

New Notions of Reduction and Non-Semantic Proofs of Strong β -Normalization in Typed λ -Calculi*

A. J. Kfoury
kfoury@cs.bu.edu
Dept. of Computer Science
Boston University

J. B. Wells
jbw@cs.bu.edu
Dept. of Computer Science
Boston University

Abstract

Two notions of reduction for terms of the λ -calculus are introduced and the question of whether a λ -term is β -strongly normalizing is reduced to the question of whether a λ -term is merely normalizing under one of the notions of reduction. This gives a method to prove strong β -normalization for typed λ -calculi. Instead of the usual semantic proof style based on Tait’s realizability or Girard’s “candidats de réductibilité”, termination can be proved using a decreasing metric over a well-founded ordering. This proof method is applied to the simply-typed λ -calculus and the system of intersection types, giving the first non-semantic proof for a polymorphic extension of the λ -calculus.

1 Introduction

Background and Motivation.

The problem of strong normalization of β -reduction (β -SN) has been actively considered for various typed λ -calculi for over 25 years. Tait first proved that all λ -terms typable in the simply-typed λ -calculus (actually, Gödel’s system **T**) are β -SN using a semantic method called “realizability” or “solvability” [Tai67]. Extending Tait’s method, Girard devised the powerful semantic method of “reducibility candidates” (“candidats de réductibilité”) to prove β -SN for systems **F** and **F ω** [Gir71, Gir72]. All later β -SN proofs for system **F** have relied on reducibility candidates, being essentially variations on Girard’s original proof, simplifying or reformulating or cleaning up many of the concepts. (Gallier’s paper [Gal90] is a useful comparative study of all proofs for system **F** published until 1990.) All of these proofs share a certain element of non-obviousness. As Gallier writes [Gal94]:

Reducibility proofs are seductive and thrilling, but also elusive. Following these proofs step-by-step, we see that they “work” (when they are not wrong!), but I claim that most of us would still admit that they are not sure *why* these proofs work!

The proofs of the β -SN property for other polymorphic type systems have also relied on the semantic methods of either Tait or Girard. Although, as originally formulated, both the system of recursive types and the system of intersection types admit non- β -SN λ -terms, both of these systems have useful restrictions which satisfy the β -SN property. Using the method of reducibility candidates, Mendler showed that the system of positive-recursive types has the β -SN property [Men91]. The system of intersection types was introduced by Coppo and Dezani [CDC80, CDCV81] with two important variants, one of which is an extension of the other. Pottinger [Pot80] and Leivant [Lei86] used reducibility candidates to show the simpler system of intersection types without the special ω type constant has the β -SN property (in fact it turns out to type exactly the β -SN λ -terms). A more recent β -SN result by van Bakel uses Tait’s method without requiring the reducibility candidates [vB92]. Until now, all β -SN proofs for these polymorphic type systems have used semantic methods.

In addition to the Tait/Girard paradigm, other methods have been used to show the β -SN property for the simply-typed λ -calculus. One style, first used by Gandy and later refined by de Vrijer and by van de Pol and Schwichtenberg, involves associating functionals with each subterm [Gan80b, dV87, vdPS95]. Although these proofs are presented in a somewhat semantic style, it appears that they can be reformulated in a more syntactic “symbol-pushing” style.

Another style of proof involves converting the β -SN question into a question of weak normalization, which merely asks whether some reduction sequence termi-

*This work is partly supported by NSF grant CCR-9113196.

nates, not whether all of them do. This is the style used in this paper. The first proof in this style was by Nederpelt for a system equivalent to the simply-typed λ -calculus [Ned73]. Klop devised a variant of this proof for the simply-typed λ -calculus [Klo80, Chap. 1, § 8] and then created a more general method which works for many “combinatory reduction systems” (sometimes called “higher-order rewrite systems”, unrelated to the polymorphic extensions of the λ -calculus) [Klo80, Chap. 2]. More recently, de Groote devised another variant of this method, which is very close to the method we use [dG93]. The β -SN proofs for the simply-typed λ -calculus yielded by this style have been very non-semantic. In de Groote’s paper, he describes a way to use Scedrov’s simplification of the conditions on reducibility candidates, which were designed for a weak normalization proof, to prove strong normalization for system **F** [dG93, Sce87]. In addition to those we have mentioned, other methods have been used to prove β -SN for the simply-typed λ -calculus.

Contributions of This Paper.

We present a strictly proof-theoretic method for proving the β -SN property which works for the simply-typed λ -calculus and the system of intersection types, relying on simple combinatorial properties of β -reduction and type-inference systems. The method consists of two parts:

1. The question of whether a λ -term is β -strongly normalizing is converted to the question of whether the λ -term is normalizing under a new notion of reduction, \star -reduction.
2. The \star -normalization of all λ -terms typable in certain typed λ -calculi is established using a decreasing metric. For didactic reasons, we first apply our method to the simply-typed λ -calculus. The proof for simple types then extends in a simple way to the system of intersection types.

The first half of our proof is in the style initiated by Nederpelt for converting an SN problem into a weak normalization problem. In Section 3, we define \star -reduction, mentioned above, which is itself based on another recent notion called γ -reduction. Essentially, γ -reduction is a simple size-preserving transformation that reorganizes λ -bindings in a λ -term without changing the “meaning” of the λ -term. The notion of γ -reduction can be seen as “raising” a λ -abstraction outside of an enclosing β -redex. \star -reduction combines a β -reduction step of an I -redex followed by reduction to γ -normal form to bypass K -redexes. This behavior leads to the results that \star -reduction preserves the β -SN property and every \star -normal form is β -SN. From

this, it can be seen that \star -normalization implies β -strong normalization. Thus, to prove the β -SN property, that *every* possible β -reduction sequence must terminate, it is sufficient to show the \star -normalization property, that there is *some* \star -reduction sequence that terminates.

For the second part, in Section 4, we show that if a λ -term M is typable in the simply-typed λ -calculus (or in the intersection-type discipline in Section 5) then we can devise a \star -reduction strategy from M and attach a particular well-founded partial ordering to it that guarantees the reduction strategy must terminate (normalize)—thus implying β -SN by the first part. This is done in the style of decreasing-metric termination proofs often found in the term rewriting literature and is very similar to the original weak normalization proof for the simply-typed λ -calculus by Turing [Gan80a]. The required reduction strategy is very simple: just reduce innermost I -redexes.

The new proof method for proving β -SN which we present is important for more than one reason. Most importantly, this is the first β -SN proof for a polymorphic extension of the typed λ -calculus which does not depend on the methods of Tait or Girard. As a secondary reason, we feel the new method compares well in understandability with previous proof methods.

Future Work.

The two typed λ -calculi for which we have carried our method all the way through give us reason for optimism regarding applying the method to other typed λ -calculi. In a sense, simple types and intersection types correspond respectively to a minimal and a maximal type discipline for which β -SN holds; every type system in use includes the simple types, while the intersection-type discipline can derive a type for every β -SN λ -term. Hence, given any other higher-order typed λ -calculus for which we are interested in proving β -SN—such as System **F** or some of its restrictions or extensions—there are two ways of proceeding. The direct way is to attach a well-founded partial ordering to a \star -reduction sequence, guaranteeing its termination. The indirect way is to translate an arbitrary derivation in the given type system into a derivation in the intersection-type discipline for the same untyped λ -term, without making use of the already-known fact that the type system types only β -SN λ -terms. We are particularly interested in system **F** (and certain restrictions and extensions of **F**) and in the positive-recursive-type discipline. Although it is well-known that the β -SN property of system **F** can not be proven within second-order Peano Arith-

metic [GLT89, p. 114], thus casting doubts on whether semantic notions can be avoided, it would be nice to carry out a proof that does not rely on reducibility candidates. In subsequent reports we wish to examine the β -SN property for these systems.

Related Research.

P. de Groote’s 1993 paper [dG93] uses a method for reducing the β -SN problem to a weak normalization problem that is the same as ours in spirit but differs in the details. Instead of γ -reduction, de Groote uses a reduction he calls β_S :

$$(((\lambda x.M)N)O) \xrightarrow{\beta_S} ((\lambda x.MO)N) \quad \text{if } x \notin M$$

Both our paper and de Groote’s paper achieve the nearly identical result that the problem of β -strong normalization is equivalent (respectively) to the problems of \star -normalization and $\beta_I\beta_S$ -normalization. The most important difference is that de Groote uses general $\beta_I\beta_S$ -reduction while we take advantage of a specific reduction strategy of $\beta_I\gamma$ -reduction which we call \star -reduction. First, de Groote shows that β_K -reduction steps can be postponed in a sequence of β -reduction and β_S -reduction steps, yielding the fact that if a $\beta_I\beta_S$ -descendent is β -SN, then the ancestor is β -SN as well. Then de Groote defines a calculus with labels to record the number of $\beta_I\beta_S$ -reduction steps that have occurred. A complex argument shows this calculus to be confluent. Since the sum of the labels in a term is a bound on the longest reduction sequence leading to that term, and since all reduction paths from a term with a normal form must eventually reach the normal form, this yields the desired result.

For a further discussion of differences in how our method’s are applied to various typed λ -calculi, see the addendum to our technical report [KW95]. This addendum also contains a summary of other research into the γ and β_S notions of reduction and a comparison of both our and de Groote’s methods with the earlier method of Klop.

Acknowledgements.

Pawel Urzyczyn spotted a wrong proof in the technical report and suggested many shorter proofs. Pawel Urzyczyn, Femke van Raamsdonk, Vincent van Oostrom, Matthias Felleisen, and Philippe de Groote mentioned much of the related research to us.

2 The Untyped λ -Calculus

In this section we present our definitions, notation, and nomenclature for standard concepts of the

untyped λ -calculus. Our notation generally follows Barendregt’s [Bar84].

2.1 λ -Terms.

The set of all λ -terms Λ is built from the countably infinite set of λ -term variables \mathcal{V} using application and λ -abstraction as specified by the usual grammar $\Lambda ::= \mathcal{V} \mid (\Lambda \Lambda) \mid (\lambda \mathcal{V}. \Lambda)$. We assume at all times that every λ -term M obeys the restriction that no variable is λ -bound more than once and no variable occurs both λ -bound and free in M . We assume α -conversion is used when necessary to make this happen.

A *context* $C[\]$ is a λ -term with one hole (sometimes more than one) and if M is a λ -term then $C[M]$ denotes the result of inserting M into the hole in $C[\]$, *including* the capture of free variables in M by the λ -bound variables of $C[\]$. Unless specified otherwise, a context has only one hole. If M and N are λ -terms, then $M \equiv N$ means that M and N are identical after allowing α -conversion. $N \subset M$ denotes that N is a proper subterm of M and $N \subseteq M$ includes the possibility that $N \equiv M$.

2.2 Reduction.

Our notation on reduction generally follows Barendregt’s [Bar84, § 3.1, p. 50–59] with some minor differences. A *reduction relation* \mathcal{R} is a set of pairs of λ -terms. If $(M, N) \in \mathcal{R}$, then we say that M is an \mathcal{R} -redex and N is its *contractum*. If $C[\]$ is a context and $(M, N) \in \mathcal{R}$, then $(C[M], C[N])$ is in the contextual closure of \mathcal{R} and we say that $C[M]$ \mathcal{R} -reduces to $C[N]$ via the redex M and we write this as $C[M] \xrightarrow{\mathcal{R}} C[N]$. If M \mathcal{R} -reduces to N by some unspecified redex, we write this as $M \xrightarrow{\mathcal{R}} N$. If $M \xrightarrow{\mathcal{R}} N$, we say that N is a \mathcal{R} -reduct of M . The transitive, reflexive closure of “ $\xrightarrow{\mathcal{R}}$ ” is written as “ $\xrightarrow{\mathcal{R}^*}$ ”.

An \mathcal{R} -normal form is a λ -term containing no \mathcal{R} -redexes which therefore does not \mathcal{R} -reduce to any other term. If $M \xrightarrow{\mathcal{R}^*} N$ and N is a \mathcal{R} -normal form, we say that M \mathcal{R} -normalizes and that N is a \mathcal{R} -normal form of M . If M has only one \mathcal{R} -normal form N , this is denoted by $\mathcal{R}\text{-nf}(M) = N$ or by $M \xrightarrow{\mathcal{R}\text{-nf}} N$. If there are no infinite \mathcal{R} -reduction sequences starting from M , then M is \mathcal{R} -strongly normalizing, also written as \mathcal{R} -SN. A λ -term M is \mathcal{R} -infinite, denoted $\mathcal{R}\text{-}\infty(M)$, if and only if M is not \mathcal{R} -SN.

The standard notion of reduction, β -reduction, is of course the least relation such that:

$$((\lambda x.P)Q) \xrightarrow{\beta} P[x := Q]$$

It is well-known that β -reduction is confluent (Church-Rosser) and that β -normal forms are unique.

3 β -Strong Normalization and \star -Normalization

In this section, we introduce two new notions of reduction, γ -reduction and \star -reduction (a combination of γ -reduction and β -reduction), which are used throughout the rest of the paper. These notions of reduction transform λ -terms in ways that are easier to analyze than β -reduction. The main result of this section is Theorem 3.11 which implies that the question of β -strong normalization can be reduced to the question of \star -normalization. Subsequent sections will then show for certain typed λ -calculi that all typable terms have \star -normal forms, implying that all typable terms are β -strongly normalizing.

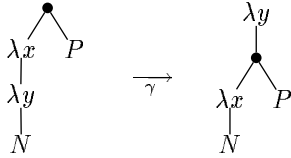
See the technical report version of the present paper for full details if necessary [KW94].

3.1 γ -Reduction and γ -Normal Forms.

Definition 3.1 γ -reduction is the least reduction relation such that:

$$((\lambda x.(\lambda y.N))P) \xrightarrow{\gamma} (\lambda y.((\lambda x.N)P))$$

We assume that $x \neq y$ and $y \notin \text{FV}(P)$, using α -conversion if necessary. In a pictorial format, this reduction looks like this:



Lemma 3.2 For every $M \in \Lambda$:

1. M is γ -SN.
2. M has a unique γ -normal form.

Proof: The claims are proved separately.

1. Count the number of pairs of subterm occurrences P and Q in M such that P is an application, Q is an abstraction, P contains Q , and there is no subterm (RS) contained within P such that Q is contained within S . Every γ -reduction step reduces this count.
2. First, we show that γ -reduction is weakly confluent (weakly Church-Rosser):

$$\begin{array}{ccc} M & \xrightarrow{\Delta} & N \\ \gamma \downarrow \Gamma & & \gamma \downarrow \Gamma \\ P & \xrightarrow{\gamma} & Q \end{array}$$

This is done by an analysis of the possible relationships between the γ -redexes Δ and Γ . Since γ -reduction is both strongly normalizing and weakly confluent, it is confluent, from which our claim follows.

■

It is easy to give an inductive definition of those λ -terms which happen to be in γ -normal form. The set Λ^γ of γ -normal forms is defined inductively as the least set satisfying these conditions:

1. $x \in \Lambda^\gamma$ if $x \in \mathcal{V}$.
2. $(MN) \in \Lambda^\gamma$ if $M, N \in \Lambda^\gamma$ and M is not a λ -abstraction.
3. $(\lambda x.M) \in \Lambda^\gamma$ if $M \in \Lambda^\gamma$ and $x \in \mathcal{V}$.
4. $((\lambda x.M)N) \in \Lambda^\gamma$ if $M, N \in \Lambda^\gamma$, M is not a λ -abstraction, and $x \in \mathcal{V}$.

Lemma 3.3 A λ -term M is in γ -nf if and only if $M \in \Lambda^\gamma$.

We will use notions of residuals relative to both β -reduction and γ -reduction of three different kinds of subterm: arbitrary subterms, β -redexes, and γ -redexes. We will use natural numbers as indexes to mark these subterms. The notation we use is in the style of Barendregt [Bar84, § 11.1.2, p. 279 and § 11.2.4, p. 284]. These three types of subterms may be marked with an index $i \in \mathbb{N}$ as follows:

1. An arbitrary subterm $N \subseteq M$ is marked as N^i .
2. For a β -redex $((\lambda x.P)Q)$, we mark its leading λ with the index in subscript position, for example $((\lambda_i x.P)Q)$.
3. For a γ -redex $((\lambda x.\lambda y.N)P)$ we also mark the leading λ of the γ -redex with an index, but this time in superscript position, for example $((\lambda^j x.\lambda y.N)P)$.

It will be possible for the same λ to be marked as part of both a β -redex and a γ -redex, in which case it will have both a subscript and a superscript.

The notions of reduction β and γ are extended to marked terms in the following manner. The notation $[i]$ means the index i may or may not be present, but if it is present in one occurrence of $[i]$ it is present in

all others.

$$((\lambda_{[i]}^{[j]}x.M^{[k]})^{[l]}N^{[m]})^{[n]} \xrightarrow{\beta} M^{[k]}[x := N^{[m]}]$$

$$\begin{aligned} & ((\lambda_{[i]}^{[j]}x.(\lambda y.M^{[k]})^{[l]}N^{[m]})^{[n]})^{[o]} \\ & \quad \downarrow \gamma \\ & (\lambda y.((\lambda_{[i]}^{[j]}x.M^{[k]})^{[m]}N^{[n]})^{[o]})^{[l]} \end{aligned}$$

It is important to notice that β -reduction of a redex that is both a marked β -redex and a marked γ -redex will erase both markings, while γ -reduction will erase only the marking of the γ -redex, preserving the marking of the β -redex. At this point, the notion of *residual* is defined from the marking with indices in the standard way.

A subterm occurrence N in M is *passive* if $N \equiv M$ or N occurs as $(PN) \subset M$ for some P or N occurs as $(\lambda x.N)$ where $(\lambda x.N)$ is passive, otherwise N is *active*. (Note that this definition is different from [Bar84, § 2.1.8 (iv), p. 25].)

Lemma 3.4 γ -reduction has the following properties. Let $M \xrightarrow{\gamma} N$.

1. Every λ -abstraction in M has exactly one residual in N and every λ -abstraction in N is the residual of a λ -abstraction in M .
2. If a λ -abstraction in M is the function of a β -redex, then its residual in N is also the function of a β -redex.
3. If a λ -abstraction in M is passive, then its residual in N is also passive.
4. If a λ -abstraction in N is the function of β -redex, then it is the residual of an active λ -abstraction in M .
5. If a λ -term is in γ -normal form, then all of its active λ -abstractions are functions of β -redexes.

3.2 γ -Reduction and β -Strong Normalization.

In this subsection, we show that if the result of γ -reduction is β -SN, then the input must also have been β -SN. (We could show that γ -reduction preserves the β -SN property, but we will not need such a general result later.)

Lemma 3.5 For every $M \in \Lambda$ it is the case that:

$$\begin{array}{ccc} M & \xrightarrow{\gamma} & \cdot \\ \beta \downarrow & & \beta \downarrow \\ \cdot & \xrightarrow{\gamma} & \cdot \end{array}$$

Proof: Analyzing the different possible relationships between a γ -redex and a β -redex yields:

$$\begin{array}{ccc} M & \xrightarrow{\gamma} & \cdot \\ \beta \downarrow & & \beta \downarrow \\ \cdot & \xrightarrow{\gamma} & \cdot \end{array}$$

Diagram chasing then produces the desired conclusion:

$$\begin{array}{ccccccc} M & \xrightarrow{\gamma} & \cdot & \xrightarrow{\gamma} & \cdot & \xrightarrow{\gamma} & \cdot \\ \beta \downarrow & & \beta \downarrow & & \beta \downarrow & & \beta \downarrow \\ \cdot & \xrightarrow{\gamma} & \cdot & \xrightarrow{\gamma} & \cdot & \xrightarrow{\gamma} & \cdot \end{array}$$

■

Lemma 3.6 For every $M \in \Lambda$, if $\gamma\text{-nf}(M)$ is β -SN then M is β -SN.

(We claim that the converse of Lemma 3.6 is also true. However, it requires a more subtle argument and it is not needed in this paper.)

Proof: Let $M' = \gamma\text{-nf}(M)$. We now prove that if $\beta\text{-}\infty(M)$ then $\beta\text{-}\infty(M')$, which is logically equivalent to the claim of the lemma. Suppose σ were an infinite β -reduction from M . Using Lemma 3.5 allows the erection of an infinite β -reduction σ' from M' :

$$\begin{array}{ccccccc} \sigma : & M & \xrightarrow{\beta} & \cdot & \xrightarrow{\beta} & \cdot & \xrightarrow{\beta} & \cdot & \xrightarrow{\beta} & \cdot & \dots \\ & \gamma\text{-nf} \downarrow & & \gamma \downarrow & & \gamma \downarrow & & \gamma \downarrow & & \gamma \downarrow & \\ \sigma' : & M' & \xrightarrow{\beta} & \cdot & \xrightarrow{\beta} & \cdot & \xrightarrow{\beta} & \cdot & \xrightarrow{\beta} & \cdot & \dots \end{array}$$

■

3.3 Preservation of β -Strong Normalization by β_I -Reduction.

Let Δ be the β -redex $((\lambda x.P)Q)$. If $x \in \text{FV}(P)$ then Δ is an *I-redex*. Otherwise, if $x \notin \text{FV}(P)$ then Δ is a *K-redex*. (Following [Bar84, § 11.3.6, p. 296].) When β -reduction is restricted to *I*-redexes (respectively *K*-redexes), we call it β_I -reduction (respectively β_K -reduction).

Lemma 3.7 Let $M \xrightarrow{\beta} N$ where Δ is an *I*-redex. Then M is β -SN if and only if N is β -SN.

Proof: This is a consequence of the Conservation Theorem [Bar84, § 13.4.12, p. 343]. ■

3.4 \star -Reduction.

In this subsection, we define \star -reduction, a combination of γ -reduction and β -reduction. We then prove the major result of Theorem 3.11, showing that the question of β -strong normalization can be reduced to the question of \star -normalization. The importance of this result is the fact that it is easier to prove a normalization result than a strong normalization result, because the reduction strategy can be chosen.

Definition 3.8 For two terms $M, N \in \Lambda^\gamma$, we define $M \xrightarrow{\star} N$ to hold if there is an I -redex Γ in M and a term $M' \in \Lambda$ such that:

$$M \xrightarrow[\beta]{\Gamma} M' \xrightarrow[\gamma\text{-nf}]{} N$$

Lemma 3.9 For all $M, N \in \Lambda^\gamma$, if $M \xrightarrow{\star} N$ and N is β -SN, then M is also β -SN.

Proof: It is sufficient to show the claim for a single \star -reduction step: if $M \xrightarrow{\star} N$ and N is β -SN, then M is also β -SN. This is a consequence of Lemma 3.7 (for the reduction $M \xrightarrow[\beta]{\Delta} M'$ where Δ is an I -redex) and Lemma 3.6 (for the reduction $M' \xrightarrow[\gamma\text{-nf}]{} N$). ■

Lemma 3.10 Let $M \in \Lambda^\gamma$ be a term containing no I -redexes. Then M is β -SN.

Proof: Reducing a K -redex in a term that is in γ -normal form and β_I -normal form cannot produce I -redexes or γ -redexes. Thus, any β -reduction sequence from M reduces only K -redexes. Reducing K -redexes is strictly size-decreasing, so any β -reduction sequence from M must terminate. ■

Theorem 3.11 For any term $M \in \Lambda$, if $\gamma\text{-nf}(M)$ is \star -normalizing (there is at least one \star -reduction from $\gamma\text{-nf}(M)$ which terminates) then M is β -SN.

(We claim the converse of Theorem 3.11 is also true, but do not prove it and do not need it.)

Proof: If $\gamma\text{-nf}(M)$ is \star -normalizing, then by Lemma 3.9 and Lemma 3.10 (since a \star -normal form belongs to Λ^γ and has no I -redexes) it holds that $\gamma\text{-nf}(M)$ is β -SN. By Lemma 3.6 we conclude that M is β -SN. ■

4 The \star -Normalization of the Simply-Typed λ -Calculus

In this section, we prove that every simply-typed λ -term is β -SN. This is a new proof for a well-known result and it is probably not any simpler than many of the other proofs in the literature already. This proof is presented to allow the reader to understand our

method in the simpler context of the simply-typed λ -calculus. In Section 5, we will generalize this proof to the more complicated intersection-type discipline.

4.1 The Simply-Typed λ -Calculus.

In this paper, it is convenient to define the simply-typed λ -calculus in an explicitly-typed manner, where every subterm and bound variable of a typed λ -term is annotated with an explicit type, written in superscript position for convenience. (This can be called “Church” style.)

The set of simple types \mathbb{T} is built from the countably infinite set of type variables \mathbb{V} using the “ \rightarrow ” type constructor as specified by the grammar $\mathbb{T} ::= \mathbb{V} \mid (\mathbb{T} \rightarrow \mathbb{T})$. A type is therefore either a *type variable* or a *\rightarrow -type*. Small Greek letters from the beginning of the alphabet (e.g. $\alpha, \beta, \gamma, \delta$) are metavariables over \mathbb{V} and small Greek letters towards the end of the alphabet (e.g. σ and τ) are metavariables over \mathbb{T} . When writing types, the arrows associate to the right so that $\sigma \rightarrow \tau \rightarrow \rho = \sigma \rightarrow (\tau \rightarrow \rho)$.

The λ -term variables are pairs of untyped variables and types, written as x^σ, y^τ, z^ρ , and so on. Instead of using type assignments (sometimes called contexts or environments), we require every typed λ -terms M to satisfy the property that:

(†) For all x^σ, y^τ in M , if $x = y$ then $\sigma = \tau$

The set Λ^\rightarrow of simply-typed λ -terms and a type-erasing function $|\cdot|$ from Λ^\rightarrow to Λ are defined inductively as follows:

1. $x^\sigma \in \Lambda^\rightarrow$ and $|x^\sigma| = x$ if $x \in \mathcal{V}$ and $\sigma \in \mathbb{T}$.
2. $(M^{\sigma \rightarrow \tau} N^\sigma)^\tau \in \Lambda^\rightarrow$ and $|(M^{\sigma \rightarrow \tau} N^\sigma)^\tau| = (|M^{\sigma \rightarrow \tau}| |N^\sigma|)$ if $M^{\sigma \rightarrow \tau} \in \Lambda^\rightarrow$ and $N^\sigma \in \Lambda^\rightarrow$.
3. $(\lambda x^\sigma. M^\tau)^{\sigma \rightarrow \tau} \in \Lambda^\rightarrow$ and $|(\lambda x^\sigma. M^\tau)^{\sigma \rightarrow \tau}| = (\lambda x. |M^\tau|)$ if $M^\tau \in \Lambda^\rightarrow$ and $\sigma \in \mathbb{T}$.

Provided all of the free and bound variables in a λ -term are annotated with types, the types annotating applications and λ -abstractions may be omitted with no loss of information.

We choose to present the simply-typed λ -terms in a “Church” style rather than a “Curry” style partly because this gives a natural interpretation for β -reduction and γ -reduction. In the Church style, using the natural extension of β -reduction to the simply-typed λ -calculus, a β -reduct or a γ -reduct of M automatically inherits a simple-typing from M . This will prove to be vital for our purposes. In the Curry style, if M is typable and $M \xrightarrow[\beta]{} N$, then N is also typable, but a mechanism must be defined to construct the typing for N from the typing for M . Also, if $M \xrightarrow[\beta]{} N$,

the reduct N may have typings that are not necessarily derived from the typing for M .

We now define explicitly how β -reduction and γ -reduction work on simply-typed λ -terms. If $M \in \Lambda^\rightarrow$ and Δ is a β -redex occurrence in M such that

$$\Delta \equiv ((\lambda x^\sigma . P^\tau)^{\sigma \rightarrow \tau} Q^\sigma)^\tau$$

and $M \equiv C[\Delta]$ where $C[\]$ is a context with exactly one hole, then:

$$M \xrightarrow{\Delta/\beta} C[P^\tau[x^\sigma := Q^\sigma]]$$

If Γ is a γ -redex occurrence in M such that

$$\Gamma \equiv ((\lambda x^\sigma . (\lambda y^\rho . N^\tau)^{\rho \rightarrow \tau})^{\sigma \rightarrow \rho \rightarrow \tau} P^\sigma)^{\rho \rightarrow \tau}$$

and $M \equiv D[\Gamma]$ where $D[\]$ is a context with exactly one hole, then:

$$M \xrightarrow{\Gamma/\gamma} D[(\lambda y^\rho . ((\lambda x^\sigma . N^\tau)^{\sigma \rightarrow \tau} P^\sigma)^\tau)^{\rho \rightarrow \tau}]$$

4.2 A Metric on Simply-Typed λ -Terms.

The proof for \star -normalization later in this section uses a metric $order^\bullet$ on λ -terms that decreases after each reduction step. This metric is defined on the types involved in the I -redexes in the λ -term.

For simple types, define the function $order$ inductively as follows:

1. $order(\alpha) = 0$ where $\alpha \in \mathbb{V}$ is a type variable.
2. $order(\sigma \rightarrow \tau) = \max\{1 + order(\sigma), order(\tau)\}$.

Let Δ be a simply-typed β -redex so that:

$$\Delta \equiv ((\lambda x^\sigma . P^\tau)^{\sigma \rightarrow \tau} Q^\sigma)^\tau$$

Define $order(\Delta) = order(\sigma)$. Let M be a simply-typed λ -term. Let the set of all I -redex occurrences in M be $\{\Delta_1, \dots, \Delta_n\}$. Define the function $order^\bullet$ from λ -terms to multisets over \mathbb{N} so that:

$$order^\bullet(M) = \{order(\Delta_1), \dots, order(\Delta_n)\}$$

Thus, for any λ -term M , $order^\bullet(M)$ is a finite multiset of natural numbers. Observe that K -redexes do not contribute to the value of $order^\bullet(M)$.

4.3 A Well-Founded Multiset Ordering.

Since the metric $order^\bullet$ computes multisets of natural numbers instead of just single natural numbers, we can not use the simple, numeric “ $<$ ” ordering. However, there is a standard multiset ordering which is suitable.

For S a multiset over \mathbb{N} and $n \in \mathbb{N}$, let $\#(S, n)$ denote the number of occurrences of n in S . For S and T finite multisets over \mathbb{N} , define $S \succ T$ if and only if:

There is some $m \in \mathbb{N}$ such that $\#(S, m) > \#(T, m)$ and for all $k > m$ it holds that $\#(S, k) = \#(T, k)$.

In plain English, if one starts with a finite multiset, removes some numbers from this multiset and replaces each of them with any finite number of strictly smaller numbers, then the result is defined to be smaller than the starting point. Using this ordering is equivalent to an induction of type ω^ω .

Lemma 4.1 *The ordering “ \succ ” is well-founded, i.e. there cannot be an infinite descending chain $S \succ S_1 \succ S_2 \succ \dots \succ S_i \succ \dots$.*

Proof: See [DM79]. ■

4.4 A Normalizing \star -Reduction Strategy.

We now prove that a particular \star -reduction strategy terminates for all simply-typable λ -terms, implying all such terms are β -SN. For a λ -term M , the I -redex Δ is *innermost* if it does not properly contain another I -redex (but Δ may contain K -redexes). Every λ -term with one or more I -redexes contains at least one innermost I -redex.

The notion of \star -reduction is defined so far only for untyped λ -terms in γ -normal form (members of Λ^γ). Denote the set of all simply-typed λ -terms M such that $|M| \in \Lambda^\gamma$ by $(\Lambda^\rightarrow)^\gamma$. Since \star -reduction is stated in terms of β -reduction and γ -reduction, it is obvious how to extend it to any simply-typed λ -term M such that $|M| \in \Lambda^\gamma$.

Lemma 4.2 *If $M, N \in (\Lambda^\rightarrow)^\gamma$ and $M \xrightarrow{\Delta/\star} N$ where Δ is an innermost I -redex, then $order^\bullet(M) \succ order^\bullet(N)$.*

Proof: Note that this proof follows ideas first presented in Turing’s weak normalization proof for the simply-typed λ -calculus [Gan80a]. We give here only the barest sketch of the proof. See the full technical report for details [KW94].

The residuals of any pre-existing β -redexes (other than the one being reduced) retain the same $order$ value, so for them we need only show they are not duplicated. The initial β -reduction step (of the \star -reduction step) does not duplicate β -redexes because Δ is innermost. Also, any new β -redexes it creates have a lower $order$ value than $order(\Delta)$. The subsequent γ -reduction steps can create new β -redexes. In every case, the λ -abstraction forming the function of the new β -redex was one of the outermost λ -abstractions of the argument in Δ , and thus the value of $order$ of the new β -redex is lower than $order(\Delta)$. ■

Lemma 4.3 *If $M \in (\Lambda^\neg)^\gamma$ then M is \star -normalizing.*

Proof: Suppose there is no \star -reduction sequence from M which reaches \star -normal form. Then there is an infinite \star -reduction of the form:

$$M \xrightarrow[\star]{\Delta_0} M_1 \xrightarrow[\star]{\Delta_1} M_2 \xrightarrow[\star]{\Delta_2} \dots$$

where each redex $\Delta_0, \Delta_1, \Delta_2, \dots$, is an innermost I -redex. By Lemma 4.2, there must be this infinite sequence:

$$\text{order}^\bullet(M) \succ \text{order}^\bullet(M_1) \succ \text{order}^\bullet(M_2) \succ \dots$$

This contradicts Lemma 4.1, so there must be a \star -reduction sequence from M which reaches \star -normal form. ■

We believe that Lemma 4.3 could be stated more strongly: If $M \in (\Lambda^\neg)^\gamma$ then M is \star -SN. However, we do not need such a strong result to prove the next theorem.

Theorem 4.4 *If $M \in \Lambda^\neg$ then M is β -SN.*

Proof: Let $N \equiv \gamma\text{-nf}(M)$. Since N is in $(\Lambda^\neg)^\gamma$, N is \star -normalizing by Lemma 4.3. Thus, by Theorem 3.11, M is β -SN. ■

5 The \star -Normalization of the Intersection-Type Discipline

In this section, we prove that every λ -term typable in the system of intersection types \star -normalizes and is therefore β -SN. The novelty of this proof is that the argument does not depend on semantic notions such as interpretations, proofs of soundness, etc. As in Section 4, this is a new proof for a well-known result. The format of this section closely follows the format of Section 4.

5.1 The Intersection-Type Discipline.

The intersection-type discipline is defined as an extension of the simply-typed λ -calculus. Cardone and Coppo call the system presented here the system of *simple* intersection types, reserving the unqualified name for the system with the ω type constant that can be assigned to any λ -term [CC90]. Sometimes, a “ \leq ” rule is included with the system for type inclusion, but this is not necessary here since this rule does not change the set of typable λ -terms.

The set of intersection types \mathbb{T}^\wedge is built from the countably infinite set of type variables \mathbb{V} using the “ \rightarrow ” and “ \wedge ” type constructor as specified by the grammar $\mathbb{T}^\wedge ::= \mathbb{V} \mid (\mathbb{T}^\wedge \rightarrow \mathbb{T}^\wedge) \mid (\mathbb{T}^\wedge \wedge \mathbb{T}^\wedge)$. A type

is therefore either a *type variable*, a *\rightarrow -type*, or a *\wedge -type*. The same notation conventions are followed as with the simply-typed λ -calculus, except that “ \wedge ” is left-associative and has higher precedence than “ \rightarrow ” so that $\sigma \wedge \tau \rightarrow \rho = (\sigma \wedge \tau) \rightarrow \rho$.

The standard Curry-style presentation of the intersection-type system makes it hard to define explicitly how β -reduction and γ -reduction work on typings. So instead we give an equivalent Church-style presentation of the intersection-type discipline. (This is done despite Barendregt’s fairly accurate observation that “for ... the system of intersection types ... it is not clear how to define a Church version” [Bar92].)

As with the simply-typed λ -calculus, λ -term variables are pairs of untyped variables and types written as x^σ, y^τ, z^ρ , etc. The λ -terms will be required to satisfy the same property (†) as before, so that type assignments can be avoided. The set Λ^\wedge of λ -terms in the intersection-type system and a type-erasing function $|\cdot|$ from Λ^\wedge to Λ are defined inductively as follows:

1. $x^\sigma \in \Lambda^\wedge$ and $|x^\sigma| = x$ if $x \in \mathcal{V}$ and $\sigma \in \mathbb{T}^\wedge$.
2. $(M^{\sigma \rightarrow \tau} N^\sigma)^\tau \in \Lambda^\wedge$ and $|(M^{\sigma \rightarrow \tau} N^\sigma)^\tau| = (|M^{\sigma \rightarrow \tau}| |N^\sigma|)$ if $M^{\sigma \rightarrow \tau} \in \Lambda^\wedge$ and $N^\sigma \in \Lambda^\wedge$.
3. $(\lambda x^\sigma . M^\tau)^{\sigma \rightarrow \tau} \in \Lambda^\wedge$ and $|(\lambda x^\sigma . M^\tau)^{\sigma \rightarrow \tau}| = (\lambda x . |M^\tau|)$ if $M^\tau \in \Lambda^\wedge$ and $\sigma \in \mathbb{T}^\wedge$.
4. $(\wedge\text{-I } M_1^{\sigma_1} \dots M_n^{\sigma_n})^{\sigma_1 \wedge \dots \wedge \sigma_n} \in \Lambda^\wedge$ and $|(\wedge\text{-I } M_1^{\sigma_1} \dots M_n^{\sigma_n})^{\sigma_1 \wedge \dots \wedge \sigma_n}| = |M_1^{\sigma_1}|$ if $|M_1^{\sigma_1}| \equiv \dots \equiv |M_n^{\sigma_n}|$ and for $1 \leq i \leq n$ it holds that σ_i is not a \wedge -type.
5. $(\wedge\text{-E } M^{\sigma_1 \wedge \dots \wedge \sigma_n})^{\sigma_i} \in \Lambda^\wedge$ where $1 \leq i \leq n$ and $|(\wedge\text{-E } M^{\sigma_1 \wedge \dots \wedge \sigma_n})^{\sigma_i}| = |M^{\sigma_1 \wedge \dots \wedge \sigma_n}|$ if $M^{\sigma_1 \wedge \dots \wedge \sigma_n} \in \Lambda^\wedge$.

This definition may seem quite unusual to some readers, but the clever reader will see that this presentation merely converts entire Curry-style derivations directly into terms. The only difference is that multiple uses of the $\wedge\text{-I}$ and $\wedge\text{-E}$ rules are compressed into one use and that no distinction is made between the type $\sigma \wedge (\tau \wedge \rho)$ and $(\sigma \wedge \tau) \wedge \rho$. Sometimes, we will be lazy and omit some of the type annotations from λ -terms in Λ^\wedge when the types are not relevant.

Now it is necessary to give an interpretation for what it means to perform β -reduction and γ -reduction on members of Λ^\wedge . Before doing that, a way to eliminate redundant pairs of $\wedge\text{-I}$ and $\wedge\text{-E}$ must be provided. Although it can be assumed that the typing of a λ -term does not have redundant uses of $\wedge\text{-I}$ and

The more complicated case is like this, where π stands for the type $(\rho_1 \rightarrow \tau_1) \wedge \dots \wedge (\rho_n \rightarrow \tau_n)$:

$$\begin{array}{c}
((\lambda x^\sigma . (\wedge - \mathcal{I} (\lambda y^{\rho_1} . P_1^{\tau_1})^{\rho_1 \rightarrow \tau_1} \\
\vdots \\
(\lambda y^{\rho_n} . P_n^{\tau_n})^{\rho_n \rightarrow \tau_n})^\pi)^\sigma \rightarrow^\pi Q^\sigma)^\pi \\
\downarrow \\
(\wedge - \mathcal{I} (\lambda y^{\rho_1} . ((\lambda x^\sigma . P_1^{\tau_1})^\sigma \rightarrow^{\tau_1} Q^\sigma)^{\tau_1})^{\rho_1 \rightarrow \tau_1} \\
\vdots \\
(\lambda y^{\rho_n} . ((\lambda x^\sigma . P_n^{\tau_n})^\sigma \rightarrow^{\tau_n} Q^\sigma)^{\tau_n})^{\rho_n \rightarrow \tau_n})^\pi
\end{array}$$

(We omit the mandatory α -conversion necessary to rename separate bound variables distinctly that is required by our convention (\dagger). Assume it happens.) Given this definition of γ 1-reduction, regular γ -reduction is now defined in terms of γ 1-reduction in the exact same way that regular β -reduction is defined in terms of β 1-reduction.

5.2 Extending the Metric to Intersection Types.

The metric of Subsection 4.2 is now extended to handle intersection types. For intersection types, define the function *order* inductively as follows:

1. $order(\alpha) = 0$ where $\alpha \in \mathbb{V}$ is a type variable.
2. $order(\sigma \rightarrow \tau) = \max\{1 + order(\sigma), order(\tau)\}$.
3. $order(\sigma \wedge \tau) = \max\{order(\sigma), order(\tau)\}$.

For a singular β 1-redex $M^\tau \equiv ((\lambda x^\sigma . P^\tau)^\sigma \rightarrow^\tau Q^\sigma)^\tau$, define $order(M^\tau) = order(\sigma)$, exactly as for the simply-typed λ -calculus. If $X = \{M_1^{\sigma_1}, \dots, M_n^{\sigma_n}\}$ is a set of β 1-redexes, then define $order(X) = \max\{order(M_1^{\sigma_1}), \dots, order(M_n^{\sigma_n})\}$. For a typed λ -term M^τ , let X_1, \dots, X_n be all of the I -redexes in M^τ . (Each X_i is a parallel set of β 1-redexes.) Then define $order^\bullet(M^\tau)$ to be the multiset $\{order(X_1), \dots, order(X_n)\}$.

5.3 Normalizing \star -Reduction for Intersection Types.

We now prove that a particular \star -reduction strategy terminates for all λ -terms typable in the intersection-type discipline, implying all such terms are β -SN. The same reduction strategy is used that was presented in Subsection 4.4.

Denote the set of all typed λ -terms $M^\tau \in \Lambda^\wedge$ such that $|M^\tau| \in \Lambda^\gamma$ by $(\Lambda^\wedge)^\gamma$. Extend \star -reduction to terms in Λ^\wedge in the obvious manner.

Lemma 5.1 *If $M, N \in (\Lambda^\wedge)^\gamma$ and $M \xrightarrow{\Delta} N$ where Δ is an innermost I -redex, then $order^\bullet(M) \succ order^\bullet(N)$.*

Proof: The structure of this proof is almost identical to the proof for Lemma 4.2. The main differences are the notation necessary to account for parallel sets and the handling of r -reduction in between β -reduction and γ -reduction steps. We sketch here the differences with Lemma 4.2. See the full technical report for details [KW94].

It is necessary to show that r -reduction does not duplicate or remove β -redexes, although it can remove β 1-redexes. It is somewhat tricky to show that r -reduction does not change the value of *order* for one of the pre-existing β -redexes. The notation for the substructure of the argument of Δ is fairly complicated to handle its parallel structure. There is also extra complexity in analyzing where the types come from that define the value of *order* for the fresh β -redexes. Otherwise, the proof is essentially the same. ■

Theorem 5.2 *If $M^\tau \in \Lambda^\wedge$, then M^τ is β -SN.*

Proof: The proof is a combination of the proofs for Lemma 4.3 and Theorem 4.4 except that it depends on Lemma 5.1 instead of Lemma 4.2. ■

References

- [Bar84] H. P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics*. North-Holland, revised edition, 1984.
- [Bar92] H. P. Barendregt. Lambda calculi with types. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, eds., *Handbook of Logic in Computer Science*, vol. 2, chapter 2, pp. 117–309. Oxford University Press, 1992.
- [CC90] F. Cardone and M. Coppo. Two extensions of Curry’s type inference system. In Odifreddi [Odi90], chapter 1, pp. 19–75.
- [CDC80] M. Coppo and M. Dezani-Ciancaglini. An extension of basic functionality theory for lambda-calculus. *Notre Dame J. Formal Log.*, 21:685–693, 1980.
- [CDCV81] M. Coppo, M. Dezani-Ciancaglini, and B. Venneri. Functional characters of solvable terms. *Z. Math. Log. Grund. Math.*, 27:45–58, 1981.
- [dG93] P. de Groote. The conservation theorem revisited. In *Int’l Conf. Typed Lambda Calculi and Applications*, vol. 664 of *LNCS*, pp. 163–178. Springer-Verlag, Mar. 1993.
- [DM79] N. Dershowitz and Z. Manna. Proving termination with multiset orderings. *J. ACM*, 22:465–476, 1979.
- [dV87] R. de Vrijer. Exactly estimating functionals and strong normalization. *Indagationes Mathematicae*, 49(4):479–493, 1987.

- [Gal90] J. H. Gallier. On Girard's "candidats de reductibilité". In Odifreddi [Odi90], pp. 123–203.
- [Gal94] J. Gallier. Proving properties of typed λ -terms using realizability, covers, and sheaves. Technical report, Dept. of Comp. and Inf. Sci., Univ. of Pennsylvania, Dec. 7, 1994.
- [Gan80a] R. O. Gandy. An early proof of normalization by A. M. Turing. In Seldin and Hindley [SH80], pp. 453–455.
- [Gan80b] R. O. Gandy. Proofs of strong normalization. In Seldin and Hindley [SH80], pp. 457–477.
- [Gir71] J.-Y. Girard. Une extension de l'interprétation de Gödel à l'analyse, et son application à l'élimination des coupures dans l'analyse et la théorie des types. In J. E. Fenstad, ed., *Proceedings of 2nd Scandinavian Logic Symposium*, pp. 63–92, Amsterdam, 1971. North Holland.
- [Gir72] J.-Y. Girard. *Interprétation Fonctionnelle et Élimination des Coupures de l'Arithmétique d'Ordre Supérieur*. Thèse d'Etat, Université Paris 7, 1972.
- [GLT89] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*. Number 7 in Cambridge Tracts in Theor. Comp. Sci. Cambridge Univ. Press, 1989.
- [Klo80] J. W. Klop. *Combinatory Reduction Systems*. Number 127 in Mathematical Centre Tracts. Mathematisch Centrum, Amsterdam, 1980. Ph.D. Thesis.
- [KW94] A. J. Kfoury and J. B. Wells. New notions of reduction and non-semantic proofs of β -strong normalization in typed λ -calculi. Tech. Rep. 94-014, Comput. Sci. Dept., Boston Univ., 1994.
- [KW95] A. J. Kfoury and J. B. Wells. Addendum to "New notions of reduction and non-semantic proofs of β -strong normalization in typed λ -calculi". Tech. Rep. 95-007, Comput. Sci. Dept., Boston Univ., 1995.
- [Lei86] D. Leivant. Typing and computational properties of lambda expressions. *Theoretical Comput. Sci.*, 44:51–68, 1986.
- [Men91] N. P. Mendler. Inductive types and type constraints in the second-order lambda calculus. *Annals of Pure and Applied Logic*, 51:159–172, 1991.
- [Ned73] R. P. Nederpelt. *Strong Normalization for a Typed Lambda Calculus with Lambda Structured Types*. PhD thesis, Technische Hogeschool Eindhoven, 1973.
- [Odi90] P. Odifreddi, ed. *Logic and Computer Science*. Number 31 in the APIC Series. Academic Press, 1990.
- [Pot80] G. Pottinger. A type assignment for the strongly normalizable λ -terms. In Seldin and Hindley [SH80], pp. 561–577.
- [Sce87] A. Scedrov. Normalization revisited. In J. W. Gray and A. Scedrov, eds., *Proc. AMS Research Conference*, pp. 357–369. American Mathematical Society, 1987.
- [SH80] J. P. Seldin and J. R. Hindley, eds. *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism*. Academic Press, 1980.
- [Tai67] W. W. Tait. Intensional interpretation of functionals of finite type I. *J. Symbolic Logic*, 32:198–212, 1967.
- [vB92] S. J. van Bakel. Complete restrictions of the intersection type discipline. *Theoretical Comput. Sci.*, 102(1):135–163, 1992.
- [vdPS95] J. van de Pol and H. Schwichtenberg. Strict functionals for termination proofs. In *2nd Int'l Conf. Typed Lambda Calculi and Applications*. Springer-Verlag, Apr. 1995.