Characterising Strong Normalisation for Explicit Substitutions

(LATIN'02, Cancun, Mexico, LNCS 2286, pp. 356-370, 2002)

Steffen van Bakel[1] and Mariangiola Dezani-Ciancaglini [2]

[1] Department of Computing, Imperial College,
180 Queen's Gate, London SW7 2BZ, UK,
[2] Dipartimento di Informatica, Università di Torino,
Corso Svizzera 185, 10149 Torino, Italy,
dezani@di.unito.it

Abstract

We characterise the strongly normalising terms of a composition-free calculus of explicit substitutions (with or without garbage collection) by means of an intersection type assignment system. The main novelty is a new cut-rule which allows to forget the context of the minor premise when the context of the main premise does not have an assumption for the cut variable.

Introduction

Intersection types disciplines originated in [7] to overcome the limitations of Curry's type assignment system and to provide a characterisation of *strongly normalising terms* of the λ-calculus [21]. Since then, intersection types disciplines were used in a series of papers for characterising *evaluation properties* of λ-terms [17, 16, 3, 13, 2, 12, 9].

We are interested here in considering *calculi of explicit substitutions*, Λx, originated in [1] for improving λ-calculus implementations. In the literature there are many different proposals for explicit substitution calculi [6, 5, 15, 22], that are powerful tools for enlightening the relations between abstraction and application, as they decompose the evaluation rule of the λ-calculus into elementary steps. For this reason, it is really crucial to characterise the computational behaviour of substitution calculi.

In a seminal paper [11], Dougherty and Lescanne show that intersection type systems can characterise normalising and head-normalising terms of a composition-free calculus of explicit substitutions (with or without garbage collection). Allowing composition between substitutions leads to the (unexpected) failure of termination of simply typed terms, as proved by Melliès in [18]. Therefore, the choice of [11] and of the present paper is to consider calculi that are composition-free.

Characterisation of strongly normalising Λx-terms using intersection types has up to now been an open problem. In part, this problem has been addressed in [11], where the type assignment system $\mathcal{D}$ has the property that all typeable terms in Λx are strongly normalising, but the converse of this property fails. The aim of the present paper is to recover from this failure with a very simple move: we add a new cut-rule to the system $\mathcal{D}$. This rule essentially takes into account that by putting a term of the shape $M\langle x = N\rangle$, where x does not occur free in M, in an arbitrary context the free variables of N will never be replaced. Therefore, we can discharge the assumptions used to type N when we derive a type for $M\langle x = N\rangle$. Our main result is then:

a term in $\Lambda\mathsf{x}$ is typeable if and only if it is strongly normalising.

In order to prove one side of this result we devise an inductive characterisation of the set of strongly normalising terms in $\Lambda\mathsf{x}$ inspired by that for pure λ-terms of [23]. This allows us to show that all strongly normalising terms have a type. Notice that only typeability (not types!) is preserved by subject expansions which do not loose strong normalisation.

In order to prove the other side we use the set-theoretic semantics of intersection types and saturated sets, which is referred to as the *reducibility method*. This is a generally accepted method for proving the strong normalisation property of various type systems such as the simply typed lambda calculus in Tait [24], the polymorphic lambda calculus in Tait [25] and Girard [14]. All the above mentioned papers characterising evaluation properties of λ-terms and of terms in $\Lambda\mathsf{x}$ by means of intersection types apply variants of this method.

1 The Calculus

Following [11], we consider the set of terms $\Lambda\mathsf{x}$ which uses names rather than De Bruijn indices.

Definition 1.1 (Set of Terms $\Lambda\mathsf{x}$) The set of terms $\Lambda\mathsf{x}$ is defined by the grammar:

$$M, N ::= x \mid (\lambda x.M) \mid (MN) \mid (M\langle x := N\rangle)$$

As usual we consider terms modulo renaming of bound variables.

In writing terms, we will use the standard conventions for removing brackets, and use the following abbreviations:

$$\begin{aligned} \vec{M} &= M_1, \ldots, M_n \ (n \geq 0) \\ M\vec{M} &= MM_1 \cdots M_n \ (n \geq 0) \\ M\overrightarrow{\langle x := N\rangle} &= M\langle x_1 := N_1\rangle \cdots \langle x_n := N_n\rangle \ (n \geq 0) \\ |\vec{M}| &= n, \text{ where } n \text{ is the number of terms appearing in } \vec{M}. \end{aligned}$$

Apart from defining the notion of *free variables* of a term, we need to single out the free variables which occur in a term without considering some explicit substitution pairs.

Definition 1.2 The set $pfv(M)$ of *proper free variables of M* is inductively defined by:

$$\begin{aligned} pfv(x) &= \{x\} \\ pfv(\lambda x.M) &= pfv(M) \setminus x \\ pfv(MN) &= pfv(M) \cup pfv(N) \\ pfv(M\langle x := N\rangle) &= \begin{cases} pfv(M) \setminus x \cup pfv(N) & \text{if } x \in pfv(M) \\ pfv(M) & \text{otherwise.} \end{cases} \end{aligned}$$

For example, $pfv(z\langle y := xx\rangle\langle z := t\rangle) = \{t\}$, while $fv(z\langle y := xx\rangle\langle z := t\rangle) = \{x, t\}$.
Notice that the set $fv(M)$ of free variables of M can be defined in the same way, but for the last clause which then states

$$fv(M\langle x := N\rangle) = fv(M) \setminus x \cup fv(N).$$

Clearly we get $pfv(M) \subseteq fv(M)$ for all tems M.

We use the reduction relation $\lambda\mathsf{x}_{\mathsf{gc}}$ on $\Lambda\mathsf{x}$ as defined in [6].

Definition 1.3 (Reduction Relation) The reduction rules of $\lambda \mathsf{x}_{\mathsf{gc}}$ are:

$$\begin{array}{lrcl}
(\mathsf{B}): & (\lambda x.M)N & \to & M\langle x{:=}N\rangle \\
(\mathsf{App}): & (MP)\langle x{:=}N\rangle & \to & (M\langle x{:=}N\rangle)(P\langle x{:=}N\rangle) \\
(\mathsf{Abs}): & (\lambda y.M)\langle x{:=}N\rangle & \to & \lambda y.(M\langle x{:=}N\rangle) \\
(\mathsf{VarI}): & x\langle x{:=}N\rangle & \to & N \\
(\mathsf{gc}): & M\langle x{:=}N\rangle & \to & M, \text{ if } x \notin fv(M)
\end{array}$$

As usual, the reduction relation we consider in this paper is the contextual, transitive closure of the relation generated by these rules, and we will write $M \to N$ if M reduces to N using this relation.

Inside $\Lambda\mathsf{x}$ we are interested in the set of strongly normalisable terms.

Definition 1.4 ($\mathcal{SN}$) We define $\mathcal{SN} = \{M \in \Lambda\mathsf{x} \mid M \text{ is strongly normalisable}\}$.

As proved in [11], the set $\mathcal{SN}$ coincides with the set of strongly normalisable terms using the reduction relation obtained by removing the rule (gc) and by adding the following rule:

$$(\mathsf{VarK}):\ y\langle x{:=}N\rangle \ \to \ y$$

To simplify the following proofs, it is useful to consider a stronger version of the reduction rule (gc), which uses proper free variables instead of free variables, and the corresponding set of strongly normalisable terms.

Definition 1.5 ((gc_p), $\to_p$ and $\mathcal{SN}_p$) The reduction relation $\to_p$ is obtained by removing rule (gc) and adding the following rule:

$$(\mathsf{gc}_p) \qquad M\langle x{:=}N\rangle \to M \text{ if } x \notin pfv(N)$$

The set $\mathcal{SN}_p$ is the set of strongly normalising terms with respect to $\to_p$.

Notice that

$$\begin{array}{ccc}
M \to N & \Rightarrow & M \to_p N \\
\mathcal{SN}_p & \subseteq & \mathcal{SN}
\end{array}$$

but $M \to_p N \Rightarrow M \to N$ does not hold. Instead we will prove that $\mathcal{SN}_p = \mathcal{SN}$ by means of inductive characterisations of the sets $\mathcal{SN}$ and $\mathcal{SN}_p$. The correctness of $\mathcal{SN}$ characterisation follows from Lemma 5 of [11]. It is easy to verify that the proof of Lemma 5 of [11] easily adapts to the set $\mathcal{SN}_p$ and so we get the correctness of $\mathcal{SN}_p$ characterisation.

Lemma 1.6 *i) The set $\mathcal{SN}$ can be defined inductively through rules (1), (2),(3), (4), (5), (6), where $\mathcal{A}$ is $\mathcal{SN}$, and (7) of Figure 1.*

ii) The set $\mathcal{SN}_p$ can be defined inductively through rules (1), (2),(3), (4), (5), (6), where $\mathcal{A}$ is $\mathcal{SN}_p$, and (8) of Figure 1.

iii) $\mathcal{SN} = \mathcal{SN}_p$.

Proof: (*i*) and (*ii*): The first seven rules with $\mathcal{A} = \mathcal{SN}$ generate only terms in $\mathcal{SN}$: for the first two rules it is trivial and for the remaining ones it is proved in Lemma 5 of [11].

Since we can show Lemma 5 of [11] for $\mathcal{SN}_p$ (after repalcing rule (8) to rule (7)) we can similarly get that the first six rules with $\mathcal{A} = \mathcal{SN}_p$ and rule (8) generate only terms in $\mathcal{SN}_p$.

$$\frac{M\in\mathcal{A}}{\lambda x.M\in\mathcal{A}}\ (1) \qquad \frac{(\lambda y.M\langle x:=N\rangle)\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{A}}{(\lambda y.M)\langle x:=N\rangle\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{A}}\ (y\notin fv(M))\ (5)$$

$$\frac{\vec{M}\in\mathcal{A}}{x\vec{M}\in\mathcal{A}}\ (2) \qquad \frac{N\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{A}}{x\langle x:=N\rangle\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{A}}\ (6)$$

$$\frac{M\langle x:=N\rangle\vec{P}\in\mathcal{A}}{(\lambda x.M)N\vec{P}\in\mathcal{A}}\ (3) \qquad \frac{M\overrightarrow{\langle z:=Q\rangle}\vec{P},N\in\mathcal{SN}}{M\langle x:=N\rangle\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{SN}}\ (x\notin fv(M))\ (7)$$

$$\frac{(U\langle x:=N\rangle)(V\langle x:=N\rangle)\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{A}}{(UV)\langle x:=N\rangle\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{A}}\ (4) \qquad \frac{M\overrightarrow{\langle z:=Q\rangle}\vec{P},N\in\mathcal{SN}_p}{M\langle x:=N\rangle\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{SN}_p}\ (x\notin pfv(M))\ (8)$$

Figure 1: Rules generating $\mathcal{SN}$ and $\mathcal{SN}_p$

To see that these rules generate all the strongly normalising terms, first notice that the terms in the conclusions of the given rules cover all possible shapes of terms in Λx, as observed also in [11] (Lemma 1). Moreover if the term in the conclusion is strongly normalising, then also the terms in the premise must be strongly normalising: this can be proved by a double induction on the length of the longest derivation to normal form (using respectively $\rightarrow$ and $\rightarrow_p$) and on the structure of terms.

(iii): Immediate from (i) and (ii) taking into account that $x\notin fv(M)$ implies $x\notin pfv(M)$ and that $\mathcal{SN}_p\subseteq\mathcal{SN}$.

■

Always from Lemma 5 of [11] we get that distribution of substitution preserves strong normalisation:

Lemma 1.7 *If* $(P\overrightarrow{\langle x:=N\rangle}\langle y:=Q\overrightarrow{\langle x:=N\rangle}\rangle)\vec{M}\in\mathcal{SN}$ *then* $((P\langle y:=Q\rangle)\overrightarrow{\langle x:=N\rangle})\vec{M}\in\mathcal{SN}$.

2 The Type Assignment

We will consider intersection types as first defined in [7] with a preorder which takes into account the idempotence, commutativity and associativity of the intersection type constructor.

Definition 2.1 (Types) *i*) The set of types considered in this paper is the set $\mathcal{T}$ of *intersection types*, defined by the following grammar:

$$\sigma,\tau\ ::=\ \varphi\mid(\sigma\rightarrow\tau)\mid(\sigma\cap\tau)$$

where φ ranges over a denumerable set of type atoms.

ii) On $\mathcal{T}$ the type inclusion relation $\leq$ is inductively defined by: $\sigma\leq\sigma$, $\sigma\cap\tau\leq\sigma$, $\sigma\cap\tau\leq\tau$, $\sigma\leq\tau\ \&\ \sigma\leq\rho\Rightarrow\sigma\leq\tau\cap\rho$, and $\sigma\leq\tau\leq\rho\Rightarrow\sigma\leq\rho$.

iii) $\sigma\sim\tau\iff\sigma\leq\tau\leq\sigma$.

In the notation of types, as usual, right-most outer-most brackets will be omitted, and, since the type constructor $\cap$ is associative and commutative, we will write $\sigma\cap\tau\cap\rho$ rather than $(\sigma\cap\tau)\cap\rho$, and we will denote $\sigma_1\cap\cdots\cap\sigma_n$ by $\cap_{\underline{n}}\sigma_i$, where $\underline{n}=\{1,\ldots,n\}$[1].

[1] We allow $n=1$ as an abus de langage.

Before presenting the type assignment system we need a last definition.

Definition 2.2 (Statements and Contexts) *i*) A *statement* is an expression of the form $M{:}\sigma$, where M is the *subject* and σ is the *predicate* of $M{:}\sigma$.

ii) A *context* Γ is a partial mapping from variables to types, and it can be seen as a set of statements with (distinct) variables as subjects.

iii) The relations $\leq$ and $\sim$ are extended to contexts by:

$$\begin{aligned}\Gamma \leq \Gamma' &\iff \forall\, x{:}\sigma' \in \Gamma'\ \exists\, x{:}\sigma \in \Gamma\ [\sigma \leq \sigma']\\ \Gamma \sim \Gamma' &\iff \Gamma \leq \Gamma' \leq \Gamma.\end{aligned}$$

We introduce the following notational conventions:

$$\begin{aligned}\Gamma \cap \Delta &= \{x{:}\sigma \cap \tau \mid x{:}\sigma \in \Gamma \;\&\; x{:}\tau \in \Delta\} \cup \{x{:}\sigma \mid x{:}\sigma \in \Gamma \;\&\; x \notin \Delta\} \cup \{x{:}\tau \mid x \notin \Gamma \;\&\; x{:}\tau \in \Delta\}\\ \cap_{\underline{n}}\Gamma_i &= \Gamma_1 \cap \cdots \cap \Gamma_n\\ \Gamma, x{:}\sigma &= \Gamma \setminus x \cup \{x{:}\sigma\}.\end{aligned}$$

For example $\{x{:}\sigma\} \cap \{x{:}\tau\}$ denotes $\{x{:}\sigma \cap \tau\}$, while $\{x{:}\sigma\}, x{:}\tau$ denotes $\{x{:}\tau\}$.

As discussed in the introduction, the key of our type assignment is a cut-rule (*cut***K**) which allows to forget the context of the minor premise. We will call the standard cut-rule (*cut***I**).

Definition 2.3 (Type Assignment Rules) *Type assignment* for terms in Λx and *derivations* are defined by the following natural deduction system:

$$(Ax):\ \frac{}{\Gamma \vdash x{:}\sigma}\ (x{:}\sigma \in \Gamma)$$

$$(\to I):\ \frac{\Gamma, x{:}\sigma \vdash M{:}\tau}{\Gamma \vdash \lambda x.M{:}\sigma\to\tau} \qquad (\to E):\ \frac{\Gamma \vdash M{:}\sigma\to\tau \quad \Gamma \vdash N{:}\sigma}{\Gamma \vdash MN{:}\tau}$$

$$(\cap I):\ \frac{\Gamma \vdash M{:}\sigma \quad \Gamma \vdash M{:}\tau}{\Gamma \vdash M{:}\sigma\cap\tau} \qquad (\cap E):\ \frac{\Gamma \vdash M{:}\sigma_1\cap\sigma_2}{\Gamma \vdash M{:}\sigma_i}\ (i \in \underline{2})$$

$$(cut\mathbf{I}):\ \frac{\Gamma, x{:}\sigma \vdash M{:}\tau \quad \Gamma \vdash N{:}\sigma}{\Gamma \vdash M\langle x{:=}N\rangle{:}\tau} \qquad (cut\mathbf{K}):\ \frac{\Gamma \vdash M{:}\tau \quad \Delta \vdash N{:}\sigma}{\Gamma \vdash M\langle x{:=}N\rangle{:}\tau}\ (x \notin \Gamma)$$

We write $\Gamma \vdash M{:}\sigma$ if there exists a derivation that has this as its conclusion.

The following example explains in detail the difference between the system of [11] and that of this paper, and shows that the counter example to the characterisation of strongly normalisabilty of that paper is dealt with successfully here.

Example 2.4 Let $D \equiv \lambda a.aa$ and $M \equiv (\lambda x.(\lambda y.z)(xx))D$. Then

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\{a{:}(\sigma\to\tau)\cap\sigma\} \vdash a{:}(\sigma\to\tau)\cap\sigma}}{\{a{:}(\sigma\to\tau)\cap\sigma\} \vdash a{:}\sigma\to\tau} \quad \dfrac{\dfrac{}{\{a{:}(\sigma\to\tau)\cap\sigma\} \vdash a{:}(\sigma\to\tau)\cap\sigma}}{\{a{:}(\sigma\to\tau)\cap\sigma\} \vdash a{:}\sigma}}{\{a{:}(\sigma\to\tau)\cap\sigma\} \vdash aa{:}\tau}}{\emptyset \vdash \lambda a.aa{:}((\sigma\to\tau)\cap\sigma)\to\tau}$$

Notice that $M \to z\langle y{:=}DD\rangle \to z\langle y{:=}DD\rangle \to \cdots$, so M is not strongly normalisable. Also $M \to (\lambda x.z\langle y{:=}xx\rangle)D \to M' \equiv z\langle y{:=}xx\rangle\langle x{:=}D\rangle \to M'' \equiv z\langle x{:=}D\rangle$. Then both M' and M'' are strongly normalisable, but only the latter is typeable in the system of [11]. Instead, in the system presented here, M' is typeable[2] (where $\mathbf{D}$ is the derivation given above):

$$\dfrac{\dfrac{}{\{z{:}\mu\} \vdash z:\mu} \qquad \dfrac{\dfrac{\dfrac{}{\{x{:}(\rho\to\nu)\cap\rho\} \vdash x:(\rho\to\nu)\cap\rho}}{\{x{:}(\rho\to\nu)\cap\rho\} \vdash x:\rho\to\nu} \quad \dfrac{\dfrac{}{\{x{:}(\rho\to\nu)\cap\rho\} \vdash x:(\rho\to\nu)\cap\rho}}{\{x{:}(\rho\to\nu)\cap\rho\} \vdash x:\rho}}{\{x{:}(\rho\to\nu)\cap\rho\} \vdash xx:\nu}}{\{z{:}\mu\} \vdash z\,\langle y:=xx\rangle:\mu} \qquad \dfrac{\mathbf{D}}{\emptyset \vdash \lambda a.aa:(\sigma\to\tau)\cap\sigma\to\tau}}{\{z{:}\mu\} \vdash z\,\langle y:=xx\rangle\,\langle x:=\lambda a.aa\rangle:\mu}$$

This implies, of course, that, in contrast to the system of [11], the terms $(\lambda x.M)N$ and $M\langle x:=N\rangle$ *no longer* have the same typing behaviour. In fact, when typing $(\lambda x.M)N$, the type used for x in the sub-derivation for $\Gamma \vdash \lambda x.M:\sigma$ *has* to be a type for N, even if x does not appear in M, whereas rule $(cut\mathbf{K})$ only needs that N is typeable, not that is has the type assumed for x; actually, in rule $(cut\mathbf{K})$, *no* type is assumed for x. This then solves the problem of [11], in that the type used for x to type xx has no relation at all to the type derived for $\lambda a.aa$.

As usual for type assignment systems, we have a Generation Lemma and a Subject Reduction Theorem. First, we formulate some general properties of our system.

Lemma 2.5 *i) If $\Gamma \vdash M:\tau$ and $\Gamma' \leq \Gamma$ and $\tau \leq \tau'$, then $\Gamma' \vdash M:\tau'$.*
ii) If $\Gamma \vdash M:\sigma\to\tau$ and $\Delta \vdash N:\sigma$, then $\Gamma\cap\Delta \vdash MN:\tau$.
iii) If $\Gamma \vdash M:\sigma$ and $x \notin \Gamma$, then $x \notin pfv(M)$.
iv) Let $\Gamma \vdash M:\sigma$, and $\Gamma' = \{x{:}\tau \in \Gamma \mid x \in pfv(M)\}$, then $\Gamma' \vdash M:\sigma$.

Proof: All proofs are by induction on deductions. The only interesting case is point (i) when the last applied rule is $(cut\mathbf{K})$

$$\frac{\Gamma \vdash M:\tau \quad \Delta \vdash N:\sigma}{\Gamma \vdash M\langle x:=N\rangle:\tau}\ (x \notin \Gamma)$$

and $\Gamma' = \Gamma, x{:}\rho$. If y is a fresh variable we can derive by induction $\Gamma' \vdash M[y/x]:\tau'$ and conclude using $(cut\mathbf{K})$

$$\frac{\Gamma' \vdash M[y/x]:\tau' \quad \Delta \vdash N:\sigma}{\Gamma' \vdash M[y/x]\,\langle x:=N\rangle:\tau'}\ (y \notin \Gamma)$$

■

Lemma 2.6 (Generation Lemma) *i) If $\Gamma \vdash x:\sigma$, then there exists $x{:}\tau \in \Gamma$ such that $\tau \leq \sigma$.*
ii) If $\Gamma \vdash MN:\sigma$, then there exist n,σ_i,ρ_i $(i \in \underline{n})$ such that $\sigma = \cap_{\underline{n}}\sigma_i$, and $\Gamma \vdash M:\rho_i\to\sigma_i$ and $\Gamma \vdash N:\rho_i$, for all $i \in \underline{n}$.
iii) If $\Gamma \vdash \lambda y.M:\sigma$, then there exist n,ρ_i,μ_i $(i \in \underline{n})$ such that $\sigma = \cap_{\underline{n}}(\rho_i\to\mu_i)$ and, for all $i \in \underline{n}$, $\Gamma, y{:}\rho_i \vdash M:\mu_i$.
iv) If $\Gamma \vdash M\langle x:=N\rangle:\sigma$, then either
a) there exists τ such that $\Gamma, x{:}\tau \vdash M:\sigma$ and $\Gamma \vdash N:\tau$, or
b) $\Gamma \setminus x \vdash M:\sigma$ and there exist Δ,τ such that $\Delta \vdash N:\tau$.

[2] By Subject Reduction (Theorem 2.7) M'' is typeable too.

Proof: Easy, using Lemma 2.5(i) and rule $(\cap I)$ for part (iv). ■

A minimal requirement of our system is that it satisfies the subject reduction property (SR). We show SR for the reduction $\rightarrow_p$: this gives us for free SR for $\rightarrow$.

Theorem 2.7 (Subject Reduction Theorem) *If $M \rightarrow_p N$, and $\Gamma \vdash M{:}\sigma$, then $\Gamma \vdash N{:}\sigma$.*

Proof: By induction on the definition of the reduction relation, '$\rightarrow$'. We only show the base cases.

(B): Then $\Gamma \vdash (\lambda x.M)N{:}\sigma$, and, by Lemma 2.6($ii$), there exist n,σ_i,ρ_i ($i \in \underline{n}$) such that $\sigma = \cap_{\underline{n}}\sigma_i$, and, for all $i \in \underline{n}$, $\Gamma \vdash \lambda x.M{:}\rho_i{\rightarrow}\sigma_i$ and $\Gamma \vdash N{:}\rho_i$. We can assume none of the σ_i to be an intersection, so, by Lemma 2.6(iii), for $i \in \underline{n}$, $\Gamma,x{:}\rho_i \vdash M{:}\sigma_i$, and therefore, by rule $(cut\mathbf{I})$, $\Gamma \vdash M\langle x{:=}N\rangle : \sigma_i$. So, by rule $(\cap I)$, $\Gamma \vdash M\langle x{:=}N\rangle : \sigma$.

(App): Then $\Gamma \vdash (MN)\langle x{:=}P\rangle : \sigma$. Let $\sigma = \cap_{\underline{n}}\sigma_i$ where none of the σ_i is an intersection. By Lemma 2.6(iv), we have two cases:

- a) there exists τ such that $\Gamma,x{:}\tau \vdash MN{:}\sigma$ and $\Gamma \vdash P{:}\tau$. Then, by Lemma 2.6(ii), for every $i \in \underline{n}$, there exists ρ_i such that $\Gamma,x{:}\tau \vdash M{:}\rho_i{\rightarrow}\sigma_i$ and $\Gamma,x{:}\tau \vdash N{:}\rho_i$. Then, by rule $(cut\mathbf{I})$, $\Gamma \vdash M\langle x{:=}P\rangle : \rho_i{\rightarrow}\sigma_i$ and $\Gamma \vdash N\langle x{:=}P\rangle : \rho_i$.
- b) $\Gamma \setminus x \vdash MN{:}\sigma$ and there exist Δ,τ such that $\Delta \vdash P{:}\tau$. As above, by Lemma 2.6(ii), there exists ρ_i such that $\Gamma \setminus x \vdash M{:}\rho_i{\rightarrow}\sigma_i$ and $\Gamma \setminus x \vdash N{:}\rho_i$. Then, by rule $(cut\mathbf{K})$ and Lemma 2.5(i), $\Gamma \vdash M\langle x{:=}P\rangle : \rho_i{\rightarrow}\sigma_i$ and $\Gamma \vdash N\langle x{:=}P\rangle : \rho_i$.

In both cases, for $i \in \underline{n}$, by rule $(\rightarrow E)$, also $\Gamma \vdash (M\langle x{:=}P\rangle)(N\langle x{:=}P\rangle){:}\sigma_i$, so by rule $(\cap I)$, $\Gamma \vdash (M\langle x{:=}P\rangle)(N\langle x{:=}P\rangle){:}\sigma$.

(Abs): Then $\Gamma \vdash (\lambda y.M)\langle x{:=}N\rangle : \sigma$. Let $\sigma = \cap_{\underline{n}}\sigma_i$ where none of the σ_i is an intersection. By Lemma 2.6(iv), we have two cases:

- a) $\Gamma,x{:}\tau \vdash (\lambda y.M){:}\sigma$ and there exists τ such that $\Gamma \vdash N{:}\tau$. By Lemma 2.6(iii), for $i \in \underline{n}$, there exist ρ_i,μ_i such that $\sigma_i = \rho_i{\rightarrow}\mu_i$ and $\Gamma,x{:}\tau,y{:}\rho_i \vdash M{:}\mu_i$. Then, by rule $(cut\mathbf{I})$, $\Gamma,y{:}\rho_i \vdash M\langle x{:=}N\rangle : \mu_i$.
- b) $\Gamma \setminus x \vdash (\lambda y.M){:}\sigma$ and there exist Δ,τ such that $\Delta \vdash N{:}\tau$. As above, there exist ρ_i,μ_i such that $\sigma_i = \rho_i{\rightarrow}\mu_i$ and $\Gamma \setminus x, y{:}\rho_i \vdash M{:}\mu_i$. Then, by rule $(cut\mathbf{K})$ and Lemma 2.5(i), $\Gamma,y{:}\rho_i \vdash M\langle x{:=}N\rangle : \mu_i$.

In both cases, by rule $(\rightarrow I)$, $\Gamma \vdash \lambda y.(M\langle x{:=}N\rangle){:}\sigma_i$, and, by rule $(\cap I)$, $\Gamma \vdash \lambda y.(M\langle x{:=}N\rangle){:}\sigma$.

(VarI): Then $\Gamma \vdash x\langle x{:=}N\rangle : \sigma$, and, by Lemma 2.6($iv$) and ($i$), there exists τ such that $\Gamma,x{:}\tau \vdash x{:}\sigma$. and $\Gamma \vdash N{:}\tau$. Then, by Lemma 2.6(i), $\tau \leq \sigma$, and, by Lemma 2.5(i), $\Gamma \vdash N{:}\sigma$.

(gc$_p$): Then $\Gamma \vdash M\langle x{:=}N\rangle : \sigma$ and $x \notin pfv(M)$. Then, by Lemma 2.6(iv),

- a) either there exists τ such that $\Gamma,x{:}\tau \vdash M{:}\sigma$, and, by Lemma 2.5($iv$), $\Gamma \vdash M{:}\sigma$.
- b) or $\Gamma \setminus x \vdash M{:}\sigma$.

In both cases we get $\Gamma \vdash M{:}\sigma$.

■

3 All Strongly Normalisable Terms are Typeable

The main result of this paper, that our system types exactly the strongly normalisable terms, comes in two parts. First we show that all terms in $\mathcal{SN}$ are typeable in our system (Theorem 3.3), and then that all typeable terms are in $\mathcal{SN}$ (Theorem 4.5).

First we show two technical lemmas which are useful for the proof of Theorem 3.3.

Lemma 3.1 *i) If $\Gamma \vdash M\langle x{:=}N\rangle : \tau$, then there exists $\Gamma' \leq \Gamma$ such that $\Gamma' \vdash (\lambda x.M)N : \tau$.*
ii) If $\Gamma \vdash (U\langle x{:=}N\rangle)(V\langle x{:=}N\rangle) : \tau$ then $\Gamma \vdash (UV)\langle x{:=}N\rangle : \tau$.
iii) If $\Gamma \vdash \lambda y.M\langle x{:=}N\rangle : \tau$ and $y \notin fv(N)$, then $\Gamma \vdash (\lambda y.M)\langle x{:=}N\rangle : \tau$.
iv) If $\Gamma \vdash M : \tau$, $\Delta \vdash N : \sigma$ and $x \notin pfv(M)$, then $\Gamma \vdash M\langle x{:=}N\rangle : \tau$.

Proof: *i*) By Lemma 2.6(*iv*), if $\Gamma \vdash M\langle x{:=}N\rangle : \tau$ then:

a) either there is ρ such that $\Gamma, x{:}\rho \vdash M : \tau$, and $\Gamma \vdash N : \rho$. But then, by rules $(\to I)$, and $(\to E)$, we get $\Gamma \vdash (\lambda x.M)N : \tau$.

b) or $\Gamma \backslash x \vdash M : \tau$ and there are Δ, ρ such that $\Delta \vdash N : \rho$. But then, by Lemma 2.5(*i*), $\Gamma, x{:}\rho \vdash M : \tau$ so, by rule $(\to I)$ and using Lemma 2.5(*ii*), $\Gamma \cap \Delta \vdash (\lambda x.M)N : \tau$. Notice that $\Gamma \cap \Delta \leq \Gamma$.

ii) By Lemma 2.6(*ii*), if $\Gamma \vdash (U\langle x{:=}N\rangle)(V\langle x{:=}N\rangle) : \tau$ there are n, ρ_i, τ_i $(i \in \underline{n})$ such that $\tau = \cap_{\underline{n}} \tau_i$, $\Gamma \vdash U\langle x{:=}N\rangle : \rho_i \to \tau_i$ and $\Gamma \vdash V\langle x{:=}N\rangle : \rho_i$. By Lemma 2.6(*iv*), for each $i \in \underline{n}$ we can have four different cases:

a) there are μ_i, ν_i such that $\Gamma, x{:}\mu_i \vdash U : \rho_i \to \tau_i$, $\Gamma \vdash N : \mu_i$, $\Gamma, x{:}\nu_i \vdash V : \rho_i$, and $\Gamma \vdash N : \nu_i$. But then, by Lemma 2.5(*ii*) and rule $(\cap I)$:

$$\Gamma, x{:}\mu_i \cap \nu_i \vdash UV : \tau_i, \text{ and } \Gamma \vdash N : \mu_i \cap \nu_i$$

so we get $\Gamma \vdash (UV)\langle x{:=}N\rangle : \tau_i$, using rule $(cut\mathbf{I})$.

b) $\Gamma \backslash x \vdash V : \rho_i$, and there are μ_i, Δ_i, ν_i such that $\Gamma, x{:}\mu_i \vdash U : \rho_i \to \tau_i$, $\Gamma \vdash N : \mu_i$, $\Delta_i \vdash N : \nu_i$. But then, by Lemma 2.5(*ii*)

$$\Gamma, x{:}\mu_i \vdash UV : \tau_i, \text{ and } \Gamma \vdash N : \mu_i$$

so we get $\Gamma \vdash (UV)\langle x{:=}N\rangle : \tau_i$, using rule $(cut\mathbf{I})$.

c) $\Gamma \backslash x \vdash U : \rho_i \to \tau_i$, and there are μ_i, Δ_i, ν_i such that $\Delta_i \vdash N : \mu_i$, $\Gamma, x{:}\nu_i \vdash V : \rho_i$, $\Gamma \vdash N : \nu_i$. The proof in this case is similar to that of the previous one.

d) $\Gamma \backslash x \vdash U : \rho_i \to \tau_i$, and there are $\Delta_i, \mu_i, \Delta'_i, \nu_i$ such that $\Delta_i \vdash N : \mu_i$, $\Gamma \backslash x \vdash V : \rho_i$, $\Delta'_i \vdash N : \nu_i$. But then, by rule $(\to E)$:

$$\Gamma \backslash x \vdash UV : \tau_i, \text{ and } \Delta_i \vdash N : \mu_i$$

so we get $\Gamma \vdash (UV)\langle x{:=}N\rangle : \tau_i$, using rule $(cut\mathbf{K})$ and Lemma 2.5(*i*).

Finally, using rule $(\cap I)$ we conclude $\Gamma \vdash (UV)\langle x{:=}N\rangle : \tau$.

iii) By Lemma 2.6(*iii*), if $\Gamma \vdash \lambda y.M\langle x{:=}N\rangle : \tau$ then there are n, μ_i, ν_i $(i \in \underline{n})$ such that $\Gamma, y{:}\mu_i \vdash M\langle x{:=}N\rangle : \nu_i$, and $\tau = \cap_{\underline{n}} \nu_i$. By Lemma 2.6(*iv*) , for each $i \in \underline{n}$, either:

a) there is ρ_i such that $\Gamma, y{:}\mu_i, x{:}\rho_i \vdash M : \nu_i$ and $\Gamma, y{:}\mu_i \vdash N : \rho_i$. Then, by rule $(\to I)$, we get $\Gamma, x{:}\rho_i \vdash \lambda y.M : \mu_i \to \nu_i$.

b) or $\Gamma \backslash x, y{:}\mu_i \vdash M : \nu_i$ and there are Δ_i, ρ_i such that $\Delta_i \vdash N : \rho_i$. Then, by rule $(\to I)$, we get $\Gamma \backslash x \vdash \lambda y.M : \mu_i \to \nu_i$.

Let $\underline{m}$ be the subset of $\underline{n}$ which match the first alternative. If $\underline{m}$ is not empty then by Lemma 2.5(*i*) (notice that by hypothesis $y \notin fv(N)$) and rule $(\cap I)$:

$$\Gamma, x{:}\cap_{\underline{m}} \rho_i \vdash \lambda y.M : \cap_{\underline{n}} (\mu_i \to \nu_i) \text{ and } \Gamma \vdash N : \cap_{\underline{m}} \rho_i.$$

We conclude, using rule $(cut\mathbf{I})$:

$$\Gamma \vdash (\lambda y.M)\langle x{:=}N\rangle : \tau.$$

Otherwise if $\underline{m}$ is empty then, by Lemma 2.5(i) and rule $(\cap I)$:

$$\Gamma \setminus x \vdash \lambda y.M : \cap_{\underline{n}}(\mu_i \to \nu_i) \;\text{ and }\; \Delta_i \vdash N : \rho_i.$$

We conclude, choosing an arbitrary $\Delta_j \vdash N : \rho_j$ and using rule ($cut\mathbf{K}$) and Lemma 2.5(i):

$$\Gamma \vdash (\lambda y.M)\,\langle x := N\rangle : \tau.$$

iv) If $\Gamma \vdash M : \tau$ and $x \notin pfv(M)$, then, by Lemma 2.5(iv), $\Gamma \setminus x \vdash M : \tau$. We conclude using rule ($cut\mathbf{K}$) and Lemma 2.5(i). ■

Lemma 3.2 i) Assume, for all Γ, *that* $\Gamma \vdash M : \sigma$ *implies* $\Gamma \vdash M' : \sigma$. *Then, for all* Δ, N, τ: $\Delta \vdash M\,\langle x := N\rangle : \tau$ *implies* $\Delta \vdash M'\,\langle x := N\rangle : \tau$;

ii) Assume, for all Γ, *that* $\Gamma \vdash M : \sigma$ *implies there exists* $\Gamma' \le \Gamma$ *such that* $\Gamma' \vdash M' : \sigma$. *Then, for all* Δ, N, τ: $\Delta \vdash MN : \tau$ *implies that there exists* $\Delta' \le \Delta$ *such that* $\Delta' \vdash M'N : \tau$. *Moreover, we get* $\Delta' = \Delta$ *whenever* $\Gamma = \Gamma'$.

Proof: i) By Lemma 2.6(iv), $\Delta \vdash M\,\langle x := N\rangle : \tau$ implies that either there is ρ such that $\Delta, x{:}\rho \vdash M : \tau$ and $\Delta \vdash N : \rho$, or $\Delta \setminus x \vdash M : \tau$ and there are Δ', ρ such that $\Delta' \vdash N : \rho$. In both cases the conclusion easily follows from the assumption.

ii) Similar but simpler. ■

The characterisation of $\mathcal{SN}$ given in Lemma 1.6(i) is crucial to prove that all strongly normalisable terms are typeable.

Theorem 3.3 *If* $M \in \mathcal{SN}$ *then* $\Gamma \vdash M : \sigma$ *for some* Γ, σ.

Proof: By induction on the rules generating $\mathcal{SN}$ (Lemma 1.6(i)) using the Generation Lemma (Lemma 2.6).

(1): By induction, $\Gamma \vdash M : \sigma$. We distinguish two cases:

- a) If $x{:}\tau \in \Gamma$, so, by rule $(\to I)$, $\Gamma \setminus x \vdash \lambda x.M : \tau \to \sigma$.
- b) If $x \notin \Gamma$ then, by Lemma 2.5(i), $\Gamma, x{:}\tau \vdash M : \sigma$, so, by rule $(\to I)$, $\Gamma \vdash \lambda x.M : \tau \to \sigma$.

(2): Let $|\vec{M}| = n$, then, by induction, there are Γ_i, σ_i $(i \in \underline{n})$ such that $\Gamma_i \vdash M_i : \sigma_i$, for $i \in \underline{n}$. Then $\cap_{\underline{n}}\Gamma_i, \{x{:}\sigma_1 \to \cdots \to \sigma_n \to \tau\} \vdash x\vec{M} : \tau$.

(3): By induction, $\Gamma \vdash M\,\langle x := N\rangle\,\vec{P} : \sigma$, and, by Lemma 3.1($i$) and 3.2($ii$), there exists $\Gamma' \le \Gamma$ such that $\Gamma' \vdash (\lambda x.M)N\vec{P} : \sigma$.

(4): By induction, $\Gamma \vdash (U\,\langle x := N\rangle\,)(V\,\langle x := N\rangle\,)\,\overrightarrow{\langle z := Q\rangle}\,\vec{P} : \sigma$, and, by Lemma 3.1($ii$) and 3.2, $\Gamma \vdash (UV)\,\langle x := N\rangle\,\overrightarrow{\langle z := Q\rangle}\,\vec{P} : \sigma$.

(5): By induction, $\Gamma \vdash (\lambda y.M\,\langle x := N\rangle\,)\,\overrightarrow{\langle z := Q\rangle}\,\vec{P} : \sigma$, and, by Lemma 3.1($iii$) and 3.2, $\Gamma \vdash (\lambda y.M)\,\langle x := N\rangle\,\overrightarrow{\langle z := Q\rangle}$

(6): This case follows immediately from Lemma 3.2 and rule ($cut\mathbf{I}$).

(7): If $x \notin fv(M)$, then $x \notin pfv(M)$. By induction, $\Gamma \vdash M\,\overrightarrow{\langle z := Q\rangle}\,\vec{P} : \sigma$ and $\Delta \vdash N : \tau$, and, by Lemma 3.1(iv) and 3.2, $\Gamma \vdash M\,\langle x := N\rangle\,\overrightarrow{\langle z := Q\rangle}\,\vec{P} : \sigma$. ■

Notice that, since subject expansions preserving strong normalisation do not preserve types, we can only assure that the expanded term is always typeable by the theorem above. For example we can derive

$$\vdash \lambda y.(\lambda z.z)\,\langle x := yy\rangle : \varphi \to \sigma \to \sigma$$

where φ is an atom, but we cannot derive

$$\vdash \lambda y.(\lambda xz.z)(yy) : \varphi \to \sigma \to \sigma.$$

A typing for the last term is for example:

$$\vdash \lambda y.(\lambda xz.z)(yy) : \tau \cap (\tau \to \rho) \to \sigma \to \sigma.$$

4 All Typeable Terms are Strongly Normalisable

The general idea of the reducibility method is to interpret types by suitable sets (saturated and stable sets in Tait [24] and Krivine [16] and admissible relations in Mitchell [19] and [20]) of terms (*reducible terms*) which satisfy the required property (e.g. strong normalisation) and then to develop semantics in order to obtain the soundness of the type assignment. A consequence of soundness, the fact that every term typeable by a type in the type system belongs to the interpretations of that type, leads to the fact that terms typeable in the type system satisfy the required property, since the type interpretations are built up in that way.

In order to develop the reducibility method we consider the applicative structure whose domain are terms in Λx and where the application is just the application of terms.

Definition 4.1 (Reducible terms) *i*) We define the collection of set of terms $\mathcal{R}^{\rho}$ inductively over types by:

$$\begin{aligned} \mathcal{R}^{\varphi} &= \mathcal{SN} \\ \mathcal{R}^{\sigma\to\tau} &= \{M \mid \forall N\, [N \in \mathcal{R}^{\sigma} \Rightarrow MN \in \mathcal{R}^{\tau}]\} \\ \mathcal{R}^{\sigma\cap\tau} &= \mathcal{R}^{\sigma} \cap \mathcal{R}^{\tau}. \end{aligned}$$

ii) We define the set $\mathcal{R}$ of *reducible terms* by: $\mathcal{R} = \{M \mid \exists \rho\, [M \in \mathcal{R}^{\rho}]\} = \bigcup_{\rho \in \mathcal{T}} \mathcal{R}^{\rho}$.

Notice that, if $M \in \mathcal{R}^{\sigma}$, not necessarily there exists a Γ such that $\Gamma \vdash M:\sigma$. For example, if φ, φ' are two different type variables, then $\lambda x.x \in \mathcal{R}^{\varphi\to\varphi'}$, since $(\lambda x.x)M \in \mathcal{SN}$ whenever $M \in \mathcal{SN}$. But we cannot derive $\vdash \lambda x.x : \varphi \to \varphi'$. Moreover, since $\lambda x.x \in \mathcal{SN}$, $\lambda x.x \in \mathcal{R}^{\varphi}$, but we cannot derive $\vdash \lambda x.x : \varphi$.

We now show that reducibility implies strongly normalisability and that all term-variables are reducible. For the latter, we need to show that all typeable strongly normalisable terms that start with a term variable are reducible.

Lemma 4.2 i) $\mathcal{R} \subseteq \mathcal{SN}$.
ii) $x\vec{N} \in \mathcal{SN} \Rightarrow \forall \rho\, [x\vec{N} \in \mathcal{R}^{\rho}]$.

Proof: By simultaneous induction on the structure of types.

$i)(\varphi)$: By Definition 4.1.

$(\sigma\to\tau)$: $M \in \mathcal{R}^{\sigma\to\tau} \Rightarrow (4.1\ \&\ IH(ii))\ Mx \in \mathcal{R}^{\tau} \Rightarrow (IH(i))\ Mx \in \mathcal{SN} \Rightarrow M \in \mathcal{SN}$.

$(\sigma\cap\tau)$: $M \in \mathcal{R}^{\sigma\cap\tau} \Rightarrow (4.1)\ M \in \mathcal{R}^{\sigma}\ \&\ M \in \mathcal{R}^{\tau} \Rightarrow (IH(i))\ M \in \mathcal{SN}$.

$ii)(\varphi)$: $x\vec{N} \in \mathcal{SN} \Rightarrow (4.1)\ x\vec{N} \in \mathcal{R}^{\varphi}$.

$(\sigma\to\tau)$: We show that for an arbitrary $M \in \mathcal{R}^{\sigma}$ we get $x\vec{N}M \in \mathcal{R}^{\tau}$. We conclude $x\vec{N} \in \mathcal{R}^{\sigma\to\tau}$ by Definition 4.1.

$M \in \mathcal{R}^{\sigma} \Rightarrow (IH(i))\ M \in \mathcal{SN}$.

$x\vec{N} \in \mathcal{SN}\ \&\ M \in \mathcal{SN} \Rightarrow (1.6(i)(2))\ x\vec{N}M \in \mathcal{SN} \Rightarrow (IH(ii))\ x\vec{N}M \in \mathcal{R}^{\tau}$.

$(\sigma\cap\tau)$: $x\vec{N} \in \mathcal{SN} \Rightarrow (IH(ii))\ x\vec{N} \in \mathcal{R}^{\sigma}\ \&\ x\vec{N} \in \mathcal{R}^{\tau} \Rightarrow (4.1)\ x\vec{N} \in \mathcal{R}^{\sigma\cap\tau}$. ■

We will now show that the reducibility predicate is closed for subject expansion, preserving strong normalisation, with respect to the reduction rules (B), (App), (Abs), (VarI), (gc_p) and with respect to distribution of substitution. These results are needed in the proof of Theorem 4.5.

Lemma 4.3 *i) If* $M\langle x:=N\rangle\,\vec{Q}\in\mathcal{R}^{\mu}$, *then* $(\lambda x.M)N\vec{Q}\in\mathcal{R}^{\mu}$.

ii) If $(M_1\overrightarrow{\langle x:=N\rangle}\,)(M_2\overrightarrow{\langle x:=N\rangle}\,)\,\vec{Q}\in\mathcal{R}^{\mu}$, *then* $(M_1M_2)\overrightarrow{\langle x:=N\rangle}\,\vec{Q}\in\mathcal{R}^{\mu}$.

iii) If $(\lambda y.M'\overrightarrow{\langle x:=N\rangle}\,)\,\vec{P}\in\mathcal{R}^{\mu}$ *and* $y\notin fv(\vec{N})$, *then* $(\lambda y.M')\overrightarrow{\langle x:=N\rangle}\,\vec{P}\in\mathcal{R}^{\mu}$.

iv) If $N\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{R}^{\mu}$, *then* $x\langle x:=N\rangle\,\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{R}^{\mu}$.

v) If $M\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{R}^{\mu}$, $N\in\mathcal{R}^{\rho}$ *and* $x\notin pfv(M)$, *then* $M\langle x:=N\rangle\,\overrightarrow{\langle z:=Q\rangle}\vec{P}\in\mathcal{R}^{\mu}$.

vi) If $(P\overrightarrow{\langle x:=N\rangle}\,\langle y:=Q\overrightarrow{\langle x:=N\rangle}\,\rangle\,)\,\vec{M}\in\mathcal{R}^{\mu}$, *then* $((P\langle y:=Q\rangle\,)\overrightarrow{\langle x:=N\rangle}\,)\,\vec{M}\in\mathcal{R}^{\mu}$.

Proof: By induction on the structure of types.

(φ): The proofs of these properties are all very similar, using Lemma 1.6(*i*) for the first four partes, Lemma 1.6(*ii*)(8) and (*iii*) for part (*v*), and Lemma 1.7 for part (*vi*). So it sufficies to show this last case.

$$\begin{array}{ll}
(P\overrightarrow{\langle x:=N\rangle}\,\langle y:=Q\overrightarrow{\langle x:=N\rangle}\,\rangle\,)\vec{M}\in\mathcal{R}^{\varphi} & \Rightarrow (4.1)\\
(P\overrightarrow{\langle x:=N\rangle}\,\langle y:=Q\overrightarrow{\langle x:=N\rangle}\,\rangle\,)\vec{M}\in\mathcal{SN} & \Rightarrow (\text{ Lemma 1.7})\\
((P\langle y:=Q\rangle\,)\overrightarrow{\langle x:=N\rangle}\,)\vec{M}\in\mathcal{SN} & \Rightarrow (4.1)\\
((P\langle y:=Q\rangle\,)\overrightarrow{\langle x:=N\rangle}\,)\vec{M}\in\mathcal{R}^{\varphi}. &
\end{array}$$

$(\sigma\to\tau)$: We consider again part (*vi*), the other parts being similar. Let $R\in\mathcal{R}^{\sigma}$ be arbitrary.

$$\begin{array}{ll}
(P\overrightarrow{\langle x:=N\rangle}\,\langle y:=Q\overrightarrow{\langle x:=N\rangle}\,\rangle\,)\vec{M}\in\mathcal{R}^{\sigma\to\tau} & \Rightarrow (4.1)\\
(P\overrightarrow{\langle x:=N\rangle}\,\langle y:=Q\overrightarrow{\langle x:=N\rangle}\,\rangle\,)\vec{M}R\in\mathcal{R}^{\tau} & \Rightarrow (IH)\\
((P\langle y:=Q\rangle\,)\overrightarrow{\langle x:=N\rangle}\,)\vec{M}R\in\mathcal{R}^{\tau} & \Rightarrow (4.1)\\
((P\langle y:=Q\rangle\,)\overrightarrow{\langle x:=N\rangle}\,)\vec{M}\in\mathcal{R}^{\sigma\to\tau}. &
\end{array}$$

$(\sigma\cap\tau)$: For all properties this case is immediate by Definition 4.1 and induction. ■

We shall prove our strong normalisation result by showing that every typeable term is reducible. For this, we need to prove a stronger property: we will show that if we substitute term variables by reducible terms in a typeable term, then we obtain a reducible term. This gives the soundness of our type interpretation.

Theorem 4.4 (Soundness) *If* $\{x_1{:}\mu_1,\ldots,x_n{:}\mu_n\}\vdash M{:}\sigma$, *and, for* $1\leq i\leq n$, $N_i\in\mathcal{R}^{\mu_i}$, *such that* $x_i\notin fv(N_j)$, *for all* $1\leq i,j\leq n$, *then* $M\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\sigma}$.

Proof: By induction on the structure of derivations. Let $\Gamma=\{x_1{:}\mu_1,\ldots,x_n{:}\mu_n\}$.

(Ax): Then $M\equiv x_j$, and $\mu_j=\sigma$, for some $1\leq j\leq n$. Since $N_j\in\mathcal{R}^{\mu_j}$, $N_j\in\mathcal{R}^{\sigma}$. Then, by Lemma 4.3(*iv*) and (*v*), $x_j\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\sigma}$.

$(\to I)$: Then $M\equiv\lambda y.M'$, $\sigma=\rho\to\tau$, and $B,y{:}\rho\vdash M'{:}\tau$. Let $N\in\mathcal{R}^{\rho}$, then, by induction, $M'\overrightarrow{\langle x:=N\rangle}\,\langle y:=N\rangle\in\mathcal{R}^{\tau}$. So, by Lemma 4.3(*i*), $(\lambda y.M'\overrightarrow{\langle x:=N\rangle}\,)N\in\mathcal{R}^{\tau}$, and, by Definition 4.1, $\lambda y.M'\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\rho\to\tau}$. We can assume $y\notin fv(N)$, so, by Lemma 4.3(*iii*), $(\lambda y.M')\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\rho\to\tau}$.

$(\to E)$: Then $M\equiv M_1M_2$ and there exists τ such that $\Gamma\vdash M_1{:}\tau\to\sigma$ and $\Gamma\vdash M_2{:}\tau$. By induction, $M_1\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\tau\to\sigma}$ and $M_2\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\tau}$. But then, by Definition 4.1, $M_1\overrightarrow{\langle x:=N\rangle}\,M_2\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\sigma}$, so, by Lemma 4.3(*ii*), $(M_1M_2)\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\sigma}$.

$(\cap I)$: Then $\sigma\equiv\sigma_1\cap\sigma_2$ and, for $i\in\underline{2}$, $\Gamma\vdash M{:}\sigma_i$. So, by induction, $M\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\sigma_1}$ and $M\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\sigma_2}$, so, by Definition 4.1, $M\overrightarrow{\langle x:=N\rangle}\,\in\mathcal{R}^{\sigma}$.

$(\cap E)$: Then there exists τ such that $\Gamma \vdash M : \sigma \cap \tau$, and, by induction, $M\overrightarrow{\langle x := N\rangle} \in \mathcal{R}^{\sigma\cap\tau}$. Then, by Definition 4.1, $M\overrightarrow{\langle x := N\rangle} \in \mathcal{R}^{\sigma}$.

$(cut\mathbf{I})$: Then $M \equiv P\langle y := Q\rangle$, and there exists τ such that $\Gamma, y{:}\tau \vdash P : \sigma$ and $\Gamma \vdash Q : \tau$. Then, by induction, $Q\overrightarrow{\langle x := N\rangle} \in \mathcal{R}^{\tau}$, so, again by induction, $P\overrightarrow{\langle x := N\rangle}\,\langle y := Q\overrightarrow{\langle x := N\rangle}\rangle \in \mathcal{R}^{\sigma}$. So, by Lemma 4.3(*vi*), $(P\langle y := Q\rangle)\overrightarrow{\langle x := N\rangle} \in \mathcal{R}^{\sigma}$.

$(cut\mathbf{K})$: Then $M \equiv P\langle y := Q\rangle$, $\Gamma \vdash P : \sigma$, $y \notin \Gamma$ and there exist Δ, τ such that $\Delta \vdash Q : \tau$. Let $\Delta = \{z_1{:}\mu_1, \ldots, z_m{:}\mu_m\}$. By Lemma 4.2(*ii*), for all $j \in \underline{m}$, $z_j \in \mathcal{R}^{\rho_j}$, and therefore, by induction, $Q \in \mathcal{R}^{\tau}$. By Lemma 2.5(*iii*), $y \notin pfv(P)$. So, by Lemma 4.3(*v*), $(P\langle y := Q\rangle)\overrightarrow{\langle x := N\rangle} \in \mathcal{R}^{\sigma}$. ■

Theorem 4.5 *If $\Gamma \vdash M : \sigma$ for some Γ, σ then $M \in \mathcal{SN}$.*

Proof: By Lemma 4.2(*ii*), all term variables are reducible for any type, so, by Theorem 4.4, for all M, $M\overrightarrow{\langle x := x\rangle}$ is reducible. Strong normalisation of $M\overrightarrow{\langle x := x\rangle}$ then follows from Lemma 4.2(*i*). Since $M\overrightarrow{\langle x := x\rangle} \to M$, also $M \in \mathcal{SN}$. ■

5 Final Remarks

Only after writing the first version of the present paper we realized that Dougherty, Lengrand and Lescanne solved the same problem in [10] with a similar type assignment system. The only difference is the formulation of rule $(cut\mathbf{K})$ which becomes:

$$\frac{\Gamma \vdash M : \tau \quad \Delta \vdash N : \sigma}{\Gamma \vdash M\langle x := N\rangle : \tau}\ (x \notin pfv(M))$$

Lemma 2.5(*iii*) and (*iv*) the two systems deduce the same types for the same terms. Deeper relations between the two systems will be object of further investigations.

If we add an universal type Ω with the axiom (Ω) [8]:

$$\frac{}{\Gamma \vdash M : \Omega}$$

then in the so obtained system, rule $(cut\mathbf{K})$ becomes admissible. But using axiom (Ω) we can type trivially all terms, and we can also type terms which are not strongly normalising with types different from Ω. An example is $y{:}\varphi \vdash (\lambda x.y)((\lambda z.zz)(\lambda z.zz)) : \varphi$. So we cannot obtain a characterisation of the set $\mathcal{SN}$.

In the line of [9], we plan to characterise by means of intersection types disciplines other compositional properties of calculi of explicit substitutions, like that of reducing to a closed term or of being a weak head-normalising term.

Finally we will explore the possibility of building suitable type intersection assignment systems for showing that some calculi of explicit substitutions enjoy the weak or strong normalisation property.

References

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

[2] R. M. Amadio and P.-L. Curien. *Domains and lambda-calculi*. Cambridge University Press, Cambridge, 1998.

[3] S. van Bakel. Complete restrictions of the intersection type discipline. *Theoret. Comput. Sci.*, 102(1):135–163, 1992.

[4] S. van Bakel. Intersection type assignment systems. *Theoret. Comput. Sci.*, 151(2):385–435, 1995. Thirteenth Conference on Foundations of Software Technology and Theoretical Computer Science (Bombay, 1993).

[5] Z. Benaissa, D. Briaud, P. Lescanne, and J. Rouyer-Degli. λv, a calculus of explicit substitutions which preserves strong normalization. *Journal of Functional Programming*, 6(5):699–722, 1996.

[6] R. Bloo and K. Rose. Preservation of strong normalization in named lambda calculi with explicit substitution and garbage collection. In *Computer Science in the Netherlands*, pages 62–72. Koninklijke Jaarbeurs, 1995.

[7] M. Coppo and M. Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre Dame J. Formal Logic*, 21(4):685–693, 1980.

[8] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Principal type schemes and λ-calculus semantics. In *To H. B. Curry: essays on combinatory logic, lambda calculus and formalism*, pages 535–560. Academic Press, London, 1980.

[9] M. Dezani-Ciancaglini, F. Honsell, and Y. Motohama. Compositional characterization of λ-terms using intersection types. In *Mathematical Foundations of Computer Science 2000*, volume 1893 of *Lecture Notes in Computer Science*, pages 304–313. Springer, 2000.

[10] D. Dougherty, S. Lengrand, and P. Lescanne. Types and explicit substitutions. to appear, 2001.

[11] D. Dougherty and P. Lescanne. Reductions, intersection types and explicit substitutions. In *Typed Lambda Calculi and Applications 2001*, volume 2044 of *Lecture Notes in Computer Science*, pages 121–135. Springer, 2001.

[12] J. Gallier. Typing untyped λ-terms, or reducibility strikes again! *Ann. Pure Appl. Logic*, 91:231–270, 1998.

[13] S. Ghilezan. Strong normalization and typability with intersection types. *Notre Dame J. Formal Logic*, 37(1):44–52, 1996.

[14] J.-Y. Girard. Une extension de l'interprétation de Gödel à l'analyse, et son application à l'elimination des coupures dans l'analyse et la théorie des types. In *2nd Scandinavian Logic Symposium*, pages 63–92. North-Holland, 1971.

[15] F. Kmamareddine and A. Rios. Extending a λ-calculus with explicit substitutions which preserves strong normalization into a colfluent calculus of open terms. *Journal of Functional Programming*, 7(4):395–420, 1997.

[16] J.-L. Krivine. *Lambda-calcul Types et modèles.* Masson, Paris, 1990.

[17] D. Leivant. Typing and computational properties of lambda expressions. *Theoret. Comput. Sci.*, 44(1):51–68, 1986.

[18] P.-A. Melliès. Typed λ-calculi with explicit substitution may not terminate. In *Typed Lambda Calculi and Applications 2001*, volume 902 of *Lecture Notes in Computer Science*, pages 328–334. Springer, 1995.

[19] J. C. Mitchell. Type systems for programming languages. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 415–431. Elsevire, Amsterdam, 1990.

[20] J. C. Mitchell. *Foundation for Programmimg Languages.* MIT Press, 1996.

[21] G. Pottinger. A type assignment for the strongly normalizable λ-terms. In *To H. B. Curry: essays on combinatory logic, lambda calculus and formalism*, pages 561–577. Academic Press, London, 1980.

[22] E. Ritter. Characterizing explicit substitutions which preserve termination. In *Typed Lambda Calculi and Applications 1999*, volume 1581 of *Lecture Notes in Computer Science*, pages 325–339. Springer, 1999.

[23] P. Severi. *Normalisation in lambda calculus and its relation to type inference.* PhD thesis, Eindhoven University of Technology, 1996.

[24] W. W. Tait. Intensional interpretations of functionals of finite type I. *Journal of Symbolic Logic*, 32:198–212, 1967.

[25] W. W. Tait. A realizability interpretation of the theory of species. In *Logic Colloquium*, volume 453 of *Lecture Notes in Mathematics*, pages 240–251. Springer, 1975.