Strong Normalization from Weak Normalization in Typed λ-Calculi

Morten Heine Sørensen

Faculty of Mathematics & Informatics
Catholic University of Nijmegen
and
Department of Computer Science
University of Copenhagen

Abstract

For some typed λ-calculi it is easier to prove weak normalization than strong normalization. Techniques to infer the latter from the former have been invented over the last twenty years by Nederpelt, Klop, Khasidashvili, Karr, de Groote, and Kfoury and Wells. However, these techniques infer strong normalization of one notion of reduction from weak normalization of a *more complicated* notion of reduction.

This paper presents a new technique to infer strong normalization of a notion of reduction in a typed λ-calculus from weak normalization of the *same* notion of reduction. The technique is demonstrated to work on some well-known systems including second-order λ-calculus and the system of positive, recursive types. It gives hope for a positive answer to the Barendregt-Geuvers conjecture stating that every pure type system which is weakly normalizing is also strongly normalizing.

The paper also analyzes the relationship between the techniques mentioned above, and reviews, in less detail, other techniques for proving strong normalization.

1. Introduction

One of the most important questions concerning a notion of reduction in a typed λ-calculus is whether it satisfies *weak* and *strong normalization.*[1] The former means that from every term there is at least one finite reduction sequence ending in a normal form; the latter means that there is no term with an infinite reduction sequence. The latter property trivially implies the former, but the converse is not obvious when it holds.

[1]Reduction on terms in typed λ-calculi is closely related to reduction on derivations in natural deduction logics via the Curry-Howard isomorphism [12, 25]. This will be implicit in the rest of the paper.

The classical proof of strong normalization for β-reduction in simply typed λ-calculus is due to Tait [67]. It was generalized to second-order typed λ-calculus by Girard [20], and subsequently simplified by Tait [68]. It has since been generalized to a variety of λ-calculi, see [3, 16, 21, 24, 43, 69].

For notions of reduction in some typed λ-calculi there is a technique to prove weak normalization that is simpler than the Tait-Girard technique to prove strong normalization. For instance, Turing [17] proves weak normalization for β-reduction in simply typed λ-calculus by giving an explicit measure which decreases in every step of a certain β-reduction sequence. Prawitz [55] independently uses the same technique to prove weak normalization for reduction of natural deduction derivations in predicate logic.

Nederpelt [50], Klop [41], Khasidashvili [38], Karr [32], de Groote [14], and Kfoury and Wells [36] have invented techniques to infer strong normalization from weak normalization. However, these techniques all infer strong normalization of one notion of reduction from weak normalization of a *more complicated* notion of reduction.

This has the undesirable consequence that, even if one knows that a notion of reduction is weakly normalizing, one has to *redo* the weak normalization proof for the complicated notion of reduction to conclude strong normalization for the original notion of reduction. This is a non-trivial process—see [37] for comments on two such proofs—which involves very different techniques for different calculi. For instance, for β-reduction in simply typed λ-calculus one can extend the Turing-Prawitz weak normalization proof to the complicated notion of reduction, but for second-order typed λ-calculus one must use some kind of reducibility predicate. A technique to uniformly infer strong normalization for one notion of reduction from weak normalization of *the same* notion of reduction would be better.

Another interest in such a technique stems from a conjecture, presented by Barendregt at *Typed Lambda-Calculus and Applications, Edinburgh 1995*, stating that for every pure type system [3] weak normalization of β-reduction implies strong normalization of β-reduction. The conjecture has also been mentioned by Geuvers [19], and, in a less concrete form, by Klop.

This paper extends Klop's technique to infer strong normalization of one notion of reduction from weak normalization of the same notion of reduction. The paper does not give an answer to the conjecture, but it does suggest one possible approach to an affirmative answer.

Section 2 presents Klop's technique. Section 3 analyzes the relationship to the similar techniques by Nederpelt and others. Section 3 briefly mentions other techniques for proving strong normalization. Section 5 presents our extension of Klop's technique. Section 6 shows that the specific extension is a special case of a more general construction. The versatility of our approach is demonstrated by application to some typed λ-calculi in Section 7 and 8.

1.1. Preliminaries

The following is explained in more detail in [2].

1.1. Notation. Λ_K is the set of type-free λ-terms. Some example terms are $\mathbf{K} \equiv \lambda xy.x$, $\mathbf{I} \equiv \lambda x.x$, $\omega \equiv \lambda x.x\,x$, and $\Omega \equiv \omega\,\omega$. $M \subseteq N$ means that M is a subterm of N. $\mathrm{FV}(M)$ is the set of free variables in M. Λ_I is the set of all λ-terms where for every subterm $\lambda x.M$, $x \in \mathrm{FV}(M)$. Familiarity is assumed with the variable convention, substitution, and notions of reduction. By R_1R_2 we denote the union of two notions of reduction R_1 and R_2. For a notion of reduction R, $\to_R$ is the compatible closure, $\twoheadrightarrow_R$ is the compatible, reflexive, transitive closure, $\twoheadrightarrow_R^+$ is the compatible, transitive closure, and $=_R$ is the transitive, reflexive, symmetric, compatible closure.

In the remainder of this section R denotes an arbitrary notion of reduction on Λ_K.

1.2. Definition. A finite or infinite sequence

$$M_0 \to_R M_1 \to_R \dots$$

is called an *R-reduction path* from M_0. We say that M_0 *has* this R-reduction path. If the sequence is finite it *ends* in the last term M_n and has *length* n.

1.3. Definition. Define the following subsets of Λ_K:

$$\begin{array}{ll} \infty_R & = \{M \mid M \text{ has an infinite } R\text{-reduction path}\} \\ \mathrm{NF}_R & = \{M \mid M \text{ has no } R\text{-reduction path of length 1 or more}\} \\ \mathrm{SN}_R & = \{M \mid M \text{ has no infinite } R\text{-reduction path}\} \\ \mathrm{WN}_R & = \{M \mid M \text{ has a finite } R\text{-reduction path ending in an } N \in \mathrm{NF}_R\} \\ \mathrm{CR}_R & = \{M \mid \text{for all } L, N, \text{ if } L \;{}_R{\twoheadleftarrow}\, M \twoheadrightarrow_R N \text{ then } L \twoheadrightarrow_R K \;{}_R{\twoheadleftarrow}\, N \text{ for a } K\} \end{array}$$

1.4. Terminology. The elements of NF_R, SN_R, and WN_R are *R-normal forms*, *R-strongly normalizing*, and *R-weakly normalizing*, respectively. We sometimes write, e.g., $\mathrm{SN}_R(M)$ instead of $M \in \mathrm{SN}_R$. We also write, e.g., SN_R to state that, for all $M \in \Lambda_K$, $M \in \mathrm{SN}_R$. We also use the above sets for notions of reduction on other sets than Λ_K with the necessary changes.

1.5. Definition. For $M \in \mathrm{SN}_R \cap \mathrm{CR}_R$, $\mathrm{nf}_R(M)$ is the $N \in \mathrm{NF}_R$ with $M \twoheadrightarrow_R N$.

2. Klop's technique

This section presents Klop's technique [41, I.8] to infer strong normalization from weak normalization. Klop uses it to prove strong normalization of β-reduction in simply typed λ-calculus and in Levy's and Hyland-Wadsworth's labeled calculi; finiteness of developments follows as a special case. We present the technique in an untyped, unlabeled setting.

The first subsection sketches the technique in a style which will also be used for the related techniques in Section 4. The second subsection proves a result that will be used in our extension in Section 5.

2.1. The idea: non-erasing reductions

2.1. DEFINITION.

(i) Let Λ_K^π be the set defined by: $M ::= x \mid \lambda x.M \mid M_1\, M_2 \mid [M_1, M_2]$.

(ii) Let Λ_I^π be the set $\{M \in \Lambda_K^\pi \mid \lambda x.P \subseteq M \ \Rightarrow\ x \in \mathrm{FV}(P)\}$.

(iii) Define notions of reduction π, β, κ on Λ_K^π by:

$$\begin{array}{lll} [P,Q]\, R & \pi & [P\, R, Q] \\ (\lambda x.P)\, Q & \beta & P\{x := Q\} \\ [P,Q] & \kappa & P \end{array}$$

(iv) Define $\iota : \Lambda_K \to \Lambda_I^\pi$ by:

$$\begin{array}{lll} \iota(x) & = & x \\ \iota(\lambda x.P) & = & \lambda x.[\iota(P), x] \\ \iota(P\, Q) & = & \iota(P)\, \iota(Q) \end{array}$$

The conservation theorem for Λ_I states for $M \in \Lambda_I$ that $M \in \mathrm{WN}_\beta$ implies $M \in \mathrm{SN}_\beta$. This fails for terms in Λ_K, as the term $\mathbf{K}\,\mathbf{I}\,\Omega$ shows, because reduction in Λ_K can erase terms, and parts of terms, with infinite reductions. To obtain a similar result for Λ_K, Klop considers $\iota(M)$ from which every β-reduction $(\lambda x.[P,x])\, Q \to_\beta [P\{x := Q\}, Q]$ makes a copy of the argument. Indeed, one can show that $\iota(M) \in \mathrm{WN}_\beta$ implies $\iota(M) \in \mathrm{SN}_\beta$. The hope is that $\iota(M) \in \mathrm{SN}_\beta$, in turn, implies $M \in \mathrm{SN}_\beta$. However, this does not hold. For example, $\iota(\mathbf{I}\,\omega\,\omega) \in \mathrm{SN}_\beta$, since the only reduction path from this term is

$$\iota(\mathbf{I}\,\omega\,\omega) \equiv (\lambda x.[x,x])\, \iota(\omega)\, \iota(\omega) \to_\beta [\iota(\omega), \iota(\omega)]\, \iota(\omega) \in \mathrm{NF}_\beta$$

However, $\mathbf{I}\,\omega\,\omega \notin \mathrm{SN}_\beta$, since

$$\mathbf{I}\,\omega\,\omega \to_\beta \omega\,\omega \to_\beta \omega\,\omega \to_\beta \ldots$$

The problem is that the pairing operator may block reductions in $\iota(M)$ which take place in M. Therefore Klop adopts the π-rule which moves a term across a copy.

2.2. THEOREM (Klop [41]). *For all $M \in \Lambda_K$,*

$$\iota(M) \in \mathrm{WN}_{\beta\pi} \ \Rightarrow\ M \in \mathrm{SN}_\beta$$

2.3. Remark. Klop's proof of Theorem 2.2 is in two steps:

$$\iota(M) \in \mathrm{WN}_{\beta\pi} \Rightarrow \iota(M) \in \mathrm{SN}_{\beta\pi} \Rightarrow M \in \mathrm{SN}_{\beta} \tag{1}$$

The first implication is a special case of Klop's conservation theorem [41] for *definable extensions* of Λ_I, and the second one is proved by the implications:

$$\iota(M) \in \mathrm{SN}_{\beta\pi} \Rightarrow \iota(M) \in \mathrm{SN}_{\beta\pi\kappa} \Rightarrow \iota(M) \in \mathrm{SN}_{\beta\kappa} \Rightarrow M \in \mathrm{SN}_{\beta} \tag{2}$$

Here the first implication follows from the fact that in an infinite $\beta\kappa\pi$-reduction one can postpone κ-reductions to get an infinite $\beta\pi$-reduction. The second implication is obvious, and the third follows from $\iota(M) \twoheadrightarrow_\kappa M$.

2.2. Proof of part of Klop's result

In Section 5 our extension uses the second implication of (1), which we therefore prove now. The proof follows the structure of (2).

2.4. Lemma (Postponement of κ across $\beta\pi$). *For all $M, N, O \in \Lambda^\pi_K$:*

$$\begin{tikzcd} M \arrow[r, "\rightarrow_\kappa"'] \arrow[d, dotted, "\twoheadrightarrow^+_{\beta\pi}"'] & N \arrow[d, "\rightarrow_{\beta\pi}"] \\ K \arrow[r, dotted, "\twoheadrightarrow_\kappa"'] & O \end{tikzcd}$$

Proof. First show that, if $M \rightarrow_\kappa N$ then

$$M\{x := L\} \rightarrow_\kappa N\{x := L\} \tag{3}$$

and

$$L\{x := M\} \twoheadrightarrow_\kappa L\{x := N\} \tag{4}$$

by induction on $M \rightarrow_\kappa N$ and L, respectively. Then proceed by induction on $M \rightarrow_\kappa N$, splitting into cases according to how $M \rightarrow_\kappa N \rightarrow_{\beta\pi} O$:

1. $M \equiv x\, P_0 \dots P_n$, where $n > 0$. Then $N \equiv x\, Q_0 \dots Q_n$, where $P_i \rightarrow_\kappa Q_i$ for one i, and $P_j \equiv Q_j$ for all $j \neq i$. Then $O \equiv x\, R_0 \dots R_n$, where $Q_l \rightarrow_{\beta\pi} R_l$ for one l, and $Q_m \equiv R_m$ for all $m \neq l$.

   1.1. $i = l$. Then $P_i \rightarrow_\kappa Q_i \rightarrow_{\beta\pi} R_i$. Then, by the induction hypothesis, $P_i \twoheadrightarrow^+_{\beta\pi} K \twoheadrightarrow_\kappa R_i$, for some K. Then

$$\begin{array}{rll} x\, P_0 \dots P_n & \twoheadrightarrow^+_{\beta\pi} & x\, Q_0 \dots Q_{i-1}\, K\, Q_{i+1} \dots Q_n \\ & \twoheadrightarrow_\kappa & x\, R_0 \dots R_n \end{array}$$

   1.2. $i \neq l$. Then $P_i \rightarrow_\kappa Q_i \equiv R_i$ and $P_l \equiv Q_l \rightarrow_{\beta\pi} R_l$. Then

$$\begin{array}{rll} x\, P_0 \dots P_n & \rightarrow_{\beta\pi} & x\, Q_0 \dots Q_{l-1}\, R_l\, Q_{l+1} \dots Q_n \\ & \rightarrow_\kappa & x\, R_0 \dots R_n \end{array}$$

2. $M \equiv (\lambda x.P_0)\, P_1 \dots P_n$, where $n \geq 0$. Then $N \equiv (\lambda x.Q_0)\, Q_1 \dots Q_n$, where $P_i \to_\kappa Q_i$ for one i, and $P_j \equiv Q_j$ for all $j \neq i$.
   - 2.1. $O \equiv (\lambda x.R_0)\, R_1 \dots R_n$, where $Q_l \to_\kappa R_l$ for one l, and $P_m \equiv Q_m$ for all $m \neq l$. Then proceed as in Case 1.
   - 2.2. $O \equiv Q_0\{x := Q_1\}\, Q_2 \dots Q_n$. Then, by (3)-(4),

$$\begin{array}{rll} (\lambda x.P_0)\, P_1 \dots P_n & \to_{\beta\pi} & P_0\{x := P_1\}\, P_2 \dots P_n \\ & \twoheadrightarrow_\kappa & Q_0\{x := Q_1\}\, Q_2 \dots Q_n \end{array}$$

3. $M \equiv [P_1, P_0]\, P_2 \dots P_n$, where $n > 0$.
   - 3.1. $N \equiv [Q_1, Q_0]\, Q_2 \dots Q_n$, where $P_i \to_\kappa Q_i$ for one i, and $P_j \equiv Q_j$ for all $j \neq i$. Then proceed as follows.
     - $O \equiv [R_1, R_0]\, R_2 \dots R_n$, where $Q_l \to_{\beta\pi} R_l$ for one l, and $Q_m \equiv R_m$ for all $m \neq l$. Then proceed as in Case 1.
     - $O \equiv [Q_1\, Q_2, Q_0]\, Q_3 \dots Q_n$. Then

$$\begin{array}{rll} [P_1, P_0]\, P_2 \dots P_n & \to_\pi & [P_1\, P_2, P_0]\, P_3 \dots P_n \\ & \to_\kappa & [Q_1\, Q_2, Q_0]\, Q_3 \dots Q_n \end{array}$$

   - 3.2. $N \equiv P_1\, P_2 \dots P_n$. Since $N \to_{\beta\pi} O$,

$$\begin{array}{rll} [P_1, P_0]\, P_2 \dots P_n & \twoheadrightarrow_\pi & [P_1 \dots P_n, P_0] \\ & \to_{\beta\pi} & [O, P_0] \\ & \to_\kappa & O \end{array}$$

This exhausts all possibilities. □

2.5. Lemma. *For all $M \in \Lambda_K^\pi$,*

$$M \in \mathrm{SN}_{\beta\pi} \;\Rightarrow\; M \in \mathrm{SN}_{\beta\pi\kappa}$$

Proof. Assume $\infty_{\beta\pi\kappa}(M)$. We must prove $\infty_{\beta\pi}(M)$.

We first show by induction on n that, for all $n \geq 0$, there is an n-tuple $\sigma_n = (M_0, M_1, \dots, M_{n-1})$ and $L_0, L_1, \dots$ such that

$$M_0 \to_{\beta\pi} M_1 \to_{\beta\pi} \dots \to_{\beta\pi} M_{n-1} \to_{\beta\pi\kappa} L_0 \to_{\beta\pi\kappa} L_1 \to_{\beta\pi\kappa} \dots$$

Put $\sigma_0 = (M)$. For $n = m + 1$ we assume:

$$M_0 \to_{\beta\pi} M_1 \to_{\beta\pi} \dots \to_{\beta\pi} M_{m-1} \to_{\beta\pi\kappa} L_0 \to_{\beta\pi\kappa} L_1 \to_{\beta\pi\kappa} \dots$$

Since κ-reductions strictly decrease term size, there is a smallest $k \geq m-1$ such that $M_k \to_{\beta\pi} M_{k+1}$. Now use Lemma 2.4 $k - (m-1)$ times to arrive at a sequence in which the n first elements constitute σ_n.

Now let N_i be the i'th element of σ_i. Then clearly

$$M \equiv N_0 \to_{\beta\pi} N_1 \to_{\beta\pi} N_2 \to_{\beta\pi} \dots$$

as required. □

2.6. LEMMA. *For all $M \in \Lambda^{\pi}_{K}$,*

$$\iota(M) \in \mathrm{SN}_{\beta\kappa} \Rightarrow M \in \mathrm{SN}_{\beta}$$

PROOF. By induction on M prove $\iota(M) \twoheadrightarrow_{\kappa} M$. This gives the lemma. □

2.7. MAIN LEMMA (Klop [41]). *For all $M \in \Lambda_K$,*

$$\iota(M) \in \mathrm{SN}_{\beta\pi} \Rightarrow M \in \mathrm{SN}_{\beta}$$

PROOF. By Lemmas 2.5 and 2.6. □

3. Variations on Klop's technique

Klop's technique [41] was inspired by Nederpelt's [50] technique, and is also related to the later techniques by Khasidashvili [38], Karr [32], de Groote [14], and Kfoury and Wells [36]. The similarity between the different approaches is sometimes blurred because each technique is described in a particular context in terms of labeled or typed terms.

This section reviews these techniques in an untyped, unlabeled setting. We begin with de Groote's technique since it resembles Klop's the most. The remaining techniques are then described in less detail. For more on the relationship between Klop's and Nederpelt's technique, see [41, II.4]. For more on the relationship between de Groote's and Kfoury and Wells' technique, see [37]. The notions of reduction discussed in this section have been considered in a number of other contexts [1, 33, 34, 35, 48, 57, 60, 71], see [37].

3.1. The technique by de Groote

This subsection presents de Groote's [14] technique to reduce strong normalization for the systems in the λ-cube [3] to weak normalization of related systems. In particular, adopting a version of the Turing-Prawitz proof, he proves strong normalization of β-reduction in the simply typed λ-calculus.

3.1. DEFINITION. Let $\beta_I, \beta_K, \beta_S$ be the notions of reduction on Λ_K:

$$\begin{array}{llll} (\lambda x.P)\, Q & \beta_I & P\{x := Q\} & \text{if } x \in \mathrm{FV}(P) \\ (\lambda y.P)\, Q & \beta_K & P & \text{if } y \notin \mathrm{FV}(P) \\ (\lambda y.P)\, Q\, R & \beta_S & (\lambda y.P\, R)\, Q & \text{if } y \notin \mathrm{FV}(P) \end{array}$$

A generalization of the conservation theorem for Λ_I states for $M \in \Lambda_K$ that $M \in \mathrm{WN}_{\beta_I}$ implies $M \in \mathrm{SN}_{\beta_I}$. If β_K-redexes could be postponed across β_I-redexes, $M \in \mathrm{SN}_{\beta_I}$ would, in turn, imply $M \in \mathrm{SN}_{\beta_I\beta_K}$, i.e. $M \in \mathrm{SN}_{\beta}$. This would give a technique to infer β-strong normalization from β_I-weak normalization. Unfortunately, postponement of β_K-redexes is not in general possible; a β_K-reduction may create a β_I-redex:

$$(\lambda y.\lambda x.P)\,Q\,R \to_{\beta_K} (\lambda x.P)\,R \qquad y \notin \mathrm{FV}(P), x \in \mathrm{FV}(P)$$

The notion of reduction β_S is used to sidestep this problem.

3.2. THEOREM (de Groote [14]). *For all* $M \in \Lambda_K$*,*

$$M \in \mathrm{WN}_{\beta_S\beta_I} \;\Rightarrow\; M \in \mathrm{SN}_\beta$$

3.3. REMARK. The proof of Theorem 3.2 by de Groote is in two parts:

$$M \in \mathrm{WN}_{\beta_I\beta_S} \;\Rightarrow\; M \in \mathrm{SN}_{\beta_I\beta_S} \;\Rightarrow\; M \in \mathrm{SN}_\beta \tag{5}$$

The *first part* of (5) is proved by a technique originally due to Nederpelt. One shows that $\mathrm{CR}_{\beta_S\beta_I}$ and that a certain measure $|\bullet|$ is strictly *increased* by $\beta_I\beta_S$-reductions ($\mathrm{INC}_{\beta_I\beta_S}$ for short). If $M \in \mathrm{WN}_{\beta_I\beta_S}$, i.e.,

$$M \twoheadrightarrow_{\beta_I\beta_S} N \in \mathrm{NF}_{\beta_I\beta_S}$$

and M also had an infinite $\beta_I\beta_S$-reduction

$$M \equiv M_0 \to_{\beta_I\beta_S} M_1 \to_{\beta_I\beta_S} \cdots$$

then $|N| < |M_k|$ for some k, by $\mathrm{INC}_{\beta_I\beta_S}$. By $\mathrm{CR}_{\beta_I\beta_S}$, $M_k \twoheadrightarrow_{\beta_I\beta_S} N$ and hence by $\mathrm{INC}_{\beta_I\beta_S}$ also $|M_k| \leq |N|$, a contradiction. In short:

$$\mathrm{INC}_{\beta_I\beta_S} \text{ and } \mathrm{CR}_{\beta_I\beta_S} \text{ and } M \in \mathrm{WN}_{\beta_I\beta_S} \;\Rightarrow\; M \in \mathrm{SN}_{\beta_I\beta_S} \tag{6}$$

The *second part* of (5) is proved by the implications:

$$M \in \mathrm{SN}_{\beta_I\beta_S} \;\Rightarrow\; M \in \mathrm{SN}_{\beta_I\beta_S\beta_K} \;\Rightarrow\; M \in \mathrm{SN}_{\beta_I\beta_K} \;\Rightarrow\; M \in \mathrm{SN}_\beta \tag{7}$$

Here the first implication follows from the fact that β_K-reductions can be postponed across $\beta_I\beta_S$-reductions, and the two others are trivial.

3.2. Klop versus de Groote

The reductions κ and β_S adopted by Klop and de Groote, respectively, are very similar. Whereas Klop considers reductions

$$[P, Q]\,R \to [P\,R, Q]$$

de Groote considers

$$(\lambda y.P)\,Q\,R \to (\lambda y.P\,R)\,Q$$

Reading $[P, Q]$ as $(\lambda y.P)\,Q$ with $y \notin P$, they are the same!

Indeed, let $\phi : \Lambda_I^\pi \to \Lambda_K$ be the map which replaces κ-redex $[M, N]$ by $(\lambda y.M)\, N$, $y \notin \mathrm{FV}(M)$. Then, for all $M \in \Lambda_I^\pi$, $N \in \Lambda_K^\pi$,

$$\begin{array}{lcl} M \to_\beta N & \Leftrightarrow & \phi(M) \to_{\beta_I} \phi(N) \\ M \to_\kappa N & \Leftrightarrow & \phi(M) \to_{\beta_K} \phi(N) \\ M \to_\pi N & \Leftrightarrow & \phi(M) \to_{\beta_S} \phi(N) \end{array}$$

This explains the similarity between the proof of Klop's Theorem 2.2 and the proof of de Groote's Theorem 3.2. In both cases, the overall proof consists of two implications—(1) and (5)—see Remarks 2.3 and 3.3. Klop and de Groote prove the first implication in (1) and (5) differently, but de Groote's proof can be adapted to Klop's setting. As for the second implication in (1) and (5), the proof consists in both cases of three implications—(2) and (7). The first two implications in (2) and (7) are proved the same way. The techniques only differ in the last implication: in Klop's technique one has to use the details of ι, while in de Groote's technique one uses $\beta = \beta_I \beta_K$.

3.3. Nederpelt's technique

Nederpelt [50] proves β-strong normalization of all terms in a typed λ-calculus from the Automath family [13], using a reduction to the problem of proving weak normalization. Nederpelt uses a somewhat unorthodox notation for λ-terms. For instance, $(\lambda x.P)\, Q$ is written $\{Q\}[x]P$. This notation has its advantages, but we present here the technique in more familiar terms.

Recently there has been new interest in Nederpelt's reductions [9, 28, 29, 30], and their relevance to explicit substitution calculi [27, 31].

3.4. Definition. Let C, D range over contexts, and $C[D]$ denote the result of substituting D for $[\,]$ in C. The set of *β-chains* $\mathcal{C}$ is defined by:[2]

$$\begin{array}{lcl} & & [] \in \mathcal{C} \\ C \in \mathcal{C}, N \in \Lambda_K & \Rightarrow & C[\lambda x.[]]\, N \in \mathcal{C} \\ C, D \in \mathcal{C} & \Rightarrow & C[D] \in \mathcal{C} \end{array}$$

Define the notions of reduction β_1, β_2 by:

$$\begin{array}{llll} C[\lambda x.P]\, R & \beta_1 & C[\lambda x.P\{x := R\}]\, R & \text{if } x \in \mathrm{FV}(P) \text{ and } C \in \mathcal{C} \\ C[\lambda y.P]\, R & \beta_2 & C[P] & \text{if } y \notin \mathrm{FV}(P) \text{ and } C \in \mathcal{C} \end{array}$$

The motivation for β_1 is that it allows postponement of β_2-reductions, just like β_S-reductions allow postponement of β_K-reductions. For example, if $x \in \mathrm{FV}(P)$ and $y \notin \mathrm{FV}(P)$, then

$$(\lambda y.\lambda x.P)\, Q\, R \to_{\beta_1} (\lambda y.\lambda x.P\{x := R\})\, Q\, R \to_{\beta_2} (\lambda y.P\{x := R\})\, Q$$

[2]One may think of abstraction and application in β-chains as left and right parenthesis. Counting inside-out the number of abstractions is never smaller than the number of applications, and the total number of abstractions equals the total number of applications.

In de Groote's setting this would be

$$(\lambda y.\lambda x.P)\, Q\, R \to_{\beta_S} (\lambda y.(\lambda x.P)\, R)\, Q \to_{\beta_K} (\lambda y.P\{x := R\})\, Q$$

None of β_S and β_1 is contained in the other: β_S is more general in that it does not require the object under λy to be an abstraction, and β_1 is more general in that it does not require the β-chain to have form $(\lambda y.[])\, Q$.

3.5. Theorem (Nederpelt [50]). *For all $M \in \Lambda_K$,*

$$M \in \mathrm{WN}_{\beta_1} \;\Rightarrow\; M \in \mathrm{SN}_{\beta}$$

3.6. Remark. The proof structure is as (5)-(7) in Remark 3.3 with β_1 in place of $\beta_I\beta_S$ and β_2 in place of β_K.

3.4. Karr's technique

Karr [32] studies general conditions under which additions to the simply typed λ-calculus remain strongly normalizing, and obtains as a special case strong normalization of $\beta\eta$-reduction and surjective pairs.

This in general works by reducing $\beta_I R$-strong normalization to $\beta_I R_\gamma$-strong normalization, where R_γ is a certain *conjugate* rule, derived mechanically from R.

3.7. Definition. Define the notion of reduction β_γ by:

$$C\{z := \lambda x.M\}\, R \quad \beta_\gamma \quad C\{z := M\{x := R\}\} \quad \text{if } x \in \mathrm{FV}(M) \text{ and } C \twoheadrightarrow_{\beta_K} z$$

The motivation for β_γ is that it allows postponement of β_K. For example, if $x \in \mathrm{FV}(P)$ and $y \notin \mathrm{FV}(P)$ then

$$(\lambda y.\lambda x.P)\, Q\, R \to_{\beta_\gamma} (\lambda y.P\{x := R\})\, Q$$

This shows that Karr's reduction β_γ obtains the effect of Nederpelt's β_1 (composed with β_2). Whereas Nederpelt requires that the C in $C[\lambda x.P]\, R$, be a β-chain, Karr requires that $C[z] \twoheadrightarrow_{\beta_K} z$.

3.8. Theorem (Karr [32]). *For all $M \in \Lambda_K$,*

$$M \in \mathrm{SN}_{\beta_I\beta_\gamma} \;\Rightarrow\; M \in \mathrm{SN}_{\beta}$$

3.9. Remark. The proof is as (7) in Remark 3.3 with β_γ in place of β_S.

3.5. Kfoury and Wells' technique

Kfoury and Wells [36] reduce the strong normalization problem of β-reduction in simply typed λ-calculus and the intersection type system to the weak normalization problem for related systems as follows.

3.10. Definition. Define the notion of reduction γ by:

$$(\lambda y.\lambda x.P)\, Q \quad \gamma \quad \lambda x.(\lambda y.P)\, Q$$

and let $M \to_* N \Leftrightarrow M \to_{\beta_I} M' \twoheadrightarrow_\gamma N \in \mathrm{NF}_\gamma$.

The idea behind γ again is that it facilitates postponement of β_K-reductions. For example,

$$(\lambda y.\lambda x.P)\, Q\, R \to_\gamma (\lambda x.(\lambda y.P)\, Q)\, R$$

Thus, whereas de Groote's β_S moves R to its matching λx, Kfoury and Wells' γ moves λx to its matching R.

3.11. Theorem (Kfoury and Wells [36]). *For all $M \in \Lambda_K$,*

$$\mathrm{nf}_\gamma(M) \in \mathrm{WN}_* \Rightarrow M \in \mathrm{SN}_\beta$$

3.12. Remark. Instead of proceeding as in (5)-(7) with γ in place of β_S, Kfoury and Wells approach the problem differently. Their proof shows that $*$-normal forms are β-strongly normalizing. Since $*$-reductions preserve the possibility of infinite β-reductions, any $*$-weakly normalizing term is β-strongly normalizing. The result then follows from the fact that γ-reductions preserve the possibility of infinite β-reductions.

3.6. More general techniques by Klop and Khasidashvili

Klop [41, II.4] generalizes the technique from Section 2 to *regular combinatory reduction systems* (such systems are described in the survey [42]). For any regular combinatory reduction system Σ, Klop introduces another one Σ_π such that if all terms are weakly normalizing in Σ_π then all terms are strongly normalizing in Σ. The proof is a generalization of Nederpelt's technique (5)-(7) in Remark 3.3 with Σ_π for $\beta_I\beta_S$ and Σ for β. As a corollary Klop obtains finiteness of developments for regular combinatory systems.

Khasidashvili [38] studies so-called *S-reductions*, which are equivalent to developments. He independently develops a technique similar to Klop's, and uses it to prove strong normalization of S-reductions (i.e., finiteness of developments), to effectively compute longest S-reductions, and to effectively compute the length of such reductions. He obtains similar results for other notions of reduction too.

In some more recent papers Khasidashvili formulates his technique for any orthogonal term rewrite system (OTRS) [40] and any so-called orthogonal expression reduction system (OERS) [39]. The proof of the result is, in both cases, similar to Nederpelt's. As applications he shows, for certain classes of orthogonal term rewriting systems, that strong normalization holds and that decidability of weak normalization implies decidability of strong normalization. He also obtains several theorems in the theory of perpetual reductions (see Section 4).

4. Other reductions of strong normalization

In this section we briefly describe other ways of proving strong normalization which involve interpretations of Λ_K in Λ_I or which use reductions to weak normalization: perpetual reductions and functional interpretations.

4.1. Perpetual reductions

A *perpetual* reduction strategy F computes for a type-free term an *infinite* reduction path, if one exists, and otherwise a finite reduction path to normal form. To prove that *all* reduction paths end in a normal form it thus suffices to prove that the one computed by F does so. This is similar to the techniques in Sections 2–3: instead of proving that *all* reduction paths are finite, one only needs to show that *one* reduction path is finite. The difference is that in the techniques from Sections 2–3 one may choose freely which path to prove finite, whereas in the present technique one must prove that the path computed by F is finite.

A *maximal* reduction strategy computes for a type-free term an *infinite* reduction path, if one exists, and otherwise a *longest* finite reduction path to normal form. Any maximal strategy is perpetual, but the converse is not generally true.

Barendregt et al. [4] give an effective perpetual strategy F_∞, and use it to prove the conservation theorem for Λ_K. Bergstra and Klop [7] generalize that result. Later, de Vrijer [72] proves that a version of F_∞ computes longest developments and that these are finite. Regnier [57] shows that F_∞ is maximal.

Independently, Khasidashvili [40, 39], as applications of his technique to reduce strong normalization to weak normalization (see Section 3.6), shows conservation theorems for OTRSs and OERSs. He also introduces the notion of a *limit strategy* and shows that any limit strategy is maximal, and uses this to obtain effective maximal strategies in so-called *persistent* OTRS and *strongly persistent* OERS, as well as effective longest developments (and effective length of such developments) in OTRS and OERS.

Also independently, Raamsdonk and Severi [56] give a characterization of strongly normalizing type-free λ-terms, internalizing F_∞ in a certain sense, and use this to prove that F_∞ is maximal and to simplify proofs of finiteness of developments and strong normalization for β-reduction in simply typed λ-calculus and the intersection type system.

Xi [75] uses a similar idea to give short proofs for finite developments, for some conservation theorems in Λ_I and Λ_K, and for strong normalization of β-reduction in simply typed λ-calculus.

The paper [64] gives perpetual and maximal strategies for $\beta\eta$-reduction and derives short proofs of conservation theorems for $\beta\eta$-reduction. Using a perpetual strategy, the paper [65] shows that any term with an infinite reduction path has Ω embedded in a certain sense, and that a term is strongly normalizing if all its embedded terms are weakly normalizing. However, this is not useful for proving strong normalization of typed λ-calculi, because a typable term may have non-typable terms embedded (see Section 7).

4.2. Functional interpretations

Gandy [18] interprets a term in a typed λ-calculus by a functional, whose value is an upper bound on the length of reductions from the term. Schwichtenberg [61] applies the technique to simply typed λ-calculus, and de Vrijer [73] computes the exact length of longest reduction paths in simply typed λ-calculus, and mentions that F_∞ computes these paths. The technique is generalized to higher-order rewrite systems by van de Pol [52] and adapted to other systems by van de Pol and Schwichtenberg [54].

Another technique for computing upper bounds on lengths of reductions is due to Howard [26] which was used by Schwichtenberg [62] to provide upper bounds for the length of reductions in simply typed λ-calculus, and by Springintveld [66] for the dependent system λP and the weak version $\lambda\underline{\omega}$ of higher-order typed λ-calculus.

Several of these techniques use translations from Λ_K to Λ_I. For the relation between these techniques and Tait's technique, see [53].

5. Extensions of Klop's technique

This section presents the main contribution of the paper: an extension of Klop's technique yielding a translation $[\bullet] : \Lambda_K \to \Lambda_I$ such that $[M] \in \mathrm{WN}_\beta$ implies $M \in \mathrm{SN}_\beta$. This result was independently discovered by Xi [76].

The first subsection gives the idea and the second subsection develops the details.

5.1. The idea: simulation of π

Theorem 2.2 shows for $M \in \Lambda_K$ that $M \in \mathrm{SN}_\beta$ follows from $\iota(M) \in \mathrm{WN}_{\beta\pi}$. We aim at a condition involving only β-weak normalization. The following definition and proposition suggest a natural approach.

5.1. Definition. A translation $\phi : \Lambda_I^\pi \to \Lambda_I$ *simulates* π if

$$\begin{array}{lcll} L \to_\beta K & \Rightarrow & \phi(L) \twoheadrightarrow_\beta^+ \phi(K) & (8) \\ L \to_\pi K & \Rightarrow & \phi(L) \twoheadrightarrow_\beta \phi(K) & (9) \end{array}$$

5.2. Proposition. *Assume $\phi : \Lambda_I^\pi \to \Lambda_I$ simulates π. For all $M \in \Lambda_K$,*

$$\phi(\iota(M)) \in \mathrm{WN}_\beta \Rightarrow M \in \mathrm{SN}_\beta$$

Proof. We first show that, for all $M \in \Lambda_K^\pi$, $\mathrm{SN}_\pi(M)$. Define $w : \Lambda_K^\pi \to \mathbb{N}$:

$$\begin{array}{lcl} w(x) & = & 1 \\ w(\lambda x.P) & = & w(P) \\ w(P\,Q) & = & w(P) + w(Q) \\ w([P,Q]) & = & 2w(P) + w(Q) \end{array}$$

Then prove, by induction on $M \to_\pi N$, that $M \to_\pi N \Rightarrow w(M) > w(N)$.

Now, assume $\phi(\iota(M)) \in \mathrm{WN}_\beta$. By the conservation theorem for Λ_I, $\phi(\iota(M)) \in \mathrm{SN}_\beta$. If $\iota(M)$ had an infinite $\beta\pi$-reduction path, then infinitely many of these steps were β-reductions, but then $\phi(\iota(M))$ also had an infinite β-reduction path, a contradiction. Hence $\iota(M) \in \mathrm{SN}_{\beta\pi}$. Then, by Main Lemma 2.7, $M \in \mathrm{SN}_\beta$. □

So, the problem is to find ϕ. One approach, mentioned by Klop [41, I.7], is to map pairs $[M,N] \in \Lambda_I^\pi$ into terms $P\ M\ N \in \Lambda_I$ where P is a fixed point combinator such that $P\,M\,N\,L \to_\beta P\,(M\,L)\,N$. For the present purposes this approach has the problem that, for the obvious choices of P, $\phi(\iota(M)) \notin \mathrm{WN}_\beta$. Moreover, $\phi \circ \iota$ fails to map typable terms to typable terms (see Section 7).

Fortunately another technique is available. It is well-known [10, 60] that one can simulate reductions like π by means of a *continuation passing style* (CPS) translation [58, 51]. More precisely, there is a CPS translation $\psi : \Lambda_I^\pi \to \Lambda_I^\pi$ and a *compacting* CPS translation $\phi : \Lambda_I^\pi \to \Lambda_I^\pi$ such that $\psi(M) \twoheadrightarrow_\beta \phi(M)$ and ϕ simulates π. Since a pair $[M,N]$ in the translated world has no notion of reduction associated, it is equivalent to yMN where y is some fresh variable. Using this idea one gets a translation into Λ_I instead of Λ_I^π.

This suggests the following principle.

5.3. Proposition. *Suppose $\psi, \phi : \Lambda_I^\pi \to \Lambda_I$ are such that ϕ simulates π and $\psi(M) \twoheadrightarrow_\beta \phi(M)$ for all $M \in \Lambda_I^\pi$. Then*

$$\psi(\iota(M)) \in \mathrm{WN}_\beta \Rightarrow M \in \mathrm{SN}_\beta$$

Proof. Assume that $\psi(\iota(M)) \in \mathrm{WN}_\beta$. By the Church-Rosser property, $\phi(\iota(M)) \in \mathrm{WN}_\beta$. Then, by Proposition 5.2, $M \in \mathrm{SN}_\beta$. □

5.2. Simulation by CPS translation

We now show how to simulate π by means of CPS translation.

The restrictions to Λ_K of the following two maps were first studied systematically by Plotkin [51]; see also [58].

5.4. Definition. Let y be a variable, not occurring in any other term.

(i) Define $\underline{\bullet} : \Lambda^{\pi}_{K} \to \Lambda_K$ by:

$$\begin{array}{lcl} \underline{x} & = & \lambda k.x\,k \\ \underline{\lambda x.P} & = & \lambda k.k\,\lambda x.\underline{P} \\ \underline{P\,Q} & = & \lambda k.\underline{P}\,\lambda m.m\,\underline{Q}\,k \\ \underline{[P,Q]} & = & \lambda k.y\,(\underline{P}\,k)\,\underline{Q} \end{array}$$

(ii) Define $\bullet{:}\bullet : \Lambda^{\pi}_{K} \times \Lambda_K \to \Lambda_K$ by:

$$\begin{array}{llcl} x & : K & = & x\,K \\ (\lambda x.P) & : K & = & K\,\lambda x.\underline{\underline{P}} \\ (P\,Q) & : K & = & P : (\lambda m.m\,\underline{\underline{Q}}\,K) \\ [P,Q] & : K & = & y\,(P : K)\,\underline{\underline{Q}} \end{array}$$

where $\underline{\underline{M}} = \lambda h.(M{:}h)$, for all $M \in \Lambda^{\pi}_{K}$.

The idea is to use Proposition 5.3 with $\psi(M) = \underline{M}$ and $\phi(M) = \underline{\underline{M}}$.

5.5. Lemma. *For all $M, N \in \Lambda^{\pi}_{K}$ and $K, L \in \Lambda_K$:*

(i) $k \notin \mathrm{FV}(M) \Rightarrow (M : K)\{k := L\} = M : (K\{k := L\})$.

(ii) $K \twoheadrightarrow^{+}_{\beta} L \Rightarrow M : K \twoheadrightarrow^{+}_{\beta} M : L$.

Proof. By induction on M. □

5.6. Lemma. *For all $M, N \in \Lambda^{\pi}_{K}$ and $K \in \Lambda_K$:*

$$(M{:}K)\{x := \underline{\underline{N}}\} \twoheadrightarrow_{\beta} (M\{x := N\}) : (K\{x := \underline{\underline{N}}\})$$

Proof. By induction on M. Let, for any $L \in \Lambda_K$, $L^* \equiv L\{x := \underline{\underline{N}}\}$.

1. $M \equiv x$. Then, by Lemma 5.5(i),

$$\begin{array}{lcl} (x{:}K)^* & \equiv & (x\,K)^* \\ & \equiv & \underline{\underline{N}}\,K^* \\ & \to_{\beta} & (N{:}h)\{h := K^*\} \\ & \equiv & N{:}K^* \\ & \equiv & (x\{x := N\}){:}K^* \end{array}$$

2. $M \equiv y \not\equiv x$. Then

$$\begin{array}{rcl} (y\!:\!K)^* & \equiv & (y\,K)^* \\ & \equiv & y\{x := N\}\,K^* \\ & \equiv & (y\{x := N\})\!:\!K^* \end{array}$$

3. $M \equiv \lambda y.P$. Then, by the induction hypothesis,

$$\begin{array}{rcl} ((\lambda y.P)\!:\!K)^* & \equiv & (K\,\lambda y.\underline{\underline{\underline{P}}})^* \\ & \twoheadrightarrow_\beta & K^*\,\lambda y.\underline{\underline{P\{x := N\}}} \\ & \equiv & (\lambda y.P\{x := N\})\!:\!K^* \\ & \equiv & ((\lambda y.P)\{x := N\})\!:\!K^* \end{array}$$

4. $M \equiv P\,Q$. Then, by the induction hypothesis and Lemma 5.5(ii),

$$\begin{array}{rcl} ((P\,Q)\!:\!K)^* & \equiv & (P\!:\!(\lambda m.m\,\underline{\underline{\underline{Q}}}\,K))^* \\ & \twoheadrightarrow_\beta & (P\{x := N\})\!:\!\lambda m.m\,\underline{\underline{Q\{x := N\}}}K^* \\ & \equiv & (P\{x := N\}\,Q\{x := N\})\!:\!K^* \\ & \equiv & ((P\,Q)\{x := N\})\!:\!K^* \end{array}$$

The remaining case is similar to Case 3. $\square$

5.7. Lemma. *For all $M, N \in \Lambda_K^\pi$ and $K \in \Lambda_K$:*

(i) $\underline{M} \twoheadrightarrow_\beta \underline{\underline{M}}$.

(ii) $M \to_\beta N \;\Rightarrow\; M : K \twoheadrightarrow_\beta^+ N : K$.

(iii) $M \to_\pi N \;\Rightarrow\; M : K \;\equiv\; N : K$.

Proof.

(i) Induction on M.

1. $M \equiv P\,Q$. Then, by the induction hypothesis and Lemma 5.5(i),

$$\begin{array}{rcl} \underline{P\,Q} & \equiv & \lambda k.\underline{P}\,\lambda m.m\,\underline{Q}\,k \\ & \twoheadrightarrow_\beta & \lambda k.\underline{\underline{P}}\,\lambda m.m\,\underline{\underline{Q}}\,k \\ & \to_\beta & \lambda k.(P\!:\!\lambda m.m\,\underline{\underline{Q}}\,k) \\ & \equiv & \lambda k.((P\,Q)\!:\!k) \\ & \equiv & \underline{\underline{P\,Q}} \end{array}$$

2. $M \equiv [P, Q]$. Then, by the induction hypothesis and Lemma 5.5(i),

$$\begin{array}{rcl} \underline{[P, Q]} & \equiv & \lambda k.y\,(\underline{P}\,k)\,\underline{Q} \\ & \twoheadrightarrow_\beta & \lambda k.y\,(\underline{\underline{P}}\,k)\,\underline{\underline{Q}} \\ & \to_\beta & \lambda k.y\,(P\!:\!k)\,\underline{\underline{Q}} \\ & \equiv & \lambda k.([P, Q]\!:\!k) \\ & \equiv & \underline{\underline{[P, Q]}} \end{array}$$

The remaining two cases are straight-forward.

(ii) Induction on $M \to_\beta N$.

1. $M \equiv (\lambda x.P)\,Q \to_\beta P\{x := Q\} \equiv N$. Then, by Lemma 5.5(i) and 5.6,
$$\begin{array}{rcl} M:K & \equiv & (\lambda m.m\,\underline{\underline{Q}}\,K)\,\lambda x.\underline{\underline{P}} \\ & \to_\beta & (\lambda x.\underline{\underline{P}})\,\underline{\underline{Q}}\,K \\ & \twoheadrightarrow_\beta & \underline{\underline{P\{x := Q\}}}\,K \\ & \twoheadrightarrow_\beta & \overline{(P\{x := Q\})}:K \end{array}$$

2. $M \equiv P\,Q \to_\beta P\,Q' \equiv N$, where $Q \to_\beta Q'$. Then, by the induction hypothesis and Lemma 5.5(ii),
$$\begin{array}{rcl} M:K & \equiv & P:(\lambda m.m\,\underline{\underline{Q}}\,K) \\ & \twoheadrightarrow_\beta^+ & P:(\lambda m.m\,\underline{\underline{Q'}}\,K) \\ & \equiv & N:K \end{array}$$

The remaining cases are similar to Case 2.

(iii) Induction on $M \to_\pi N$.

1. $M \equiv [P,Q]\,R \to_\pi [P,R]\,Q \equiv N$. Then, by the induction hypothesis,
$$\begin{array}{rcl} M:K & \equiv & y\,(P:(\lambda m.m\,\underline{\underline{R}}\,K))\,\underline{\underline{Q}} \\ & \equiv & y\,((P\,R):K)\,\underline{\underline{Q}} \\ & \equiv & N:K \end{array}$$

2. $M \equiv P\,Q \to_\pi P\,Q' \equiv N$, where $Q \to_\pi Q'$. Then, by the induction hypothesis,
$$\begin{array}{rcl} M:K & \equiv & P:(\lambda m.m\,\underline{\underline{Q}}\,K) \\ & \equiv & P:(\lambda m.m\,\underline{\underline{Q'}}\,K) \\ & \equiv & N:K \end{array}$$

The remaining cases are similar to Case 2. □

5.8. THEOREM. *For all $M \in \Lambda_K$*
$$\underline{\iota(M)} \in \mathrm{WN}_\beta \;\Rightarrow\; M \in \mathrm{SN}_\beta$$

PROOF. By Proposition 5.3 and Lemma 5.7, since $\underline{\bullet}, \underline{\underline{\bullet}} : \Lambda_I^\pi \to \Lambda_I$. □

The following corollary states this more explicitly. For comparison with a later construction in Section 7 and 8, the translation in the corollary omits some η-redexes.

5.9. Corollary. *Define* $[\bullet] : \Lambda_K \to \Lambda_I$ *by:*

$$\begin{array}{lcl} [x] & = & \lambda k.x\, k \\ [\lambda x.P] & = & \lambda k.k\, \lambda x.\lambda h.y\, ([P]\, h)\, x \\ [P\, Q] & = & \lambda k.[P]\, \lambda m.m\, [Q]\, k \end{array}$$

For all $M \in \Lambda_K$,

$$[M] \in \mathrm{WN}_\beta \;\Rightarrow\; M \in \mathrm{SN}_\beta$$

Proof. By a well-known result [2, 15.1.5], for all $M \in \Lambda_K$,

$$M \in \mathrm{WN}_\beta \;\Leftrightarrow\; M \in \mathrm{WN}_{\beta\eta}$$

Assume $[M] \in \mathrm{WN}_\beta$, i.e., $[M] \in \mathrm{WN}_{\beta\eta}$. By induction on M, show that

$$\underline{\iota(M)} \twoheadrightarrow_\eta [M]$$

Therefore, $\underline{\iota(M)} \in \mathrm{WN}_{\beta\eta}$. Hence $\underline{\iota(M)} \in \mathrm{WN}_\beta$. Now use Theorem 5.8. □

Xi [76] independently discovers Corollary 5.9 and uses it to prove that weak normalization implies strong normalization in simply and second-order typed λ-calculus, and mentions that the technique extends to higher-order typed λ-calculus. Whereas the present paper obtains the translation $[\bullet]$ as the composition of Klop's translation with a CPS translation, Xi studies the composition directly. The resulting proof of Corollary 5.9 is very short, but—in our opinion—less transparent.

5.10. Remark. One might wonder whether the assumption $[M] \in \mathrm{WN}_\beta$ can be replaced by a weaker condition, e.g., that $[M]$ has a head normal form or weak head normal form. None of these two weaker conditions are sufficient as the example $M \equiv \lambda x.\Omega$ shows.

5.11. Remark. It is natural to wonder whether our extension of Klop's technique has analogous extensions of the techniques by Nederpelt, de Groote, etc. Indeed, the rule β_{lift} in [60] which generalizes β_S can be simulated by a CPS translation [60], as was also noted in [37]. However, this yields the property

$$\underline{M} \in \mathrm{WN}_{\beta_I} \;\Rightarrow\; M \in \mathrm{SN}_\beta$$

as opposed to our

$$\underline{\iota(M)} \in \mathrm{WN}_\beta \;\Rightarrow\; M \in \mathrm{SN}_\beta$$

In the former case one has to prove that $\underline{M} \in \mathrm{WN}_{\beta_I}$. This is not the same as $\underline{\iota(M)} \in \mathrm{WN}_\beta$ (none of the two sets is contained in the other). Thus, with the former technique one does not infer strong normalization of one notion of reduction from weak normalization of the same notion of reduction

6. Simulations from permutative inner interpretations

In this section we show that simulation by CPS translation is a special case of simulation by a general model-like construction. To do so we replace the specific CPS translation by a generic translation, and replace the specific colon translation by a generalization of Sabry and Felleisen's [60] compacting CPS translation. The colon translation cannot be generalized directly because it exploits the fact that an explicit translation is given.

The first subsection introduces permutative interpretations. The second subsection shows how to derive simulations of π from permutative inner interpretations. The third subsection shows that the technique based on CPS-translations from Section 5 is a special case. The fourth subsection gives another special case due to Loader [45]. The last subsection explains the relation to the notion of an inner model.

6.1. Permutative inner interpretations

6.1. Definition.

(i) An *inner interpretation* is a tuple $I = \langle E, F, G, H\rangle$ of terms from Λ_K.

(ii) The map $[\![\bullet]\!]_I : \Lambda_K \to \Lambda_K$ *determined by* I is defined by:

$$\begin{array}{lcl} [\![x]\!]_I & = & E\,x \\ [\![P\,Q]\!]_I & = & F\,[\![P]\!]_I\,[\![Q]\!]_I \\ [\![\lambda y.P]\!]_I & = & G\,(\lambda y.H\,[\![P]\!]_I\,(E\,y)) \end{array}$$

6.2. Notation. Given an inner interpretation $I = \langle E, F, G, H\rangle$, the term $[\![M]\!]_I$ has a number of occurrences of the terms E, F, G, H introduced by the translation. However, there may be subterm occurrences in $[\![M]\!]_I$ identical to one of E, F, G, H which were not introduced by the translation. For instance, if E and M are both the free variable y, then $[\![M]\!]_I \equiv y\,y$ has two occurrences of E, but only one were introduced by the translation.

We assume that the set V of variables in Λ_K is divided into two denumerable, disjoint sets V_0 and V_1. In implicit α-conversions, variables are renamed by other variables in the same set. All terms are assumed to use variables from V_0, except the terms E, F, G, H in an inner interpretation, which always use variables from V_1. Define the notions of reduction β_0 and β_1 by:[3]

$$(\lambda x.P)\,Q \quad \beta_i \quad P\{x := Q\} \quad \text{if } x \in V_i$$

Then $\beta = \beta_0 \cup \beta_1$.

6.3. Definition. Let $I = \langle E, F, G, H\rangle$ be a permutative inner interpretation.

[3]No connection with Nederpelt's β_1 is intended.

(i) The language $\mathcal{L}(I)$ *determined by* I is defined by:

$$M ::= E\,x \mid F\,M_1\,M_2 \mid G\,\lambda y.M \mid H\,M_1\,M_2$$

(ii) I is *permutative* if, for all $X, Y, Z \in \mathcal{L}(I)$,

1. $E\,X =_{\beta_1} X$;
2. $F\,(G\,\lambda x.X)\,Y =_{\beta_1} E\,((\lambda x.X)\,Y)$;
3. $F\,(H\,X\,Y)\,Z =_{\beta_1} H\,(F\,X\,Z)\,Y$.

(iii) I is a *sound* if, for all $X, Y, Z \in \mathcal{L}(I)$, 1-2 hold, and

4. $H\,X\,Y =_{\beta_1} X$.

6.4. Remark. Any inner interpretation which is sound is also permutative, but the converse is not generally true.

Given any permutative inner interpretation $I = \langle E, F, G, H\rangle$, we shall show that, if E, F, G, H are linear terms, then $[\![M]\!]_I \in \mathrm{WN}_\beta$ implies $M \in \mathrm{SN}_\beta$, for all $M \in \Lambda_K$.

6.2. Simulations from permutative inner interpretations

The following is a convenient auxiliary notion.

6.5. Definition. Given an inner interpretation $I = \langle E, F, G, H\rangle$, define the map $\{\![\bullet]\!\}_I : \Lambda_K^\pi \to \Lambda_K$ by:

$$\begin{array}{lcl}
\{\![x]\!\}_I & = & E\,x \\
\{\![P\,Q]\!\}_I & = & F\,\{\![P]\!\}_I\,\{\![Q]\!\}_I \\
\{\![\lambda y.P]\!\}_I & = & G\,\lambda y.\{\![P]\!\}_I \\
\{\![[P, Q]]\!\}_I & = & H\,\{\![P]\!\}_I\,\{\![Q]\!\}_I
\end{array}$$

6.6. Remark. $\{\![\iota(M)]\!\}_I \equiv [\![M]\!]_I$ and $\{\![N]\!\}_I \in \mathcal{L}(I)$ for all $N \in \Lambda_K^\pi$ and $M \in \Lambda_K$.

6.7. Lemma. *Let $I = \langle E, F, G, H\rangle$ be a permutative inner interpretation. For all $M, N \in \Lambda_K^\pi$,*

$$\{\![M]\!\}_I\{x := \{\![N]\!\}_I\} =_{\beta_1} \{\![M\{x := N\}]\!\}_I$$

Proof. By induction on M.

1. $M \equiv x$. Then

$$\begin{array}{lcl}
\{\![x]\!\}_I\{x := \{\![N]\!\}_I\} & \equiv & E\,\{\![N]\!\}_I \\
& =_{\beta_1} & \{\![N]\!\}_I \\
& \equiv & \{\![x\{x := N\}]\!\}_I
\end{array}$$

2. $M \equiv y \not\equiv x$. Then

$$\begin{array}{rcl} \{\![y]\!\}_I\{x := \{\![N]\!\}_I\} & \equiv & E\,y \\ & \equiv & \{\![y]\!\}_I \\ & \equiv & \{\![y\{x := N\}]\!\}_I \end{array}$$

In the remaining cases, apply the induction hypothesis. □

6.8. Lemma. *For all $M, N \in \Lambda^\pi_K$,*

(i)
$$\begin{tikzcd}
M \arrow[rrr, "\rightarrow_\beta"'] \arrow[d, "\{\![\bullet]\!\}_I"] & & & N \arrow[d, "\{\![\bullet]\!\}_I"'] \\
\{\![M]\!\}_I \arrow[r, dotted, no head, "=_{\beta_1}"'] & L \arrow[r, dotted, "\rightarrow_{\beta_0}"'] & L' \arrow[r, dotted, no head, "=_{\beta_1}"'] & \{\![N]\!\}_I
\end{tikzcd}$$

(ii)
$$\begin{tikzcd}
M \arrow[r, "\rightarrow_\pi"'] \arrow[d, "\{\![\bullet]\!\}_I"] & N \arrow[d, "\{\![\bullet]\!\}_I"'] \\
\{\![M]\!\}_I \arrow[r, dotted, no head, "=_{\beta_1}"'] & \{\![N]\!\}_I
\end{tikzcd}$$

Proof.

(i) Induction on $M \rightarrow_\beta N$. If $M \equiv (\lambda x.P)\,Q \rightarrow_{\beta_0} P\{x := Q\} \equiv N$, then

$$\begin{array}{rcl} \{\![M]\!\}_I & \equiv & F\,(G\,\lambda x.\{\![P]\!\}_I)\,\{\![Q]\!\}_I \\ & =_{\beta_1} & E\,((\lambda x.\{\![P]\!\}_I)\,\{\![Q]\!\}_I) \\ & \rightarrow_{\beta_0} & E\,(\{\![P]\!\}_I\{x := \{\![Q]\!\}_I\}) \\ & =_{\beta_1} & E\,(\{\![P\{x := Q\}]\!\}_I) \\ & =_{\beta_1} & \{\![P\{x := Q\}]\!\}_I \\ & \equiv & \{\![N]\!\}_I \end{array}$$

In the remaining cases, apply the induction hypothesis.

(ii) Induction on $M \rightarrow_\pi N$. If $M \equiv [P, Q]\,R \rightarrow_\pi [P\,R, Q] \equiv N$, then

$$\begin{array}{rcl} \{\![M]\!\}_I & \equiv & F\,(H\,\{\![P]\!\}_I\,\{\![Q]\!\}_I)\,\{\![R]\!\}_I \\ & =_{\beta_1} & H\,(F\,\{\![P]\!\}_I\,\{\![R]\!\}_I)\,\{\![Q]\!\}_I \\ & \equiv & \{\![N]\!\}_I \end{array}$$

In the remaining cases, use the induction hypothesis. □

6.9. Definition.

(i) Define for $M \in \Lambda_K$ and variable z, $\|M\|$ and $\|M\|_z$ by:

$$\begin{array}{ll@{\qquad}lll} \|x\| & = 1 & \|x\|_z & = 1 & \text{if } z \equiv x, \text{ else } 0 \\ \|\lambda x.M\| & = 1 + \|M\| & \|\lambda x.M\|_z & = \|M\|_z & \text{if } z \not\equiv x, \text{ else } 0 \\ \|M\,N\| & = 1 + \|M\| + \|N\| & \|M\,N\|_z & = \|M\|_z + \|N\|_z & \end{array}$$

(ii) $\Lambda_L = \{M \in \Lambda_K \mid \lambda x.P \subseteq M \text{ and } x \in V_1 \;\Rightarrow\; \|P\|_x = 1\}$

The following lemma shows that $\mathrm{nf}_{\beta_1}(M)$ is well-defined for $M \in \Lambda_L$.

6.10. Lemma.

(i) *For all* $M \in \Lambda_L : M \to_{\beta_1} N \;\Rightarrow\; N \in \Lambda_L$.

(ii) *For all* $M \in \Lambda_L : \mathrm{SN}_{\beta_1}(M)$

(iii) *For all* $M \in \Lambda_K : \mathrm{CR}_{\beta_1}(M)$.

Proof.

(i) Prove by induction on P that for all $P, Q \in \Lambda_L$ and $k \neq l$:

$$\|P\{k := Q\}\|_l = \|P\|_l + \|Q\|_l \cdot \|P\|_k$$

Using this prove by induction on $M \to_{\beta_1} N$ that for all $M \in \Lambda_L$:

$$M \to_{\beta_1} N \;\Rightarrow\; \|M\|_l = \|N\|_l \tag{10}$$

Then prove by induction on P that for all $P, Q \in \Lambda_L$ and $k \in V_1$:

$$P\{k := Q\} \in \Lambda_L \tag{11}$$

Finally prove (i) by induction on $M \to_{\beta_1} N$ using (10)-(11):

1. $M \equiv (\lambda k.P)\, Q \to_{\beta_1} P\{k := Q\} \equiv N$. Then, by (11), $N \in \Lambda_L$.
2. $M \equiv \lambda k.P \to_{\beta_1} \lambda k.Q \equiv N$, where $P \to_{\beta_1} Q$. Since $M \in \Lambda_L$, also $P \in \Lambda_L$ and $\|P\|_k = 1$. By the induction hypothesis $Q \in \Lambda_L$, and by (10), $\|Q\|_k = 1$. Therefore, $N \in \Lambda_L$.

In the remaining cases, apply the induction hypothesis directly.

(ii) Prove by induction on P that for all $P, Q, \in \Lambda_L$:

$$\|P\{k := Q\}\| = \|P\| + (\|Q\| - 1) \cdot \|P\|_k$$

Use this to prove by induction on $M \to_{\beta_1} N$ that for all $M \in \Lambda_L$:

$$M \to_{\beta_1} N \;\Rightarrow\; \|M\| > \|N\| \tag{12}$$

Now (ii) follows by (i) and (12).

(iii) By the technique due to Tait and Martin-Löf, see [2]. $\square$

6.11. Lemma. *For all* $M, M' \in \Lambda_L, N \in \Lambda_K$:

$$\begin{tikzcd}
M \arrow[r, "\to_{\beta_0}"] \arrow[d, "\twoheadrightarrow_{\beta_1}"'] & N \arrow[d, dotted, "\twoheadrightarrow_{\beta_1}"] \\
M' \arrow[r, dotted, "\to_{\beta_0}"'] & N'
\end{tikzcd}$$

Proof. It suffices, by Lemma 6.10(i) and transitivity, to prove the assertion when $M \to_{\beta_1} M'$.

First show, for any $M, N, L, K \in \Lambda_K$ with $\|K\|_k = 1$,

$$\begin{array}{llr}
\text{If } M \to_{\beta_1} N & \text{then} \quad M\{x := L\} \to_{\beta_1} N\{x := L\} & (13) \\
\text{If } M \to_{\beta_1} N & \text{then} \quad L\{x := M\} \twoheadrightarrow_{\beta_1} L\{x := N\} & (14) \\
\text{If } M \to_{\beta_0} N & \text{then} \quad M\{k := L\} \to_{\beta_0} N\{k := L\} & (15) \\
\text{If } M \to_{\beta_0} N & \text{then} \quad K\{k := M\} \to_{\beta_0} K\{k := N\} & (16)
\end{array}$$

Here (13),(15) are by induction on $M \to_\beta N$, (14),(16) by induction on L.

We now proceed by induction on $M \to_\beta N$ using (13)-(16):

1. $M \equiv (\lambda x.P)\ Q \to_{\beta_0} P\{x := Q\} \equiv N$. Then $M \to_{\beta_1} (\lambda x.P')\ Q' \equiv M'$, where $P \to_{\beta_1} P'$ and $Q \equiv Q'$, or vice versa. With $N' \equiv P'\{x := Q'\}$, both $M' \to_{\beta_0} N'$ and $N \twoheadrightarrow_{\beta_1} N'$, by (13)-(14).
2. $M \equiv (\lambda k.P)\ Q \to_{\beta_1} P\{k := Q\} \equiv M'$. Then $M \to_{\beta_0} (\lambda k.R)\ S \equiv N$ where $P \to_{\beta_0} R$ and $Q \equiv S$, or vice versa. With $N' \equiv R\{k := S\}$, $N \to_{\beta_1} N'$ and $M' \to_{\beta_0} N'$, by (15)-(16).

In the remaining cases, use the induction hypothesis. □

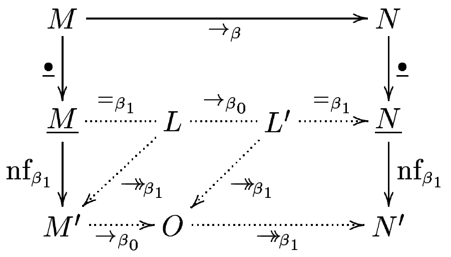
6.12. Lemma. *Let I be a permutative inner interpretation of Λ_L terms. For all $M, N \in \Lambda_K^\pi$:*

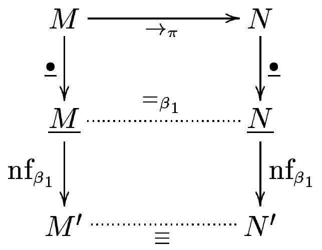
(i) $\{\![M]\!\}_I \twoheadrightarrow_\beta \mathrm{nf}_{\beta_1}(\{\![M]\!\}_I)$.

(ii) $M \to_\beta N \Rightarrow \mathrm{nf}_{\beta_1}(\{\![M]\!\}_I) \twoheadrightarrow_\beta^+ \mathrm{nf}_{\beta_1}(\{\![N]\!\}_I)$.

(iii) $M \to_\beta N \Rightarrow \mathrm{nf}_{\beta_1}(\{\![M]\!\}_I) \equiv \mathrm{nf}_{\beta_1}(\{\![N]\!\}_I)$.

Proof. (i) is obvious and for (ii)-(iii) we have the diagrams


$$\begin{tikzcd}
M \arrow[rrr, "\to_\beta"] \arrow[d, "\bullet"'] & & & N \arrow[d, "\bullet"] \\
\underline{M} \arrow[r, dotted, dash, "=_{\beta_1}"] \arrow[d, "\mathrm{nf}_{\beta_1}"'] & L \arrow[r, dotted, dash, "\to_{\beta_0}"] \arrow[dl, dotted, "\twoheadrightarrow_{\beta_1}"] & L' \arrow[r, dotted, "=_{\beta_1}"] \arrow[dl, dotted, "\twoheadrightarrow_{\beta_1}"] & \underline{N} \arrow[d, "\mathrm{nf}_{\beta_1}"] \\
M' \arrow[r, dotted, "\to_{\beta_0}"'] & O \arrow[rr, dotted, "\twoheadrightarrow_{\beta_1}"'] & & N'
\end{tikzcd}$$


$$\begin{tikzcd}
M \arrow[r, "\to_\pi"] \arrow[d, "\bullet"'] & N \arrow[d, "\bullet"] \\
\underline{M} \arrow[r, dotted, dash, "=_{\beta_1}"] \arrow[d, "\mathrm{nf}_{\beta_1}"'] & \underline{N} \arrow[d, "\mathrm{nf}_{\beta_1}"] \\
M' \arrow[r, dotted, dash, "\equiv"'] & N'
\end{tikzcd}$$

by Lemmas 6.8, and 6.11. □

6.13. Theorem. *Let I be a permutative inner interpretation of Λ_L terms. For all $M \in \Lambda_K$,*

$$[\![M]\!]_I \in \mathrm{WN}_\beta \Rightarrow M \in \mathrm{SN}_\beta$$

Proof. By proposition 5.3, Remark 6.6, and Lemma 6.12. □

6.3. CPS translation as a permutative inner interpretation

6.14. Proposition. *Let $I = \langle E, F, G, H\rangle$, where for some fixed variable y,*

$$\begin{array}{rcl} E & \equiv & \lambda X.\lambda k.X\,k \\ F & \equiv & \lambda M.\lambda N.\lambda k.M\,\lambda m.m\,N\,k \\ G & \equiv & \lambda M.\lambda k.k\,M \\ H & \equiv & \lambda M.\lambda N.\lambda k.y\,(M\,k)\,N \end{array}$$

Then I is a permutative inner interpretation of Λ_L terms.

Proof. We prove that the terms E, F, G, H satisfy the equations of a permutative inner interpretation.

(i) For all $P \in \mathcal{L}(I)$, $P =_{\beta_1} \lambda l.R$ for some R and $l \in V_1$. Thus,

$$\begin{array}{rcl} E\,P & =_{\beta_1} & \lambda k.(\lambda l.R)\,k \\ & =_{\beta_1} & \lambda k.R\{k := l\} \\ & \equiv & \lambda l.R \\ & \equiv & P \end{array}$$

(ii) For all $P, Q \in \mathcal{L}(I)$,

$$\begin{array}{rcl} F\,(G\,\lambda x.P)Q & =_{\beta_1} & \lambda k.(\lambda h.h\,\lambda x.P)\,\lambda m.m\,Q\,k \\ & =_{\beta_1} & \lambda k.(\lambda m.m\,Q\,k)\,\lambda x.P \\ & =_{\beta_1} & \lambda k.(\lambda x.P)\,Q\,k \\ & =_{\beta_1} & E\,((\lambda x.P)\,Q) \end{array}$$

(iii) For all $P, Q, R \in \mathcal{L}(I)$,

$$\begin{array}{rcl} F\,(H\,P\,Q)\,R & \equiv & \lambda k.(\lambda l.y\,(P\,l)\,Q)\,\lambda m.m\,R\,k \\ & =_{\beta_1} & \lambda k.y\,(P\,\lambda m.m\,R\,k)\,Q \\ & =_{\beta_1} & \lambda k.y\,((\lambda h.P\,\lambda m.m\,R\,h)\,k)\,Q \\ & =_{\beta_1} & H\,(F\,P\,R)\,Q \end{array}$$

This concludes the proof. □

6.4. Loader's permutative inner interpretation

Loader [45] uses a translation $(\!|\bullet|\!)$ mapping a typed term in simply and second-order typed λ-calculus into constructive evidence for the statement that the term is strongly normalizing. He uses the translation to prove that weak normalization implies strong normalization in these calculi, and mentions that the technique extends to higher-order typed λ-calculus.

More specifically, in the case of simply typed λ-calculus, Loader's translation $(\!|\bullet|\!)$ can be viewed as follows:

$$\begin{array}{rcl} (\!|x|\!) & = & x \\ (\!|P\,Q|\!) & = & (\!|P|\!)\,(\!|Q|\!) \\ (\!|\lambda y.P|\!) & = & \lambda y.H_{\sigma\to\tau}\,(\!|P|\!)\,y \end{array}$$

where H_τ is a family of simply typed Λ_L terms satisfying, for $X, Y, Z \in \Lambda_K$,

$$(H_{\sigma\to\tau}\, X\, Y)\, Z =_{\beta_1} H_{\sigma\to\tau}\, (X\, Z)\, Y$$

and where the choice of $\sigma \to \tau$ in the third clause is made on the basis of the type of $\lambda y.P$. Thus, his translation can be viewed as the permutative inner interpretation $\langle \mathbf{I}, \mathbf{I}, \mathbf{I}, H_\tau\rangle$ of Λ_L terms, where we allow a family of H's.

6.5. Inner models versus sound inner interpretations

We end the section by explaining the relation between sound inner interpretations and inner models, as presented in, e.g., [5].

6.15. DEFINITION.

(i) A pair $I = \langle F, G\rangle$ of Λ_K terms is an *inner model* if $\lambda x.F\,(G\,x) =_\beta \mathbf{I}$.

(ii) The map $[\![\bullet]\!]'_I : \Lambda_K \to \Lambda_K$ determined by I is defined by:

$$\begin{array}{lcl} [\![x]\!]'_I & = & x \\ [\![P\,Q]\!]'_I & = & F\,[\![P]\!]'_I\,[\![Q]\!]'_I \\ [\![\lambda y.P]\!]'_I & = & G\,(\lambda y.[\![P]\!]'_I) \end{array}$$

6.16. PROPOSITION. *If $\langle F, G\rangle$ is an inner model, then $\langle \mathbf{I}, F, G, \mathbf{K}\rangle$ is a sound inner interpretation.*

PROOF. If $\langle F, G\rangle$ is an inner model, then, by the Church-Rosser property, $F\,(G\,x) \twoheadrightarrow_{\beta_1} x$ for any variable x, and hence $F\,(G\,X)\,Y =_{\beta_1} X\,Y =_{\beta_1} \mathbf{I}\,(X\,Y)$ for any $X, Y \in \Lambda_K$. The remaining two axioms of sound interpretations are clearly satisfied. $\square$

The converse is not generally true. However, the main property of inner models is that $M =_\beta N$ implies $[\![M]\!]'_I =_\beta [\![M]\!]'_I$ for all $M, N \in \Lambda_K$. The same holds for sound inner interpretations. Thus, the notion of a sound inner interpretation is weaker than that of an inner model, but strong enough to entail the main property of an inner model.

6.17. REMARK. Inner models are related to *term models* of the untyped λ-calculus, see [2].

7. Application to typed λ-calculi à la Curry

In this section we use the CPS-translation from Section 5 to prove that weak normalization implies strong normalization in some typed λ-calculi à la Curry.

The first subsection introduces such calculi in general. The three next subsections consider simple types $\lambda\!\to$, positive recursive types $\lambda\mu^+$, and subtypes $\lambda\subseteq$; see, e.g., [3, 70, 47], respectively. The last subsection studies the use of permutative inner interpretations, in general, to prove that weak normalization implies strong normalization; for simplicity we consider only simply typed λ-calculus.

7.1. Typed λ-calculi à la Curry

7.1. Definition.

(i) The set Context(Θ) of *contexts* over a set Θ is the set of all

$$\{x_1, \tau_1, \ldots, x_n : \tau_n\}$$

where $\tau_1, \ldots, \tau_n \in \Theta$, $x_1, \ldots, x_n \in V$ (variables of Λ_K) and where $x_i \not\equiv x_j$ for $i \neq j$.

(ii) For context $\Gamma = \{x_1 : \tau_1, \ldots, x_n : \tau_n\}$, we write $\mathrm{dom}(\Gamma) = \{x_1, \ldots, x_n\}$.

(iii) We write $x : \tau$ for $\{x : \tau\}$ and Γ, Γ' for $\Gamma \cup \Gamma'$ if $x : \sigma \in \Gamma$ and $x : \tau \in \Gamma$ implies $\sigma \equiv \tau$.

(iv) A *typed λ-calculus à la Curry* λS is a pair $(\Theta, \vdash)$, where

$$\vdash \quad \subseteq \quad \mathrm{Context}(\Theta) \times \Lambda_K \times \Theta$$

(v) $M \in \Lambda_K$ is *typable* in λS if $\Gamma \vdash M : \tau$ for some $\Gamma \in \mathrm{Context}(\Theta)$, $\tau \in \Theta$.

(vi) We write $\lambda S \models \mathrm{WN}_\beta$ if $M \in \mathrm{WN}_\beta$ for all M typable in λS. Similarly, we write $\lambda S \models \mathrm{SN}_\beta$.

To prove that weak normalization implies strong normalization in λS it suffices to show that $[\bullet]$ preserves typability.

7.2. Proposition. *Let λS be a typed λ-calculus à la Curry. If, for all $M \in \Lambda_K$,*

$$M \textit{ typable} \Rightarrow [M] \textit{ typable}$$

then

$$\lambda S \models \mathrm{WN}_\beta \Rightarrow \lambda S \models \mathrm{SN}_\beta$$

Proof. Assume $\lambda{\rightarrow} \models \mathrm{WN}_\beta$ and let M be a typable term. By assumption, $[M]$ is typable, so $[M] \in \mathrm{WN}_\beta$. By Corollary 5.9, $M \in \mathrm{SN}_\beta$. $\square$

It is well-known that various CPS translations preserve typability in various typed λ-calculi, see, e.g., [11, 22, 23, 44, 46].

7.2. Simple types

7.3. Definition. The simply typed λ-calculus $\lambda{\to} = (\text{Type}(\lambda{\to}), \vdash)$ is:

(i) $\text{Type}(\lambda{\to})$ is defined by the grammar:

$$\tau, \sigma ::= \alpha \mid \tau \to \sigma$$

where U is a set of *type variables* ranged over by α.

(ii) The relation $\vdash$ is defined by:

$$\frac{}{\Gamma, x : \tau \vdash x : \tau} \qquad \frac{\Gamma, x : \sigma \vdash P : \tau}{\Gamma \vdash \lambda x.P : \sigma \to \tau} \qquad \frac{\Gamma \vdash P : \sigma \to \tau \quad \Gamma \vdash Q : \sigma}{\Gamma \vdash P\,Q : \tau}$$

7.4. Definition. Let $\bot$ be a fixed type, and $\neg\sigma \equiv \sigma \to \bot$. Define maps

$$[\bullet], [\bullet]' : \text{Type}(\lambda{\to}) \to \text{Type}(\lambda{\to})$$

by:

$$\begin{array}{lcl} [\sigma] & = & \neg\neg[\sigma]' \\ & & \\ [\alpha]' & = & \alpha \\ [\sigma \to \tau]' & = & [\sigma] \to [\tau] \end{array}$$

Also, $[\Gamma] = \{x : [\sigma] \mid x : \sigma \in \Gamma\}$.

7.5. Convention. From now on we assume that the translation $[\bullet]$ from Section 5 does not introduce several occurrences of the free variable y, but rather a single occurrence of each of a number of distinct free variables. Thus, instead of

$$[\lambda x.\lambda z.x] \equiv \lambda k.k\ \lambda x.\lambda h.y\ ([\lambda l.l\ \lambda z.\lambda m.y\ (x\ m)\ z]\ h)\ x$$

we shall now have

$$[\lambda x.\lambda z.x] \equiv \lambda k.k\ \lambda x.\lambda h.y_1\ ([\lambda l.l\ \lambda z.\lambda m.y_2\ (x\ m)\ z]\ h)\ x$$

This clearly has no influence on the normalization properties of $[M]$, but will be important for typing properties.

7.6. Lemma. *For all $M \in \Lambda_K$, $\sigma \in Type(\lambda{\to})$, $\Gamma \in Context(Type(\lambda{\to}))$,*

$$\Gamma \vdash M : \sigma \Rightarrow \Delta, [\Gamma] \vdash [M] : [\sigma]$$

for a Δ with $dom(\Delta) = \text{FV}([M]) \backslash \text{FV}(M)$.

Proof. Induction on $\Gamma \vdash M : \sigma$ using Convention 7.5. □

7.7. Corollary. $\lambda{\to} \models \text{WN}_\beta \Rightarrow \lambda{\to} \models \text{SN}_\beta$.

7.3. Positive, recursive types

7.8. Definition. $\lambda\mu^+$ is as $\lambda\!\rightarrow$ but with extra types of form:

$$\tau, \sigma ::= \ldots \mid \mu\alpha.\sigma$$

where α occurs only *positively* in σ, see, e.g., [70], and with the extra rule:

$$\frac{\Gamma \vdash M : \sigma \quad \sigma \sim \tau}{\Gamma \vdash M : \tau}$$

where $\sim$ is the least congruence on $\mathrm{Type}(\lambda\mu^+)$ with $\mu\alpha.\sigma \sim \sigma\{\alpha := \mu\alpha.\sigma\}$.

7.9. Definition. Define $[\bullet], [\bullet]' : \mathrm{Type}(\lambda\mu^+) \to \mathrm{Type}(\lambda\mu^+)$ as for $\lambda\!\rightarrow$ and:

$$[\mu\alpha.\sigma]' \quad = \quad \mu\alpha.[\sigma]'$$

That $[\sigma], [\sigma]' \in \mathrm{Type}(\lambda\mu^+)$ is easily established by induction on σ.

7.10. Lemma.

(i) $[\sigma]'\{\alpha := [\tau]'\} \equiv [\sigma\{\alpha := \tau\}]'$.

(ii) $\sigma \sim \tau \;\Rightarrow\; [\sigma] \sim [\tau]$

(iii) $\Gamma \vdash M : \sigma \Rightarrow \Delta, [\Gamma] \vdash [M] : [\sigma]$, *for a* Δ *with* $\mathrm{dom}(\Delta) = \mathrm{FV}([M]) \backslash \mathrm{FV}(M)$.

Proof.

(i) Induction on σ.

(ii) Since $\sim$ is a congruence, $\sigma \sim \tau$ implies $\neg\neg\sigma \sim \neg\neg\tau$. Now prove by induction on $\sigma \sim \tau$ that $\sigma \sim \tau$ implies $[\sigma] \sim [\tau]$, using (i).

(iii) Induction on $\Gamma \vdash M : \sigma$ using (ii). □

7.11. Corollary. $\lambda\mu^+ \models \mathrm{WN} \;\Rightarrow\; \lambda\mu^+ \models \mathrm{SN}$.

7.4. Subtypes

7.12. Definition. $\lambda\!\subseteq$ is as $\lambda\!\rightarrow$ but with some extra base types:

$$\tau, \sigma ::= \ldots \mid b$$

and with the extra rule:

$$\frac{\Gamma \vdash M : \sigma \quad \sigma \subseteq \tau}{\Gamma \vdash M : \tau}$$

where $\subseteq$ is any relation on $\mathrm{Type}(\lambda\!\subseteq)$ closed under the following rules:

$$\sigma \subseteq \sigma \qquad \frac{\sigma' \subseteq \sigma, \tau \subseteq \tau'}{\sigma \to \tau \subseteq \sigma' \to \tau'} \qquad \frac{\sigma \subseteq \tau, \tau \subseteq \rho}{\sigma \subseteq \rho}$$

7.13. Definition. Define $[\bullet], [\bullet]' : \mathrm{Type}(\lambda\subseteq) \to \mathrm{Type}(\lambda\subseteq)$ as for $\lambda\!\!\to$ and:

$$[b]' = b$$

7.14. Lemma.

(i) $\sigma \subseteq \tau \Rightarrow [\sigma] \subseteq [\tau]$

(ii) $\Gamma \vdash M : \sigma \Rightarrow \Delta, [\Gamma] \vdash [M] : [\sigma]$, *where* $\mathrm{dom}(\Delta) = \mathrm{FV}([M])\backslash\mathrm{FV}(M)$.

Proof.

(i) First note that $\sigma \subseteq \tau$ implies $\neg\neg\sigma \subseteq \neg\neg\tau$. Now prove by induction on $\sigma \subseteq \tau$ that $\sigma \subseteq \tau$ implies $[\sigma] \subseteq [\tau]$.

(ii) Induction on $\Gamma \vdash M : \sigma$ using (i). □

7.15. Corollary. $\lambda\subseteq\models \mathrm{WN} \Rightarrow \lambda\subseteq\models \mathrm{SN}$.

7.5. Inner type interpretations in $\lambda\!\!\to$

We have shown that a specific permutative inner interpretation preserves typability in some calculi à la Curry and hence that weak normalization implies strong normalization in these calculi. In this subsection we present a condition guaranteeing that the map determined by any permutative inner interpretation preserves typability in $\lambda\!\!\to$. Each linear permutative inner interpretation satisfying the condition hence gives a technique to prove that weak normalization implies strong normalization in $\lambda\!\!\to$; similar conditions can be derived for other systems.

7.16. Definition.

(i) $T : \mathrm{Type}(\lambda\!\!\to) \to \mathrm{Type}(\lambda\!\!\to)$ is an *inner type interpretation* of $\lambda\!\!\to$ if

$$T(\sigma)\{\alpha := \tau\} \equiv T(\sigma\{\alpha := \tau\})$$

(ii) The map $[\![\bullet]\!]_T : \mathrm{Type}(\lambda\!\!\to) \to \mathrm{Type}(\lambda\!\!\to)$ determined by T is given by:

$$\begin{array}{lcl} [\![\sigma]\!]_T & = & T[\![\sigma]\!]'_T \\ \\ [\![\alpha]\!]'_T & = & \alpha \\ [\![\sigma \to \tau]\!]'_T & = & [\![\sigma]\!]_T \to [\![\tau]\!]_T \end{array}$$

Also, $[\![\Gamma]\!]_T = \{x : [\![\sigma]\!]_T \mid x : \sigma \in \Gamma\}$.

(iii) An inner interpretation $I = \langle E, F, G, H\rangle$ *agrees with* inner type interpretation T if, for all $\sigma, \tau, \rho \in \mathrm{Type}(\lambda\!\!\to)$, there is a Δ such that

$$\begin{array}{l} \Delta \vdash E : [\![\tau]\!]_T \to [\![\tau]\!]_T \\ \Delta \vdash F : T([\![\sigma]\!]_T \to [\![\tau]\!]_T) \to ([\![\sigma]\!]_T \to [\![\tau]\!]_T) \\ \Delta \vdash G : ([\![\sigma]\!]_T \to [\![\tau]\!]_T) \to T([\![\sigma]\!]_T \to [\![\tau]\!]_T) \\ \Delta \vdash H : [\![\tau]\!]_T \to [\![\sigma]\!]_T \to [\![\tau]\!]_T \end{array}$$

7.17. PROPOSITION. *If an inner interpretation I agrees with an inner type interpretation T, then*

$$\Gamma \vdash M : \sigma \Rightarrow \Delta, [\![\Gamma]\!]_T \vdash [\![M]\!]_I : [\![\sigma]\!]_T$$

for a Δ with $\mathrm{dom}(\Delta) = \mathrm{FV}([\![M]\!]_I) \backslash \mathrm{FV}(M)$.

PROOF. By induction on $\Gamma \vdash M : \sigma$ using Convention 7.5. □

7.18. REMARK. Inner interpretations agreeing with inner type interpretations resemble *Kleisli triples* and *monads*, see, e.g., [48, 49, 15, 59].

8. Application to typed λ-calculi à la Church

In this section we consider typed λ-calculi à la Church: second-order types $\lambda 2$ and higher-order types $\lambda\omega$. It is convenient to study so-called domain-free [6] variants of these calculi in which abstractions have form $\lambda x.M$ rather than $\lambda x{:}\sigma \,.\, M$.

8.1. REMARK. Domain-free systems are *not* generally Curry systems. In systems à la Curry the terms are those of the untyped λ-calculus; in domain-free systems the terms are those of systems à la Church with type tags omitted. For $\lambda{\rightarrow}$ the two views are equivalent, but for more powerful systems the two views diverge. An example term and type in $\lambda 2$ à la Church is

$$\lambda\alpha{:}* \,.\, \lambda x{:}\alpha \,.\, x \;:\; \forall\alpha.\alpha \to \alpha$$

In $\lambda 2$ à la Curry the similar term and type is

$$\lambda x.x \;:\; \forall\alpha.\alpha \to \alpha$$

The similar term and type in the domain-free approach is

$$\lambda\alpha.\lambda x.x \;:\; \forall\alpha.\alpha \to \alpha$$

8.1. Second-order types

8.2. DEFINITION. The system $\lambda 2$ is:

(i) $\lambda 2$ has *types* $\sigma, \tau \in \mathrm{Type}(\lambda 2)$:

$$\sigma, \tau ::= \alpha \mid \tau \to \sigma \mid \forall\alpha.\sigma$$

(ii) $\lambda 2$ has *terms* $P, Q \in \mathrm{Term}(\lambda 2)$:

$$P, Q ::= x \mid \lambda x.P \mid P\,Q \mid \lambda\alpha.P \mid P\,\sigma$$

(iii) The notion of reduction β on Term($\lambda 2$) is:

$$\begin{array}{lll} (\lambda\alpha.P)\,\sigma & \beta & P\{\alpha := \sigma\} \\ (\lambda x.P)\,Q & \beta & P\{x := Q\} \end{array}$$

(iv) $\lambda 2$ has *inference rules*:

$$\frac{}{\Gamma, x:\tau \ \vdash\ x:\tau} \qquad \frac{\Gamma, x:\sigma \ \vdash\ P:\tau}{\Gamma \ \vdash\ \lambda x.P : \sigma \to \tau} \qquad \frac{\Gamma \ \vdash\ P:\sigma\to\tau \quad \Gamma \ \vdash\ Q:\sigma}{\Gamma \ \vdash\ P\,Q:\tau}$$

$$\frac{\Gamma \ \vdash\ P:\tau \quad \alpha \notin \mathrm{FV}(\Gamma)}{\Gamma \ \vdash\ \lambda\alpha.P : \forall\alpha.\tau} \qquad \frac{\Gamma \ \vdash\ P:\forall\alpha.\sigma}{\Gamma \ \vdash\ P\,\tau : \sigma\{\alpha := \tau\}}$$

8.3. Definition. Let $\perp$ be any type, $\neg\sigma \equiv \sigma \to \perp$, and define the maps $[\bullet], [\bullet]' : \mathrm{Type}(\lambda 2) \to \mathrm{Type}(\lambda 2)$ by:

$$\begin{array}{lll} [\sigma] & = & \neg\neg[\sigma] \\ \\ [\alpha]' & = & \alpha \\ [\forall\alpha.\sigma]' & = & \forall\alpha.[\sigma] \\ [\sigma\to\tau]' & = & [\sigma]\to[\tau] \end{array}$$

A term M is *legal* if $\Gamma \ \vdash\ M:\sigma$ for some Γ,σ.

8.4. Definition. Define $[\bullet] : \mathrm{Term}(\lambda 2) \to \mathrm{Term}(\lambda 2)$ by:[4]

$$\begin{array}{lll} [x] & = & \lambda k.x\,k \\ [\lambda x.P] & = & \lambda l.l\ \lambda x.\lambda h.y\ ([P]\,h)\ x \\ [P\,Q] & = & \lambda l.[P]\ \lambda m.m\ [Q]\ l \\ [\lambda\alpha.P] & = & \lambda l.l\ \lambda\alpha.\lambda h.y\ ([P]\,h)\ \alpha \\ [P\,\sigma] & = & \lambda l.[P]\ \lambda m.m\ [\sigma]'\ l \end{array}$$

8.5. Theorem. $[M] \in \mathrm{WN}_\beta \ \Rightarrow\ M \in \mathrm{SN}_\beta$.

Proof. Like the proof of Theorem 6.13. □

8.6. Lemma.

(i) $[\sigma]'\{\alpha := [\tau]'\} \equiv [\sigma\{\alpha := \tau\}]'$.

(ii) $\Gamma \ \vdash\ M:\sigma \ \Rightarrow\ \Delta, [\Gamma] \ \vdash\ [M] : [\sigma]$, *where* $\mathrm{dom}(\Delta) = \mathrm{FV}([M]) \backslash \mathrm{FV}(M)$.

Proof.

[4]A small technical difficulty appears in 8.4 and 8.10. Suppose M is the term to be translated and $\lambda x.P$ a subterm. Then the second clause should—strictly speaking—read: $[\lambda x.P] = \lambda l.l\,\lambda x.\lambda h.(y\,\alpha_1 \ldots \alpha_n)\,([P]\,h)\,x$, where $\lambda\alpha_1, \ldots, \lambda\alpha_n$ are all the type abstractions in M whose scope $\lambda x.P$ is in.

(i) Induction on σ.

(ii) Induction on $\Gamma \vdash M : \sigma$ using (i). $\square$

Writing $\lambda 2 \models \mathrm{WN}_\beta$ to mean that all legal terms in $\lambda 2$ are weakly normalizing, and similarly with SN_β, we have the following.

8.7. COROLLARY. $\lambda 2 \models \mathrm{WN}_\beta \;\Rightarrow\; \lambda 2 \models \mathrm{SN}_\beta$.

8.2. Higher-order types

8.8. DEFINITION. The system $\lambda\omega$ is:

(i) $\lambda\omega$ has *kinds* $k, k' \in \mathrm{Kind}(\lambda\omega)$:

$$k, k' ::= * \mid k \to k'$$

(ii) $\lambda\omega$ has *constructors* $\sigma, \tau \in \mathrm{Con}(\lambda\omega)$ *of kind* k:

1. $\alpha^k : k$ for every kind k and $\alpha \in V$, where V is a set of variables.
2. $\sigma\,\tau : k'$ if $\sigma : k \to k'$ and $\tau : k$.
3. $\lambda\alpha^k.\sigma : k \to k'$ if $\sigma : k'$.
4. $\Pi\alpha^k.\sigma : *$ if $\sigma : *$.
5. $\sigma \to \tau : *$ if $\sigma, \tau : *$.

(iii) $\lambda\omega$ has *terms* $P, Q \in \mathrm{Term}(\lambda\omega)$:

$$P, Q ::= x \mid \lambda x.P \mid P\,Q \mid \lambda\alpha^k.P \mid P\,\sigma$$

(iv) The notion of reduction β on $\mathrm{Term}(\lambda\omega)$ and $\mathrm{Con}(\lambda\omega)$ is:

$$\begin{array}{lll} (\lambda\alpha^k.\tau)\,\sigma & \beta & \tau\{\alpha^k := \sigma\} \\ (\lambda x.P)\,Q & \beta & P\{x := Q\} \\ (\lambda\alpha^k.P)\,\sigma & \beta & P\{\alpha^k := \sigma\} \end{array}$$

(v) $\lambda\omega$ has *inference rules*:

$$\frac{}{\Gamma, x : \tau \vdash x : \tau} \qquad \frac{\Gamma, x : \sigma \vdash P : \tau}{\Gamma \vdash \lambda x.P : \sigma \to \tau} \qquad \frac{\Gamma \vdash P : \sigma \to \tau \quad \Gamma \vdash Q : \sigma}{\Gamma \vdash P\,Q : \tau}$$

$$\frac{\Gamma \vdash P : \sigma \quad \sigma =_\beta \tau}{\Gamma \vdash P : \tau} \qquad \frac{\Gamma \vdash P : \tau \quad \alpha \notin \mathrm{FV}(\Gamma)}{\Gamma \vdash \lambda\alpha^k.P : \Pi\alpha^k.\tau} \qquad \frac{\Gamma \vdash P : \Pi\alpha^k.\sigma \quad \tau : k}{\Gamma \vdash P\,\tau : \sigma\{\alpha^k := \tau\}}$$

(vi) A term M is *legal* if $\Gamma \vdash M : \sigma$ for some Γ, σ.

The following is inspired by [19, 2.2.16]; see also [23].

8.9. Definition. Let $\bot$ be a constructor of kind $*$, $\neg\sigma \equiv \sigma \to \bot$. Define maps $[\bullet], [\bullet]' : \mathrm{Con}(\lambda\omega) \to \mathrm{Con}(\lambda\omega)$ by:

$$\begin{array}{lcl} [\sigma] & = & \neg\neg[\sigma]' \\ \\ [\alpha^k]' & = & \alpha^k \\ [\sigma\,\tau]' & = & [\sigma]'\,[\tau]' \\ [\lambda\alpha^k.\sigma]' & = & \lambda\alpha^k.[\sigma]' \\ [\Pi\alpha^k.\sigma]' & = & \Pi\alpha^k.[\sigma] \\ [\sigma \to \tau]' & = & [\sigma] \to [\tau] \end{array}$$

8.10. Definition. Define $[\bullet] : \mathrm{Term}(\lambda\omega) \to \mathrm{Term}(\lambda\omega)$ by:

$$\begin{array}{lcl} [x] & = & \lambda k.x\,k \\ [\lambda x.P] & = & \lambda l.l\,\lambda x.\lambda h.y\,([P]\,h)\,x \\ [P\,Q] & = & \lambda l.[P]\,\lambda m.m\,[Q]\,l \\ [\lambda\alpha^k.P] & = & \lambda l.l\,\lambda\alpha^k.\lambda h.y\,([P]\,h)\,\alpha^k \\ [P\,\sigma] & = & \lambda l.[P]\,\lambda m.m\,[\sigma]'\,l \end{array}$$

8.11. Theorem. *$[M] \in \mathrm{WN}_\beta \Rightarrow M \in \mathrm{SN}_\beta$.*

Proof. Like the proof of Theorem 6.13. □

8.12. Lemma.

(i) $\sigma : k \Rightarrow [\sigma]' : k$.

(ii) $[\sigma]'\{\alpha := [\tau]'\} \equiv [\sigma\{\alpha := \tau\}]'$.

(iii) $\sigma =_\beta \tau \Rightarrow [\sigma] =_\beta [\tau]$.

(iv) $\Gamma \vdash M : \sigma \Rightarrow \Delta, [\Gamma] \vdash [M] : [\sigma]$, *where* $\mathrm{dom}(\Delta) = \mathrm{FV}([M])\backslash\mathrm{FV}(M)$.

Proof.

(i) Note that $\sigma : *$ implies $\neg\neg\sigma : *$ and use induction on $\sigma : k$.

(ii) By induction on σ.

(iii) Note that $\sigma =_\beta \tau$ implies $\neg\neg\sigma =_\beta \neg\neg\tau$, and prove by induction on $\sigma =_\beta \tau$ that $\sigma =_\beta \tau$ implies $[\sigma]' =_\beta [\tau]'$, using (ii).

(iv) By induction on $\Gamma \vdash M : \sigma$ using (i),(iii). □

Writing $\lambda\omega \models \mathrm{WN}_\beta$ to mean that all legal terms in $\lambda\omega$ are weakly normalizing, and similarly with SN_β, we have the following.

8.13. Corollary. *$\lambda\omega \models \mathrm{WN}_\beta \Rightarrow \lambda\omega \models \mathrm{SN}_\beta$.*

This shows that weak normalization of all terms implies strong normalization of all terms, but states nothing about *constructors*. However, the constructors of $\lambda\omega$ are essentially equivalent to the terms of $\lambda{\to}$ and this can be used to prove that weak normalization of all constructors implies strong normalization of all constructors.

9. Conclusion

We have shown that our extension of Klop's technique works on the calculi à la Curry $\lambda\!\rightarrow$, $\lambda\mu^{+}$, and $\lambda\subseteq$. In both $\lambda\mu^{+}$ and $\lambda\subseteq$, the smoothness of the proof stems from the fact that $\sim, \subseteq$ are *congruences*, and so in particular apply to types under negations. For other formulations of $\lambda\mu^{+}$ [70] and for the Curry systems $\lambda 2$ and $\lambda\cap^{-}$ [3] the straight-forward technique fails, because generalization and intersection introduction does not work under double negations.

We have also applied our extension to versions of $\lambda 2$ and $\lambda\omega$ à la Church. For dependent type systems our technique is limited by the fact that it is presently not clear how to express CPS-translations for dependent type systems, see, e.g., [11, 74]. Moreover, in such systems terms occur in types. To preserve typability the translation must map equal terms to equal terms, which does not hold with our CPS-translation. In the terminology of Section 6, the inner interpretation must be sound, not just permutative.

Acknowledgments. It is a pleasure to acknowledge my debt to Henk Barendregt for posing the problems dealt with in this paper, for comments to my ideas, and for his hospitality. The referees provided very valuable criticism and corrections. Thanks also to Paula Severi, Femke van Raamsdonk, Herman Geuvers, Ralph Loader, Gilles Barthe, John Hatcliff, and Zurab Khasidashvili for discussions and to the λ-group in Nijmegen and the TOPPS-group at DIKU for creating pleasent working environments.

References

[1] Z.M. Ariola, M. Felleisen, J. Maraist, M. Odersky, and P. Wadler. A call-by-need lambda-calculus. In *Conference Record of the Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 233–246. ACM Press, 1995.

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

[3] H.P. Barendregt. Lambda calculi with types. In S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume II, pages 117–309. Oxford University Press, 1992.

[4] H.P. Barendregt, J. Bergstra, J.W. Klop, and H. Volken. Degrees, reductions and representability in the lambda calculus. Technical Report Preprint 22, University of Utrecht, Department of Mathematics, 1976.

[5] E. Barendsen. *Types and Computations in Lambda Calculi and Graph Rewrite Systems.* PhD thesis, University of Nijmegen, 1995.

[6] G. Barthe and M.H. Sørensen. Domain-free pure type systems. In S. Adian and A. Nerode, editors, *Symposium on Logical Foundations of Computer Science*, volume 1234 of *Lecture Notes in Computer Science*, pages 9–20. Springer-Verlag, 1994.

[7] J.A. Bergstra and J.W. Klop. Strong normalization and perpetual reductions in the lambda calculus. *Journal of Information Processing and Cybernetics*, 18:403–417, 1982.

[8] M. Bezem and J.F. Groote, editors. *Typed Lambda Calculus and Applications*, volume 664 of *Lecture Notes in Computer Science.* Springer-Verlag, 1993.

[9] R. Bloo, F. Kammareddine, and R. Nederpelt. The Barendregt cube with definitions and generalised reduction. *Journal of Information and Computation*, 126(2):123–143, 1996.

[10] C. Consel and O. Danvy. For a better support of static data flow. In J. Hughes, editor, *Conference on Functional Programming and Computer Architecture*, volume 523 of *Lecture Notes in Computer Science*, pages 495–519. Springer-Verlag, 1991.

[11] T. Coquand and H. Herbelin. A-translation and looping combinators in pure type systems. *Journal of Functional Programming*, 4(1):77–88, 1994.

[12] H.B. Curry and R. Feys. *Combinatory Logic.* North-Holland, 1958.

[13] N.G. de Bruijn. A survey of the project Automath. In Seldin and Hindley [63], pages 579–606.

[14] P. de Groote. The conservation theorem revisited. In Bezem and Groote [8], pages 163–178.

[15] A. Filinski. Representing monads. In *Conference Record of the Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 446–457. ACM Press, 1994.

[16] J.H. Gallier. On Girard's "candidats de reductibilité". In P. Oddifreddi, editor, *Logic and Computer Science*, pages 123–203. Academic Press Limited, 1990.

[17] R.O. Gandy. An early proof of normalization by A.M. Turing. In Seldin and Hindley [63], pages 453–455.

[18] R.O. Gandy. Proofs of strong normalization. In Seldin and Hindley [63], pages 457–477.

[19] J.H. Geuvers. *Logics and Type Systems.* PhD thesis, University of Nijmegen, 1993.

[20] J.-Y. Girard. *Interprétation fonctionelle et élimination des coupures dans l'arithmétique d'ordre supérieur.* PhD thesis, Université Paris VII, 1972.

[21] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 1989.

[22] T.G. Griffin. A formulae-as-types notion of control. In *Conference Record of the Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 47–58. ACM Press, 1990.

[23] R. Harper and M. Lillibridge. Explicit polymorphism and CPS conversion. In *Conference Record of the Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 206–219. ACM Press, 1993.

[24] J.R. Hindley and J.P. Seldin. *Introduction to Combinators and λ-calculus.* Cambridge University Press, 1986.

[25] W. Howard. The formulae-as-types notion of construction. In Seldin and Hindley [63], pages 479–490.

[26] W. Howard. Ordinal analysis of terms of finite type. *Journal of Symbolic Logic*, 45(3):493–504, 1980.

[27] F. Kammareddine. A reduction relation for which postponement of k-contractions, conservation, and preservation of strong normalisation hold. Technical report, Glasgow University, 1996.

[28] F. Kammareddine and R. Nederpelt. A unified approach to type theory through a refined λ-calculus. *Theoretical Computer Science*, 136:183–216, 1994.

[29] F. Kammareddine and R. Nederpelt. Refining reduction in the lambda calculus. *Journal of Functional Programming*, 5(4):637–651, 1995.

[30] F. Kammareddine and R. Nederpelt. A useful λ-notation. *Theoretical Computer Science*, 155:85–109, 1996.

[31] F. Kammareddine and A. Ríos. Generalized β-reduction and explicit substitution. In H. Kuchen and D.S. Swierstra, editors, *Programming Languages: Implementations, Logics and Programs*, volume 1140 of *Lecture Notes in Computer Science*, pages 378–392. Springer-Verlag, 1996.

[32] M. Karr. "Delayability" in proofs of strong normalizability in the typed lambda calculus. In H. Ehrig, C. Floyd, M. Nivat, and J. Thatcher, editors, *Mathematical Foundations of Computer Software*, volume 185 of *Lecture Notes in Computer Science*, pages 208–222. Springer-Verlag, 1985.

[33] A. Kfoury and J. Tiuryn. Type reconstruction in finite-rank fragments of the second-order λ-calculus. *Journal of Information and Computation*, 98(2):228–257, 1992.

[34] A. Kfoury, J. Tiuryn, and P. Urzyczyn. An analysis of ML-typability. *Journal of the Association for Computing Machinery*, 41(2):368–398, 1994.

[35] A.J. Kfoury and J. Wells. A direct algorithm for type inference in the rank-2 fragment of second-order λ-calculus. In *ACM Conference on Lisp and Functional Programming*, 1994.

[36] A.J. Kfoury and J. Wells. New notions of reduction and non-semantic proofs of β-strong normalization in typed λ-calculi. Technical Report 94-01, Boston University Computer Science Department, 1994. Also in Logic in Computer Science, 1995.

[37] A.J. Kfoury and J. Wells. Addendum to "new notions of reduction and non-semantic proofs of β-strong normalization in typed λ-calculi". Technical Report 95-007, Boston University Computer Science Department, 1995.

[38] Z. Khasidashvili. *Form Reduction Systems and Reductions of Contracted Forms and Lambda-Terms.* PhD thesis, Tbilisi State University, 1988. In Russian.

[39] Z. Khasidashvili. The longest perpetual reductions in orthogonal expression reduction systems. In A. Nerode and Yu. V. Matiyasevich, editors, *Symposium on Logical Foundations of Computer Science*, volume 813 of *Lecture Notes in Computer Science*, pages 191–203. Springer-Verlag, 1994.

[40] Z. Khasidashvili. Perpetuality and strong normalization in orthogonal term rewriting systems. In P. Enjalbert, E.W. Mayr, and K.W. Wagner, editors, *Symposium on Theoretical Aspects of Computer Science*, volume 775 of *Lecture Notes in Computer Science*, pages 163–174. Springer-Verlag, 1994.

[41] J.W. Klop. *Combinatory Reduction Systems.* PhD thesis, Utrecht University, 1980. Volume 127 of CWI Tracts, Amsterdam.

[42] J.W. Klop, V. van Oostrom, and F. van Raamsdonk. Combinatory reduction systems, introduction and survey. *Theoretical Computer Science*, 121(1-2):279–308, 1993. Special issue in honour of Corrado Böhm.

[43] J.-L. Krivine. *Lambda-Calculus, Types and Models.* Ellis Horwood Series in Computers and their Applications. Masson and Ellis Horwood, English Edition, 1993.

[44] D. Leivant. Syntactic translations and provably recursive functions. *Journal of Symbolic Logic*, 50(3):682–688, 1985.

[45] R. Loader. Normalisation by translation. Presented at the BRA TYPES workshop, Turin, 1995.

[46] A.R. Meyer and M. Wand. Continuation semantics in typed lambda-calculi (summary). In R. Parikh, editor, *Logics of Programs*, volume 193 of *Lecture Notes in Computer Science*, pages 219–224. Springer-Verlag, 1985.

[47] J.C. Mitchell. Type inference and simplle subtypes. *Journal of Functional Programming*, 1(3):245–285, 1991.

[48] E. Moggi. Computational lambda-calculus and monads. In *Logic in Computer Science*, pages 14–23. IEEE Computer Society Press, 1989.

[49] E. Moggi. Notions of computation and monads. *Journal of Information and Computation*, 93:55–92, 1991.

[50] R. Nederpelt. *Strong normalization for a typed lambda calculus with lambda structured types.* PhD thesis, Eindhoven, 1973.

[51] G. Plotkin. Call-by-name, call-by-value and the λ-calculus. *Theoretical Computer Science*, 1:125–159, 1975.

[52] J. van de Pol. Termination proofs for higher-order rewrite systems. In J. Heering et al., editor, *Higher Order Algebra, Logic and Term Rewriting*, volume 816 of *Lecture Notes in Computer Science*, pages 305–325. Springer-Verlag, 1994.

[53] J. van de Pol. Two *different* strong normalization proofs? In G. Dowek, J. Heering, K. Meinke, and B. Möller, editors, *Higher Order Algebra, Logic and Term Rewriting*, volume 1074 of *Lecture Notes in Computer Science*, pages 201–220. Springer-Verlag, 1996.

[54] J. van de Pol and H. Schwichtenberg. Strict functionals for termination proofs. In M. Dezani-Ciancaglini and G. Plotkin, editors, *Typed Lambda Calculus and Applications*, volume 902 of *Lecture Notes in Computer Science*, pages 350–364. Springer-Verlag, 1995.

[55] D. Prawitz. *Natural Deduction: A proof theoretical study.* Almquist & Wiksell, 1965.

[56] F. van Raamsdonk and P. Severi. On normalisation. Technical Report CS-R9545, CWI, 1995.

[57] L. Regnier. Une équivalence sur les lambda-termes. *Theoretical Computer Science*, 126:281–292, 1994.

[58] J.C. Reynolds. The discoveries of continuations. *LISP and Symbolic Computation*, 6:233–248, 1993.

[59] A. Sabry. A reflection on call-by-value. In *International Conference on Functional Programming*, pages 13–24. ACM Press, 1996.

[60] A. Sabry and M. Felleisen. Reasoning about programs in continuation-passing style. *Lisp and Symbolic Computation*, 6:289–360, 1993.

[61] H. Schwichtenberg. Complexity of normalization in the pure typed lambda-calculus. In A.S. Troelstra and D. van Dalen, editors, *The L.E.J. Brouwer Centenary Symposium*, pages 453–457. North-Holland, 1982.

[62] H. Schwichtenberg. An upper bound for reduction sequences in the typed lambda-calculus. *Archive for Mathematical Logic*, 30:405–408, 1991.

[63] J.P. Seldin and J.R. Hindley, editors. *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism.* Academic Press Limited, 1980.

[64] M.H. Sørensen. Effective longest and infinite reduction paths in untyped λ-calculi. In H. Kirchner, editor, *Colloquium on Trees in Algebra and Programming*, volume 1059 of *Lecture Notes in Computer Science*, pages 287–301. Springer-Verlag, 1996.

[65] M.H. Sørensen. Properties of infinite reduction paths in untyped λ-calculus. In *Tbilisi Symposium on Language, Logic, and Computation*, CLSI Lecture Notes, 1996. To appear.

[66] J. Springintveld. Lower and upper bounds for reductions of types in $\lambda\underline{\omega}$ and λP. In Bezem and Groote [8], pages 391–405.

[67] W.W. Tait. Intensional interpretations of functionals of finite type I. *Journal of Symbolic Logic*, 32(2):190–212, 1967.

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

[69] A.S. Troelstra. *Metamathematical Investigation of Intuitionistic Arithmetic and Analysis*, volume 344 of *Lecture Notes in Mathematics.* Springer-Verlag, 1973.

[70] P. Urzyczyn. Positive recursive type assigment. In J. Wiedermann and P. Hájek, editors, *Mathematical Foundations of Computer Science*, volume 969 of *Lecture Notes in Computer Science*, pages 382–391. Springer-Verlag, 1995.

[71] D. Vidal. *Nouvelles Notions de Réduction en Lambda Calcul.* PhD thesis, Université de Nancy, 1989.

[72] R.C. de Vrijer. A direct proof of the finite developments theorem. *Journal of Symbolic Logic*, 50:339–343, 1985.

[73] R.C. de Vrijer. *Surjective Pairing and Strong Normalization: Two Themes in Lambda Calculus.* PhD thesis, University of Amsterdam, 1987.

[74] B. Werner. Continuations, evaluation styles and types systems. Manuscript, 1992.

[75] H. Xi. An induction measure on λ-terms and its applications. Research Report 96-192, Department of Mathematical Sciences, Carnegie Mellon University, 1996.

[76] H. Xi. On weak and strong normalisations. Research Report 96-187, Department of Mathematical Sciences, Carnegie Mellon University, 1996.