Notre Dame Journal of Formal Logic
Volume 45, Number 1, 2004

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

Steffen van Bakel

Abstract This paper defines reduction on derivations (cut-elimination) in the Strict Intersection Type Assignment System of an earlier paper and shows a strong normalization result for this reduction. Using this result, new proofs are given for the approximation theorem and the characterization of normalizability of terms using intersection types.

1 Introduction

Strong normalization 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 normalization of derivation reduction (also called cut-elimination in, for example, Barendregt et al. [2]) which mimics the normal reduction on terms to which the types are assigned. This area also has been well studied.

For intersection type assignment systems, proofs of strong normalization of derivation reduction have at best been indirect, that is, 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—Dezani-Ciancaglini et al. [9]—that deals adequately with intersection and ω but strong normalization of cut-elimination has not been shown yet for it.) This paper will use the Strict Type Assignment System of van Bakel [19] (which contains ω) and will present a proof for the property directly in the system itself.

The Intersection Type Discipline (ITD) as presented in Coppo and Dezani-Ciancaglini [3] (a more enhanced system was presented in [2]; for an overview of

Received September 12, 2002; accepted February 13, 2004; printed September 1, 2004
2000 Mathematics Subject Classification: Primary, 03B15, 03F05; Secondary, 68Q55
Keywords: intersection types, lambda calculus, termination, cut-elimination

the various existing systems, see van Bakel [20]) was introduced mainly to overcome the limitations of Curry's type assignment system (Curry [5], Curry and Feys [6]) and has been used to characterize normalization 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 nonunifiable, roles. This slight generalization 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:

$$M =_\beta N \Rightarrow (B \vdash M:\sigma \Leftrightarrow B \vdash N:\sigma).$$

The ITD is most renowned for providing proofs for the following *characterization of (head/strong) normalization* by assignable types (where ω is a type-constant and stands for the universal type, that is, all terms are typeable by ω):

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

These properties immediately show that type assignment, even in the system that does not contain ω [19], is undecidable.

As with many strong normalization results in the context of types, the strong normalization result of this paper also is obtained using the technique of Computability Predicates (Tait [18], Girard et al. [12]). 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 normalization properties (or similar results) as, for example, in Pottinger [14], Coppo et al. [4], Dezani-Ciancaglini and Margaria [8], Ronchi Della Rocca [17], Krivine [13], [19], [20], Ghilezan [11], van Bakel and Fernández [25], van Bakel et al. [22], Dougherty and Lescanne [10], and van Bakel and Dezani-Ciancaglini [23] (this list is by no means intended to be complete).

Also, as in [20], the technique proved to be sufficient to show a head-normalization as well as an approximation result, which will be shown again here (see Theorems 6.5 and 6.4, respectively). In this paper these results—the three characterization results and the approximation result—are shown to be a direct consequence of the main result in that all normal characterizations of (strong/head) normalization are consequences of the strong normalization 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 van Bakel and Fernández [24] and [26], to obtain this result in the context of Combinator Systems or Term Rewriting Systems, a more general solution was needed: *strong normalization of cut-elimination*. Perhaps surprisingly, the machinery involved to prove this gives the characterization 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 normalizing and that all characterization 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 3); unlike a 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 exist but few related results in the literature. As in Retoré [15], where a strong normalization result was proved for derivation reduction in the setting of the notion of intersection type assignment known as $\mathcal{D}$ [13], most papers consider the BCD-system [2] without the type-constant ω. Since we consider the type ω here, together with a type inclusion relation $\leq$, that strong normalization result itself is a true special case of the results of this paper presented in Section 6.

The *Approximation Theorem* hinted at above is a (perhaps lesser known) fundamental result for ITD and is more relevant in the context of semantics. Essentially following [27] and [1], the set of terms can be extended by adding the term-constant $\bot$. Adding also the reduction rules $\bot N \rightarrow_{\beta\bot} \bot$, and $\lambda x.\bot \rightarrow_{\beta\bot} \bot$ to the notion of reduction gives rise to the notion of *approximate normal forms* that are in essence finite rooted segments of Böhm-trees [1], 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

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

(see Ronchi Della Rocca and Venneri [16] and van Bakel [19] and [20]; for a uniform proof for many systems, see [7]). From this it also follows, that is, 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 [2], [19], and [20]).

The kind of intersection type assignment considered in this paper is that of [19], that is, the *strict* intersection type assignment system, a restricted version of the BCD-system of [2], 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 [19]); 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 [21].

2 Strict Intersection Type Assignment

In this section, we will present the Strict Intersection Type Assignment System as first presented in [19], which can be seen as a restricted version of the BCD-system as presented in [2]. The major feature of this restricted system, compared to the BCD-system, is that the $\leq$ relation on types is no longer contravariant 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 [1]; 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 2.1 (Lambda terms and β-equality [1])

1. The set Λ of *lambda terms* is defined by the syntax

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

2. The reduction relation $\rightarrow_\beta$ is defined as the contextual (i.e., compatible [1]) 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 2.2 (Types, statements, and bases)

1. 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
   (a) the set $\mathcal{T}_S$ of *strict types* is inductively defined by
   $$\sigma \ ::= \ \varphi \mid ((\sigma_1 \cap \cdots \cap \sigma_n) \rightarrow \sigma), \ (n \geq 0);$$
   (b) the set $\mathcal{T}$ of *intersection types* is defined by
   $$\{\sigma_1 \cap \cdots \cap \sigma_n \mid n \geq 0 \ \& \ \forall i \in \underline{n} \ [\sigma_i \text{ is a strict type}]\}.$$
   We will write ω for the empty intersection type.
2. 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 $M : \sigma$.
3. 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.
4. For bases $B_1, \ldots, B_n$, the basis $\cap\{B_1, \ldots, B_n\}$ is defined by $x{:}\sigma_1 \cap \cdots \cap \sigma_m \in \cap\{B_1, \ldots, B_n\}$ if and only if $\{x{:}\sigma_1, \ldots, x{:}\sigma_m\}$ is the (nonempty) set of all statements about x that occur in $B_1 \cup \cdots \cup B_n$.

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 $\cap\{B, \{x{:}\sigma\}\}$, 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, rightmost outermost brackets will be omitted.

In papers such as [2], [19], and [20], the type constant ω is introduced separately and is used as the universal type, that is, 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 [2], [19], and [20]) is defined as the set of terms that it can be assigned to (see Definition 2.5). Notice that if M can be assigned the type $\sigma_1 \cap \cdots \cap \sigma_n$, it can also be assigned $\sigma_1 \cap \cdots \cap \sigma_{n-1}$, so we get

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

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 [2], [19], and [20] all terms are typeable by ω, also $[\![\ ]\!] = [\![\omega]\!]$.

We will consider a preorder on types which takes into account the idempotence, commutativity, and associativity of the intersection type constructor and defines ω to be the maximal element.

Definition 2.3 (Relations on types)

1. The relation $\leq$ is defined as the least preorder (i.e., reflexive and transitive relation) on $\mathcal{T}$ such that
   $$\sigma_1 \cap \cdots \cap \sigma_n \ \leq \ \sigma_i, \text{ for all } i \in \underline{n}, \quad n \geq 1;$$
   $$\tau \leq \sigma_i, \text{ for all } i \in \underline{n} \Rightarrow \tau \leq \sigma_1 \cap \cdots \cap \sigma_n, \ n \geq 0.$$

2. The equivalence relation $\sim$ on types is defined by $\sigma \sim \tau \Leftrightarrow \sigma \leq \tau \leq \sigma$, and we will consider types modulo $\sim$.
3. 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 $B \sim B' \Leftrightarrow B \leq B' \leq B$.

$\mathcal{T}$ 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 $\sigma_1 \cap \cdots \cap \sigma_n$ (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 [2] (apart from dealing with intersection types occurring on the right of the arrow type constructor) or [20] also contained the alternative

$$\rho \leq \sigma \;\&\; \tau \leq \mu \Rightarrow \sigma \to \tau \leq \rho \to \mu.$$

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 2.4 (Cf. [20]) For all $\sigma, \tau \in \mathcal{T}$, $\sigma \leq \tau$ if and only if there are $n, m \geq 0$, $\sigma_i (\forall i \in \underline{n})$, $\tau_j (\forall j \in \underline{m})$ 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 2.5 (Strict type assignment and derivations)

1. *Strict intersection type assignment* and *strict intersection derivations* are defined by the following natural deduction system (where σ in rules $(\to E)$ and $(\to I)$ is in $\mathcal{T}$):

$$(Ax): \frac{}{B, x{:}\cap_{\underline{n}}\sigma_i \vdash x{:}\sigma_i} \; (n \geq 1, i \in \underline{n})$$

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

$$(\cap I): \frac{B \vdash M{:}\sigma_1 \quad \cdots \quad B \vdash M{:}\sigma_n}{B \vdash M{:}\cap_{\underline{n}}\sigma_i} \; (n \geq 0)$$

$$(\to I): \frac{B, x{:}\sigma \vdash M{:}\tau}{B \vdash \lambda x.M{:}\sigma \to \tau}$$

2. We write $B \vdash M{:}\sigma$ if this statement is derivable using a strict intersection derivation, and write $D :: B \vdash M{:}\sigma$ 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 2.6 $\exists p [B, x{:}\rho \vdash M : \sigma \ \& \ B \vdash N : \rho] \Leftrightarrow B \vdash M[N/x] : \sigma$.

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

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

$$\begin{array}{ll}
M \equiv y \neq x : & \Rightarrow: \exists \rho \ [B, x{:}\rho \vdash y : \sigma \ \& \ B \vdash N : \rho] \Rightarrow B \vdash y[N/x] : \sigma . \\
& \Leftarrow: B \vdash y[N/x] : \sigma \Rightarrow B \vdash y : \sigma \ \& \ B \vdash N : \omega .
\end{array}$$

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

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

□

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 2.7

1. 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 D $::$ $B, x{:}\cap_{\underline{n}} \sigma_i \vdash x : \sigma_j$ with $j \in \underline{n}$; we then write
$$\mathrm{D} = \langle Ax \rangle :: B, x{:}\cap_{\underline{n}} \sigma_i \vdash x : \sigma_j .$$
2. 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 subderivation $\mathrm{D}_1 :: B, x{:}\alpha \vdash M_1 : \beta$ in D; we then write
$$\mathrm{D} = \langle \mathrm{D}_1, \to I \rangle :: B \vdash \lambda x.M_1 : \alpha \to \beta .$$
3. 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 subderivations $\mathrm{D}_1 :: B \vdash P : \tau \to \sigma$ and $\mathrm{D}_2 :: B \vdash Q : \tau$ in D; we then write
$$\mathrm{D} = \langle \mathrm{D}_1, \mathrm{D}_2, \to E \rangle :: B \vdash PQ : \sigma .$$

4. If derivation D finishes with rule $(\cap I)$, there are $n \geq 0$, $\sigma_i\,(\forall i \in \underline{n})$ such that
$$D :: B \vdash M : \cap_{\underline{n}}\sigma_i,$$
and, for all $i \in \underline{n}$, there exists a $D_i :: B \vdash M : \sigma_i$ that is a subderivation of D; we then write
$$D = \langle D_1, \ldots, D_n, \cap I \rangle :: B \vdash M : \cap_{\underline{n}}\sigma_i.$$

We will often abbreviate the shorthand notation for derivations and, for example, write $\langle D_1, D_2, \to E\rangle$ instead of $\langle D_1, D_2, \to E\rangle :: B \vdash PQ : \sigma$.

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 subterms typed by ω) and say that these have the *same structure*; the types derived need not be the same.

As partially shown in [19], we have the following property.

Theorem 2.8 (Cf. [19]) *The following rules are admissible:*

$$(\leq) : \frac{B \vdash M : \sigma}{B' \vdash M : \tau}\,(B' \leq B, \sigma \leq \tau) \qquad (=_\beta) : \frac{B \vdash M : \sigma}{B \vdash N : \sigma}\,(M =_\beta N)$$

$$(\text{cut}) : \frac{B, x{:}\sigma \vdash M : \tau \quad B \vdash N : \sigma}{B \vdash M[N/x] : \tau}$$

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, $B \vdash (\lambda x.M)N : \sigma \Leftrightarrow B \vdash M[N/x] : \sigma$, 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 2.6.

(cut) By Lemma 2.6. □

3 Derivation Reduction

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

$$\langle\langle D_1 :: B, x{:}\rho \vdash P : \tau, \to I\rangle :: B \vdash \lambda x.P : \rho \to \tau,$$
$$D_2 :: B \vdash Q : \rho,\ \to E\rangle :: B \vdash (\lambda x.P)Q : \tau.$$

A derivation of this structure will be called a *redex*. We will prove in Section 4 that this notion of reduction is terminating, that is, strongly normalizable.

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 must—because the system at hand uses intersection types, including ω—be defined with care, since in $D :: B \vdash M : \sigma$ it is possible that M contains a redex whereas D does not.

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

$$\dfrac{\dfrac{\dfrac{}{B, x{:}\sigma \cap \tau \vdash x : \sigma}\,(Ax)}{B \vdash \lambda x.x : \sigma \cap \tau \to \sigma}\,(\to I) \qquad \dfrac{\begin{array}{c} D_1 \\ B \vdash N : \sigma \end{array} \qquad \begin{array}{c} D_2 \\ B \vdash N : \tau \end{array}}{B \vdash N : \sigma \cap \tau}\,(\cap I)}{B \vdash (\lambda x.x)N : \sigma}\,(\to E)$$

This derivation will reduce to $D_1 :: B \vdash N : \sigma$.

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

$$\begin{array}{l} \langle\langle D_1 :: B, x{:}\cap_{\underline{n}}\rho_i \vdash P : \tau, \to I\rangle :: B \vdash \lambda x.P : \cap_{\underline{n}}\rho_i \to \tau, \\ \qquad\qquad D_2 :: \langle D_2^1, \ldots, D_2^n, \cap I\rangle :: B \vdash Q : \cap_{\underline{n}}\rho_i, \to E\rangle :: B \vdash (\lambda x.P)Q : \tau; \end{array}$$

then the derivation is shaped like

$$\dfrac{\dfrac{\begin{array}{c} \dfrac{}{B, x{:}\cap_{\underline{n}}\rho_i \vdash x : \rho_{k_1}}\,(Ax) \cdots \dfrac{}{B, x{:}\cap_{\underline{n}}\rho_i \vdash x : \rho_{k_m}}\,(Ax) \\ \hline D_1 \\ B, x{:}\cap_{\underline{n}}\rho_i \vdash P : \tau \end{array}}{B \vdash \lambda x.P : \cap_{\underline{n}}\rho_i \to \tau}\,(\to I) \qquad \dfrac{\begin{array}{c} D_2^1 \\ B \vdash Q : \rho_1 \end{array} \cdots \begin{array}{c} D_2^n \\ B \vdash Q : \rho_n \end{array}}{B \vdash Q : \cap_{\underline{n}}\rho_i}\,(\cap I)}{B \vdash (\lambda x.P)Q : \tau}\,(\to E)$$

Contracting this redex will construct a derivation for the term $P[Q/x]$ and will be written as $D_1[D_2/x{:}\cap_{\underline{n}}\rho_i] :: B \vdash P[Q/x] : \tau$.

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

$$\dfrac{\begin{array}{c} D_1 \\ B, x{:}\cap_{\underline{n}}\rho_i \vdash P : \tau \end{array} \qquad \begin{array}{c} D_2 \\ B \vdash Q : \cap_{\underline{n}}\rho_i \end{array}}{B \vdash P[Q/x] : \tau}\,(\text{cut})$$

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* subderivation 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 $D_2^{k_j} :: B \vdash Q : \rho_{k_j}$ 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 3.1 (Derivation substitution) For $D :: B, x{:}\sigma \vdash M : \tau$ and $D_0 :: B \vdash N : \sigma$, the derivation $D[D_0/x{:}\sigma] :: B \vdash M[N/x] : \tau$, the result of *substituting* D_0 for $x{:}\sigma$ in D, is inductively defined by

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

   $$D_0 = \langle D_0^1 :: B \vdash N : \sigma_1, \ldots, D_0^n :: B \vdash N : \sigma_n, \cap I \rangle :: B \vdash N : \cap_{\underline{n}} \sigma_i,$$

   so, in particular, $D_0^j :: B \vdash N : \sigma_j$. Then $D\,[D_0/x{:}\sigma] = D_0^j$.

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

   $$D\,[D_0/x{:}\sigma] = \langle Ax \rangle :: B \vdash y : \tau.$$

3. $D = \langle D_1 :: B, x{:}\sigma, y{:}\alpha \vdash M_1 : \beta, \to I \rangle :: B, x{:}\sigma \vdash \lambda y.M_1 : \alpha \to \beta$. Let

   $$D' = D_1\,[D_0/x{:}\sigma] :: B, y{:}\alpha \vdash M_1[N/x] : \beta.$$

   Then $\langle D_1, \to I \rangle\,[D_0/x{:}\sigma] = \langle D', \to I \rangle :: B \vdash (\lambda y.M_1)[N/x] : \alpha \to \beta$.

4. $D = \langle D_1 :: B, x{:}\sigma \vdash P : \rho \to \tau, D_2 :: B, x{:}\sigma \vdash Q : \rho, \to E \rangle :: B, x{:}\sigma \vdash PQ : \tau$. Let

   $$\begin{aligned} D_1' &= D_1\,[D_0/x{:}\sigma] :: B \vdash P[N/x] : \rho \to \tau, \text{ and} \\ D_2' &= D_2\,[D_0/x{:}\sigma] :: B \vdash Q[N/x] : \rho; \end{aligned}$$

   then $\langle D_1, D_2, \to E \rangle [D_0/x{:}\sigma] = \langle D_1', D_2', \to E \rangle :: B \vdash (PQ)[N/x] : \tau$.

5. $D = \langle D_1, \ldots, D_n, \cap I \rangle :: B, x{:}\sigma \vdash M : \cap_{\underline{n}} \tau_i$. Let, for all $i \in \underline{n}$,

   $$D_i' = D_i\,[D_0/x{:}\sigma] :: B \vdash M[N/x] : \tau_i;$$

   then $\langle D_1, \ldots, D_n, \cap I \rangle [D_0/x{:}\sigma] = \langle D_1', \ldots, D_n', \cap I \rangle :: B \vdash M[N/x] : \cap_{\underline{n}} \tau_i$.

Before coming to the definition of derivation-reduction, we need to define the notion of 'position of a subderivation in a derivation'. This notion is needed in Definition 3.3 to make sure that when contracting a redex in one subderivation (branch) in a derivation ending with rule $(\cap I)$ all its "siblings" in neighboring branches are contracted as well.

Definition 3.2 Let D be a derivation and D′ be a subderivation of D. The position p of D′ in D is defined by

1. if $D' = D$, then $p = \varepsilon$;
2. if the position of D′ in D_1 is q and $D = \langle D_1, \to I \rangle$, or $D = \langle D_1, D_2, \to E \rangle$, then $p = 1q$;
3. if the position of D′ in D_2 is q and $D = \langle D_1, D_2, \to E \rangle$, then $p = 2q$;
4. if the position of D′ in D_i ($i \in \underline{n}$) is q, and $D = \langle D_1, \ldots, D_n, \cap I \rangle$, 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 subderivation with type different from ω.

Definition 3.3 (Derivation reduction) We say that the derivation $D :: B \vdash M:\sigma$ *reduces to* $D' :: B \vdash M':\sigma$ *at position* p *with redex* R, if and only if

$\sigma \in \mathcal{T}_S$

1. $D = \langle\langle D_1, \to I\rangle, D_2, \to E\rangle :: B \vdash (\lambda x.M)N:\sigma$ (a derivation of this shape is called a *redex*); then D reduces to
$$D_1\,[D_2/x{:}\rho] :: B \vdash M[N/x]:\sigma$$
at position ε with redex $(\lambda x.M)N$.
2. If D_1 reduces to D'_1 at position p with redex R, then
   (a) $D = \langle D_1, \to I\rangle :: B \vdash \lambda x.M_1:\alpha \to \beta$ reduces at position $1p$ with redex R to $D' = \langle D'_1, \to I\rangle :: B \vdash \lambda x.M'_1:\alpha \to \beta$.
   (b) $D = \langle D_1, D_2, \to E\rangle :: B \vdash PQ:\sigma$ reduces at position $1p$ with redex R to $D' = \langle D'_1, D_2, \to E\rangle :: B \vdash P'Q:\sigma$.
   (c) $D = \langle D_2, D_1, \to E\rangle :: B \vdash PQ:\sigma$ reduces at position $2p$ with redex R to $D' = \langle D_2, D'_1, \to E\rangle :: B \vdash PQ':\sigma$.

$\sigma = \cap_{\underline{n}}\sigma_i$

If $D :: B \vdash M:\cap_{\underline{n}}\sigma_i$, then, for every $i \in \underline{n}$, there are $D_i :: B \vdash M:\sigma_i$ such that $D = \langle D_1, \ldots, D_n, \cap I\rangle$. 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

1. there is no redex at position p because there is no subderivation at that position; since R is a subterm of M, it has to be part of a term that is typed with ω in D_j; let $R \to_\beta R'$ and $D'_j = D_j[R'/R]$ (i.e., D_j where each R is replaced by R'), or
2. D_j reduces to D'_j at position p with redex R.

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

We write $D \to_{\mathcal{D}} D'$ 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 $D_1 \to_{\mathcal{D}} D_2 \to_{\mathcal{D}} D_3$, then $D_1 \to_{\mathcal{D}} D_3$.

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

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

Example 3.4 Let D_1 be the derivation (with $B_1 = x{:}(\alpha \to \beta \to \gamma) \cap \alpha, y{:}(\gamma \to \delta) \cap \beta$, and $\Theta \equiv \lambda xy.y(xxy)$):

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

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

$$\dfrac{\begin{array}{c}\mathrm{D}_2\\ \vdash \Theta : \tau \to (\omega \to \rho) \to \rho\end{array} \quad \begin{array}{c}\mathrm{D}_1\\ \vdash \Theta : \tau\end{array}}{\vdash \Theta\Theta : (\omega \to \rho) \to \rho}\,(\to E)$$

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

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

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 3.5 *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 3.3. □

The following states some standard properties of strong normalization.

Lemma 3.6

1. *If* $SN(\langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle)$, *then* $SN(\mathrm{D}_1)$ *and* $SN(\mathrm{D}_2)$.
2. *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* $SN(\langle \mathrm{D}_1, \mathrm{D}_2, \to E\rangle :: \cap\{B_1, B_2\} \vdash xM_1, \ldots, M_n N : \tau)$.
3. $\forall i \in \underline{n}\ [SN(\mathrm{D}_i :: B \vdash M : \sigma_i)]$ *if and only if* $SN(\langle \mathrm{D}_1, \ldots, \mathrm{D}_n, \cap I\rangle)$.
4. *If* $SN(\langle \ldots \langle \mathrm{D}_1[\mathrm{D}_2/y{:}\rho]\rangle \ldots\rangle :: B \vdash M[N/x]\overrightarrow{P} : \sigma)$ *and* $SN(\mathrm{D}_2 :: B \vdash N : \rho)$, *then* $SN(\langle \ldots \langle\langle \mathrm{D}_1, \to I\rangle, \mathrm{D}_2, \to E\rangle \ldots\rangle :: B \vdash (\lambda y.M)Q\overrightarrow{P} : \sigma)$.

Proof Easy, by Definition 3.3. □

4 Strong Normalization of Derivation Reduction

In this section, we will prove a strong normalization result for derivation reduction. In order to prove that each derivation in $\vdash$ is strongly normalizable with respect to $\rightarrow_{\mathcal{D}}$, a notion of computable ([18], [12]) derivations will be introduced. We will show that all computable derivations are strongly normalizable with respect to derivation reduction and then that all derivations in $\vdash$ are computable.

Definition 4.1 (Computability Predicate) *Comp*(D) is defined recursively on types by

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

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

Lemma 4.2 *If* $Comp(\mathrm{D} :: B \vdash M : \sigma)$, $B' \leq B$, $\sigma \leq \sigma'$, *then* $Comp(\mathrm{D}' :: B' \vdash M : \sigma')$ *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 normalization, and that, for the so-called neutral objects, also the converse holds.

Lemma 4.3

1. $Comp(\mathrm{D} :: B \vdash M : \sigma) \Rightarrow SN(\mathrm{D})$.
2. $SN(\mathrm{D} :: B \vdash x M_1, \ldots, M_m : \sigma) \Rightarrow Comp(\mathrm{D})$.

Proof By simultaneous induction on the structure of types.

$\sigma = \varphi$ Directly by Definition 4.1.

$\sigma = \alpha \rightarrow \beta$ (1) 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 (2), $Comp(\mathrm{D}')$. Since $Comp(\mathrm{D})$, by Lemma 4.2, also $Comp(\mathrm{D}'' :: B, x{:}\alpha \vdash M : \alpha \rightarrow \beta)$ (notice that D and D″ are almost identical, but for the occurrences of $x{:}\alpha$ in the basis), so, by Definition 4.1, $Comp(\langle \mathrm{D}'', \mathrm{D}', \rightarrow E \rangle :: B, x{:}\alpha \vdash Mx : \beta)$. Then, by induction (1), $SN(\langle \mathrm{D}'', \mathrm{D}', \rightarrow E \rangle)$, and, by Lemma 3.6(1), $SN(\mathrm{D}'')$. Then also $SN(\mathrm{D})$.

(2) Assume $Comp(\mathrm{D}' :: B' \vdash N : \alpha)$, then by induction (1), $SN(\mathrm{D}')$. Then, by Lemma 3.6(2), $SN(\langle \mathrm{D}, \mathrm{D}', \rightarrow E \rangle :: \cap\{B, B'\} \vdash x M_1, \ldots, M_m N : \beta)$. Then $Comp(\langle \mathrm{D}, \mathrm{D}', \rightarrow E \rangle)$ by induction (2), so by Definition 4.1, $Comp(\mathrm{D})$.

$\sigma = \cap_{\underline{n}} \sigma_i$ Easy, using Definition 4.1, Lemma 3.6(3), and induction.

□

The following theorem (4.5) shows that, in a derivation, replacing subderivations for term-variables by computable derivations yields a computable derivation. Before coming to this result, first an auxiliary lemma must be proved that formulates that the computability predicate is closed for subject-expansion with respect to derivation reduction.

Lemma 4.4 *Let* $\mathrm{D} = \langle \ldots \mathrm{D}_1[\mathrm{D}_2/y{:}\rho] \ldots, \to E \rangle :: B \vdash M[Q/y]\overrightarrow{P} : \sigma$*; if* $Comp(\mathrm{D})$ *and* $Comp(\mathrm{D}_2 :: B \vdash Q : \rho)$*, then*

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

Proof By induction on the structure of types.

$\sigma = \varphi$ $\quad Comp(\mathrm{D})$ & $Comp(\mathrm{D}_2 :: B \vdash Q : \rho) \Rightarrow$ 4.3(1)
$SN(\mathrm{D})$ & $SN(\mathrm{D}_2) \Rightarrow$ 3.6(4)
$SN(\langle \ldots \langle\langle \mathrm{D}_1, \to I\rangle, \mathrm{D}_2, \to E\rangle \ldots, \to E\rangle :: B \vdash (\lambda y.M)Q\overrightarrow{P} : \varphi) \Rightarrow$ (4.1)
$Comp(\langle \ldots \langle\langle \mathrm{D}_1, \to I\rangle, \mathrm{D}_2, \to E\rangle \ldots, \to E\rangle)$.

$\sigma = \alpha \to \beta$ $\quad Comp(\mathrm{D})$ & $Comp(\mathrm{D}_2)$ & $Comp(\mathrm{D}' :: B \vdash N : \alpha) \Rightarrow$ (4.1)
$Comp(\mathrm{D}_2)$ & $Comp(\langle \mathrm{D}, \mathrm{D}', \to E\rangle :: B \vdash M[Q/y]\overrightarrow{P}N : \beta) \Rightarrow$ (IH)
$Comp(\langle\langle\langle \ldots \langle \mathrm{D}_1, \to I\rangle, \mathrm{D}_2, \to E\rangle \ldots, \to E\rangle, \mathrm{D}', \to E\rangle :: B \vdash (\lambda y.M)Q\overrightarrow{P}N : \beta) \Rightarrow$ (4.1)
$Comp(\langle \ldots \langle\langle \mathrm{D}_1, \to I\rangle, \mathrm{D}_2, \to E\rangle \ldots, \to E\rangle)$.

$\sigma = \cap_{\underline{n}}\sigma_i$ $\quad$ By induction and Definition 4.1.

□

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 $[N_1/x_1, \ldots, N_n/x_n]$, and so on.

Theorem 4.5 (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*

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

Proof By induction on the structure of derivations.

(Ax) We distinguish the following two cases:

$M \equiv x$: 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 4.1, $Comp(\mathrm{D}_i)$, and by Definition 3.1, $\mathrm{D}^0\,[\overrightarrow{\mathrm{D}/x{:}\mu}] = \mathrm{D}_i$.

$M \equiv y, y \neq x$: By Definition 3.1, $\mathrm{D}^0\,[\overrightarrow{\mathrm{D}/x{:}\mu}] = \langle Ax\rangle :: B \vdash y : \tau$, which is computable by Lemma 4.3(2).

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

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

and, by Definition 4.1, $Comp(D^0[\overrightarrow{D/x{:}\mu}] :: B \vdash M[\overrightarrow{N/x}]:\cap_{\underline{n}}\sigma_i)$.

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

$$\begin{array}{ll} \forall j \in \underline{m}\ [Comp(D_j)]\ \&\ Comp(D') & \Rightarrow \text{(IH)} \\ Comp(D^1[\overrightarrow{D/x{:}\mu}, D'/y{:}\rho] :: B \vdash M[\overrightarrow{N/x}, Q/y]:\tau) & \Rightarrow (4.4) \\ Comp(\langle\langle D^1[\overrightarrow{D/x{:}\mu}], \rightarrow I\rangle, D', \rightarrow E\rangle :: B \vdash (\lambda y.M[\overrightarrow{N/x}])Q{:}\tau) & \end{array}$$

so, by Definition 4.1, $Comp(\langle D^1[\overrightarrow{D/x{:}\mu}], \rightarrow I\rangle{::}B \vdash \lambda y.M[\overrightarrow{N/x}]\rho \rightarrow \tau)$, so also

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

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

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

Then, by Definition 4.1,
$Comp(\langle D^1[\overrightarrow{D/x{:}\mu}], D^2[\overrightarrow{D/x{:}\mu}], \rightarrow E\rangle :: B \vdash M_1[\overrightarrow{N/x}]M_2[\overrightarrow{N/x}]:\sigma)$,
so also $Comp(\langle D_1, D_2, \rightarrow E\rangle[\overrightarrow{D/x{:}\mu}] :: B \vdash (M_1M_2)[\overrightarrow{N/x}]:\sigma)$.

□

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

Theorem 4.6 *If* $D :: B \vdash M:\sigma$, *then* $SN(D)$.

Proof By Lemma 4.3(2), for every $x{:}\tau \in B$, $Comp(D_x :: B \vdash x:\tau)$, so by Theorem 4.5, $Comp(D :: B \vdash M:\sigma)$. Then, by Lemma 4.3(1), $SN(D)$. □

5 Approximation

In Sections 6 and 7 we will show two main results that are both direct consequences of the strong normalization result proved in Section 4. Both results have been proven, at least partially, in [19] and [20]. 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 [1], $\bot$ 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 [1]).

Definition 5.1 (Approximate normal forms)

1. The set of $\Lambda\bot$-*terms* is defined like the set Λ of lambda terms by

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

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

2. The notion of reduction $\rightarrow_{\beta\bot}$ is defined as $\rightarrow_\beta$, extended by $\lambda x.\bot \rightarrow_{\beta\bot} \bot$ and $\bot M \rightarrow_{\beta\bot} \bot$.
3. The set of *normal forms for elements of* $\Lambda\bot$ *with respect to* $\rightarrow_{\beta\bot}$ is the set $\mathcal{N}$ of $\lambda\bot$-*normal forms* or *approximate normal forms* ranged over by A, inductively defined by

$$A ::= \bot \mid \lambda x.A \ (A \not\equiv \bot) \mid x A_1, \ldots, A_n \ (n \geq 0).$$

The rules of the system $\vdash$ are generalized 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 $n = 0$, that is, in a subterm typed with ω. Notice that the terms $\lambda x.\bot$ and $\bot M_1 \ldots M_n$ are typeable by ω only.

Definition 5.2 ($\sqsubseteq$, (direct) approximants)

1. The partial order $\sqsubseteq \subseteq (\Lambda\bot)^2$ is defined as the least preorder such that

$$\bot \sqsubseteq M$$
$$M \sqsubseteq M' \Rightarrow \lambda x.M \sqsubseteq \lambda x.M'$$
$$M_1 \sqsubseteq M_1' \ \& \ M_2 \sqsubseteq M_2' \Rightarrow M_1 M_2 \sqsubseteq M_1' M_2'.$$

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

2. The relation $\underset{\sim}{\sqsubseteq} \ \subseteq \mathcal{N} \times \Lambda$ is defined by

$$A \underset{\sim}{\sqsubseteq} M \Leftrightarrow \exists M' =_\beta M \, [A \sqsubseteq M'].$$

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

Lemma 5.3 $B \vdash M : \sigma \ \& \ M \sqsubseteq M' \Rightarrow B \vdash M' : \sigma$.

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 5.4 (Join, compatible terms)

1. On $\Lambda\bot$, the partial mapping *join*, $\sqcup : \Lambda\bot \times \Lambda\bot \rightarrow \Lambda\bot$, is defined by

$$\bot \sqcup M \equiv M \sqcup \bot \equiv M$$
$$x \sqcup x \equiv x$$
$$(\lambda x.M) \sqcup (\lambda x.N) \equiv \lambda x.(M \sqcup N)$$
$$(M_1 M_2) \sqcup (N_1 N_2) \equiv (M_1 \sqcup N_1)\,(M_2 \sqcup N_2).$$

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

We will use $\sqcup_{\underline{n}} M_i$ for the term $M_1 \sqcup \cdots \sqcup M_n$. Note that $\bot$ can be defined as the empty join, that is, 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

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

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 5.5 *If $M_1 \sqsubseteq M$, and $M_2 \sqsubseteq M$, then $M_1 \sqcup M_2$ is defined, and*

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

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

1. 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.)

2. 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 1, and for the other, $M_1 \sqcup M_2 \equiv x$. Obviously, $x \sqsubseteq x \sqcup x$, $x \sqsubseteq x \sqcup x$, and $x \sqcup x \sqsubseteq x$.

3. 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 1, 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 $\lambda x.N_1 \sqcup N_2 \equiv (\lambda x.N_1) \sqcup (\lambda x.N_2)$.

4. 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 1, 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 $(P_1 \sqcup P_2)(Q_1 \sqcup Q_2) \equiv (P_1 Q_1) \sqcup (P_2 Q_2)$. □

6 Normalization Results

In what follows below, first an approximation result will be proved, that is, 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 characterization of (head-)normalization of lambda terms using intersection types follows easily, that is, all terms having a (head) normal form are typeable in $\vdash$ (with a type without ω-occurrences). The second result is the well-known characterization of strong normalization of typeable lambda terms, that is, all terms, typeable in $\vdash$ without using the type-constant ω, are strongly normalizable.

First we give some auxiliary definitions and results. The first is a notion of type assignment similar to that of Definition 2.5 but differs in that it assigns ω only to the term $\bot$.

Definition 6.1 $\bot$-*type assignment* and $\bot$-*derivations* are defined by the following natural deduction system (where σ in rules $(\rightarrow E)$ and $(\rightarrow I)$ is in $\mathcal{T}$):

$$(Ax): \quad \frac{}{B, x{:}\cap_{\underline{n}}\sigma_i \vdash_\bot x:\sigma_i} \; (n \geq 1, i \in \underline{n})$$

$$(\rightarrow E): \quad \frac{B \vdash_\bot M:\sigma \rightarrow \tau \quad B \vdash_\bot N:\sigma}{B \vdash_\bot MN:\tau}$$

$$(\cap I): \quad \frac{B \vdash_\bot M_1:\sigma_1 \quad \ldots \quad B \vdash_\bot M_n:\sigma_n}{B \vdash_\bot \sqcup_{\underline{n}} M_i : \cap_{\underline{n}}\sigma_i} \; (n \geq 0)$$

$$(\rightarrow I): \quad \frac{B, x{:}\sigma \vdash_\bot M:\tau}{B \vdash_\bot \lambda x.M:\sigma \rightarrow \tau}$$

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

Notice that, by rule $(\cap I)$, $\varnothing \vdash_\bot \bot:\omega$, 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 6.2

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

Proof

(1) By induction on the structure of derivations in $\vdash_\bot$.

- (Ax) Immediate.
- $(\cap I)$ Then there are $n \geq 0$, σ_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 5.5, $M_i \sqsubseteq M$ for all $i \in \underline{n}$, by Lemma 5.3, for every $i \in \underline{n}$, $B \vdash M:\sigma_i$, so by $(\cap I)$, $B \vdash M:\cap_{\underline{n}}\sigma_i$.
- $(\rightarrow I)$ Then $M \equiv \lambda x.M'$, and $\sigma = \alpha \rightarrow \beta$, and $B, x{:}\alpha \vdash_\bot M':\beta$. Then, by induction, $B, x{:}\alpha \vdash M':\beta$, so by $(\rightarrow I)$, $B \vdash \lambda x.M':\alpha \rightarrow \beta$.
- $(\rightarrow E)$ Then $M \equiv M_1 M_2$, and there exists τ such that $B \vdash_\bot M_1:\tau \rightarrow \sigma$, and $B \vdash_\bot M_2:\tau$. Then, by induction, $B \vdash M_1:\tau \rightarrow \sigma$, and $B \vdash M_2:\tau$, so by $(\rightarrow E)$, $B \vdash M_1 M_2:\sigma$.

(2) 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 $B \vdash_\bot M_i:\sigma_i$ (notice that then, by Lemma 5.5, these M_i are compatible). Then, by rule $(\cap I)$, we have $B \vdash_\bot \sqcup_{\underline{n}} M_i:\sigma_i$. Notice that, by Lemma 5.5, $\sqcup_{\underline{n}} M_i \sqsubseteq M$.
- $(\rightarrow I)$ Then $M \equiv \lambda x.M_1$, and $\sigma = \alpha \rightarrow \beta$, and $B, x{:}\alpha \vdash M_1:\beta$. So, by induction, there exists $M_1' \sqsubseteq M_1$ such that $B, x{:}\alpha \vdash_\bot M_1':\beta$. Then, by rule $(\rightarrow I)$ we obtain $B \vdash_\bot \lambda x.M_1':\alpha \rightarrow \beta$. Notice that $\lambda x.M_1' \sqsubseteq \lambda x.M_1$.
- $(\rightarrow E)$ Then $M \equiv M_1 M_2$, and there is a τ such that $B \vdash M_1:\tau \rightarrow \sigma$, and $B \vdash M_2:\tau$. Then, by induction, there are $M_1' \sqsubseteq M_1$ and

$M'_2 \sqsubseteq M_2$ such that $B \vdash_\bot M'_1 : \tau \to \sigma$, and $B \vdash_\bot M'_2 : \tau$. Then, by $(\to E)$, $B \vdash_\bot M'_1 M'_2 : \sigma$. 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 $\sqcup_{\underline{n}} M_i = \bot$. Moreover, since M' need not be the same as M, the second derivation in part 2 is not exactly the same; however, it has the same structure in terms of applied derivation rules.

Example 6.3 Let D'_1 be the derivation D_1 from Example 3.4 but built using the rules of $\vdash_\bot$ 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 3.4) the type derived in D'_1, then the $\vdash_\bot$-variant of $\langle D_2, D_1, \to E\rangle$ will be

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

Notice that $\lambda xy.y\,\bot \sqsubseteq \Theta$. This derivation reduces to

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

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

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

Using Theorem 4.6, as for the BCD-system (see [16]) and the system of [20], the relation between types assignable to a lambda term and those assignable to its approximants can be formulated as follows.

Theorem 6.4 (Approximation) $B \vdash M : \sigma \Leftrightarrow \exists A \in \mathcal{A}(M)[B \vdash A : \sigma]$.

Proof

($\Rightarrow$) If $D :: B \vdash M : \sigma$, then, by Theorem 4.6, $SN(D)$. Let $D' :: B \vdash N : \sigma$ be a normal form of D with respect to $\to_{\mathcal{D}}$, then by Lemma 3.5, $M \twoheadrightarrow_\beta N$ and, by Lemma 6.2(2), there exists $P \sqsubseteq N$ such that $D' :: B \vdash_\bot P : \sigma$. So, in particular, P contains no redexes (no typed redexes since D' is in normal form, and none untyped since only $\bot$ 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 5.3, $B \vdash M' : \sigma$, and, by Theorem 2.8, also $B \vdash M : \sigma$. □

In [20], this result was obtained separately using a computability predicate. Using the previous theorem, the following becomes easy.

Theorem 6.5 (Head-normalization [19]) *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 6.4, there exists an $A \in \mathcal{A}(M)$ such that $B \vdash A : \sigma$. Then, by Definition 5.2, there exists $M' =_\beta M$ such that $A \sqsubseteq M'$. Since $\sigma \in \mathcal{T}_S$, $A \not\equiv \bot$, so A is either $\lambda x.A'$ or $x A_1, \ldots, A_n$. Since M' matches A ($A \sqsubseteq M'$), M' is either $\lambda x.M_1$ or $x M_1, \ldots, M_n$; 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 $x M_1, \ldots, M_n$, 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 $B = B' \setminus x$, and $\sigma = \tau \to \sigma'$, otherwise $B = B'$ and $\sigma = \omega \to \sigma'$.

(b) $M' \equiv x M_1, \ldots, M_n$. Take $B = x{:}\omega \to \cdots \to \omega \to \varphi$ and $\sigma = \varphi$.

Notice that, in all cases, $B \vdash M' : \sigma$ and $\sigma \in \mathcal{T}_S$. Then, by Theorem 2.8, $B \vdash M : \sigma$. □

7 ω-free Type Assignment

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

Definition 7.1 (ω-free types)

1. 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 $\mathcal{T}_\omega$ of *ω-free intersection types* is defined by

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

2. The relation $\leq$ is defined on ω-free types as the least preorder on $\mathcal{T}_\omega$ 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 2.3.

3. The equivalence relation $\sim$ on types is defined by $\sigma \sim \tau \Leftrightarrow \sigma \leq \tau \leq \sigma$, and we will work with types modulo $\sim$.

Definition 7.2 (ω-free type assignment)

1. *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 $(\rightarrow I)$ is in $\mathcal{T}$):

$$(Ax): \quad \frac{}{B, x{:}\cap_{\underline{n}}\sigma_i \;\vdash_{\not\omega}\; x:\sigma_i}\;(n \geq 1, i \in \underline{n})$$

$$(\rightarrow E): \quad \frac{B \;\vdash_{\not\omega}\; M:\sigma \rightarrow \tau \quad B \;\vdash_{\not\omega}\; N:\sigma}{B \;\vdash_{\not\omega}\; MN:\tau}$$

$$(\cap I): \quad \frac{B \;\vdash_{\not\omega}\; M:\sigma_1 \quad \ldots \quad B \;\vdash_{\not\omega}\; M:\sigma_n}{B \;\vdash_{\not\omega}\; M:\cap_{\underline{n}}\sigma_i}\;(n \geq 1)$$

$$(\rightarrow I): \quad \frac{B, x{:}\sigma \;\vdash_{\not\omega}\; M:\tau}{B \;\vdash_{\not\omega}\; \lambda x.M:\sigma \rightarrow \tau}$$

2. We will write $B \vdash_{\not\omega} M:\sigma$ if this statement is derivable using a ω-free intersection derivation and write $D :: B \vdash_{\not\omega} M:\sigma$ to specify that this result was obtained through the derivation D.

The following properties hold.

Lemma 7.3

1. $B \vdash_{\not\omega} M:\sigma \Rightarrow \{x{:}\sigma \in B \mid x \in fv(M)\} \vdash_{\not\omega} M:\sigma$.
2. $B \vdash_{\not\omega} M:\sigma \;\&\; B' \leq B \Rightarrow B' \vdash_{\not\omega} M:\sigma$.
3. *If* $D :: B \vdash_{\not\omega} M:\sigma$*, then* $D :: B \vdash M:\sigma$.

Proof Easy. □

To prepare the characterization of terms by their assignable types, first it 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 normalizable terms are typeable without ω.

Lemma 7.4 ([20])

1. *If $B \vdash_{\not\omega} A:\sigma$, and B, σ are ω-free, then A is $\bot$-free.*
2. *If A is $\bot$-free, then there are B and σ such that $B \vdash_{\not\omega} A:\sigma$.*

Proof By induction on the structure of approximate normal forms.

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

$A \equiv x$	Immediate.
$A \equiv \bot$	Impossible, by inspecting the rules of $\vdash_{\not\omega}$.
$A \equiv \lambda x.A'$	By $(\rightarrow I)$ there are α, β such that $\sigma = \alpha \rightarrow \beta$ and $B, x{:}\alpha \vdash_{\not\omega} A':\beta$. 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 $(\rightarrow E)$ and (Ax) there are $m, \sigma_i (\forall i \in \underline{n}), \tau_j (\forall j \in \underline{m})$ such that for every $i \in \underline{n}$, $B \vdash_{\not\omega} A_i:\sigma_i$, and $x{:}\cap_{\underline{m}}\tau_i \in B$, and, for some $j \in m$, $\sigma_1 \rightarrow \cdots \rightarrow \sigma_n \rightarrow \sigma = \tau_j$. 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.

(2)

$A \equiv x$	$x{:}\varphi \vdash_{\omega} x : \varphi$.
$A \equiv \lambda x.A'$	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, $B \setminus x \vdash_{\omega} \lambda x.A' : \sigma \to \tau$.
$A \equiv x A_1, \ldots, A_n$	By induction there are $\sigma_i (\forall i \in \underline{n})$ such that $B \vdash_{\omega} A_i : \sigma_i$ for every $i \in \underline{n}$. Then $\cap\{B, \{x{:}\sigma_1 \to \cdots \to \sigma_n \to \varphi\}\} \vdash_{\omega} x A_1, \ldots, A_n : \varphi$.

□

Now, as also shown in [19], it is possible to characterize normalizable terms.

Theorem 7.5 ([19], [20]) *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

- (⇒) If $B \vdash M : \sigma$, by Theorem 6.4, there exists $A \in \mathcal{A}(M)$ such that $B \vdash A : \sigma$. Since B, σ are ω-free, by Lemma 7.4(1), this A is $\perp$-free. By Definition 5.1 there exists $M' =_{\beta} M$ such that $A \sqsubseteq M'$. Since A contains no $\perp$, $A \equiv M'$, so M' is a normal form, so, especially, M has a normal form.
- (⇐) If M' is the normal form of M, then it is a $\perp$-free approximate normal form. Then by Lemma 7.4(2) there are B, σ such that $B \vdash_{\omega} M' : \sigma$. Then, by Theorem 2.8, $B \vdash M : \sigma$, 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 7.6 *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 generalization. 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 7.3(1) 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

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

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

$$\begin{array}{ll}
B \vdash_{\omega} M_1[N/x]M_2[N/x]:\sigma \;\&\; B \vdash_{\omega} N:\rho & \Rightarrow (\rightarrow E) \\
\exists \tau\, [B \vdash_{\omega} M_1[N/x]:\tau \rightarrow \sigma \;\&\; B \vdash_{\omega} M_2[N/x]:\tau] \;\&\; B \vdash_{\omega} N:\rho & \Rightarrow (\mathrm{IH}) \\
\exists \tau\, [B \vdash_{\omega} (\lambda x.M_1)N:\tau \rightarrow \sigma \;\&\; B \vdash_{\omega} (\lambda x.M_2)N:\tau] & \Rightarrow (\rightarrow E), \\
 & \quad (\rightarrow I), \& \; (\cap I) \\
\exists \rho_1, n, \rho_2^i, \tau, \tau_i\; (\forall i \in \underline{n})\; [B, x{:}\rho_1 \vdash_{\omega} M_1:\tau \rightarrow \sigma \;\&\; B \vdash_{\omega} N:\rho_1 & \\
\qquad \& \; \tau = \cap_{\underline{n}} \tau_i \;\&\; \forall i \in \underline{n}\, [B, x{:}\rho_2^i \vdash_{\omega} M_2:\tau_i \;\&\; B \vdash_{\omega} N:\rho_2^i]] & \Rightarrow (\cap I),\, 7.3(2) \\
\exists \rho_1, \rho_2, \tau [B, x{:}\rho_1 \cap \rho_2 \vdash_{\omega} M_1:\tau \rightarrow \sigma \;\&\; B \vdash_{\omega} N:\rho_1 \cap \rho_2 & \\
\qquad \& \; B, x{:}\rho_1 \cap \rho_2 \vdash_{\omega} M_2:\tau] & \Rightarrow (\rightarrow E) \\
\exists \rho' [B, x{:}\rho' \vdash_{\omega} M_1 M_2:\sigma \;\&\; B \vdash_{\omega} N:\rho'] & \Rightarrow (\rightarrow I) \\
\exists \rho'\, [B \vdash_{\omega} \lambda x.(M_1 M_2):\rho' \rightarrow \sigma \;\&\; B \vdash_{\omega} N:\rho'] & \Rightarrow (\rightarrow E) \\
B \vdash_{\omega} (\lambda x.(M_1 M_2))N:\sigma &
\end{array}$$

□

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

if $B \vdash_{\omega} \mathrm{C}[M[N/x]]:\sigma$ and $B \vdash_{\omega} N:\rho$, 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 counterexample, 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

$$B, z{:}\sigma, y{:}\sigma \rightarrow \tau \vdash_{\omega} (\lambda b.z)(yz):\sigma.$$

We can therefore only state that we can derive

$$B \vdash_{\omega} \lambda yz.(\lambda b.z)(yz):(\sigma \rightarrow \tau) \rightarrow \rho \text{ and } B \vdash_{\omega} \lambda yz.z:\tau \rightarrow \rho$$

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

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

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 normalizable term, no matter what basis or type is used.

Lemma 7.6 is also essentially the proof for the statement that each strongly normalizable term can be typed in the system $\vdash_{\omega}$, a property that we will now show. Theorem 7.9 shows that the set of strongly normalizable terms is exactly the set of terms typeable in the intersection system without using the type constant ω. The same result was stated in [19] for the BCD-system, but the proof there was not complete. The proof of the crucial lemma as presented below (Lemma 7.8) and part ($\Leftarrow$) of the proof of Theorem 7.9 are essentially due to Venneri and goes by induction on the leftmost outermost reduction path.

First we will introduce the notion of leftmost outermost reduction.

Definition 7.7 An occurrence of a redex $\mathrm{R} = (\lambda x.P)Q$ in a term M is called the *leftmost outermost redex of* M ($lor(M)$) if

1. there is no redex R' in M such that $\mathrm{R}' = \mathrm{C}[\mathrm{R}]$ (*outermost*);
2. there is no redex R' in M such that $M = \mathrm{C}_0[\mathrm{C}_1[\mathrm{R}']\mathrm{C}_2[\mathrm{R}]]$ (*leftmost*).

$M \rightarrow_{lor} N$ 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 leftmost outermost reduction.

Lemma 7.8 *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 The proof is 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 $M \equiv \lambda x_1, \ldots, x_k.V P_1, \ldots, P_n$ $(k, n \geq 0)$, where either

1. 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
2. $V \equiv y$, so there is a $j \in \underline{n}$ such that $lor(M) = lor(P_j)$, and $P_j \to_{lor} P'$, and $N \equiv \lambda x_1, \ldots, x_k.y P_1, \ldots, P', \ldots, P_n$.

In either case we have, by Lemma 7.3, 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):

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

We distinguish two cases:

(1) $V' \equiv P[Q/y]$. Let $B_2 = B'$, then $\cap\{B_1, B_2\} \vdash_{\not\omega} (\lambda y.P)Q{:}\gamma_1 \to \cdots \to \gamma_n \to \beta$, by Lemma 7.6.

(2) $V' \equiv y$. Then, by induction, there are B', ρ such that $B' \vdash_{\not\omega} P_j:\rho$. Take $\mu = \gamma_1 \to \cdots \rho \cdots \to \gamma_n \to \beta$, $B_2 = B', y{:}\mu$, then $\cap\{B_1, B_2\} \vdash_{\not\omega} y:\mu$.

In either case, $\cap\{B_1, B_2\} \vdash_{\not\omega} V P_1, \ldots, P_n:\beta$. Let, for all $i \in \underline{k}$, $x_i{:}\beta_i \in \cap\{B_1, B_2\}$, then $\cap\{B_1, B_2\} \setminus x_1, \ldots, x_k \vdash_{\not\omega} \lambda x_1, \ldots, x_k.V P_1, \ldots, P_n:\beta_1 \to \cdots \to \beta_k \to \beta$. □

We can now show that all strongly normalizable terms are exactly those typeable in $\vdash_{\not\omega}$.

Theorem 7.9 $\exists B, \sigma\ [B \vdash_{\not\omega} M:\sigma] \Leftrightarrow M$ *is strongly normalizable with respect to* $\to_\beta$.

Proof

- ($\Rightarrow$) If D :: $B \vdash_{\not\omega} M:\sigma$, then, by Lemma 7.3(3), also D :: $B \vdash M:\sigma$. Then, by Theorem 4.6, D is strongly normalizable 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 normalizable with respect to $\to_\beta$.
- ($\Leftarrow$) With induction on the maximum of the lengths of lor-reduction sequences for a strongly normalizable term to its normal form (denoted by $\#(M)$).
  - (a) If $\#(M) = 0$, then M is in normal form, and by Lemma 7.4(2), there exist B and $\sigma \in \mathcal{T}$ such that $B \vdash_{\not\omega} M:\sigma$.
  - (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 subterm of a redex in M), so by induction $B \vdash_{\not\omega} M:\sigma$ and $B' \vdash_{\not\omega} Q:\tau$, for some B, B', σ, and τ. Then, by Lemma 7.8, there exist B_1, ρ such that $B_1 \vdash_{\not\omega} M:\rho$.

□

8 Conclusions and Future Work

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

Appendix A Extended Examples

We give an example of a nonstrongly normalizing 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 derivation has a normal from and construct the reduction sequences. The derivation we will construct is similar to the one of Example 3.4 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 3.4. From these two, by applying $(\cap I)$, we get $D_3 :: \vdash \Theta : (\tau \to (\omega \to \rho) \to \rho) \cap \tau$:

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

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

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

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

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

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

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

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

$$\dfrac{\begin{array}{c} D_5 \\ v{:}(\rho \to \rho) \cap (\omega \to \rho) \vdash \Theta\Theta : (\rho \to \rho) \cap (\omega \to \rho) \to \rho \end{array} \qquad \begin{array}{c} D_6 \\ v{:}(\rho \to \rho) \cap (\omega \to \rho) \vdash v : (\rho \to \rho) \cap (\omega \to \rho) \end{array}}{v{:}(\rho \to \rho) \cap (\omega \to \rho) \vdash \Theta\Theta v : \rho}\,(\to E)$$

Notice that $\Theta\Theta v$ is not strongly normalizable since

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

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 normalizable.

Example A.2 As argued in Example 6.3, D'_1 and D'_2, the $\vdash_\omega$-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 normalizable, as argued above.

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

Example A.3 Take $\Theta, D_1, \ldots, D_7$ 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

$$\dfrac{\dfrac{\begin{array}{c} D_4 \\ B \vdash \Theta : \alpha \end{array} \qquad \dfrac{\begin{array}{c} D_2 \\ B \vdash \Theta : \tau \to (\omega \to \rho) \to \rho \end{array} \quad \begin{array}{c} D_1 \\ B \vdash \Theta : \tau \end{array}}{B \vdash \Theta : (\tau \to (\omega \to \rho) \to \rho) \cap \tau}\,(\cap I)}{B \vdash \Theta\Theta : (\rho \to \rho) \cap (\omega \to \rho) \to \rho}\,(\to E) \qquad \begin{array}{c} D_6 \\ B \vdash v : (\rho \to \rho) \cap (\omega \to \rho) \end{array}}{B \vdash \Theta\Theta v : \rho}\,(\to E)$$

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

$$\mathrm{D}_4: \quad \dfrac{\dfrac{\dfrac{}{B' \vdash y:\rho\to\rho} \quad \dfrac{\dfrac{\dfrac{}{B' \vdash x:\tau\to(\omega\to\rho)\to\rho} \quad \dfrac{}{B' \vdash x:\tau}}{B' \vdash xx:(\omega\to\rho)\to\rho} \quad \dfrac{}{B' \vdash y:\omega\to\rho}}{B' \vdash xxy:\rho}}{\dfrac{B' \vdash y(xxy):\rho}{\dfrac{B'\setminus y \vdash \lambda y.y(xxy):(\rho\to\rho)\cap(\omega\to\rho)\to\rho}{B \vdash \lambda xy.y(xxy):\alpha}\;(\to I)}} \quad \dfrac{\begin{array}{c}\mathrm{D}_2\\ B\vdash \Theta:\tau\to(\omega\to\rho)\to\rho\end{array} \quad \begin{array}{c}\mathrm{D}_1\\ B\vdash\Theta:\tau\end{array}}{B \vdash \Theta:(\tau\to(\omega\to\rho)\to\rho)\cap\tau}}{B \vdash \Theta\Theta:(\rho\to\rho)\cap(\omega\to\rho)\to\rho}\;(\to E)$$

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

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B''\vdash y:\rho\to\rho} \quad \dfrac{\dfrac{\begin{array}{c}\mathrm{D}_2\\ B''\vdash\Theta:\tau\to(\omega\to\rho)\to\rho\end{array} \quad \begin{array}{c}\mathrm{D}_1\\ B''\vdash\Theta:\tau\end{array}}{B''\vdash\Theta\Theta:(\omega\to\rho)\to\rho}\;(\to E) \quad \dfrac{}{B''\vdash y:\omega\to\rho}}{B''\vdash\Theta\Theta y:\rho}}{B''\vdash y(\Theta\Theta y):\rho}}{B\vdash\lambda y.y(\Theta\Theta y):(\rho\to\rho)\cap(\omega\to\rho)\to\rho}\;(\to I) \quad \begin{array}{c}\mathrm{D}_6\\ B\vdash v:(\rho\to\rho)\cap(\omega\to\rho)\end{array}}{B\vdash(\lambda y.y(\Theta\Theta y))v:\rho}\;(\to E)$$

Now D_8 has *two* redexes (notice that D_2 finishes with rule $(\to I)$); contracting the outermost distributes (the two subderivations of) D_6 and creates

$$\mathrm{D}_9: \quad \dfrac{\dfrac{}{B\vdash v:\rho\to\rho} \quad \dfrac{\dfrac{\begin{array}{c}\mathrm{D}_2\\ B\vdash\Theta:\tau\to(\omega\to\rho)\to\rho\end{array} \quad \begin{array}{c}\mathrm{D}_1\\ B\vdash\Theta:\tau\end{array}}{B\vdash\Theta\Theta:(\omega\to\rho)\to\rho}\;(\to E) \quad \dfrac{}{B\vdash v:\omega\to\rho}}{B\vdash\Theta\Theta v:\rho}}{B\vdash v(\Theta\Theta v):\rho}$$

As illustrated by Example 3.4, contracting the remaining redex of D_9 creates

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

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

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

This derivation now is in normal form; again, the term $v(v(\Theta\Theta v))$ is not. On the other hand, contracting first the innermost redex of D_8 creates D_{12}:

$$\dfrac{\dfrac{}{B''\vdash y:\rho\to\rho} \quad \dfrac{\dfrac{\dfrac{\dfrac{}{B'', z{:}\omega\to\rho\vdash z:\omega\to\rho} \quad \dfrac{}{B'', z{:}\omega\to\rho\vdash \Theta\Theta z:\omega}}{B'', z{:}\omega\to\rho\vdash z(\Theta\Theta z):\rho}}{B''\vdash\lambda z.z(\Theta\Theta z):(\omega\to\rho)\to\rho}\,(\to I) \quad \dfrac{}{B''\vdash y:\omega\to\rho}}{B''\vdash(\lambda z.z(\Theta\Theta z))y:\rho}\,(\to E)}{\dfrac{B''\vdash y((\lambda z.z(\Theta\Theta z))y):\rho}{\dfrac{B\vdash\lambda y.y((\lambda z.z(\Theta\Theta z))y):(\rho\to\rho)\cap(\omega\to\rho)\to\rho \quad \begin{array}{c}\mathrm{D}_6\\ B\vdash v:(\rho\to\rho)\cap(\omega\to\rho)\end{array}}{B\vdash(\lambda y.y((\lambda z.z(\Theta\Theta z))y))v:\rho}\,(\to E)}\,(\to I)}$$

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

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B''\vdash y:\rho\to\rho} \quad \dfrac{\dfrac{}{B''\vdash y:\omega\to\rho} \quad \dfrac{}{B''\vdash\Theta\Theta y:\omega}}{B''\vdash y(\Theta\Theta y):\rho}}{B''\vdash y(y(\Theta\Theta y)):\rho}}{B\vdash\lambda y.y(y(\Theta\Theta y)):(\rho\to\rho)\cap(\omega\to\rho)\to\rho}\,(\to I) \quad \begin{array}{c}\mathrm{D}_6\\ B\vdash v:(\rho\to\rho)\cap(\omega\to\rho)\end{array}}{B\vdash(\lambda y.y(y(\Theta\Theta y)))v:\rho}\,(\to E)$$

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

References

[1] Barendregt, H. P., *The Lambda Calculus. Its Syntax and Semantics*, revised edition, vol. 103 of *Studies in Logic and the Foundations of Mathematics*, North-Holland Publishing Co., Amsterdam, 1984. Zbl 0551.03007. MR 86a:03012. 37, 48

[2] Barendregt, H., M. Coppo, and M. Dezani-Ciancaglini, "A filter lambda model and the completeness of type assignment," *The Journal of Symbolic Logic*, vol. 48 (1983), pp. 931–40. Zbl 0545.03004. MR 85j:03014. 35, 37, 38, 39

[3] Coppo, M., and M. Dezani-Ciancaglini, "An extension of the basic functionality theory for the λ-calculus," *Notre Dame Journal of Formal Logic*, vol. 21 (1980), pp. 685–93. Zbl 0423.03010. MR 82d:03023. 35

[4] Coppo, M., M. Dezani-Ciancaglini, and M. Zacchi, "Type theories, normal forms, and D_∞-lambda-models," *Information and Computation*, vol. 72 (1987), pp. 85–116. Zbl 0645.03011. MR 88m:03025. 36

[5] Curry, H. B., "Functionality in combinatory logic," *Proceedings of the National Academy of Science USA*, vol. 20 (1934), pp. 584–90. Zbl 0010.24201. 36

[6] Curry, H. B., and R. Feys, *Combinatory Logic. Vol. I*, Studies in Logic and the Foundations of Mathematics, North-Holland Publishing Co., Amsterdam, 1958. Zbl 0081.24104. MR 20:817. 36

[7] Dezani-Ciancaglini, M., F. Honsell, and Y. Motohama, "Approximation theorems for intersection type systems. Type and set theoretical foundations of computation," *Journal of Logic and Computation*, vol. 11 (2001), pp. 395–417. Zbl 0984.03013. MR 2003d:03016. 37

[8] Dezani-Ciancaglini, M., and I. Margaria, "A characterization of F-complete type assignments," *Theoretical Computer Science*, vol. 45 (1986), pp. 121–57. Zbl 0619.03014. MR 88k:68058. 36

[9] Dezani-Ciancaglini, M., R. K. Meyer, and Y. Motohama, "The semantics of entailment omega," *Notre Dame Journal of Formal Logic*, vol. 43 (2003), pp. 129–45. Zbl 02067615. MR 2032579. 35

[10] Dougherty, D., and P. Lescanne, "Reductions, intersection types, and explicit substitutions," pp. 121–35 in *Typed Lambda Calculi and Applications (Kraków, 2001)*, vol. 2044 of *Lecture Notes in Computer Science*, Springer, Berlin, 2001. Zbl 0981.68024. MR 1890269. 36

[11] Ghilezan, S., "Strong normalization and typability with intersection types," *Notre Dame Journal of Formal Logic*, vol. 37 (1996), pp. 44–52. Zbl 0859.03007. MR 97h:03018. 36

[12] Girard, J.-Y., P. Taylor, and Y. Lafont, *Proofs and Types*, vol. 7 of *Cambridge Tracts in Theoretical Computer Science*, Cambridge University Press, Cambridge, 1989. Zbl 0671.68002. MR 90g:03013. 36, 46

[13] Krivine, J.-L., *Lambda-Calcul. Types et Modèles*, Études et Recherches en Informatique. Masson, Paris, 1990. Zbl 0697.03004. MR 93i:03019. 36, 37

[14] Pottinger, G., "A type assignment for the strongly normalizable λ-terms," pp. 561–77 in *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, Academic Press, London, 1980. MR 82j:03014. 36

[15] Retoré, C., "A note on intersection types," INRIA Rapport de recherche 2431, INRIA, France, 1994. 37

[16] Ronchi Della Rocca, S., and B. Venneri, "Principal type schemes for an extended type theory," *Theoretical Computer Science*, vol. 28 (1984), pp. 151–69. Zbl 0535.03007. MR 86d:03017. 37, 52

[17] Ronchi Della Rocca, S., "Principal type scheme and unification for intersection type discipline," *Theoretical Computer Science*, vol. 59 (1988), pp. 181–209. International Joint Conference on Theory and Practice of Software Development (Pisa, 1987). Zbl 0656.68022. MR 90k:03014. 36

[18] Tait, W. W., "Intensional interpretations of functionals of finite type. I," *The Journal of Symbolic Logic*, vol. 32 (1967), pp. 198–212. Zbl 0174.01202. MR 36:2483. 36, 46

[19] van Bakel, S., "Complete restrictions of the intersection type discipline," *Theoretical Computer Science*, vol. 102 (1992), pp. 135–63. Zbl 0762.03006. MR 94b:03025. 35, 36, 37, 38, 41, 48, 53, 55, 56

[20] van Bakel, S., "Intersection type assignment systems," *Theoretical Computer Science*, vol. 151 (1995), pp. 385–435. Thirteenth Conference on Foundations of Software Technology and Theoretical Computer Science (Bombay, 1993). Zbl 0871.68031. MR 96m:68102. 36, 37, 38, 39, 48, 52, 53, 54, 55

[21] van Bakel, S., "Strongly normalising cut-elimination with strict intersection types (extended abstract)," *Proceedings of Intersection Types and Related Systems (ITRS '02)*, vol. 70 of *Electronic Notes in Theoretical Computer Science*, Elsevier, Amsterdam, 2003. http://www.elsevier.nl/locate/entcs/volume70.html. 37

[22] van Bakel, S., F. Barbanera, M. Dezani-Ciancaglini, and F.-J. de Vries, "Intersection types for λ-trees," *Theoretical Computer Science*, vol. 272 (2002), pp. 3–40. Theories of types and proofs (Tokyo, 1997). Zbl 0984.68033. MR 2002m:03015. 36

[23] van Bakel, S., and M. Dezani-Ciancaglini, "Characterising strong normalisation for explicit substitutions," pp. 356–70 in *LATIN 2002: Theoretical Informatics (Cancun)*, vol. 2286 of *Lecture Notes in Computer Science*, Springer, Berlin, 2002. MR 1966135. 36

[24] van Bakel, S., and M. Fernández, "Approximation and normalization results for typeable term rewriting systems," pp. 17–36 in *Higher-order Algebra, Logic, and Term Rewriting (Paderborn, 1995)*, vol. 1074 of *Lecture Notes in Computer Science*, Springer, Berlin, 1996. Zbl 1019.68059. MR 97f:68084. 36

[25] van Bakel, S., and M. Fernández, "Normalization results for typeable rewrite systems," *Information and Computation*, vol. 133 (1997), pp. 73–116. Zbl 0877.68072. MR 98j:68072. 36

[26] van Bakel, S., and M. Fernández, "Normalization, approximation, and semantics for combinator systems," *Theoretical Computer Science*, vol. 290 (2003), pp. 975–1019. Zbl 1019.68059. MR 2003j:03023. 36

[27] Wadsworth, C. P., "The relation between computational and denotational properties for Scott's D_∞-models of the lambda-calculus," *SIAM Journal of Computing*, vol. 5 (1976), pp. 488–521. Zbl 0346.02013. MR 58:21493. 37, 48

Acknowledgments

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.

Department of Computing
Imperial College
180 Queen's Gate
London SW7 2BZ UNITED KINGDOM
svb@doc.ic.ac.uk
http://www.doc.ic.ac.uk/~svb