Normalisation is Insensible to λ-term Identity or Difference

Makoto Tatsuta
National Institute of Informatics
2-1-2 Hitotsubashi, 101-8430 Tokyo, Japan
e-mail: tatsuta@nii.ac.jp

Mariangiola Dezani-Ciancaglini
Dipartimento di Informatica
corso Svizzera 185, 10149 Torino, Italy
e-mail: dezani@di.unito.it

Abstract

This paper analyses the computational behaviour of λ-term applications. The properties we are interested in are weak normalisation (i.e. there is a terminating reduction) and strong normalisation (i.e. all reductions are terminating).

One can prove that the application of a λ-term M to a fixed number n of copies of the same arbitrary strongly normalising λ-term is strongly normalising if and only if the application of M to n different arbitrary strongly normalising λ-terms is strongly normalising. I.e. one has that $M\underbrace{X\dots X}_{n}$ is strongly normalising, for an arbitrary strongly normalising X, if and only if $MX_1\dots X_n$ is strongly normalising for arbitrary strongly normalising $X_1,\dots,X_n$. The analogous property holds when replacing strongly normalising by weakly normalising.

As an application of the result on strong normalisation the λ-terms whose interpretation is the top element (in the environment which associates the top element to all variables) of the Honsell-Lenisa model turn out to be exactly the λ-terms which, applied to an arbitrary number of strongly normalising λ-terms, always produces strongly normalising λ-terms. This proof uses a finitary logical description of the model by means of intersection types. This answers an open question stated by Dezani, Honsell and Motohama.

1 Introduction

In the present paper we analyse when a weakly or strongly normalising λ-term preserves this property under application to a fixed or arbitrary number of λ-terms having the same property. The rather surprising result is the independence of this property from the arguments being copies of the same λ-term or being arbitrary λ-terms.

As the first simple illustration we consider the *head normalisation* or solvability property without restrictions on the arguments. We recall that a λ-term M is solvable if M reduces to a λ-term of the shape $\lambda x_1\dots\lambda x_p.yM_1\dots M_m$ [2, 6]. So the question when the application of M to arbitrary λ-terms has head normal form can be simply answered by cases on the head variable y. If y is free, then for all integers n and all λ-terms $X_1,\dots,X_n$ the λ-term $MX_1\dots X_n$ reduces to an head normal form with the same head variable y. Instead, if y is bound, i.e. there is an index i such that $x_i{=}y$, then $MX_1\dots X_n$ reduces to an head normal form for all λ-terms $X_1,\dots,X_n$ only if $n < i$. In fact when $n \geq i$ we can choose $X_i = (\lambda x.xx)(\lambda x.xx)$ and clearly $MX_1\dots X_n$ is an unsolvable term. Since we use only one argument (i.e. the i-th argument) as an immediate consequence we get that:

$$(1)\qquad \forall\ \lambda\text{-term } X.M\underbrace{X\dots X}_{n} \text{ is solvable iff}$$
$$\forall\ \lambda\text{-terms } X_1,\dots,X_n.MX_1\dots X_n \text{ is solvable.}$$

So we claim that head normalisation is insensible to having in applications the same repeated argument or different arguments, when the arguments are arbitrary λ-terms.

Instead, if we require the arguments to be also head normalising λ-terms, we loose this property. As a counterexample, consider the combinator $\mathbf{B} = \lambda xyz.x(yz)$ and $n = 2$. The application $\mathbf{B}\Delta(\mathbf{K}\Delta)$, where $\Delta = \lambda x.xx$ and $\mathbf{K} = \lambda xy.x$, reduces to $\lambda z.\Delta\Delta$, but there is no solvable X such that $\mathbf{B}XX$ is unsolvable. In fact let $\lambda x.X'$ be the head normal form of X. If the head variable of X' is x, then $\mathbf{B}XX$ reduces to $\lambda z.Y$ and the head variable of Y is z. Otherwise, if the head variable of X' is different from x — let it be y —, then $\mathbf{B}XX$ reduces to a λ-term whose head variable is y. So in both cases $\mathbf{B}XX$ is a solvable term.

We recall that a λ-term is *weakly normalising* if it has a finite reduction sequence to normal form, and it is *strongly normalising* if all reduction sequences are finite [2, 6]. We denote by WN and SN the sets of weakly normalising and strongly normalising λ-terms, respectively.

The main result of the paper is that weak normalisation and strong normalisation are insensible to having the same repeated argument or different arguments in WN and SN, respectively, i.e. we get:

$$(2)\qquad \forall\ X \in \mathsf{WN}.M\underbrace{X\dots X}_{n} \in \mathsf{WN} \text{ iff}$$
$$\forall\ X_1,\dots,X_n \in \mathsf{WN}.MX_1\dots X_n \in \mathsf{WN}$$

(3) $\quad \forall\, X \in \mathsf{SN}. M\underbrace{X \ldots X}_{n} \in \mathsf{SN}$ iff

$\forall\, X_1, \ldots, X_n \in \mathsf{SN}. MX_1 \ldots X_n \in \mathsf{SN}$

We will show the following statements:

if $M[x_i := X, x_j := X] \in \mathsf{WN}$ (resp. SN) for all $X \in \mathsf{WN}$ (resp. SN) and for all i, j $(1 \leq i, j \leq n)$, then $M[x_1 := X_1, \ldots, x_n := X_n] \in \mathsf{WN}$ (resp. SN) for all $X_1, \ldots, X_n \in \mathsf{WN}$ (resp. SN).

We call these statements "Substitution Theorems", since they allow to substitute different λ-terms in WN (resp. SN), instead of the same λ-term in WN (resp. SN) for different variables, preserving the weak (strong) normalisation property. We will get both claims (2) and (3) as corollaries of the corresponding "Substitution Theorems".

For dealing with weak normalisation, we start from the analysis of substitutions of β-normal forms inside β-normal forms done in [4]. Our key result is that if there are $X_1, \ldots, X_n \in \mathsf{WN}$ such that $MX_1 \ldots X_n \notin \mathsf{WN}$, then the β-normal form of M has a particular shape. Moreover, from the same assumption we have that at most two λ-terms X_i, X_j need to be chosen appropriately, while all other λ-terms are arbitrary. We can observe that we can choose $X_i = X_j$ and this allows us to conclude the proof of (2).

For strong normalisation the situation is more involved, since we cannot forget reductions. We then consider Klop's extended λ-calculus [9] (a generalisation of Nederpelt's calculus [10]), where all reductions are non erasing. More precisely we use the variant of Klop's λ-calculus discussed in [3]: we call it λ^*-calculus. In λ^*-calculus a redex $(\lambda x.M)N$, with x not in the free variables of M, reduces to the pair $[M, N]$, instead of reducing to M. In this way no subterm is discarded, and strong normalisation coincides with weak normalisation as proved in [3].

Let κ-normal forms be the normal forms of λ^*-calculus: we generalise to κ-normal forms the analysis of substitutions for λ-calculus done in the weak normalisation case. This generalisation is not trivial and requires to associate to each κ-normal form a set of β-normal forms in λ-calculus. In this way we are able to show (3).

As an application of (3) we will prove the completeness of an inductive definition of strongly normalising and persistently strongly normalising λ-terms, where a λ-term M is persistently strongly normalising if for all n and all $X_1, \ldots, X_n \in \mathsf{SN}$ we get $MX_1 \ldots X_n \in \mathsf{SN}$. This inductive definition, which generalises the inductive definition of strongly normalising λ-terms given in [11], is interesting in itself. By means of this inductive definition, we can show that the persistently strongly normalising λ-terms are exactly the λ-terms which Honsell-Lenisa model [7] interprets as the top element, in the environment which associates the top element to all variables. Since for this proof we use a finitary logical description of Honsell-Lenisa model through intersection types, we will also obtain a characterisation of persistently strongly normalising λ-terms by means of typing. Such a characterisation was stated as an open problem in the conclusion of [4].

The present paper is organised as follows: Section 2 shows the "Substitution Theorem" for weak normalisation using a detailed analysis on how variables are substituted inside β-normal forms. Section 3 introduces the λ^*-calculus and associates to each λ^*-term a set of λ-terms. This allows us to prove the "Substitution Theorem" for strong normalisation. Section 4 presents the inductive definition of persistently strongly normalising terms and proves its soundness and completeness by using the "Substitution Theorem" for strong normalisation. Section 5 discusses Honsell-Lenisa model, and shows the characterisation of persistently strongly normalising λ-terms in that model via intersection types.

2 Weak Normalisation

We assume the standard definitions of λ-calculus and β-reduction, see [2], [6].

We use $x, y, z, w, t, u, v, \ldots$ to range over variables, and $M, N, L, P, X, Y, \ldots$ to range over λ-terms. We denote by Λ the set of λ-terms. In writing λ-terms we use vector notation as in [2], i.e. $\vec{M}$ denotes the sequence $M_1, M_2, \ldots, M_n$ for $n \geq 0$ and $\vec{M}$ is empty when $n = 0$; $\lambda\overrightarrow{x}.M\overrightarrow{N}$ is short for $\lambda x_1 \ldots x_n.MN_1 \ldots N_m$, where $\overrightarrow{x}$ is $x_1, \ldots, x_n$ and $\overrightarrow{N}$ is $N_1, \ldots, N_m$. If $\mathcal{A}$ is a set of λ-terms, $\vec{M} \in \mathcal{A}$ means $N \in \mathcal{A}$ for all $N \in \vec{M}$. By $M[x_1 := N_1, \ldots, x_n := N_n]$ and $M[\overrightarrow{x} := \overrightarrow{N}]$ we denote simultaneous capture-free substitutions. We use $\mathrm{lh}(\ )$ to denote the vector length. We assume the Barendregt convention [2] (page 26), i.e. that all names of bound variables are different from each other and different from those of the free variables.

For reduction: $\rightarrow_\beta$ is one step of β-reduction, and $\rightarrow_\beta^*$ is the reflexive transitive closure of the relation $\rightarrow_\beta$. We use $=$ to denote syntactical equality.

We recall that a λ-term is *weakly normalising* if it has a finite reduction sequence to normal form. Let WN be the *set of weakly normalising* λ-terms.

We want to show the following theorem, which allows us to substitute different λ-terms in WN, instead of the same λ-term in WN, for different variables preserving the weak normalisation property.

Theorem 2.1 (Substitution Theorem for WN)

1. *If* $M[x_i := X, x_j := X] \in \mathsf{WN}$ *for all* $X \in \mathsf{WN}$ *and for all* i, j $(1 \leq i, j \leq n)$, *then* $M[x_1 := X_1, \ldots, x_n := X_n] \in \mathsf{WN}$ *for all* $X_1, \ldots, X_n \in \mathsf{WN}$.
2. *If* $M[x_1 := X, \ldots, x_n := X] \in \mathsf{WN}$ *for all* $X \in \mathsf{WN}$, *then* $M[x_1 := X_1, \ldots, x_n := X_n] \in \mathsf{WN}$ *for all* $X_1, \ldots, X_n \in \mathsf{WN}$.

To this aim, following essentially [4], we say that a λ-term M is *persistently weakly normalising* if for all n and

for all $X_1, \ldots, X_n \in \mathsf{WN}$ we get $MX_1 \ldots X_n \in \mathsf{WN}$. We denote by PWN the set of persistently weakly normalising λ-terms. A useful relation between WN and PWN is that the application of a λ-term in WN to a λ-term in PWN is a λ-term in WN. The proof is given in [4].

Lemma 2.2 *If $M \in \mathsf{WN}$ and $N \in \mathsf{PWN}$ then $MN \in \mathsf{WN}$.*

In the following, we do not allow α-conversion on β-normal forms: this allows us to identify bound variables by their names. Although we use this convention in our proofs, our results hold also for usual λ-calculus with α-conversion. We recall that a β-normal form has the shape $\lambda\overrightarrow{x}.y\overrightarrow{M}$. We denote by β-NF the set of β-normal forms.

We start by introducing the notions of *controls* and of *adjacent controls*. These notions simplify the notions of replacement paths and adjacent replacement paths of [4], respectively. Our aim is to understand which λ-terms we can substitute for free variables in normal forms in order to get non weakly normalising λ-terms. We consider some examples, where we use $\Delta = \lambda z.zz$. If $N_0 = xx$, we can choose $X = \Delta$, obtaining $N_0[x := X] = \Delta\Delta$. Instead, if $N_1 = x(\lambda x_1.x_1 y)$, a suitable choice is $X = \lambda u.u\Delta$ and $Y = \Delta$, since $N_1[x := X, y := Y] = (\lambda u.u\Delta)(\lambda x_1.x_1\Delta) \to_\beta (\lambda x_1.x_1\Delta)\Delta \to_\beta \Delta\Delta$. It is important to notice that Δ is substituted for the (bound) variable x_1, since $\lambda u.u\Delta$ is substituted for the (free) variable x. Really we can substitute an arbitrary λ-term X_1 for x_1 in N_1 just by putting $X = \lambda u.uX_1$. We can then say that the (free) variable x *controls* the (bound) variable x_1 (notation $x \rightsquigarrow x_1$) in N_1. As another example we consider $N_2 = x(\lambda x_1.x_1(\lambda x_2.x_2 y))$, for which a suitable choice is $X = \lambda v.v(\lambda u.u\Delta)$ and $Y = \Delta$. In N_2 the (free) variable x controls the (bound) variable x_1, and the (bound) variable x_1 controls the (bound) variable x_2: by transitivity the (free) variable x controls the (bound) variable x_2. Also in this example we can substitute an arbitrary λ-term X_2 for x_2 in N_2 just by putting $X = \lambda v.v(\lambda u.uX_2)$. In the above examples the two variable occurrences which "produce" $\Delta\Delta$ are xx, $x_1 y$ and $x_2 y$, respectively. We can find appropriate substitutions also when the variable occurrences are separated by applications and/or abstractions, the necessary and sufficient requirement being that they occur as function and as argument, respectively. In this case we say that their controls are *adjacent*. For example, let $N_3 = xz(\lambda t.y)$: the controls $x \rightsquigarrow x$ and $y \rightsquigarrow y$ are adjacent in N_3. A suitable choice for N_3 is $X = \lambda uv.vu\Delta$ and $Y = \Delta$.

Definition 2.3 (Control) *1. The relation x* controls *y ($x \rightsquigarrow y$) in $N \in \beta$-NF is the smallest reflexive and transitive relation such that if $x\overrightarrow{L}(\lambda\overrightarrow{t}\,y.M)$ is a subterm of N then $x \rightsquigarrow y$ in N.*

2. If x controls y in $N \in \beta$-NF, then $x \rightsquigarrow y$ is a control *in $N \in \beta$-NF.*

Notice that if $x \rightsquigarrow y$ is a control in N then x occurs in N and y is bound in N, but y could not occur in N.

Example 2.4 *The set of controls in $N = \lambda y.x(\lambda v.vvy)(\lambda t.t(\lambda uz.xz))$ is $x \rightsquigarrow x$, $y \rightsquigarrow y$, $v \rightsquigarrow v$, $t \rightsquigarrow t$, $u \rightsquigarrow u$, $z \rightsquigarrow z$,$x \rightsquigarrow v$, $x \rightsquigarrow t$, $t \rightsquigarrow u$, $t \rightsquigarrow z$, $x \rightsquigarrow u$, $x \rightsquigarrow z$.*

Definition 2.5 *1. Two controls $x \rightsquigarrow z$ and $y \rightsquigarrow t$ in $N \in \beta$-NF are* adjacent *for $\{x, y\}$ in N if x, y are free in N and N contains a subterm of the shape $z\overrightarrow{M}(\lambda\overrightarrow{u}.t\overrightarrow{L})$.*

2. There are adjacent controls for $\overrightarrow{x}$ in $N \in \beta$-NF if there are adjacent controls for $\{x_i, x_j\}$ in N where $x_i, x_j \in \overrightarrow{x}$.

Notice that in the point (1) of the previous definition we do not require x, y, z, t be different variable names, so the set $\{x, y\}$ can contain only one element and the two controls can coincide. Moreover notice that x, y, z, t occur in N.

Example 2.6 *If N is defined as in Example 2.4, then the pairs of adjacent controls for $\{x\}$ are: $x \rightsquigarrow x$ and $x \rightsquigarrow v$, $x \rightsquigarrow v$ and $x \rightsquigarrow v$, $x \rightsquigarrow x$ and $x \rightsquigarrow t$, $x \rightsquigarrow t$ and $x \rightsquigarrow x$, $x \rightsquigarrow x$ and $x \rightsquigarrow z$.*

The first relation between adjacent controls and weak normalisation is stated in the following lemma, which is Lemma A.13 of [4] modulo the obvious mapping between controls and replacement paths.

Lemma 2.7 *If there are adjacent controls for $\{x\}$ in $N \in \beta$-NF, then there is $X \in \beta$-NF such that $N[x := X] \notin \mathsf{WN}$.*

Example 2.8 *If N is defined as in Example 2.4, and $X = \lambda w.w\Delta$ and $\Delta = \lambda r.rr$, then $N[x := X] = \lambda y.X(\lambda v.vvy)(\lambda t.t(\lambda uz.xz)) \to_\beta \lambda y.(\lambda v.vvy)\Delta(\lambda t.t(\lambda uz.xz)) \to_\beta \lambda y.\Delta\Delta y(\lambda t.t(\lambda uz.xz))$.*

From the previous lemma we get:

Lemma 2.9 *If N is in β-NF and for all $X \in \mathsf{WN}$ we have $N[x := X, y := X] \in \mathsf{WN}$, then there are no adjacent controls for $\{x, y\}$ in N.*

PROOF. Assume that there are adjacent controls for $\{x, y\}$ in N, then there are adjacent controls for $\{x\}$ in $N[y := x]$. By Lemma 2.7, there is $X \in \mathsf{WN}$ such that $N[y := x][x := X] \notin \mathsf{WN}$, that is, $N[x := X, y := X] \notin \mathsf{WN}$, and this gives a contradiction. □

The key result for proving the "Substitution Theorem" is the reverse of Lemma 2.9. The formulation of the following lemma underlines that, in order to characterise the β-normal forms which remain in WN after variable substitutions, we need also to characterise the β-normal forms which remain in PWN after variable substitutions.

Lemma 2.10 (Key Lemma) *1. If there are no adjacent controls for $\overrightarrow{x}$ in $N \in \beta$-NF, then $N[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{WN}$ for all $\overrightarrow{X} \in \mathsf{WN}$.*

2. *If* $N = \lambda\overrightarrow{y}.z\overrightarrow{L} \in \beta$-NF, *and there are no adjacent controls for* $\overrightarrow{x}, \overrightarrow{y}$ *in* $\overrightarrow{L}$, *and* $z \notin \overrightarrow{x}, \overrightarrow{y}$, *then* $N[\overrightarrow{x} := \overrightarrow{X}] \in$ PWN *for all* $\overrightarrow{X} \in$ WN.

PROOF. We show (1) and (2) simultaneously by induction on N. Let $\overrightarrow{x} = x_1, \ldots, x_n$ and $N = \lambda\overrightarrow{y}.z\overrightarrow{L}$.

If $z \notin \overrightarrow{x}$ we have $N[\overrightarrow{x} := \overrightarrow{X}] = \lambda\overrightarrow{y}.z\overrightarrow{L}[\overrightarrow{x} := \overrightarrow{X}]$. Since there are no adjacent controls for $\overrightarrow{x}$ in $\overrightarrow{L}$, by induction hypothesis on (1) $\overrightarrow{L}[\overrightarrow{x} := \overrightarrow{X}] \in$ WN, so we are done for (1). Moreover, if $z \notin \overrightarrow{y}$, there are no adjacent controls for $\overrightarrow{x}, \overrightarrow{y}$ in $\overrightarrow{L}$ and $\mathrm{lh}(\overrightarrow{y}) = \mathrm{lh}(\overrightarrow{Y})$, we have $N[\overrightarrow{x} := \overrightarrow{X}]\overrightarrow{Y}\overrightarrow{Z} = z\overrightarrow{L}[\overrightarrow{x} := \overrightarrow{X}][\overrightarrow{y} := \overrightarrow{Y}]\overrightarrow{Z}$ for all $\overrightarrow{Y}, \overrightarrow{Z} \in$ WN. By induction hypothesis on (1) $\overrightarrow{L}[\overrightarrow{x} := \overrightarrow{X}][\overrightarrow{y} := \overrightarrow{Y}] \in$ WN, and therefore we conclude (2).

If $z = x_h$ $(1 \leq h \leq n)$ we have $N[\overrightarrow{x} := \overrightarrow{X}] = \lambda\overrightarrow{y}.X_h\overrightarrow{L}[\overrightarrow{x} := \overrightarrow{X}]$. Let $Q = \lambda\overrightarrow{u}.v\overrightarrow{P} \in \overrightarrow{L}$ with $\mathrm{lh}(\overrightarrow{u}) = m$. We can observe that $v \notin \overrightarrow{x}, \overrightarrow{u}$, since otherwise $x_h \rightsquigarrow x_h$ and $v \rightsquigarrow v$ or $x_h \rightsquigarrow x_h$ and $x_h \rightsquigarrow v$ would be adjacent controls for $\overrightarrow{x}$ in N. Moreover, there are no adjacent controls $u_i \rightsquigarrow w_1$ and $u_j \rightsquigarrow w_2$ for $\{u_i, u_j\}$ $(1 \leq i, j \leq m)$ in $\overrightarrow{P}$, since otherwise $x_h \rightsquigarrow w_1$ and $x_h \rightsquigarrow w_2$ would be adjacent controls for $\{x_h\}$ in N. Lastly, there are no adjacent controls $x_i \rightsquigarrow w_1$ and $u_j \rightsquigarrow w_2$ for $\{x_i, u_j\}$ $(1 \leq j \leq n, 1 \leq j \leq m)$ in $\overrightarrow{P}$, since otherwise $x_i \rightsquigarrow w_1$ and $x_h \rightsquigarrow w_2$ would be adjacent controls for $\{x_i, x_h\}$ in N. Thus, there are no adjacent controls for $\overrightarrow{x}, \overrightarrow{u}$ in $\overrightarrow{P}$. Therefore by induction hypothesis on (2) we get $Q[\overrightarrow{x} := \overrightarrow{X}] \in$ PWN. From Lemma 2.2 we conclude $X_h\overrightarrow{L}[\overrightarrow{x} := \overrightarrow{X}] \in$ WN for all for all $X_1, \ldots, X_n \in$ WN. □

We can now show the "Substitution Theorem".

PROOF OF THEOREM 2.1. We show (1), being the proof of (2) almost the same. Let $\overrightarrow{x} = x_1, \ldots, x_n$ and N be the β-normal form of M. By Lemma 2.9 there are no adjacent controls for any $\{x_i, x_j\}$ $(1 \leq i, j \leq n)$ in N. Hence there are no adjacent controls for $\overrightarrow{x}$. By Lemma 2.10(1) we get $N[\overrightarrow{x} := \overrightarrow{X}] \in$ WN for all $\overrightarrow{X} \in$ WN. Since $M[\overrightarrow{x} := \overrightarrow{X}]$ reduces to $N[\overrightarrow{x} := \overrightarrow{X}]$, we have $M[\overrightarrow{x} := \overrightarrow{X}] \in$ WN for all $\overrightarrow{X} \in$ WN. □

From Theorem 2.1(2) we get the claim (2) of the introduction

Corollary 2.11 *If* $M\underbrace{X \ldots X}_{n} \in$ WN *for all* $X \in$ WN, *then* $MX_1 \ldots X_n \in$ WN *for all* $X_1, \ldots, X_n \in$ WN.

3 Strong Normalisation

We recall that a λ-term is *strongly normalising* if all reductions starting from it are finite. Let SN be the *set of strongly normalising* λ-terms.

In the proof of the "Substitution Theorem" for weak normalisation, Theorem 2.1, the β-normal forms have played a crucial role. The reason is that in studying weak normalisation we can freely substitute the β-normal form of a weakly normalising λ-term for the term itself, and look at its adjacent controls. This is not true for strong normalisation, since strong normalisation is not invariant under β-conversion. So to properly deal with strong normalisation we need to consider "non-erasing" reductions in the λ-calculus. There are many ways of formalising "non-erasing" reductions in the λ-calculus, for a survey see [12] and [8]. We consider the variant of Klop's calculus [9] proposed by Boudol in [3]. We call it λ^*-calculus.

The present section is organised as follows: first we introduce λ^*-calculus in Subsection 3.1, then we associate to each λ^*-term a set of λ-terms (called "projections") which share computational properties with the original λ^*-term (Subsection 3.2) and lastly we show the "Substitution Theorem" for strong normalisation (Subsection 3.3).

3.1 λ^*-calculus

Following [9], we extend the syntax of λ-terms with a pairing operator $[\ ,\ ]$, i.e. we have the following syntax for the set Λ^* of λ^*-terms:

$$S ::= x \mid \lambda x.S \mid SS \mid [S, S].$$

We use $S, T, U, V, P, Q, R, K, X, Y$ to range over λ^*-terms. Let $[S, T_1, \ldots, T_n]$ and $[S, \overrightarrow{T}]$ be short for $[\ldots[[S, T_1], T_2], \ldots T_n]$: they become S for $n = 0$ and $\overrightarrow{T}$ empty.

On Λ^* Boudol [3] defines the following reduction rules:

$$[\lambda x.S, U_1, \ldots, U_n]T \to_\kappa [S[x := T], U_1, \ldots, U_n] \quad \text{if } x \in \mathrm{FV}(S)$$
$$[\lambda x.S, U_1, \ldots, U_n]T \to_\kappa [S, U_1, \ldots, U_n, T] \quad \text{if } x \notin \mathrm{FV}(S)$$

where $\mathrm{FV}(\ )$ is the set of the free variables.

The relation $\to_\kappa$ is the contextual closure of these rules, and the relation $\to_\kappa^*$ is the reflexive transitive closure of $\to_\kappa$.

For example $\mathbf{BO}\Delta \to_\kappa (\lambda yz.\mathbf{O}(yz))\Delta \to_\kappa \lambda z.\mathbf{O}(\Delta z) \to_\kappa \lambda z.[\lambda t.t, \Delta z] \to_\kappa \lambda z.[\lambda t.t, zz]$, where $\mathbf{B} = \lambda xyz.x(yz)$, $\mathbf{O} = \lambda vt.t$, $\Delta = \lambda u.uu$.

A λ^*-term K is a κ-normal form if there does not exist a λ^*-term S such that $K \to_\kappa S$. We use κ-NF to denote the set of κ-normal forms.

The notions of weakly and strongly κ-normalizing λ^*-terms are defined in a similar way to the corresponding notions for λ-terms. WN^* and SN^* are the set of weakly and strongly κ-normalizing λ^*-terms, respectively. In [3] Boudol showed:

Theorem 3.1 *1.* $\mathsf{SN}^* = \mathsf{WN}^*$.

2. $\mathsf{SN} \supseteq \mathsf{SN}^* \cap \Lambda$.

As expected the reverse of Theorem 3.1(2) also holds, as proved in Theorem 3.3 using the following lemma.

Lemma 3.2 *$S\overrightarrow{U} \in \mathsf{SN}^*$ and $T \in \mathsf{SN}^*$ imply $[S,T]\overrightarrow{U} \in \mathsf{SN}^*$.*

PROOF. Assume ad absurd that there is an infinite reduction out of $[S,T]\overrightarrow{U}$. This implies that there is an infinite reduction out of either $S\overrightarrow{U}$ or T. □

Theorem 3.3 $\mathsf{SN} \subseteq \mathsf{SN}^*$.

PROOF. The proof is by a double induction on the longest β-reduction of M to normal form (denoted by $|M|$) and on the structure of M.

If $M = \lambda\overrightarrow{x}.x\overrightarrow{N}$, then $N \in \mathsf{SN}$ for all $N \in \overrightarrow{N}$. Since $|N| \leq |M|$, by induction hypothesis we get $N \in \mathsf{SN}^*$, so we conclude $M \in \mathsf{SN}^*$.

If $M = \lambda\overrightarrow{x}.(\lambda y.P)Q\overrightarrow{N}$, then $\lambda\overrightarrow{x}.P[y := Q]\overrightarrow{N} \in \mathsf{SN}$ and $Q \in \mathsf{SN}$. Since $|\lambda\overrightarrow{x}.P[y := Q]\overrightarrow{N}| < |M|$ and $|Q| < |M|$, by induction hypothesis we get $\lambda\overrightarrow{x}.P[y := Q]\overrightarrow{N} \in \mathsf{SN}^*$ and $Q \in \mathsf{SN}^*$. If $y \in \mathrm{FV}(P)$, we have $M \to_\kappa \lambda\overrightarrow{x}.P[y := Q]\overrightarrow{N}$, which implies $M \in \mathsf{SN}^*$ by Theorem 3.1(1). Otherwise, if $y \notin \mathrm{FV}(P)$, we have $M \to_\kappa \lambda\overrightarrow{x}.[P,Q]\overrightarrow{N}$, which implies $M \in \mathsf{SN}^*$ by Lemma 3.2. □

Analogously to the weak case, we also consider the set PSN^* of *persistently strongly normalising* λ^*-terms. A λ^*-term S is persistently strongly normalising if S preserves the strong κ-normalisation property under application to strongly κ-normalising λ^*-terms, i.e. $S \in \mathsf{PSN}^*$ if for all n and all $X_1, \ldots, X_n \in \mathsf{SN}^*$, we get $SX_1 \ldots X_n \in \mathsf{SN}^*$.

We need to show that the pairing of a λ^*-term in PSN^* with a λ^*-term in SN^* remains in PSN^* and that the application of a λ^*-term in SN^* to a λ^*-term in PSN^* remains in SN^*: these are the claims of the following lemma, respectively.

Lemma 3.4 *1. $S \in \mathsf{PSN}^*$ and $T \in \mathsf{SN}^*$ imply $[S,T] \in \mathsf{PSN}^*$.*

2. $S \in \mathsf{SN}^*$ and $T \in \mathsf{PSN}^*$ imply $ST \in \mathsf{SN}^*$.

PROOF. (1) Immediate from Lemma 3.2.

(2) First we show that $U \in \mathsf{SN}^*$ and $T \in \mathsf{PSN}^*$ imply $U[x := T] \in \mathsf{SN}^*$. By Theorem 3.1(1), we can suppose $U, T \in \kappa\text{-}\mathsf{NF}$. Using (1) and by structural induction on U we have the claim. Then (2) is immediate by letting U be Sx, where x is fresh. □

Clearly Lemma 3.4(2) is analogous to Lemma 2.2.

3.2 Projections

In this subsection we associate to each λ^*-term S a set of λ-terms (called the *projections* of S) which are built out of subterms of S in order to represent all possible κ-reductions of S by β-reductions.

In the next definition we use the following notation for sets $\mathcal{A}, \mathcal{B}$ of λ-terms:

$$\begin{array}{lcl} \lambda x.\mathcal{A} & = & \{\lambda x.M \mid M \in \mathcal{A}\}, \\ \mathcal{A}\mathcal{B} & = & \{MN \mid M \in \mathcal{A}, N \in \mathcal{B}\}, \\ a\mathcal{A} & = & \{aM \mid M \in \mathcal{A}\}. \end{array}$$

Definition 3.5 *Fix a fresh variable a. The set of projections of S with respect to a (notation $\mathcal{P}_a(S)$) is defined by induction on S as follows:*

$$\begin{array}{lcl} \mathcal{P}_a(x) & = & \{x\} \\ \mathcal{P}_a(\lambda x.S) & = & \lambda x.\mathcal{P}_a(S) \\ \mathcal{P}_a(ST) & = & \mathcal{P}_a(S)\mathcal{P}_a(T) \\ \mathcal{P}_a([S,T]) & = & \mathcal{P}_a(S) \cup a\mathcal{P}_a(T). \end{array}$$

For example, we have $\mathcal{P}_a(\lambda x.[x,y][zx,wx]) = \{\lambda x.x(zx), \lambda x.x(a(wx)), \lambda x.ay(zx), \lambda x.ay(a(wx))\}$.

We want to prove that projections agree with substitutions and that if a κ-term S has a projection which is not strongly β-normalising, then also S is not strongly κ-normalising. These are the claims of the following lemma.

Lemma 3.6 *1. If $M \in \Lambda$, then $\mathcal{P}_a(M) = \{M\}$.*

2. If $M \in \mathcal{P}_a(S)$ and $N \in \mathcal{P}_a(T)$, then $M[x := N] \in \mathcal{P}_a(S[x := T])$.

3. If $S \in \mathsf{SN}^*$, then $\mathcal{P}_a(S) \subseteq \mathsf{SN}$.

PROOF. (1) is immediate by definition.

(2) is easily proved by induction on S.

(3) First we will show the auxiliary claim: if $M \in \mathcal{P}_a(S)$ and $M \to_\beta M'$, then there exists S' such that $S \to_\kappa S'$ and $M' \in \mathcal{P}_a(S')$. This is proved by induction on S. The interesting case is $S = S_1S_2$, $M = (\lambda x.M_3)M_2$, and $M' = M_3[x := M_2]$. Then we have $\lambda x.M_3 \in \mathcal{P}_a(S_1)$ and $M_2 \in \mathcal{P}_a(S_2)$. Hence $S_1 = [\lambda x.S_3, \overrightarrow{S_4}]$ and $M_3 \in \mathcal{P}_a(S_3)$. Case 1. If $x \in \mathrm{FV}(S_3)$, we can choose $S' = [S_3[x := S_2], \overrightarrow{S_4}]$. By (2), $M' \in \mathcal{P}_a(S_3[x := S_2]) \subseteq \mathcal{P}_a(S')$.
Case 2. If $x \notin \mathrm{FV}(S_3)$, then $x \notin \mathrm{FV}(M_3)$, because we can easily show by induction on S that $x \in \mathrm{FV}(M) \in \mathcal{P}_a(S)$ implies $x \in \mathrm{FV}(S)$. Then we have $M' = M_3$ and we can choose $S' = [S_3, \overrightarrow{S_4}, S_2]$. This concludes the proof of the auxiliary claim.

Assume $M \in \mathcal{P}_a(S)$ and $M \notin \mathsf{SN}$. We have an infinite reduction sequence $M = M_0 \to_\beta M_1 \to_\beta M_2 \to_\beta \ldots$. By the auxiliary claim, we have an infinite reduction sequence $S = S_0 \to_\kappa S_1 \to_\kappa S_2 \to_\kappa \ldots$ such that $M_i \in \mathcal{P}_a(S_i)$ for all i. Hence $S \notin \mathsf{SN}^*$. □

3.3 Substitution Theorem

Some further notational conventions are handy in the following treatment. We use $\lambda_{\overrightarrow{T}}\overrightarrow{x}.S$ as short for $[\lambda\overrightarrow{x}.S, \overrightarrow{T}]$: it becomes $[S, \overrightarrow{T}]$ for $\overrightarrow{x}$ empty. We *ambiguously* denote by $S\overrightarrow{T}$ either ST

or $[S,T]$. For example, $[\lambda_{S_1,S_2}xy.xS_3\overline{S_4}S_5,S_6]$ stands for $[[[\lambda xy.xS_3S_4S_5,S_1],S_2],S_6]$ or $[[[\lambda xy.[xS_3,S_4]S_5,S_1],S_2],S_6]$.

We can characterise the set of κ-normal forms K by

$$K ::= J \mid \lambda x.K \mid [K,K]$$
$$J ::= x \mid JK \mid [J,K]$$

One can show by induction on the definitions that these λ^*-terms are irreducible and by induction on Λ^* that all irreducible λ^*-terms have this shape. By the vector notation, we have $K ::= \overrightarrow{\lambda_K \overrightarrow{x}}.J$. By the bar notation, we have $J ::= x \mid J\overline{K}$. Again by the vector notation, we have $J ::= x\overrightarrow{\overline{K}}$. Finally we get the following lemma, which characterises the shape of κ-normal forms.

Lemma 3.7 *The set κ-NF is defined by*

$$K ::= \overrightarrow{\lambda_{\overrightarrow{K}} \overrightarrow{x}}.x\overrightarrow{\overline{K}}.$$

For example, $\lambda x.\lambda_x y.\lambda_y z.z\overline{yx}$ represents one of the following κ-normal forms: $\lambda x.[\lambda y.[\lambda z.zyx,y],x]$, $\lambda x.[\lambda y.[\lambda z.[z,y]x,y],x]$, $\lambda x.[\lambda y.[\lambda z.[zy,x],y],x]$, and $\lambda x.[\lambda y.[\lambda z.[[z,y],x],y],x]$.

The next lemma (easily proved by structural induction) gives a clue to analyse controls in κ-NF through controls in β-NF.

Lemma 3.8 *If $K \in \kappa$-NF, then $\mathcal{P}_a(K) \subseteq \beta$-NF.*

We can then define the set of controls in a κ-normal form as the union of the sets of controls in its projections.

Definition 3.9 *The* adjacent controls *for $\overrightarrow{x}$ in K are the adjacent controls for $\overrightarrow{x}$ in M for some $M \in \mathcal{P}_a(K)$.*

The next lemma extends Lemma 2.9 to λ^*-calculus.

Lemma 3.10 *If $K \in \kappa$-NF and $K[x := X, y = X] \in \mathsf{SN}^*$ for all $X \in \beta$-NF, then there are no adjacent controls for $\{x,y\}$ in K.*

PROOF. Assume ad absurdum that there exists $M \in \mathcal{P}_a(K)$ such that there are adjacent controls for $\{x,y\}$ in M. By Lemma 2.9, there exists $X \in \beta$-NF such that $M[x := X, y := X] \notin \mathsf{WN}$. Then we have $M[x := X, y := X] \notin \mathsf{SN}$. By Lemma 3.6(1), we have $X \in \mathcal{P}_a(X)$. Then by Lemma 3.6(2) we get $M[x := X, y := X] \in \mathcal{P}_a(K[x := X, y := X])$. By Lemma 3.6(3) we conclude $K[x := X, y := X] \notin \mathsf{SN}^*$. □

The next lemma is the key lemma to prove the "Substitution Theorem " for strong normalisation. It is the extension of Lemma 2.10 to κ-normal forms.

Lemma 3.11 (Key Lemma) *1. If there are no adjacent controls for $\overrightarrow{x}$ in $K \in \kappa$-NF, then $K[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{SN}^*$ for all $\overrightarrow{X} \in \mathsf{SN}^*$.*

2. If $K = \overrightarrow{\lambda_{\overrightarrow{T}} \overrightarrow{y}}.z\overrightarrow{\overline{S}}$, there are no adjacent controls for $\overrightarrow{x}, \overrightarrow{y}$ in $\overrightarrow{\overline{T}}, z\overrightarrow{\overline{S}}$, and $z \notin \overrightarrow{x}, \overrightarrow{y}$, then $K[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{PSN}^*$ for all $\overrightarrow{X} \in \mathsf{SN}^*$.

PROOF. We use two mappings from the set of λ^*-terms of the shape $x\overrightarrow{\overline{S}}$ to sequences of λ^*-terms. These mappings, denoted by $^\circ$ and $^\times$, list up the arguments of variable applications and the second components of pairs, respectively. They are defined by induction as follows:

$$x^\circ = \epsilon \qquad x^\times = \epsilon$$
$$(x\overrightarrow{\overline{S}}T)^\circ = (x\overrightarrow{\overline{S}})^\circ, T \qquad (x\overrightarrow{\overline{S}}T)^\times = (x\overrightarrow{\overline{S}})^\times$$
$$([x\overrightarrow{\overline{S}},T])^\circ = (x\overrightarrow{\overline{S}})^\circ \qquad ([x\overrightarrow{\overline{S}},T])^\times = (x\overrightarrow{\overline{S}})^\times, T$$

where ϵ denotes the empty sequence. For example, we have $([[xS_1S_2,S_3]S_4,S_5])^\circ = S_1,S_2,S_4$ and $([[xS_1S_2,S_3]S_4,S_5])^\times = S_3,S_5$.

We show (1) and (2) simultaneously by induction on K. Let $\overrightarrow{x} = x_1,\ldots,x_n$ and $K = \overrightarrow{\lambda_{\overrightarrow{T}} \overrightarrow{y}}.z\overrightarrow{\overline{S}}$. First let us observe that, by the definitions of projections, if $\overrightarrow{\lambda_{\overrightarrow{T}} \overrightarrow{y}}$ is $\lambda_{\overrightarrow{T}_1}\overrightarrow{y}_1\ldots\lambda_{\overrightarrow{T}_m}\overrightarrow{y}_m$, then:

$$\begin{aligned}\mathcal{P}_a(K) \;=\; & \{\lambda\overrightarrow{\overline{y}}\mathcal{P}_a(z\overrightarrow{H}) \mid \overrightarrow{H} = (z\overrightarrow{\overline{S}})^\circ\}\cup \\ & \{\lambda\overrightarrow{\overline{y_1}}\ldots\overrightarrow{\overline{y_{j-1}}}.a\mathcal{P}_a(J) \mid J \in \overrightarrow{T_j}\ (1 \le j \le m)\}\cup \\ & \{\lambda\overrightarrow{\overline{y}}a\mathcal{P}_a(J) \mid J \in (z\overrightarrow{\overline{S}})^\times\}\end{aligned}$$

Therefore, by the definitions of controls:

- if $\overrightarrow{H} = (z\overrightarrow{\overline{S}})^\circ$, then adjacent controls for $\overrightarrow{x}$ in $z\overrightarrow{H}$ are adjacent controls for $\overrightarrow{x}$ in K,
- adjacent controls for $\overrightarrow{x}$ in $\overrightarrow{\overline{T}}$ or in $(z\overrightarrow{\overline{S}})^\times$ are adjacent controls for $\overrightarrow{x}$ in K.

In the following we write U' as short for $U[\overrightarrow{x} := \overrightarrow{X}]$.

If $z \notin \overrightarrow{x}$ we have $K' = \overrightarrow{\lambda_{\overrightarrow{T}} \overrightarrow{y}}.z\overrightarrow{\overline{S'}}$. Since there are no adjacent controls for $\overrightarrow{x}$ in $\overrightarrow{\overline{T}}$ and in $z\overrightarrow{\overline{S}}$, by induction hypothesis on (1) $\overrightarrow{\overline{T'}} \in \mathsf{SN}^*$ and $(z\overrightarrow{\overline{S'}})^\circ, (z\overrightarrow{\overline{S'}})^\times \in \mathsf{SN}^*$, so we conclude (1). Moreover, if $z \notin \overrightarrow{y}$, there are no adjacent controls for $\overrightarrow{x}, \overrightarrow{y}$ in $\overrightarrow{\overline{T}}, z\overrightarrow{\overline{S}}$, and $\mathrm{lh}(\overrightarrow{y}) = \mathrm{lh}(\overrightarrow{Y})$, then for all $\overrightarrow{Y}, \overrightarrow{Z} \in \mathsf{SN}^*$ there exists some subsequence $\overrightarrow{V}$ of $\overrightarrow{Y}$ such that $K'\overrightarrow{Y}\overrightarrow{Z}$ reduces to $[z\overrightarrow{\overline{S''}}, \overrightarrow{\overline{T''_m}}, \ldots, \overrightarrow{\overline{T''_1}}, \overrightarrow{V}]\overrightarrow{Z}$, where U'' stands for $U'[\overrightarrow{y} := \overrightarrow{Y}]$ for any U. By induction hypothesis on (1) $(z\overrightarrow{\overline{S''}})^\circ, (z\overrightarrow{\overline{S''}})^\times, \overrightarrow{\overline{T''}}$ are in SN^*. By using Theorem 3.1 (1), $K'\overrightarrow{Y}\overrightarrow{Z}$ is in SN^*, and therefore we conclude (2).

If $z = x_h$ $(1 \le h \le n)$ we have $K' = \overrightarrow{\lambda_{\overrightarrow{T'}} \overrightarrow{y}}.X_h\overrightarrow{\overline{S'}}$. Notice that in this case the λ^*-terms belonging to $(x_h\overrightarrow{\overline{S}})^\circ$

are arguments of X_h. Let $J \in \overrightarrow{T}, (x_h \overrightarrow{S})^{\times}$: by hypothesis there are no adjacent controls for $\vec{x}$ in J, so by induction hypothesis on (1) we get $J' \in \mathsf{SN}^*$. Let $H = \overrightarrow{\lambda_{\overrightarrow{Q}} \overrightarrow{u}}.v\overrightarrow{P} \in (x_h \overrightarrow{S})^{\circ}$. We can observe that $v \notin \overrightarrow{x}, \overrightarrow{u}$, since otherwise $x_h \rightsquigarrow x_h$ and $v \rightsquigarrow v$ or $x_h \rightsquigarrow x_h$ and $x_h \rightsquigarrow v$ would be adjacent controls for $\overrightarrow{x}$ in K. There are no adjacent controls $u_i \rightsquigarrow w_1$ and $u_j \rightsquigarrow w_2$ for $\{u_i, u_j\}$ $(1 \leq i, j \leq m)$ in $\overrightarrow{Q}, v\overrightarrow{P}$, since otherwise $x_h \rightsquigarrow w_1$ and $x_h \rightsquigarrow w_2$ would be adjacent controls for $\{x_h\}$ in K. Moreover, there are no adjacent controls $x_i \rightsquigarrow w_1$ and $u_j \rightsquigarrow w_2$ for $\{x_i, u_j\}$ $(1 \leq i \leq n, 1 \leq j \leq m)$ in $\overrightarrow{Q}, v\overrightarrow{P}$, since otherwise $x_i \rightsquigarrow w_1$ and $x_h \rightsquigarrow w_2$ would be adjacent controls for $\{x_i, x_h\}$ in K. Thus there are no adjacent controls for $\overrightarrow{x}, \overrightarrow{u}$ in $\overrightarrow{Q}, v\overrightarrow{P}$. Therefore by induction hypothesis on (2) we get $H' \in \mathsf{PSN}^*$. By applying Lemma 3.4(1) to all J' such that $J \in \overrightarrow{T}, (x_h \overrightarrow{S})^{\times}$ and Lemma 3.4(2) to all H' such that $H \in (x_h \overrightarrow{S})^{\circ}$, we conclude $K' \in \mathsf{SN}^*$. □

Theorem 3.12 (Substitution Theorem for SN) *If $M[x_i := X, x_j := X] \in \mathsf{SN}$ for all $X \in \mathsf{SN}$ and for all i, j $(1 \leq i, j \leq n)$, then $M[x_1 := X_1, \ldots, x_n := X_n] \in \mathsf{SN}$ for all $X_1, \ldots, X_n \in \mathsf{SN}$.*

PROOF. Let $\overrightarrow{x} = x_1, \ldots, x_n$. Since $M \in \mathsf{SN}$, by Theorem 3.3 we have $M \in \mathsf{SN}^*$. Then there exists $K \in \kappa\text{-}\mathsf{NF}$ such that $M \to_{\kappa}^* K$. Since $M[x_i := X, x_j := X] \in \mathsf{SN}$ for all i, j $(1 \leq i, j \leq n)$ and all $X \in \beta\text{-}\mathsf{NF}$, by Theorem 3.3, we have $M[x_i := X, x_j := X] \in \mathsf{SN}^*$. Then we get $K[x_i := X, x_j := X] \in \mathsf{SN}^*$. By Lemma 3.10, there are no adjacent controls for $\{x_i, x_j\}$ in K for all i, j $(1 \leq i, j \leq n)$. Hence, there are no adjacent controls for $\overrightarrow{x}$ in K. By Lemma 3.11(1), for arbitrary $\overrightarrow{X} \in \mathsf{SN}^*$, we have $K[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{SN}^*$. By Theorem 3.1(1), we have $K[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{WN}^*$, and hence we have $M[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{WN}^*$. By Theorem 3.1(1), we get $M[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{SN}^*$, and then, by Theorem 3.1(2), we conclude $M[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{SN}$. □

Claim (3) of the introduction immediately follows by letting M be $Mx_1 \ldots x_n$ in previous theorem, since substitution preserves non strong normalisation.

Corollary 3.13 *If $M\underbrace{X \ldots X}_{n} \in \mathsf{SN}$ for all $X \in \mathsf{SN}$, then $MX_1 \ldots X_n \in \mathsf{SN}$ for all $X_1, \ldots, X_n \in \mathsf{SN}$.*

4 Inductive Definitions of PSN

We define the set PSN of *persistent strongly normalising* λ-terms as the set of λ-terms which preserve the strong normalisation property under application to strongly normalising λ-terms, i.e. $M \in \mathsf{PSN}$ if for all $X_1, \ldots, X_n \in \mathsf{SN}$ we get $MX_1 \ldots X_n \in \mathsf{SN}$.

$$\frac{\lambda\overrightarrow{x}.N \in \mathsf{SN}_n^\sharp\ (\forall N \in \overrightarrow{N}) \quad \mathrm{lh}(\overrightarrow{x}) = n \quad y \notin \overrightarrow{x}}{\lambda\overrightarrow{x}.y\overrightarrow{N} \in \mathsf{PSN}^\sharp}$$

$$\frac{\lambda\overrightarrow{x}.M[y := N]\overrightarrow{L} \in \mathsf{PSN}^\sharp \quad \lambda\overrightarrow{x}.N \in \mathsf{SN}_n^\sharp \quad \mathrm{lh}(\overrightarrow{x}) = n}{\lambda\overrightarrow{x}.(\lambda y.M)N\overrightarrow{L} \in \mathsf{PSN}^\sharp}$$

$$\frac{\lambda\overrightarrow{x}.N \in \mathsf{PSN}^\sharp\ (\forall N \in \overrightarrow{N}) \quad \mathrm{lh}(\overrightarrow{x}) = n \quad x \in \overrightarrow{x}}{\lambda\overrightarrow{x}.x\overrightarrow{N} \in \mathsf{SN}_n^\sharp}$$

$$\frac{\lambda\overrightarrow{x}.N \in \mathsf{SN}_m^\sharp\ (\forall N \in \overrightarrow{N}) \quad \mathrm{lh}(\overrightarrow{x}) = m \quad y \notin \overrightarrow{x}}{\lambda\overrightarrow{x}.y\overrightarrow{N} \in \mathsf{SN}_n^\sharp}$$

$$\frac{\lambda\overrightarrow{x}.N \in \mathsf{SN}_n^\sharp \quad \mathrm{lh}(\overrightarrow{x}) = n \quad \mathrm{lh}(\overrightarrow{y}) > 0}{\lambda\overrightarrow{x}\,\overrightarrow{y}.N \in \mathsf{SN}_n^\sharp}$$

$$\frac{\lambda\overrightarrow{x}.M[y := N]\overrightarrow{L} \in \mathsf{SN}_n^\sharp \quad \lambda\overrightarrow{x}.N \in \mathsf{SN}_m^\sharp \quad \mathrm{lh}(\overrightarrow{x}) = m}{\lambda\overrightarrow{x}.(\lambda y.M)N\overrightarrow{L} \in \mathsf{SN}_n^\sharp}$$

Figure 1. Definition of $\mathsf{PSN}^\sharp$ and $\mathsf{SN}_n^\sharp$.

The inductive definition of PSN which we will discuss in this section was inspired by the following inductive definition of SN given in [11]:

$$\frac{\overrightarrow{M} \in \mathsf{SN}}{x\overrightarrow{M} \in \mathsf{SN}} \qquad \frac{M \in \mathsf{SN}}{\lambda x.M \in \mathsf{SN}} \qquad \frac{M[x := N]\overrightarrow{L} \in \mathsf{SN} \quad N \in \mathsf{SN}}{(\lambda x.M)N\overrightarrow{L} \in \mathsf{SN}}$$

It is handy to consider the class SN_n of λ-terms which preserves the strong normalisation property under application to n strongly normalising λ-terms, i.e. $M \in \mathsf{SN}_n$ if for all $X_1, \ldots, X_n \in \mathsf{SN}$ we get $MX_1 \ldots X_n \in \mathsf{SN}$. Clearly $\mathsf{SN}_0 = \mathsf{SN}$. Figure 1 defines the sets $\mathsf{PSN}^\sharp$ and $\mathsf{SN}_n^\sharp$: in the remaining of the present section we will show that $\mathsf{PSN}^\sharp = \mathsf{PSN}$ and $\mathsf{SN}_n^\sharp = \mathsf{SN}_n$.

The first lemma shows easy properties of the sets SN and PSN.

Lemma 4.1 *1. $M \in \mathsf{SN}$ and $N \in \mathsf{PSN}$ imply $MN \in \mathsf{SN}$.*

2. Let $\mathrm{lh}(\overrightarrow{x}) = \mathrm{lh}(\overrightarrow{N})$, then $M[\overrightarrow{x} := \overrightarrow{N}]\overrightarrow{L} \in \mathsf{SN}$ and $\overrightarrow{N} \in \mathsf{SN}$ imply $(\lambda\overrightarrow{x}.M)\overrightarrow{N}\overrightarrow{L} \in \mathsf{SN}$.

PROOF. (1) We can now show by induction on L that $L \in \mathsf{SN}$ and $N \in \mathsf{PSN}$ imply $L[x := N] \in \mathsf{SN}$. Then we conclude taking L as Mx, where x is fresh.

(2) Assume ad absurd that there is an infinite reduction out of $(\lambda\overrightarrow{x}.M)\overrightarrow{N}\overrightarrow{L}$. This implies that there is an infinite reduction either out of $M[\overrightarrow{x} := \overrightarrow{N}]\overrightarrow{L}$ or out of $\overrightarrow{N}$. □

The following lemma, which is the key result for proving the completeness of the given inductive definition, uses in a crucial way the " Substitution Theorem" for SN, Theorem 3.12.

Lemma 4.2 *If* $\lambda\overrightarrow{x}.x\overrightarrow{N} \in \mathsf{SN}_n$, *where* $x \in \overrightarrow{x}$ *and* $\mathrm{lh}(\overrightarrow{x}) = n$, *then* $\lambda\overrightarrow{x}.N \in \mathsf{PSN}$ *for all* $N \in \overrightarrow{N}$.

PROOF. For arbitrary $\overrightarrow{X} \in \mathsf{SN}$ with $\mathrm{lh}(\overrightarrow{X}) = n$, we have $(x\overrightarrow{N})[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{SN}$. Suppose $\mathrm{lh}(\overrightarrow{N}) = m$ and $y \notin x\overrightarrow{N}$. By Theorem 3.12, $(y\overrightarrow{N})[\overrightarrow{x} := \overrightarrow{X}, y := Y] \in \mathsf{SN}$ holds for all $\overrightarrow{X}, Y \in \mathsf{SN}$. For N_i ($1 \leq i \leq m$), we show $(\lambda\overrightarrow{x}.N_i)\overrightarrow{X}\,\overrightarrow{Z} \in \mathsf{SN}$ for arbitrary $\overrightarrow{X}, \overrightarrow{Z} \in \mathsf{SN}$. Let Y be $\lambda\overrightarrow{z}.z_i\overrightarrow{Z}$ and $\mathrm{lh}(\overrightarrow{z}) = m$. Then we have $(y\overrightarrow{N})[\overrightarrow{x} := \overrightarrow{X}, y := Y] = (\lambda\overrightarrow{z}.z_i\overrightarrow{Z})\overrightarrow{N}[\overrightarrow{x} := \overrightarrow{X}] \to_\beta^* N_i[\overrightarrow{x} := \overrightarrow{X}]\overrightarrow{Z}$. Hence $N_i[\overrightarrow{x} := \overrightarrow{X}]\overrightarrow{Z}$ is in SN. By Lemma 4.1(2), we have $(\lambda\overrightarrow{x}.N_i)\overrightarrow{X}\,\overrightarrow{Z} \in \mathsf{SN}$. Therefore $\lambda\overrightarrow{x}.N_i \in \mathsf{PSN}$. □

We can show now the soundness and completeness of the given inductive characterisations.

Theorem 4.3 $\mathsf{PSN}^\sharp = \mathsf{PSN}$ *and* $\mathsf{SN}_n^\sharp = \mathsf{SN}_n$.

PROOF. We will show that *the rules generate ONLY terms which satisfy the given conditions,* that is, $\mathsf{PSN}^\sharp \subseteq \mathsf{PSN}$ and $\mathsf{SN}_n^\sharp \subseteq \mathsf{SN}_n$. This claim is proved by induction on the formation rules. It suffices to show that the premises implies the conclusion in each rule without $\sharp$. For example for the rule

$$\frac{\lambda\overrightarrow{x}.N \in \mathsf{PSN}^\sharp\ (\forall N \in \overrightarrow{N}) \quad \mathrm{lh}(\overrightarrow{x}) = n \quad x \in \overrightarrow{x}}{\lambda\overrightarrow{x}.x\overrightarrow{N} \in \mathsf{SN}_n^\sharp}$$

we will show $(\lambda\overrightarrow{x}.x\overrightarrow{N})\overrightarrow{X} \in \mathsf{SN}$ for all $\overrightarrow{X} \in \mathsf{SN}$ of length n. By induction hypothesis, we have $\lambda\overrightarrow{x}.N \in \mathsf{PSN}$. Then $N[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{PSN}$. Let $x_j = x$. By Lemma 4.1(1), we have $X_j\overrightarrow{N}[\overrightarrow{x} := \overrightarrow{X}] \in \mathsf{SN}$. By Lemma 4.1(2), we have $(\lambda\overrightarrow{x}.x\overrightarrow{N})\overrightarrow{X} \in \mathsf{SN}$.

We will show that *the rules generate ALL terms which satisfy the given conditions,* that is, $\mathsf{PSN}^\sharp \supseteq \mathsf{PSN}$ and $\mathsf{SN}_n^\sharp \supseteq \mathsf{SN}_n$. First notice that the conclusions of the given rules cover all possible shapes of λ-terms, but $\lambda\overrightarrow{x}.x\overrightarrow{N}$ with $\mathrm{lh}(\overrightarrow{x}) = n$ and $x \in \overrightarrow{x}$ for both $\mathsf{PSN}^\sharp$ and $\mathsf{SN}_m^\sharp$ with $n < m$. This is sound since if $\Delta = \lambda x.xx$, $\Delta_k = \lambda y_1 \ldots y_k.\Delta$, $k = \mathrm{lh}(\overrightarrow{N})$, $\mathrm{lh}(\overrightarrow{\Delta_k}) = n$, $\mathrm{lh}(\overrightarrow{\Delta}) = m - n$, then $(\lambda\overrightarrow{x}.x\overrightarrow{N})\overrightarrow{\Delta_k}\overrightarrow{\Delta}$ does not have normal form.

The proof is by a double induction on the length of the longest reduction to normal form and on the structure of terms. We show that if the statement holds for the conclusion then it must hold for the premises in each rule without $\sharp$.

The most interesting case is that of the rule

$$\frac{\lambda\overrightarrow{x}.N \in \mathsf{PSN}\ (\forall N \in \overrightarrow{N}) \quad \mathrm{lh}(\overrightarrow{x}) = n \quad x \in \overrightarrow{x}}{\lambda\overrightarrow{x}.x\overrightarrow{N} \in \mathsf{SN}_n}$$

We begin with $\lambda\overrightarrow{x}.x\overrightarrow{N} \in \mathsf{SN}_n$ and this implies $\lambda\overrightarrow{x}.N \in \mathsf{PSN}$ for all $N \in \overrightarrow{N}$ by Lemma 4.2.

The proofs for the other rules immediately follow by using induction hypothesis. We show the case of the rule

$$\frac{\lambda\overrightarrow{x}.M[y := N]\overrightarrow{L} \in \mathsf{SN}_n \quad \lambda\overrightarrow{x}.N \in \mathsf{SN}_m \quad \mathrm{lh}(\overrightarrow{x}) = m}{\lambda\overrightarrow{x}.(\lambda y.M)N\overrightarrow{L} \in \mathsf{SN}_n}$$

We assume $n \geq m$, the proof for $n < m$ being similar. For all $\overrightarrow{X}, \overrightarrow{Y} \in \mathsf{SN}$ of lengths m, $n - m$, respectively, we get $(\lambda\overrightarrow{x}.(\lambda y.M)N\overrightarrow{L})\overrightarrow{X}\,\overrightarrow{Y} \in \mathsf{SN}$. Then we have $(\lambda y.M')N'\overrightarrow{L}'\overrightarrow{Y} \in \mathsf{SN}$, where P' denotes $P[\overrightarrow{x} := \overrightarrow{X}]$ for every λ-term P. Hence we get $M'[y := N']\overrightarrow{L}'\overrightarrow{Y} \in \mathsf{SN}$ and $N' \in \mathsf{SN}$. By Lemma 4.1(2), we have $(\lambda\overrightarrow{x}.M[y := N]\overrightarrow{L})\overrightarrow{X}\,\overrightarrow{Y} \in \mathsf{SN}$ and $(\lambda\overrightarrow{x}.N)\overrightarrow{X} \in \mathsf{SN}$. Therefore we conclude $\lambda\overrightarrow{x}.M[y := N]\overrightarrow{L} \in \mathsf{SN}_n$ and $\lambda\overrightarrow{x}.N \in \mathsf{SN}_m$.

The induction hypothesis applies since the λ-terms in the premises can either be obtained by reducing the λ-term in the conclusion or are smaller than the λ-term in the conclusion. □

5 The Model $\mathcal{H}L_\infty$

We start by recalling the definition of the $\mathcal{D}_\infty$-model $\mathcal{H}L_\infty$ introduced in [7] to analyse perpetual strategies in λ-calculus.

Let $\mathcal{D}_0$ be the three point lattice $\bot \sqsubseteq \mathtt{s} \sqsubseteq \top$ and $\mathcal{D}_1 = [\mathcal{D}_0 \to_\bot \mathcal{D}_0]$ be the set of strict continuous functions from $\mathcal{D}_0$ to $\mathcal{D}_0$, where a continuous function f is strict if $f(\bot) = \bot$.

Moreover let $\mathtt{i}_0$ be the initial projection defined by:

$$\mathtt{i}_0(\bot) = \bot \Rightarrow \bot \quad \mathtt{i}_0(\mathtt{s}) = \top \Rightarrow \mathtt{s} \quad \mathtt{i}_0(\top) = \mathtt{s} \Rightarrow \top$$

where $\mathsf{d}_1 \Rightarrow \mathsf{d}_2$ denotes the step function defined by $(\mathsf{d}_1 \Rightarrow \mathsf{d}_2)(\mathsf{e}) =$ if $\mathsf{e} \sqsupseteq \mathsf{d}_1$ then d_2 else $\bot$.

The *inverse limit construction* $\mathcal{H}L_\infty$ obtained starting from $\mathcal{D}_0$ and $\mathtt{i}_0$ is a model of the $\lambda\mathbf{I}$-calculus and of the $\lambda\mathsf{N}\mathbf{K}$-calculus (see Definition 11 in [7]) as shown in [7].

The interpretation of λ-terms in $\mathcal{H}L_\infty$ is defined in the standard way:

$$\begin{array}{l}[\![x]\!]_\rho = \rho(x)\\ [\![MN]\!]_\rho = \mathsf{F}[\![M]\!]_\rho[\![N]\!]_\rho\\ [\![\lambda x.M]\!]_\rho =\\ \qquad \mathsf{G}(\lambda\!\!\lambda \mathsf{d} \in \mathcal{H}L_\infty.\ \text{if } \mathsf{d} \neq \bot \text{ then } [\![\mathsf{M}]\!]_{\rho[\mathsf{d}/\mathsf{x}]} \text{ else } \bot)\end{array}$$

where (F, G) is the strict retraction from $[\mathcal{H}L_\infty \to_\bot \mathcal{H}L_\infty]$ to $\mathcal{H}L_\infty$ induced by $\mathtt{i}_0$. We recall that a pair of functions (f, g) is a strict retraction from $\mathcal{D}$ to $\mathcal{E}$ if they satisfy all the

$$\sigma \leq \sigma \cap \sigma \qquad \sigma \cap \tau \leq \sigma \qquad \sigma \cap \tau \leq \tau$$
$$\sigma \leq \sigma',\ \tau \leq \tau' \Rightarrow \sigma \cap \sigma' \leq \tau \cap \tau'$$
$$\sigma' \leq \sigma,\ \tau \leq \tau' \Rightarrow \sigma \to \tau \leq \sigma' \to \tau'$$
$$(\sigma \to \tau) \cap (\sigma \to \zeta) \leq \sigma \to \tau \cap \zeta$$
$$\varphi \sim \omega \to \varphi \qquad \omega \sim \varphi \to \omega \qquad \omega \leq \varphi$$
$$\sigma \leq \sigma \qquad \sigma \leq \tau,\ \tau \leq \zeta \ \Rightarrow \sigma \leq \zeta$$

Figure 2. Type preorder

following conditions: f and g are continuous; $f : \mathcal{E} \to \mathcal{D}$; $g : \mathcal{D} \to \mathcal{E}$; $f \circ g = \mathsf{id}_{\mathcal{D}}$; $g \circ f(\bot_{\mathcal{E}}) = \bot_{\mathcal{E}}$.

As proved in [7], we can give a finitary logical description of $\mathcal{H}L_\infty$ using intersection types. In other words we can define an intersection type theory $\mathcal{HL}$ which is the Stone dual of $\mathcal{H}L_\infty$ in the sense of [1].

The *set of types* of $\mathcal{HL}$ is built out of the constants φ and ω by the arrow and intersection constructors:

$$\tau ::= \varphi \mid \omega \mid \tau \to \tau \mid \tau \cap \tau$$

We define a preorder relation on types whose axioms and rules are justified by:

- viewing "$\to$" as the function space constructor and "$\cap$" as set intersection,
- considering the types φ and ω in correspondence with the elements $\mathtt{s}, \top$, respectively, but reversing the partial order in $\mathcal{H}L_\infty$ (this correspondence will be made explicit by the mapping $\mathtt{m}$ defined below).

Figure 2 defines the preorder $\leq$: we write $\tau \sim \sigma$ as short for $\tau \leq \sigma$ and $\sigma \leq \tau$. It is easy to check by induction on $\leq$ that ω and φ are the smallest and the biggest types, respectively.

We recall that *filters* of types are sets of types upper closed and closed under intersection. Let $\mathcal{F}$ be the set of all filters: it is easy to check that $\mathcal{F}$ is an ω-algebraic complete lattice with respect to set theoretic inclusion, whose bottom element is the empty set, and whose top element is the set of all types. Moreover, as shown in [7], $\mathcal{F}$ is isomorphic to $\mathcal{H}L_\infty$ through the mapping:

$$\hat{\mathtt{m}}(X) = \bigsqcup_{\tau \in X} \mathtt{m}(\tau)$$

where $\mathtt{m}(\varphi) = \mathtt{s}$, $\mathtt{m}(\omega) = \top$, $\mathtt{m}(\tau_1 \to \tau_2) = \mathtt{m}(\tau_1) \Rightarrow \mathtt{m}(\tau_2)$, $\mathtt{m}(\tau_1 \cap \tau_2) = \mathtt{m}(\tau_1) \sqcup \mathtt{m}(\tau_2)$.

We recall the intersection type assignment system of [7]: the typing rules are shown in Figure 3. We denote by $\vdash$ derivability in this system.

We can now formulate Stone duality for the model $\mathcal{H}L_\infty$ as follows:

$$[\![M]\!]_\rho = \bigsqcup \{\mathtt{m}(\tau) \mid \exists \Gamma \models \rho . \Gamma \vdash M : \tau\}$$

$$\frac{(x:\sigma) \in \Gamma}{\Gamma \vdash x : \sigma}\ (\mathrm{Ax}) \qquad \frac{\Gamma, x:\sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}\ (\to \mathrm{I})$$

$$\frac{\Gamma \vdash M : \sigma \to \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}\ (\to \mathrm{E})$$

$$\frac{\Gamma \vdash M : \sigma \quad \sigma \leq \tau}{\Gamma \vdash M : \tau}\ (\leq) \qquad \frac{\Gamma \vdash M : \sigma \quad \Gamma \vdash M : \tau}{\Gamma \vdash M : \sigma \cap \tau}\ (\cap \mathrm{I})$$

Figure 3. Typing rules

where $\Gamma \models \rho$ if $x : \sigma \in \Gamma$ implies $\mathtt{m}(\sigma) \sqsubseteq \rho(x)$. This statement is proved in [7].

Let $\rho_\top$ be the environment which associates $\top$ to all variables. We can characterise strongly normalising and persistently strongly normalising λ-terms in the model $\mathcal{H}L_\infty$ as the λ-terms whose interpretation in the environment $\rho_\top$ is different from $\bot$ and equal to $\top$, respectively. I.e. we have:

Theorem 5.1 (Main Theorem) *1. A λ-term M is strongly normalising iff $[\![M]\!]_{\rho_\top} \neq \bot$.*

2. A λ-term M is persistently strongly normalising iff $[\![M]\!]_{\rho_\top} = \top$.

The proof of this theorem uses the above discussed isomorphism between $\mathcal{H}L_\infty$ and $\mathcal{F}$. The theorem in fact can be reformulated as follows:

Theorem 5.2 *Let $\Gamma_\omega = \{x{:}\omega \mid x \in Var\}$.*

1. A λ-term $M \in \mathsf{SN}$ iff $\Gamma_\omega \vdash M : \varphi$.

2. A λ-term $M \in \mathsf{PSN}$ iff $\Gamma_\omega \vdash M : \omega$.

The *if parts* of Theorem 5.2(1) and (2) are shown in [7] (see Definition 35 and the following discussion) by means of the realizability interpretation of intersection types $\mathcal{V}$ defined by:

$$\begin{array}{lcl} \mathcal{V}(\varphi) & = & \mathsf{SN} \\ \mathcal{V}(\omega) & = & \mathsf{PSN} \\ \mathcal{V}(\sigma \to \tau) & = & \{M \in \Lambda \mid \forall N \in \mathcal{V}(\sigma)\ \ MN \in \mathcal{V}(\sigma)\} \\ \mathcal{V}(\sigma \cap \tau) & = & \mathcal{V}(\sigma) \cap \mathcal{V}(\tau). \end{array}$$

The *only if parts* of Theorem 5.2(1) and (2) can be shown using the inductive definitions of SN_n and PSN given in Section 4, and the following properties of the type assignment system which are proved in [4] and [7], respectively.

Lemma 5.3 (Generation Lemma) *1. $\Gamma \vdash x : \tau$ iff there is σ such that $x{:}\sigma \in \Gamma$ and $\sigma \leq \tau$.*

2. $\Gamma \vdash MN : \tau$ iff there is σ such that $\Gamma \vdash M : \sigma \to \tau$ and $\Gamma \vdash N : \sigma$.

3. $\Gamma \vdash \lambda x.M : \sigma \to \tau$ iff $\Gamma, x{:}\sigma \vdash M : \tau$.

Theorem 5.4 (Subject Expansion) *If $\Gamma \vdash M[x := N] : \tau$ and $\Gamma \vdash N : \varphi$, then $\Gamma \vdash (\lambda x.M)N : \tau$.*

More precisely the proof of the only if part easily follows from the following lemma.

Lemma 5.5 *1. If $M \in \mathsf{PSN}^\sharp$, then $\Gamma_\omega \vdash M : \omega$.*

2. If $M \in \mathsf{SN}^\sharp_n$, then $\Gamma_\omega \vdash M : \varphi^n \to \varphi$ where $\varphi^n \to \varphi = \underbrace{\varphi \to \ldots \to \varphi \to}_{n} \varphi$.

PROOF. The proof is by induction on the formation rules of $\mathsf{SN}^\sharp_n$ and $\mathsf{PSN}^\sharp$. For example for the rule

$$\frac{\lambda\overrightarrow{x}.M[y := N]\overrightarrow{L} \in \mathsf{SN}^\sharp_n \quad \lambda\overrightarrow{x}.N \in \mathsf{SN}^\sharp_m \quad \mathrm{lh}(\overrightarrow{x}) = m}{\lambda\overrightarrow{x}.(\lambda y.M)N\overrightarrow{L} \in \mathsf{SN}^\sharp_n}$$

by induction we know that $\Gamma_\omega \vdash \lambda\overrightarrow{x}.M[y := N]\overrightarrow{L} : \varphi^n \to \varphi$ and $\Gamma_\omega \vdash \lambda\overrightarrow{x}.N : \varphi^m \to \varphi$. We assume $n \geq m$, the proof for $n < m$ being similar. By Lemma 5.3(3) we get $\Gamma_\omega, \overrightarrow{x} : \overrightarrow{\varphi} \vdash M[y := N]\overrightarrow{L} : \varphi^{n-m} \to \varphi$ and $\Gamma_\omega, \overrightarrow{x} : \overrightarrow{\varphi} \vdash N : \varphi$. By Lemma 5.3(2) there exists $\overrightarrow{\tau}$ such that $\Gamma_\omega, \overrightarrow{x} : \overrightarrow{\varphi} \vdash M[y := N] : \overrightarrow{\tau} \to \varphi^{n-m} \to \varphi$ and $\Gamma_\omega, \overrightarrow{x} : \overrightarrow{\varphi} \vdash L_i : \tau_i$ for $1 \leq i \leq h$, where $h = \mathrm{lh}(\overrightarrow{L})$. This implies $\Gamma_\omega, \overrightarrow{x} : \overrightarrow{\varphi} \vdash (\lambda y.M)N : \overrightarrow{\tau} \to \varphi^{n-m} \to \varphi$ by Theorem 5.4 and so we conclude $\Gamma_\omega \vdash \lambda\overrightarrow{x}.(\lambda y.M)N\overrightarrow{L} : \varphi^n \to \varphi$ by the rules ($\to$ E) and ($\to$ I). □

6 Concluding Remarks

We have shown that the computational behaviour of λ-term application does not depend on having as arguments copies of the same λ-term or different λ-terms when we restrict to weakly or strongly normalising arguments. We call this result "Substitution Theorem". As an application of the "Substitution Theorem" for strong normalisation, we proved the completeness of the inductive definitions of persistently strongly normalising terms given in Figure 1 and shown that for a λ-term M the following four conditions are equivalent:

1. $MX_1 \ldots X_n$ is strongly normalising for all n and all strong normalising $X_1, \ldots, X_n$.
2. $M \in \mathsf{PSN}^\sharp$ as defined in Figure 1.
3. $\Gamma_\omega \vdash M : \omega$ in the intersection type assignment system induced by the type theory $\mathcal{HL}$.
4. $[\![M]\!]_{\rho_\top} = \top$ in the model $\mathcal{HL}_\infty$.

The equivalence between the third condition and the fourth one was proved in [7] and the equivalence among all of them is new. In particular, the equivalence between the first condition and the third one has solved an open problem in the conclusion of [4]. After this submission, we found we could extend the discussion in this paper to Λ^*, and we proved the above equivalencies for Λ^* in [5].

As an application of the "Substitution Theorem" for weak normalisation we get that:

$$\exists X_1, \ldots, X_n \in \mathsf{WN}. MX_1 \ldots X_n \notin \mathsf{WN} \Rightarrow \exists X \in \mathsf{WN}. M\underbrace{X \ldots X}_{n} \notin \mathsf{WN}$$

and similarly for strong normalisation. Therefore we plan to investigate consequences of this theorem in the study of infinite reductions.

Acknowledgements. We are grateful to Henk Barendregt, Flavio Corradini, Jan Willem Klop, Yukiyoshi Kameyama, and Makoto Kanazawa for stimulating discussions on the paper subject. We strongly improved on the submitted version thanks to the precious suggestions of careful referees.

References

[1] S. Abramsky. Domain Theory in Logical Form. *Annals of Pure and Applied Logic*, 51(1-2):1–77, 1991.

[2] H. P. Barendregt. *The Lambda Calculus: its Syntax and Semantics*. North-Holland, revised edition, 1984.

[3] G. Boudol. On Strong Normalization in the Intersection Type Discipline. In M. Hofmann, editor, *TLCA'03*, volume 2701 of *Lecture Notes in Computer Science*, pages 60 –74. Springer-Verlag, 2003.

[4] M. Dezani-Ciancaglini, F. Honsell, and Y. Motohama. Compositional Characterization of λ-terms using Intersection Types. *Theoretical Computer Science*, 340(3):459–495, 2005.

[5] M. Dezani-Ciancaglini and M. Tatsuta. A Behavioural Model for Klop's Calculus. In F. Corradini and C. Toffalori, editors, *Logic, Model and Computer Science*, ENTCS. Elsevier, 2006. to appear.

[6] J. R. Hindley and J. P. Seldin. *Introduction to Combinators and Lambda Calculus*. Cambridge University Press, 1986.

[7] F. Honsell and M. Lenisa. Semantical Analysis of Perpetual Strategies in λ-calculus. *Theoretical Computer Science*, 212(1-2):183–209, 1999.

[8] F. Kamareddine. Postponement, Conservation and Preservation of Strong Normalisation for Generalised Reduction. *Logic and Computation*, 10(5):721–738, 2000.

[9] J. W. Klop. *Combinatory Reduction Systems*. PhD thesis, Utrecht University, 1980. Appeared as Mathematical Centre Tracts 127, Kruislaan 413, 1098 SJ Amsterdam.

[10] R. P. Nederpelt. *Strong Normalisation for a Typed Lambda Calculus with Lambda Structured Types*. PhD thesis, Eindhoven University, 1973.

[11] P. Severi. *Normalisation in Lambda calculus and its Relation to Type Inference*. PhD thesis, Eindhoven University of Technology, 1996.

[12] M. Sørensen. Strong Normalization from Weak Normalization in Typed λ-calculi. *Information and Computation*, 133(1):37–71, 1997.