calculus

The finitary presentation of the scheme suggested by Nadathur is gained by the introduction of a sort to represent natural numbers and an adequate set of rewriting rules to compute with them.

The set Λ_ϕ of well formed expressions is defined by the following grammar:

[1] In this paper we use λ_σ to designate the locally confluent calculus proposed in [ACCL91].

$(\lambda M)N$	$\longrightarrow$	$M[N \cdot \uparrow^0]$	(Beta)
$(\lambda M)[S]$	$\longrightarrow$	$\lambda M[\mathbf{1} \cdot (S \circ \uparrow^{Suc(0)})]$	(Lambda)
$(M\ N)[S]$	$\longrightarrow$	$M[S]\ N[S]$	(App)
$M[S][T]$	$\longrightarrow$	$M[S \circ T]$	(Clos)
$\mathbf{1}[M \cdot S]$	$\longrightarrow$	M	(VarCons)
$M[\uparrow^0]$	$\longrightarrow$	M	(Id)
$(M \cdot S) \circ T$	$\longrightarrow$	$M[T] \cdot (S \circ T)$	(Map)
$\uparrow^0 \circ S$	$\longrightarrow$	S	(IdS)
$\uparrow^{Suc(n)} \circ (M \cdot S)$	$\longrightarrow$	$\uparrow^n \circ S$	(ShiftCons)
$\uparrow^{Suc(n)} \circ \uparrow^m$	$\longrightarrow$	$\uparrow^n \circ \uparrow^{Suc(m)}$	(ShiftShift)
$\mathbf{1} \cdot \uparrow^{Suc(0)}$	$\longrightarrow$	$\uparrow^0$	(Shift0)
$\mathbf{1}[\uparrow^n] \cdot \uparrow^{Suc(n)}$	$\longrightarrow$	$\uparrow^n$	(ShiftS)

Figure 1: The rewrite system λ_ϕ.

Naturals	n	::=	$0 \mid Suc(n)$
Terms	M, N	::=	$\mathbf{1} \mid \lambda M \mid M\ N \mid M[S]$
Substitutions	S, T	::=	$\uparrow^n \mid M \cdot S \mid S \circ T$

We define the system λ_ϕ in Fig. 1. The system ϕ is obtained by dropping the (Beta) rule from λ_ϕ.

Technically, λ_ϕ is not a left-linear rewriting system due to the (ShiftS) rule. However, its deviation is not harmful since the term with a double occurrence in this rule can be considered as a constant. In other words, since natural numbers are never reduced, the λ_ϕ-calculus can be considered as left-linear in the sort of terms and substitutions.

We remark that the rewriting system ϕ is not confluent, not even locally confluent, on open expressions. We can check mechanically, for example using the RRL ([KZ89]) system, in order-manual option, that ϕ has the following critical pairs:

- **(Id-Clos)**. $M[S] \xleftarrow{\phi^+} M[S][\uparrow^0] \xrightarrow{\phi^+} M[S \circ \uparrow^0]$.
- **(Clos-Clos)**. $M[(S \circ T) \circ T'] \xleftarrow{\phi^+} M[S][T][T'] \xrightarrow{\phi^+} M[S \circ (T \circ T')]$.
- **(Shift0-Map)**. $S \xleftarrow{\phi^+} (\mathbf{1} \cdot \uparrow^{Suc(0)}) \circ S \xrightarrow{\phi} \mathbf{1}[S] \cdot (\uparrow^{Suc(0)} \circ S)$.
- **(ShiftS-Map)**. $\uparrow^n \circ S \xleftarrow{\phi} (\mathbf{1}[\uparrow^n] \cdot \uparrow^{Suc(n)}) \circ S \xrightarrow{\phi^+} \mathbf{1}[\uparrow^n \circ S] \cdot (\uparrow^{Suc(n)} \circ S)$.
- **(Lambda-Clos)**. $\lambda M[\mathbf{1} \cdot ((S \circ \uparrow^{Suc(0)}) \circ (\mathbf{1} \cdot (T \circ \uparrow^{Suc(0)})))] \xleftarrow{\phi^+} (\lambda M)[S][T] \xrightarrow{\phi^+}$ $\lambda M[\mathbf{1} \cdot ((S \circ T) \circ \uparrow^{Suc(0)})]$.

If we consider the (Beta) rule, we have additionally the following critical pair with the (App) rule: $M[N[S] \cdot S] \xleftarrow{\lambda_\phi{}^+} ((\lambda M)\ N)[S] \xrightarrow{\lambda_\phi{}^+} M[N[S] \cdot ((S \circ \uparrow^{Suc(0)}) \circ (M \cdot \uparrow^0))]$.

However, we are going to prove that these critical pairs are ϕ-joinable in the set of expressions that contain meta-variables only of sort term, i.e. no meta-variables of sorts substitution or natural. This set is defined as the set of *quasi-open* expressions of Λ_ϕ and it is noted by $\Lambda_\mathcal{Q}$.

Definition 1 (Addition) *We define the* addition *function on the set of ground naturals.*

$$\begin{array}{lcl} 0+m & = & m \\ Suc(n)+m & = & Suc(n+m) \end{array}$$

The following properties are proved by structural induction on naturals.

Lemma 1 (Addition Properties) *For any n, m and r ground naturals:*

1. *(Commutativity)* $n+m=m+n$.
2. *(Associativity)* $n+(m+r)=(n+m)+r$.
3. *(Right Zero)* $n+0=n$.
4. *(Rigth Successor)* $n+Suc(m)=Suc(m+n)$.
5. *(Arrow Composition)* $\uparrow^n \circ \uparrow^m \xrightarrow{\phi^*} \uparrow^{n+m}$.

Proposition 1 *The set of ϕ-normal forms in $\Lambda_{\mathcal{Q}}$ is defined by the following grammar:*

$$\begin{array}{llll} \textbf{Naturals} & n & ::= & 0 \mid Suc(n) \\ \textbf{Terms} & M,N & ::= & \mathbf{1} \mid \mathbf{1}[\uparrow^{Suc(n)}] \mid \lambda M \mid M\ N \mid x \mid x[S'] \\ \textbf{Substitutions} & S & ::= & \uparrow^0 \mid S' \\ & S' & ::= & \uparrow^{Suc(n)} \mid M \cdot S \text{ with } M \cdot S \neq \mathbf{1} \cdot \uparrow^{Suc(0)} \text{ and} \\ & & & M \cdot S \neq \mathbf{1}[\uparrow^n] \cdot \uparrow^{Suc(n)} \end{array}$$

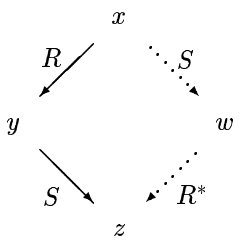
where x is a meta-variable of sort term.

Proof. First we check that the forms described by the above grammar are ϕ-normal forms. Then we verify that there are no any more. □

3 ϕ-Normalization

The idea of this normalization proof[2] is to split the ϕ-calculus in two terminating systems, and to use a sufficient condition to prove termination of the union of the systems. A similar technique is used in [Muñ96] and [FKP96] to prove termination of explicit substitutions calculi.

Definition 2 (Commutation) *Let R and S be two relations defined on the set X. We say that R* commutes over *S if and only if for any $x,y,z \in X$ such that $x \xrightarrow{R} y$ and $y \xrightarrow{S} z$ there exists $w \in X$ such that $x \xrightarrow{S} w$ and $w \xrightarrow{R^*} z$, i.e. the following diagram holds:*



We use the following well known property of commuting relations.

Lemma 2 (Bachmair-Dershowitz) *Let R and S be two relations defined on a set X such that: 1. R and S are terminating and 2. R commutes over S, then the relation $R \cup S$ is terminating.*

$\uparrow^{Suc(n)} \circ (M_1 \cdot \ldots \cdot M_{n+1} \cdot S)$	$\longrightarrow$	S	(ShiftCons')
$\uparrow^{Suc(n)+m} \circ (M_1 \cdot \ldots \cdot M_{n+1} \cdot \uparrow^{m'})$	$\longrightarrow$	$\uparrow^{m+m'}$	(ShiftCons")
$\uparrow^{Suc(n)} \circ \uparrow^{m}$	$\longrightarrow$	$\uparrow^{Suc(n+m)}$	(ShiftShift')

Figure 2: The rewrite system ϕ_3

$[\![\mathbf{1}]\!]$	=	1
$[\![\lambda M]\!]$	=	$[\![M]\!]$
$[\![M\ N]\!]$	=	$[\![M]\!] + [\![N]\!]$
$[\![M[S]]\!]$	=	$[\![M]\!] + [\![S]\!]$
$[\![\uparrow^n]\!]$	=	$[\![n]\!]$
$[\![M \cdot S]\!]$	=	$[\![M]\!] + [\![S]\!]$
$[\![S \circ T]\!]$	=	$2[\![S]\!] + [\![T]\!]$
$[\![0]\!]$	=	1
$[\![Suc(n)]\!]$	=	$1 + [\![n]\!]$

Figure 3: Interpretation for proving the termination of ϕ_1

The ϕ-system is split as follows: $\phi_1 = \{$(ShiftCons), (ShiftShift)$\}$ and $\phi_2 = \{$(Lambda), (App), (Clos), (VarCons), (Id), (Map), (IdS), (Shift0), (ShiftS)$\}$. In order to apply Lemma 2, we need to extend ϕ_2 with the infinite additional rules represented by the rules schemes of ϕ_3 (Fig. 2).

If we take the set $\Lambda_{\mathcal{Q}}$ as X, ϕ_1 as R and $(\phi_2 \cup \phi_3)$ as S, then the conditions of Lemma 2 are satisfied. First we prove that ϕ_1 and $(\phi_2 \cup \phi_3)$ are both terminating systems.

Proposition 2 *ϕ_1 is terminating.*

Proof. We verify that the polynomial interpretation given in Fig. 3 defines a reduction order for ϕ_1.

- (ShiftCons). $[\![\uparrow^{Suc(n)} \circ (M \cdot S)]\!] = 2 + 2[\![n]\!] + [\![M]\!] + [\![S]\!] > 2[\![n]\!] + [\![S]\!] = [\![\uparrow^n \circ S]\!]$.
- (ShiftShift). $[\![\uparrow^{Suc(n)} \circ \uparrow^m]\!] = 2 + 2[\![n]\!] + [\![m]\!] > 2[\![n]\!] + 1 + [\![m]\!] = [\![\uparrow^n \circ \uparrow^{Suc(m)}]\!]$.

□

In order to verify that $(\phi_2 \cup \phi_3)$-reductions are simulated by σ-reductions (Fig. 4), we define the following mapping from Λ_ϕ-expressions to Λ_σ-expressions:

[2] Recently, Hans Zantema has found a simpler proof of termination for the ϕ-calculus ([Zan96]). His proof uses the semantic labelling technique ([Zan94]).

$(M\ M')[S]$	$\longrightarrow$	$M[S]\ M'[S]$	(App)
$(\lambda M)[S]$	$\longrightarrow$	$\lambda(M[\mathbf{1}\cdot(S\circ\uparrow)])$	(Lambda)
$M[S][S']$	$\longrightarrow$	$M[S\circ S']$	(Clos)
$\mathbf{1}[M\cdot S]$	$\longrightarrow$	M	(VarCons)
$M[id]$	$\longrightarrow$	M	(Id)
$(S\circ S')\circ S''$	$\longrightarrow$	$S\circ(S'\circ S'')$	(Ass)
$(M\cdot S)\circ S'$	$\longrightarrow$	$M[S']\cdot(S\circ T)$	(Map)
$id\circ S$	$\longrightarrow$	S	(Idl)
$S\circ id$	$\longrightarrow$	S	(Idr)
$\uparrow\circ(M\cdot S)$	$\longrightarrow$	S	(ShiftCons)
$\mathbf{1}\cdot\uparrow$	$\longrightarrow$	id	(VarShift)
$\mathbf{1}[S]\cdot(\uparrow\circ S)$	$\longrightarrow$	S	(SCons)

Figure 4: The rewriting system σ

$$
\begin{array}{lcl}
\overline{\mathbf{1}} & = & \mathbf{1} \\
\overline{\lambda M} & = & \lambda\overline{M} \\
\overline{M\ N} & = & \overline{M}\ \overline{N} \\
\overline{M[S]} & = & \overline{M}[\overline{S}] \\
\overline{\uparrow^0} & = & id \\
\overline{\uparrow^{Suc(0)}} & = & \uparrow \\
\overline{\uparrow^{Suc(Suc(n))}} & = & \uparrow\circ\overline{\uparrow^{Suc(n)}} \\
\overline{M\cdot S} & = & \overline{M}\cdot\overline{S} \\
\overline{S\circ T} & = & \overline{S}\circ\overline{T}
\end{array}
$$

Lemma 3 (Simulation of (ShiftCons')) $\overline{\uparrow^{Suc(n)}\circ(M_1\cdot\ldots\cdot M_{\mathrm{n}+1}\cdot S)} \xrightarrow{\sigma^+} \overline{S}$.

Proof. By structural induction on n.

- $n=0$. By definition, $\overline{\uparrow^{Suc(0)}\circ(M_1\cdot S)} = \uparrow\circ(\overline{M_1}\cdot\overline{S})$. But also, $\uparrow\circ(\overline{M_1}\cdot\overline{S}) \xrightarrow{\sigma} \overline{S}$.
- $n=Suc(n')$. By definition, $\overline{\uparrow^{Suc(Suc(n'))}\circ(M_1\cdot\ldots\cdot M_{\mathrm{n'}+2}\cdot S)} = (\uparrow\circ\overline{\uparrow^{Suc(n')}})\circ(\overline{M_1\cdot\ldots\cdot M_{\mathrm{n'}+2}\cdot S})$. But also, $(\uparrow\circ\overline{\uparrow^{Suc(n')}})\circ(\overline{M_1\cdot\ldots\cdot M_{\mathrm{n'}+2}\cdot S}) \xrightarrow{\sigma} \uparrow\circ(\overline{\uparrow^{Suc(n')}\circ(M_1\cdot\ldots\cdot M_{\mathrm{n'}+2}\cdot S)}) \xrightarrow{\text{(I.H.)}\ \sigma^+} \uparrow\circ(\overline{M_{\mathrm{n'}+2}\cdot S}) = \uparrow\circ(\overline{M_{\mathrm{n'}+2}}\cdot\overline{S}) \xrightarrow{\sigma} \overline{S}$.

□

Lemma 4 *In* $\Lambda_{\mathcal{Q}}$, $\uparrow\circ\overline{\uparrow^n} \xrightarrow{\sigma^*} \overline{\uparrow^{Suc(n)}}$.

Proof. We reason by case analysis on n. □

Lemma 5 (Simulation of (ShiftCons")) $\overline{\uparrow^{Suc(n)+m}\circ(M_1\cdot\ldots\cdot M_{\mathrm{n}+1}\cdot\uparrow^{n'})} \xrightarrow{\sigma+} \overline{\uparrow^{m+n'}}$.

Proof. By structural induction on m.

- $m=0$. By Lemma 3, $\overline{\uparrow^{Suc(n)}\circ(M_1\cdot\ldots\cdot M_{\mathrm{n}+1}\cdot\uparrow^{n'})} \xrightarrow{\sigma^+} \overline{\uparrow^{n'}}$.

- $m = Suc(m')$. By definition, $\overline{\uparrow^{Suc(n)+Suc(m')} \circ (M_1 \cdot \ldots \cdot M_{n+1} \cdot \uparrow^{n'})} = (\uparrow \circ \overline{\uparrow^{Suc(n)+m'}}) \circ \overline{(M_1 \cdot \ldots \cdot M_{n+1} \cdot \uparrow^{n'})}$. But also, $(\uparrow \circ \overline{\uparrow^{Suc(n)+m'}}) \circ \overline{(M_1 \cdot \ldots \cdot M_{n+1} \cdot \uparrow^{n'})} \xrightarrow{\sigma} \uparrow \circ \overline{(\uparrow^{Suc(n)+m'} \circ (M_1 \cdot \ldots \cdot M_{n+1} \cdot \uparrow^{n'}))} \xrightarrow{\text{(I.H.) } \sigma^+} \uparrow \circ \overline{\uparrow^{m'+n'}} \xrightarrow{\text{(Lemma 4) } \sigma^*} \overline{\uparrow^{Suc(m'+n')}} = \overline{\uparrow^{Suc(m')+n}} = \overline{\uparrow^{m+n'}}$.

□

Lemma 6 (Simulation of (ShiftShift')) $\overline{\uparrow^{Suc(n)} \circ \uparrow^m} \xrightarrow{\sigma^*} \overline{\uparrow^{Suc(n+m)}}$.

Proof. By structural induction on n.

- $n = 0$. By definition, $\overline{\uparrow^{Suc(0)} \circ \uparrow^m} = \uparrow \circ \overline{\uparrow^m}$. But also, $\uparrow \circ \overline{\uparrow^m} \xrightarrow{\text{(Lemma 4) } \sigma^*} \overline{\uparrow^{Suc(m)}}$.
- $n = Suc(n')$. By definition, $\overline{\uparrow^{Suc(Suc(n'))} \circ \uparrow^m} = (\uparrow \circ \overline{\uparrow^{Suc(n')}}) \circ \overline{\uparrow^m}$. But also, $(\uparrow \circ \overline{\uparrow^{Suc(n')}}) \circ \overline{\uparrow^m} \xrightarrow{\sigma} \uparrow \circ (\overline{\uparrow^{Suc(n')} \circ \uparrow^m}) \xrightarrow{\text{(I.H.) } \sigma^*} \uparrow \circ \overline{\uparrow^{Suc(n'+m)}} = \uparrow \circ \overline{\uparrow^{n+m}} \xrightarrow{\text{(Lemma 4) } \sigma^*} \overline{\uparrow^{Suc(n+m)}}$.

□

Lemma 7 (Simulation of (ShiftS)) $\overline{\mathbf{1}[\uparrow^n] \cdot \uparrow^{Suc(n)}} \xrightarrow{\sigma^+} \overline{\uparrow^n}$.

Proof. By case analysis on n.

- $n = 0$. By definition, $\overline{\mathbf{1}[\uparrow^0] \cdot \uparrow^{Suc(0)}} = \mathbf{1}[id] \cdot \uparrow$. But also, $\mathbf{1} \cdot \uparrow \xrightarrow{\sigma} id = \overline{\uparrow^0}$.
- $n = Suc(n')$. By definition, $\overline{\mathbf{1}[\uparrow^{Suc(n')}] \cdot \uparrow^{Suc(Suc(n'))}} = \mathbf{1}[\overline{\uparrow^{Suc(n')}}] \cdot (\uparrow \circ \overline{\uparrow^{Suc(n')}})$. But also, $\mathbf{1}[\overline{\uparrow^{Suc(n')}}] \cdot (\uparrow \circ \overline{\uparrow^{Suc(n')}}) \xrightarrow{\sigma} \overline{\uparrow^{Suc(n')}} = \overline{\uparrow^n}$.

□

Now we show that $(\phi_2 \cup \phi_3)$-reductions are simulated via σ-reductions.

Lemma 8 *Let x and y be in $\Lambda_{\mathcal{Q}}$ such that $x \xrightarrow{(\phi_2 \cup \phi_3)} y$, then $\overline{x} \xrightarrow{\sigma^*} \overline{y}$.*

Proof. By induction on the depth of the redex reduced in x. At the base case x is a ϕ-redex. Lemma 3, Lemma 5, Lemma 6 and Lemma 7 prove the cases (ShiftCons'), (ShiftCons"), (ShiftShift') and (ShiftS). The other cases are obvious by definition of the transformation. At the induction step we have the following cases:

- $x = \lambda M$ and $y = \lambda M'$, with $M \xrightarrow{\phi} M'$. By definition, $\overline{x} = \lambda \overline{M}$ and $\overline{y} = \lambda \overline{M'}$. By induction hypothesis, $\overline{M} \xrightarrow{\sigma^*} \overline{M'}$ and so, $\overline{x} \xrightarrow{\sigma^*} \overline{y}$.
- $x = M\ N$ and $y = M'\ N$, with $M \xrightarrow{\phi} M'$. By definition, $\overline{x} = \overline{M}\ \overline{N}$ and $\overline{y} = \overline{M'}\ \overline{N}$. By induction hypothesis, $\overline{M} \xrightarrow{\sigma^*} \overline{M'}$ and so, $\overline{x} \xrightarrow{\sigma^*} \overline{y}$.
- The other cases are similar to the previous one.

□

Proposition 3 $(\phi_2 \cup \phi_3)$ *is terminating.*

Proof. Let x, y be in $\Lambda_{\mathcal{Q}}$. We say that $x < y$ if and only if $\overline{y} \xrightarrow{\sigma^+} \overline{x}$ or ($\overline{y} = \overline{x}$ and $[\![x]\!] < [\![y]\!]$). Notice that σ^+ is a well founded relation ([HL86],[Zan94]). We verify, by using Lemma 8, that the order $<$ is a reduction order for $(\phi_2 \cup \phi_3)$. □

We address the final condition of Lemma 2.

Proposition 4 ϕ_1 *commutes over* $(\phi_2 \cup \phi_3)$.

Proof. Assume that an arbitrary expression x in Λ_Q reduces in one ϕ_1-step to y and y reduces in one $(\phi_2 \cup \phi_3)$-step to z. We prove, by induction on the depth of the redex reduced in x, that there exists w such that $x \xrightarrow{(\phi_2 \cup \phi_3)} w \xrightarrow{\phi_1^*} z$. At the base case x is a ϕ_1-redex:

- (ShiftCons). We have the following sub-cases:
  - $x = \uparrow^{Suc(n)} \circ (M \cdot S) \xrightarrow{\text{(ShiftCons)}} \uparrow^n \circ S = y$ and $\uparrow^n \circ S \xrightarrow{\phi_2} \uparrow^n \circ S' = z$, with $S \xrightarrow{\phi_2} S'$. But also, $x = \uparrow^{Suc(n)} \circ (M \cdot S) \xrightarrow{\phi_2} \uparrow^{Suc(n)} \circ (M \cdot S') \xrightarrow{\text{(ShiftCons)}} \uparrow^n \circ S' = z$.
  - $x = \uparrow^{Suc(0)} \circ (M \cdot S) \xrightarrow{\text{(ShiftCons)}} \uparrow^0 \circ S = y$ and $\uparrow^0 \circ S \xrightarrow{\text{(IdS)}} S = z$. But also, $x = \uparrow^{Suc(0)} \circ (M \cdot S) \xrightarrow{\text{(ShiftCons')}} S = z$.
  - $x = \uparrow^{Suc(Suc(n))} \circ (M_1 \cdot \ldots \cdot M_{n+2} \cdot S) \xrightarrow{\text{(ShiftCons)}} \uparrow^{Suc(n)} \circ (M_2 \cdot \ldots \cdot M_{n+2} \cdot S) = y$ and $y \xrightarrow{\text{(ShiftCons')}} S = z$. But also, $x \xrightarrow{\text{(ShiftCons')}} S = z$.
  - $x = \uparrow^{Suc(Suc(n)+m)} \circ (M_1 \cdot \ldots \cdot M_{n+2} \cdot \uparrow^{m'}) \xrightarrow{\text{(ShiftCons)}} \uparrow^{Suc(n)+m} \circ (M_2 \cdot \ldots \cdot M_{n+2} \cdot \uparrow^{m'}) = y$ and $y \xrightarrow{\text{(ShiftCons")}} \uparrow^{m+m'} = z$. But also, $x = \uparrow^{Suc(Suc(n))+m} \circ (M_1 \cdot \ldots \cdot M_{n+2} \cdot \uparrow^{m'}) \xrightarrow{\text{(ShiftCons")}} \uparrow^{m+m'} = z$.
  - $x = \uparrow^{Suc(Suc(n))} \circ (M \cdot \uparrow^m) \xrightarrow{\text{(ShiftCons)}} \uparrow^{Suc(n)} \circ \uparrow^m = y$ and $\uparrow^{Suc(n)} \circ \uparrow^m \xrightarrow{\text{(ShiftShift')}} \uparrow^{Suc(n+m)} = z$. But also, $x = \uparrow^{Suc(0)+Suc(n)} \circ (M \cdot \uparrow^m) \xrightarrow{\text{(ShiftCons")}} \uparrow^{Suc(n+m)} = z$.
- (ShiftShift). We have the following sub-cases:
  - $x = \uparrow^{Suc(0)} \circ \uparrow^m \xrightarrow{\text{(ShiftShift)}} \uparrow^0 \circ \uparrow^{Suc(m)} = y$ and $\uparrow^0 \circ \uparrow^{Suc(m)} \xrightarrow{\text{(IdS)}} \uparrow^{Suc(m)} = z$. But also, $x = \uparrow^{Suc(0)} \circ \uparrow^m \xrightarrow{\text{(ShiftShift')}} \uparrow^{Suc(0+m)} = \uparrow^{Suc(m)} = z$.
  - $x = \uparrow^{Suc(Suc(n))} \circ \uparrow^m \xrightarrow{\text{(ShiftShift)}} \uparrow^{Suc(n)} \circ \uparrow^{Suc(m)} = y$ and $\uparrow^{Suc(n)} \circ \uparrow^{Suc(m)} \xrightarrow{\text{(ShiftShift')}} \uparrow^{Suc(n+Suc(m))} = z$. But also, $x \xrightarrow{\text{(ShiftShift')}} \uparrow^{Suc(Suc(n)+m)} = \uparrow^{n+Suc(m)} = z$.

At the induction step we have three cases:

- x is not ϕ_1-redex but y is a $(\phi_2 \cup \phi_3)$-redex. Notice that right-hand side of a ϕ_1-rule can never overlap with a strict sub-expression of a $(\phi_2 \cup \phi_3)$-redex. In this case is not difficult to verify that the diagram is closed in the following way: $x \xrightarrow{\phi_2} w \xrightarrow{\phi_1^*} z$. We show, for example, one case with the (App) rule: $x = (M\ N)[S] \xrightarrow{\phi_1} (M\ N)[S'] = y$, with $S \xrightarrow{\phi_1} S'$, and $y = (M\ N)[S'] \xrightarrow{\text{(App)}} M[S']\ N[S'] = z$. But also, $x = (M\ N)[S] \xrightarrow{\text{(App)}} M[S]\ N[S] \xrightarrow{\phi_1^*} M[S']\ N[S'] = z$. The rest of cases are similar.
- $x = C[x', y']$ [3] $\xrightarrow{\phi_1} C[x'', y'] = y$, with $x' \xrightarrow{\phi_1} x''$, and $y = C[x'', y'] \xrightarrow{\phi_2} C[x'', y''] = z$, with $y' \xrightarrow{\phi_1} y''$. But also, $x = C[x', y'] \xrightarrow{\phi_2} C[x', y''] \xrightarrow{\phi_1} C[x'', y''] = z$.
- $x = C[x'] \xrightarrow{\phi_1} C[y'] = y$, with $x' \xrightarrow{\phi_1} y'$, and $y = C[y'] \xrightarrow{\phi_2} C[z'] = z$, with $y' \xrightarrow{\phi_2} z'$. By induction hypothesis, $x' \xrightarrow{\phi_2} w \xrightarrow{\phi_1^*} z'$, therefore $x = C[x'] \xrightarrow{\phi_2} C[w] \xrightarrow{\phi_1^*} C[z'] = z$.

□

Theorem 1 *The system ϕ is terminating.*

[3] $C[x, y]$ and $C[w]$ are notations for context expressions that contain, respectively, two disjoint sub-expressions x and y, and a strict sub-expression w.

Proof. By Lemma 2, Proposition 2, Proposition 3 and Proposition 4, $(\phi_1 \cup \phi_2 \cup \phi_3)$ is terminating. Therefore, $\phi = (\phi_1 \cup \phi_2)$ is also terminating. □

Corollary 1 *Let x be in $\Lambda_{\mathcal{Q}}$, then x has at least one ϕ-normal form.*

4 ϕ-Confluence

The test for confluence of ϕ is much harder than the test for confluence of σ. Even if the two systems are terminating. In fact, ϕ is not locally confluent on open expressions and [KNO90] proves that local ground confluence cannot be mechanically verified. However, the Critical Pair's lemma ([Hue80]), i.e. a term rewriting system is locally confluent if its critical pairs are joinable, holds also for ground expressions ([KNO90]): a term rewriting system is locally confluent on ground terms if its critical pairs are ground-joinable.

We want to prove the confluence of ϕ on the set of quasi-open expressions, i.e. confluence of a many-sorted system with mixed open and ground sorts. The Critical Pair's Lemma is not true for general many-sorted systems, but [SS89] proposes a generalization for *sort compatible systems*. A many-sorted rewriting system is sort compatible if for every rule $l \longrightarrow r$, the sort of l and the sort of r are the same.

Lemma 9 (Critical Pair's Lemma) *Let R be a sort compatible rewriting system, R is locally confluent if every critical pair is joinable.*

Proof. [SS89]. □

Before proving ϕ-confluence, we show that critical pairs of ϕ are ϕ-joinable in the set of quasi-open expressions $\Lambda_{\mathcal{Q}}$.

Proposition 5 ((Id-Clos) Critical Pair) *In $\Lambda_{\mathcal{Q}}$,$M[S]$ and $M[S \circ \uparrow^0]$ are ϕ-joinable.*

Proof. It suffices to prove that S and $S \circ \uparrow^0$ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$. Let $\hat{S}$ any ϕ-normal form of S (Corollary 1), then $S \xrightarrow{\phi^*} \hat{S}$ and $S \circ \uparrow^0 \xrightarrow{\phi^*} \hat{S} \circ \uparrow^0$. Now we can prove by structural induction on $\hat{S}$ that $\hat{S} \circ \uparrow^0 \xrightarrow{\phi^*} \hat{S}$. □

Lemma 10 *In $\Lambda_{\mathcal{Q}}$, $(\uparrow^n \circ S) \circ T$ and $\uparrow^n \circ (S \circ T)$ are ϕ-joinable.*

Proof. We proceed by structural induction on n. □

Lemma 11 *If S is a ϕ-normal form, then $S \circ (T \circ T')$ and $(S \circ T) \circ T'$ are ϕ-joinable on $\Lambda_{\mathcal{Q}}$.*

Proof. By structural induction on S. By Proposition 1, we have the following cases:

- $S = M \cdot S'$. Then, $S \circ (T \circ T') \xrightarrow{\text{(Map)}} M[T \circ T'] \cdot (S' \circ (T \circ T'))$. But also, $(S \circ T) \circ T' \xrightarrow{\text{(Map;Map)}} M[T][T'] \cdot ((S' \circ T) \circ T') \xrightarrow{\text{(Clos)}} M[T \circ T'] \cdot ((S' \circ T) \circ T')$. By induction hypothesis, $S' \circ (T \circ T')$ and $(S' \circ T) \circ T'$ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$.
- $S = \uparrow^n$. By Lemma 10, $\uparrow^n \circ (T \circ T')$ and $(\uparrow^n \circ T) \circ T'$ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$.

□

Proposition 6 ((Clos-Clos) Critical Pair) *In $\Lambda_{\mathcal{Q}}$, $M[(S \circ T) \circ T']$ and $M[S \circ (T \circ T')]$ are ϕ-joinable.*

Proof. It suffices to prove that $(S \circ T) \circ T'$ and $S \circ (T \circ T')$ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$. Let $\hat{S}$ any ϕ-normal form of S (Corollary 1), then $(S \circ T) \circ T' \xrightarrow{\phi^*} (\hat{S} \circ T) \circ T'$ and $S \circ (T \circ T') \xrightarrow{\phi^*} \hat{S} \circ (T \circ T')$. By Lemma 11, these two last substitutions are ϕ-joinable in $\Lambda_{\mathcal{Q}}$. □

Proposition 7 ((Shift0-Map) Critical Pair) *In $\Lambda_{\mathcal{Q}}$, $\mathbf{1}[S] \cdot (\uparrow^{Suc(0)} \circ S)$ and S are ϕ-joinable.*

Proof. Let $\hat{S}$ any ϕ-normal form of S (Corollary 1). By Proposition 1, we have the following cases:

- $\hat{S} = \uparrow^0$. Then, $\mathbf{1}[S] \cdot (\uparrow^{Suc(0)} \circ S) \xrightarrow{\phi^*} \mathbf{1}[\uparrow^0] \cdot (\uparrow^{Suc(0)} \circ \uparrow^0) \xrightarrow{\text{(Id;ShiftShift;IdS)}} \mathbf{1} \cdot \uparrow^{Suc(0)} \xrightarrow{\text{(Shift0)}} \uparrow^0$. But also, $S \xrightarrow{\phi^*} \uparrow^0$.
- $\hat{S} = \uparrow^{Suc(n)}$. Then, $\mathbf{1}[S] \cdot (\uparrow^{Suc(0)} \circ S) \xrightarrow{\phi^*} \mathbf{1}[\uparrow^{Suc(n)}] \cdot (\uparrow^{Suc(0)} \circ \uparrow^{Suc(n)}) \xrightarrow{\text{(ShiftShift;IdS)}}$ $\mathbf{1}[\uparrow^{Suc(n)}] \cdot \uparrow^{Suc(Suc(n))} \xrightarrow{\text{(ShiftS)}} \uparrow^{Suc(n)}$. But also, $S \xrightarrow{\phi^*} \uparrow^{Suc(n)}$.
- $\hat{S} = M \cdot T$. Then, $\mathbf{1}[S] \cdot (\uparrow^{Suc(0)} \circ S) \xrightarrow{\phi^*} \mathbf{1}[M \cdot T] \cdot (\uparrow^{Suc(0)} \circ (M \cdot T)) \xrightarrow{\text{(VarCons;ShiftCons)}}$ $M \cdot (\uparrow^0 \circ T) \xrightarrow{\text{(IdS)}} M \cdot T$. But also, $S \xrightarrow{\phi^*} M \cdot T$.

□

Proposition 8 ((ShiftS-Map) Critical Pair) *In $\Lambda_{\mathcal{Q}}$, $\mathbf{1}[\uparrow^n \circ S] \cdot (\uparrow^{Suc(n)} \circ S)$ and $\uparrow^n \circ S$ are ϕ-joinable.*

Proof. By structural induction on n.

- $n = 0$. Then, $\mathbf{1}[\uparrow^n \circ S] \cdot (\uparrow^{Suc(n)} \circ S) \xrightarrow{\text{(IdS)}} \mathbf{1}[S] \cdot (\uparrow^{Suc(0)} \circ S)$. But also, $\uparrow^n \circ S \xrightarrow{\text{(IdS)}} S$. By Proposition 7, $\mathbf{1}[S] \cdot (\uparrow^{Suc(0)} \circ S)$ and S are ϕ-joinable in $\Lambda_{\mathcal{Q}}$.
- $n = Suc(n')$. Let $\hat{S}$ any ϕ-normal form of S (Corollary 1). By Proposition 1, we have the following cases:
  - $\hat{S} = \uparrow^m$. Then, $\mathbf{1}[\uparrow^n \circ S] \cdot (\uparrow^{Suc(n)} \circ S) \xrightarrow{\phi^*} \mathbf{1}[\uparrow^n \circ \uparrow^m] \cdot (\uparrow^{Suc(n)} \circ \uparrow^m)$. By Lemma 1, $\mathbf{1}[\uparrow^n \circ \uparrow^m] \cdot (\uparrow^{Suc(n)} \circ \uparrow^m) \xrightarrow{\phi^*} \mathbf{1}[\uparrow^{n+m}] \cdot \uparrow^{Suc(n)+m}$. By Def. 1, $Suc(n) + m = Suc(n + m)$, then $\mathbf{1}[\uparrow^{n+m}] \cdot \uparrow^{Suc(n+m)} \xrightarrow{\text{(ShiftS)}} \uparrow^{n+m}$. But also, $\uparrow^n \circ S \xrightarrow{\phi^*} \uparrow^{n+m}$.
  - $\hat{S} = M \cdot T$. Then, $\mathbf{1}[\uparrow^n \circ S] \cdot (\uparrow^{Suc(n)} \circ S) \xrightarrow{\phi^*} \mathbf{1}[\uparrow^{Suc(n')} \circ (M \cdot T)] \cdot (\uparrow^{Suc(Suc(n'))} \circ (M \cdot T))$ $\xrightarrow{\text{(ShiftCons;ShiftCons)}} \mathbf{1}[\uparrow^{n'} \circ T] \cdot (\uparrow^{Suc(n')} \circ T)$. But also, $\uparrow^n \circ S \xrightarrow{\phi^*} \uparrow^{n'} \circ T$. By induction hypothesis, $\uparrow^{n'} \circ T$ and $\mathbf{1}[\uparrow^{n'} \circ T] \cdot (\uparrow^{Suc(n')} \circ T)$ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$.

□

Lemma 12 *If S is a ϕ-normal form, then $(S \circ \uparrow^{Suc(0)}) \circ (M \cdot (T \circ T'))$ and $(S \circ T) \circ T'$ are ϕ-joinable on $\Lambda_{\mathcal{Q}}$.*

Proof. We proceed by structural induction on S. Take $T'' = M \cdot (T \circ T')$, by Proposition 1 we have the following cases:

- $S = M' \cdot S'$. Then, $(S \circ \uparrow^{Suc(0)}) \circ T'' \xrightarrow{\text{(Map;Map)}} M'[\uparrow^{Suc(0)}][T''] \cdot ((S' \circ \uparrow^{Suc(0)}) \circ T'')$ $\xrightarrow{\text{(Clos;ShiftCons;IdS)}} M'[T \circ T'] \cdot ((S' \circ \uparrow^{Suc(0)}) \circ T'')$. But also, $(S \circ T) \circ T' \xrightarrow{\phi^*}$ $M'[T \circ T'] \cdot ((S' \circ T) \circ T')$. By induction hypothesis, $(S' \circ \uparrow^{Suc(0)}) \circ T''$ and $(S' \circ T) \circ T'$ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$.
- $S = \uparrow^n$. By Lemma 1 and Lemma 1, $(S \circ \uparrow^{Suc(0)}) \circ T'' \xrightarrow{\phi^*} \uparrow^{Suc(n)} \circ T'' \xrightarrow{\text{(ShiftCons)}} \uparrow^n \circ (T \circ T')$. By Lemma 10, $\uparrow^n \circ (T \circ T')$ and $(\uparrow^n \circ T) \circ T'$ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$.

□

Proposition 9 ((Lambda-Clos) Critical Pair) *In $\Lambda_{\mathcal{Q}}$, $\lambda M[\mathbf{1} \cdot ((S \circ \uparrow^{Suc(0)}) \circ (\mathbf{1} \cdot (T \circ \uparrow^{Suc(0)})))]$ and $\lambda M[\mathbf{1} \cdot ((S \circ T) \circ \uparrow^{Suc(0)})]$ are ϕ-joinable.*

Proof. It suffices to prove that $(S \circ \uparrow^{Suc(0)}) \circ (\mathbf{1} \cdot (T \circ \uparrow^{Suc(0)}))$ and $(S \circ T) \circ \uparrow^{Suc(0)}$ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$. Let $\hat{S}$ any ϕ-normal form of S (Corollary 1). Then, $(S \circ \uparrow^{Suc(0)}) \circ (\mathbf{1} \cdot (T \circ \uparrow^{Suc(0)})) \xrightarrow{\phi^*}$ $(\hat{S} \circ \uparrow^{Suc(0)}) \circ (\mathbf{1} \cdot (T \circ \uparrow^{Suc(0)}))$, and $(S \circ T) \circ \uparrow^{Suc(0)} \xrightarrow{\phi^*} (\hat{S} \circ T) \circ \uparrow^{Suc(0)}$. By Lemma 12, these two last substitutions are ϕ-joinable in $\Lambda_{\mathcal{Q}}$. □

The following lemma is used in the next section to prove that λ_ϕ is confluent on $\Lambda_{\mathcal{Q}}$. It is proved by structural induction on S in the same way that Lemma 12.

Lemma 13 *Let $(S \circ \uparrow^{Suc(0)}) \circ (M \cdot \uparrow^0)$ and S be two substitutions in $\Lambda_{\mathcal{Q}}$. If S is a ϕ-normal form, then $(S \circ \uparrow^{Suc(0)}) \circ (M \cdot \uparrow^0) \xrightarrow{\phi^*} S$.*

Proposition 10 *ϕ is locally confluent on $\Lambda_{\mathcal{Q}}$.*

Proof. First, λ_ϕ is a sort compatible system, i.e. terms reduce to terms and substitutions reduce to substitutions; and second, critical pairs of ϕ are ϕ-joinable in $\Lambda_{\mathcal{Q}}$ (Proposition 5, Proposition 6, Proposition 7, Proposition 8 and Proposition 9). The proof follows straightforwardly the proof of Lemma 9 in [SS89]. Remark that if two expressions are Λ_ϕ-joinable, then they are in particular $\Lambda_{\mathcal{Q}}$-joinable. Then it is sufficient to concentrate on those critical pairs of R that are not joinable. □

Theorem 2 *ϕ is confluent in $\Lambda_{\mathcal{Q}}$.*

Proof. By Theorem 1, ϕ is terminating and by Proposition 10, ϕ is locally confluent on $\Lambda_{\mathcal{Q}}$, so by Newman's Lemma, ϕ is confluent. □

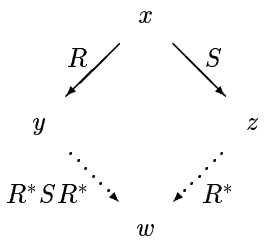
Corollary 2 *Let x be in $\Lambda_{\mathcal{Q}}$, then x has exactly one ϕ-normal form.*

Proof. By Corollary 1 and Theorem 2. □

5 λ_ϕ-Confluence

An useful technique to prove confluence in explicit substitutions calculi is the *interpretation method* ([Har89]). Based on this method, [Kes96] proposes a general scheme of abstract properties which are sufficient to guarantee ground confluence on explicit substitutions calculi. Although the interpretation method can be used to prove confluence on terms with meta-variables ([Río93]), we use a technique that was coined in [YH88]: the Yokouchi-Hikita's Lemma. This lemma is used in [CHL96] to prove confluence of $\lambda_{\Uparrow}$, and it turns out that it is more direct than the interpretation method for left-linear calculi of explicit substitutions ([Pag96]).

Lemma 14 (Yokouchi-Hikita's Lemma) *Let R and S be two relations defined on a set X such that: 1. R is confluent and terminating, 2. S is strongly confluent and 3. S and R commute in the following way, for any $x, y, z \in X$, if $x \xrightarrow{R} y$ and $x \xrightarrow{S} z$, then there exists $w \in X$ such that $y \xrightarrow{R^* S R^*} w$ and $z \xrightarrow{S} w$, i.e. the following diagram holds:*



Then the relation $R^* S R^*$ is confluent.

Proof. [CHL96]. □

We take the set $\Lambda_{\mathcal{Q}}$ as X, ϕ as R and $B_{\|}$ as S, where $B_{\|}$ is the parallelization of (Beta) defined by:

$$\frac{}{x \longrightarrow x}\,(\mathrm{Refl}_{\|}) \qquad \frac{M \longrightarrow N}{\lambda M \longrightarrow \lambda N}\,(\mathrm{Lambda}_{\|})$$

$$\frac{M \longrightarrow M' \qquad N \longrightarrow N'}{M\ N \longrightarrow M'\ N'}\,(\mathrm{App}_{\|}) \qquad \frac{M \longrightarrow N \qquad S \longrightarrow T}{M[S] \longrightarrow N[T]}\,(\mathrm{Clos}_{\|})$$

$$\frac{M \longrightarrow N \qquad S \longrightarrow T}{M \cdot S \longrightarrow N \cdot T}\,(\mathrm{Cons}_{\|}) \qquad \frac{S \longrightarrow S' \qquad T \longrightarrow T'}{S \circ T \longrightarrow S' \circ T'}\,(\mathrm{Comp}_{\|})$$

$$\frac{M \longrightarrow M' \qquad N \longrightarrow N'}{(\lambda M)\ N \longrightarrow M'[N' \cdot \uparrow^0]}\,(\mathrm{Beta}_{\|})$$

Proposition 11 *In $\Lambda_{\mathcal{Q}}$, ϕ and $B_{\|}$ satisfy the conditions of Lemma 14. Therefore, $\phi^* B_{\|} \phi^*$ is confluent on $\Lambda_{\mathcal{Q}}$.*

Proof. (1) By Theorem 1 and Theorem 2, ϕ is terminating and confluent on $\Lambda_{\mathcal{Q}}$; (2) $B_{\|}$ is strongly confluent, since (Beta) by itself is a left-linear system with no critical pairs ([Hue80]); and (3), assume that an arbitrary expression x in $\Lambda_{\mathcal{Q}}$ reduces in one ϕ-step to y, and in one $B_{\|}$-step to z. We prove, by induction on the depth of the ϕ-redex reduced in x, that there exists w such that $y \xrightarrow{\phi^* B_{\|} \phi^*} w$ and $z \xrightarrow{\phi^*} w$. At the base case x is a ϕ-redex:

- (App). There are two cases:
  - $x = (M\ N)[S] \xrightarrow{(\mathrm{App})} M[S]\ N[S] = y$ and $(M\ N)[S] \xrightarrow{B_{\|}} (M'\ N')[S'] = z$, with $M \xrightarrow{B_{\|}} M'$, $N \xrightarrow{B_{\|}} N'$ and $S \xrightarrow{B_{\|}} S'$. By definition of $B_{\|}$, $M[S]\ N[S] \xrightarrow{B_{\|}} M'[S']\ N'[S'] = w$. But also, $(M'\ N')[S'] \xrightarrow{(\mathrm{App})} M'[S']\ N'[S'] = w$.
  - $x = ((\lambda M)\ N)[S] \xrightarrow{(\mathrm{App})} (\lambda M)[S]\ N[S] = y$ and $((\lambda M)\ N)[S] \xrightarrow{B_{\|}} M'[N' \cdot \uparrow^0][S'] = z$, with $M \xrightarrow{B_{\|}} M'$, $N \xrightarrow{B_{\|}} N'$ and $S \xrightarrow{B_{\|}} S'$. Let $\hat{S}'$ the ϕ-normal form of S' (Corollary 2). Then, $y = (\lambda M)[S]\ N[S] \xrightarrow{(\mathrm{Lambda})} (\lambda M[\mathbf{1} \cdot (S \circ \uparrow^{Suc(0)})])\ N[S] \xrightarrow{B_{\|}} M'[\mathbf{1} \cdot (S' \circ \uparrow^{Suc(0)})][N'[S'] \cdot \uparrow^0]$ $\xrightarrow{(\mathrm{Clos;Map;VarCons})} M'[N'[S'] \cdot ((S' \circ \uparrow^{Suc(0)}) \circ (N'[S'] \cdot \uparrow^0))] \xrightarrow{\phi^*}$ $M'[N'[\hat{S}'] \cdot ((\hat{S}' \circ \uparrow^{Suc(0)}) \circ (N'[\hat{S}'] \cdot \uparrow^0))] \xrightarrow{(\mathrm{Lemma\ 13})\ \phi^*} M'[N'[\hat{S}'] \cdot \hat{S}']$. But also, $M'[N' \cdot \uparrow^0][S']$ $\xrightarrow{(\mathrm{Clos;Map;IdS})} M'[N'[S'] \cdot S'] \xrightarrow{\phi^*} M'[N'[\hat{S}'] \cdot \hat{S}']$. This case is the only interesting one.
- (Lambda). $x = (\lambda M)[S] \xrightarrow{(\mathrm{Lambda})} \lambda M[\mathbf{1} \cdot (S \circ \uparrow^{Suc(0)})] = y$ and $x = (\lambda M)[S] \xrightarrow{B_{\|}} (\lambda M')[S'] = z$, with $M \xrightarrow{B_{\|}} M'$ and $S \xrightarrow{B_{\|}} S'$. By definition of $B_{\|}$, $\lambda M[\mathbf{1} \cdot (S \circ \uparrow^{Suc(0)})] \xrightarrow{B_{\|}} \lambda M'[\mathbf{1} \cdot (S' \circ \uparrow^{Suc(0)})]$ $= w$. But also, $(\lambda M')[S'] \xrightarrow{(\mathrm{Lambda})} \lambda M'[\mathbf{1} \cdot (S' \circ \uparrow^{Suc(0)})] = w$.
- The other cases are similar to the previous one.

At the induction step we have three cases:

- The ϕ-redex is a sub-expression of the $B_{\|}$-redex. Remark that a strict sub-expression of a $B_{\|}$-redex can never overlap with a ϕ-redex. This means that $B_{\|}$-redexes are preserved before internal ϕ-reductions. In this case the diagram is closed in the following way: $y \xrightarrow{B_{\|}} w$ and $z \xrightarrow{\phi^*} w$.

- The ϕ-redex and the $B_{\|}$-redex are disjoint. In this case, $x = C[x', y'] \xrightarrow{\phi} C[x'', y'] = y$, with $x' \xrightarrow{\phi} x''$, and $x = C[x', y'] \xrightarrow{B_{\|}} C[x', y''] = z$, with $y' \xrightarrow{B_{\|}} y''$. But also, $y = C[x'', y'] \xrightarrow{B_{\|}} C[x'', y'']$ $= w$ and $z = C[x', y''] \xrightarrow{\phi} C[x'', y''] = w$.

- The $B_{\|}$-redex is a sub-expression of the ϕ redex. In this case, $x = C[x'] \xrightarrow{\phi} C[y'] = y$, with $x' \xrightarrow{\phi} y'$, and $C[x'] \xrightarrow{B_{\|}} C[z']$. By induction hypothesis, $y' \xrightarrow{\phi^* B_{\|} \phi^*} w'$ and $z' \xrightarrow{\phi^*} w'$, therefore $C[y'] \xrightarrow{\phi^* B_{\|} \phi^*} C[w'] = w$ and $C[z'] \xrightarrow{\phi^*} C[w'] = w$.

□

Theorem 3 (Confluence) *λ_ϕ is confluent on $\Lambda_{\mathcal{Q}}$.*

Proof. Notice that $\lambda_\phi \subseteq \phi^* B_{\|} \phi^* \subseteq {\lambda_\phi}^*$. Let x be in $\Lambda_{\mathcal{Q}}$. If $x \xrightarrow{{\lambda_\phi}^*} y$ and $x \xrightarrow{{\lambda_\phi}^*} z$, then by Proposition 11, there exist w such that $y \xrightarrow{(\phi^* B_{\|} \phi^*)^*} w$ and $z \xrightarrow{(\phi^* B_{\|} \phi^*)^*} w$. So, $y \xrightarrow{{\lambda_\phi}^*} w$ and $z \xrightarrow{{\lambda_\phi}^*} w$. □

6 The simply typed version

We consider a simply-typed theory, where types are generated from a set of basic types and the arrow "$\rightarrow$" type constructor. Without lost of generality we take only one basic type named ι. Like the simply-typed λ-calculus in de Bruijn's notation, typing contexts (of free variables) are structured as lists of types.

The grammar of terms and contexts is:

$$\begin{array}{llll} \textbf{Types} & A, B & ::= & \iota \mid A \rightarrow B \\ \textbf{Contexts} & \Gamma & ::= & nil \mid A.\Gamma \end{array}$$

Typed terms differ from untyped ones only in abstraction expressions. We prefer a *Church style* notation where types of abstracted variables appear explicitly in the syntax.

$$\textbf{Terms} \quad M, N \quad ::= \quad \ldots \mid \lambda_A.M \mid \ldots$$

The λ_ϕ-calculus is modified according to this new syntax of abstractions. However, it is not difficult to see that properties of Section 2 and 3 are preserved.

Typing rules are inspired from those of λ_σ with open terms ([DHK95]):

$$\frac{}{A.\Gamma \vdash \mathbf{1} : A}\,(\text{Var}) \qquad \frac{A.\Gamma \vdash M : B}{\Gamma \vdash \lambda_A.M : A \rightarrow B}\,(\text{Lambda})$$

$$\frac{\Gamma \vdash M : A \rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash (M\ N) : B}\,(\text{App}) \qquad \frac{\Gamma \vdash S \triangleright \Delta \quad \Delta \vdash M : A}{\Gamma \vdash M[S] : A}\,(\text{Clos})$$

$$\frac{}{\Gamma_X \vdash X : A_X}\,(\text{Metavar})$$

$$\frac{}{\Gamma \vdash \uparrow^0 \triangleright \Gamma}\,(\text{Id}) \qquad \frac{\Gamma \vdash \uparrow^n \triangleright \Delta}{A.\Gamma \vdash \uparrow^{Suc(n)} \triangleright \Delta}\,(\text{Shift})$$

$$\frac{\Gamma \vdash S \triangleright \Delta' \quad \Delta' \vdash T \triangleright \Delta}{\Gamma \vdash T \circ S \triangleright \Delta}\,(\text{Comp}) \qquad \frac{\Gamma \vdash M : A \quad \Gamma \vdash S \triangleright \Delta}{\Gamma \vdash M \cdot S \triangleright A.\Delta}\,(\text{Cons})$$

Lemma 15 (Inversion) *If $\Gamma \vdash M : A$ and $\Gamma \vdash S \triangleright \Delta$, then*

1. *if* $M = \mathbf{1}$, *then* $\Gamma = A.\Gamma'$;
2. *if* $M = X$ *(a meta-variable), then* $\Gamma = \Gamma_X$ *and* $A = A_X$;
3. *if* $M = \lambda_{A'}.M'$, *then* $A'.\Gamma \vdash M' : B'$ *and* $A = A' \to B'$;
4. *if* $M = (M'\ N')$, *then* $\Gamma \vdash M' : B \to A$ *and* $\Gamma \vdash N' : B$;
5. *if* $M = M'[T]$, *then* $\Gamma \vdash T \triangleright \Delta'$ *and* $\Delta' \vdash M' : A$;
6. *if* $S = \uparrow^0$, *then* $\Gamma = \Delta$;
7. *if* $S = \uparrow^{Suc(n)}$, *then* $\Gamma = A'.\Gamma'$ *and* $\Gamma' \vdash \uparrow^n \triangleright \Delta$;
8. *if* $S = M' \cdot T$, *then* $\Gamma \vdash M' : A'$, $\Gamma \vdash T \triangleright \Delta'$ *and* $\Delta = A'.\Delta'$; *and*
9. *if* $S = S' \circ T$, *then* $\Gamma \vdash T \triangleright \Delta'$, $\Delta' \vdash S' \triangleright \Delta$.

Proof. By induction on proof derivation. Notice that typing rules are structural, i.e. there is one rule for each constructor of terms and substitutions. □

Theorem 4 (Uniqueness of types)

1. *If* $\Gamma_1 \vdash M : A_1$ *and* $\Gamma_2 \vdash M : A_2$, *then* $A_1 = A_2$.
2. *If* $\Gamma_1 \vdash S \triangleright \Delta_1$ *and* $\Gamma_2 \vdash S \triangleright \Delta_2$, *then* $\Delta_1 = \Delta_2$.

Proof. By simultaneous structural induction on M and S. □

6.1 λ_ϕ-Subject Reduction

Lemma 16 (Beta-redex) *If* $\Gamma \vdash (\lambda_B.M)\ N : A$, *then* $\Gamma \vdash M[N \cdot \uparrow^0] : A$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash \lambda_B.M : B \to A$ and (b) $\Gamma \vdash N : B$, by Lemma 15 applied to hypothesis of lemma.
2. $B.\Gamma \vdash M : A$, by Lemma 15 applied to (1-a).
3. $\Gamma \vdash \uparrow^0 \triangleright \Gamma$, by the (Id) rule.
4. $\Gamma \vdash N \cdot \uparrow^0 \triangleright B.\Gamma$, by the (Cons) rule applied to (1-b) and (3).

We conclude with the (Clos) rule applied to (2) and (4) that $\Gamma \vdash M[N \cdot \uparrow^0] : A$. □

Lemma 17 (Lambda-redex) *If* $\Gamma \vdash (\lambda_B.M)[S] : A$, *then* $\Gamma \vdash \lambda_B.M[\mathbf{1} \cdot (S \circ \uparrow^{Suc(0)})] : A$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash S \triangleright \Delta$ and (b) $\Delta \vdash \lambda_B.M : A$, by Lemma 15 applied to hypothesis of lemma.
2. (a) $B.\Delta \vdash M : A'$ and (b) $A = B \to A'$, by Lemma 15 applied to (1-b).
3. $\Gamma \vdash \uparrow^0 \triangleright \Gamma$, by the (Id) rule.
4. $B.\Gamma \vdash \uparrow^{Suc(0)} \triangleright \Gamma$, by the (Shift) rule applied to (3).
5. $B.\Gamma \vdash \mathbf{1} : B$, by the (Var) rule.
6. $B.\Gamma \vdash S \circ \uparrow^{Suc(0)} \triangleright \Delta$, by the (Comp) rule applied to (1-a) and (4).

7. $B.\Gamma \vdash \mathbf{1} \cdot (S \circ \uparrow^{Suc(0)}) \triangleright B.\Delta$, by the (Cons) rule applied to (5) and (6).
8. $B.\Gamma \vdash M[\mathbf{1} \cdot (S \circ \uparrow^{Suc(0)})] : A'$, by the (Clos) rule applied to (2-a) and (7).

We conclude with the (Lambda) rule applied to (8) that $\Gamma \vdash \lambda_B.M[\mathbf{1} \cdot (S \circ \uparrow^{Suc(0)})] : B \to A'$. □

Lemma 18 (App-redex) *If* $\Gamma \vdash (M\ N)[S] : A$, *the* $\Gamma \vdash (M[S]\ N[S]) : A$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash S \triangleright \Delta$ and (b) $\Delta \vdash (M\ N) : A$, by Lemma 15 applied to hypothesis of lemma.
2. (a) $\Delta \vdash M : B \to A$ and (b) $\Delta \vdash N : B$, by Lemma 15 applied to (1-b).
3. $\Gamma \vdash M[S] : B \to A$, by the (Clos) rule applied to (1-a) and (2-a).
4. $\Gamma \vdash N[S] : B$, by the (Clos) rule applied to (1-a) and (2-b).

We conclude with the (App) rule applied to (3) and (4) that $\Gamma \vdash (M[S]\ N[S]) : A$. □

Lemma 19 (Clos-redex) *If* $\Gamma \vdash M[S][T] : A$, *then* $\Gamma \vdash M[S \circ T] : A$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash T \triangleright \Delta$ and (b) $\Delta \vdash M[S] : A$, by Lemma 15 applied to hypothesis of lemma.
2. (a) $\Delta \vdash S \triangleright \Delta'$ and (b) $\Delta' \vdash M : A$, by Lemma 15 applied to (1-b).
3. $\Gamma \vdash S \circ T \triangleright \Delta'$, by the (Clos) rule applied to (1-a) and (2-a).

We conclude with the (Clos) rule applied to (2-b) and (3) that $\Gamma \vdash M[S \circ T] : A$. □

Lemma 20 (VarCons-redex) *If* $\Gamma \vdash \mathbf{1}[M \cdot S] : A$, *then* $\Gamma \vdash M : A$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash M \cdot S \triangleright \Delta$ and (b) $\Delta \vdash \mathbf{1} : A$, by Lemma 15 applied to hypothesis of lemma.
2. $\Delta = A.\Delta'$, by Lemma 15 applied to (1-b).

We conclude with Lemma 15 applied to (1-a) and (2) that $\Gamma \vdash M : A$. □

Lemma 21 (Id-redex) *If* $\Gamma \vdash M[\uparrow^0] : A$, *then* $\Gamma \vdash M : A$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash \uparrow^0 \triangleright \Delta$ and (b) $\Delta \vdash M : A$, by Lemma 15 applied to hypothesis of lemma.
2. $\Delta = \Gamma$, by Lemma 15 applied to (1-a).

We conclude with (1-b) and (2) that $\Gamma \vdash M : A$. □

Lemma 22 (Map-redex) *If* $\Gamma \vdash (M \cdot S) \circ T \triangleright \Delta$, *then* $\Gamma \vdash M[T] \cdot (S \circ T) \triangleright \Delta$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash T \triangleright \Delta'$ and (b) $\Delta' \vdash M \cdot S \triangleright \Delta$, by Lemma 15 applied to hypothesis of lemma.
2. (a) $\Delta' \vdash M : A$, (b) $\Delta' \vdash S \triangleright \Delta''$ and (c) $\Delta = A.\Delta''$, by Lemma 15 applied to (1-b).
3. $\Gamma \vdash M[T] : A$, by the (Clos) rule applied to (1-a) and (2-a).

4. $\Gamma \vdash S \circ T \triangleright \Delta''$, by the (Comp) rule applied to (1-a) and (2-b).

We conclude with the (Cons) rule applied to (3) and (4) that $\Gamma \vdash M[T] \cdot (S \circ T) \triangleright A \cdot \Delta''$. □

Lemma 23 (IdS-redex) *If* $\Gamma \vdash \uparrow^0 \circ S \triangleright \Delta$, *then* $\Gamma \vdash S \triangleright \Delta$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash S \triangleright \Delta'$ and (b) $\Delta' \vdash \uparrow^0 \triangleright \Delta$, by Lemma 15 applied to hypothesis of lemma.
2. $\Delta' = \Delta$, by Lemma 15 applied to (1-b).

We conclude with (1-a) and (2) that $\Gamma \vdash S \triangleright \Delta$. □

Lemma 24 (ShiftCons-redex) *If* $\Gamma \vdash \uparrow^{Suc(n)} \circ (M \cdot S) \triangleright \Delta$, *then* $\Gamma \vdash \uparrow^n \circ S \triangleright \Delta$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash M \cdot S \triangleright \Delta'$ and (b) $\Delta' \vdash \uparrow^{Suc(n)} \triangleright \Delta$, by Lemma 15 applied to hypothesis of lemma.
2. (a) $\Gamma \vdash M : A$, (b) $\Gamma \vdash S \triangleright \Delta''$, (c) $\Delta' = A.\Delta''$, by Lemma 15 applied to (1-a).
3. $\Delta'' \vdash \uparrow^n \triangleright \Delta$, by Lemma 15 applied to (1-b) and (2-c).

We conclude with the (Comp) rule applied to (2-b) and (3) that $\Gamma \vdash \uparrow^n \circ S \triangleright \Delta$. □

Lemma 25 *If* $\Gamma \vdash \uparrow^n \triangleright A.\Delta$, *then* $\Gamma \vdash \uparrow^{Suc(n)} \triangleright \Delta$.

Proof. By structural induction on n.

- $n = 0$. We have the following hypothesis:
  1. $\Gamma = A.\Delta$, by Lemma 15 applied to hypothesis.
  2. $\Delta \vdash \uparrow^0 \triangleright \Delta$, by the (Id) rule.
  3. $A.\Delta \vdash \uparrow^{Suc(0)} \triangleright \Delta$, by the (Shift) rule applied to (2).

  We conclude with (1) and (3) that $\Gamma \vdash \uparrow^{Suc(0)} \triangleright \Delta$.
- $n = Suc(m)$. We have the following hypothesis:
  1. (a) $\Gamma = B.\Gamma'$ and (b) $\Gamma' \vdash \uparrow^m \triangleright A.\Delta$, by Lemma 15 applied to hypothesis.
  2. $\Gamma' \vdash \uparrow^{Suc(m)} \triangleright \Delta$, by induction hypothesis applied to (1-b).

  We conclude with the (Shift) rule applied to (2) that $B.\Gamma' \vdash \uparrow^{Suc(Suc(m))} \triangleright \Delta$, i.e. $\Gamma \vdash \uparrow^{Suc(n)} \triangleright \Delta$.

□

Lemma 26 (ShiftShift-redex) *If* $\Gamma \vdash \uparrow^{Suc(n)} \circ \uparrow^m \triangleright \Delta$, *then* $\Gamma \vdash \uparrow^n \circ \uparrow^{Suc(m)} \triangleright \Delta$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash \uparrow^m \triangleright \Delta'$ and (b) $\Delta' \vdash \uparrow^{Suc(n)} \triangleright \Delta$, by Lemma 15 applied to hypothesis of lemma.
2. (a) $\Delta' = A.\Delta''$ and (b) $\Delta'' \vdash \uparrow^n \triangleright \Delta$, by Lemma 15 applied to (1-b).
3. $\Gamma \vdash \uparrow^{Suc(m)} \triangleright \Delta''$, by Lemma 25 applied to (1-a) and (2-a).

We conclude with the (Comp) rule applied to (2-b) and (3) that $\Gamma \vdash \uparrow^n \circ \uparrow^{Suc(m)} \triangleright \Delta$. □

Lemma 27 (Shift0-redex) *If* $\Gamma \vdash \mathbf{1} \cdot \uparrow^{Suc(0)} \triangleright \Delta$, *then* $\Gamma \vdash \uparrow^0 \triangleright \Delta$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash \mathbf{1} : A$, (b) $\Gamma \vdash \uparrow^{Suc(0)} \triangleright \Delta'$ and (c) $\Delta = A.\Delta'$, by Lemma 15 applied to hypothesis of lemma.
2. $\Gamma = A.\Gamma'$, by Lemma 15 applied to (1-a).
3. $\Gamma' \vdash \uparrow^0 \triangleright \Delta'$, by Lemma 15 applied to (1-b) and (2).
4. $\Gamma' = \Delta'$, by Lemma 15 applied to (3).
5. $\Gamma = \Delta$, by (1-c), (2) and (4).
6. $\Gamma \vdash \uparrow^0 \triangleright \Gamma$, by the (Id) rule.

We conclude with (5) and (6) that $\Gamma \vdash \uparrow^0 \triangleright \Delta$. □

Lemma 28 (ShiftS-redex) *If* $\Gamma \vdash \mathbf{1}[\uparrow^n] \cdot \uparrow^{Suc(n)} \triangleright \Delta$, *then* $\Gamma \vdash \uparrow^n \triangleright \Delta$.

Proof. We have the following hypothesis:

1. (a) $\Gamma \vdash \mathbf{1}[\uparrow^n] : A$, (b) $\Gamma \vdash \uparrow^{Suc(n)} \triangleright \Delta'$ and (c) $\Delta = A.\Delta'$, by Lemma 15 applied to hypothesis of lemma.
2. (a) $\Gamma \vdash \uparrow^n \triangleright \Delta''$ and (b) $\Delta'' \vdash \mathbf{1} : A$, by Lemma 15 applied to (1-a).
3. $\Delta'' = A.\Omega$, by Lemma 15 applied to (2-b).
4. $\Gamma \vdash \uparrow^{Suc(n)} \triangleright \Omega$, by Lemma 25 applied to (2-a) and (3).
5. $\Omega = \Delta'$, by Theorem 4 applied to (1-b) and (4).
6. $\Delta = \Delta''$, by (1-c), (3) and (5).

We conclude with (2-a) and (6) that $\Gamma \vdash \uparrow^n \triangleright \Delta$. □

Theorem 5 (Subject Reduction) *Let* x *and* y *be in* $\Lambda_{\mathcal{Q}}$ *such that* $x \xrightarrow{\lambda_\phi{}^*} y$, *then*

- *if* x *is a Term and* $\Gamma \vdash x : A$, *then* $\Gamma \vdash y : A$; *and*
- *if* x *is a Substitution and* $\Gamma \vdash x \triangleright \Delta$, *then* $\Gamma \vdash y \triangleright \Delta$.

Proof. We show that typing is preserved for one-step reductions (i.e. $\xrightarrow{\lambda_\phi}$), and then it is also for the reflexive and transitive closure (i.e. $\xrightarrow{\lambda_\phi{}^*}$). Let $x \xrightarrow{R} y$ be a one-step reduction, we proceed by induction on the depth of the redex reduced in x. At the initial case x is reduced at the top level, and we conclude with Lemma 16, Lemma 17, Lemma 18, Lemma 19, Lemma 20, Lemma 21, Lemma 22, Lemma 23, Lemma 24, Lemma 26, Lemma 27 and Lemma 28. At the induction step we resolve with Lemma 15 and induction hypothesis. □

6.2 λ_ϕ-Weak Normalization

Strong normalization of typed terms does not hold for λ_ϕ. In fact, as shown by [Mel95], λ_σ does not preserve strong normalization of λ-calculus. The same counterexample may be reproduced with some minor modifications in explicit substitutions calculi as $\lambda_{\Uparrow}$ and λ_ϕ.

Nevertheless, we prove that λ_ϕ is weakly normalizing on typed expressions, i.e. there exists a strategy to find λ_ϕ-normal forms in typed expressions. In particular, we show that for ϕ-normal forms, the reduction of (Beta) followed by a ϕ-normalization is strongly normalizing on typed expressions.

The proof we provide can be adapted to λ_σ in a straightforward way. This gives an alternative proof to that independently developed by [GL97]. That proof is based on a clever translation to simply-typed λ-calculus. In contrast, we follow straightforwardly the proof of strong normalization for the Calculus of Construction proposed in [Geu94]. The complexity of both proofs is similar. The technique that we use is extended to dependent types in [Muñ97].

We define $\mathcal{NF}_\phi$ as the set that contains all the ϕ-normal forms of $\Lambda_\mathcal{Q}$. If x in $\Lambda_\mathcal{Q}$, $\overline{x}$ denotes its ϕ-normal form.

Definition 3 *Let $x, y \in \mathcal{NF}_\phi$, we say that x β_ϕ-converts to y, noted by $x \xrightarrow{\beta_\phi} y$, iff $x \xrightarrow{\text{(Beta)}} w$ and $y = \overline{w}$.*

Let $\mathcal{SN}$ be the set of β_ϕ-strongly normalizing expressions of $\mathcal{NF}_\phi$.

Lemma 29 *If M is a term in $\mathcal{SN}$ and S is a substitution in $\mathcal{SN}$, then $M \cdot S \in \mathcal{SN}$.*

Proof. Since $M, S \in \mathcal{SN}$, we can reason by induction on $\nu(M) + \nu(S)$[4]. We check one-step β_ϕ-reductions of $M \cdot T$.

- $M \cdot T \xrightarrow{\beta_\phi} M \cdot T'$, with $T \xrightarrow{\beta_\phi} T'$. If $T \in \mathcal{SN}$, then $T' \in \mathcal{SN}$, and $\nu(M) + \nu(T) > \nu(M) + \nu(T')$. By induction hypothesis, $M \cdot T' \in \mathcal{SN}$.
- $M \cdot T \xrightarrow{\beta_\phi} M' \cdot T$, with $M \xrightarrow{\beta_\phi} M'$. If $M \in \mathcal{SN}$, then $M' \in \mathcal{SN}$, and $\nu(M) + \nu(T) > \nu(M') + \nu(T)$. By induction hypothesis, $M' \cdot T \in \mathcal{SN}$.
- $M \cdot T \xrightarrow{\beta_\phi} \uparrow^n$. But $\uparrow^n$ is a β_ϕ-normal form, then it is in $\mathcal{SN}$.

In every case, $M \cdot T$ reduces in one-step to a β_ϕ-strongly normalizing substitution, then $M \cdot T \in \mathcal{SN}$. □

Lemma 30 *Let $M, S \in \mathcal{NF}_\phi$, for any substitution T, (1) if $\overline{M[T]} \in \mathcal{SN}$, then $M \in \mathcal{SN}$; and (2) if $\overline{S \circ T} \in \mathcal{SN}$, then $S \in \mathcal{SN}$.*

Proof. We prove by simultaneous structural induction on M and S that if $M \xrightarrow{\beta_\phi} M'$ and $S \xrightarrow{\beta_\phi} S'$, then for any substitution T, $\overline{M[T]} \xrightarrow{\beta_\phi} \overline{M'[T]}$ and $\overline{S \circ T} \xrightarrow{\beta_\phi} \overline{S' \circ T}$.

Let T be a substitution such that $\overline{M[T]} \in \mathcal{SN}$. We reason by induction on $\nu(\overline{M[T]})$. We check one-step β_ϕ-reductions of M. Let M' be such that $M \xrightarrow{\beta_\phi} M'$. By the above property we have $\overline{M[T]} \xrightarrow{\beta_\phi} \overline{M'[T]}$. Since $\overline{M[T]} \in \mathcal{SN}$, $\overline{M'[T]} \in \mathcal{SN}$ and $\nu(\overline{M[T]}) > \nu(\overline{M'[T]})$. By induction hypothesis, $M' \in \mathcal{SN}$. In any case we obtain an expression which is strongly β_ϕ-normalizing, hence $M \in \mathcal{SN}$. Using a similar argument we can prove that $S \in \mathcal{SN}$. □

Definition 4 *The set of* base terms *$\mathcal{B}$ in $\mathcal{NF}_\phi$ is defined by*

1. *$x \in \mathcal{B}$, if $x \in \{\mathbf{1}, \mathbf{1}[\uparrow^{Suc(n)}]\}$,*
2. *If X is a meta-variable and $S \neq \uparrow^0$ is a substitution in $\mathcal{SN}$, then $X \in \mathcal{B}$ and $X[S] \in \mathcal{B}$,*
3. *If $M \in \mathcal{B}$ and N is a term in $\mathcal{SN}$, then $(M\ N) \in \mathcal{B}$,*

Definition 5 *The* key redex *of a term $M \in \mathcal{NF}_\phi$ is defined by*

1. *If M is a β_ϕ-redex, then M is its key redex,*
2. *If M has key redex M', then $(M\ N)$ has key redex M'.*

[4] "If x is strongly normalizing, $\nu(x)$ is a number which bounds the length of every normalization sequence beginning with x"([GTL89]).

We write $M \xrightarrow[k]{\beta_\phi} M'$ when M β_ϕ-converts to M' by reducing its key redex.

Definition 6 *A set of terms $\Lambda \subseteq \mathcal{NF}_\phi$ is* saturated *if*

1. $\Lambda \subseteq \mathcal{SN}$,
2. $\mathcal{B} \subseteq \Lambda$,
3. *If $M \xrightarrow[k]{\beta_\phi} M'$, $M' \in \Lambda$ and $M \in \mathcal{SN}$, then $M \in \Lambda$.*

Remark: $\mathcal{SN}$ is saturated and all the saturated sets are nonempty.
We note the collection of saturated sets by **SAT**, i.e. $\mathbf{SAT} = \{\Lambda \subseteq \mathcal{NF}_\phi \mid \Lambda \text{ is saturated}\}$.

Definition 7 *If $\Lambda, \Lambda' \in \mathbf{SAT}$, we define the set $\Lambda \to \Lambda' = \{M \in \mathcal{NF}_\phi \mid \forall N \in \Lambda : (M\ N) \in \Lambda'\}$.*

Lemma 31 **SAT** *is closed under function spaces, i.e. if $\Lambda, \Lambda' \in \mathbf{SAT}$, then $\Lambda \to \Lambda' \in \mathbf{SAT}$.*

Proof. We must prove:

1. If $x \in \Lambda \to \Lambda'$, then $x \in \mathcal{SN}$;
2. If $x \in \mathcal{B}$, then $x \in \Lambda \to \Lambda'$; and
3. If $x \in \Lambda \to \Lambda'$, $y \xrightarrow[k]{\beta_\phi} x$ and $y \in \mathcal{SN}$, then $y \in \Lambda \to \Lambda'$.

The first two properties are proved directly from definitions. We only consider the last case. Let $z \in \Lambda$, so by definition, $(y\ z) \xrightarrow[k]{\beta_\phi} (x\ z)$ and $(x\ z) \in \Lambda'$. Hence, it suffices to prove that $(y\ z) \in \mathcal{SN}$. Since $y, z \in \mathcal{SN}$, we prove by a simple induction on $\nu(y) + \nu(z)$ that if $(y\ z) \xrightarrow{\beta_\phi} x'$, then $x' \in \mathcal{SN}$. Therefore, $(y\ z) \in \mathcal{SN}$. □

Definition 8 *The* type interpretation function *is defined inductively on types as follows:*

$$\begin{array}{lcl} [\![\iota]\!] & = & \mathcal{SN} \\ [\![A \to B]\!] & = & [\![A]\!] \to [\![B]\!] \end{array}$$

Remark: By Lemma 31, for any type A, $[\![A]\!] \in \mathbf{SAT}$.

Lemma 32 *Let $M \in \mathcal{NF}_\phi$, if for all $N \in [\![A]\!]$, $\overline{M[N \cdot \uparrow^0]} \in [\![B]\!]$, then $\lambda_A.M \in [\![A]\!] \to [\![B]\!]$.*

Proof. Let $N \in [\![A]\!]$. By definition, $((\lambda_A.M)\ N) \xrightarrow[k]{\beta_\phi} \overline{M[N \cdot \uparrow^0]}$, and by hypothesis, $\overline{M[N \cdot \uparrow^0]} \in [\![B]\!]$. Hence, it suffices to prove that $((\lambda_A.M)\ N) \in \mathcal{SN}$. But, $\overline{M[N \cdot \uparrow^0]} \in \mathcal{SN}$, so by Lemma 30, $M \in \mathcal{SN}$; and by hypothesis, $N \in \mathcal{SN}$. We prove by a simple induction on $\nu(M) + \nu(N)$ that if $((\lambda_A.M)\ N) \xrightarrow{\beta_\phi} M'$, then $M' \in \mathcal{SN}$. Therefore, $((\lambda_A.M)\ N) \in \mathcal{SN}$. □

Definition 9 *The* valuations *of Γ, noted by $[\![\Gamma]\!]$, is a set of substitutions in $\mathcal{NF}_\phi$ defined inductively on Γ as follows:*

$$\begin{array}{lcl} [\![nil]\!] & = & \{\uparrow^n \mid \textit{for any natural } n\} \\ [\![A.\Gamma']\!] & = & [\![nil]\!] \cup \{M \cdot S \mid M \in [\![A]\!], S \in [\![\Gamma']\!]\} \end{array}$$

Lemma 33 *If $M \in [\![A]\!]$ and $S \in [\![\Gamma]\!]$, then $\overline{M \cdot S} \in [\![A.\Gamma]\!]$.*

Proof. Notice that $M \cdot S$ is not necessarily in $\mathcal{NF}_\phi$. But there are two cases: $\overline{M \cdot S} = M \cdot S$ or $\overline{M \cdot S} = \uparrow^n$. In both cases we verify that $\overline{M \cdot S} \in [\![A.\Gamma]\!]$. □

Lemma 34 *If $S \in [\![\Gamma]\!]$ and $\Gamma \vdash \uparrow^n \triangleright \Delta$, then $\overline{\uparrow^n \circ S} \in [\![\Delta]\!]$.*

Proof. By structural induction on n and S.

- $n = 0$. By Lemma 15(6) we have $\Gamma = \Delta$, and by definition, $\overline{\uparrow^0 \circ S} = S$, so by hypothesis, $\overline{\uparrow^n \circ S} \in [\![\Gamma]\!] = [\![\Delta]\!]$.
- $n = Suc(n')$, $S = \uparrow^m$. By definition, $\overline{\uparrow^n \circ \uparrow^m} = \uparrow^k \in [\![\Delta]\!]$.
- $n = Suc(n')$, $S = M \cdot T$. By definition, $\Gamma = A.\Gamma'$ and $T \in [\![\Gamma']\!]$. By Lemma 15(7), $\Gamma' \vdash \uparrow^{n'} \triangleright \Delta$, so by induction hypothesis, $\overline{\uparrow^{n'} \circ T} \in [\![\Delta]\!]$. But also, $\overline{\uparrow^{Suc(n')} \circ (M \cdot T)} = \overline{\uparrow^{n'} \circ T}$.

□

Lemma 35 *For any* Γ, $[\![\Gamma]\!] \subseteq \mathcal{SN}$.

Proof. We prove by structural induction on S that if $S \in [\![\Gamma]\!]$, then $S \in \mathcal{SN}$.

- $S = \uparrow^n$. In this case S is a β_ϕ-normal form, then the conclusion is trivial.
- $S = M \cdot T$. By definition, $\Gamma = A.\Gamma'$, $T \in [\![\Gamma']\!]$, $M \in [\![A]\!] \subseteq \mathcal{SN}$. By induction hypothesis, $T \in \mathcal{SN}$, thus we conclude with Lemma 29 that $S \in \mathcal{SN}$.

Definition 10 *Let* $M, S \in \mathcal{NF}_\phi$, *we define*

1. Γ *satisfies that* M *is of type* A, *noted by* $\Gamma \models M : A$, *iff* $\overline{M[T]} \in [\![A]\!]$ *for any* $T \in [\![\Gamma]\!]$.
2. Γ *satisfies that* S *is of type* Δ, *noted by* $\Gamma \models S \triangleright \Delta$, *iff* $\overline{S \circ T} \in [\![\Delta]\!]$ *for any* $T \in [\![\Gamma]\!]$.

Proposition 12 (Soundness of $\models$)

1. *If* $\Gamma \vdash M : A$, *then* $\Gamma \models M : A$,
2. *If* $\Gamma \vdash S \triangleright \Delta$, *then* $\Gamma \models S \triangleright \Delta$.

Proof. By simultaneous structural induction on M and S.

- $M = \mathbf{1}$. Let $T \in [\![\Gamma]\!]$, there are three cases:
  - $T = \uparrow^0$. Therefore, $\overline{\mathbf{1}[T]} = \mathbf{1} \in \mathcal{B} \subseteq [\![A]\!]$.
  - $T = \uparrow^{Suc(n)}$. Therefore, $\overline{\mathbf{1}[T]} = \mathbf{1}[\uparrow^{Suc(n)}] \in \mathcal{B} \subseteq [\![A]\!]$.
  - $T = M' \cdot S'$. Therefore, $\overline{\mathbf{1}[T]} = M'$. By definition and Lemma 15(1), $\Gamma = A.\Gamma'$ and $M \in [\![A]\!]$.
- $M = \mathbf{1}[\uparrow^{Suc(n)}]$. By Lemma 15(5), $\Gamma \vdash \uparrow^{Suc(n)} \triangleright \Delta$ and $\Delta \vdash \mathbf{1} : A$. Let $T \in [\![\Gamma]\!]$, by induction hypothesis, $\overline{\uparrow^{Suc(n)} \circ T} \in [\![\Delta]\!]$. Notice that $\overline{\mathbf{1}[\uparrow^{Suc(n)}][T]} = \overline{\mathbf{1}[\uparrow^{Suc(n)} \circ T]} = \overline{\mathbf{1}[\overline{\uparrow^{Suc(n)} \circ T}]}$. By induction hypothesis, $\overline{\mathbf{1}[\overline{\uparrow^{Suc(n)} \circ T}]} \in [\![A]\!]$, and thus, $\overline{\mathbf{1}[\uparrow^{Suc(n)}][T]} \in [\![A]\!]$.
- $M = X$ (X is a meta-variable). Let $T \in [\![\Gamma]\!]$, there are two cases:
  - $T = \uparrow^0$. Therefore, $\overline{X[T]} = X \in \mathcal{B} \subseteq [\![A]\!]$.
  - $T \neq \uparrow^0$. Therefore, $\overline{X[T]} = X[T]$. By Lemma 35, $T \in \mathcal{SN}$, then by definition, $X[T] \in \mathcal{B} \subseteq [\![A]\!]$.
- $M = X[S']$ (X is a meta-variable). By Lemma 15(7), $\Gamma \vdash S' \triangleright \Delta$, $\Delta \vdash X : A$ and $\Gamma \vdash X[S'] : A$. Let $T \in [\![\Gamma]\!]$, by induction hypothesis, $\overline{S' \circ T} \in [\![\Delta]\!]$. Notice that $\overline{X[S'][T]} = \overline{X[S' \circ T]} = \overline{X[\overline{S' \circ T}]}$. By induction hypothesis, $\overline{X[\overline{S' \circ T}]} \in [\![A]\!]$, and thus, $\overline{X[S'][T]} \in [\![A]\!]$.

- $M = \lambda_{A_1}.M_1$. By Lemma 15(3), $A_1.\Gamma \vdash M_1 : B_1$, $\Gamma \vdash \lambda_{A_1}.M_1 : A_1 \to B_1$ and $A = A_1 \to B_1$. By definition, $[\![A]\!] = [\![A_1 \to B_1]\!] = [\![A_1]\!] \to [\![B_1]\!]$. Let $T \in [\![\Gamma]\!]$ and $\Uparrow(T)$ be a notation for $\mathbf{1} \cdot (S \circ \uparrow^{Suc(0)})$. By definition, $\overline{(\lambda_{A_1}.M_1)[T]} = \lambda_{A_1}.\overline{M_1[\Uparrow(T)]}$. By Lemma 32, it suffices to prove that for any $N \in [\![A_1]\!]$, $\overline{M_1[\Uparrow(T)]}[N \cdot \uparrow^0] \in [\![B_1]\!]$. By hypothesis and Lemma 33, $\overline{N \cdot T} \in [\![A_1.\Gamma]\!]$, then by induction hypothesis, $\overline{M[\overline{N \cdot T}]} = \overline{\overline{M_1[\Uparrow(T)]}[N \cdot \uparrow^0]} \in [\![B_1]\!]$.

- $M = (M_1\ N_1)$. By Lemma 15(4), $\Gamma \vdash M_1 : B \to A$ and $\Gamma \vdash N_1 : B$. Let $T \in [\![\Gamma]\!]$, by definition, $\overline{(M_1\ N_1)[T]} = (\overline{M_1[T]}\ \overline{N_1[T]})$. By induction hypothesis, $\overline{M_1[T]} \in [\![B \to A]\!] = [\![B]\!] \to [\![A]\!]$ and $\overline{N_1[T]} \in [\![B]\!]$. Hence, $\overline{(M_1\ N_1)[T]} \in [\![A]\!]$.

- $S = \uparrow^n$. In this case we apply Lemma 34.

- $S = M' \cdot S'$. By Lemma 15(8), $\Gamma \vdash M' : A'$, $\Gamma \vdash S' \triangleright \Delta'$ and $A'.\Delta' = \Delta$. Let $T \in [\![\Gamma]\!]$, so by definition, $\overline{S \circ T} = \overline{\overline{M'[T]} \cdot \overline{S' \circ T}}$. By induction hypothesis, $\overline{M'[T]} \in [\![A']\!]$ and $\overline{S' \circ T} \in [\![\Delta']\!]$. From Lemma 33 we conclude that $\overline{\overline{M'[T]} \cdot \overline{S' \circ T}} \in [\![\Delta]\!]$.

□

Theorem 6 *Let M, S be expressions in $\mathcal{NF}_\phi$.*

1. *If $\Gamma \vdash M : A$, then $M \in \mathcal{SN}$.*
2. *If $\Gamma \vdash S \triangleright \Delta$, then $S \in \mathcal{SN}$.*

Proof. We have $\uparrow^0 \in [\![\Gamma]\!]$. Hence,

1. By Proposition 12, $\overline{M[\uparrow^0]} = M \in [\![A]\!]$, and by definition, $[\![A]\!] \subseteq \mathcal{SN}$.
2. By Proposition 12, $\overline{S \circ \uparrow^0} = S \in [\![\Delta]\!]$, and by Lemma 35, $[\![\Delta]\!] \subseteq \mathcal{SN}$.

□

Theorem 7 *If $\Gamma \vdash M : A$ and $\Gamma \vdash S \triangleright \Delta$, then M and S are weakly normalizing, and thus M and S have $\lambda\Pi_\sigma$-normal forms.*

Proof. Let $N = \overline{M}$ and $T = \overline{S}$, the subject reduction property (Theorem 5) says that typing is preserved under reductions, hence $\Gamma \vdash N : A$ and $\Gamma \vdash T \triangleright \Delta$. Therefore, by Theorem 6, N and T are both in $\mathcal{SN}$. Finally note that a β_ϕ-normal form in $\mathcal{NF}_\phi$ is a $\lambda\Pi_\sigma$-normal form too. □

7 Conclusions

We have proposed a variant of λ_σ, namely λ_ϕ. This calculus enjoys the same general properties of λ_σ:

- a simple and finitary first-order rewriting system,
- confluent on terms with meta-variables,
- weakly terminating on typed terms and
- with composition of substitutions and simultaneous substitutions.

But in contrast to λ_σ, λ_ϕ has not the (SCons) rule and so, it is left-linear in the sort of terms and substitutions.

Although λ_ϕ was designed to allows meta-variables, it happen to be useful in the same frame where λ_σ is. In particular both calculi share the same description of normal forms. For example, the higher-order unification algorithm via explicit substitutions proposed in [DHK95] can be expressed in λ_ϕ, almost without modifications. Moreover, since λ_ϕ has not the surjective pairing rule, it is useful for applications where

these feature of λ_σ pose technical problems, for instance higher-order equational unification via explicit substitutions ([KR96]) or higher-order type systems ([Muñ97]).

Another left-linear variant of λ_σ is the $\lambda_{\Uparrow}$-calculus ([CHL96]). $\lambda_{\Uparrow}$ is fully confluent on open terms, not only with meta-variables of terms but also with meta-variables of substitutions. However, $\lambda_{\Uparrow}$ is incompatible with the extensional rule (η) due to the fact that substitutions id and $\mathbf{1} \cdot \uparrow$ are not $\lambda_{\Uparrow}$-convertible. A key point in λ_ϕ is the preservation of this extensional equivalence. We conjecture that λ_ϕ enjoys the confluence property in its extensional version too.

Acknowledgments

Many thanks to all persons contributing to this work, in particular to Gilles Dowek, Benjamin Werner, Bruno Barras and Gopalan Nadathur for their useful remarks and suggestions on the subject of this paper. The author is very grateful with Thomas Arts and Hans Zantema for their help with Theorem 1, in particular Hans Zantema send to me a simpler proof to that presented here in a personal communication ([Zan96]).

References

[ACCL91] M. Abadi, L. Cardelli, P.-L. Curien, and J.-J. Lévy. Explicit substitution. *Journal of Functional Programming*, 1(4):375–416, 1991.

[BR95] R. Bloo and K. H. Rose. Preservation of strong normalisation in named lambda calculi with explicit substitution and garbage collection. In *CSN-95: Computer Science in the Netherlands*, November 1995.

[CHL96] P.-L. Curien, T. Hardin, and J.-J. Lévy. Confluence properties of weak and strong calculi of explicit substitutions. *Journal of the ACM*, 43(2):362–397, March 1996.

[DHK95] G. Dowek, T. Hardin, and C. Kirchner. Higher-order unification via explicit substitutions (extended abstract). In *Proceedings, Tenth Annual IEEE Symposium on Logic in Computer Science*, pages 366–374, San Diego, California, 26–29 June 1995. IEEE Computer Society Press.

[FKP96] M. C. F. Ferreira, D. Kesner, and L. Puel. λ-calculi with explicit substitutions and composition which preserve β-strong normalization. *LNCS*, 1139, 1996.

[Geu94] H. Geuvers. A short and flexible proof of Strong Normalization for the Calculus of Constructions. In P. Dybjer and B. Nordström, editors, *Types for Proofs and Programs, International Workshop TYPES'94*, volume 996 of *LNCS*, pages 14–38, Båstad, Sweden, 1994. Springer.

[GL97] J. Goubault-Larrecq. Une preuve de terminaison faible du $\lambda\sigma$-calcul. Technical Report RR-3090, Unité de recherche INRIA-Rocquencourt, Janvier 1997.

[GTL89] J.-Y. Girard, P. Taylor, and Y. Lafont. *Proof and Types*. Cambridge University Press, 1989.

[Har89] T. Hardin. Confluence results for the Pure Strong Categorical Logic CCC: λ-calculi as subsystems of CCL. *Theoretical Computer Science*, 65(2):291–342, 1989.

[HL86] T. Hardin and A. Laville. Proof of termination of the rewriting system subst on ccl. *Theoretical Computer Science*, 46:291–342, 1986.

[HMP95] T. Hardin, L. Maranget, and B. Pagano. Functional back-ends and compilers within the lambda-sigma calculus. In Thomas Johnsson, editor, *The Workshop on the Implementation of Functional Languages '95*. Bastad, Sweden, September 1995.

[Hue80] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *J.A.C.M.*, 27(4), October 1980.

[Kes96] D. Kesner. Confluence properties of extensional and non-extensional λ-calculi with explicit substitutions (extended abstract). In Harald Ganzinger, editor, *Proceedings of the 7th International Conference on Rewriting Techniques and Applications (RTA-96)*, volume 1103 of *LNCS*, pages 184–199, New Brunswick, NJ, USA, 1996. Springer-Verlag.

[KNO90] D. Kapur, P. Narendran, and F. Otto. On ground-confluence of term rewriting systems. *Information and Computation*, 86(1):14–31, May 1990.

[KR95] F. Kamareddine and A. Rios. A λ-calculus à la de Bruijn with explicit substitutions. In *PLILP*. LNCS, 1995.

[KR96] C. Kirchner and C. Ringeissen. Higher order equational unification via explicit substitutions. Personal communication, 1996.

[KZ89] D. Kapur and H. Zhang. RRL: A rewrite rule laboratory-user's manual. Technical Report 89-03, Department of Computer Science, The University of Iowa, 1989.

[Les94] P. Lescanne. From $\lambda\sigma$ to $\lambda\upsilon$ a journey through calculi of explicit substitutions. In *Proceedings of the 21st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 60–69, January 1994.

[LRD95] P. Lescanne and J. Rouyer-Degli. Explicit substitutions with de Bruijn's levels. In J. Hsiang, editor, *Rewriting Techniques and Applications*, volume 914 of *LNCS*, pages 294–308, Chapel Hill, North Carolina, 1995. Springer-Verlag.

[Mel95] P.-A. Melliès. Typed λ-calculi with explicit substitutions may not terminate. In *Typed Lambda Calculi and Applications*, number 902 in LNCS. Second International Conference TLCA'95, Springer-Verlag, 1995.

[Muñ96] C. Muñoz. Confluence and preservation of strong normalisation in an explicit substitutions calculus (extended abstract). In *Proceedings, Eleven Annual IEEE Symposium on Logic in Computer Science*, New Brunswick, New Jersey, July 1996. IEEE Computer Society Press.

[Muñ97] C. Muñoz. Meta-theoretical properties of dependent type systems with explicit substitutions. In preparation, 1997.

[Nad96] G. Nadathur. Personal communication, 1996.

[Pag96] B. Pagano. Confluent extensions of $\lambda_{\Uparrow}$. Personal communication, 1996.

[Río93] A. Ríos. *Contributions à l'étude de λ-calculs avec des substitutions explicites.* PhD thesis, U. Paris VII, 1993.

[SS89] M. Schmidt-Schauss. *Computational aspects of an order-sorted logic with term declarations*, volume 395 of *Lecture Notes in Computer Science and Lecture Notes in Artificial Intelligence*. Springer-Verlag Inc., New York, NY, USA, 1989.

[YH88] H. Yokouchi and T. Hikita. A rewriting system for categorical combinators with multiple arguments. Preprint, 1988.

[Zan94] H. Zantema. Termination of term rewriting: interpretation and type elimination. *Journal of Symbolic Computation*, 17(1):23–50, January 1994.

[Zan96] H. Zantema. Personal communication, 1996.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 46 avenue Félix Viallet, 38031 GRENOBLE Cedex 1
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105,
78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS
Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
ISSN 0249-6399