An operational characterization of β strong normalization *

Luca Paolini
Università di Torino (ITALY), Dipartimento di Informatica
Elaine Pimentel
Universidade Federal de Minas Gerais (BRAZIL), Departamento de Matemática
Simona Ronchi Della Rocca
Università di Torino (ITALY), Dipartimento di Informatica

Abstract

We present a new calculus, the $\lambda\Phi$-calculus, that models a non lazy call-by-value evaluation. The $\lambda\Phi$-calculus can be seen as an alternative to Plotkin's $\lambda\beta_v$-calculus, which is intrinsically lazy. Besides confluence and standardization, this new calculus enjoys some other interesting properties. The most remarkable one is the fact that the set of potentially Φ-valuable terms coincides with the strongly β-normalizing terms. Since a similar property holds for Plotkin's calculus – namely that the set of potentially valuable terms in that calculus coincides with the set of *lazy* strongly normalizing terms – the results presented here show an interesting connection between call-by-name and call-by-value evaluation.

1 Introduction

The standard λ-calculus equipped with the β-reduction is the paradigmatic language for the call-by-name functional computation. Its call-by-value version, historically called $\lambda\beta_v$-calculus, has been introduced by Plotkin in 1975 [11]. The $\lambda\beta_v$-calculus is based on a restriction of the β-rule, firing it only when the argument belongs to a particular subset of terms, called values. In [10, 13] two co-authors of this paper, in order to treat these two different calculi in an uniform way, introduced the $\lambda\Delta$-calculus, parametric with respect to a subset Δ of terms, called *input values*, which generalizes the idea of Plotkin's values. Some conditions on Δ have been stated, assuring some good properties for the calculus, in particular confluence and standardization. These conditions are very natural: the set of input values must contain the set of variables and it must be closed under substitution and Δ-reduction (a further condition is necessary for having standardization).

*Paper partially supported by MIUR-PRIN'04 FOLLIA Project and by CNPq.

Note that, in this setting, the standard λ-calculus now can be seen as a degenerated case of a call-by-value calculus, where all terms are input values (that is, the set of input values is Λ). Plotkin's calculus coincides with the $\lambda\Gamma$-calculus, where $\Gamma = \mathrm{Var} \cup \{\lambda x.M \mid M \in \Lambda\}$.

The formalization of the $\lambda\Delta$-calculus, where both call-by-name and call-by-value calculus can be uniformely representable, is an useful tool for studying the relations between these two notions of computations. Some interesting properties relating $\lambda\Lambda$ and $\lambda\Gamma$-calculus have been already proved. In the $\lambda\Gamma$-calculus, the notion of normal form is meaningless since there are different Γ-normal forms that can be consistently equated (see [13]). But the notion of β-normal form has an important meaning also in this calculus: in [7] it was proved that two different $\beta\eta$-normal forms can be separated in the $\lambda\Gamma$-calculus, and hence they cannot be consistently equated in any model. In [8] we further explored the relationship between β-normal forms and Γ-evaluation. Let the lazy β-reduction be the closure of the β-rule under application, but not under abstraction; this corresponds operationally to do not perform reduction under the λ-abstraction[1]. In [8], we proved that the set of strongly normalizing terms, with respect to this reduction, coincides with the set of potentially Γ-valuable terms. A term is potentially Δ-valuable (where Δ is any set of input values) if and only if there is a substitution, replacing variables by input values, such that the substituted term reduces to an input value. Being Δ-input values the only terms that the Δ-calculus can manipulate, in particular that can be arguments of a function, this class of terms is particularly interesting. For example, in the operational semantics of Plotkin for the $\lambda\Gamma$-calculus [11], based on the SECD machine [6], a potentially Γ-valuable term is always different from a not potentially Γ-valuable one, and the not potentially Γ-valuable terms are all equated [13].

A natural question is if this analogy between a call-by-name strong normalization and a call-by-value evaluation can be further developed. In particular, *if there is a set of input values Φ such that the set of potentially Φ-valuable terms coincides with the set of strongly β-normalizing terms.*

The set of lazy β-strong normalizing terms and the set Γ of input values have an interesting structural analogy: in order to find a lazy β-normal form of a λ-term it is not necessary to reduce under a λ-abstraction, and in order to check if a λ-term belongs to Γ, so if it is a Γ-input value, it is not necessary to look under a λ-abstraction, since all λ-abstractions are in Γ. Let us call *weak* a set of input values containing all abstrations. To be weak has some consequences. Let us extend the notion of solvability to a generic Δ-calculus in the natural way, i.e., a term M is Δ-solvable if and only if there is a sequence of Δ input values $P_1, ..., P_n$ such that $(\lambda\vec{x}.M)P_1...P_n =_\Delta \lambda x.x$ (where $\lambda\vec{x}.M$ is the term obtained from M by abstracting it with respect to all its free variables). In the $\lambda\Gamma$-calculus, the set of solvable terms is a proper subset of the set of potentially valuable terms. This depends from the fact that Γ is weak: in fact, for every Γ-unsolvable term U, $\lambda x.U \in \Gamma$, and $\lambda x.U$ is unsolvable too. So another question

[1] In the field of real functional languages, "lazy" is used with a different meaning. We use it according to the definition given in [13].

arises: *is there a $\lambda\Delta$-calculus such that the set of Δ potentially valuable terms coincides with the set of Δ-solvable terms?*

Certainly such a Δ could not be weak.

Intuitively, the set of Φ-input values we are looking for cannot be weak, since in order to reach the β normal form of a term it is necessary to perform the evaluation under the λ-abstraction. So we asked ourselved a further question: *is there a calculus solving both the previous cited problems?*

The answer to the last problem, and so to the previous two, is the $\lambda\Phi$-calculus, where Φ is a set of input values which is the minimal solution of a recursive equation. Φ is non weak, since it is a proper subset of the set of Φ-normal forms. It turns out that the $\lambda\Phi$-calculus enjoys both confluence and standardization. Moreover we will prove that, with respect to the first problem, Φ is a minimal solution.

A further comment is in order. The motivation of Plotkin in designing the $\lambda\Gamma$-calculus was to propose a paradigmatic language for the call-by-value evaluation in real programming languages, and from this point of view the choice of a weak set of input values is natural, for modeling the notion of closure. Instead, we are doing a theoretical investigation, and our $\lambda\Phi$-calculus is not an alternative proposal for designing new call-by-value languages. But we think such a kind of study can be interesting. In fact, Plotkin himself, in [11], posed the question of an existence of call-by-value λ-language alternative to $\lambda\Gamma$. Indeed, he said that the natural proposal was to choose as set of input values the set of the β-normal forms; but the resulting language does not enjoys the confluence property. So the $\lambda\Phi$-calculus, enjoying both confluence and standardization, gives an aswer to this further question too.

The paper is organized as follows: Section 2 contains basic notions of the parametric $\lambda\Delta$-calculus; in Section 3 the $\lambda\Phi$-calculus is introduced and finally in Section 4 the main theorem is stated and proved.

2 The Parametric λ-Calculus

A calculus is a language equipped with some reduction rules. We will consider here calculi sharing the same language, the language of λ-calculus, while they differ from each other in their reduction rules.

In order to treat them in an uniform way we will use the notion of parametric calculus, the $\lambda\Delta$-calculus, that gives rise to different calculi by different instantiations of the parameter Δ. The $\lambda\Delta$-calculus has been studied in [10, 13]. We use the terminology of [2, 13].

Definition 1 (The language Λ)
Let Var *be a countable set of variables. The set Λ of λ-terms is defined by the following grammar:*

$$M ::= x \mid MM \mid \lambda x.M$$

λ-terms will be ranged over by Latin capital letters. Sets of λ-terms will be denoted by Greek capital letters. If Θ denotes a set of terms $(\Theta)^0$ is the set of closed terms belonging to Θ.

Sometimes, we will refer to λ-terms simply as terms. As usual, terms will be considered modulo α-conversion, i.e., modulo names of bound variables. The symbol $\equiv$ will denote syntactical identity of terms, up to α-equivalence.

We will use the following abbreviations, in order to avoid an excessive number of parentheses, thereby $\lambda x_1...x_n.M$ will stand for $(\lambda x_1(...(\lambda x_n.M)...))$ and $MN_1N_2...N_n$ will stand for $(...((MN_1)N_2)...N_n)$. Moreover $\vec{M}$ will denote a sequence of terms $M_1, \ldots, M_n$, for some $n \geq 0$, and $\lambda\vec{x}.M$ and $\vec{M}\vec{N}$, will denote respectively $\lambda x_1 \ldots x_n.M$ and $M_1 \ldots M_m N_1 \ldots N_n$, for some $n, m \geq 0$. The length of the sequence $\vec{N}$ is denoted by $\|\vec{N}\|$.

The $\lambda\Delta$-calculus consists of the language Λ equipped with a set $\Delta \subseteq \Lambda$ of input values, satisfying some closure conditions. Informally, input values represent already evaluated terms, that can be passed as parameters. The set Δ of input values and the reduction $\rightarrow_\Delta$, induced by it, are defined below.

Definition 2 *Let $\Delta \subseteq \Lambda$.*

i) *The* Δ-reduction *($\rightarrow_\Delta$) is the contextual closure of the following rule:*

$$(\lambda x.M)N \rightarrow M[N/x] \textit{ if and only if } N \in \Delta.$$

$(\lambda x.M)N$ *is a* Δ-redex *(or simply redex).*

ii) *$\rightarrow^+_\Delta$, $\rightarrow^*_\Delta$ and $=_\Delta$ are respectively the transitive closure of $\rightarrow_\Delta$, the reflexive and transitive closure of $\rightarrow_\Delta$ and the symmetric, reflexive and transitive closure of $\rightarrow_\Delta$.*

iii) *A set $\Delta \subseteq \Lambda$ is a* set of input values*, when the following conditions are satisfied:*

- $\mathrm{Var} \subseteq \Delta$ *(*Var-closure*);*
- *$P, Q \in \Delta$ implies $P[Q/x] \in \Delta$, for each $x \in \mathrm{Var}$* *(*substitution closure*);*
- *$M \in \Delta$ and $M \rightarrow_\Delta N$ imply $N \in \Delta$* *(*reduction closure*).*

The closure conditions on the set of input values assure us that the $\lambda\Delta$-calculus enjoys the confluence property for every Δ , i.e., the following theorem holds.

Theorem 2.1 (Confluence) *[10, 13] Let $M \rightarrow^*_\Delta N_1$ and $M \rightarrow^*_\Delta N_2$. There is Q such that both $N_1 \rightarrow^*_\Delta Q$ and $N_2 \rightarrow^*_\Delta Q$.*

Two particular instantiations of Δ give rise to the call-by-name and the call-by-value λ-calculus. The call-by-name λ-calculus (i.e., the standard λ-calculus equipped with the β-reduction) coincides with the $\lambda\Lambda$-calculus. The call-by-value λ-calculus (defined by Plotkin in [11]) coincides with the $\lambda\Gamma$-calculus, where $\Gamma = \text{Var} \cup \{\lambda x.M \mid M \in \Lambda\}$.

Let Δ be a set of input values. A term of the $\lambda\Delta$-calculus is in Δ-*normal form* if and only if it does no contain occurrences of Δ-redexes. A term M is *strongly* Δ-*normalizing* if both M has a Δ-normal form and every reduction sequence starting from M eventually stops.

The set Δ-NF of Δ-normal forms can be defined in the following recursive way:

$$
\begin{aligned}
\Delta\text{-NF} = \quad \text{Var} \quad & \cup \{xM_1 ... M_n \mid M_k \in \Delta\text{-NF } (1 \leq k \leq n)\} \\
& \cup \{\lambda \vec{x}.M \mid M \in \Delta\text{-NF}\} \\
& \cup \{(\lambda x.P)QM_1...M_n \mid P, Q, M_i \in \Delta\text{-NF}, Q \notin \Delta\}
\end{aligned}
$$

Note that for the $\lambda\Lambda$-calculus, being Λ its set of input values, the last case cannot happen, i.e., there are no normal forms of the shape $(\lambda x.P)QM_1...M_n$.

In the $\lambda\Gamma$-calculus, the notion of normal form is meaningless. In fact, there are different Γ-normal forms that can be consistently equated. The key notion, in a call by value setting, is the one of (potential) valuability, given in the next definition (see [9],[13]).

Definition 3

i) *A term M is* Δ-valuable *if and only if there is $N \in \Delta$ such that $M \to_\Delta^* N$.*

ii) *A term M is* potentially Δ-valuable *if and only if there is a substitution* $\mathbf{s}$*, replacing variables by closed terms belonging to Δ, such that $\mathbf{s}(M)$ is Δ-valuable.*

It is immediate to verify that a closed term is potentially Δ-valuable if and only if it is Δ-valuable. Note that the notion of Δ-normal form and that one of potentially Δ-valuable are orthogonal. As an example, consider the $\lambda\Gamma$-calculus, and the term $M \equiv (\lambda z.D)(yI)D$, where $D \equiv (\lambda z.zz)$. M is in Γ-normal form, but it is neither an input value nor potentially Γ-valuable. In fact, consider $M[Q/y]$, for some $Q \in (\Gamma)^0$. If QI reduces to an element in Γ then $M[Q/y] \equiv (\lambda z.D)(QI)D$ reduces to DD, which is not an input value. Otherwise $M[Q/y] \to_\Gamma^* (\lambda z.D)Q'D$, for every Q' such that $QI \to_\Gamma^* Q'$, which is not an input value. Thus $(\lambda z.D)(QI)D$ is not Γ-valuable. In general, we call Δ-*liar-normal forms* terms which are in Δ-normal form but that are not potentially Δ-valuable.

In the $\lambda\Lambda$-calculus, the notion of solvability plays an important role, since in some sense the solvable terms represents the meaningfull computations [2]. In [9], Γ-solvable and potentially Γ-valuable terms has been characterized. This notion has been extended to the parametric $\lambda\Delta$-calculus in [13].

Definition 4 (i) *A context $C[.]$ is Δ-valuable if and only if $C[.] \equiv (\lambda\vec{x}.[.])\vec{P}$ where each $P \in \vec{P}$ is such that $P \in \Delta$.*

(ii) *A term M is* Δ-solvable *if and only if there is a* Δ*-valuable context* $C[.]$ *such that:*

$$C[M] =_\Delta I.$$

(iii) *A term is* Δ-unsolvable *if and only if it is not* Δ*-solvable.*

Note that $(\lambda\vec{x}.M)\vec{N} =_\Delta I$ means $(\lambda\vec{x}.M)\vec{N} \to^*_\Delta I$, since I is in Δ-NF, for every Δ.

3 The $\lambda\Phi$-calculus

First of all, we need to fix precisely what kind of "non lazy" calculus we are looking for. In order to design a language with a reasonable operational semantics, we ask that a term is completely evaluated if and only if all its subterms are completely evaluated.

A first attempt then is to define a set Δ of input values satisfying the condition that Δ coincides with its set of normal forms (i.e. $\Delta = \Delta$-NF). Remembering the recursive definition of Δ-normal form, that Δ must be a set of input values, the following equations would be obtained:

(1.1) $\Delta = \text{Var} \cup \{xM_1...M_n \mid M_k \in \Delta\ (1 \leq k \leq n)\} \cup \{\lambda\vec{x}.M \mid M \in \Delta\} \cup \{(\lambda x.P)QM_1...M_n \mid P, Q, M_k \in \Delta\ (1 \leq k \leq n), Q \notin \Delta\}$;

(1.2) $M, P \in \Delta$ implies $M[P/x] \in \Delta$.

Note that the reduction closure for Δ is trivially satisfied, since we asked that terms in Δ are Δ-normal forms.
Unfortunately, the only solution to the equation (1.1) is $\Delta = \Lambda$; in fact the set $\{(\lambda x.P)QM_1...M_n \mid P, Q, M_k \in \Delta\ (1 \leq k \leq n), Q \notin \Delta\}$ need to be empty, due to the contradictory condition on Q.

Thus we should look for a set Δ which is a proper subset of the Δ-normal forms. In order to satisfy equation (1.2), we will impose that Δ contain only closed terms, besides variables. Consequently, the previous pair of equations now become:

(2.1) $\Theta = \text{Var} \cup \{xM_1...M_n \mid M_k \in \Theta\ (1 \leq k \leq n)\} \cup \{\lambda\vec{x}.M \mid M \in \Theta\} \cup \{(\lambda x.P)QM_1...M_n \mid P, Q, M_k \in \Theta\ (1 \leq k \leq n), Q \notin \Delta\}$;

(2.2) $\Delta = Var \cup (\Theta)^0$.

But the last condition on equation (2.1) is too weak again, since now the set Θ may contain some Δ-liar-normal forms. As an example, let $M \equiv (\lambda z.D)(yI)D$ where $D \equiv (\lambda z.zz)$. Clearly M is a Δ-liar-normal form. The following property will help us in excluding such dangerous terms.

Property 3.1 *Let Δ be a set of input values, let Θ be such that $\Theta \subset \Delta$-NF and let $\Delta = (\Theta)^0 \cup Var$. If $\mathbf{s}(P[Q/x]\vec{M}) \to^*_\Delta R \in \Delta$ and $\mathbf{s}(Q) \to^*_\Delta Q' \in \Delta$ then $(\lambda x.P)Q\vec{M}$ is potentially Δ-valuable.*

Proof. Let $\mathbf{s}$ be such that $\mathbf{s}(P[Q/x]\vec{M}) \equiv \mathbf{s}(P)[\mathbf{s}(Q)/x]\mathbf{s}(\vec{M}) \rightarrow^*_\Delta N \in \Delta$. Since input values are closed under substitutions, without loss of generality we can assume that $\mathbf{s}(P[Q/x]\vec{M}) \in \Lambda^0$. So $\mathbf{s}(P)[\mathbf{s}(Q)/x]\mathbf{s}(\vec{M}) \rightarrow^*_\Delta \mathbf{s}(P)[Q'/x]\mathbf{s}(\vec{M}) =_\Delta (\lambda x.\mathbf{s}(P))Q'\mathbf{s}(\vec{M}) =_\Delta \mathbf{s}(\lambda x.P)\mathbf{s}(Q)\mathbf{s}(\vec{M})$ □

Taking into account the previous property, the pair of equations now becomes:

(3.1) $\Theta = \text{Var} \cup \{xM_1...M_n \mid M_k \in \Theta\ (1 \leq k \leq n)\} \cup \{\lambda\vec{x}.M \mid M \in \Theta\} \cup \{(\lambda x.P)QM_1...M_n \mid Q \in \Theta, Q \notin \Delta, P[Q/x]M_1...M_n \rightarrow^*_\Delta R \in \Theta\}$;

(3.2) $\Delta = Var \cup (\Theta)^0$.

The minimal solution of this pair of recursive equations is defined next.

Definition 5 *The sets of λ-terms Υ_i, Φ_i ($i \in \mathbb{N}$) are defined by mutual induction, as follows*

$$\begin{array}{lcl}
\Upsilon_0 & = & \text{Var} \\
\Phi_i & = & \text{Var} \cup (\Upsilon_i)^0 \\
\Upsilon_{i+1} & = & \text{Var} \cup \{xM_1...M_n \mid M_k \in \Upsilon_i (1 \leq k \leq n)\} \cup \{\lambda\vec{x}.M \mid M \in \Upsilon_i\} \cup \\
 & & \{(\lambda x.P)QM_1...M_n \mid Q \in \Upsilon_i - (\Lambda^0 \cup \text{Var}), P[Q/x]M_1 \ldots M_n \rightarrow^*_{\Phi_i} R \in \Upsilon_i\}
\end{array}$$

Moreover, we define $\Upsilon = \cup_i \Upsilon_i$ and $\Phi = \text{Var} \cup (\Upsilon)^0$.

For example, $\Phi_0 = \text{Var}$, $\Upsilon_1 = \text{Var} \cup \{xy_1...y_n \mid y_i \in Var\} \cup \{\lambda\vec{x}.y \mid y \in \text{Var}\}$ and $\Phi_1 = \text{Var} \cup \{\lambda x_1...x_m.x_j \mid 1 \leq j \leq m\}$. It is easy to check that $\Phi = \cup_i \Phi_i$.

Lemma 1 (i) *Φ_i is a set of input value, for all $i \in \mathbb{N}$.*

(ii) *Φ is a set of input values.*

(iii) *Υ and Υ_i are not sets of input values, for all $i \in \mathbb{N}$.*

Proof. Trivial. □

It is easy to check that the following properties hold.

Property 3.2 (i) *$\Upsilon_i \subset \Upsilon_{i+1}$, $\Phi_i \subset \Phi_{i+1}$ and $\rightarrow_{\Phi_i} \subseteq \rightarrow_{\Phi_{i+1}}$, for all $i \in \mathbb{N}$;*

(ii) *$\Upsilon_i \subset \Upsilon$, $\Phi_i \subset \Phi$ and $\rightarrow_{\Phi_i} \subseteq \rightarrow_\Phi$, for all $i \in \mathbb{N}$;*

(iii) *$\Upsilon^0 = \Phi^0$;*

(iv) *$M \in \Phi^0$ implies $M \equiv \lambda z.P$, for some $z \in \text{Var}$ and $P \in \Upsilon$ (note that $\Phi \subseteq \Gamma$);*

(v) *$\Phi \subseteq \Upsilon$ and $\Upsilon \subseteq \Phi$-NF;*

(vi) *Φ-NF $\not\subseteq \Upsilon$ and Φ-NF $\not\subseteq \Phi$.*

Proof. (*vi*) Let $M \equiv \lambda z.(\lambda x.D)(zI)D$. $M \in \Phi$-NF since $zI \notin \Phi$. But $M \notin \Upsilon$ and $M \notin \Phi$. □

Lemma 1.ii) implies that the $\lambda\Phi$-calculus enjoys the confluence property. Moreover it is possible to check that it also satisfies the additional necessary condition for standardization, stated in [10, 13].

4 The Main Result

The $\lambda\Phi$-calculus, besides confluence and standardization, has some further interesting properties. The most important one is that the set of potentially Φ-valuable terms coincides with the set of strongly Λ-normalizing terms. Remembering that the set of potentially Γ-valuable terms coincides with the set of lazy strongly Λ-normalizing terms [8], this property shows that the $\lambda\Phi$-calculus is the right choice for modelling the non lazy call-by-value evaluation.

Other properties characterize completely the operational behavior of the $\lambda\Phi$-calculus. In particular, a term is Φ-solvable if and only if it is potentially Φ-valuable *and* if and only if it Φ-reduces to a term in Υ.

Theorem 4.1 (Main Theorem) *The following statements are equivalent:*

i) M is strongly Λ-normalizing;

ii) M is Φ-solvable;

iii) M is potentially Φ-valuable;

iv) $M \to^*_\Phi R \in \Upsilon$;

Proof. i) implies ii) by Theorem 4.9.
ii) implies iii) by Theorem 4.4.ii.
iii) implies iv) by Theorem 4.4.i.
iv) implies i) by Theorem 4.6. □

We can also prove that the set Φ is a minimal set of input values between these having as potentially terms the set of β-strongly normalizing terms.

Property 4.2 *Let $\Delta^\star$ be a set of input values.*
If the potentially $\Delta^\star$-valuable terms are (exactly) the strongly Λ-normalizing terms and $\Delta^\star \subset \Phi$ then $\Delta^\star = \Phi$.

Proof. Clearly $\Delta^\star = \Phi$ if and only if $(\Delta^\star)^0 = \Phi^0$; so, suppose $M \in \Phi^0$.
Note that M is Φ-valuable, potentially Φ-valuable and also in Φ-normal form. Moreover M is a closed strongly Λ-normalizing term, by the Main Theorem. Thus, M is potentially $\Delta^\star$-valuable by hypothesis and $M \in \Lambda^0$ implies that M is $\Delta^\star$-valuable. But $\Delta^\star \subset \Phi$ implies Φ-NF $\subset \Delta^\star$-NF, hence $M \in \Delta^\star$-NF. Then $M \in \Delta^\star$ and the proof is done. □

However, it is easy to check that Φ is not the minimum set such that the potentially Φ-valuable terms are (exactly) the strongly Λ-normalizing terms. You can consider the minimum solution to the following equations:

$$\begin{aligned}
\Theta = \quad & \{\lambda x_0 ... x_n.y | y \neq x_i \ \ (i \in \mathbb{N})\ \} \cup \{x M_1 ... M_n \mid M_k \in \Theta \ (1 \leq k \leq n)\} \cup \\
& \{\lambda \vec{x}.M \mid M \in \Theta\} \cup \{(\lambda x.P) Q M_1 ... M_n \mid Q \in \Theta, Q \notin \Delta, P[Q/x] M_1 ... M_n \to^*_\Delta R \in \Theta\} \\
\Delta = \quad & \{\lambda x_0 ... x_n.y | y \neq x_i \ \ (i \in \mathbb{N})\ \} \cup (\Theta)^0
\end{aligned}$$

The rest of the paper is devoted to the proof of the Main Theorem.

4.1 Potential Φ-Valuability and Φ-Solvability

First of all, we will introduce the *weight* of a terms, as measure for carrying out some inductive proofs. The weight of a term M is an upper bound to the number of symbols of M, to the lenght of its leftmost Λ-reduction sequence and to the lenght of its Φ-reduction sequence according to the standard policy [10].

Definition 6 *The* weight $\langle \_ \rangle : \Lambda \longrightarrow \mathbb{N}$ *is the partial function defined as follows:*

- $\langle \lambda x.M' \rangle = 1 + \langle M' \rangle$.
- $\langle x M_1 ... M_m \rangle = 1 + \langle M_1 \rangle + ... + \langle M_m \rangle$.
- $\langle (\lambda x.M_0) M_1 \dots M_m \rangle = 1 + \langle M_1 \rangle + \langle M_0[M_1/x] M_2 \dots M_m \rangle$.

As examples $\langle x \rangle = 1$, $\langle xx \rangle = 2$, $\langle \lambda x.xx \rangle = 3$. It is easy to check that $M \to_\Phi N$ implies $M[P/z] \to_\Phi N[P/z]$.

Lemma 2 (i) *If $M \in \Upsilon$ then $\langle M \rangle \in \mathbb{N}$.*

(ii) *If $M \to_\Phi^+ N$ and $\langle N \rangle \in \mathbb{N}$ then, both $\langle M \rangle \in \mathbb{N}$ and $\langle N \rangle < \langle M \rangle$.*

(iii) *If $M \to_\Phi^+ N$ and $\langle M \rangle \in \mathbb{N}$ then, both $\langle N \rangle \in \mathbb{N}$ and $\langle N \rangle < \langle M \rangle$.*

Proof. (i) The proof can be done by induction on the Υ stratification.

(ii) The proof is given by induction on $\langle N \rangle$.

- If $M \equiv \lambda x.P \to_\Phi^+ \lambda x.P' \equiv N$ then $\langle P' \rangle \in \mathbb{N}$ implies $\langle P' \rangle < \langle P \rangle \in \mathbb{N}$ by induction, hence $\langle N \rangle < 1 + \langle P \rangle = \langle M \rangle$.
- Let $M \equiv x M_1 ... M_m \to_\Phi^+ x N_1 ... N_m \equiv N$ $(m \geq 1)$ where either $M_k \to_\Phi^+ N_k$ or $M_k \equiv N_k$ $(1 \leq k \leq m)$. Note that there is at least one $h \in \mathbb{N}$ such that $M_h \to_\Phi^+ N_h$ and $\langle N_h \rangle < \langle M_h \rangle$ $(1 \leq h, k \leq m)$. Thus the proof follows easily by induction.
- Let $M \equiv (\lambda z.M_0) M_1 ... M_m \to_\Phi^+ N$ $(m \geq 1)$.
  Either $M \to_\Phi^* (\lambda x.N_0) N_1 ... N_m \to_\Phi N_0[N_1/x] N_1 ... N_m \to_\Phi^* N$ or $M \to_\Phi^+ (\lambda x.N_0) N_1 ... N_m \equiv N$, where $M_k \to_\Phi^+ N_k$ or $M_k \equiv N_k$ $(1 \leq k \leq m)$. In all cases, the proof follows by induction.

(iii) The proof is given by induction on $\langle M \rangle$.

- If $M \equiv \lambda x.M_0 \to_\Phi^+ \lambda x.N_0 \equiv N$ then $\langle M_0 \rangle \in \mathbb{N}$. Hence $\langle N_0 \rangle < \langle M_0 \rangle$ by induction, thus $\langle N \rangle = 1 + \langle N_0 \rangle < 1 + \langle M_0 \rangle = \langle M \rangle$.
- Let $M \equiv x M_1 ... M_m \to_\Phi^+ x N_1 ... N_m \equiv N$ $(m \geq 1)$, where either $M_k \to_\Phi^+ N_k$ or $M_k \equiv N_k$ $(1 \leq k \leq m)$. Note that there is $h \in \mathbb{N}$ such that $M_h \to_\Phi^+ N_h$ and $\langle N_h \rangle < \langle M_h \rangle$. Thus the proof follows easily by induction.

- Let $M \equiv (\lambda z.M_0)M_1...M_m \to_\Phi^+ N$ $(m \geq 1)$.
  Either $M \to_\Phi^* (\lambda x.N_0)N_1...N_m \to_\Phi N_0[N_{1/x}]N_1...N_m \to_\Phi^* N$ or $M \to_\Phi^+ (\lambda x.N_0)N_1...N_m \equiv N$, where $M_k \to_\Phi^+ N_k$ or $M_k \equiv N_k$ $(1 \leq k \leq m)$.
  If $M_1 \to_\Phi^+ N_1$ then $\langle N_1 \rangle < \langle M_1 \rangle$ and the proof follows by induction. Otherwise $M_1 \equiv N_1$ and $M_0[M_1/x]M_2 \ldots M_m \to_\Phi^+ N_0[M_1/x]N_2 \ldots N_m$, hence $\langle M_0[M_1/x]M_2 \ldots M_m \rangle < \langle N_0[M_1/x]N_2 \ldots N_m \rangle$ and the proof follows immediately, in all cases.

$\square$

Next, it follows a characterization of terms for which the weight is defined.

Corollary 4.3 *$\langle M \rangle$ is defined if and only if $M \to_\Phi^* R$, for some $R \in \Upsilon$.*

Proof. $\Leftarrow$ The proof follows by Lemma 2.i) and Lemma 2.ii).

$\Rightarrow$ The proof is given by induction on $\langle M \rangle$.
If $M \equiv \lambda x.M_0$ or $M \equiv xM_1...M_m$ $(m \in \mathbb{N})$ then the proof follows by induction. Let $M \equiv (\lambda z.M_0)M_1...M_m$ $(m \geq 1)$, so $\langle M_1 \rangle < \langle M \rangle$ and by induction $M_1 \to_\Phi^* S \in \Upsilon$.

- If $M \in \Phi$-NF then $M_1 \in \Upsilon$, but $M_1 \notin \Phi = \Upsilon^0 \cup \text{Var}$. Furthermore $\langle M_0[M_1/x]M_2 \ldots M_m \rangle < 1 + \langle M_1 \rangle + \langle M_0[M_1/x]M_2 \ldots M_m \rangle = \langle M \rangle$ implies that $M_0[M_1/x]M_2 \ldots M_m \to_\Phi^* R'$, for some $R' \in \Upsilon$. The proof follows by definition of Υ.
- Otherwise $M \to_\Phi^+ N$, so the proof follows by Lemma 2.iii).

$\square$

If the weight of a term is defined, then the weight of all its subterms is also defined. The next lemma proves this statement in some particular cases.

Lemma 3 (i) *If $M[N/z] \to_\Phi^* R \in \Upsilon$ then $M \to_\Phi^* S \in \Upsilon$.*

(ii) *If $MN \to_\Phi^* R \in \Upsilon$ then $M \to_\Phi^* S \in \Upsilon$.*

Proof. (i) We will prove that $\langle M[N/z] \rangle \in \mathbb{N}$ implies $\langle M \rangle \in \mathbb{N}$ by induction on $h = \langle M[N/z] \rangle$, thus the proof follows by Corollary 4.3.
Let $M \equiv \lambda x.M_0$. $\langle M_0[N/z] \rangle \in \mathbb{N}$ implies $\langle M_0 \rangle < h$ by induction and the proof follows. If $M \equiv xM_1...M_m$ $(m \geq 1)$ then, in all cases, $\langle M_i[N/z] \rangle < h$ $(1 \leq i \leq m)$ and the proof follows by induction. If $M \equiv (\lambda x.M_0)M_1...M_m$ $(m \geq 1)$ then $\langle M[N/z] \rangle = 1 + \langle M_1[N/z] \rangle + \langle (M_0[M_1/x]M_2...M_m)[N/z] \rangle$. Thus $\langle M_1 \rangle$ and $\langle M_0[M_1/x]M_2...M_m \rangle$ are defined by induction and the proof follows.

(ii) We will prove that $\langle MN \rangle \in \mathbb{N}$ implies $\langle M \rangle \in \mathbb{N}$ by induction on $\langle MN \rangle$, thus the proof follows by Corollary 4.3.
If $M \equiv \lambda x.M_0$ then $\langle MN \rangle = 1 + \langle N \rangle + \langle M_0[N/z] \rangle$, so $\langle M_0[N/z] \rangle \in \mathbb{N}$ and the proof follows by the previous point of this Theorem and the definition of weight. The other cases are easier.

$\square$

Theorem 4.4

(i) *M is potentially Φ-valuable implies that $M \to_\Phi^* R \in \Upsilon$, for some $R \in \Upsilon$.*

(ii) *M is Φ-solvable implies that M is potentially Φ-valuable.*

Proof. (i) M is potentially Φ-valuable means that there is a substitution $\mathbf{s}$, replacing variables by closed terms belonging to Φ, such that $\mathbf{s}(M) \to_\Phi^* N \in \Phi$. Since $\Phi \subseteq \Upsilon$, the proof follows by Lemma 3.i).

(ii) M is Φ-solvable means that there is a Φ-valuable context $C[.] \equiv (\lambda\vec{x}.[.])\vec{N}$ such that $C[M] \to_\Phi^* I$. Since $C[M] \to_\Phi^* I$ implies $C[M]I...I \to_\Phi^* I$, we can assume $\|\vec{x}\| \leq \|\vec{N}\|$ without loss of generality. Moreover $C[M] \to_\Phi^* I$ implies $C[M][N/z] \to_\Phi^* I \equiv I[N/z]$ for all $N \in \Phi^0$, so we can also assume that $C[M] \in \Lambda^0$.
If $\vec{N} \equiv \vec{N_0}\vec{N_1}$ and $\|\vec{x}\| = \|\vec{N_0}\|$ then $C[M] \to_\Phi^* M[\vec{N_0}/\vec{x}]\vec{N_1} \to_\Phi^* I \in \Phi$, thus $M[\vec{N_0}/\vec{x}] \to_\Phi^* S' \in \Upsilon^0$, by Lemma 3.ii). The proof is done, since $\Upsilon^0 = \Phi^0$.

□

4.2 Strongly Λ-normalization and Φ-reduction

In order to prove both that the terms strongly Λ-normalizing are also Φ-solvable and that terms which Φ-reduce to an element of Υ are strongly Λ-normalizing, we will use an intersection type assignment system [1, 4] that types exactly the strongly Λ-normalizing terms [5, 12].

Definition 7 (i) *Let C be a countable set of* type-constants *(ranging over $\alpha, \beta, ..$). The set $T(\mathsf{C})$ of* types*, ranging over by $\sigma, \tau, \pi, \rho, ..$ is inductively defined as follows:*

$$\begin{array}{lcl} \sigma \in \mathsf{C} & \Rightarrow & \sigma \in T(\mathsf{C}) \\ \sigma, \tau \in T(\mathsf{C}) & \Rightarrow & (\sigma \to \tau) \in T(\mathsf{C}) \\ \sigma, \tau \in T(\mathsf{C}) & \Rightarrow & (\sigma \wedge \tau) \in T(\mathsf{C}). \end{array}$$

We use the convention that the constructor $\wedge$ take precedence over $\to$.

(ii) *A basis is a partial function from* Var *to $T(\mathsf{C})$ having a finite domain of definition. If B is a basis then $B[\sigma/x]$ denotes the basis such that*

$$B[\sigma/x](y) = \begin{cases} \sigma & \textit{if } y \equiv x, \\ B(y) & \textit{otherwise.} \end{cases}$$

Furthermore, the basis B such that $dom(B) = \{x_1, ..., x_n\}$ and $B(x_i) = \sigma_i$, for $1 \leq i \leq n$ will be often denoted by $[\sigma_1/x_1, ..., \sigma_n/x_n]$.

(iii) *The* type assignment system $\vdash$ *is a formal system proving typing judgments of the shape:*

$$B \vdash M : \sigma$$

where M is a term, $\sigma \in T(\mathsf{C})$ and B is a basis.

The type assignment system $\vdash$ consists of the following rules:

$$\frac{}{B[\sigma/x] \vdash x : \sigma}\ (var)$$

$$\frac{B[\sigma/x] \vdash M : \tau}{B \vdash \lambda x.M : \sigma \to \tau}\ (\to I) \qquad \frac{B \vdash M : \sigma \to \tau \quad B \vdash N : \sigma}{B \vdash MN : \tau}\ (\to E)$$

$$\frac{B \vdash M : \sigma \qquad B \vdash M : \tau}{B \vdash M : \sigma \wedge \tau}\ (\wedge I)$$

$$\frac{B \vdash M : \sigma \wedge \tau}{B \vdash M : \sigma}\ (\wedge E_l) \qquad \frac{B \vdash M : \sigma \wedge \tau}{B \vdash M : \tau}\ (\wedge E_r)$$

If B, B' are bases then $B \cap B'$ is the basis defined as follows:

$$(B \cap B')(y) = \begin{cases} B(y) \wedge B'(y) & \text{if both } B(y) \text{ and } B'(y) \text{ are defined,} \\ B(y) & \text{if } B(y) \text{ is defined and } B'(y) \text{ is undefined,} \\ B'(y) & \text{if } B'(y) \text{ is defined and } B(y) \text{ is undefined,} \\ \text{undefined} & \text{otherwise.} \end{cases}$$

The type systems $\vdash$ enjoys the subject-reduction property and a restricted form of subject-expansion.

Property 4.5 (Subject-Reduction and Typed Subject-Expansion)

(i) *If $B \vdash M : \sigma$ and $M \to_\Lambda N$ then $B \vdash N : \sigma$.*

(ii) *Let $C[.]$ be a context. Then $B \vdash C[P[Q/x]] : \sigma$ and $B' \vdash Q : \tau$ imply $B \cap B' \vdash C[(\lambda x.P)Q] : \sigma$.*

Proof. See [4]. □

Lemma 4 *If $M \in \Upsilon$ then $B \vdash M : \sigma$, for some basis B and $\sigma \in T(\mathsf{C})$.*

Proof. The proof is given by induction on the stratification of Υ. The case $M \in \Upsilon_0$ is trivial. If $M \in \Upsilon_{i+1}$, the cases $M \equiv xM_1...M_m$ and $M \equiv \lambda x.P$ follow easily from the inductive hypothesis, using the rules of the system. If $M \equiv (\lambda x.P)QM_0...M_m$ then both $Q \in \Upsilon_i - (\Lambda^0 \cup \mathrm{Var})$ and $P[Q/x]M_1 \dots M_m \to^*_{\Phi_i} R \in \Upsilon_i$. Therefore $B \vdash R : \sigma$, by induction. Since the $\to^*_{\Phi_i}$ reduction reduces only in case the argument belongs to Φ_i this can be typed by induction, and hence $B \vdash P[Q/x]M_1 \dots M_m : \sigma$ by Property 4.5.ii). Since $B' \vdash Q : \tau$ by induction, the proof follows by Property 4.5.ii) . □

Theorem 4.6 *$M \to^*_\Phi R \in \Upsilon$ implies M is Λ-strongly normalizing.*

Proof. By Lemma 4, $B \vdash R : \sigma$ for some basis B and $\sigma \in T(\mathsf{C})$. Thus $B \vdash M : \sigma$, by Lemma 4 and by Property 4.5.ii). Then the proof follows from the fact that the system characterizes the strongly Λ-normalizing terms [5, 12]. $\square$

4.3 Strongly Λ-Normalization and Φ-Solvability

In order to prove this result, we will use a computability argument, which is an adaptation to intersection types of the reducibility candidates method [3].

Let $\mathcal{P}(M)$ be the following predicate: "there is $r \in \mathbb{N}$ such that, for all $h, k \geq r$,

$$M[O^h/x_1, ..., O^h/x_m] \underbrace{O^h.....O^h}_{k} \to^*_\Phi O^s$$

for some $s \in \mathbb{N}$, where $O^h \equiv \lambda x_0...x_h.x_h$ and $\mathrm{FV}(M) \subseteq \{x_1, ..., x_m\}$ $(m \geq 0)$".

Property 4.7 *i) $\mathcal{P}(M)$ implies M is Φ-solvable.*

ii) $\mathcal{P}(x\overrightarrow{M})$ and $\mathcal{P}(N)$ imply $\mathcal{P}(x\overrightarrow{M}N)$.

Proof. i) $M[O^h/x_1, ..., O^h/x_m] \underbrace{O^h.....O^h}_{k} \to^*_\Phi O^s$ implies that

$$M[O^h/x_1, ..., O^h/x_m] \underbrace{O^h.....O^h}_{k} \underbrace{I.....I}_{s} \to^*_\Phi I.$$

ii) Let $\overrightarrow{M} \equiv M_1...M_n$ $(n \in \mathbb{N})$ and let $\mathrm{FV}(x\overrightarrow{M}N) \subseteq \{x_1, ..., x_m\}$ $(m \in \mathbb{N})$.

$$\exists r_0 \in \mathbb{N},\ \forall h_0, k_0 \geq r_0,\ \exists s_0 \in \mathbb{N},\ x\overrightarrow{M}[O^{h_0}/x_1, ..., O^{h_0}/x_m] \underbrace{O^{h_0}.....O^{h_0}}_{k_0} \to^*_\Phi O^{s_0},$$

$$\exists r_1 \in \mathbb{N},\ \forall h_1, k_1 \geq r_1,\ \exists s_1 \in \mathbb{N},\ N[O^{h_1}/x_1, ..., O^{h_1}/x_m] \underbrace{O^{h_1}.....O^{h_1}}_{k_1} \to^*_\Phi O^{s_1},$$

by hypothesis. So the proof follows by putting $r = \max\{r_0, r_1, n+1\}$. $\square$

The predicate $\mathcal{P}$ is used to define the computability predicate.

Definition 8 *The predicate Comp is defined by induction on types as follows:*

- *$Comp(B, \alpha, M)$ if and only if $\mathcal{P}(M)$, $\alpha \in \mathsf{C}$ and B is a basis;*
- *$Comp(B, \sigma \to \tau, M)$ if and only if, for all $N \in \Lambda$, $Comp(B', \sigma, N)$ implies $Comp(B \cap B', \tau, MN)$;*
- *$Comp(B, \sigma \wedge \tau, M)$ if and only if $Comp(B, \sigma, M)$ and $Comp(B, \tau, M)$.*

We prove that $B \vdash M : \sigma$ implies $Comp(B, \sigma, M)$, which in its turn implies $\mathcal{P}(M)$. It is easy to check that $Comp(B, \sigma, M)$ does not imply $B \vdash M : \sigma$.

Lemma 5 (i) $\mathcal{P}(x\overrightarrow{M})$ *implies* $Comp(B, \sigma, x\overrightarrow{M})$*, for all* B *and* $\sigma \in T(\mathsf{C})$*.*

(ii) $Comp(B, \sigma, M)$ *implies* $\mathcal{P}(M)$*, for all* B *and* $\sigma \in T(\mathsf{C})$*.*

Proof. The proof is by mutual induction on σ. The only case which is not obvious is when $\sigma = \tau \to \rho$.

(i) We will prove that $Comp(B', \tau, N)$ implies $Comp(B \cap B', \rho, x\overrightarrow{M}N)$, thus $Comp(B, \tau \to \rho, x\overrightarrow{M})$ follows by definition. $Comp(B', \tau, N)$ implies $\mathcal{P}(N)$, by induction. But $\mathcal{P}(x\overrightarrow{M})$ by hypothesis, thus $\mathcal{P}(x\overrightarrow{M}N)$ by Property 4.7.ii).

(ii) $\mathcal{P}(z)$ holds so, in particular, $Comp(B', \tau, z)$ holds by induction on i). Thus $Comp(B \cap B', \rho, Mz)$ by definition of $Comp$ and this implies $\mathcal{P}(Mz)$ by induction. That is, there is $r \in \mathbb{N}$ such that, for all $h, k \geq r$,

$$Mz[O^h/z, O^h/x_1, ..., O^h/x_m] \underbrace{O^h.....O^h}_{k} \to^*_\Phi O^s$$

where $s \in \mathbb{N}$, $O^h \equiv \lambda x_0...x_h.x_h$ and $\mathrm{FV}(Mz) \subseteq \{z, x_1, ..., x_m\}$ $(m \geq 0)$. So, if $r' = r + 1$, for all $h, k \geq r'$, $M[O^h/x_1, ..., O^h/x_m] \underbrace{O^h.....O^h}_{k} \to^*_\Phi O^s$,

for some s. Hence $\mathcal{P}(M)$ holds.

□

Property 4.8 *Let* Q *be such that* $\mathcal{P}(Q)$*.*
If $Comp(B, \sigma, P[Q/x]\overrightarrow{Q})$ *then* $Comp(B, \sigma, (\lambda x.P)Q\overrightarrow{Q})$*.*

Proof. The proof is by induction on the structure of types.
Assume $\sigma \in \mathsf{C}$. Note that $\mathcal{P}(Q)$ and Lemma 3.i) imply

$$\exists r_0 \in \mathbb{N},\ \forall h_0, k_0 \geq r_0,\ \ Q[O^{h_0}/x_1, ..., O^{h_0}/x_m] \to^*_\Phi M \in \Upsilon^0 = \Phi^0.$$

Moreover, $Comp(B, \sigma, P[Q/x]\overrightarrow{Q})$ implies $\mathcal{P}(P[Q/x]\overrightarrow{Q})$ by definition, thus

$$\exists r_1,\ \forall h_1, k_1 \geq r_1,\ \exists s_1,\ (P[Q/x]\overrightarrow{Q})[O^{h_1}/x_1, ..., O^{h_1}/x_m] \underbrace{O^{h_1}.....O^{h_1}}_{k_1} \to^*_\Phi O^{s_1}.$$

Let $r = \max\{r_0, r_1\}$. Then, $\forall h, k \geq r$, $((\lambda x.P)Q\overrightarrow{Q})[O^h/x_1, ..., O^h/x_m] \underbrace{O^h.....O^h}_{k}$

$=_\Phi (P[Q/x]\overrightarrow{Q})[O^h/x_1, ..., O^h/x_m] \underbrace{O^h.....O^h}_{k} \to^*_\Phi O^s$, for some $s \in \mathbb{N}$.

Hence $\mathcal{P}(Q)$ and $\sigma \in \mathsf{C}$ imply $Comp(B, \sigma, (\lambda x.P)Q\overrightarrow{Q})$.
Let $\sigma = \tau \to \rho$. Thus $Comp(B, \tau \to \rho, P[Q/x]\overrightarrow{Q})$ implies that $\forall N$ such that $Comp(B', \tau, N)$, $Comp(B \cap B', \rho, P[Q/x]\overrightarrow{Q}N)$ holds. Therefore by induction, $Comp(B \cap B', \rho, (\lambda x.P)Q\overrightarrow{Q}N)$ and hence $Comp(B \cap B', \tau \to \rho, (\lambda x.P)Q\overrightarrow{Q})$ by definition of $Comp$. The case $\sigma = \tau \wedge \rho$ is trivial. □

Lemma 6 *Let* $\mathrm{FV}(M) \subseteq \{x_1, ..., x_n\}$ *and* $B(x_i) = \sigma_i$ *(*$1 \leq i \leq n$*).*
If $Comp(B_i, \sigma_i, N_i)$ $(1 \leq i \leq n)$ *and* $B \vdash M : \tau$*, then*

$$Comp(B_1 \cap ... \cap B_n, \tau, M[N_1/x_1, ..., N_n/x_n]).$$

Proof. By induction on the derivation d of $B \vdash M : \tau$. The most interesting case is when the last rule applied on d is $(\rightarrow I)$. Let $M \equiv \lambda x.M'$, $\tau = \mu \rightarrow \rho$ and

$$\frac{B[\mu/x] \vdash M' : \rho}{B \vdash \lambda x.M' : \mu \rightarrow \rho}\ {\scriptstyle(\rightarrow I)}$$

Suppose that $Comp(B', \mu, N)$ holds. Then $\mathcal{P}(N)$ holds by Lemma 5.ii). Thus, by induction

$$Comp(B' \cap B_1 \cap ... \cap B_n, \rho, M'[N_1/x_1, ..., N_n/x_n, N/x])$$

and $Comp(B' \cap B_1 \cap ... \cap B_n, \rho, (\lambda x.M'[N_1/x_1, ..., N_n/x_n])N)$ by Property 4.8. Hence, $Comp(B_1 \cap ... \cap B_n, \mu \rightarrow \rho, M[N_1/x_1, ..., N_n/x_n])$ by definition of *Comp*. All other cases follow directly from the inductive hypothesis. □

Theorem 4.9 *M is Λ-strongly normalizing implies that M is Φ-solvable.*

Proof. In [12, 5] it is proved that the system $\vdash$ characterizes the strongly Λ-normalizing terms. So, let $B \vdash M : \sigma$, $\mathrm{FV}(M) \subseteq \{x_1, ..., x_n\}$ and $B(x_i) = \sigma_i$. Since $Comp(B, \sigma_i, x_i)$ $(1 \leq i \leq n)$ by Lemma 5.i), then $Comp(B, \sigma, M)$ by Lemma 6. Thus $\mathcal{P}(M)$ by Lemma 5.ii). The proof follows by Property 4.7.i). □

References

[1] van Bakel S., "Intersection type assignment systems," Theoretical Computer Science, 38(2):246-269, Elsevier, 1997.

[2] Barendregt H.P., "The Lambda Calculus: its syntax and semantics," N.103 in Studies in Logic and the Foundations of Mathematics (revised edition), North-Holland, Amsterdam, 1994.

[3] Coppo M., Dezani-Ciancaglini M., Zacchi M., "Type Theories, Normal Forms and $\mathcal{D}_\infty$ Lambda Models", *Information and Control*, 72, 2, 1987, pp.85-116.

[4] Coppo M., Dezani-Ciancaglini M., "An Extension of the Basic Functionality Theory for the λ-Calculus Notre-Dame Journal of Formal Logic, 21(4), pp. 685-693, October 1980.

[5] Krivine J.L., "Lambda-Calculus, Types and Models", Ellis Horwood Series in Computers and Their Applications. Masson, Paris, and Ellis Horwood, Hemel Hempstead, 1993.

[6] Landin P.J., The mechanical evaluation of expressions. *Computer Journal, 1964*

[7] Paolini L.,“ Call-by-value separability and computability”. ICTCS’01, Restivo, Ronchi Della Rocca, and Roversi, eds, LNCS 2202, Springer-Verlag, 74-89.

[8] Paolini L., Pimentel E., Ronchi Della Rocca S. “Lazy strong normalization”. ITRS ’04, ENTCS, to appear.

[9] Paolini L., Ronchi Della Rocca S., “Call-by-value Solvability,” Theoretical Informatics and Applications, 33(6), 507-534, 1999.

[10] Paolini L., Ronchi Della Rocca S., “The Parametric Parameter Passing λ-calculus”, Information and Computation, 189(1):87-106, 2004.

[11] Plotkin G.D., “Call-by-name, call-by-value and the λ-calculus,” Theoretical Computer Science, 1 125-159, 1975.

[12] Pottinger G., “A type assignment for the strongly normalizable λ-terms”, in To H.B. Curry: essays on combinatory logic, lambda calculus and formalism, pp.561-577, Academic Press, London, 1980.

[13] Ronchi Della Rocca S., Paolini L., “The Parametric λ-calculus. A metamodel for computation”. Texts in Theoretical Computer Science: an EATCS Series, Springer-Verlag, Berlin, 2004.