Cut-Elimination in the Strict Intersection Type Assignment System is Strongly Normalising

(Notre Dame, Journal of Formal Logic 45(1), pp. 35-63, 2004)

Steffen van Bakel

Department of Computing, Imperial College, 180 Queen's Gate, London SW7 2BZ, U.K.
svb@doc.ic.ac.uk

Abstract

This paper defines reduction on derivations (cut-elimination) in the Strict Intersection Type Assignment System of [1] and shows a strong normalisation result for this reduction. Using this result, new proofs are given for the approximation theorem and the characterisation of normalisability of terms, using intersection types.

keywords: intersection types, lambda calculus, termination, cut-elimination.

Introduction

Strong normalisation of cut-elimination is a well established property in the area of logic that has been studied profoundly in the past. In the area of type assignment for the Lambda Calculus (LC), the corresponding property is that of strong normalisation of derivation reduction (also called cut-elimination in, for example, [10]), which mimics the normal reduction on terms to which the types are assigned, and also this area has been well studied.

For intersection type assignment systems, proofs of strong normalisation of derivation reduction have at best been indirect, i.e. obtained through a mapping from the derivations into a logic, where the property has been established before. Since in those logics the type-constant ω cannot be adequately mapped, the intersection systems studied in that way are ω-free. (There exists a logic that deals adequately with intersection and ω [17], but strong normalisation of cut-elimination has not been shown yet for it.) This paper will use the Strict Type Assignment System of [1] (which contains ω), and will present a proof for the property directly in the system itself.

The Intersection Type Discipline (ITD) as presented in [11] (a more enhanced system was presented in [10]; for an overview of the various existing systems, see [2]), was introduced mainly to overcome the limitations of Curry's type assignment system [13, 14] and has been used to characterise normalisation using types. It is an extension of Curry's system in that term variables (and terms) are allowed to have more than one type: in the context of a certain term M, a term-variable x can play different, even non-unifiable, roles. This slight generalisation of Curry's system causes a great change in complexity; although type assignment in Curry's system is decidable, in ITD it is not, which is illustrated by the fact that type assignment is closed for β-equality:

$$M =_\beta N \Rightarrow (B \vdash M{:}\sigma \iff B \vdash N{:}\sigma).$$

The ITD is most renown for providing proofs for the following *characterisation of (head/strong) normalisation* by assignable types (where ω is a type-constant, and stands for the universal

type, i.e. all terms are typeable by ω):

$$\begin{array}{rcl} M \text{ has a head normal form} & \Longleftrightarrow & B \vdash M{:}\sigma \ \& \ \sigma \neq \omega \\ M \text{ has a normal form} & \Longleftrightarrow & B \vdash M{:}\sigma \ \& \ \omega \text{ does not occur in } B,\sigma \\ M \text{ is strongly normalisable} & \Longleftrightarrow & B \vdash M{:}\sigma, \text{ where } \omega \text{ is not used at all.} \end{array}$$

These properties immediately show that type assignment, even in the system that does not contain ω [1], is undecidable.

As many strong normalisation results in the context of types, also the strong normalisation result of this paper is obtained using the technique of Computability Predicates [26, 20]. This technique provides a means for proving termination of typeable terms using a predicate defined by induction on the structure of types, and has been widely used to study normalisation properties (or similar results), as for example in [22, 12, 15, 24, 21, 1, 2, 19, 7, 4, 18, 5] (this list is by no means intended to be complete).

Also, as in [2], the technique proved to be sufficient to show a head-normalisation as well as an approximation result, which will be shown again here (see Theorem 5.5 and 5.4, respectively). In this paper these results –the three characterisation results and the approximation result– are shown to be a direct consequence of the main result, in that all normal characterisations of (strong/head) normalisation are consequences of the strong normalisation of cut-elimination.

In the context of *weak* reduction, the approximation result is no longer obtained via a straightforward application of the same computability technique as used in LC. Rather, as argued and shown in [6, 8], to obtain this result in the context of Combinator Systems or Term Rewriting Systems, a more general solution was needed: *strong normalisation of cut-elimination*. Perhaps surprisingly, the machinery involved to prove this gives the characterisation results for typeable terms as a corollary.

In this paper, we will show these results in the context of LC: we will show that cut-elimination is strongly normalising, and that all characterisation results are direct consequences of it. The added complexity of intersection types implies that, unlike for ordinary systems of type assignment, there is a significant difference between derivation reduction and ordinary reduction (see the beginning of Section 2); unlike normal typed- or type assignment system, in $\vdash$ not every term-redex occurs with types in a derivation.

As far as cut-elimination in the context of intersection types is concerned, there exists but few related results in the literature. As [23], where a strong normalisation result was proved for derivation reduction in the setting of the notion of intersection type assignment known as $\mathcal{D}$ [21], most papers consider the BCD-system [10] without the type-constant ω. Since we consider the type ω here, together with a type inclusion relation $\leq$, that strong normalisation result itself is a true special case of the results of this paper presented in Section 5.

The *Approximation Theorem* hinted at above is a (perhaps less known) fundamental result for ITD, and is more relevant in the context of semantics. Essentially following [27, 9], the set of terms can be extended by adding the term-constant $\bot$. Adding also the reduction rules $\bot N \to_{\beta\bot} \bot$, and $\lambda x.\bot \to_{\beta\bot} \bot$ to the notion of reduction gives rise to the notion of *approximate normal forms* that are in essence finite rooted segments of Böhm-trees [9], and a model for the LC can be obtained by interpreting a term M by the set of approximants that can be associated to it, $\mathcal{A}(M)$. The Approximation Theorem now states that there exists a very precise relation between types assignable to a term and those assignable to its approximants and is formulated as

$$B \vdash M{:}\sigma \iff \exists A \in \mathcal{A}(M)\ [B \vdash A{:}\sigma]$$

(see [25, 1, 2]; for a uniform proof for many systems, see [16]). From this it also follows, i.e. next to the direct proofs, that the set of intersection types assignable to a term can be used

to define a model for the LC (see [10, 1, 2]).

The kind of intersection type assignment considered in this paper is that of [1], i.e. the *strict* intersection type assignment system, a restricted version of the BCD-system of [10], that is equally powerful in terms of typeability and expressiveness. The major feature of this restricted system, compared to the BCD-system, is a restricted version of the derivation rules and the use of strict types (first introduced in [1]); notably, the strict system differs from the BCD-system in terms of expressivity in that it is *not* closed for η-reduction.

This paper is the full, revised version of [3].

1 Strict intersection type assignment

In this section, we will present the Strict Intersection Type Assignment System as first presented in [1], which can be seen as a restricted version of the BCD-system as presented in [10]. The major feature of this restricted system, compared to the BCD-system, is that the $\leq$ relation on types is no longer contra-variant on the argument type in arrow-types, but restricted to the one induced by $\sigma \cap \tau \leq \sigma$ and taking ω to be the maximal type.

We assume the reader to be familiar with the LC [9]; we just recall the definition of lambda terms and β-equality. We will write $\underline{n}$ for $\{1, \ldots, n\}$, where $n \geq 0$.

Definition 1.1 (Lambda terms and β-equality [9]) *i*) The set Λ of *lambda terms* is defined by the syntax:

$$M \ ::= \ x \mid \lambda x.M \mid M_1 M_2$$

ii) The reduction relation $\rightarrow_\beta$ is defined as the contextual (i.e. compatible [9]) closure of the rule:

$$(\lambda x.M)N \ \rightarrow_\beta \ M[N/x]$$

The relation $\twoheadrightarrow_\beta$ is the reflexive and transitive closure of $\rightarrow_\beta$, and the $=_\beta$ is the equivalence relation generated by $\twoheadrightarrow_\beta$.

Definition 1.2 (Types, statements, and bases) *i*) Let Φ be a countable (infinite) set of type-variables, ranged over by φ. $\mathcal{T}_S$, the set of *strict types*, and the set $\mathcal{T}$ of *intersection types*, both ranged over by Greek characters like $\sigma, \tau, \ldots$, are defined through:

- The set $\mathcal{T}_S$ of *strict types* is inductively defined by:

$$\sigma \ ::= \ \varphi \mid ((\sigma_1 \cap \ldots \cap \sigma_n) \rightarrow \sigma), \ (n \geq 0)$$

- The set $\mathcal{T}$ of *intersection types* is defined by:

$$\{\sigma_1 \cap \ldots \cap \sigma_n \mid n \geq 0 \ \& \ \forall i \in \underline{n} \ [\sigma_i \text{ is a strict type}]\}$$

We will write ω for the empty intersection type.

ii) A *statement* is an expression of the form $M{:}\sigma$, with $M \in \Lambda$, and $\sigma \in \mathcal{T}$. M is the *subject* and σ the *predicate* of $M{:}\sigma$.

iii) A *basis* is a partial mapping from term variables to intersection types, and is represented as a set of statements with only distinct variables as subjects.

iv) For bases $B_1, \ldots, B_n$, the basis $\bigcap\{B_1, \ldots, B_n\}$ is defined by:
$x{:}\sigma_1 \cap \ldots \cap \sigma_m \in \bigcap\{B_1, \ldots, B_n\}$ if and only if $\{x{:}\sigma_1, \ldots, x{:}\sigma_m\}$ is the (non-empty) set of all statements about x that occur in $B_1 \cup \ldots \cup B_n$.

Notice that strict types are either type-variables, φ, or arrow types. In an arrow type, the

type on the right of the arrow type constructor is always strict; the type on the left of the arrow is an intersection type, but since $\mathcal{T}_S$ is a proper subset of $\mathcal{T}$, it can be strict.

We will write $B, x{:}\sigma$ for the basis $\bigcap\{B, \{x{:}\sigma\}\}$, when x does not occur in B, and we will omit the brackets '{' and '}' when writing a basis explicitly. Also, in the notation of types, as usual, right-most outer-most brackets will be omitted.

In papers like [10, 1, 2], the type constant ω is introduced separately, and is used as the universal type, i.e. all terms can be typed with ω. For succinctness of proofs and definitions, ω is treated here as an intersection of zero strict types, which is justified by the following. The semantics of a type σ, $[\![\sigma]\!]$ (see [10, 1, 2]), is defined as the set of terms that it can be assigned to (see Definition 1.5). Notice that, if M can be assigned the type $\sigma_1 \cap \ldots \cap \sigma_n$, it can also be assigned $\sigma_1 \cap \ldots \cap \sigma_{n-1}$, so we get

$$[\![\sigma_1 \cap \ldots \cap \sigma_n]\!] \subseteq [\![\sigma_1 \cap \ldots \cap \sigma_{n-1}]\!] \subseteq \ldots \subseteq [\![\sigma_1 \cap \sigma_2]\!] \subseteq [\![\sigma_1]\!].$$

It is natural to extend this sequence with $[\![\sigma_1]\!] \subseteq [\![\ ]\!]$, and therefore to define that the semantics of the empty intersection is Λ; since in [10, 1, 2] all terms are typeable by ω, also $[\![\ ]\!] = [\![\omega]\!]$.

We will consider a pre-order on types which takes into account the idem-potence, commutativity and associativity of the intersection type constructor, and defines ω to be the maximal element.

Definition 1.3 (Relations on types) *i*) The relation $\leq$ is defined as the least pre-order (i.e. reflexive and transitive relation) on $\mathcal{T}$ such that:

$$\begin{array}{rcl} \sigma_1 \cap \ldots \cap \sigma_n & \leq & \sigma_i, \ \text{ for all } i \in \underline{n}, \ n \geq 1 \\ \tau \leq \sigma_i, \ \text{ for all } i \in \underline{n} & \Rightarrow & \tau \leq \sigma_1 \cap \ldots \cap \sigma_n, \quad n \geq 0 \end{array}$$

ii) The equivalence relation $\sim$ on types is defined by: $\sigma \sim \tau \iff \sigma \leq \tau \leq \sigma$, and we will consider types modulo $\sim$.

iii) We write $B \leq B'$ if and only if for every $x{:}\sigma' \in B'$ there is an $x{:}\sigma \in B$ such that $\sigma \leq \sigma'$, and $B \sim B' \iff B \leq B' \leq B$.

$\mathcal{T}$ may be considered modulo $\sim$; then $\leq$ becomes a partial order.

Notice that $\sigma \leq \sigma$, and $\sigma \leq \omega$, for all σ; it is easy to show that both $(\sigma \cap \tau) \cap \rho \sim \tau \cap (\sigma \cap \rho)$ and $\sigma \cap \tau \sim \tau \cap \sigma$, so the type constructor $\cap$ is associative and commutative, and we will write $\sigma \cap \tau \cap \rho$ rather than $(\sigma \cap \tau) \cap \rho$. Moreover, we will, when appropriate, write $\cap_{\underline{n}} \sigma_i$ for $\sigma_1 \cap \ldots \cap \sigma_n$ (where $\cap_{\underline{1}} \sigma_i = \sigma_1$) and we will then assume, unless stated explicitly otherwise, that each σ_i is a strict type.

The definition of the $\leq$-relation as given in [10] (apart from dealing with intersection types occurring on the right of the arrow type constructor) or [2] also contained the alternative:

$$\rho \leq \sigma \ \& \ \tau \leq \mu \quad \Rightarrow \quad \sigma \to \tau \leq \rho \to \mu$$

This was added mainly to obtain a notion of type assignment closed for η-reduction (i.e. β-reduction extended with $\lambda x.Mx \to_\eta M$, if x is not free in M), a feature that is not considered here.

The following property is easy to show:

Property 1.4 (cf. [2]) *For all $\sigma, \tau \in \mathcal{T}$, $\sigma \leq \tau$ if and only if there are $n, m \geq 0$, σ_i $(\forall i \in \underline{n})$, τ_j $(\forall j \in \underline{m})$ such that $\sigma = \cap_{\underline{n}} \sigma_i$, $\tau = \cap_{\underline{m}} \tau_i$, and, for all $j \in \underline{m}$ there exists $i \in \underline{n}$ such that $\tau_j = \sigma_i$.*

Definition 1.5 (Strict type assignment and derivations) *i*) *Strict intersection type assignment* and *strict intersection derivations* are defined by the following natural deduction sys-

tem (where σ in rules $(\to E)$ and $(\to I)$ is in $\mathcal{T}$):

$$(Ax):\ \frac{}{B,x{:}\cap_{\underline{n}}\sigma_i \vdash x{:}\sigma_i}\ (n \geq 1, i \in \underline{n}) \qquad (\to E):\ \frac{B \vdash M{:}\sigma\to\tau \quad B \vdash N{:}\sigma}{B \vdash MN{:}\tau}$$

$$(\cap I):\ \frac{B \vdash M{:}\sigma_1 \quad \ldots \quad B \vdash M{:}\sigma_n}{B \vdash M{:}\cap_{\underline{n}}\sigma_i}\ (n \geq 0) \qquad (\to I):\ \frac{B,x{:}\sigma \vdash M{:}\tau}{B \vdash \lambda x.M{:}\sigma\to\tau}$$

ii) We write $B \vdash M{:}\sigma$ if this statement is derivable using a strict intersection derivation, and write $\mathrm{D} :: B \vdash M{:}\sigma$ to specify that this result was obtained through the derivation D.

To illustrate that the strict system is not closed for η-reduction, notice that we can give a derivation for $\vdash \lambda xy.xy{:}(\sigma\to\tau)\to(\rho\cap\sigma)\to\tau$, but not for $\vdash \lambda x.x{:}(\sigma\to\tau)\to(\rho\cap\sigma)\to\tau$.

Notice that, since ω is considered to be the empty intersection, the derivation rule

$$(\omega):\frac{}{B \vdash M{:}\omega}$$

is implicit in rule $(\cap I)$.

The following lemma shows a term-substitution result.

Lemma 1.6 $\exists\rho[B,x{:}\rho \vdash M{:}\sigma \ \&\ B \vdash N{:}\rho] \iff B \vdash M[N/x]{:}\sigma.$

Proof: By induction on the structure of terms; only the case $\sigma \in \mathcal{T}_S$ is considered.

$$\begin{array}{llll}
(M \equiv x): & (\Rightarrow): & \exists\rho\ [B,x{:}\rho \vdash x{:}\sigma \ \&\ B \vdash N{:}\rho] & \Rightarrow \quad (Ax) \\
 & & \exists\sigma_i\ (\forall i \in \underline{n}), i \in \underline{n}\ [\sigma = \sigma_i \ \&\ B \vdash N{:}\cap_{\underline{n}}\sigma_i] & \Rightarrow \quad (\leq) \\
 & & B \vdash x[N/x]{:}\sigma_i. & \\
 & (\Leftarrow): & B \vdash x[N/x]{:}\sigma \Rightarrow B,x{:}\sigma \vdash x{:}\sigma \ \&\ B \vdash N{:}\sigma. & \\
(M \equiv y \neq x) & (\Leftrightarrow): & \exists\rho\ [B,x{:}\rho \vdash y{:}\sigma \ \&\ B \vdash N{:}\rho] \Rightarrow B \vdash y[N/x]{:}\sigma. & \\
 & (\Leftarrow): & B \vdash y[N/x]{:}\sigma \Rightarrow B \vdash y{:}\sigma \ \&\ B \vdash N{:}\omega. &
\end{array}$$

$$\begin{array}{lll}
(M \equiv \lambda y.M'): & \exists\rho\ [B,x{:}\rho \vdash \lambda y.M'{:}\sigma \ \&\ B \vdash N{:}\rho] & \iff \quad (\to I) \\
 & \exists\rho,\alpha,\beta\ [B,x{:}\rho,y{:}\alpha \vdash M'{:}\beta \ \&\ \sigma = \alpha\to\beta \ \&\ B \vdash N{:}\rho] & \iff \quad (IH) \\
 & \exists\alpha,\beta\ [B,y{:}\alpha \vdash M'[N/x]{:}\beta \ \&\ \sigma = \alpha\to\beta] & \iff \quad (\to I) \\
 & B \vdash \lambda y.M'[N/x]{:}\sigma. &
\end{array}$$

$$\begin{array}{lll}
(M \equiv M_1M_2): & B \vdash M_1M_2[N/x]{:}\sigma & \iff \quad (\to E) \\
 & \exists\tau\ [B \vdash M_1[N/x]{:}\tau\to\sigma \ \&\ B \vdash M_2[N/x]{:}\tau] & \iff \quad (IH) \\
 & \exists\rho_1,\rho_2,\tau\ [B,x{:}\rho_i \vdash M_1{:}\tau\to\sigma \ \&\ B \vdash N{:}\rho_1 \ \&\ B,x{:}\rho_2 \vdash M_2{:}\tau & \\
 & \qquad \& \ B \vdash N{:}\rho_2] \iff (\rho = \rho_1\cap\rho_2) \ \&\ (\cap I) \ \&\ (\leq) & \\
 & \exists\rho\ [B,x{:}\rho \vdash M_1M_2{:}\sigma \ \&\ B \vdash N{:}\rho]. &
\end{array}$$

We will use the following notation for derivations, that aims to show the structure, in linear notation, of the derivation in terms of rules applied.

Definition 1.7 *i*) If derivation D consists of an application of rule (Ax), then there are $n \geq 1, \sigma_i$ $(\forall i \in \underline{n})$, and B such that $\mathrm{D} :: B,x{:}\cap_{\underline{n}}\sigma_i \vdash x{:}\sigma_j$ with $j \in \underline{n}$; we then write

$$\mathrm{D} = \langle Ax \rangle :: B,x{:}\cap_{\underline{n}}\sigma_i \vdash x{:}\sigma_j.$$

ii) If derivation D finishes with rule $(\to I)$, there are M_1,α,β such that

$$\mathrm{D} :: B \vdash \lambda x.M_1{:}\alpha\to\beta,$$

and there is a sub-derivation $\mathrm{D}_1 :: B, x{:}\alpha \vdash M_1 : \beta$ in D; we then write

$$\mathrm{D} = \langle \mathrm{D}_1, \to I \rangle :: B \vdash \lambda x.M_1 : \alpha \to \beta.$$

iii) If derivation D finishes with rule $(\to E)$, there are P, Q, such that $\mathrm{D} :: B \vdash PQ : \sigma$, and there are τ and sub-derivations $\mathrm{D}_1 :: B \vdash P : \tau \to \sigma$ and $\mathrm{D}_2 :: B \vdash Q : \tau$ in D; we then write

$$\mathrm{D} = \langle \mathrm{D}_1, \mathrm{D}_2, \to E \rangle :: B \vdash PQ : \sigma.$$

iv) If derivation D finishes with rule $(\cap I)$, there are $n \geq 0, \sigma_i$ $(\forall i \in \underline{n})$ such that

$$\mathrm{D} :: B \vdash M : \cap_{\underline{n}} \sigma_i,$$

and, for all $i \in \underline{n}$, there exists a $\mathrm{D}_i :: B \vdash M : \sigma_i$ that is a sub-derivation of D; we then write

$$\mathrm{D} = \langle \mathrm{D}_1, \ldots, \mathrm{D}_n, \cap I \rangle :: B \vdash M : \cap_{\underline{n}} \sigma_i.$$

We will often abbreviate the short-hand notation for derivations, and, for example, write $\langle \mathrm{D}_1, \mathrm{D}_2, \to E \rangle$ instead of $\langle \mathrm{D}_1, \mathrm{D}_2, \to E \rangle :: B \vdash PQ : \sigma$.

We will identify derivations that have the same structure in that they have the same rules applied in the same order (so derivations involving the same term, apart from sub-terms typed by ω) and say that these have the *same structure*; the types derived need not be the same.

As partially shown in [1], we have the following property.

Theorem 1.8 (CF. [1]) *The following rules are admissible:*

$$(\leq) : \frac{B \vdash M : \sigma}{B' \vdash M : \tau} \, (B' \leq B, \sigma \leq \tau) \quad (=_\beta) : \frac{B \vdash M : \sigma}{B \vdash N : \sigma} \, (M =_\beta N) \quad (cut) : \frac{B, x{:}\sigma \vdash M : \tau \quad B \vdash N : \sigma}{B \vdash M[N/x] : \tau}$$

Proof: $(\leq)$: Easy; part $B' \leq B$ follows from rule (Ax), and part $\sigma \leq \tau$ follows by induction on $\leq$, using rule $(\cap I)$.

$(=_\beta)$: By induction on the definition of $=_\beta$. The only part that needs attention is that of a redex, $B \vdash (\lambda x.M)N : \sigma \iff B \vdash M[N/x] : \sigma$, where $\sigma \in \mathcal{T}_S$; all other cases follow by straightforward induction. To conclude, notice that, if $B \vdash (\lambda x.M)N : \sigma$, then, by $(\to E)$ and $(\to I)$, there exists a ρ such that $B, x{:}\rho \vdash M : \sigma$ and $B \vdash N : \rho$; the converse of this result holds, obviously, as well. The result then follows by Lemma 1.6.

(cut): By Lemma 1.6.

2 Derivation reduction

The notion of reduction on derivations $\mathrm{D} :: B \vdash M : \sigma$ defined in this section will follow ordinary reduction (on terms), by contracting typed redexes that occur in D, i.e. redexes for sub-terms of M of the shape $(\lambda x.P)Q$, for which the following is a sub-derivation of D:

$$\langle \langle \mathrm{D}_1 :: B, x{:}\rho \vdash P : \tau, \to I \rangle :: B \vdash \lambda x.P : \rho \to \tau, \mathrm{D}_2 :: B \vdash Q : \rho, \to E \rangle :: B \vdash (\lambda x.P)Q : \tau,$$

A derivation of this structure will be called a *redex*.

We will prove in Section 3 that this notion of reduction is terminating, i.e. strongly normalisable.

The effect of this reduction will be that the derivation for the redex $(\lambda x.P)Q$ will be replaced by a derivation for the contractum $P[Q/x]$; this has, because the system at hand uses intersection types, including ω, to be defined with care, since in $\mathrm{D} :: B \vdash M : \sigma$ it is possible that M contains a redex whereas D does not.

Take the following derivation for $B \vdash (\lambda x.x)N : \sigma$.

$$\dfrac{\dfrac{\dfrac{}{B, x{:}\sigma\cap\tau \vdash x : \sigma}\,(Ax)}{B \vdash \lambda x.x : \sigma\cap\tau\to\sigma}\,(\to I) \qquad \dfrac{\boxed{\mathrm{D}_1}\quad\boxed{\mathrm{D}_2}}{B \vdash N : \sigma \quad B \vdash N : \tau}\,(\cap I)}{B \vdash (\lambda x.x)(N) : \sigma}\,(\to E)$$

This derivation will reduce to $\mathrm{D}_1 :: B \vdash N : \sigma$.

For the general case, consider a derivation for the redex $(\lambda x.P)Q$:

$$\begin{aligned}\langle\langle \mathrm{D}_1 :: B, x{:}\cap_{\underline{n}}\rho_i \vdash P : \tau, \to I\rangle :: B \vdash \lambda x.P : \cap_{\underline{n}}\rho_i \to \tau,\\ \mathrm{D}_2 :: \langle \mathrm{D}_2^1, \ldots, \mathrm{D}_2^n, \cap I\rangle :: B \vdash Q : \cap_{\underline{n}}\rho_i,\ \to E\rangle :: B \vdash (\lambda x.P)Q : \tau,\end{aligned}$$

then the derivation is shaped like:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B, x{:}\cap_{\underline{n}}\rho_i \vdash x : \rho_{k_1}}\,(Ax) \quad \cdots \quad \dfrac{}{B, x{:}\cap_{\underline{n}}\rho_i \vdash x : \rho_{k_m}}\,(Ax)}{\boxed{\mathrm{D}_1}}}{B, x{:}\cap_{\underline{n}}\rho_i \vdash P : \tau}}{B \vdash \lambda x.P : \cap_{\underline{n}}\rho_i \to \tau}\,(\to I) \qquad \dfrac{\boxed{\mathrm{D}_2^1} \quad \cdots \quad \boxed{\mathrm{D}_2^n}}{B \vdash Q : \rho_1 \quad \cdots \quad B \vdash Q : \rho_n}\,(\cap I)}{B \vdash Q : \cap_{\underline{n}}\rho_i}$$
$$\dfrac{}{B \vdash (\lambda x.P)Q : \tau}\,(\to E)$$

Contracting this redex will construct a derivation for the term $P[Q/x]$, and will be written as $\mathrm{D}_1[\mathrm{D}_2/x{:}\cap_{\underline{n}}\rho_i] :: B \vdash P[Q/x] : \tau$.

Notice that the admissible rule (cut) can be applied directly to the derivations D_1 and D_2, thus obtaining

$$\dfrac{\boxed{\mathrm{D}_1} \qquad \boxed{\mathrm{D}_2}}{B, x{:}\cap_{\underline{n}}\rho_i \vdash P : \tau \qquad B \vdash Q : \cap_{\underline{n}}\rho_i}\,(cut)$$
$$B \vdash P[Q/x] : \tau$$

Removing this occurrence of (cut), so to obtain a derivation for $B \vdash P[Q/x] : \tau$ from the one above in which this specific occurrence no longer appears, would require exactly the operations specified in this paper for derivation reduction, and therefore we also use the term cut-elimination for derivation reduction.

When creating a derivation for $P[Q/x]$, it is *not* the case that the derivation D_2 will just be inserted in the positions of D_1 where a type for the variable x is derived: notice that *no* sub-derivation for $B \vdash x : \cap_{\underline{n}}\rho_i$ need exist in D_1. Instead, since each ρ_{k_j} occurs in $\cap_{\underline{n}}\rho_i$, the approach used in this paper for derivation substitution will be to replace all derivations $\langle Ax\rangle :: B, x{:}\cap_{\underline{n}}\rho_i \vdash x : \rho_{k_j}$ by the derivation $\mathrm{D}_2^{k_j} :: B \vdash Q : \rho_{k_j}$, and replace x by Q in P throughout the derivation D_1.

Before formally defining reduction on derivations, we will first define a notion of substitution on derivations.

Definition 2.1 (Derivation substitution) For $\mathrm{D} :: B, x{:}\sigma \vdash M : \tau$, and $\mathrm{D}_0 :: B \vdash N : \sigma$, the derivation $\mathrm{D}[\mathrm{D}_0/x{:}\sigma] :: B \vdash M[N/x] : \tau$, the result of *substituting* D_0 *for* $x{:}\sigma$ *in* D, is inductively defined by:

i) $\mathrm{D} = \langle Ax\rangle :: B, x{:}\sigma \vdash x : \tau$. Let $\sigma = \cap_{\underline{n}}\sigma_i$, then $\tau = \sigma_j$ with $j \in \underline{n}$. Then

$$\mathrm{D}_0 = \langle \mathrm{D}_0^1 :: B \vdash N : \sigma_1, \ldots, \mathrm{D}_0^n :: B \vdash N : \sigma_n, \cap I\rangle :: B \vdash N : \cap_{\underline{n}}\sigma_i,$$

so, in particular, $\mathrm{D}_0^j :: B \vdash N : \sigma_j$. Then $\mathrm{D}[\mathrm{D}_0/x{:}\sigma] = \mathrm{D}_0^j$.

ii) $\mathrm{D} = \langle Ax\rangle :: B, x{:}\sigma \vdash y{:}\tau$ with $x \not\equiv y$. Then

$$\mathrm{D}\,[\mathrm{D}_0/x{:}\sigma] = \langle Ax\rangle :: B \vdash y{:}\tau.$$

iii) $\mathrm{D} = \langle \mathrm{D}_1 :: B, x{:}\sigma, y{:}\alpha \vdash M_1{:}\beta, \to I\rangle :: B, x{:}\sigma \vdash \lambda y.M_1{:}\alpha\to\beta$. Let

$$\mathrm{D}' \;=\; \mathrm{D}_1\,[\mathrm{D}_0/x{:}\sigma] :: B, y{:}\alpha \vdash M_1[N/x]{:}\beta.$$

Then $\langle \mathrm{D}_1, \to I\rangle\,[\mathrm{D}_0/x{:}\sigma] \;=\; \langle \mathrm{D}', \to I\rangle :: B \vdash (\lambda y.M_1)[N/x]{:}\alpha\to\beta$.

iv) $\mathrm{D} = \langle \mathrm{D}_1 :: B, x{:}\sigma \vdash P{:}\rho\to\tau, \mathrm{D}_2 :: B, x{:}\sigma \vdash Q{:}\rho, \to E\rangle :: B, x{:}\sigma \vdash PQ{:}\tau$. Let

$$\begin{aligned} \mathrm{D}_1' \;&=\; \mathrm{D}_1\,[\mathrm{D}_0/x{:}\sigma] :: B \vdash P[N/x]{:}\rho\to\tau, \text{ and} \\ \mathrm{D}_2' \;&=\; \mathrm{D}_2\,[\mathrm{D}_0/x{:}\sigma] :: B \vdash Q[N/x]{:}\rho, \end{aligned}$$

then $\langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle[\mathrm{D}_0/x{:}\sigma] \;=\; \langle \mathrm{D}_1', \mathrm{D}_2', \to E\rangle :: B \vdash (PQ)[N/x]{:}\tau$.

v) $\mathrm{D} = \langle \mathrm{D}_1, \ldots, \mathrm{D}_n, \cap I\rangle :: B, x{:}\sigma \vdash M{:}\cap_{\underline{n}}\tau_i$. Let, for all $i \in \underline{n}$,

$$\mathrm{D}_i' = \mathrm{D}_i\,[\mathrm{D}_0/x{:}\sigma] :: B \vdash M[N/x]{:}\tau_i,$$

then $\langle \mathrm{D}_1, \ldots, \mathrm{D}_n, \cap I\rangle[\mathrm{D}_0/x{:}\sigma] \;=\; \langle \mathrm{D}_1', \ldots, D_n', \cap I\rangle :: B \vdash M[N/x]{:}\cap_{\underline{n}}\tau_i$.

Before coming to the definition of derivation-reduction, we need to define the notion of 'position of a sub-derivation in a derivation'. This notion is needed in Definition 2.3, to make sure that, when contracting a redex in one sub-derivation (branch) in a derivation ending with rule $(\cap I)$, all its 'siblings' in neighbouring branches are contracted as well.

Definition 2.2 Let D be a derivation, and D′ be a sub-derivation of D. The position p of D′ in D is defined by:

i) If $\mathrm{D}' = \mathrm{D}$, then $p = \varepsilon$.

ii) If the position of D′ in D_1 is q, and $\mathrm{D} = \langle \mathrm{D}_1, \to I\rangle$, or $\mathrm{D} = \langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle$, then $p = 1q$.

iii) If the position of D′ in D_2 is q, and $\mathrm{D} = \langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle$, then $p = 2q$.

iv) If the position of D′ in D_i ($i \in \underline{n}$) is q, and $\mathrm{D} = \langle \mathrm{D}_1, \ldots, \mathrm{D}_n, \cap I\rangle$, then $p = q$.

We can now define a notion of reduction on derivations; notice that this reduction corresponds to contracting a redex in the term involved only if that redex appears in the derivation in a sub-derivation with type different from ω.

Definition 2.3 (Derivation reduction) We say that the derivation $\mathrm{D} :: B \vdash M{:}\sigma$ *reduces to* $\mathrm{D}' :: B \vdash M'{:}\sigma$ *at position* p *with redex* R, if and only if:

$(\sigma \in \mathcal{T}_S)$: *a*) $\mathrm{D} = \langle\langle \mathrm{D}_1, \to I\rangle, \mathrm{D}_2, \to E\rangle :: B \vdash (\lambda x.M)N{:}\sigma$ (a derivation of this shape is called a *redex*); then D reduces to

$$\mathrm{D}_1\,[\mathrm{D}_2/x{:}\rho] :: B \vdash M[N/x]{:}\sigma$$

at position ε with redex $(\lambda x.M)N$.

b) If D_1 reduces to D_1' at position p with redex R, then

- $\mathrm{D} = \langle \mathrm{D}_1, \to I\rangle :: B \vdash \lambda x.M_1{:}\alpha\to\beta$ reduces to $\mathrm{D}' = \langle \mathrm{D}_1', \to I\rangle :: B \vdash \lambda x.M_1'{:}\alpha\to\beta$ at position $1p$ with redex R.
- $\mathrm{D} = \langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle :: B \vdash PQ{:}\sigma$ reduces to $\mathrm{D}' = \langle \mathrm{D}_1', \mathrm{D}_2, \to E\rangle :: B \vdash P'Q{:}\sigma$ at position $1p$ with redex R.
- $\mathrm{D} = \langle \mathrm{D}_2, \mathrm{D}_1, \to E\rangle :: B \vdash PQ{:}\sigma$ reduces to $\mathrm{D}' = \langle \mathrm{D}_2, \mathrm{D}_1', \to E\rangle :: B \vdash PQ'{:}\sigma$ at position $2p$ with redex R.

$(\sigma = \cap_{\underline{n}}\sigma_i)$: If $\mathrm{D} :: B \vdash M{:}\cap_{\underline{n}}\sigma_i$, then, for every $i \in \underline{n}$, there are $\mathrm{D}_i :: B \vdash M{:}\sigma_i$ such that $\mathrm{D} = \langle \mathrm{D}_1, \ldots, \mathrm{D}_n, \cap I\rangle$. If there is an $i \in \underline{n}$ such that D_i reduces to D'_i at position p with redex R, then, for all $j \neq i \in \underline{n}$, either

a) there is no redex at position p because there is no sub-derivation at that position. Since R is a sub-term of M, it has to be part of a term that is typed with ω in D_j. Let $\mathrm{R} \to_\beta \mathrm{R}'$ and $\mathrm{D}'_j = \mathrm{D}_j[\mathrm{R}'/\mathrm{R}]$ (i.e. D_j where each R is replaced by R′), or

b) D_j reduces to D'_j at position p with redex R.

Then D reduces to $\langle \mathrm{D}'_1, \ldots, \mathrm{D}'_n, \cap I\rangle$ at position p with redex R.

We write $\mathrm{D} \to_{\mathcal{D}} \mathrm{D}'$ if there exists a position p and redex R such that D reduces to D′ at position p with redex R. We will use the symbol $\to_{\mathcal{D}}$ also for its transitive closure: if $\mathrm{D}_1 \to_{\mathcal{D}} \mathrm{D}_2 \to_{\mathcal{D}} \mathrm{D}_3$, then $\mathrm{D}_1 \to_{\mathcal{D}} \mathrm{D}_3$.

We say that D is *normalisable* is there exists a redex-free D′ such that $\mathrm{D} \to_{\mathcal{D}} \mathrm{D}'$, and that D is *strongly normalisable* if all reduction sequences starting in D are of finite length. We abbreviate 'D is strongly normalisable' by '$SN(\mathrm{D})$'.

It is worth noting that typeable terms need not be strongly normalising, even when we do not allow the use of ω to type a redex, as clearly illustrated by the following example.

Example 2.4 Let D_1 be the derivation (with $B_1 = x{:}(\alpha\to\beta\to\gamma)\cap\alpha, y{:}(\gamma\to\delta)\cap\beta$, and $\Theta \equiv \lambda xy.y(xxy)$):

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B_1 \vdash y{:}\gamma\to\delta}(Ax) \qquad \dfrac{\dfrac{\dfrac{}{B_1 \vdash x{:}\alpha\to\beta\to\gamma}(Ax) \quad \dfrac{}{B_1 \vdash x{:}\alpha}(Ax)}{B_1 \vdash xx{:}\beta\to\gamma}(\to E) \quad \dfrac{}{B_1 \vdash y{:}\beta}(Ax)}{B_1 \vdash xxy{:}\gamma}(\to E)}{B_1 \vdash y(xxy){:}\delta}(\to E)}{B_1\backslash y \vdash \lambda y.y(xxy){:}(\gamma\to\delta)\cap\beta \to \delta}(\to I)}{\vdash \Theta{:}(\alpha\to\beta\to\gamma)\cap\alpha \to (\gamma\to\delta)\cap\beta \to \delta}(\to I)$$

Let $B_2 = x{:}\tau, y{:}\omega\to\rho$, and $\tau = (\alpha\to\beta\to\gamma)\cap\alpha \to (\gamma\to\delta)\cap\beta \to \delta$ (the type derived in D_1), then we can construct D_2:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B_2 \vdash y{:}\omega\to\rho}(Ax) \quad \dfrac{}{B_2 \vdash xxy{:}\omega}(\cap I)}{B_2 \vdash y(xxy){:}\rho}(\to E)}{x{:}\tau \vdash \lambda y.y(xxy){:}(\omega\to\rho)\to\rho}(\to I)}{\vdash \Theta{:}\tau\to(\omega\to\rho)\to\rho}(\to I)$$

From D_1 and D_2 we can now construct:

$$\dfrac{\boxed{\mathrm{D}_2} \quad \boxed{\mathrm{D}_1}}{\vdash \Theta{:}\tau\to(\omega\to\rho)\to\rho \quad \vdash \Theta{:}\tau}(\to E)$$

Notice that the term $\Theta\Theta$ has only *one* redex, that is not typed with ω. Also, this derivation has only one (derivation)-redex, and contracting it gives:

$$\dfrac{\dfrac{\dfrac{}{y{:}\omega\to\rho \vdash y{:}\omega\to\rho}(Ax) \qquad \dfrac{}{y{:}\omega\to\rho \vdash \Theta\Theta y{:}\omega}(\cap I)}{y{:}\omega\to\rho \vdash y(\Theta\Theta y){:}\rho}(\to E)}{\vdash \lambda y.y(\Theta\Theta y){:}(\omega\to\rho)\to\rho}(\to I)$$

Notice that this last derivation is in normal form, although $\lambda y.y(\Theta\Theta y)$ obviously is not.

For another, more involved example of derivation reduction, see Example A.3 in the appendix.

The following lemma formulates the relation between derivation reduction and β-reduction.

Lemma 2.5 *Let* $\mathrm{D} :: B \vdash M:\sigma$, *and* $\mathrm{D} \to_{\mathcal{D}} \mathrm{D}' :: B \vdash N:\sigma$, *then* $M \twoheadrightarrow_{\beta} N$.

Proof: By Definition 2.3.

The following states some standard properties of strong normalisation.

Lemma 2.6 *i) If* $SN(\langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle)$, *then* $SN(\mathrm{D}_1)$ *and* $SN(\mathrm{D}_2)$.

ii) If $SN(\mathrm{D}_1 :: B_1 \vdash xM_1 \ldots M_n : \sigma \to \tau)$ *and* $SN(\mathrm{D}_2 :: B_2 \vdash N:\sigma)$, *then* $SN(\langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle :: \bigcap\{B_1, B_2\} \vdash xM_1 \ldots M_n N : \tau)$.

iii) $\forall i \in \underline{n}\ [SN(\mathrm{D}_i :: B \vdash M:\sigma_i)]$ *if and only if* $SN(\langle \mathrm{D}_1, \ldots, \mathrm{D}_n, \cap I\rangle)$.

iv) If $SN(\langle \ldots \langle \mathrm{D}_1[\mathrm{D}_2/y{:}\rho]\rangle \ldots\rangle :: B \vdash M[N/x]\vec{P}:\sigma)$ *and* $SN(\mathrm{D}_2 :: B \vdash N:\rho)$, *then* $SN(\langle \ldots \langle\langle \mathrm{D}_1, \to I\rangle, \mathrm{D}_2, \to E\rangle \ldots\rangle :: B \vdash (\lambda y.M)Q\vec{P}:\sigma)$.

Proof: Easy, by Definition 2.3.

3 Strong normalisation of derivation reduction

In this subsection, we will prove a strong normalisation result for derivation reduction. In order to prove that each derivation in $\vdash$ is strongly normalisable with respect to $\to_{\mathcal{D}}$, a notion of computable [26, 20] derivations will be introduced. We will show that all computable derivations are strongly normalisable with respect to derivation reduction, and then that all derivations in $\vdash$ are computable.

Definition 3.1 (Computability Predicate) *Comp* (D) is defined recursively on types by:

$$\begin{array}{lll}
\mathit{Comp}(\mathrm{D} :: B \vdash M:\varphi) & \iff & SN(\mathrm{D}) \\
\mathit{Comp}(\mathrm{D} :: B \vdash M:\alpha\to\beta) & \iff & \\
\qquad \forall \mathrm{D}'\ [\mathit{Comp}(\mathrm{D}' :: B \vdash N:\alpha) & \Rightarrow & \mathit{Comp}(\langle \mathrm{D}, \mathrm{D}', \to E\rangle :: B \vdash MN:\beta)] \\
\mathit{Comp}(\langle \mathrm{D}_1, \ldots, \mathrm{D}_n, \cap I\rangle :: B \vdash M:\cap_{\underline{n}}\sigma_i) & \iff & \forall i \in \underline{n}\ [\mathit{Comp}(\mathrm{D}_i :: B \vdash M:\sigma_i)]
\end{array}$$

Notice that, as a special case for the third rule, we get $\mathit{Comp}(\langle \cap I\rangle :: B \vdash M:\omega)$

Lemma 3.2 *If* $\mathit{Comp}(\mathrm{D} :: B \vdash M:\sigma)$, $B' \leq B$, $\sigma \leq \sigma'$, *then* $\mathit{Comp}(\mathrm{D}' :: B' \vdash M:\sigma')$ *for some* D'.

Proof: By straightforward induction on the structure of types.

We will prove that *Comp* satisfies the standard properties of computability predicates, being that computability implies strong normalisation, and that, for the so-called *neutral* objects, also the converse holds.

Lemma 3.3 *i)* $\mathit{Comp}(\mathrm{D} :: B \vdash M:\sigma) \Rightarrow SN(\mathrm{D})$.

ii) $SN(\mathrm{D} :: B \vdash xM_1 \ldots M_m : \sigma) \Rightarrow \mathit{Comp}(\mathrm{D})$.

Proof: By simultaneous induction on the structure of types.

$(\sigma = \varphi)$: Directly by Definition 3.1.

$(\sigma = \alpha{\to}\beta)$: *a*) Let x be a variable not appearing in B and M, and let $\mathrm{D}' = \langle Ax\rangle :: B, x{:}\alpha \vdash x{:}\alpha$, then, by induction ((*ii*)), *Comp* (D').
Since *Comp* (D), by Lemma 3.2, also *Comp* $(\mathrm{D}'' :: B, x{:}\alpha \vdash M{:}\alpha{\to}\beta)$ (notice that D and D″ are almost identical, but for the occurrences of $x{:}\alpha$ in the basis), so, by Definition 3.1, *Comp* $(\langle \mathrm{D}'', \mathrm{D}', {\to}E\rangle :: B, x{:}\alpha \vdash Mx{:}\beta)$. Then, by induction ((*i*)), *SN* $(\langle \mathrm{D}'', \mathrm{D}', {\to}E\rangle)$, and, by Lemma 2.6:(*i*), *SN* (D''). Then also *SN* (D).

b) Assume *Comp* $(\mathrm{D}' :: B' \vdash N{:}\alpha)$, then by induction ((*i*)), *SN* (D'). Then, by Lemma 2.6:(*ii*), *SN* $(\langle \mathrm{D}, \mathrm{D}', {\to}E\rangle :: \bigcap\{B, B'\} \vdash xM_1 \ldots M_m N{:}\beta)$. Then *Comp* $(\langle \mathrm{D}, \mathrm{D}', {\to}E\rangle)$ by induction ((*ii*)), so by Definition 3.1, *Comp* (D).

$(\sigma = \cap_{\underline{n}}\sigma_i)$: Easy, using Definition 3.1, Lemma 2.6:(*iii*), and induction.

The following theorem (3.5) shows that, in a derivation, replacing sub-derivations for term-variables by computable derivations yields a computable derivation. Before coming to this result, first an auxiliary lemma has to be proved, that formulates that the computability predicate is closed for subject-expansion with respect to derivation reduction.

Lemma 3.4 *Let* $\mathrm{D} = \langle \ldots \mathrm{D}_1[\mathrm{D}_2/y{:}\rho] \ldots, {\to}E\rangle :: B \vdash M[Q/y]\vec{P}{:}\sigma$;
if *Comp* (D) *and* *Comp* $(\mathrm{D}_2 :: B \vdash Q{:}\rho)$, *then*

$$\mathit{Comp}\,(\langle \ldots \langle\langle \mathrm{D}_1, {\to}I\rangle, \mathrm{D}_2, {\to}E\rangle \ldots, {\to}E\rangle :: B \vdash (\lambda y.M)Q\vec{P}{:}\sigma).$$

Proof: By induction on the structure of types.

i) $\sigma = \varphi$.

Comp (D) & *Comp* $(\mathrm{D}_2 :: B \vdash Q{:}\rho)$	$\Rightarrow$ (3.3:(*i*))
SN (D) & *SN* (D_2)	$\Rightarrow$ (2.6:(*iv*))
SN $(\langle \ldots \langle\langle \mathrm{D}_1, {\to}I\rangle, \mathrm{D}_2, {\to}E\rangle \ldots, {\to}E\rangle :: B \vdash (\lambda y.M)Q\vec{P}{:}\varphi)$	$\Rightarrow$ (3.1)
Comp $(\langle \ldots \langle\langle \mathrm{D}_1, {\to}I\rangle, \mathrm{D}_2, {\to}E\rangle \ldots, {\to}E\rangle)$.	

ii) $\sigma = \alpha{\to}\beta$.

Comp (D) & *Comp* (D_2) & *Comp* $(\mathrm{D}' :: B \vdash N{:}\alpha)$	$\Rightarrow$ (3.1)
Comp (D_2) & *Comp* $(\langle \mathrm{D}, \mathrm{D}', {\to}E\rangle :: B \vdash M[Q/y]\vec{P}N{:}\beta)$	$\Rightarrow$ (*IH*)
Comp $(\langle\langle\langle \ldots \langle \mathrm{D}_1, {\to}I\rangle, \mathrm{D}_2, {\to}E\rangle \ldots, {\to}E\rangle, \mathrm{D}', {\to}E\rangle :: B \vdash (\lambda y.M)Q\vec{P}N{:}\beta)$	$\Rightarrow$ (3.1)
Comp $(\langle \ldots \langle\langle \mathrm{D}_1, {\to}I\rangle, \mathrm{D}_2, {\to}E\rangle \ldots, {\to}E\rangle)$.	

iii) $\sigma = \cap_{\underline{n}}\sigma_i$. By induction and Definition 3.1.

We now come to the Replacement Theorem, which states that, for every derivation in $\vdash$, if the assumptions in the derivation are to be replaced by computable derivations, then the result itself will be computable. We will use an abbreviated notation, and write $[\overrightarrow{N/x}]$ for $[N_1/x_1, \ldots, N_n/x_n]$, etc.

Theorem 3.5 (Replacement Theorem) *Let* $B' = x_1{:}\mu_1, \ldots, x_m{:}\mu_m$, $\mathrm{D}^0 :: B' \vdash M{:}\sigma$, *and assume, for all* $i \in \underline{m}$, *that there are* D_i, N_i *such that* *Comp* $(\mathrm{D}_i :: B \vdash N_i{:}\mu_i)$. *Then*

$$\mathit{Comp}\,(\mathrm{D}^0\,[\overrightarrow{\mathrm{D}/x{:}\mu}] :: B \vdash M[\overrightarrow{N/x}]{:}\sigma).$$

Proof: By induction on the structure of derivations.

(Ax): Then $x{:}\cap_{\underline{n}}\sigma_i \in B'$, $\sigma = \sigma_i$ for some $i \in \underline{n}$, and $\mathrm{D}_i :: B \vdash N_i{:}\sigma_i$. By Definition 3.1, *Comp* (D_i), and by Definition 2.1, $\mathrm{D}^0\,[\overrightarrow{\mathrm{D}/x{:}\mu}] = \mathrm{D}_i$.

$(\cap I)$: Then $\sigma = \cap_{\underline{n}}\sigma_i$, and, for all $i \in \underline{n}$, there exists $\mathrm{D}^i :: B' \vdash M{:}\sigma_i$ such that $\mathrm{D}^0 = \langle \mathrm{D}^1, \ldots, \mathrm{D}^n, \cap I\rangle$.

Then, by induction, for all $i \in \underline{n}$,

$$Comp\,(\mathrm{D}^i[\overrightarrow{\mathrm{D}/x{:}\mu}] :: B \vdash M[\overrightarrow{N/x}]{:}\sigma_i),$$

and, by Definition 3.1, $Comp\,(\mathrm{D}^0[\overrightarrow{\mathrm{D}/x{:}\mu}] :: B \vdash M[\overrightarrow{N/x}]{:}\cap_{\underline{n}}\sigma_i)$.

$(\to I)$: Then $\sigma = \rho\to\tau$, $\mathrm{D}^0 = \langle \mathrm{D}^1 :: B', y{:}\rho \vdash M'{:}\tau, \to I\rangle :: B' \vdash \lambda y.M'{:}\rho\to\tau$.
Assume $Comp\,(\mathrm{D}' :: B \vdash Q{:}\rho)$, then:

$$\begin{array}{ll} \forall j \in \underline{m}\ [Comp\,(\mathrm{D}_j)]\ \&\ Comp\,(\mathrm{D}') & \Rightarrow (IH) \\ Comp\,(\mathrm{D}^1[\overrightarrow{\mathrm{D}/x{:}\mu}, \mathrm{D}'/y{:}\rho] :: B \vdash M[\overrightarrow{N/x}, Q/y]{:}\tau) & \Rightarrow (3.4) \\ Comp\,(\langle\langle \mathrm{D}^1[\overrightarrow{\mathrm{D}/x{:}\mu}], \to I\rangle, \mathrm{D}', \to E\rangle :: B \vdash (\lambda y.M[\overrightarrow{N/x}])Q{:}\tau) & \end{array}$$

so, by Definition 3.1, $Comp\,(\langle \mathrm{D}^1[\overrightarrow{\mathrm{D}/x{:}\mu}], \to I\rangle :: B \vdash \lambda y.M[\overrightarrow{N/x}]{:}\rho\to\tau)$, so also

$$Comp\,(\langle \mathrm{D}^1, \to I\rangle[\overrightarrow{\mathrm{D}/x{:}\mu}] :: B \vdash (\lambda y.M)[\overrightarrow{N/x}]{:}\rho\to\tau).$$

$(\to E)$: Then $M \equiv M_1M_2$, there are $\mathrm{D}^1, \mathrm{D}^2$, and τ such that $\mathrm{D}^0 = \langle \mathrm{D}^1, \mathrm{D}^2, \to E\rangle$, $\mathrm{D}^1 :: B' \vdash M_1{:}\tau\to\sigma$, and $\mathrm{D}^2 :: B' \vdash M_2{:}\tau$. Then, by induction,

$$\begin{array}{l} Comp\,(\mathrm{D}^1[\overrightarrow{\mathrm{D}/x{:}\mu}] :: B \vdash M_1[\overrightarrow{N/x}]{:}\tau\to\sigma), \text{ and} \\ Comp\,(\mathrm{D}^2[\overrightarrow{\mathrm{D}/x{:}\mu}] :: B \vdash M_2[\overrightarrow{N/x}]{:}\tau). \end{array}$$

Then, by Definition 3.1,

$$Comp\,(\langle \mathrm{D}^1[\overrightarrow{\mathrm{D}/x{:}\mu}], \mathrm{D}^2[\overrightarrow{\mathrm{D}/x{:}\mu}], \to E\rangle :: B \vdash M_1[\overrightarrow{N/x}]M_2[\overrightarrow{N/x}]{:}\sigma)$$

so also $Comp\,(\langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle[\overrightarrow{\mathrm{D}/x{:}\mu}] :: B \vdash (M_1M_2)[\overrightarrow{N/x}]{:}\sigma)$.

Using this last result, we now prove a strong normalisation result for derivation reduction in $\vdash$.

Theorem 3.6 *If* $\mathrm{D} :: B \vdash M{:}\sigma$, *then* $SN\,(\mathrm{D})$.

Proof: By Lemma 3.3:(*ii*), for every $x{:}\tau \in B$, $Comp\,(\mathrm{D}_x :: B \vdash x{:}\tau)$, so $Comp\,(\mathrm{D} :: B \vdash M{:}\sigma)$ by Theorem 3.5. Then, by Lemma 3.3:(*i*), $SN\,(\mathrm{D})$.

4 Approximation

In Sections 5 and 6 we will show two main results, that are both direct consequences of the strong normalisation result proved in Section 3. Both results have been proven, at least partially, in [1, 2]. In fact, some of the theorems and lemmas presented here were already presented in those papers and are repeated here, for completeness, with their new proofs. Before we come to those results, we will revise approximants.

The notion of approximant for lambda terms was first presented in [27], and is defined using the notion of terms in $\lambda\bot$-normal form (as in [9], $\bot$ is used, instead of Ω; also, the symbol $\sqsubseteq$ is used as a relation on $\Lambda\bot$-terms, inspired by a similar relation defined on Böhm-trees in [9]).

Definition 4.1 (Approximate normal forms) *i*) The set of $\Lambda\bot$-*terms* is defined like the set Λ of lambda terms, by:

$$M \ ::= \ x \mid \bot \mid \lambda x.M \mid M_1M_2$$

The symbol $\bot$ is called *bottom*.

ii) The notion of reduction $\to_{\beta\bot}$ is defined as $\to_\beta$, extended by: $\lambda x.\bot \to_{\beta\bot} \bot$ and $\bot M \to_{\beta\bot} \bot$.

iii) The set of *normal forms for elements of* $\Lambda\bot$ *with respect to* $\to_{\beta\bot}$ is the set $\mathcal{N}$ of $\lambda\bot$-*normal*

forms or *approximate normal forms*, ranged over by A, inductively defined by:

$$A \ ::= \ \bot \mid \lambda x.A \ (A \neq \bot) \mid xA_1 \ldots A_n \ (n \geq 0)$$

The rules of the system $\vdash$ are generalised to terms containing $\bot$ by allowing for the terms to be elements of $\Lambda\bot$. Then, if $\bot$ occurs in a term M and D :: $B \vdash M{:}\sigma$, in D, $\bot$ has to appear in a position where the rule $(\cap I)$ is used with $n = 0$, i.e., in a sub-term typed with ω. Notice that the terms $\lambda x.\bot$ and $\bot M_1 \ldots M_n$ are typeable by ω only.

Definition 4.2 ($\sqsubseteq$, (DIRECT) APPROXIMANTS) *i*) The partial order $\sqsubseteq \ \subseteq (\Lambda\bot)^2$ is defined as the least pre-order such that:

$$\begin{array}{rcl} \bot & \sqsubseteq & M \\ M \sqsubseteq M' & \Rightarrow & \lambda x.M \sqsubseteq \lambda x.M' \\ M_1 \sqsubseteq M_1' \ \& \ M_2 \sqsubseteq M_2' & \Rightarrow & M_1M_2 \sqsubseteq M_1'M_2'. \end{array}$$

If $A \in \mathcal{N}$, $M \in \Lambda$, and $A \sqsubseteq M$, then A is called a *direct approximant* of M.

ii) The relation $\sqsubseteq\!\!\!\sim \ \subseteq \mathcal{N} \times \Lambda$ is defined by:

$$A \sqsubseteq\!\!\!\sim M \iff \exists M' =_\beta M\,[A \sqsubseteq M'].$$

iii) If $A \sqsubseteq\!\!\!\sim M$, then A is called an *approximant* of M, and $\mathcal{A}(M) = \{\, A \in \mathcal{N} \mid A \sqsubseteq\!\!\!\sim M \,\}$.

Lemma 4.3 $B \vdash M{:}\sigma \ \& \ M \sqsubseteq M' \ \Rightarrow \ B \vdash M'{:}\sigma$.

Proof: By easy induction on the definition of $\sqsubseteq$; the base case, $\bot \sqsubseteq M$, follows from the fact that then $\sigma = \omega$.

The following definition introduces an operation of join on $\Lambda\bot$-terms.

Definition 4.4 (JOIN, COMPATIBLE TERMS) *i*) On $\Lambda\bot$, the partial mapping *join*, $\sqcup : \Lambda\bot \times \Lambda\bot \to \Lambda\bot$, is defined by:

$$\begin{array}{rcl} \bot \sqcup M & \equiv & M \sqcup \bot \equiv M \\ x \sqcup x & \equiv & x \\ (\lambda x.M) \sqcup (\lambda x.N) & \equiv & \lambda x.(M \sqcup N) \\ (M_1M_2) \sqcup (N_1N_2) & \equiv & (M_1 \sqcup N_1)\,(M_2 \sqcup N_2) \end{array}$$

ii) If $M \sqcup N$ is defined, then M and N are called *compatible*.

We will use $\sqcup_{\underline{n}} M_i$ for the term $M_1 \sqcup \ldots \sqcup M_n$. Note that $\bot$ can be defined as the empty join, i.e. if $M \equiv \sqcup_{\underline{n}} M_i$, and $n = 0$, then $M \equiv \bot$.

The last alternative in the definition of $\sqcup$ defines the join on applications in a more general way than Scott's, that would state that

$$(M_1M_2) \sqcup (N_1N_2) \ \sqsubseteq \ (M_1 \sqcup N_1)(M_2 \sqcup N_2),$$

since it is not always sure if a join of two arbitrary terms exists. However, we will use our more general definition only on terms that are compatible, so the conflict is only apparent.

The following lemma shows that the join acts as least upper bound of compatible terms.

Lemma 4.5 If $M_1 \sqsubseteq M$, and $M_2 \sqsubseteq M$, then $M_1 \sqcup M_2$ is defined, and:

$$M_1 \sqsubseteq M_1 \sqcup M_2, \ \ M_2 \sqsubseteq M_1 \sqcup M_2, \ \ \textit{and} \ \ M_1 \sqcup M_2 \sqsubseteq M.$$

Proof: By induction on the definition of $\sqsubseteq$.

i) If $M_1 \equiv \bot$, then $M_1 \sqcup M_2 \equiv M_2$, so $M_1 \sqsubseteq M_1 \sqcup M_2, M_2 \sqsubseteq M_1 \sqcup M_2$, and $M_1 \sqcup M_2 \sqsubseteq M_2 \sqsubseteq M$. (The case $M_2 \equiv \bot$ goes similarly.)

ii) If $M_1 \equiv x$, then $M \equiv x$, and either $M_2 = \bot$ or also $M_2 \equiv x$. The first case has been dealt with in part (*i*), and for the other: $M_1 \sqcup M_2 \equiv x$. Obviously, $x \sqsubseteq x \sqcup x$, $x \sqsubseteq x \sqcup x$, and $x \sqcup x \sqsubseteq x$.

iii) If $M_1 \equiv \lambda x.N_1$, then $M \equiv \lambda x.N$, $N_1 \sqsubseteq N$, and either $M_2 = \bot$ or $M_2 \equiv \lambda x.N_2$. The first case has been dealt with in part (*i*), and for the other: then $N_2 \sqsubseteq N$. Then, by induction, $N_1 \sqsubseteq N_1 \sqcup N_2$, $N_2 \sqsubseteq N_1 \sqcup N_2$, and $N_1 \sqcup N_2 \sqsubseteq N$. Then also $\lambda x.N_1 \sqsubseteq \lambda x.N_1 \sqcup N_2$, $\lambda x.N_2 \sqsubseteq \lambda x.N_1 \sqcup N_2$, and $\lambda x.N_1 \sqcup N_2 \sqsubseteq \lambda x.N$. Notice that $\lambda x.N_1 \sqcup N_2 \equiv (\lambda x.N_1) \sqcup (\lambda x.N_2)$.

iv) If $M_1 \equiv P_1 Q_1$, then $M \equiv PQ$, $P_1 \sqsubseteq P$, $Q_1 \sqsubseteq Q$, and either $M_2 = \bot$ or $M_2 \equiv P_2 Q_2$. The first case has been dealt with in part (*i*), and for the other: then $P_2 \sqsubseteq P$, $Q_2 \sqsubseteq Q$. By induction, we know $P_1 \sqsubseteq P_1 \sqcup P_2$, $P_2 \sqsubseteq P_1 \sqcup P_2$, and $P_1 \sqcup P_2 \sqsubseteq P$, as well as $Q_1 \sqsubseteq Q_1 \sqcup Q_2$, $Q_2 \sqsubseteq Q_1 \sqcup Q_2$, and $Q_1 \sqcup Q_2 \sqsubseteq Q$. Then also $P_1 Q_1 \sqsubseteq (P_1 \sqcup P_2)(Q_1 \sqcup Q_2)$, $P_2 Q_2 \sqsubseteq (P_1 \sqcup P_2)(Q_1 \sqcup Q_2)$, and $(P_1 \sqcup P_2)(Q_1 \sqcup Q_2) \sqsubseteq PQ$ Notice that $(P_1 \sqcup P_2)(Q_1 \sqcup Q_2) \equiv (P_1 Q_1) \sqcup (P_2 Q_2)$.

5 Normalisation results

In what follows below, first an approximation result will be proved, i.e. for every M, B and σ such that $B \vdash M : \sigma$, there exists an $A \in \mathcal{A}(M)$ such that $B \vdash A : \sigma$. From this, the well-known characterisation of (head-)normalisation of lambda terms using intersection types follows easily, i.e. all terms having a (head) normal form are typeable in $\vdash$ (with a type without ω-occurrences). The second result is the well-known characterisation of strong normalisation of typeable lambda terms, i.e. all terms, typeable in $\vdash$ without using the type-constant ω, are strongly normalisable.

First we give some auxiliary definitions and results. The first is a notion of type assignment similar to that of Definition 1.5, but differs in that it assigns ω only to the term $\bot$.

Definition 5.1 $\bot$*-type assignment* and $\bot$*-derivations* are defined by the following natural deduction system (where σ in rules $(\to E)$ and $(\to I)$ is in $\mathcal{T}$):

$$(Ax): \frac{}{B, x{:}\cap_{\underline{n}}\sigma_i \vdash_\bot x : \sigma_i}\ (n \geq 1, i \in \underline{n}) \qquad (\to E): \frac{B \vdash_\bot M : \sigma \to \tau \quad B \vdash_\bot N : \sigma}{B \vdash_\bot MN : \tau}$$

$$(\cap I): \frac{B \vdash_\bot M_1 : \sigma_1 \quad \ldots \quad B \vdash_\bot M_n : \sigma_n}{B \vdash_\bot \sqcup_{\underline{n}} M_i : \cap_{\underline{n}} \sigma_i}\ (n \geq 0) \qquad (\to I): \frac{B, x{:}\sigma \vdash_\bot M : \tau}{B \vdash_\bot \lambda x.M : \sigma \to \tau}$$

We write $B \vdash_\bot M : \sigma$ if this statement is derivable using a $\bot$-derivation.

Notice that, by rule $(\cap I)$, $\emptyset \vdash_\bot \bot : \omega$, and that this is the only way to assign ω to a term. Moreover, in that rule, the terms M_j need to be compatible (otherwise their join would not be defined).

Lemma 5.2 *i) If* $\mathrm{D} :: B \vdash_\bot M : \sigma$, *then* $\mathrm{D} :: B \vdash M : \sigma$.

ii) If $\mathrm{D} :: B \vdash M : \sigma$, *then there exists* $M' \sqsubseteq M$, *such that* $\mathrm{D} :: B \vdash_\bot M' : \sigma$.

Proof: *i*) By induction on the structure of derivations in $\vdash_\bot$.

(Ax): Immediate.

$(\cap I)$: Then, there are $n \geq 0, \sigma_i, M_i$ $(\forall i \in \underline{n})$ such that $\sigma = \cap_{\underline{n}} \sigma_i$, $M = \sqcup_{\underline{n}} M_i$, and, for every $i \in \underline{n}$, $B \vdash_\bot M_i : \sigma_i$. Then, by induction, for every $i \in \underline{n}$, $B \vdash M_i : \sigma_i$. Since, by Lemma 4.5,

$M_i \sqsubseteq M$ for all $i \in \underline{n}$, by Lemma 4.3, for every $i \in \underline{n}$, $B \vdash M : \sigma_i$, so by $(\cap I)$, $B \vdash M : \cap_{\underline{n}} \sigma_i$.

$(\to I)$: Then $M \equiv \lambda x.M'$, and $\sigma = \alpha \to \beta$, and $B, x{:}\alpha \vdash_\perp M' : \beta$. Then, by induction, $B, x{:}\alpha \vdash M' : \beta$, so by $(\to I)$, $B \vdash \lambda x.M' : \alpha \to \beta$.

$(\to E)$: Then $M \equiv M_1 M_2$, and there exists τ such that $B \vdash_\perp M_1 : \tau \to \sigma$, and $B \vdash_\perp M_2 : \tau$. Then, by induction, $B \vdash M_1 : \tau \to \sigma$, and $B \vdash M_2 : \tau$, so by $(\to E)$, $B \vdash M_1 M_2 : \sigma$.

ii) By induction on the structure of derivations in $\vdash$.

(Ax): Immediate.

$(\cap I)$: Then $\sigma = \cap_{\underline{n}} \sigma_i$ and, for every $i \in \underline{n}$, $B \vdash M : \sigma_i$, and, by induction, for every $i \in \underline{n}$ there exists $M_i \sqsubseteq M$ such that $B \vdash_\perp M_i : \sigma_i$ (notice that then these M_i are compatible). Then, by rule $(\cap I)$, we have $B \vdash_\perp \sqcup_{\underline{n}} M_i : \sigma_i$. Notice that, by Lemma 4.5, $\sqcup_{\underline{n}} M_i \sqsubseteq M$.

$(\to I)$: Then $M \equiv \lambda x.M_1$, and $\sigma = \alpha \to \beta$, and $B, x{:}\alpha \vdash M_1 : \beta$. So, by induction, there exists $M_1' \sqsubseteq M_1$ such that $B, x{:}\alpha \vdash_\perp M_1' : \beta$. Then, by rule $(\to I)$ we obtain $B \vdash_\perp \lambda x.M_1' : \alpha \to \beta$. Notice that $\lambda x.M_1' \sqsubseteq \lambda x.M_1$.

$(\to E)$: Then $M \equiv M_1 M_2$, and there is a τ such that $B \vdash M_1 : \tau \to \sigma$, and $B \vdash M_2 : \tau$. Then, by induction, there are $M_1' \sqsubseteq M_1$, and $M_2' \sqsubseteq M_2$, such that $B \vdash_\perp M_1' : \tau \to \sigma$, and $B \vdash_\perp M_2' : \tau$. Then, by $(\to E)$, $B \vdash_\perp M_1' M_2' : \sigma$. Notice that $M_1' M_2' \sqsubseteq M_1 M_2$.

Notice that the case $\sigma = \omega$ is present in the case $(\cap I)$ of the proof. Then $n = 0$, and $\sqcup_{\underline{n}} M_i = \perp$. Moreover, since M' need not be the same as M, the second derivation in part (*ii*) is not exactly the same; however, it has the same structure in terms of applied derivation rules.

Example 5.3 Let D_1' be the derivation D_1 from Example 2.4, but built using the rules of $\vdash_\perp$ rather than $\vdash$ (notice that ω is not used in D_1, so there is no difference); let $B_2 = \{x{:}\tau, y{:}\omega \to \rho\}$, and τ (as in Example 2.4) the type derived in D_1', then the $\vdash_\perp$-variant of $\langle \mathrm{D}_2, \mathrm{D}_1, \to E \rangle$ will be:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B_2 \vdash_\perp y : \omega \to \rho}\,(Ax) \quad \dfrac{}{B_2 \vdash_\perp \perp : \omega}\,(\cap I)}{B_2 \vdash_\perp y\perp : \rho}\,(\to E)}{x{:}\tau \vdash_\perp \lambda y.y\perp : (\omega \to \rho) \to \rho}\,(\to I)}{\vdash_\perp \lambda xy.y\perp : \tau \to (\omega \to \rho) \to \rho}\,(\to I) \quad \begin{array}{c}\boxed{\mathrm{D}_1'} \\ \vdash_\perp \Theta : \tau\end{array}}{\vdash_\perp (\lambda xy.y\perp)\Theta : (\omega \to \rho) \to \rho}\,(\to E)$$

Notice that $\lambda xy.y\perp \sqsubseteq \Theta$. This derivation reduces to:

$$\dfrac{\dfrac{\dfrac{}{y{:}\omega \to \rho \vdash_\perp y : \omega \to \rho}\,(Ax) \quad \dfrac{}{y{:}\omega \to \rho \vdash_\perp \perp : \omega}\,(\cap I)}{y{:}\omega \to \rho \vdash_\perp y\perp : \rho}\,(\to E)}{\vdash_\perp \lambda y.y\perp : (\omega \to \rho) \to \rho}\,(\to I)$$

Notice that x does not appear in $\lambda y.y\perp$, so the term in the derivation does not change. This last derivation is in normal form, as is the term $\lambda y.y\perp$.

One might think that, since in $\vdash_\perp$, only $\perp$ can be typed with ω, all typeable terms would be strongly normalisable. This is not the case, as argued in Example A.2, which can be found in the appendix.

Using Theorem 3.6, as for the BCD-system (see [25]) and the system of [2], the relation between types assignable to a lambda term and those assignable to its approximants can be formulated as follows:

Theorem 5.4 (Approximation) $B \vdash M{:}\sigma \iff \exists A \in \mathcal{A}(M)\ [B \vdash A{:}\sigma]$.

Proof: $\Rightarrow$) If $\mathrm{D} :: B \vdash M{:}\sigma$, then, by Theorem 3.6, $SN(\mathrm{D})$. Let $\mathrm{D}' :: B \vdash N{:}\sigma$ be a normal form of D with respect to $\to_{\mathcal{D}}$, then by Lemma 2.5, $M \twoheadrightarrow_\beta N$ and, by Lemma 5.2:(*ii*), there exists $P \sqsubseteq N$ such that $\mathrm{D}' :: B \vdash_\perp P{:}\sigma$. So, in particular, P contains no redexes (no typed redexes since D' is in normal form, and none untyped since only $\perp$ can be typed with ω), so $P \in \mathcal{N}$, and therefore $P \in \mathcal{A}(M)$.

$\Leftarrow$) Since $A \in \mathcal{A}(M)$, there is an M' such that $M' =_\beta M$ and $A \sqsubseteq M'$. Then, by Lemma 4.3, $B \vdash M'{:}\sigma$, and, by Theorem 1.8, also $B \vdash M{:}\sigma$.

In [2], this result was obtained separately, using a computability predicate.

Using the previous theorem, the following becomes easy.

Theorem 5.5 (Head-normalisation [1]) *There exists B,σ such that $B \vdash M{:}\sigma$ and $\sigma \in \mathcal{T}_S$, if and only if M has a head normal form.*

Proof: $\Rightarrow$) If $B \vdash M{:}\sigma$, then, by Theorem 5.4, there exists an $A \in \mathcal{A}(M)$ such that $B \vdash A{:}\sigma$. Then, by Definition 4.2, there exists $M' =_\beta M$ such that $A \sqsubseteq M'$. Since $\sigma \in \mathcal{T}_S$, $A \not\equiv \perp$, so A is either $x, \lambda x.A'$ or $xA_1 \ldots A_n$. Since M' matches A ($A \sqsubseteq M'$), M' is either $x, \lambda x.M_1$ or $xM_1 \ldots M_n$; so M' is in head-normal from. Then M has a head-normal form.

$\Leftarrow$) If M has a head-normal form, then there exists $M' =_\beta M$ such that M' is either $\lambda x.M_1$ with M_1 in head-normal form, or $xM_1 \ldots M_n$, with $n \geq 0$ and each $M_i \in \Lambda$.

a) $M' \equiv \lambda x.M_1$. Since M_1 is in head-normal form, by induction there are B', σ' such that $B' \vdash M_1{:}\sigma'$. If $x{:}\tau \in B'$, take $B = B' \backslash x$, and $\sigma = \tau \to \sigma'$, otherwise $B = B'$ and $\sigma = \omega \to \sigma'$.

b) $M' \equiv xM_1 \ldots M_n$. Take $B = x{:}\omega \to \ldots \to \omega \to \varphi$ and $\sigma = \varphi$.

Notice that, in all cases, $B \vdash M'{:}\sigma$, and $\sigma \in \mathcal{T}_S$. Then, by Theorem 1.8, $B \vdash M{:}\sigma$.

6 ω-free type assignment

In this section we revisit the strong normalisation proof, for which we first define a notion of derivability obtained from $\vdash$ by removing the type constant ω.

Definition 6.1 (ω-free types) *i*) The set of *ω-free strict types* is inductively defined by:

$$\sigma \ ::= \ \varphi \mid (\cap_{\underline{n}} \sigma_i \to \sigma), \ \ (n \geq 1)$$

The set $\mathcal{T}_{\not\omega}$ of *ω-free intersection types* is defined by:

$$\{\cap_{\underline{n}} \sigma_i \mid n \geq 1 \ \& \ \forall i \in \underline{n}\ [\sigma_i \text{ is an } \omega\text{-free strict type}]\}$$

ii) The relation $\leq$ is defined in ω-free types as the least pre-order on $\mathcal{T}_{\not\omega}$ such that:

$$\begin{array}{rcll} \cap_{\underline{n}} \sigma_i & \leq & \sigma_i, & \text{for all } i \in \underline{n} \\ \tau \leq \sigma_i, \ \text{for all } i \in \underline{n} & \Rightarrow & \tau \leq \cap_{\underline{n}} \sigma_i & (n \geq 1) \end{array}$$

This relation is extended onto bases as in Definition 1.3.

iii) The equivalence relation $\sim$ on types is defined by: $\sigma \sim \tau \iff \sigma \leq \tau \leq \sigma$, and we will work with types modulo $\sim$.

Definition 6.2 (ω-free type assignment) *i*) *ω-free intersection type assignment* and *ω-free intersection derivations* are defined by the following natural deduction system (where σ in

rules $(\to E)$ and $(\to I)$ is in $\mathcal{T}$):

$$(Ax):\ \frac{}{B,x{:}\cap_{\underline{n}}\sigma_i \vdash_{\omega} x:\sigma_i}\ (n\geq 1, i\in\underline{n}) \qquad (\to E):\ \frac{B\vdash_{\omega} M:\sigma\to\tau \quad B\vdash_{\omega} N:\sigma}{B\vdash_{\omega} MN:\tau}$$

$$(\cap I):\ \frac{B\vdash_{\omega} M:\sigma_1 \quad \ldots \quad B\vdash_{\omega} M:\sigma_n}{B\vdash_{\omega} M:\cap_{\underline{n}}\sigma_i}\ (n\geq 1) \qquad (\to I):\ \frac{B,x{:}\sigma\vdash_{\omega} M:\tau}{B\vdash_{\omega} \lambda x.M:\sigma\to\tau}$$

ii) We will write $B\vdash_{\omega} M:\sigma$ if this statement is derivable using a ω-free intersection derivation, and write $\mathrm{D} :: B\vdash_{\omega} M:\sigma$ to specify that this result was obtained through the derivation D.

The following properties hold:

Lemma 6.3 *i)* $B\vdash_{\omega} M:\sigma \Rightarrow \{x{:}\sigma\in B \mid x\in fv(M)\}\vdash_{\omega} M:\sigma$.
ii) $B\vdash_{\omega} M:\sigma$ & $B'\leq B \Rightarrow B'\vdash_{\omega} M:\sigma$.
iii) If $\mathrm{D} :: B\vdash_{\omega} M:\sigma$, *then* $\mathrm{D} :: B\vdash M:\sigma$.

Proof: Easy.

To prepare the characterisation of terms by their assignable types, first is proved that a term in $\lambda\bot$-normal form is typeable without ω, if and only if it does not contain $\bot$. This forms the basis for the result that all normalisable terms are typeable without ω.

Lemma 6.4 ([2]) *i) If* $B\vdash_{\omega} A:\sigma$, *and* B,σ *are* ω*-free, then* A *is* $\bot$*-free.*
ii) If A *is* $\bot$*-free, then there are* B, *and* σ, *such that* $B\vdash_{\omega} A:\sigma$.

Proof: By induction on the structure of approximate normal forms.

i) As before, only the part that σ is strict is shown.

$(A\equiv x)$: Immediate.

$(A\equiv \bot)$: Impossible, by inspecting the rules of $\vdash_{\omega}$.

$(A\equiv \lambda x.A')$: By $(\to I)$ there are α,β such that $\sigma=\alpha\to\beta$, and $B,x{:}\alpha\vdash_{\omega} A':\beta$. Of course also $B,x{:}\alpha$, and β are ω-free, so by induction, A' is $\bot$-free, so also $\lambda x.A'$ is $\bot$-free.

$(A\equiv xA_1\ldots A_n)$: Then by $(\to E)$ and (Ax) there are m,σ_i $(\forall i\in\underline{n})$, τ_j $(\forall j\in\underline{m})$, such that for every $i\in\underline{n}$, $B\vdash_{\omega} A_i:\sigma_i$, and $x{:}\cap_{\underline{m}}\tau_i\in B$, and, for some $j\in m$, $\sigma_1\to\ldots\to\sigma_n\to\sigma=\tau_j$. Since each σ_i occurs in τ_j, which occurs in B, all are ω-free, so by induction each A_i is $\bot$-free. Then also $xA_1\ldots A_n$ is $\bot$-free.

ii) $(A\equiv x)$: $x{:}\varphi\vdash_{\omega} x:\varphi$.

$(A\equiv \lambda x.A')$: By induction there are B,τ such that $B\vdash_{\omega} A':\tau$. If x does not occur in B, take a $\sigma\in\mathcal{T}_{\omega}$; otherwise, there exist $x{:}\sigma\in B$, and σ is ω-free. In either case, $B\backslash x\vdash_{\omega} \lambda x.A':\sigma\to\tau$.

$(A\equiv xA_1\ldots A_n)$: By induction there are σ_i $(\forall i\in\underline{n})$ such that $B\vdash_{\omega} A_i:\sigma_i$ for every $i\in\underline{n}$. Then $\bigcap\{B,\{x{:}\sigma_1\to\ldots\to\sigma_n\to\varphi\}\}\vdash_{\omega} xA_1\ldots A_n:\varphi$.

Now, as also shown in [1], it is possible to characterise normalisable terms.

Theorem 6.5 ([1, 2]) *There exists* B,σ *such that* $B\vdash M:\sigma$ *and* B,σ *are* ω*-free, if and only if* M *has a normal form.*

Proof: $\Rightarrow$) If $B\vdash M:\sigma$, by Theorem 5.4, there exists $A\in\mathcal{A}(M)$ such that $B\vdash A:\sigma$. Since B,σ

are ω-free, by Lemma 6.4:(*i*), this A is $\bot$-free. By Definition 4.1 there exists $M' =_\beta M$ such that $A \sqsubseteq M'$. Since A contains no $\bot$, $A \equiv M'$, so M' is a normal form, so, especially, M has a normal form.

$\Leftarrow$) If M' is the normal form of M, then it is a $\bot$-free approximate normal form. Then by Lemma 6.4:(*ii*) there are B,σ such that $B \vdash_{\omega} M' : \sigma$. Then, by Theorem 1.8, $B \vdash M : \sigma$, and B,σ are ω-free.

(Notice that, in the second part, in general, the property that ω is not used at all, is lost.)

The following lemma shows a subject expansion result for the ω-free system.

Lemma 6.6 If $B \vdash_{\omega} M[N/x] : \sigma$ and $B \vdash_{\omega} N : \rho$, then $B \vdash_{\omega} (\lambda x.M)N : \sigma$.

Proof: We focus on the case that σ is strict; the case that σ is an intersection is just a generalisation. We can assume that x does not occur in B, and proceed by induction on the structure of M.

$(M \equiv x)$: Then $M[N/x] \equiv N$. From $B \vdash_{\omega} N : \sigma$ we obtain $B \vdash_{\omega} (\lambda x.x)N : \sigma$.

$(M \equiv y \neq x)$: If $B \vdash_{\omega} y : \sigma$, then, by Lemma 6.3:(*i*) and $(\to I)$, $B \vdash_{\omega} \lambda x.y : \rho \to \sigma$ so also $B \vdash_{\omega} (\lambda x.y)N : \sigma$.

$(M \equiv \lambda y.M')$: Then $(\lambda y.M')[N/x] \equiv \lambda y.(M'[N/x])$, and $\sigma = \alpha \to \beta$. Notice that, by α-conversion, we can assume that $y \notin fv(N)$. Then:

$$\begin{array}{ll}
B \vdash_{\omega} \lambda y.(M'[N/x]) : \alpha \to \beta \ \&\ B \vdash_{\omega} N : \rho & \Rightarrow (\to I) \\
B, y{:}\alpha \vdash_{\omega} M'[N/x] : \beta \ \&\ B \vdash_{\omega} N : \rho & \Rightarrow (IH) \\
B, y{:}\alpha \vdash_{\omega} (\lambda x.M')N : \beta & \Rightarrow (\beta \text{ is strict}) \ \&\ (\to E) \\
\exists \gamma\ [B, y{:}\alpha \vdash_{\omega} \lambda x.M' : \gamma \to \beta \ \&\ B, y{:}\alpha \vdash_{\omega} N : \gamma] & \Rightarrow (\to I) \ \&\ (y \notin fv(N)) \\
\exists \gamma\ [B, y{:}\alpha, x{:}\gamma \vdash_{\omega} M' : \beta \ \&\ B \vdash_{\omega} N : \gamma] & \Rightarrow (\to I) \\
\exists \gamma\ [B \vdash_{\omega} \lambda xy.M' : \gamma \to \alpha \to \beta \ \&\ B \vdash_{\omega} N : \gamma] & \Rightarrow (\to E) \\
B \vdash_{\omega} (\lambda xy.M')N : \alpha \to \beta &
\end{array}$$

$(M \equiv M_1 M_2)$: Then $(M_1 M_2)[N/x] \equiv M_1[N/x]M_2[N/x]$.

$$\begin{array}{ll}
B \vdash_{\omega} M_1[N/x]M_2[N/x] : \sigma \ \&\ B \vdash_{\omega} N : \rho & \Rightarrow (\to E) \\
\exists \tau\ [B \vdash_{\omega} M_1[N/x] : \tau \to \sigma \ \&\ B \vdash_{\omega} M_2[N/x] : \tau] \ \&\ B \vdash_{\omega} N : \rho & \Rightarrow (IH) \\
\exists \tau\ [B \vdash_{\omega} (\lambda x.M_1)N : \tau \to \sigma \ \&\ B \vdash_{\omega} (\lambda x.M_2)N : \tau] & \Rightarrow (\to E), (\to I) \ \&\ (\cap I) \\
\exists \rho_1, n, \rho_2^i, \tau, \tau_i\ (\forall i \in \underline{n})\ [B, x{:}\rho_1 \vdash_{\omega} M_1 : \tau \to \sigma \ \&\ B \vdash_{\omega} N : \rho_1 & \\
\qquad \& \ \tau = \cap_{\underline{n}} \tau_i \ \&\ \forall i \in \underline{n}\, [B, x{:}\rho_2^i \vdash_{\omega} M_2 : \tau_i \ \&\ B \vdash_{\omega} N : \rho_2^i]] & \Rightarrow (\cap I) \ \&\ (6.3{:}(ii)) \\
\exists \rho_1, \rho_2, \tau [B, x{:}\rho_1 \cap \rho_2 \vdash_{\omega} M_1 : \tau \to \sigma \ \&\ B \vdash_{\omega} N : \rho_1 \cap \rho_2 & \\
\qquad \& \ B, x{:}\rho_1 \cap \rho_2 \vdash_{\omega} M_2 : \tau] & \Rightarrow (\to E) \\
\exists \rho' [B, x{:}\rho' \vdash_{\omega} M_1 M_2 : \sigma \ \&\ B \vdash_{\omega} N : \rho'] & \Rightarrow (\to I) \\
\exists \rho'\ [B \vdash_{\omega} \lambda x.(M_1 M_2) : \rho' \to \sigma \ \&\ B \vdash_{\omega} N : \rho'] & \Rightarrow (\to E) \\
B \vdash_{\omega} (\lambda x.(M_1 M_2))N : \sigma &
\end{array}$$

This result extends by induction (easily) to all contexts:

$$\text{if } B \vdash_{\omega} \mathrm{C}[M[N/x]] : \sigma \text{ and } B \vdash_{\omega} N : \rho, \text{ then } B \vdash_{\omega} \mathrm{C}[(\lambda x.M)N] : \sigma.$$

Notice that the condition $B \vdash_{\omega} N : \rho$ in the formulation of the lemma is essential. As counter example, take the two lambda terms $\lambda yz.(\lambda b.z)(yz)$ and $\lambda yz.z$. Notice that the first strongly reduces to the latter. We know that

$$z{:}\sigma, y{:}\tau \vdash_{\omega} z : \sigma$$

but it is impossible to give a derivation for $(\lambda b.z)(yz):\sigma$ from the same basis without using ω. This is caused by the fact that we can only type $(\lambda b.z)(yz)$ in the system without ω from a basis in which the predicate for y is an arrow type. We can, for example, derive

$$B, z{:}\sigma, y{:}\sigma\to\tau \vdash_{\not\omega} (\lambda b.z)(yz):\sigma.$$

We can therefore only state that we can derive

$$B \vdash_{\not\omega} \lambda yz.(\lambda b.z)(yz):(\sigma\to\tau)\to\rho \text{ and } B \vdash_{\not\omega} \lambda yz.z:\tau\to\rho$$

but that we are not able to give a derivation without ω for the statement

$$\lambda yz.(\lambda b.z)(yz):\tau\to\rho.$$

So the type assignment without ω is not closed for β-equality, but of course this is not imperative. We only want to be able to derive *a* type for each strongly normalisable term, no matter what basis or type is used.

Lemma 6.6 is also essentially the proof for the statement that each strongly normalisable term can be typed in the system $\vdash_{\not\omega}$, a property that we will now show.

Theorem 6.9 shows that the set of strongly normalisable terms is exactly the set of terms typeable in the intersection system without using the type constant ω. The same result was stated in [1] for the BCD-system, but the proof there was not complete. The proof of the crucial lemma as presented below (Lemma 6.8) and part ($\Leftarrow$) of the proof of Theorem 6.9 are essentially due to Betti Venneri, of the University of Florence, Italy, and goes by induction on the left-most outer-most reduction path.

First we will introduce the notion of left-most, outer-most reduction.

Definition 6.7 An occurrence of a redex $\mathrm{R} = (\lambda x.P)Q$ in a term M is called the *left-most outer-most redex of* M ($lor(M)$), if:

i) There is no redex R' in M such that $\mathrm{R}' = \mathrm{C}[\mathrm{R}]$ (*outer-most*).

ii) There is no redex R' in M such that $M = \mathrm{C}_0[\mathrm{C}_1[\mathrm{R}']\mathrm{C}_2[\mathrm{R}]]$ (*left-most*).

$M \to_{lor} N$ is used to indicate that M reduces to N by contracting $lor(M)$.

The following lemma formulates a subject expansion result for $\vdash_{\not\omega}$ with respect to left-most outer-most reduction.

Lemma 6.8 Let $M \to_{lor} N$, $lor(M) = (\lambda x.P)Q$, $B \vdash_{\not\omega} N:\sigma$, and $B' \vdash_{\not\omega} Q:\tau$, then there exists B_0, ρ such that $B_0 \vdash_{\not\omega} M:\rho$.

Proof: By induction on the structure of types, of which only the part $\sigma \in \mathcal{T}_S$ will be shown, by induction on the structure of terms; note that $M \equiv \lambda x_1 \ldots x_k.VP_1 \ldots P_n$ ($k, n \geq 0$), where either

i) V is a redex, so $V \equiv (\lambda y.P)Q$, and $N \equiv \lambda x_1 \ldots x_k.(P[Q/y])P_1 \ldots P_n$, (notice that $lor(M) = V$) or

ii) $V \equiv y$, so there is an $j \in \underline{n}$ such that $lor(M) = lor(P_j)$, and $P_j \to_{lor} P'$, and $N \equiv \lambda x_1 \ldots x_k.yP_1 \ldots P' \ldots P_n$.

In either case, we have, by Lemma 6.3, that there are α_j ($\forall j \in \underline{k}$), γ_i ($\forall i \in \underline{n}$), and β such that (where $B_1 = B, x_1{:}\alpha_1, \ldots, x_k{:}\alpha_k$, and V' is either $P[Q/y]$ or y):

$$\sigma = \alpha_1\to\ldots\to\alpha_k\to\beta,\ B_1 \vdash_{\not\omega} V':\gamma_1\to\ldots\to\gamma_n\to\beta, \text{ and } B_1 \vdash_{\not\omega} P_i:\gamma_i\ (\forall i \in \underline{n}).$$

We distinguish two cases:

i) $V' \equiv P[Q/y]$. Let $B_2 = B'$, then $\bigcap\{B_1, B_2\} \vdash_{\not\omega} (\lambda y.P)Q:\gamma_1\to\ldots\to\gamma_n\to\beta$, by Lemma 6.6.

ii) $V' \equiv y$. Then, by induction, there are B', ρ such that $B' \vdash_{\omega} P_j : \rho$.
Take $\mu = \gamma_1 \to \ldots \rho \ldots \to \gamma_n \to \beta, B_2 = B', y{:}\mu$, then $\bigcap\{B_1, B_2\} \vdash_{\omega} y : \mu$.

In either case, $\bigcap\{B_1, B_2\} \vdash_{\omega} VP_1 \ldots P_n : \beta$. Let, for all $i \in \underline{k}$, $x_i{:}\beta_i \in \bigcap\{B_1, B_2\}$, then $\bigcap\{B_1, B_2\} \backslash x_1, \ldots, x_k \vdash_{\omega} \lambda x_1 \ldots x_k.VP_1 \ldots P_n : \beta_1 \to \ldots \to \beta_k \to \beta$.

We can now show that all strongly normalisable terms are exactly those typeable in $\vdash_{\omega}$.

Theorem 6.9 $\exists B, \sigma\ [B \vdash_{\omega} M : \sigma] \iff$ *M is strongly normalisable with respect to* $\to_{\beta}$.

Proof: $\Rightarrow$) If D :: $B \vdash_{\omega} M : \sigma$, then by Lemma 6.3:(*iii*), also D :: $B \vdash M : \sigma$. Then, by Theorem 3.6, D is strongly normalisable with respect to $\to_{\mathcal{D}}$. Since D contains no ω, all redexes in M correspond to redexes in D, a property that is preserved by derivation reduction (it does not introduce ω). So also M is strongly normalisable with respect to $\to_{\beta}$.

$\Leftarrow$) With induction on the maximum of the lengths of *lor*-reduction sequences for a strongly normalisable term to its normal form (denoted by $\#(M)$).

a) If $\#(M) = 0$, then M is in normal form, and by Lemma 6.4:(*ii*), there exist B and $\sigma \in \mathcal{T}$ such that $B \vdash_{\omega} M : \sigma$.

b) If $\#(M) \geq 1$, so M contains a redex, then let $M \to_{lor} N$ by contracting the redex $(\lambda x.P)Q$. Then $\#(N) < \#(M)$, and $\#(Q) < \#(M)$ (since Q is a proper sub-term of a redex in M), so by induction $B \vdash_{\omega} M : \sigma$ and $B' \vdash_{\omega} Q : \tau$, for some B, B', σ, and τ. Then, by Lemma 6.8, there exist B_1, ρ such that $B_1 \vdash_{\omega} M : \rho$.

Conclusions and future work

We have shown that cut-elimination is strongly normalising also for an intersection type assignment systems that contains ω, and that all standard characterisations of normalisation are consequences of this result. A future extension of this result could be to consider a type-inclusion relation that is contra-variant over the arrow, so to consider a system that is closed for η-reduction.

Acknowledgements

I am greatly indebted to Maribel Fernández for the close cooperation and never failing persistence that was needed for the development of the technique I applied here, and would like to thank Betty Venneri for allowing me to publish her proof.

References

[1] S. van Bakel. Complete restrictions of the Intersection Type Discipline. *Theoretical Computer Science*, 102(1):135–163, 1992.

[2] S. van Bakel. Intersection Type Assignment Systems. *Theoretical Computer Science*, 151(2):385–435, 1995.

[3] S. van Bakel. Strongly Normalising Cut-Elimination with Strict Intersection Types (Extended Abstract). In *Proceedings of Intersection Types and Related Systems (ITRS '02)*, Electronic Notes in Theoretical Computer Science, 70, 2002. http://www.elsevier.nl/locate/entcs/volume70.html

[4] S. van Bakel, F. Barbanera, M. Dezani-Ciacaglini, and F.-J. de Vries. Intersection Types for λ-Trees. *Theoretical Computer Science*, 272:3–40, 2002.

[5] S. van Bakel and M. Dezani-Ciacaglini. Characterising Strong Normalisation for Explicit Substitutions. In *Proceedings of Latin American Theoretical Informatics (LATIN'02)*, 2002. In *Proceedings of*

Latin American Theoretical Informatics (LATIN'02), Cancùn, Mexico, volume 2286 of *Lecture Notes in Computer Science*, pages 356-370. Springer-Verlag, 2002.

[6] S. van Bakel and M. Fernández. Approximation and Normalization Results for Typeable Term Rewriting Systems. In Gilles Dowek, Jan Heering, Karl Meinke, and Bernhard Möller, editors, *Proceedings of HOA '95. Second International Workshop on Higher Order Algebra, Logic and Term Rewriting*, Paderborn, Germany. *Selected Papers*, volume 1074 of *Lecture Notes in Computer Science*, pages 17–36. Springer-Verlag, 1996.

[7] S. van Bakel and M. Fernández. Normalization Results for Typeable Rewrite Systems. *Information and Computation*, 133(2):73–116, 1997.

[8] S. van Bakel and M. Fernández. Normalisation, Approximation, and Semantics for Combinator Systems. *Theoretical Computer Science*, 290:975-1019, 2003.

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

[10] H. Barendregt, M. Coppo, and M. Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *Journal of Symbolic Logic*, 48(4):931–940, 1983.

[11] M. Coppo and M. Dezani-Ciancaglini. An Extension of the Basic Functionality Theory for the λ-Calculus. *Notre Dame Journal of Formal Logic*, 21(4):685–693, 1980.

[12] M. Coppo, M. Dezani-Ciancaglini, and M. Zacchi. Type Theories, Normal Forms and D_∞-Lambda-Models. *Information and Computation*, 72(2):85–116, 1987.

[13] H.B. Curry. Functionality in Combinatory Logic. In *Proc. Nat. Acad. Sci. U.S.A.*, volume 20, pages 584–590, 1934.

[14] H.B. Curry and R. Feys. *Combinatory Logic*, volume 1. North-Holland, Amsterdam, 1958.

[15] M. Dezani-Ciancaglini and I. Margaria. A characterisation of F-complete type assignments. *Theoretical Computer Science*, 45:121–157, 1986.

[16] M. Dezani-Ciancaglini, F. Honsell, and Y. Motohama. Approximation Theorems for Intersection Type Systems. *Logic and Computation*, 11(3): 395–417, 2001.

[17] M. Dezani-Ciancaglini, R. K. Meyer, and Y. Motohama. The semantics of entailment omega. *Notre Dame Journal of Formal Logic*. To appear.

[18] 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-Verlag, 2001.

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

[20] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1989.

[21] J-L. Krivine. *Lambda-Calcul – Types et Modèles*. Etudes et Recherches en Informatique. Masson, Paris, 1990.

[22] G. Pottinger. A type assignment for the strongly normalizable λ-terms. In J. R. Hindley and J. P. Seldin, editors, *To H. B. Curry, Essays in combinatory logic, lambda-calculus and formalism*, pages 561–577. Academic press, New York, 1980.

[23] C. Retoré. A note on intersection types. INRIA Rapport de recherche 2431, INRIA, France, 1994.

[24] S. Ronchi Della Rocca. Principal type scheme and unification for intersection type discipline. *Theoretical Computer Science*, 59:181–209, 1988.

[25] S. Ronchi Della Rocca and B. Venneri. Principal type schemes for an extended type theory. *Theoretical Computer Science*, 28:151–169, 1984.

[26] W.W. Tait. Intensional interpretation of functionals of finite type I. *Journal of Symbolic Logic*, 32(2):198–223, 1967.

[27] C.P. Wadsworth. The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus. *SIAM J. Comput.*, 5:488–521, 1976.

Appendix A Extended examples

We give an example of a non-strongly normalising term for which it is possible to find a derivation such that no redex is covered with ω; moreover, for all the β-reducts of this term, the

same property holds. We will show that this first derivation has a normal from, and construct the reduction sequences. The derivation we will construct is similar to the one of Example 2.4, but differs in the type derived for $\Theta\Theta$: $(\rho\to\rho)\cap(\omega\to\rho) \to \rho$ rather than $(\omega\to\rho)\to\rho$.

Example A.1 Take $\Theta = \lambda xy.y(xxy)$, then $\Theta\Theta$ is typeable in $\vdash$, without covering a redex by ω. Let $\tau = ((\alpha\to\beta\to\gamma)\cap\alpha)\to((\gamma\to\delta)\cap\beta)\to\delta$, and take the derivations $D_1 :: \vdash \Theta:\tau$ and $D_2 :: \vdash \Theta:\tau\to(\omega\to\rho)\to\rho$ of Example 2.4. From these two, by applying $(\cap I)$, we get $D_3 :: \vdash \Theta:(\tau\to(\omega\to\rho)\to\rho)\cap\tau$:

$$\dfrac{\boxed{D_2} \quad \boxed{D_1}}{\vdash \lambda xy.y(xxy):\tau\to(\omega\to\rho)\to\rho \qquad \vdash \lambda xy.y(xxy):\tau \qquad \vdash \Theta:(\tau\to(\omega\to\rho)\to\rho)\cap\tau}\,(\cap I)$$

Also, we can construct D_4 (taking $B' = x{:}(\tau\to(\omega\to\rho)\to\rho)\cap\tau, y{:}(\rho\to\rho)\cap(\omega\to\rho)$):

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B'\vdash y:\rho\to\rho}\,(Ax) \quad \dfrac{\dfrac{\dfrac{}{B'\vdash x:\tau\to(\omega\to\rho)\to\rho}\,(Ax) \quad \dfrac{}{B'\vdash x:\tau}\,(Ax)}{B'\vdash xx:(\omega\to\rho)\to\rho}\,(\to E) \quad \dfrac{}{B'\vdash y:\omega\to\rho}\,(Ax)}{B'\vdash xxy:\rho}\,(\to E)}{B'\vdash y(xxy):\rho}\,(\to E)}{B'\backslash y\vdash \lambda y.y(xxy):(\rho\to\rho)\cap(\omega\to\rho)\to\rho}\,(\to I)}{\vdash \Theta:(\tau\to(\omega\to\rho)\to\rho)\cap\tau\to(\rho\to\rho)\cap(\omega\to\rho)\to\rho}\,(\to I)$$

Then, by applying $(\to E)$, we get $D_5 :: \vdash \Theta\Theta:(\rho\to\rho)\cap(\omega\to\rho)\to\rho$:

$$\dfrac{\boxed{D_4} \quad \boxed{D_3}}{\vdash \Theta:(\tau\to(\omega\to\rho)\to\rho)\cap\tau\to(\rho\to\rho)\cap(\omega\to\rho)\to\rho \qquad \vdash \Theta:(\tau\to(\omega\to\rho)\to\rho)\cap\tau \qquad \vdash \Theta\Theta:(\rho\to\rho)\cap(\omega\to\rho)\to\rho}\,(\to E)$$

Let $D_6 :: v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash v:(\rho\to\rho)\cap(\omega\to\rho)$ be:

$$\dfrac{\dfrac{}{v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash v:\rho\to\rho}\,(Ax) \quad \dfrac{}{v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash v:\omega\to\rho}\,(Ax)}{v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash v:(\rho\to\rho)\cap(\omega\to\rho)}\,(\cap I)$$

then, adding a statement for v to the derivation D_5, we get also D_7:

$$\dfrac{\boxed{D_5} \quad \boxed{D_6}}{v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash \Theta\Theta:(\rho\to\rho)\cap(\omega\to\rho)\to\rho \qquad v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash v:(\rho\to\rho)\cap(\omega\to\rho) \qquad v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash \Theta\Theta v:\rho}\,(\to E)$$

Notice that $\Theta\Theta v$ is not strongly normalisable, since

$$\Theta\Theta v \twoheadrightarrow_\beta v(\Theta\Theta v) \twoheadrightarrow_\beta v(v(\Theta\Theta v)) \twoheadrightarrow_\beta \ldots.$$

Moreover, all these reducts are typeable in $\vdash$ such that no redex is typed with ω: since we can derive both $v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash v:\rho\to\rho$, and $v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash \Theta\Theta v:\rho$, we get $v{:}(\rho\to\rho)\cap(\omega\to\rho)\vdash v(\Theta\Theta v):\rho$ by rule $(\to E)$, and so on.

We will now show that, in $\vdash_\perp$, typeable terms need not be strongly normalisable.

Example A.2 As argued in Example 5.3, D_1' and D_2', the $\vdash_{\not\omega}$-variants of the derivations D_1 and D_2 of Example A.1, now consider different terms, namely Θ and $\lambda xy.y\perp$. Notice that applying rule $(\cap I)$ of $\vdash_\perp$ requires the terms to be compatible, which these are, and then types the join,

which is Θ. So we get $D'_3 = \langle D'_1, D'_2, (\cap I)\rangle :: \vdash_\perp \Theta : (\tau\to(\omega\to\rho)\to\rho)\cap\tau$.

Notice, moreover, that rule $(\cap I)$ is not used to assign ω in the derivations D_4, D_5, D_6, and D_7, so the $\vdash_\perp$ variants of these derivations would be identical, and we would obtain $D'_7 :: v{:}(\rho\to\rho)\cap(\omega\to\rho) \vdash_\perp \Theta\Theta v : \rho$. The term $\Theta\Theta v$ is not strongly normalisable, as argued above.

The next example show all the reduction sequences starting form the final derivation given in Example A.1.

Example A.3 Take Θ, $D_1,\ldots,D_7$ as in Example A.1, then, using $B = v{:}(\rho\to\rho)\cap(\omega\to\rho)$ (to save space, we use α for $(\tau\to(\omega\to\rho)\to\rho)\cap\tau \to (\rho\to\rho)\cap(\omega\to\rho) \to \rho)$, then the last derivation of the previous example, D_7, looks like:

$$\dfrac{\dfrac{\boxed{D_4}\quad \dfrac{\boxed{D_2}\quad \boxed{D_1}}{B\vdash\Theta:\tau\to(\omega\to\rho)\to\rho \quad B\vdash\Theta:\tau}(\cap I)}{B\vdash\Theta:\alpha \quad B\vdash\Theta:(\tau\to(\omega\to\rho)\to\rho)\cap\tau}(\to E)}{B\vdash\Theta\Theta:(\rho\to\rho)\cap(\omega\to\rho)\to\rho}(\to E) \quad \boxed{D_6}\ B\vdash v:(\rho\to\rho)\cap(\omega\to\rho)}{B\vdash\Theta\Theta v:\rho}(\to E)$$

This derivation has only *one* redex $\langle D_4, \langle D_2, D_1, \cap I\rangle, \to E\rangle$; remark that D_4 finishes with an application of rule $(\to I)$ (where $B' = B, x{:}(\tau\to(\omega\to\rho)\to\rho)\cap\tau, y{:}(\rho\to\rho)\cap(\omega\to\rho)$):

$$D_4: \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{B'\vdash y:\rho\to\rho}(Ax) \quad \dfrac{\dfrac{}{B'\vdash x:\tau\to(\omega\to\rho)\to\rho}(Ax) \quad \dfrac{}{B'\vdash x:\tau}(Ax)}{B'\vdash xx:(\omega\to\rho)\to\rho}(\to E) \quad \dfrac{}{B'\vdash y:\omega\to\rho}(Ax)}{B'\vdash xxy:\rho}(\to E)}{B'\vdash y(xxy):\rho}(\to E)}{B'\backslash y\vdash \lambda y.y(xxy):(\rho\to\rho)\cap(\omega\to\rho)\to\rho}(\to I)}{B\vdash\lambda xy.y(xxy):\alpha}(\to I) \quad \dfrac{\boxed{D_2}\quad \boxed{D_1}}{B\vdash\Theta:\tau\to(\omega\to\rho)\to\rho \quad B\vdash\Theta:\tau}(\cap I)}{B\vdash\Theta\Theta:(\rho\to\rho)\cap(\omega\to\rho)\to\rho}(\to E)$$

Contracting this redex makes D_7 reduce to D_8 (where $B'' = B, y{:}(\rho\to\rho)\cap(\omega\to\rho)$):

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B''\vdash y:\rho\to\rho}(Ax) \quad \dfrac{\dfrac{\boxed{D_2}\quad \boxed{D_1}}{B''\vdash\Theta:\tau\to(\omega\to\rho)\to\rho \quad B''\vdash\Theta:\tau}(\to E)}{B''\vdash\Theta\Theta:(\omega\to\rho)\to\rho} \quad \dfrac{}{B''\vdash y:\omega\to\rho}(Ax)}{B''\vdash\Theta\Theta y:\rho}(\to E)}{B''\vdash y(\Theta\Theta y):\rho}(\to E)}{B\vdash\lambda y.y(\Theta\Theta y):(\rho\to\rho)\cap(\omega\to\rho)\to\rho}(\to I) \quad \boxed{D_6}\ B\vdash v:(\rho\to\rho)\cap(\omega\to\rho)}{B\vdash(\lambda y.y(\Theta\Theta y))v:\rho}(\to E)$$

Now D_8 has *two* redexes (notice that D_2 finishes with rule $(\to I)$); contracting the outer-most distributes (the two sub-derivations of) D_6 and creates:

$$\mathrm{D}_9 : \dfrac{\dfrac{}{B \vdash v : \rho \to \rho}\,(Ax) \qquad \dfrac{\dfrac{\boxed{\mathrm{D}_2}}{B \vdash \Theta : \tau \to (\omega \to \rho) \to \rho} \qquad \dfrac{\boxed{\mathrm{D}_1}}{B \vdash \Theta : \tau}}{B \vdash \Theta\Theta : (\omega \to \rho) \to \rho}\,(\to E) \qquad \dfrac{}{B \vdash v : \omega \to \rho}\,(Ax)}{B \vdash \Theta\Theta v : \rho}\,(\to E)}{B \vdash v(\Theta\Theta v) : \rho}\,(\to E)$$

As illustrated by Example 2.4, contracting the remaining redex of D_9 creates:

$$\mathrm{D}_{10} : \dfrac{\dfrac{}{B \vdash v : \rho \to \rho}\,(Ax) \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{B, z{:}\omega \to \rho \vdash z : \omega \to \rho}\,(Ax) \qquad \dfrac{}{B, z{:}\omega \to \rho \vdash \Theta\Theta z : \omega}\,(\cap I)}{B, z{:}\omega \to \rho \vdash z(\Theta\Theta z) : \rho}\,(\to E)}{B \vdash \lambda z.z(\Theta\Theta z) : (\omega \to \rho) \to \rho}\,(\to I) \qquad \dfrac{}{B \vdash v : \omega \to \rho}\,(Ax)}{B \vdash (\lambda z.z(\Theta\Theta z))v : \rho}\,(\to E)}{B \vdash v((\lambda z.z(\Theta\Theta z))v) : \rho}\,(\to E)$$

This derivation has again one redex: contracting it will generate the derivation D_{11}:

$$\mathrm{D}_{11} : \dfrac{\dfrac{}{B \vdash v : \rho \to \rho}\,(Ax) \qquad \dfrac{\dfrac{}{B \vdash v : \omega \to \rho}\,(Ax) \qquad \dfrac{}{B \vdash \Theta\Theta v : \omega}\,(\cap I)}{B \vdash v(\Theta\Theta v) : \rho}\,(\to E)}{B \vdash v(v(\Theta\Theta v)) : \rho}\,(\to E)$$

This derivation now is in normal form; again, the term $v(v(\Theta\Theta v))$ is not.

On the other hand, contracting first the inner-most redex of D_8 creates D_{12}:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B'' \vdash y : \rho \to \rho}\,(Ax) \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{B'', z{:}\omega \to \rho \vdash z : \omega \to \rho}\,(Ax) \qquad \dfrac{}{B'', z{:}\omega \to \rho \vdash \Theta\Theta z : \omega}\,(\cap I)}{B'', z{:}\omega \to \rho \vdash z(\Theta\Theta z) : \rho}\,(\to E)}{B'' \vdash \lambda z.z(\Theta\Theta z) : (\omega \to \rho) \to \rho}\,(\to I) \qquad \dfrac{}{B'' \vdash y : \omega \to \rho}\,(Ax)}{B'' \vdash (\lambda z.z(\Theta\Theta z))y : \rho}\,(\to E)}{B'' \vdash y((\lambda z.z(\Theta\Theta z))y) : \rho}\,(\to E)}{B \vdash \lambda y.y((\lambda z.z(\Theta\Theta z))y) : (\rho \to \rho) \cap (\omega \to \rho) \to \rho}\,(\to I) \qquad \dfrac{\boxed{\mathrm{D}_6}}{B \vdash v : (\rho \to \rho) \cap (\omega \to \rho)}}{B \vdash (\lambda y.y((\lambda z.z(\Theta\Theta z))y))v : \rho}\,(\to E)$$

This derivation has again two redexes. Contracting the outer-most creates D_{10} which in turn reduces to D_{11}. Alternatively, contracting the inner-most redex creates D_{13}:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B'' \vdash y : \rho \to \rho}\,(Ax) \qquad \dfrac{\dfrac{}{B'' \vdash y : \omega \to \rho}\,(Ax) \qquad \dfrac{}{B'' \vdash \Theta\Theta y : \omega}\,(\cap I)}{B'' \vdash y(\Theta\Theta y) : \rho}\,(\to E)}{B'' \vdash y(y(\Theta\Theta y)) : \rho}\,(\to E)}{B \vdash \lambda y.y(y(\Theta\Theta y)) : (\rho \to \rho) \cap (\omega \to \rho) \to \rho}\,(\to I) \qquad \dfrac{\boxed{\mathrm{D}_6}}{B \vdash v : (\rho \to \rho) \cap (\omega \to \rho)}}{B \vdash (\lambda y.y(y(\Theta\Theta y)))v : \rho}\,(\to E)$$

This derivation has only one redex, and reduces to D_{11}.