NORMALIZATION

IN λ-CALCULUS AND TYPE THEORY

DISSERTATION

M.H.B. Sørensen

Department of Computer Science
University of Copenhagen

Supervisors:

H.P. Barendregt, *Catholic University of Nijmegen.*

N.D. Jones, *University of Copenhagen.*

External reviewers:

T. Coquand, *Chalmers University of Technology, Gothenburg.*

O. Danvy, *University of Aarhus.*

J.W. Klop, *CWI, Amsterdam.*

... it can apparently happen that someone, without much exact learning
and with little of the information collected by earlier generations in his head,
that such an individual, passing his days like other artists
in the creation of more or less fantastic pictures,
can one day feel ripen in himself a conscious wish to use his imaginary images
to approach infinity as purely and as closely as possible.
Deep, deep infinity! Quietness. To dream away from the tensions of daily living;
to sail over a calm sea at the prow of a ship, toward a horizon that always recedes;
to stare at the passing waves and listen to their monotonous soft murmur;
to dream away into unconsciousness ...

M.C. Escher.

Preface

The present monograph constitutes the author's Ph.D. thesis submitted to the Department of Computer Science, University of Copenhagen on April 1st, 1997. The research reported in the thesis was conducted during several periods between September 1994 and March 1997. The topic of the thesis is λ-*calculus* and *type theory;* more precisely, the thesis addresses certain questions regarding *normalization* in these theories.

The thesis contains no tutorial on λ-calculus or type theory, except for a very brief introduction presenting enough concepts and terminology to explain the motivation and achievements of the thesis to someone who does not know the subject. In the rest of the thesis, the reader is expected to have an elementary knowledge of λ-calculus corresponding, e.g., to chapters 2–3 of Barendregt's book [3]. Chapters 6–9, 11, and 13 of the same book would also be useful, but are not essential. The reader is also expected to be familiar with type theory as presented, e.g., in the handbook chapter [4].

Nevertheless, the thesis is self-contained in that it does not rely on any notions from papers or textbooks. The few exceptions are mentioned in the text; for instance, the definition of substitution is assumed to be known.

The bulk of the thesis is made up of three chapters, which have been so written that they can be read in any desired order; each chapter begins with a presentation of the fundamental notions used in that particular chapter (this leads to a bit of duplication). However, the deepest appreciation of ideas in later chapters is obtained by reading earlier chapters first.

To avoid proliferation of the same idea in several contexts, the thesis is concerned with untyped and typed versions of λ-calculus only; thus, no attempt is made to modify results so as to hold for combinatory logic or for various notions of higher-order rewriting systems. Nevertheless, references to relevant similar results in the literature for such systems are given.

Acknowledgments

I owe a huge debt to Henk Barendregt for posing interesting problems and for comments to my ideas. This thesis is a product of Henk's knowledge of the field; I am proud to be able to call myself his student. For his hospitality during my stay in Nijmegen I am more than grateful. It is also a pleasure

to acknowledge my debt to Neil Jones for educating me as a researcher, for helping me out in ways too numerous to be mentioned here, and for supplying ideas for a number of interesting research projects reported in several papers. That I ever became a researcher is due to Neil.

I am grateful to Thierry Coquand, Olivier Danvy, and Jan Willem Klop for agreeing to become members of the thesis committee.

I am thankful to my co-workers on various research projects. In particular, thanks to John Hatcliff and Gilles Barthe whose knowledge of CPS translations and pure type systems, respectively, had a significant influence on my ideas. I would not have come this far without John and Gilles.

Thanks also to Herman Geuvers, Femke van Raamsdonk, Paula Severi, Hongwei Xi, Ralph Loader, Zurab Khasidashvili, Amir Ben-Amram, Torben Mogensen, and Laurent Regnier for discussions and comments.

Thanks to Wei-Ngan Chin, Chet Murthy, and Peter Sestoft for being excellent committee members on previous projects. Thanks also to Fritz Henglein for being a very inspiring supervisor on an early project. Oh, and thanks also to Carsten Gomard, who was the first to see that it might be relevant for me to do research.

Many thanks to Zurab Khasidashvili, Masami Hagiya, Sacchio Hirokawa, Andrei Klimov, Sergei Romanenko, Sergei Abramov, Andrei Nemytykh, Valentin Turchin, Gilles Barthe, Femke van Raamsdonk, and Paula Severi for hospitality during visits in Tbilisi, Tokyo, Kyushu, Moscow, Pereslavl, New York, and Amsterdam.

Thanks to the λ-group in Nijmegen and the TOPPS group at DIKU for providing inspiring working environments.

Thanks to the Computer Department at DIKU for good support, to Erik Barendsen and Henning Niss for help with LaTeX, and to Kristoffer Rose for help with XY-pic.

Thanks to my friends in the Dead Computer Scientists' Society for all those happy evenings with discussions about Church, Gödel, and VIC 20. Thanks also to Peter Harry Eidorff with whom I have spent great undergraduate days, and to my brother in spirit, Jakob Rehof, who shares my interest for λ-calculus, type theory, and Clint Eastwood.

To my parents and to my girl-friend, Mette Bjørnlund, I shall remain forever indebted.

M.H.S., April 1997

Preface to the revised edition

In this revised version, appearing as a DIKU report, Chapter 1 has been updated according to suggestions from referees of a paper based on the previous version of the chapter. Thanks to Zurab Khasidashvili, Vincent van Oostrom, and Roel de Vrijer whose comments significantly improved that paper. Corrections and additions suggested by the thesis committee have also been incorporated throughout the thesis, and an index has been added.

Since the appearance of the first edition of this thesis, an interesting translation has been developed by Xi [144], which is similar to the continuation passing style translation in Chapter 2. This new translation is mentioned in passing in Chapter 1, but has not been worked into Chapters 2 and 3. In fact, the new translation can be viewed as a so-called *thunkification* translation—see [47]—and like the continuation passing style translation, the thunkification translation can be viewed as a permutative inner interpretation. Since the thunkification translation is simpler than the continuation passing style translation, it would be interesting to see whether it can be used to prove the Barendregt-Geuvers-Klop conjecture for a larger class of pure type systems than that studied in Chapter 3.

Also, several other parts of the thesis call for elaboration. The strong normalization proofs by Gandy, de Vrijer, etc. mentioned in Chapter 1, which establish upper bounds for length of reduction sequences, seem to yield reductions of strong normalization to weak normalization of the same notion of reduction, and this should be investigated in greater detail. The reduction of strong normalization to weak normalization by Loader mentioned in Chapter 2 should also be examined more closely. These techniques may provide alternative approaches to attack the Barendregt-Geuvers-Klop conjecture. Another idea for attacking the conjecture is to generalize the translation due to Harper, Honsell, and Plotkin which eliminates dependent types.

The relation between permutative inner interpretations and monads in Chapter 2 should be explained in greater detail. Also, the technique in Chapter 2 should be applied to a greater variety of systems, e.g., systems with the permutative conversions known from proof normalization and Gödel's system T, if possible. It would also be interesting to study type systems which are weakly but not strongly normalizing (at present no such systems are known among the pure type systems, but one could choose among other systems).

Finally, the connection in Chapter 3 to the K-conjecture and to the looping combinators of Coquand and Herbelin should be elaborated.

These issues will be addressed elsewhere.

M.H.B.S., April 1998

Contents

Introduction

λ-*calculus* is a collection of formal theories of interest in, e.g., computer science and logic. The objects of study in these theories are λ-*terms,* which express functions and applications of functions in a pure form. For instance,

$$\lambda x.x$$

is a λ-term which, intuitively, denotes the function that maps any argument to itself, i.e., the identity function. This is similar to the notation $n \mapsto n$ employed in mathematics. However, $\lambda x.x$ is a *string* over an alphabet with symbols λ, x, etc., whereas $n \mapsto n$ is a *function.*

The λ-term $\lambda x.x$ is henceforth called $\mathbf{I}$, in short:

$$\mathbf{I} \equiv \lambda x.x.$$

As in the notation $n \mapsto n$, the name of the *bound variable* x in $\lambda x.x$ is not significant; thus, we might as well have written

$$\mathbf{I} \equiv \lambda y.y.$$

Another λ-term is

$$\mathbf{K}^* \equiv \lambda y.\lambda x.x$$

which, intuitively, denotes the function that maps any argument to a *function,* namely the one that maps any argument to itself, i.e., the identity function. This is similar to programming languages where a procedure may return a procedure as a result. A related λ-term is

$$\mathbf{K} \equiv \lambda y.\lambda x.y$$

which, intuitively, denotes the function that maps any argument to the function that, for any argument, returns the former argument. λ-terms of the form $\lambda x.P$ are generally called *abstractions.*

Since λ-terms intuitively denote functions, there is a way to express *application* of one λ-term to another; this is expressed by juxtaposition. Thus, the λ-term

$$\mathbf{I}\,\mathbf{K}$$

expresses application of $\mathbf{I}$ to $\mathbf{K}$. Since $\mathbf{K}$ intuitively denotes a function too, $\mathbf{I}$ denotes a function which may have another function as argument. This is similar to programming languages where a procedure may receive another procedure as argument.

In mathematics we usually write application of a function, say, $f(n) = n^2$ to an argument, say, 4 with the argument in parenthesis: $f(4)$. By tradition this is not done in λ-calculus. However, we do need to put some parentheses to delimit the scope of applications and abstractions. For instance,

$$(\lambda x.x)\ \mathbf{I} \qquad \lambda x.(x\ \mathbf{I})$$

are not the same λ-term; the first is $\mathbf{I}$ applied to $\mathbf{I}$, whereas the second expects an argument x which is applied to $\mathbf{I}$. To save parentheses, it is customary to omit the parentheses in the second of the two λ-terms.

Intuitively, if $\lambda x.M$ denotes a function, and N denotes an argument, then the the result of the function on the argument is denoted by the λ-term that arises by *substitution* of N for x in M. This latter λ-term is written[1]

$$M\{x := N\}.$$

This is similar to common practice in mathematics; if f is as above, then $f(4) = 4^2$, and we get from the application $f(4)$ to the result 4^2 by substituting 4 for n in the body of the definition of f.

The process of calculating results is formalized by *β-reduction.* One writes $M \to_\beta N$ if N arises from M by replacing a *β-redex,* i.e., a part of form

$$(\lambda x.P)\ Q$$

by its *β-contractum*

$$P\{x := Q\}.$$

For instance,

$$\mathbf{I}\,\mathbf{K} \equiv (\lambda x.x)\ \mathbf{K} \to_\beta x\{x := \mathbf{K}\} \equiv \mathbf{K}.$$

If $M \to_\beta \ldots \to_\beta N$ in zero or more steps, one writes $M \twoheadrightarrow_\beta N$.

Since a λ-term M may contain several β-redexes, i.e., several parts of form $(\lambda x.P)\ Q$, there may be several N such that $M \to_\beta N$. For instance,

$$\mathbf{K}\ (\mathbf{I}\,\mathbf{I}) \to_\beta \lambda x.(\mathbf{I}\,\mathbf{I})$$

and also

$$\mathbf{K}\ (\mathbf{I}\,\mathbf{I}) \to_\beta \mathbf{K}\,\mathbf{I}.$$

However, the celebrated *Church-Rosser theorem* states that if

$$M \twoheadrightarrow_\beta M_1$$

[1] Some care must be taken in the substitution operation to avoid confusion between different variables; such problems are beyond the scope of this introduction.

and

$$M \twoheadrightarrow_\beta M_2,$$

then a single λ-term M_3 can be found with

$$M_1 \twoheadrightarrow_\beta M_3$$

and

$$M_2 \twoheadrightarrow_\beta M_3.$$

In particular, if M_1 and M_2 are *β-normal forms,* i.e., λ-terms that admit no further β-reductions, then they must be the same λ-term, since the β-reductions from M_1 and M_2 to M_3 must be in zero steps. This is similar to the fact that when we calculate the value of an arithmetical expression, e.g.,

$$(4+2)\cdot(3+7)\cdot 11,$$

the end result is independent of the order in which we do the calculations.

The idea that any λ-term M denotes a function also gives rise to another type of reduction, namely *η-reduction,* which states that $M \to_\eta N$, if N arises from M by replacing a part of form $\lambda x.P\,x$ by P, where x does not appear in P; a λ-term of the former kind is an *η-redex.* For instance,

$$\lambda y.\mathbf{I}\,y \to_\eta \mathbf{I}.$$

Usually one considers either β-reduction alone or β-reduction together with η-reduction. To stress the distinction, one speaks of *$\lambda\beta$-calculus* and *$\lambda\beta\eta$-calculus.* In the rest of this introduction we are concerned with β-reduction alone, and adopt the usual convention of omitting "β-" from the notions *β-redex, β-reduction,* etc.

λ-calculus is a *type-free* formalism. Unlike common mathematical practice, we do not insist that λ-terms denote functions from certain domains, e.g., the natural numbers, and that arguments be drawn from these domains. In particular, we may have self-application as in the λ-term

$$\omega \equiv \lambda x.x\,x,$$

and we may apply *this* λ-term to *itself* as in the λ-term

$$\Omega \equiv \omega\,\omega.$$

The type-free nature of λ-calculus leads to some interesting phenomena; for instance, a λ-term may reduce to itself as in

$$\Omega \equiv (\lambda x.x\,x)\,\omega \to_\beta \omega\,\omega \equiv \Omega.$$

Therefore, there are also λ-terms with infinite reduction sequences, like

$$\Omega \to_\beta \Omega \to_\beta \dots .$$

Some λ-terms admit both an infinite reduction sequence:

$$\mathbf{K}^* \, \Omega \ \to_\beta \mathbf{K}^* \, \Omega \ \to_\beta \ldots$$

(where the reductions are in Ω) and a finite one to normal form:

$$\mathbf{K}^* \, \Omega \ \to_\beta \mathbf{I}.$$

A λ-term is *weakly normalizing* if it admits a reduction sequence ending in a normal form. A λ-term is *strongly normalizing* if all its reduction sequences eventually end in normal forms. The latter trivially implies the former, but not vice versa as the above example illustrates.

However, the *normalization theorem,* due to Curry and Feys, states that repeatedly reducing the left-most redex in a weakly normalizing λ-term eventually leads to a normal form—even if the λ-term is not strongly normalizing. Another way to state this is that the *reduction strategy* which always reduces the left-most redex is *normalizing.* There is also a reduction strategy with the property that, for any λ-term admitting an infinite reduction sequence, reducing according to the strategy does *not* lead to a normal-form. Such a strategy is called *perpetual.* A normalizing reduction strategy which, for any weakly normalizing λ-term, computes a shortest reduction sequence among all those leading to a normal form is called *minimal.* Similarly, a perpetual reduction strategy which, for any strongly normalizing λ-term, computes a longest reduction sequence is called *maximal.*

Interestingly, the existence of λ-terms that admit both a reduction sequence to normal form as well as an infinite reduction sequence vanishes if we allow the formation of abstractions $\lambda x.P$ only when x occurs in P. This result is known as the *conservation theorem for λI,* due to Church and Rosser. The terminology stems from the fact that the resulting fragment of λ-calculus is called *λI-calculus,* and to make the distinction explicit, general λ-calculus is sometimes called *λK-calculus.*

A reduction sequence from a λ-term M which only reduces redexes that are present in M, in a certain sense, is called a *development.* For instance

$$(\lambda x.x) \; ((\lambda y.y) \; z) \to_\beta (\lambda y.y) \; z \to_\beta z$$

is a development, but

$$(\lambda x.x \; z) \; (\lambda y.y) \to_\beta (\lambda y.y) \; z \to_\beta z$$

is not, because the redex contracted in the last step is not present in the original term; in contrast, it is created during the reduction path. Curry and Feys' *finite developments theorem* states that there are no infinite developments.

In some variants of λ-calculus, collectively known as *type theory,* restrictions are made regarding the λ-terms that may be applied to other λ-terms. One considers *types*, e.g.,

$$\mathbf{0} \to (\mathbf{0} \to \mathbf{0}).$$

Intuitively, $\mathbf{0}$ denotes some set, and $A \to B$ denotes the set of functions from A to B. However, like λ-terms, types are *strings.*

One then stipulates that each variable x has some type A. Moreover, if x has type A and M has type B, then $\lambda x.M$ has type $A \to B$. This reflects the intuition that if M denotes an element of B for each x in A, then $\lambda x.M$ denotes a function from A to B. In a similar vein, if M has type $A \to B$ and N has type A, then $M\,N$ has type B.

A type theory is *weakly* and *strongly normalizing* if all terms that have a type are weakly and strongly normalizing, respectively. Again, the latter trivially implies the former. One can prove, using a classical technique due to Tait and Girard, that many type theories satisfy both properties.

λ-calculus and the related systems of *combinatory logic* were originally proposed as a foundation of mathematics around 1930 by Church and Curry, but the proposed systems were subsequently shown to be inconsistent by Kleene and Rosser in 1935. However, the subsystem described above consisting of the λ-terms equipped with β-reduction turned out to be useful for formalizing the intuitive notion of effective computability, and this was a main motivation for the development of *recursion theory.* [2]

With the invention of physical computers came also programming languages, and λ-calculus and combinatory logic have proved to be useful tools in the theory and implementation of programming languages. For instance, λ-calculus may be considered an idealized sublanguage of higher-order programming languages like Lisp. In this case, β-reduction expresses an elementary computation step, and, roughly, normal forms are the results of computations. Also, λ-calculus is useful for expressing semantics of programming languages as done in *denotational semantics.* Hindley and Seldin [50, p43] summarize the situation: "λ-calculus and combinatory logic are regarded as 'test-beds' in the study of higher-order programming languages: techniques are tried out on these two simple languages, developed, and then applied to other more 'practical' languages."

Similarly, type theory is useful for the study and implementation of programming languages with types like Pascal and ML.[3] Type theory is also of interest in logic due to the so-called *Curry-Howard Isomorphism,* which interprets types as formulae in formal logic and λ-terms as representations of formal proofs. In this case, β-reduction expresses reductions of proofs, studied independently in *proof theory.*

In these applications of λ-calculus, the property of weak normalization is of considerable importance. For instance, in a programming language, weak normalization of a term guarantees that the term has a result; in

[2] 'For more on the history of λ-calculus, see, e.g., [3] or [50]. First hand information may be obtained from Kleene and Rosser's eye witness statements [75, 110], and from Curry and Feys' book [29], which contains a wealth of historical information.

[3] For instance, type theories are used as intermediate languages for the compilation of realistic higher-order typed programming languages.

general programming languages not all terms have this property, and it is not decidable, in general, whether a term has the property—this is a variant of the *halting problem.* In proof theory, weak normalization of type theories is used to prove logical *consistency* of formal logics.

In some applications it is necessary, or at least more convenient, to know that a λ-term or a type theory is not merely weakly normalizing, but in fact strongly normalizing. For instance, weak normalization of a λ-term implies that the λ-term has a normal form, but to find this normal form one needs a reduction strategy that is guaranteed to find the normal form, e.g., left-most reduction. Knowing that all reduction sequences eventually terminate allows us choose between different reduction strategies according to, e.g., efficiency concerns. As another example, some proof techniques, e.g., *Newman's Lemma*, require strong normalization of λ-terms. Finally, as van de Pol [102, p3] puts it: "After all, it is quite natural to ask whether *all* reduction sequences eventually lead to a normal form."

As mentioned above, strong normalization of a λ-term or of a type theory trivially implies weak normalization of the λ-term or the type theory, so the benefits of weak normalization are inherited by strong normalization. However, for some λ-terms and type theories, weak normalization is easier to prove than strong normalization. This raises the idea of studying techniques to infer strong normalization from weak normalization in λ-calculus and type theory. This thesis is concerned with such techniques.

Overview

We end the introduction with a brief overview of the remaining three chapters, emphasizing the research contribution of each chapter.

The first chapter surveys a part of the theory of β-reduction in λ-calculus which might aptly be called *perpetual reductions.* The theory is concerned with *perpetual reduction strategies,* i.e., reduction strategies that compute infinite reduction paths from λ-terms (when possible), and with *perpetual redexes,* i.e., redexes whose contraction in λ-terms preserves the possibility (when present) of infinite reduction paths. The survey not only recasts classical theorems in a unified setting, but also offers new results and proof techniques, as well as a number of applications to problems in λ-calculus and type theory. In particular, the theory provides techniques to infer strong normalization from weak normalization which are used in the following two chapters to address the connection between weak and strong normalization in type theories.

The chapter begins with a classification of redexes and reduction strategies and proves equivalence between some classifications from the literature.

Next a lemma is proved which we call the *fundamental lemma of perpetuality.* The lemma is used—often implicitly—in many proofs in the literature, e.g., in the Tait & Girard strong normalization proofs. An attempt is then made to show that the core of the recent techniques by van Raamsdonk and Severi [106] and by Xi [140] for proving strong normalization results is captured by this lemma.

Several known perpetual reduction strategies are then presented; the proofs or perpetuality are an immediate consequence of the fundamental lemma of perpetuality.

A stronger form of the fundamental lemma of perpetuality, which we call the *fundamental lemma of maximality,* is then presented. This result is often used implicitly in strong normalization proofs which establish upper bounds for the length of reduction paths. We use the lemma to show maximality of a certain effective reduction strategy.

A new result is proved (Proposition 1.3.27) which states that to compute an upper bound for the length of reduction sequences from some λ-term, one cannot do better, in a certain sense, than simply try to reduce the λ-term to normal form, using this maximal reduction strategy, and count the number of reductions along the way. This shows that there is no analog in general λ-calculus of the techniques known for type theories and developments for computing upper bounds for length of reductions.

A new result, which we call the *Ω-theorem,* is then proved, stating that every λ-term in every infinite reduction sequence contains Ω as a substring. This result gives a technique to infer strong normalization from weak normalization of λ-terms, and throws some light on a certain false conjecture. It also implies some results that previously relied on tedious case analyses.

We then study approaches to proving strong normalization of *simply typed λ-calculus* based on the fundamental lemma of perpetuality and based on the related techniques by van Raamsdonk and Severi and by Xi. In particular, a new proof is presented.

We also study approaches to proving *finiteness of developments,* based on a version of the fundamental lemma of perpetuality for developments, and in particular give a new, perspicuous proof of this theorem.

Next a well-known proof technique is refined and used to give smooth proofs of the conservation theorem for λI, of the so-called conservation theorem for λK, and of a related theorem due to Bergstra and Klop; these theorems amount to characterizations of perpetual redexes, and also give a method to infer strong normalization from weak normalization. The technique is also demonstrated to yield the normalization theorem with little effort. We also show that the normalization theorem implies the conservation theorem for λI.

We conclude the first chapter with a new technique to compute shortest developments. This result does not belong to the main path of the chapter, but arises by an interesting principle of duality from a technique to compute longest developments due to de Vrijer [135].

For some type theories it is easier to prove weak normalization than strong normalization. More precisely, although it is equally difficult to prove weak and strong normalization using the Tait & Girard method, there is, for some type theories, a method to prove weak normalization which is substantially simpler than the Tait & Girard method.

A number of techniques to strong normalization from weak normalization have been invented over the last twenty years by Nederpelt [92], Klop [76], Khasidashvili [69], Karr [62], de Groote [31], and Kfoury and Wells [66], but these techniques 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 which involves very different techniques for different calculi. A technique to uniformly infer strong normalization for one notion of reduction from weak normalization of *the same* notion of reduction would be better.

The second chapter presents a new technique to infer strong normalization of a notion of reduction in a type theory from weak normalization of the *same* notion of reduction. The technique not only simplifies the task of proving strong normalization as compared to previous approaches, but also suggests an approach to an open problem in type theory (see below).

The chapter begins with an account of Klop's technique, which is based on an interpretation of λK in λI and the conservation theorem for λI. Klop's technique is then compared to related techniques.

Our technique is then presented as an extension of Klop's technique using a *continuation passing style* translation. The technique is used to infer strong normalization from weak normalization in *simply, second-order,* and *higher-order typed λ-calculus*, a system with *subtypes,* and the system of *positive, recursive types.*

Loader [85] independently uses a somewhat different translation to infer strong normalization from weak normalization in simply and second-order typed λ-calculus. Xi [141] independently uses a translation similar to ours to infer strong normalization from weak normalization in the same two calculi.

The chapter also shows that the continuation passing style translation used in our technique is a special case of a class of translations, which we call *permutative inner interpretations,* each of which gives rise to a similar extension of Klop's technique. The translation studied by Loader may be viewed as another special case.

The *Barendregt-Geuvers-Klop* conjecture states that every weakly normalizing *pure type system* is also strongly normalizing—pure type systems are a general formalism of which specific type theories can be viewed as special cases. In the third chapter, we show that the conjecture is true for an infinite class of pure type systems that includes, e.g., the left hand side of Barendregt's λ-*cube* as well as the well-known system λU. This seems to be the first result giving a positive answer to the conjecture not merely for some concrete systems for which strong normalization is known to hold—for such systems the conjecture is trivially true—but for a uniform *class* of systems in which not all systems are strongly normalizing.

The chapter introduces the notion of a *generalized non-dependent pure type system,* in which types do not depend on terms. This property allows us to give separate continuation passing style translations on terms and types, and these are used to extend the technique of the previous chapter to the class of generalized non-dependent pure type systems.

This class is a generalization of Coquand and Herbelin's [27] *logical non-dependent* pure type system, and our continuation passing style translations generalize Coquand and Herbelin's translations of logical non-dependent pure type systems.

The first chapter (except the last section) is based on [118, 119, 120], and is also inspired by the papers [106, 140], as is elaborated in the chapter. A paper based on the chapter (excluding the last section) has been accepted for publication [107] as joint work with F. van Raamsdonk, P. Severi, and H. Xi. Another paper based on the last section has been submitted for publication [122].

The second chapter is based on [121].

A paper based on the third chapter has been submitted for publication [11] as joint work with G. Barthe and J. Hatcliff.

Not all work of the author relevant for the third chapter has been included. Some difficulties with the system of *higher-order* λ*-calculus* in [121] lead to the study of so-called *domain-free pure type systems* [14, 15] joint with G. Barthe, and to the study of continuation passing style translations into such systems [10, 12] joint with G. Barthe and J. Hatcliff. These techniques were subsequently used to study a framework for λ-calculi corresponding to classical logics via the Curry-Howard isomorphism [9]. Also, the technique for defining general CPS translations was generalized to a general induction principle in joint work with G. Barthe and J. Hatcliff [13]. The chapter uses ideas developed in these projects, but does not use directly the techniques developed in the papers.

CHAPTER 1

Perpetual Reductions in λ-Calculus

This chapter surveys a part of the theory of β-reduction in λ-calculus which might aptly be called *perpetual reductions.* The theory is concerned with *perpetual reduction strategies,* i.e., reduction strategies that compute infinite reduction paths from λ-terms (when possible), and with *perpetual redexes,* i.e., redexes whose contraction in λ-terms preserves the possibility (when present) of infinite reduction paths. The survey not only recasts classical theorems in a unified setting, but also offers new results and proof techniques, as well as a number of applications to problems in λ-calculus and type theory. In particular, the theory provides techniques to infer strong normalization from weak normalization which are used in the next two chapters.

1.1. Introduction

Considerable attention has been devoted to classification of *reduction strategies* in λ-calculus [5, 17, 18, 29, 76, 84, 138]—see also [3, Ch. 13]. We are concerned with strategies differing in the *length* of reduction paths.

(i) A *maximal* strategy computes for a term a *longest* reduction path to normal form, if one exists, otherwise some infinite reduction path.

(ii) A *minimal* strategy computes for a term a *shortest* reduction path to normal form, if one exists, otherwise some infinite reduction path.

(iii) A *perpetual* strategy computes for a term an *infinite* reduction path, if one exists, otherwise some finite reduction path to normal form.

(iv) A *normalizing* strategy computes for a term a *finite* reduction path to normal form, if one exists, otherwise some infinite reduction path.[1]

[1] In this presentation, attention is restricted to the usual λ-calculus. In the so-called *infinite λ-calculus* one also studies infinite reductions ending in infinite normal forms.

Perpetual and normalizing strategies are opposite, in some sense, as are maximal and minimal strategies.

Another classification is concerned with *redexes* rather than strategies. For instance, a redex Δ with contractum Δ' is *perpetual* if, for any context C such that $C[\Delta]$ has an infinite reduction path, $C[\Delta']$ also has an infinite reduction path. This chapter presents a theory of perpetual and maximal β-reduction strategies and β-redexes. The chapter not only recasts in a unified setting classical theorems due to Barendregt, Bergstra, Klop, and Volken, to Church and Rosser, to Curry and Feys, and to de Vrijer, but also presents new results, proofs, and techniques, as well as a number of applications to problems in λ-calculus and type theory demonstrating the elegance and relevance of the theory.

The chapter is organized as follows. Section 1.2 classifies reduction strategies and redexes in λ-calculus and proves equivalence between different formulations of perpetual and maximal strategies and redexes.

Section 1.3 is about perpetual and maximal β-reduction strategies. This is a central theme in work of de Vrijer [135, 136, 138], who uses the technique of counting steps to establish several strong normalization results. The counting functions in fact define reduction strategies.

We first prove a result which we call the *fundamental lemma of perpetuality.* The lemma is used—often implicitly—in many strong normalization proofs in the literature. An attempt is then made to show that the core of the recent techniques by van Raamsdonk and Severi and by Xi for proving strong normalization results is captured by this lemma. The section presents several perpetual reduction strategies; perpetuality is in each case an immediate consequence of the fundamental lemma of perpetuality.

The section then proves a stronger form of the fundamental lemma of perpetuality which we call the *fundamental lemma of maximality.* This result is often used implicitly in strong normalization proofs which establish upper bounds for the length of reduction paths. We use the lemma to show maximality of a certain reduction strategy and to give a certain, trivial technique for computing upper bounds for the length of reduction paths from λ-terms without infinite reduction paths. We also prove that, in a certain sense, the trivial technique cannot be improved.

Sections 1.4–1.6 give applications of perpetual and maximal β-reduction strategies. Section 1.4 presents the recent Ω-*theorem,* stating that every λ-term in every infinite reduction path contains the λ-term Ω as a substring. The proof uses a certain perpetual reduction strategy. Section 1.5 studies approaches to proving strong normalization of *simply typed λ-calculus* based on the fundamental lemma of perpetuality and based on the related techniques by van Raamsdonk and Severi and by Xi. In particular, a new perspicuous proof is presented. Section 1.6 similarly studies approaches to proving *finiteness of developments* and in particular gives a new, perspicuous proof of this theorem.

Section 1.7 is about perpetual β-redexes (as we shall see, maximal β-redexes turn out to be trivial). A well-known proof technique is refined and used to give smooth proofs of the conservation theorem for Λ_I, of the conservation theorem for Λ_K, and of a related theorem due to Bergstra and Klop; these results together give characterizations of perpetual redexes in Λ_I and Λ_K. The technique is also demonstrated to yield the normalization theorem with little effort. The section ends with a very short proof of the conservation theorem for Λ_I using the normalization theorem.

We conclude the chapter with a new technique to compute shortest developments. This result does not belong to the main path of the chapter, but arises by an interesting principle of duality from a technique to compute longest developments due to de Vrijer [135].

Klop [77] surveys some results about reduction strategies in first-order term rewriting systems. Due to the absence of abstractions and the presence of patterns in the term language, some parts of that theory are rather different from what is presented in this chapter; therefore, we shall not consider such systems any further. Several notions of higher-order term rewriting system exist, some of which contain as special cases λ-calculus with β-reduction. We will not consider such systems, although we do try to give references to results that generalize those for λ-calculus presented in this chapter.

1.2. Classification of strategies and redexes

In this section we classify strategies and redexes as outlined in the introduction. The first subsection reviews preliminary notions. The second subsection introduces some notation and properties pertaining to reductions. The third and fourth subsections then classify strategies and redexes and prove equivalence between different classifications from the literature.

1.2.1. Preliminaries

Most notation, terminology, and conventions are adopted from [3]; in this subsection we merely fix the notation for some well-known concepts.

Λ_K is the set of type-free λ-terms. Some example terms are $\mathbf{K} \equiv \lambda x.\lambda y.x$, $\mathbf{I} \equiv \lambda x.x$, $\omega \equiv \lambda x.x\,x$, and $\Omega \equiv \omega\,\omega$. We use $x, y, z, \ldots$ to range over the set V of variables. Familiarity is assumed with conventions for omitting parentheses in λ-terms. Familiarity is also assumed with the notions of free and bound variables, the variable convention, substitution, and the subterm relation, which is denoted by $\subseteq$. Syntactic equality up to renaming of bound variables is denoted by $\equiv$. $\mathrm{FV}(M)$ denotes the set of free variables in M. $\|M\|_x$ denotes the number of free occurrences of x in M. $\|M\|$ denotes the size of M, i.e., the number of occurrences of abstractions, applications, and variables in M. Λ_I is the set of all λ-terms where for every subterm $\lambda x.M$, $x \in \mathrm{FV}(M)$. Thus, $\mathbf{I}, \omega, \Omega \in \Lambda_I$, whereas $\mathbf{K} \notin \Lambda_I$. A λ-context C is a term with a single occurrence of the symbol []; the result of replacing [] by the term M in C is denoted by $C[M]$. Occasionally the name of bound variables matters, e.g., when dealing with contexts. In such cases, $\mathrm{BV}(M)$ denotes the set of variables bound in M.

We occasionally use vector notation $\vec{P}$ for a sequence of terms $P_1 \ldots P_n$ (where $n \geq 0$), e.g., $Q\,\vec{P}$ for $Q\,P_1 \ldots P_n$, and $\vec{P} \in S$ for $P_1, \ldots, P_n \in S$.

A notion of reduction on a set S is a binary relation $R \subseteq S \times S$. If $M\ R\ N$, then M is an R-redex and N its R-contractum. By $R_1 R_2$ we denote the union of two notions of reduction R_1 and R_2. For a notion of reduction R, the corresponding reduction relation $\to_R$ is the compatible closure (relative to some set of contexts). For a reduction relation $\to_R$, $\twoheadrightarrow_R$ is the reflexive, transitive closure, $\twoheadrightarrow_R^+$ is the transitive closure, and $=_R$ is the transitive, reflexive, symmetric closure. We assume the reader is familiar with the notion of reduction β on Λ_K. Several elementary properties about substitution and β-reduction will be used implicitly.

Let $\mathbb{N}^* = \mathbb{N} \cup \{\infty\}$. The following calculation rules are convenient: $\min \emptyset = \max \emptyset = \infty$. Moreover, $\max U = \infty$ if $U \subseteq \mathbb{N}$ is *unbounded*, i.e., if, for all $m \in U$, there is an $n \in U$ with $n > m$. Also, $\infty - k = \infty + k = \infty + \infty = k \cdot \infty = \infty$, for any $k \in \mathbb{N}$. Finally, for $m^*, n^* \in \mathbb{N}^*$ we write $m^* < n^*$ iff either $m^* \neq \infty$ and $n^* = \infty$, or $m^*, n^* \in \mathbb{N}$ and $m^* < n^*$ by the usual ordering on $\mathbb{N}$. We write $m^* \leq n^*$ iff $m^* < n^*$ or $m^* = n^*$.

We use $\Rightarrow, \Leftrightarrow, \&, \forall, \exists$ as connectives and quantifiers in the informal metalanguage. For a map $F : S \to S$ on some set S, we define $F^0(M) = M$ and $F^{n+1}(M) = F(F^n(M))$.

1.2.2. Some notation concerning normalization

In this subsection R denotes a notion of reduction on some set S, and $\to_R$ denotes the corresponding reduction relation.

1.2.1. Definition. A finite or infinite sequence

$$M_0 \to_R M_1 \to_R \ldots$$

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, and then we write $M_0 \twoheadrightarrow_R^n M_n$. If the sequence is infinite, it has length ∞.

1.2.2. Definition.

$$\begin{array}{ll} \infty_R & = \{M \mid M \text{ has an infinite } R\text{-reduction path}\}. \\ n_R & = \{M \mid M \text{ has an } R\text{-reduction path of length } n\}. \\ \mathrm{NF}_R & = \{M \mid M \text{ has no } R\text{-reduction path of length 1 or more}\}. \\ \mathrm{SN}_R & = \{M \mid M \text{ has no infinite } R\text{-reduction path}\}. \\ \mathrm{WN}_R & = \{M \mid M \text{ has a finite } R\text{-reduction path ending in an } N \in \mathrm{NF}_R\}. \end{array}$$

In the notation n_R, we require $n \in \mathbb{N}$.

1.2.3. Definition.

$\mathrm{CR}_R = \{M \mid \text{for all } L, N, \text{ if } L \;{}_R\!\twoheadleftarrow M \twoheadrightarrow_R N \text{ then } L \twoheadrightarrow_R K \;{}_R\!\twoheadleftarrow N \text{ for a } K\}.$
$\mathrm{FB}_R = \{M \mid M \to_R N \text{ for only finitely many different } N\}.$

1.2.4. Terminology.

(i) $M \in \mathrm{NF}_R \Leftrightarrow M$ is an *R-normal form.*

(ii) $M \in \mathrm{SN}_R \Leftrightarrow M$ is *R-strongly normalizing.*

(iii) $M \in \mathrm{WN}_R \Leftrightarrow M$ is *R-weakly normalizing.*

(iv) $M \in \mathrm{CR}_R \Leftrightarrow M$ is *R-Church-Rosser.*

(v) $M \in \mathrm{FB}_R \Leftrightarrow M$ is *R-finitely branching.*

We often omit R, relying on the context to resolve the ambiguity. When R is a notion of reduction on a set S, and $M \in \mathrm{FB}_R$ for all $M \in S$, we simply write FB_R. Similarly with the other sets introduced above.

1.2.5. Lemma. *Assume* FB_R. *Then* $M \in \infty_R \Leftrightarrow \forall n \in \mathbb{N} : M \in n_R$.

Proof. "$\Rightarrow$" is obvious; "$\Leftarrow$" is by König's Lemma. $\square$

We shall denote by $s_R(M) \in \mathbb{N}^*$ the length of a shortest finite reduction path from M to normal form, if a finite reduction path to normal form exists; otherwise $s_R(M) = \infty$. Also, $l_R(M) \in \mathbb{N}^*$ denotes the length of a longest finite reduction path from M to normal form, if there is an upper bound on the length of these reduction paths; otherwise $l_R(M) = \infty$. In symbols:

1.2.6. Definition.

(i) $s_R(M) = \min\{n \mid \exists N \in \mathrm{NF}_R : M \twoheadrightarrow_R^n N\}$.

(ii) $l_R(M) = \max\{n \mid \exists N \in \mathrm{NF}_R : M \twoheadrightarrow_R^n N\}$.

1.2.7. Lemma. *Assume* CR_R, FB_R. *Then*

(i) $M \in \mathrm{WN}_R \Leftrightarrow s_R(M) < \infty$.

(ii) $M \in \infty_R \Leftrightarrow l_R(M) = \infty$.

Proof.

(i) "$\Rightarrow$": If $M \in \mathrm{WN}_R$ then $M \twoheadrightarrow_R^n N \in \mathrm{NF}_R$ for an $n \in \mathbb{N}$, so $s_R(M) < \infty$.
"$\Leftarrow$": If $s_R(M) < \infty$ then $M \twoheadrightarrow_R^n N \in \mathrm{NF}_R$ for an $n \in \mathbb{N}$, so $M \in \mathrm{WN}_R$.

(ii) "$\Rightarrow$": Assume $M \in \infty_R$.

1. $M \notin \mathrm{WN}_R$. Then $l_R(M) = \infty$.

2. $M \in \mathrm{WN}_R$. Then $M \twoheadrightarrow_R N \in \mathrm{NF}_R$ for some N. Since $M \in \infty_R$, for any $n \in \mathbb{N}$ there is K such that $M \twoheadrightarrow^n_R K$. By CR_R, $K \twoheadrightarrow_R N$. Thus, for any $m \in \mathbb{N}$ there is $n > m$ such that $M \twoheadrightarrow^n_R N \in \mathrm{NF}_R$. Then $l_R(M) = \infty$.

"$\Leftarrow$": Assume $l_R(M) = \infty$. There are two ways this can happen.

1. $M \notin \mathrm{WN}_R$. Then $M \in \infty_R$.
2. For arbitrarily large $n \in \mathbb{N}$ there is $N \in \mathrm{NF}_\beta$ with $M \twoheadrightarrow^n_R N$. Then $M \in \infty_R$ by Lemma 1.2.5. □

1.2.8. Remark. Although seemingly trivial, the above proof uses the rules $\min \emptyset = \max \emptyset = \max U = \infty$ (U unbounded) in subtle ways. For instance, as shown in (ii) "$\Rightarrow$", if $m \in \infty_R$, then $\{n \mid \exists N \in \mathrm{NF}_R : M \twoheadrightarrow^n_R N\}$ is either empty (if $M \notin \mathrm{WN}_R$) or unbounded (if $M \in \mathrm{WN}_R$). In either event, the two latter conventions imply $l_R(M) = \infty$.

1.2.9. Remark. The statement formulated in Lemma 1.2.7(ii) will be used at various places later; an equivalent statement is: $M \in \mathrm{SN}_R \Leftrightarrow l_R(M) < \infty$.

1.2.3. Classification of strategies

In this subsection we introduce rigorously the classification of reduction strategies that was mentioned informally in the introduction. Throughout the subsection, R denotes a notion of reduction on some set S, and $\to_R$ denotes the corresponding reduction relation.

1.2.10. Definition (Barendregt et al. [3, 5]).

(i) An *R-reduction strategy* is a map $F : S \to S$ such that $M \to_R F(M)$ if $M \notin \mathrm{NF}_R$, and $F(M) = M$ otherwise.

(ii) Let F be an R-reduction strategy. Define

$$L_F(M) = \min\{n \mid F^n(M) \in \mathrm{NF}_R\}.$$

The *F-reduction path* from M is the reduction path

$$M \to_R F(M) \to_R F^2(N) \to_R \cdots$$

of length $L_F(M)$.

1.2.11. Remark. Reduction strategies are *history insensitive;* that is, given some $M \in \Lambda_K$, the act of a reduction strategy on M is independent on how we might have arrived at M. For instance, "for any $M \in \Lambda_K$, reduce alternately the left-most and right-most β-redex, beginning with the left-most one" does not specify a reduction strategy; a reduction strategy receives a term as input and must return as output another term that arises from the former by one reduction step.

Barendregt et al. [3, 5] use the terminology *one-step reduction strategy* for what we call *reduction strategy.* In the following definition, (ii)-(iv) are also taken from [3, 5], but what we call *minimal* is there called L-1-*optimal.*

1.2.12. Definition. Let F be an R-reduction strategy.

(i) F is R-*maximal* iff $L_F(M) = l_R(M)$.

(ii) F is R-*minimal* iff $L_F(M) = s_R(M)$.

(iii) F is R-*perpetual* iff $M \in \infty_R \Rightarrow L_F(M) = \infty$.

(iv) F is R-*normalizing* iff $M \in \mathrm{WN}_R \Rightarrow L_F(M) < \infty$.

This classification of strategies is "global" in that it is formulated in terms of the whole reduction path of the strategy. The following formulations of minimality and maximality are "local" in that they are formulated in terms of one step of the strategy. The local classifications have the advantage that they give rise to analogous classifications of redexes.

1.2.13. Lemma. *Assume* CR_R, FB_R. *Let* F *be an* R-*reduction strategy.*

(i) F *is* R-*minimal iff for all* $M \notin \mathrm{NF}_R$: $s_R(M) = s_R(F(M)) + 1$.

(ii) F *is* R-*maximal iff for all* $M \notin \mathrm{NF}_R$: $l_R(M) = l_R(F(M)) + 1$.

Proof.

(i) "$\Rightarrow$": Assume F is R-minimal. Then, for any $M \notin \mathrm{NF}_R$,

$$\begin{array}{rcl} s_R(M) & = & L_F(M) \\ & = & \min\{n \mid F^n(M) \in \mathrm{NF}_R\} \\ & = & \min\{n \mid F^n(F(M)) \in \mathrm{NF}_R\} + 1 \\ & = & L_F(F(M)) + 1 \\ & = & s_R(F(M)) + 1. \end{array}$$

"$\Leftarrow$": Assume for all $M \notin \mathrm{NF}_R$ that $s_R(M) = s_R(F(M)) + 1$. If $s_R(M) = \infty$, then also $L_F(M) = \infty$. Now assume $s_R(M) < \infty$. We show by induction on $s_R(M)$ that $s_R(M) = L_F(M)$.

1. $s_R(M) = 0$. Then $M \in \mathrm{NF}_R$, so $L_F(M) = 0$.
2. $0 < s_R(M) < \infty$. Then $M \notin \mathrm{NF}_R$. By the induction hypothesis,

$$\begin{array}{rcl} s_R(M) & = & s_R(F(M)) + 1 \\ & = & L_F(F(M)) + 1 \\ & = & L_F(M). \end{array}$$

(ii) "$\Rightarrow$": Assume F is maximal. Then, for any $M \notin \mathrm{NF}_R$,

$$\begin{array}{rcl} l_R(M) & = & L_F(M) \\ & = & L_F(F(M)) + 1 \\ & = & l_R(F(M)) + 1. \end{array}$$

"$\Leftarrow$": Assume for all $M \notin \mathrm{NF}_R$ that $l_R(M) = l_R(F(M)) + 1$. If $L_F(M) = \infty$, then, by Lemma 1.2.7, also $l_R(M) = \infty$. Now assume $L_F(M) < \infty$. We show $l_R(M) = L_F(M)$ by induction on $L_F(M)$.

1. $L_F(M) = 0$. Then $M \in \mathrm{NF}_R$, so $l_R(M) = 0$.
2. $0 < L_F(M) < \infty$. Then $N \notin \mathrm{NF}_R$. By the induction hypothesis,

$$\begin{array}{rcl} L_F(M) & = & L_F(F(M)) + 1 \\ & = & l_R(F(M)) + 1 \\ & = & l_R(M). \end{array}$$

Note that we need Lemma 1.2.7 in (ii), but not in (i). □

The following gives another local formulation of perpetuality and maximality, due to Bergstra and Klop [18] and Regnier [108], respectively.

1.2.14. LEMMA. *Assume* CR_R, FB_R. *Let F be an R-reduction strategy.*

(i) *F is* R-perpetual *iff for all M:* $M \in \infty_R \Rightarrow F(M) \in \infty_R$.

(ii) *F is* R-maximal *iff for all M and $n \geq 1$:* $M \in n_R \Rightarrow F(M) \in (n-1)_R$.

PROOF.

(i) "$\Rightarrow$": Assume $M \in \infty_R$. By assumption, $L_F(M) = \infty$, i.e., the path $M \to_R F(M) \to_R F^2(M) \to_R \ldots$ is infinite, so $F(M) \in \infty_R$.
"$\Leftarrow$": Assume $M \in \infty_R$. By induction on n show that $F^n(M) \in \infty_R$, in particular $F^n(M) \notin \mathrm{NF}_R$, so $L_F(M) = \infty$.

(ii) "$\Rightarrow$": Assume that $M \in n_R$. By CR_R, $n \leq l_R(M) = L_F(M)$, i.e., $F^{n-1}(M) \notin \mathrm{NF}_R$, so $F(M) \in (n-1)_R$.
"$\Leftarrow$": If $L_F(M) = \infty$, then, by Lemma 1.2.7, $l_R(M) = \infty$. Assume $L_F(M) < \infty$. We show $L_F(M) = l_R(M)$ by induction on $L_F(M)$.

1. $L_F(M) = 0$. Then $M \in \mathrm{NF}_R$, so $l_R(M) = 0$.
2. $0 < L_F(M) < \infty$. Then $M \notin \mathrm{NF}_R$. By the induction hypothesis and Lemma 1.2.13,

$$\begin{array}{rcl} L_F(M) & = & L_F(F(M)) + 1 \\ & = & l_R(F(M)) + 1 \\ & = & l_R(M). \end{array}$$ □

1.2.15. PROPOSITION. *Assume* $\mathrm{CR}_R, \mathrm{FB}_R$. *Let F be an R-reduction strategy.*

(i) *If F is R-maximal then F is R-perpetual.*

(ii) *If F is R-minimal then F is R-normalizing.*

PROOF.

(i) If $M \in \infty_R$ then, by Lemma 1.2.7, $L_F(M) = l_R(M) = \infty$.

(ii) If $M \in \mathrm{WN}_R$ then, by Lemma 1.2.7, $L_F(M) = s_R(M) < \infty$. $\square$

1.2.16. REMARK. No other general containment exists between our four types of strategies than the two mentioned above.

Perpetual reduction strategies are often useful to prove properties about infinite reduction paths. In these cases we are usually not interested in how the strategy behaves on strongly normalizing terms. This motivates the following.

1.2.17. DEFINITION. A *partial, perpetual R-reduction strategy* is a mapping $F : \infty_R \to \infty_R$ such that for all $M \in \infty_R$: $M \to_R F(M)$.

1.2.4. Classification of redexes

In this subsection we introduce rigorously the classification of redexes from the introduction. Throughout the subsection, R denotes a notion of reduction on Λ_K, and $\to_R$ denotes the corresponding reduction relation.

In the following definition, (i) is taken from [18].

1.2.18. DEFINITION. Let Δ be an R-redex with contractum Δ'.

(i) Δ is *R-perpetual* iff, for all C: $C[\Delta] \in \infty_R \Rightarrow C[\Delta'] \in \infty_R$.

(ii) Δ is *R-maximal* iff, for all $n \geq 1$ and C: $C[\Delta] \in n_R \Rightarrow C[\Delta'] \in (n-1)_R$.

1.2.19. REMARK. As was the case for strategies, one can vary the formulation of perpetual and maximal redexes; we shall not study such equivalent formulations.

1.2.20. DEFINITION. Let Δ be an R-redex with contractum Δ'. Then Δ is *R-minimal* iff for all C: $s_R(C[\Delta]) = s_R(C[\Delta']) + 1$.

1.2.21. DISCUSSION. A strategy that always contracts perpetual redexes is perpetual. Similarly, strategies that always contract maximal and minimal redexes are maximal and minimal, respectively. This is easy to verify simply by noting the analogy between on the one hand the local formulations of perpetual, maximal, and minimal strategies in Lemmas 1.2.13 and 1.2.14, and on the other hand the formulations of perpetual, maximal, and minimal redexes in Definitions 1.2.18 and 1.2.20.

Perpetual strategies may also contract non-perpetual redexes. The reason is that a *strategy* is confronted with a redex in a given context, and needs only to make sure that contracting the redex in *this particular* context preserves the possibility, if present, of an infinite reduction. A perpetual *redex*, on the other hand, must preserve the existence of infinite reduction paths in *all* contexts. Similar remarks apply to maximal and minimal strategies.

We do not know how to give a formulation of the notion of a *normalizing redex* which satisfies the property that a strategy contracting only normalizing redexes is itself normalizing. This problem stems from the fact that the above classifications of redexes were derived from *local* formulations of the notions of a perpetual, maximal, and minimal strategy, whereas we have no local formulation of the notion of a normalizing strategy.

1.2.22. PROPOSITION. *Assume* FB_R. *A redex which is* R*-maximal is also* R*-perpetual.*

PROOF. Given R-maximal redex Δ with contractum Δ' and a context C, assume $C[\Delta] \in \infty_R$. To prove $C[\Delta'] \in \infty_R$ it suffices by Lemma 1.2.5 to show that $C[\Delta'] \in n_R$ for all $n \in \mathbb{N}$. Since $C[\Delta] \in \infty_R$ we have by Lemma 1.2.5 for all $n \in \mathbb{N}$, $C[\Delta] \in n_R$ and thereby $C[\Delta] \in (n+1)_R$. Thus $C[\Delta'] \in n_R$ for all $n \in \mathbb{N}$ by maximality. $\square$

1.2.23. REMARK. The converse of the preceding proposition does not hold.

1.3. Perpetual and maximal strategies

In this section we study perpetual and maximal β-reduction strategies. The first subsection presents the *fundamental lemma of perpetuality.* The second subsection presents two recent characterizations of strongly normalizing terms due to van Raamsdonk and Severi and to Xi, respectively, and shows that the core of these characterizations is made up of the fundamental lemma of perpetuality and a certain lexicographic induction principle. The third subsection presents two (partial) perpetual β-reduction strategies; the proof of perpetuality in each case uses the fundamental lemma of perpetuality.

The fourth subsection presents the *fundamental lemma of maximality,* analogous to the fundamental lemma of perpetuality. The fifth subsection presents an effective, maximal β-reduction strategy; the proof of maximality uses the fundamental lemma of maximality. The sixth subsection shows that to compute an upper bound on the length of a longest β-reduction path for some term, one cannot do better, in a certain sense, than try to reduce the term to normal form and count the number of steps along the way.

The property CR_β is used freely in this and the following sections.

1.3.1. The fundamental lemma of perpetuality

The following lemma is used in many strong normalization proofs in the literature—see Section 1.5. As will be seen below, the lemma is also useful to show that reduction strategies are perpetual.

1.3.1. Lemma (Fundamental lemma of perpetuality). *Assume that* $M_1 \in \mathrm{SN}_\beta$ *if* $x \notin \mathrm{FV}(M_0)$. *For all* $n \geq 1$*:*

$$M_0\{x := M_1\}\, M_2 \dots M_n \in \mathrm{SN}_\beta \;\Rightarrow\; (\lambda x.M_0)\, M_1 \dots M_n \in \mathrm{SN}_\beta.$$

Proof. Let $M_0\{x := M_1\}\, M_2 \dots M_n \in \mathrm{SN}_\beta$. Then $M_0, M_2, \dots, M_n \in \mathrm{SN}_\beta$. If $x \notin \mathrm{FV}(M_0)$, then, by assumption, $M_1 \in \mathrm{SN}_\beta$. If $x \in \mathrm{FV}(M_0)$, then also $M_1 \subseteq M_0\{x := M_1\}\, M_2 \dots M_n$, so $M_1 \in \mathrm{SN}_\beta$. If $(\lambda x.M_0)\, M_1 \dots M_n \in \infty_\beta$, then any infinite reduction must therefore have the form

$$\begin{array}{rll} (\lambda x.M_0)\, M_1 \dots M_n & \twoheadrightarrow_\beta & (\lambda x.M_0')\, M_1' \dots M_n' \\ & \to_\beta & M_0'\{x := M_1'\}\, M_2' \dots M_n' \\ & \to_\beta & \dots . \end{array}$$

Since

$$M \twoheadrightarrow_\beta M' \;\&\; N \twoheadrightarrow_\beta N' \;\Rightarrow\; M\{x := N\} \twoheadrightarrow_\beta M'\{x := N'\},$$

there is an infinite reduction sequence

$$\begin{array}{rll} M_0\{x := M_1\}\, M_2 \dots M_n & \twoheadrightarrow_\beta & M_0'\{x := M_1'\}\, M_2' \dots M_n' \\ & \to_\beta & \dots , \end{array}$$

contradicting $M_0\{x := M_1\}\, M_2 \dots M_n \in \mathrm{SN}_\beta$. □

1.3.2. Corollary. *If* $M_1 \in \mathrm{SN}_\beta$*, then for all* $n \geq 1$*:*

$$M_0\{x := M_1\}\, M_2 \dots M_n \in \mathrm{SN}_\beta \;\Rightarrow\; (\lambda x.M_0)\, M_1 \dots M_n \in \mathrm{SN}_\beta.$$

Proof. By the fundamental lemma of perpetuality. □

1.3.3. Remark. The fundamental lemma of perpetuality gives a condition ensuring that a contraction $(\lambda x.M_0)\, M_1 \dots M_n \to_\beta M_0\{x := M_1\}\, M_2 \dots M_n$ preserves the possibility, if present, of an infinite reduction. The corollary requires a slightly simpler condition.

1.3.2. Two characterizations of strongly normalizing terms

Next we introduce two characterizations of SN_β due to van Raamsdonk and Severi [106] (also [105, 117]) and to Xi [140], respectively.

1.3.4. Definition. Let $X \subseteq \Lambda_K$ be the smallest set closed under:

(i) $M_1, \dots, M_n \in X \;\Rightarrow\; x\, M_1 \dots M_n \in X$.

(ii) $M \in X \;\Rightarrow\; \lambda x.M \in X$.

(iii) $M_1 \in X \;\&\; M_0\{x := M_1\}\, M_2 \dots M_n \in X \;\Rightarrow\; (\lambda x.M_0)\, M_1 \dots M_n \in X$.

1.3.5. PROPOSITION. $\mathrm{SN}_\beta = X$.

PROOF. We first prove $M \in \mathrm{SN}_\beta \Rightarrow M \in X$ by induction on lexicographically ordered pairs $\langle l_\beta(M), \|M\|\rangle$.

1. $M \equiv x\, P_1 \ldots P_n$. Then $P_1, \ldots, P_n \in \mathrm{SN}_\beta$. By the induction hypothesis $P_1, \ldots, P_n \in X$, so $M \in X$.
2. $M \equiv \lambda x.P$. Similar to Case 1.
3. $M \equiv (\lambda x.P_0)\, P_1 \ldots P_n$. Then $P_1 \in \mathrm{SN}_\beta$, $P_0\{x := P_1\}\, P_2 \ldots P_n \in \mathrm{SN}_\beta$, so by the induction hypothesis, $P_1 \in X$, $P_0\{x := P_1\}\, P_2 \ldots P_n \in X$, so $M \in X$.

It remains to prove $M \in X \;\Rightarrow\; M \in \mathrm{SN}_\beta$. We proceed by induction on the derivation of $M \in X$.

1. $M \equiv x\, P_1 \ldots P_n$ where $P_1, \ldots, P_n \in X$. By the induction hypothesis $P_1, \ldots, P_n \in \mathrm{SN}_\beta$, so $M \in \mathrm{SN}_\beta$.
2. $M \equiv \lambda x.P$. Similar to Case 1.
3. $M \equiv (\lambda x.P_0)\, P_1 \ldots P_n$ where $P_1 \in X$, $P_0\{x := P_1\}\, P_2 \ldots P_n \in X$. By the induction hypothesis, $P_1 \in \mathrm{SN}_\beta$, $P_0\{x := P_1\}\, P_2 \ldots P_n \in \mathrm{SN}_\beta$, so by the fundamental lemma of perpetuality, $M \in \mathrm{SN}_\beta$. $\square$

1.3.6. REMARK. Given an assertion of form $M \in \mathrm{SN}_\beta \Rightarrow P(M)$, we may prove instead $M \in X \Rightarrow P(M)$ by induction on the derivation of $M \in X$; this is very similar to proving the original assertion by induction on lexicographically ordered pairs $\langle l_\beta(M), \|M\|\rangle$. Given an assertion of form $P(M) \Rightarrow M \in \mathrm{SN}_\beta$, we may prove instead $P(M) \Rightarrow M \in X$; this is very similar to proving the original assertion and using the fundamental lemma of perpetuality in the case $M \equiv (\lambda x.P_0)\, P_1 \ldots P_n$. Thus, the two main ingredients in the proof of Proposition 1.3.5—lexicographic induction on $\langle l_\beta(M), \|M\|\rangle$ and the fundamental lemma of perpetuality—are used implicitly when one uses X to reason about SN_β. Van Raamsdonk and Severi [106] prove strong normalization results in λ-calculus using this characterization—see Sections 1.5 and 1.6.

1.3.7. DEFINITION. Define $F_l : \Lambda_K \to \Lambda_K$ as follows. If $M \in \mathrm{NF}_\beta$ then $F_l(M) = M$; otherwise,

$$
\begin{array}{lcll}
F_l(x\, \vec{P}\, Q\, \vec{R}) & = & x\, \vec{P}\, F_l(Q)\, \vec{R} & \text{if } \vec{P} \in \mathrm{NF}, Q \notin \mathrm{NF}_\beta \\
F_l(\lambda x.P) & = & \lambda x.F_l(P) & \\
F_l((\lambda x.P)\, Q\, \vec{R}) & = & P\{x := Q\}\, \vec{R}. &
\end{array}
$$

Write $M \to_l N$ if $M \notin \mathrm{NF}_\beta$ and $F_l(M) \equiv N$, and $M \in \infty_l$ if $L_{F_l}(M) = \infty$.

1.3.8. Definition. Define the relation $\triangleright$ by:

$$\triangleright \;=\; \sqsupset \;\cup\; \to_l$$

where $\sqsupset$ denotes the smallest relation closed under the rules:

$$\lambda x.M \sqsupset M \qquad M_1\, M_2 \sqsupset M_1 \qquad M_1\, M_2 \sqsupset M_2.$$

Define

$$\mathcal{H}(M_0) = \max\{n \mid M_0 \triangleright M_1 \triangleright \ldots \triangleright M_n\} \in \mathbb{N}^*.$$

1.3.9. Proposition. $\mathrm{SN}_\beta = \{M \in \Lambda_K \mid \mathcal{H}(M) < \infty\}$.

Proof. We first prove $M \in \mathrm{SN}_\beta \Rightarrow \mathcal{H}(M) < \infty$ by induction on lexicographically ordered pairs $\langle l_\beta(M), \|M\|\rangle$. First note that if $\mathcal{H}(M_0) = \infty$ then by König's lemma there is an infinite sequence $M_0 \triangleright M_1 \triangleright \ldots$, and so there is an M_1 with $M_0 \triangleright M_1$ and $\mathcal{H}(M_1) = \infty$.

1. $M \equiv x$. Then $\mathcal{H}(M) = 0 < \infty$.
2. $M \equiv P\,Q$. Then $P, Q \in \mathrm{SN}_\beta$. Moreover, if $M \to_l M'$ then $M' \in \mathrm{SN}_\beta$. By the induction hypothesis $\mathcal{H}(P) < \infty$, $\mathcal{H}(Q) < \infty$, and $\mathcal{H}(M') < \infty$. Thus, for all N with $M \triangleright N$, $\mathcal{H}(N) < \infty$. Thus, $\mathcal{H}(M) < \infty$.
3. $M \equiv \lambda x.P$. Similar to Case 2.

Next we prove $\mathcal{H}(M) < \infty \Rightarrow M \in \mathrm{SN}_\beta$ by induction on $\mathcal{H}(M)$.

1. $M \equiv x\, P_1 \ldots P_n$. Then $\mathcal{H}(P_1) < \infty, \ldots, \mathcal{H}(P_n) < \infty$. By the induction hypothesis $P_1, \ldots, P_n \in \mathrm{SN}_\beta$, so $M \in \mathrm{SN}_\beta$.
2. $M \equiv \lambda x.P$. Similar to Case 1.
3. $M \equiv (\lambda x.P_0)\, P_1 \ldots P_n$. Then $\mathcal{H}(P_0\{x := P_1\}\, P_2 \ldots P_n) < \infty$ and $\mathcal{H}(P_1) < \infty$. By the induction hypothesis, $P_0\{x := P_1\}\, P_2 \ldots P_n \in \mathrm{SN}_\beta$ and $P_1 \in \mathrm{SN}_\beta$. By the fundamental lemma of perpetuality it then follows that $M \in \mathrm{SN}_\beta$. $\square$

1.3.10. Remark. The point in Remark 1.3.6 may be repeated with "$M \in X$" replaced by "$\mathcal{H}(M) < \infty$." Xi [140] proves strong normalization results in λ-calculus using this characterization—see Sections 1.5 and 1.6.

1.3.11. Remark. The above characterizations of SN_β, especially the second one, are similar to the *successor relation,* defined by Terlouw [128], who proves this relation to be well-founded and who uses it to show a connection between higher type levels and transfinite recursion (see also [130]).

Whether one should prove results in λ-calculus using the fundamental lemma of perpetuality and lexicographic induction, or one should use one of the characterizations by van Raamsdonk and Severi and by Xi, seems to be a matter of taste.

1.3.3. Some perpetual β-reduction strategies

The following strategy is due to Bergstra and Klop [18].

1.3.12. Definition. Define $F_1 : \infty_\beta \to \Lambda_K$ by:

$$\begin{array}{lcll} F_1(x\,\vec{P}\,Q\,\vec{R}) & = & x\,\vec{P}\,F_1(Q)\,\vec{R} & \text{if } \vec{P} \in \mathrm{SN}_\beta, Q \notin \mathrm{SN}_\beta \\ F_1(\lambda x.P) & = & \lambda x.F_1(P) & \\ F_1((\lambda x.P)\,Q\,\vec{R}) & = & P\{x := Q\}\,\vec{R} & \text{if } Q \in \mathrm{SN}_\beta \\ F_1((\lambda x.P)\,Q\,\vec{R}) & = & (\lambda x.P)\,F_1(Q)\,\vec{R} & \text{if } Q \notin \mathrm{SN}_\beta. \end{array}$$

1.3.13. Remark. For every $M \in \infty_\beta$ either $M \equiv x\,P_1 \ldots P_n$ where $n \geq 1$ and $P_i \in \infty_\beta$ for some i, or $M \equiv \lambda x.P$, or $M \equiv (\lambda x.P_0)\,P_1 \ldots P_n$ where $n \geq 1$. It follows that F_1 is defined on all elements of ∞_β.

1.3.14. Proposition. *F_1 is a partial, perpetual β-reduction strategy.*

Proof. By induction on the size of M prove that $M \in \infty_\beta \Rightarrow F_1(M) \in \infty_\beta$; the only non-trivial case is when $M \equiv (\lambda x.P)\,Q\,\vec{R}$ and $Q \in \mathrm{SN}_\beta$, in which case use Corollary 1.3.2. □

The following strategy is a variant of a strategy in [120].

1.3.15. Definition. Define $F_2 : \infty_\beta \to \Lambda_K$ by:

$$\begin{array}{lcll} F_2(x\,\vec{P}\,Q\,\vec{R}) & = & x\,\vec{P}\,F_2(Q)\,\vec{R} & \text{if } \vec{P} \in \mathrm{SN}_\beta, Q \notin \mathrm{SN}_\beta \\ F_2(\lambda x.P) & = & \lambda x.F_2(P) & \\ F_2((\lambda x.P)\,Q\,\vec{R}) & = & P\{x := Q\}\,\vec{R} & \text{if } P \in \mathrm{SN}_\beta,\ Q \in \mathrm{SN}_\beta \\ F_2((\lambda x.P)\,Q\,\vec{R}) & = & (\lambda x.F_2(P))\,Q\,\vec{R} & \text{if } P \notin \mathrm{SN}_\beta \\ F_2((\lambda x.P)\,Q\,\vec{R}) & = & (\lambda x.P)\,F_2(Q)\,\vec{R} & \text{if } P \in \mathrm{SN}_\beta, Q \notin \mathrm{SN}_\beta. \end{array}$$

1.3.16. Proposition. *F_2 is partial, perpetual β-reduction strategy.*

Proof. By induction on the size of M prove that $M \in \infty_\beta \Rightarrow F_2(M) \in \infty_\beta$; the only non-trivial case is when $M \equiv (\lambda x.P)\,Q\,\vec{R}$ and $P, Q \in \mathrm{SN}_\beta$, in which case use Corollary 1.3.2. □

1.3.4. The fundamental lemma of maximality

The following lemma is used in some of the strong normalization proofs in the literature which, in addition to proving strong normalization, establish upper bounds for the length of reduction paths—see Section 1.5.

1.3.17. Definition. Define for any variable x the map $\notin_x \colon \Lambda_K \to \{0,1\}$ by:

$$\notin_x(M) = \begin{cases} 1 & \text{if } x \notin \mathrm{FV}(M) \\ 0 & \text{if } x \in \mathrm{FV}(M). \end{cases}$$

1.3.18. LEMMA (Fundamental lemma of maximality). *For all* $n \geq 1$,

$$l_\beta((\lambda x.M_0)\, M_1 \ldots M_n) = l_\beta(M_0\{x := M_1\}\, M_2 \ldots M_n) + \not\in_x(M_0) \cdot l_\beta(M_1) + 1.$$

PROOF. If $l_\beta((\lambda x.M_0)\; M_1 \ldots M_n) = \infty$, then by Lemma 1.2.7 and the fundamental lemma of perpetuality, also $l_\beta(M_0\{x := M_1\}\; M_2 \ldots M_n) = \infty$ or $\not\in_x(M_1) \cdot l_\beta(M_1) = \infty$. Thus, in this case the equality holds.

If $l_\beta((\lambda x.M_0)M_1M_2 \ldots M_n) < \infty$, then $M_0, \ldots, M_n \in \mathrm{SN}_\beta$ by Lemma 1.2.7. We consider two cases.

1. $x \notin \mathrm{FV}(M_0)$. A longest reduction from $(\lambda x.M_0)\; M_1 \ldots M_n$ has the form

$$\begin{array}{rll} (\lambda x.M_0)\; M_1 \ldots M_n & \twoheadrightarrow_\beta^m & (\lambda x.M_0')\; M_1' \ldots M_n' \\ & \to_\beta & M_0'\; M_2' \ldots M_n' \\ & \twoheadrightarrow_\beta^k & K \in \mathrm{NF}_\beta, \end{array}$$

where $M_0 \twoheadrightarrow_\beta^{m_0} M_0', \ldots, M_n \twoheadrightarrow_\beta^{m_n} M_n'$, and where $m_0 + \ldots + m_n = m$, $l_\beta(M_1) = m_1$, and $l_\beta((\lambda x.M_0)\; M_1 \ldots M_n) = m + k + 1$. Then

$$\begin{array}{rll} (\lambda x.M_0)\; M_1 \ldots M_n & \twoheadrightarrow_\beta^{m_1} & (\lambda x.M_0)\; M_1'\; M_2 \ldots M_n \\ & \to_\beta & M_0\; M_2 \ldots M_n \\ & \twoheadrightarrow_\beta^{m-m_1} & M_0'\; M_2' \ldots M_n' \\ & \twoheadrightarrow_\beta^k & K \in \mathrm{NF}_\beta \end{array}$$

is another longest reduction path from $(\lambda x.M_0)\; M_1 \ldots M_n$. Thus, $M_0\; M_2 \ldots M_n \twoheadrightarrow_\beta^{m-m_1+k} K$ is also a longest reduction path from $M_0\; M_2 \ldots M_n$, i.e., $l_\beta(M_0\; M_2 \ldots M_n) = m - m_1 + k$. Thus,

$$\begin{array}{rcl} l_\beta((\lambda x.M_0)\; M_1 \ldots M_n) & = & m + k + 1 \\ & = & (m - m_1 + k) + m_1 + 1 \\ & = & l_\beta(M_0\; M_2 \ldots M_n) + l_\beta(M_1) + 1. \end{array}$$

2. $x \in \mathrm{FV}(M_0)$. A longest reduction from $(\lambda x.M_0)\; M_1 \ldots M_n$ has the form

$$\begin{array}{rll} (\lambda x.M_0)\; M_1 \ldots M_n & \twoheadrightarrow_\beta^m & (\lambda x.M_0')\; M_1' \ldots M_n' \\ & \to_\beta & M_0'\{x := M_1'\}\; M_2' \ldots M_n' \\ & \twoheadrightarrow_\beta^k & K \in \mathrm{NF}_\beta, \end{array}$$

where $M_0 \twoheadrightarrow_\beta^{m_0} M_0', \ldots, M_n \twoheadrightarrow_\beta^{m_n} M_n'$, and where $m_0 + \ldots + m_n = m$ and $l_\beta((\lambda x.M_0)\; M_1 \ldots M_n) = m + k + 1$. Since

$$M \twoheadrightarrow_\beta^m M' \;\&\; N \twoheadrightarrow_\beta^n N' \;\Rightarrow\; M\{x := N\} \twoheadrightarrow_\beta^{m+n\cdot\|M\|_x} M'\{x := N'\},$$

also

$$\begin{array}{rll} (\lambda x.M_0)\; M_1 \ldots M_n & \to_\beta & M_0\{x := M_1\}\; M_2 \ldots M_n \\ & \twoheadrightarrow_\beta^{m_0+m_1\cdot\|M_0\|_x} & M_0'\{x := M_1'\}\; M_2 \ldots M_n \\ & \twoheadrightarrow_\beta^{m_2+\ldots+m_n} & M_0'\{x := M_1'\}\; M_2' \ldots M_n' \\ & \twoheadrightarrow_\beta^k & K \in \mathrm{NF}_\beta. \end{array}$$

Since $\|M_0\|_x \geq 1$, $m_0 + m_1 \cdot \|M_0\|_x + m_2 \ldots + m_n + k + 1 \geq m + k + 1$, so this is, in fact, another longest reduction from $(\lambda x.M_0)\, M_1 \ldots M_n$, so $l_\beta(M_0\{x := M_1\}\, M_2 \ldots M_n) = m_0 + m_1 \cdot \|M_0\|_x + m_2 \ldots + m_n + k$. Thus,

$$\begin{array}{rcl} l_\beta((\lambda x.M_0)\, M_1 \ldots M_n) & = & m_0 + m_1 + \ldots + m_n + k + 1 \\ & \leq & m_0 + m_1 \cdot \|M_0\|_x + m_2 + \ldots + m_n + k + 1 \\ & = & l_\beta(M_0\{x := M_1\}\, M_2 \ldots M_n) + 1. \end{array}$$

The converse inequality is trivial. □

1.3.19. Corollary. *For all $n \geq 1$,*

$$l_\beta((\lambda x.M_0)\, M_1 \ldots M_n) \leq l_\beta(M_0\{x := M_1\}\, M_2 \ldots M_n) + l_\beta(M_1) + 1.$$

Proof. By the fundamental lemma of maximality. □

1.3.20. Remark. The fundamental lemma of perpetuality and its corollary are special cases of the fundamental lemma of maximality and its corollary, respectively.

1.3.5. An effective maximal strategy

The following strategy is due to Barendregt et al. [3, 5].

1.3.21. Definition. Define $F_\infty : \Lambda_K \to \Lambda_K$ as follows. If $M \in \mathrm{NF}_\beta$ then $F_\infty(M) = M$; otherwise

$$\begin{array}{llll} F_\infty(x\, \vec{P}\, Q\, \vec{R}) & = & x\, \vec{P}\, F_\infty(Q)\, \vec{R} & \text{if } \vec{P} \in \mathrm{NF}_\beta, Q \notin \mathrm{NF}_\beta \\ F_\infty(\lambda x.P) & = & \lambda x.F_\infty(P) & \\ F_\infty((\lambda x.P)\, Q\, \vec{R}) & = & P\{x := Q\}\, \vec{R} & \text{if } x \in \mathrm{FV}(P) \text{ or } Q \in \mathrm{NF}_\beta \\ F_\infty((\lambda x.P)\, Q\, \vec{R}) & = & (\lambda x.P)\, F_\infty(Q)\, \vec{R} & \text{if } x \notin \mathrm{FV}(P) \text{ and } Q \notin \mathrm{NF}_\beta. \end{array}$$

The following theorem has been folklore for some time. De Vrijer [136, 138] uses F_∞ to calculate the maximal length of a reduction path of a simply typed λ-term. In fact, the proof of [136, Thm 4.9] shows that F_∞ is maximal—see also [136, 2.3.3 and 4.9.2], and the discussion of related work in Section 5. Later, the theorem was proved independently by Regnier [108], Khasidashvili [71], van Raamsdonk and Severi [106] and the author [119]. The proof below is a simplification of the two latter proofs.

1.3.22. Theorem. *F_∞ is an effective, maximal β-reduction strategy.*

Proof. It is clear that F_∞ is an effective β-reduction strategy. To prove maximality we use the formulation from Lemma 1.2.14. Given $M \in \Lambda_K$ and $m \geq 1$, we must show that $M \in m_\beta \Rightarrow F_\infty(M) \in (m-1)_\beta$. We proceed by induction on M.

1. $M \equiv x\,\vec{P}\,Q\,\vec{R}$ where $\vec{P} \in \mathrm{NF}_\beta$, $Q \notin \mathrm{NF}_\beta$. Let $\vec{R} = R_1, \ldots, R_n$. Then $Q \in m^0_\beta, R_1 \in m^1_\beta, \ldots, R_n \in m^n_\beta$, where $m = m^0 + m^1 + \ldots + m^n$, and $m^0 \geq 1$. By the induction hypothesis, $F_\infty(Q) \in (m^0 - 1)_\beta$. Then $F_\infty(M) = x\,\vec{P}\,F_\infty(Q)\,\vec{R} \in (m-1)_\beta$.
2. $M \equiv \lambda x.P$. Similar to Case 1.
3. $M \equiv (\lambda x.P)\,Q\,\vec{R}$ where $x \in \mathrm{FV}(P)$ or $Q \in \mathrm{NF}_\beta$. By the fundamental lemma of maximality,
$$l_\beta(P\{x := Q\}\,\vec{R}) + 1 = l_\beta(M) \geq m.$$
Therefore, $l_\beta(P\{x := Q\}\,\vec{R}) \geq m - 1$, i.e., $F_\infty(M) \in (m-1)_\beta$.
4. $M \equiv (\lambda x.P)\,Q\,\vec{R}$ where $x \notin \mathrm{FV}(P)$ and $Q \notin \mathrm{NF}_\beta$. By the fundamental lemma of maximality,
$$l_\beta(P\,\vec{R}) + l_\beta(Q) + 1 = l_\beta(M) \geq m.$$
We consider two cases.
   - 4.1. $Q \in \infty_\beta$. Then, for any $n \geq 1$, $Q \in n_\beta$. By the induction hypothesis, for any $n \geq 1$, $F_\infty(Q) \in (n-1)_\beta$. In particular, $F_\infty(Q) \in (m-1)_\beta$, and then $F_\infty(M) \in (m-1)_\beta$.
   - 4.2. $Q \notin \infty_\beta$. Then $l_\beta(Q) < \infty$ by Lemma 1.2.7. By the induction hypothesis, $l_\beta(F_\infty(Q)) \geq l_\beta(Q) - 1$. Then
$$\begin{array}{rcl} l_\beta(F_\infty(M)) & = & l_\beta((\lambda x.P)\,F_\infty(Q)\,\vec{R}) \\ & = & l_\beta(P\,\vec{R}) + l_\beta(F_\infty(Q)) + 1 \\ & \geq & l_\beta(P\,\vec{R}) + l_\beta(Q) \\ & = & l_\beta(M) - 1 \\ & \geq & m - 1. \end{array}$$
Thus, $F_\infty(M) \in (m-1)_\beta$. □

1.3.23. Corollary (Barendregt et al. [3, 5]). *F_∞ is perpetual.*

1.3.24. Remark. As pointed out by van Raamsdonk and Severi [106], the proof in [3, 5] of this corollary can be simplified by using the fundamental lemma of perpetuality or one of the related characterizations.

Khasidashvili [71] studies so-called *limit* reduction strategies in orthogonal expression reduction systems (of which β-reduction on Λ_K is a special case), and shows that any limit reduction strategy is maximal and that F_∞ is a limit reduction strategy in λ-calculus. the author [119] presents a $\beta\eta$-reduction strategy H_∞ and shows that it is $\beta\eta$-maximal and thereby $\beta\eta$-perpetual.

1.3.6. On upper bounds for length of reductions

One can effectively compute upper bounds for the length of longest developments and longest reduction paths in several typed λ-calculi (see Sections 1.5 and 1.6). This raises the question whether there is some formula for upper bounds for lengths of reduction paths in type-free λ-calculus. In this subsection we give a positive and a negative answer to this question.

The following definition gives the most obvious way of counting the number of steps in a longest reduction to normal form.

1.3.25. Definition. Define $h : \mathrm{SN}_\beta \to \mathbb{N}$ by:

$$\begin{array}{lcll} h(x\,P_1 \ldots P_n) & = & h(P_1) + \ldots + h(P_n) & \\ h(\lambda x.P) & = & h(P) & \\ h((\lambda x.P)\,Q\,\vec{R}) & = & h(P\{x := Q\}\,\vec{R}) + 1 & \text{if } x \in \mathrm{FV}(P) \text{ or } Q \in \mathrm{NF}_\beta \\ h((\lambda x.P)\,Q\,\vec{R}) & = & h(P\,\vec{R}) + h(Q) + 1 & \text{if } x \notin \mathrm{FV}(P) \text{ and } Q \notin \mathrm{NF}_\beta. \end{array}$$

1.3.26. Proposition. *For any* $M \in \mathrm{SN}_\beta$:$h(M) = l_\beta(M)$.

Proof. By induction on $l_\beta(M)$ using the fundamental lemma of maximality. □

The map h is defined only for elements in SN_β. It is natural to ask whether there is a "simple formula" f such that $f(M)$ is the length of a longest β-reduction from M when $M \in \mathrm{SN}_\beta$, and $f(M)$ is some unpredictable number when $M \in \infty_\beta$. One could hope that the freedom to return arbitrary values on terms with infinite reductions could give a simple formula on strongly normalizing terms. A reasonable formalization of "simple formula" is the notion of a primitive recursive function. The following proposition, which answers a more general question, shows that our hopes are in vain.

1.3.27. Proposition. *There is no total effective* $l : \Lambda_K \to \mathbb{N}$ *such that, for all* $M \in \mathrm{SN}_\beta$,

$$l(M) \geq l_\beta(M).$$

Proof. Suppose such an l existed and consider $c : \Lambda_K \to \mathbb{N}$:

$$c(M) = \begin{cases} 0 & \text{if } F_\infty^{l(M)}(M) \in \mathrm{NF}_\beta. \\ 1 & \text{if } F_\infty^{l(M)}(M) \notin \mathrm{NF}_\beta. \end{cases}$$

Here c is total and effective. Consider the following two cases.

1. $c(M) = 0$. Then $F_\infty^{l(M)}(M) \in \mathrm{NF}_\beta$, i.e., $L_{F_\infty}(M) \leq l(M) < \infty$, so $M \in \mathrm{SN}_\beta$ by perpetuality of F_∞.

2. $c(M) = 1$. Then $F_\infty^{l(M)}(M) \notin \mathrm{NF}_\beta$. By maximality of F_∞ it follows that $l_\beta(M) = L_{F_\infty}(M) > l(M)$. By definition of l, $M \notin \mathrm{SN}_\beta$.

Thus, c gives a procedure to decide for any M whether $M \in \mathrm{SN}_\beta$, which is known to be impossible, a contradiction. □

1.4. The Ω-theorem

In the type-free λ-calculus some terms have an infinite reduction path. The simplest example is the term $\Omega \equiv \omega\,\omega$, where $\omega \equiv \lambda x.x\,x$. It has an infinite reduction path where the term reduces to itself in every step:

$$\Omega \to_\beta \Omega \to_\beta \dots .$$

There are terms that do have an infinite reduction path, but where the path does not have this simple form.[2] For instance, the term $\Psi \equiv \psi\,\psi$, where $\psi \equiv \lambda x.x\,x\,y$, has the infinite reduction path:

$$\Psi \to_\beta \Psi\,y \to_\beta \Psi\,y\,y \to_\beta \dots .$$

In every step the redex Ψ appears as a subterm, and the context of the redex is extended with an application $\bullet y$. As a more complicated example consider the term $\upsilon\,y\,\upsilon$, where $\upsilon \equiv \lambda a.\lambda x.x\,(a\,y)\,x$. It has the infinite reduction path

$$\upsilon y \upsilon \to_\beta (\lambda x.x(yy)x)\upsilon \to_\beta \upsilon(yy)\upsilon \to_\beta (\lambda x.x(yyy)x)\upsilon \to_\beta \upsilon(yyy)\upsilon \to_\beta \dots .$$

This path is similar to the preceding one, but the extra application $\bullet\,y$ is added *inside* the redex.

Although these three reduction paths have their differences they have a common property: in all three paths every term has Ω as a substring. It is natural to ask whether this property is shared by all infinite reduction paths. In this section we present the *Ω-theorem,* taken from [120], which states that this is indeed the case. The proof exploits perpetuality of the strategy F_2 from Section 1.3.3.

The first subsection introduces the set of all terms that do not have Ω as substring, and the second subsection shows that the elements of this set are strongly normalizing. The third subsection studies applications.

1.4.1. The set Λ_Ω

We first formalize what it means that one term is a substring of another.

[2] Lercher [82] shows that $M \to_\beta M$ iff $M \equiv C[\Omega]$ for some context C.

1.4.1. Definition. Define the relation $\trianglelefteq$ ("substring") on Λ_K by:

$$\begin{array}{lcll}
x \trianglelefteq x & & & \\
P \trianglelefteq Q & \Rightarrow & P \trianglelefteq \lambda x.Q & \text{if } x \notin \mathrm{FV}(P) \\
P \trianglelefteq Q & \Rightarrow & P \trianglelefteq Q\,Z & \\
P \trianglelefteq Q & \Rightarrow & P \trianglelefteq Z\,Q & \\
P \trianglelefteq Q & \Rightarrow & \lambda x.P \trianglelefteq \lambda x.Q & \\
P_1 \trianglelefteq Q_1 \;\&\; P_2 \trianglelefteq Q_2 & \Rightarrow & P_1\,P_2 \trianglelefteq Q_1\,Q_2. &
\end{array}$$

1.4.2. Example.

(i) $\omega \trianglelefteq \lambda x.x\,x\,Z$.

(ii) $\Omega \trianglelefteq (\lambda x.x\,x\,Z)\,(\lambda x.x\,x\,Z)$.

(iii) $\omega \trianglelefteq \lambda a.\lambda x.x\,Z\,x$.

(iv) $\Omega \trianglelefteq (\lambda a.\lambda x.x\,Z\,x)\,Z\,(\lambda a.\lambda x.x\,Z\,x)$.

(v) $\omega \trianglelefteq \lambda x.x\,(\lambda y.x)$.

(vi) $\lambda x.x\,y \not\trianglelefteq (\lambda x.x)\,y$.

(vii) $\lambda x.y \not\trianglelefteq (\lambda x.x)\,y$.

(viii) $\lambda x.x\,y \not\trianglelefteq (\lambda x.x)\,y$.

(ix) $\omega \not\trianglelefteq \lambda x.x\,(\lambda x.x)$.

(x) $\Omega \not\trianglelefteq \lambda x.(x\,x)\,\omega$.

It is convenient to introduce an inductively defined set Λ_Ω of all terms that do not contain Ω as a substring, and show that all elements of this set are strongly normalizing. The following auxiliary set Λ_ω, studied by Komori [79], Hindley [49], and Jacobs [56], is the set of all terms that do not contain ω as a substring.

1.4.3. Definition.

(i) Define Λ_ω by:

$$\begin{array}{lcl}
x \in \Lambda_\omega & & \\
P \in \Lambda_\omega, \|P\|_x \leq 1 & \Rightarrow & \lambda x.P \in \Lambda_\omega \\
P, Q \in \Lambda_\omega & \Rightarrow & P\,Q \in \Lambda_\omega.
\end{array}$$

(ii) Define, for $M \in \Lambda_K$, $\|M\|_\omega \in \mathbb{N}$ by:

$$\begin{array}{lcl}
\|x\|_\omega & = & 0 \\
\|\lambda x.P\|_\omega & = & \left\{\begin{array}{ll} \|P\|_\omega & \text{if } \|P\|_x \leq 1 \\ 1 + \|P\|_\omega & \text{if } \|P\|_x > 1 \end{array}\right. \\
\|P\,Q\|_\omega & = & \|P\|_\omega + \|Q\|_\omega.
\end{array}$$

(iii) An abstraction $\lambda x.P$ is *duplicating* if $\|P\|_x > 1$.

1.4.4. Remark. The following equivalences are easily established:

$$\|M\|_\omega = 0 \Leftrightarrow M \in \Lambda_\omega \Leftrightarrow \omega \not\trianglelefteq M.$$

Each of these equivalent conditions state that M does not contain a subterm which is a duplicating abstraction.

One easily shows that Λ_ω is closed under reduction. The intuition is that if $M \in \Lambda_\omega$ and $N \notin \Lambda_\omega$, then M has no duplicating abstractions while N has at least one. Thus, the reduction $M \to_\beta N$ must duplicate a variable in the body of some abstraction, but this would require a duplicating abstraction in M. It is also easy to prove that reduction in Λ_ω decreases term size, since every step removes an application and an abstraction. With the preceding property this implies that every term in Λ_ω is strongly normalizing.

1.4.5. Definition. Define the set Λ_Ω as follows.

$$\begin{array}{lll}
(1) & x \in \Lambda_\Omega & \\
(2) & M \in \Lambda_\Omega & \Rightarrow \quad \lambda x.M \in \Lambda_\Omega \\
(3) & M \in \Lambda_\Omega, N \in \Lambda_\omega & \Rightarrow \quad M\,N \in \Lambda_\Omega \\
(4) & M \in \Lambda_\omega, N \in \Lambda_\Omega & \Rightarrow \quad M\,N \in \Lambda_\Omega.
\end{array}$$

1.4.6. Remark. It is easy to show $\Lambda_\omega \subseteq \Lambda_\Omega$ and the following equivalence:

$$M \in \Lambda_\Omega \Leftrightarrow \Omega \not\trianglelefteq M.$$

Informally, these two equivalent conditions state that M does not contain two disjoint subterms that are both duplicating abstractions.

Next we show that Λ_Ω is closed under reduction. The intuition is as follows. If $M \in \Lambda_\Omega$ and $N \notin \Lambda_\Omega$, then M has no disjoint duplicating abstractions, while N has at least two. If $M \to_\beta N$ then non-disjoint duplicating abstractions in M are also non-disjoint in N. Therefore, the two disjoint duplicating abstractions in N must arise from M either by duplication into disjoint positions of a single duplicating abstraction, or by duplication of a variable in the body of a non-duplicating abstraction which is disjoint with a duplicating abstraction. Both cases are impossible because they entail that M has two disjoint duplicating abstractions.

1.4.7. Lemma. $M \in \Lambda_\Omega \;\&\; M \to_\beta N \Rightarrow N \in \Lambda_\Omega$.

Proof. First prove by induction on the derivation of $M \in \Lambda_\omega$ that

$$M \in \Lambda_\omega \;\&\; N \in \Lambda_\omega \Rightarrow M\{x := N\} \in \Lambda_\omega \tag{1.1}$$

and

$$M \in \Lambda_\omega \;\&\; \|M\|_x \leq 1 \;\&\; N \in \Lambda_\Omega \Rightarrow M\{x := N\} \in \Lambda_\Omega. \tag{1.2}$$

Show by induction on the derivation of $M \in \Lambda_\Omega$, using (1.1) and $\Lambda_\omega \subseteq \Lambda_\Omega$,

$$M \in \Lambda_\Omega, N \in \Lambda_\omega \Rightarrow M\{x := N\} \in \Lambda_\Omega. \tag{1.3}$$

Now proceed by induction on the derivation of $M \to_\beta N$ using (1.2-1.3). $\square$

1.4.2. Strong normalization of terms in Λ_Ω

As for Λ_ω, the idea for proving that all terms in Λ_Ω are strongly normalizing is to find a decreasing measure, but term size $\| \bullet \|$ does not work. Instead we consider the lexicographically ordered measure $\langle \| \bullet \|_\omega, \| \bullet \| \rangle$.

Suppose $M \to_\beta N$ by contraction of the redex $\Delta \equiv (\lambda x.P)\, Q$. If $\lambda x.P$ is non-duplicating, contraction of Δ creates no new duplicating abstractions. Moreover, the size of N is strictly smaller than the size of M, so the reduction step decreases the measure.

If $\lambda x.P$ is duplicating, the reduction step removes one duplicating abstraction, and any new duplicating abstractions have to come either from proliferation of duplicating abstractions in Q or from duplication of variables in the body of some abstraction. The first case is impossible, since it implies that M has two disjoint duplicating abstractions. In the second case, new duplicating abstractions *may* be created, but they must have their λ to the left of Δ.

Recall that a *standard* reduction path $M_0 \to_\beta M_1 \to_\beta \ldots$ is such that whenever a redex Δ is contracted in M_i all abstractions to the left of Δ are marked, and a redex with marked abstraction is not allowed to be contracted in M_j for any $j > i$. If a term has an infinite reduction path, then it has a standard infinite reduction path [18].

The idea then is as follows. Suppose some $M \in \Lambda_\Omega$ has an infinite reduction path and hence a standard infinite reduction path. Then the measure $\langle \|\bullet\|_\omega, \|\bullet\| \rangle$ is decreasing on this reduction path if we insist that $\|\bullet\|_\omega$ count only non-marked abstractions, and thus we arrive at a contradiction.

To formalize this reasoning we use the strategy F_2 from Section 1.3.3, which computes standard infinite reductions. The following map V isolates the part of a term in which F_2 contracts a redex. This part of the term contains all the abstractions to be counted by our measure.

1.4.8. Definition. Define $V : \infty_\beta \to \Lambda_K$ by:

$$\begin{array}{lcll}
V(x\,\vec{P}\,Q\,\vec{R}) & = & V(Q) & \text{if } \vec{P} \in \mathrm{SN}_\beta, Q \notin \mathrm{SN}_\beta \\
V(\lambda x.P) & = & V(P) & \\
V((\lambda x.P)\,Q\,\vec{R}) & = & (\lambda x.P)\,Q\vec{R} & \text{if } P \in \mathrm{SN}_\beta, Q \in \mathrm{SN}_\beta \\
V((\lambda x.P)\,Q\,\vec{R}) & = & V(P) & \text{if } P \notin \mathrm{SN}_\beta \\
V((\lambda x.P)\,Q\,\vec{R}) & = & V(Q) & \text{if } P \in \mathrm{SN}_\beta, Q \notin \mathrm{SN}_\beta.
\end{array}$$

1.4.9. Lemma. *For all $M \in \infty_\beta$: $V(M) \subseteq M$.*

Proof. By induction on M. □

1.4.10. Lemma. *For all $M \in \infty_\beta$,*

$$V(M) = (\lambda y.K)\, L\, \vec{N}$$

for some $K, L, \vec{N} \in \Lambda_K$ *with*

$$V(F_2(M)) \subseteq K\{x := L\}\, \vec{N}.$$

PROOF. Induction on M using perpetuality of F_2.

1. $M \equiv x\, \vec{P}\, Q \vec{R}$ where $\vec{P} \in \mathrm{SN}_\beta$, $Q \notin \mathrm{SN}_\beta$. By the induction hypothesis,

$$V(M) = V(Q) = (\lambda y.K)\, L\, \vec{N}$$

for some $K, L, \vec{N}$. By the induction hypothesis and perpetuality of F_2,

$$V(F_2(M)) = V(x\, \vec{P}\, F_2(Q)\, \vec{R}) = V(F_2(Q)) \subseteq K\{y := L\}\, \vec{N}.$$

2. $M \equiv \lambda x.P$. Similar to Case 1.
3. $M \equiv (\lambda y.P)\, Q\, \vec{R}$ where $P \in \mathrm{SN}_\beta$ and $Q \in \mathrm{SN}_\beta$. Then

$$V(M) = (\lambda x.P)\, Q\, \vec{R},$$

and by Lemma 1.4.9,

$$V(F_2(M)) = V(P\{x := Q\}\, \vec{R}) \subseteq P\{x := Q\}\, \vec{R}.$$

The remaining two cases are similar to Case 1. □

1.4.11. LEMMA. $\|P\{x := Q\}\|_\omega = \|P\|_\omega + \|P\|_x \cdot \|Q\|_\omega$.

PROOF. By induction on P. □

1.4.12. PROPOSITION. $M \in \Lambda_\Omega \;\Rightarrow\; M \in \mathrm{SN}_\beta$.

PROOF. Suppose $M \in \Lambda_\Omega$ and $M \in \infty_\beta$. By perpetuality of F_2, there is an infinite reduction path

$$M_0 \to_\beta M_1 \to_\beta \dots$$

such that for all i, $F_2(M_i) = M_{i+1}$ and, by Lemma 1.4.7, $M_i \in \Lambda_\Omega$. We now claim that for all i

$$\langle \|V(M_i)\|_\omega, \|V(M_i)\| \rangle > \langle \|V(M_{i+1})\|_\omega, \|V(M_{i+1})\| \rangle. \tag{1.4}$$

This implies that we have an infinite sequence

$$\langle \|V(M_0)\|_\omega, \|V(M_0)\| \rangle > \langle \|V(M_1)\|_\omega, \|V(M_1)\| \rangle > \dots ,$$

which is clearly a contradiction. Thus $M \in \mathrm{SN}_\beta$, provided we can prove (1.4).

To prove this, first note that by Lemma 1.4.10 and 1.4.9:

$$\begin{aligned} V(M_i) &= (\lambda y.K)\, L\, \vec{N} \quad \subseteq M_i && (1.5)\\ V(M_{i+1}) &\subseteq K\{y := L\}\, \vec{N} && (1.6) \end{aligned}$$

for some $K, L, \vec{N}$. Since $(\lambda y.K)\, L \subseteq M_i \in \Lambda_\Omega$, also $(\lambda y.K)\, L \in \Lambda_\Omega$. We now prove (1.4) splitting into the following two cases. Let $\vec{N} = N_1, \dots, N_n$.

1. $\|K\|_y > 1$. Then $\lambda y.K \in \Lambda_\Omega \backslash \Lambda_\omega$, so $L \in \Lambda_\omega$, and hence $\|L\|_\omega = 0$. By (1.6), Lemma 1.4.11, and (1.5):

$$\begin{array}{rcl} \|V(M_{i+1})\|_\omega & \leq & \|K\{y := L\}\,\vec{N}\|_\omega \\ & = & \|K\|_\omega + \|K\|_y \cdot \|L\|_\omega + \|N_1\|_\omega + \ldots + \|N_n\|_\omega \\ & = & \|K\|_\omega + \|N_1\|_\omega + \ldots + \|N_n\|_\omega \\ & < & \|\lambda y.K\|_\omega + \|N_1\|_\omega + \ldots + \|N_n\|_\omega \\ & = & \|(\lambda y.K)\; L\; \vec{N}\|_\omega \\ & = & \|V(M_i)\|_\omega . \end{array}$$

2. $\|K\|_y \leq 1$. Then by (1.6), Lemma 1.4.11, and (1.5):

$$\begin{array}{rcl} \|V(M_{i+1})\|_\omega & \leq & \|K\{y := L\}\,\vec{N}\|_\omega \\ & = & \|K\|_\omega + \|K\|_y \cdot \|L\|_\omega + \|N_1\|_\omega + \ldots + \|N_n\|_\omega \\ & \leq & \|K\|_\omega + \|L\|_\omega + \|N_1\|_\omega + \ldots + \|N_n\|_\omega \\ & = & \|(\lambda y.K)\; L\; \vec{N}\|_\omega \\ & = & \|V(M_i)\|_\omega . \end{array}$$

   Moreover, by (1.6) and (1.5):

$$\begin{array}{rcl} \|V(M_{i+1})\| & \leq & \|K\{y := L\}\,\vec{N}\| \\ & = & \|K\| + \|K\|_y \cdot (\|L\| - 1) + \|N_1\| + \ldots + \|N_n\| + n \\ & < & \|K\| + \|L\| + 2 + \|N_1\| + \ldots + \|N_n\| + n \\ & = & \|(\lambda y.K)\; L\; \vec{N}\| \\ & = & \|V(M_i)\| . \end{array}$$

   as required. □

We finally have the Ω-theorem, from [120]:

1.4.13. Theorem. *If $M \in \infty_\beta$ then $\Omega \trianglelefteq M$.*

Proof. By Remark 1.4.6 and Proposition 1.4.12. □

1.4.14. Remark. The term $M \equiv (\lambda x.y\,x\,x)\,(\lambda x.y\,x\,x)$ shows that $\Omega \trianglelefteq M$ does not generally imply $M \in \infty_\beta$. This should come as no surprise: if $\Omega \trianglelefteq M$ had been equivalent to $M \in \infty_\beta$, we would have had a simple syntactic (in particular effective) algorithm for deciding whether $M \in \mathrm{SN}_\beta$, which is an undecidable problem.

Following Gramlich [43] (see also Plaisted [99]) we call an infinite reduction path *constricting* if it has the form

$$C_1[M_1] \to_\beta C_1[C_2[M_2]] \to_\beta C_1[C_2[C_3[M_3]]] \ldots ,$$

where M_i is the minimal superterm with an infinite reduction path of the redex contracted in the step $C_1[\ldots C_i[M_i]\ldots] \to_\beta C_1[\ldots C_i[C_{i+1}[M_{i+1}]]\ldots]$.

Van Oostrom [95] sketches a variant of the above proof which, instead of using the perpetual strategy F_2 to obtain standard infinite reductions, uses a so-called *zoom-in* strategy (see Melliès [86]). This is a constricting strategy which in each term contracts the leftmost redex of a minimal subterm with an infinite reduction path. The proof presented above is very similar, since F_2 is also constricting—indeed, Lemma 1.4.10 expresses a very similar property. However, in $(\lambda x.x)\, z\, \Omega$, F_2 contracts the left-most redex, so F_2 is not a zoom-in strategy in the above sense. The following variation F_3, studied by the author [120], is a zoom-in strategy:

1.4.15. Definition. Define $F_3 : \infty_\beta \to \Lambda_K$ by:

$$\begin{array}{lcll}
F_3(x\,\vec{P}\,Q\,\vec{R}) & = & x\,\vec{P}\,F_3(Q)\,\vec{R} & \text{if } \vec{P} \in \mathrm{SN}_\beta, Q \notin \mathrm{SN}_\beta \\
F_3(\lambda x.P) & = & \lambda x.F_3(P) & \\
F_3((\lambda x.P)\,Q\,\vec{R}) & = & P\{x := Q\}\,\vec{R} & \text{if } P, Q, \vec{R} \in \mathrm{SN}_\beta \\
F_3((\lambda x.P)\,Q\,\vec{R}) & = & (\lambda x.F_3(P))\,Q\,\vec{R} & \text{if } P \notin \mathrm{SN}_\beta \\
F_3((\lambda x.P)\,\vec{R}\,Q\,\vec{S}) & = & (\lambda x.P)\,\vec{R}\;F_3(Q)\,\vec{S} & \text{if } P, \vec{R} \in \mathrm{SN}_\beta, Q \notin \mathrm{SN}_\beta.
\end{array}$$

Khasidashvili and Ogawa [74] study strategies which in a term contract a so-called *external* redex of a minimal subterm of M with an infinite reduction path; in particular, in λ-terms the leftmost redex of a minimal subterm with an infinite reduction is external. They show that any such strategy is perpetual. They also show that the strategy which in each step contracts the leftmost among all such redexes is constricting.

Xi [142] calls a reduction path $M_0 \to_\beta M_1 \to_\beta \ldots$ *canonical* if, whenever a redex Δ is contracted in M_i all redexes containing Δ as a subterm have their abstractions marked, and a redex with marked abstraction is not allowed to be contracted in M_j for any $j > i$. Any standard reduction is also canonical, but the converse is not true, since a canonical path may contract disjoint redexes from right to left. However, whenever a term M has a canonical reduction which is infinite (or ends in N) then M also has a standard reduction which is infinite (or ends in N). Xi uses canonical reductions to give proofs of the finite developments theorem, the standardization theorem, the conservation theorem for Λ_I, and the normalization theorem.

Böhm et al. [21] and Böhm and Dezani-Ciancaglini [22] give, for any β-normal form M, a constructive definition of a set of β-normal forms N for which $M\,N$ has a β-normal form. Since any λ-term can be transformed to an equivalent term which is an applicative combination of β-normal forms, this can be used to generally approximate whether a term has a β-normal form or not. On the other hand, Λ_Ω directly characterizes a class of terms with arbitrary nesting of λ's and application which are all β-strongly normalizing.

1.4.3. Applications

An *S-term* in combinatory logic is a term built of only the S-combinator and application, e.g., $S(SS)SSSS$ and $SSS(SS)SS$. Barendregt et al. [5] show that these two S-terms have infinite reduction paths. Duboué has verified by computer that the remaining 130 other S-terms with 7 or fewer occurrences of S are strongly normalizing. The following shows that only one among the 2622 closed λ-terms of size 9 or less has an infinite reduction path.[3]

1.4.16. Corollary. *Let $M \in \infty_\beta$. Then*

(i) $\|\Omega\| \leq \|M\|$.

(ii) $\|M\| \leq \|\Omega\| \;\Rightarrow\; M \equiv \Omega$.

Proof. (i): By the Ω-theorem, since $O \trianglelefteq M$ clearly implies $\|O\| \leq \|M\|$. (ii): By the Ω-theorem and (i) using the fact that $O \trianglelefteq M$ and $\|M\| \leq \|O\|$ implies $M \equiv O$. □

The next application gives a technique to reduce proofs that some term is strongly normalizing to proofs that terms are weakly normalizing. The latter is usually easier.

1.4.17. Corollary. *If $N \in \mathrm{WN}_\beta$ for all $N \trianglelefteq M$, then $M \in \mathrm{SN}_\beta$.*

Proof. If $M \in \infty_\beta$ then, by the Ω-theorem, $\Omega \trianglelefteq M$, and $\Omega \notin \mathrm{WN}_\beta$. □

The following shows how this corollary may be used to prove strong normalization of a set of terms.

1.4.18. Proposition. *Let $S \subseteq \Lambda_K$ and let $\preceq$ be a relation on Λ_K with*

(i) *If $N \in \mathrm{WN}_\beta$ for all $N \preceq M$, then $M \in \mathrm{SN}_\beta$.*

(ii) *If $M \in S$ and $N \preceq M$ then $N \in S$.*

Then $S \subseteq \mathrm{WN}_\beta \;\Rightarrow\; S \subseteq \mathrm{SN}_\beta$.

Proof. Assume that $\preceq$ satisfies (i)-(ii) and assume $S \subseteq \mathrm{WN}_\beta$. Given an $M \in S$. By (ii), $N \in S$ for all $N \preceq M$. Then, by assumption, $N \in \mathrm{WN}_\beta$ for all $N \preceq M$. Then by (i), $M \in \mathrm{SN}_\beta$, as required. □

1.4.19. Remark. The previous result has motivated the search for relations satisfying (i)-(ii) for various sets S, notably the set $\Lambda^{\rightarrow}$ of terms typable in simply typed λ-calculus à la Curry (see Section 1.5). With such a relation at hand, one can show that all elements of $\Lambda^{\rightarrow}$ are strongly normalizing by demonstrating that they are all weakly normalizing.

[3] T. Mogensen gives a formula $f(n,m)$ for the number of λ-terms of size $n \geq 1$ with at most $m \geq 0$ free variables:

$$\begin{array}{lcl} f(1,m) & = & m \\ f(n+1,m) & = & f(n,m+1) + \sum_{i=1}^{n-1} f(i,m) \cdot f(n-i,m). \end{array}$$

As Corollary 1.4.17 shows, $\trianglelefteq$ satisfies (i). In fact, the proof of the corollary shows that any relation $\preceq$ satisfying $M \in \infty_\beta \Rightarrow \Omega \preceq M$ also satisfies (i). However, $\trianglelefteq$ does not satisfy (ii) for $\Lambda^{\rightarrow}$. For instance, $\lambda x.x\,(x\,\lambda y.y)$ has type $((\alpha \to \alpha) \to (\alpha \to \alpha)) \to (\alpha \to \alpha)$ in simply typed λ-calculus, but $\lambda x.x\,x \trianglelefteq \lambda x.x\,(x\,\lambda y.y)$ has no type.

The author [121] and Xi [141, 144] study relations $\preceq$ satisfying (i) and (ii) for $\Lambda^{\rightarrow}$ which are defined by translations, i.e., $M \preceq N$ iff $t(N) = M$ for certain translations $t : \Lambda_K \to \Lambda_I$—see Chapter 2.

1.4.20. Problem. Hindley [49] shows that $M \in \Lambda_\omega \Rightarrow M \in \Lambda^{\rightarrow}$, i.e., every $M \in \Lambda_\omega$ can be typed in simply typed λ-calculus à la Curry. Can every M in Λ_Ω be typed in second-order typed λ-calculus à la Curry?

1.5. Strong normalization in type theory

As mentioned in Section 1.3, many strong normalization proofs in the literature make use of the fundamental lemma of perpetuality or the fundamental lemma of maximality (see also Remark 1.3.6 and 1.3.10). In this section we study such proofs in more detail in the context of the simply typed λ-calculus.

The first subsection presents the version of simply typed λ-calculus with which we shall be concerned. The second subsection presents a new proof of strong normalization of simply typed λ-calculus due to van Raamsdonk and Severi [106]. While their original proof uses their characterization of SN_β, the present version uses the fundamental lemma of perpetuality. Other proofs are reviewed in less detail.

1.5.1. Simply typed λ-calculus

1.5.1. Definition. Let T_0 be a set of constants, called *base types*. The set T of *simple types* is the smallest set such that

(i) $T_0 \subseteq T$.

(ii) $A, B \in T \Rightarrow A \to B \in T$.

For $A \in T$, $\|A\|$ denotes the number of arrows in A.

We use association to the right, so $A \to B \to C$ means $A \to (B \to C)$.

1.5.2. Convention. It is convenient to assume that the set V (the set of variables of Λ_K) is divided into mutually exclusive and together exhaustive non-empty classes V_A where $A \in T$, i.e.,

$$V = \bigcup_{A \in T} V_A \qquad \& \qquad A \neq B \Rightarrow V_A \neq V_B \qquad \& \qquad V_A \neq \emptyset.$$

1.5.3. Definition. For every $A \in T$, the set of *simply typed λ-terms of type A*, written $\Lambda^{\rightarrow}_A$, is the smallest set such that

(i) $x \in V_A \;\Rightarrow\; x \in \Lambda^{\rightarrow}_A$.

(ii) $x \in V_A \;\&\; M \in \Lambda^{\rightarrow}_B \;\Rightarrow\; \lambda x.M \in \Lambda^{\rightarrow}_{A\rightarrow B}$.

(iii) $M \in \Lambda^{\rightarrow}_{B\rightarrow A} \;\&\; N \in \Lambda^{\rightarrow}_B \;\Rightarrow\; M\,N \in \Lambda^{\rightarrow}_A$.

The set of *simply typed λ-terms,* written $\Lambda^{\rightarrow}$, is defined by:

$$\Lambda^{\rightarrow} = \bigcup_{A\in T} \Lambda^{\rightarrow}_A.$$

The following two properties, known as the *substitution lemma* and the *uniqueness of types property,* will be used in the next subsection.

1.5.4. Lemma.

(i) $P \in \Lambda^{\rightarrow}_B \;\&\; x \in \Lambda^{\rightarrow}_A \;\&\; N \in \Lambda^{\rightarrow}_A \;\Rightarrow\; P\{x := N\} \in \Lambda^{\rightarrow}_B$.

(ii) $P \in \Lambda^{\rightarrow}_A \;\&\; P \in \Lambda^{\rightarrow}_B \;\Rightarrow\; A = B$.

Proof. (i)-(ii): by induction on the derivation of $P \in \Lambda^{\rightarrow}_B$. □

1.5.2. Strong normalization of simply typed λ-calculus

An attempt to prove directly, by induction on the derivation of $M \in \Lambda^{\rightarrow}_A$, that $M \in \mathrm{SN}_\beta$ breaks down in the application case: $P \in \mathrm{SN}_\beta$ and $Q \in \mathrm{SN}_\beta$ does not imply $P\,Q \in \mathrm{SN}_\beta$. One way of overcoming this difficulty is to introduce the set $\mathrm{SN}^{\rightarrow}_A$ of strongly normalizing terms of type A and show that $M \in \Lambda^{\rightarrow}_A$ implies $M \in \mathrm{SN}^{\rightarrow}_A$. The crucial step then is to show for any $M \in \mathrm{SN}^{\rightarrow}_{A\rightarrow B}$ and $N \in \mathrm{SN}^{\rightarrow}_A$ that $M\,N \in \mathrm{SN}^{\rightarrow}_B$. This idea is carried out below, following [106].

1.5.5. Definition. For $A \in T$ define $\mathrm{SN}^{\rightarrow}_A = \mathrm{SN}_\beta \cap \Lambda^{\rightarrow}_A$, and

$$\mathrm{SN}^{\rightarrow} = \bigcup_{A\in T} \mathrm{SN}^{\rightarrow}_A.$$

1.5.6. Remark. For every type A: $\emptyset \subsetneq V_A \subseteq \mathrm{SN}^{\rightarrow}_A \subseteq \Lambda^{\rightarrow}_A$.

1.5.7. Definition. For $X, Y \subseteq \Lambda_K$ define

$$X \rightarrow Y = \{M \in \Lambda_K \mid \forall N \in X : M\,N \in Y\}.$$

1.5.8. Lemma. $\Lambda^{\rightarrow}_{A\rightarrow B} = \Lambda^{\rightarrow}_A \rightarrow \Lambda^{\rightarrow}_B$.

PROOF. Let $M \in \Lambda^{\to}_{A\to B}$. For all $N \in \Lambda^{\to}_{A}$, $M\,N \in \Lambda^{\to}_{B}$, so $M \in \Lambda^{\to}_{A} \to \Lambda^{\to}_{B}$. Hence $\Lambda^{\to}_{A\to B} \subseteq \Lambda^{\to}_{A} \to \Lambda^{\to}_{B}$. Conversely, let $M \in \Lambda^{\to}_{A} \to \Lambda^{\to}_{B}$. Pick some $N \in \Lambda^{\to}_{A}$. Then $M\,N \in \Lambda^{\to}_{B}$. Therefore, $M \in \Lambda^{\to}_{C\to B}$ for some $C \in T$ with $N \in \Lambda^{\to}_{C}$. By uniqueness of types, $A = C$, so $M \in \Lambda^{\to}_{A\to B}$. Hence $\Lambda^{\to}_{A} \to \Lambda^{\to}_{B} \subseteq \Lambda^{\to}_{A\to B}$. ☐

1.5.9. LEMMA. $\mathrm{SN}^{\to}_{A\to B} \supseteq \mathrm{SN}^{\to}_{A} \to \mathrm{SN}^{\to}_{B}$.

PROOF. Let $M \in \mathrm{SN}^{\to}_{A} \to \mathrm{SN}^{\to}_{B}$. Pick some $N \in \mathrm{SN}^{\to}_{A}$. Then $M\,N \in \mathrm{SN}^{\to}_{B}$. In particular, $M\,N \in \mathrm{SN}_{\beta}$, and then $M \in \mathrm{SN}_{\beta}$. Moreover, since $M\,N \in \Lambda^{\to}_{B}$ and $N \in \Lambda^{\to}_{A}$, also $M \in \Lambda^{\to}_{A\to B}$ by uniqueness of types. In conclusion, $M \in \mathrm{SN}^{\to}_{A\to B}$. ☐

The converse of the preceding lemma is more difficult to prove. We need the following lemma.

1.5.10. LEMMA. *Let* $P \in \mathrm{SN}^{\to}_{B}$, $x \in \Lambda^{\to}_{A_1\to\ldots\to A_m}$, *and* $N \in \mathrm{SN}^{\to}_{A_1} \to \ldots \to \mathrm{SN}^{\to}_{A_m}$ *where* A_m *is a base type. Then* $P\{x := N\} \in \mathrm{SN}^{\to}_{B}$.

PROOF. We use the abbreviation $L^* \equiv L\{x := N\}$ for any $L \in \Lambda^{\to}$. By Lemma 1.5.9, $N \in \mathrm{SN}^{\to}_{A_1\to\ldots\to A_m}$. By the substitution lemma, $P^* \in \Lambda^{\to}_{B}$. It remains to show $P^* \in \mathrm{SN}_{\beta}$. We show this by induction on lexicographically ordered pairs $\langle l_\beta(P), \|P\|\rangle$.

1. $P \equiv y\,P_1 \ldots P_n$. Then $P_1, \ldots, P_n \in \mathrm{SN}_{\beta}$. Also, $y \in \Lambda^{\to}_{B_1\to\ldots\to B_n\to B}$ and $P_1 \in \Lambda^{\to}_{B_1}, \ldots, P_n \in \Lambda^{\to}_{B_n}$, i.e., $P_1 \in \mathrm{SN}^{\to}_{B_1}, \ldots, P_n \in \mathrm{SN}^{\to}_{B_n}$. By the induction hypothesis, $P_1^*, \ldots, P_n^* \in \mathrm{SN}_{\beta}$. Consider two subcases.
   1.1. $y \not\equiv x$. Then $P^* \equiv y\,P_1^* \ldots P_n^* \in \mathrm{SN}_{\beta}$.
   1.2. $y \equiv x$. Then $B_1 = A_1, \ldots, B_n = A_n$ and $B = A_{n+1} \to \ldots \to A_m$. By Lemma 1.5.9, $\mathrm{SN}^{\to}_{A_{n+1}} \to \ldots \to \mathrm{SN}^{\to}_{A_m} \subseteq \mathrm{SN}^{\to}_{B}$. Therefore, $\mathrm{SN}^{\to}_{A_1} \to \ldots \to \mathrm{SN}^{\to}_{A_m} \subseteq \mathrm{SN}^{\to}_{A_1} \to \ldots \to \mathrm{SN}^{\to}_{A_n} \to \mathrm{SN}^{\to}_{B}$. So $N \in \mathrm{SN}^{\to}_{A_1} \to \ldots \to \mathrm{SN}^{\to}_{A_n} \to \mathrm{SN}^{\to}_{B}$. By the substitution lemma, $P_1^* \in \Lambda^{\to}_{B_1}, \ldots, P_n^* \in \Lambda^{\to}_{B_n}$, i.e., $P_1^* \in \mathrm{SN}^{\to}_{A_1}, \ldots, P_n^* \in \mathrm{SN}^{\to}_{A_n}$. Therefore, $P^* \equiv N P_1^* \ldots P_n^* \in \mathrm{SN}^{\to}_{B}$.
2. $P \equiv \lambda y.P_0$. Then $P_0 \in \mathrm{SN}_{\beta}$. Also, $B = B_1 \to B_0$ and $P_0 \in \Lambda^{\to}_{B_0}$, i.e., $P_0 \in \mathrm{SN}^{\to}_{B_0}$. By the induction hypothesis, $P_0^* \in \mathrm{SN}_{\beta}$. Therefore also $P^* \equiv \lambda y.P_0^* \in \mathrm{SN}_{\beta}$.
3. $P \equiv (\lambda y.P_0)\,P_1 P_2 \ldots P_n$. Then $P_0\{y := P_1\}P_2 \ldots P_n \in \mathrm{SN}_{\beta}$, $P_1 \in \mathrm{SN}_{\beta}$. Also, $P_1 \in \Lambda^{\to}_{B_1}, \ldots, P_n \in \Lambda^{\to}_{B_n}$, $y \in \Lambda^{\to}_{B_1}$, and $P_0 \in \Lambda^{\to}_{B_2\to\ldots\to B_n\to B}$. By the induction hypothesis,

$$(P_0\{y := P_1\}P_2 \ldots P_n)^* \equiv P_0^*\{y := P_1^*\}P_2^* \ldots P_n^* \in \mathrm{SN}_{\beta}$$

   and $P_1^* \in \mathrm{SN}_{\beta}$. Then $P^* \equiv (\lambda y.P_0^*)\,P_1^* P_2^* \ldots P_n^* \in \mathrm{SN}_{\beta}$, by the fundamental lemma of perpetuality. ☐

The following crucial lemma states that $M \in \mathrm{SN}^{\rightarrow}_{A\rightarrow B}$ and $N \in \mathrm{SN}^{\rightarrow}_{A}$ implies $M\,N \in \mathrm{SN}^{\rightarrow}_{B}$.

1.5.11. LEMMA. $\mathrm{SN}^{\rightarrow}_{A\rightarrow B} \subseteq \mathrm{SN}^{\rightarrow}_{A} \rightarrow \mathrm{SN}^{\rightarrow}_{B}$.

PROOF. We prove that $M \in \mathrm{SN}^{\rightarrow}_{A\rightarrow B}$ implies $M \in \mathrm{SN}^{\rightarrow}_{A} \rightarrow \mathrm{SN}^{\rightarrow}_{B}$. The proof is by induction on lexicographically ordered pairs $\langle \|A\|, l_\beta(M)\rangle$. For each $N \in \mathrm{SN}^{\rightarrow}_{A}$ we must prove that $M\,N \in \mathrm{SN}^{\rightarrow}_{B}$. Since obviously $M\,N \in \Lambda^{\rightarrow}_{B}$, it suffices to show in each case that $M\,N \in \mathrm{SN}_\beta$.

1. $M \equiv y\,P_1 \dots P_n$. Then $P_1, \dots, P_n \in \mathrm{SN}_\beta$. Since $N \in \mathrm{SN}_\beta$, it follows that $M\,N \equiv y\,P_1 \dots P_n N \in \mathrm{SN}_\beta$.
2. $M \equiv \lambda x.P$. Then $P \in \mathrm{SN}_\beta$. Since $A = A_1 \rightarrow \dots \rightarrow A_m$ for some base type A_m, the induction hypothesis yields $N \in \mathrm{SN}^{\rightarrow}_{A_1} \rightarrow \dots \rightarrow \mathrm{SN}^{\rightarrow}_{A_m}$. Since $P \in \mathrm{SN}^{\rightarrow}_{B}$, Lemma 1.5.10 implies that $P\{x := N\} \in \mathrm{SN}_\beta$. Then $M\,N \equiv (\lambda x.P)\,N \in \mathrm{SN}_\beta$ by the fundamental lemma of perpetuality.
3. $M \equiv (\lambda y.P_0)\,P_1 P_2 \dots P_n$. Then $P_0\{y := P_1\}P_2 \dots P_n \in \mathrm{SN}_\beta$ and also $P_1 \in \mathrm{SN}_\beta$. Since $P_0\{y := P_1\}P_2 \dots P_n \in \Lambda^{\rightarrow}_{A\rightarrow B}$, the induction hypothesis yields $P_0\{y := P_1\}P_2 \dots P_n N \in \mathrm{SN}_\beta$. Since $P_1 \in \mathrm{SN}_\beta$, also $M\,N \equiv (\lambda y.P_0)\,P_1 P_2 \dots P_n N \in \mathrm{SN}_\beta$ by the fundamental lemma of perpetuality. $\square$

1.5.12. THEOREM. *Let A be a simple type. If $M \in \Lambda^{\rightarrow}_{A}$ then $M \in \mathrm{SN}_\beta$.*

PROOF. By induction on the derivation of $M \in \Lambda^{\rightarrow}_{A}$.

1. $M \equiv x \in V_A$. Then $x \in \mathrm{SN}_\beta$.
2. $M = \lambda x.P$, where $A = A_0 \rightarrow A_1$ and $P \in \Lambda^{\rightarrow}_{A_1}$. By the induction hypothesis, $P \in \mathrm{SN}_\beta$, and therefore $\lambda x.P \in \mathrm{SN}_\beta$.
3. $M \equiv P\,Q$, where $P \in \Lambda^{\rightarrow}_{B\rightarrow A}$ and $Q \in \Lambda^{\rightarrow}_{B}$. By the induction hypothesis, $P \in \mathrm{SN}^{\rightarrow}_{B\rightarrow A}$ and $Q \in \mathrm{SN}^{\rightarrow}_{B}$. By Lemma 1.5.11, $P \in \mathrm{SN}^{\rightarrow}_{B} \rightarrow \mathrm{SN}^{\rightarrow}_{A}$. Then $P\,Q \in \mathrm{SN}^{\rightarrow}_{A} \subseteq \mathrm{SN}_\beta$. $\square$

1.5.13. REMARK. A similar technique for handling the difficult application case is due to Xi [140].

There are many other proofs of strong normalization of simply typed λ-calculus. The following is an incomplete list. Tait [125] proves weak normalization of several systems, but the method can be adapted to prove strong normalization. The resulting classical proof makes use of the notion of strong computability and is quite short but complex. The proof uses the fundamental lemma of perpetuality to show that the set of strongly computable terms is closed under certain expansions—see, e.g., [50, App. 2, Lem. 2] .

Girard [41] introduces the notion of candidate of reducibility. He extends Tait's method in order to prove strong normalization of second- and higher-order λ-calculus. In the version of this proof technique expressed in terms of saturated sets, the fundamental lemma of perpetuality is used to show that SN_β is a saturated set—see, e.g., [4, Lem. 4.3.3].

Terlouw [129] interprets Tait's proof of strong normalization of simply typed lambda calculus in a general model-theoretic framework. This yields a proof of strong normalization of the Calculus of Constructions and other advanced type systems.

Gandy [36] interprets a term in a typed λ-calculus by a strict monotonic functional whose value is an upper bound for the length of reductions from the term—the form of the upper bound is elaborated by Schwichtenberg [114]. Gandy's technique uses implicitly the weak form of the fundamental lemma of maximality (Corollary 1.3.19). The technique is generalized to higher-order rewrite systems by van de Pol [101] and applied to a variety of systems by van de Pol and Schwichtenberg [103]. Van de Pol [102] discusses the relationship between the proof by Gandy and the proof by Tait.

De Vrijer [138, 136] proves strong normalization of simply typed λ-calculus by translating terms into functionals computing the exact length of the longest reduction path to normal form, and shows that F_∞ computes this path. De Vrijer's proof uses the fundamental lemma of maximality—see the proof of [136, Thm. 4.9], and also [136, 2.3.3 and 4.9.2].

Another technique for computing upper bounds on lengths of reductions is due to Howard [53] which is used by Schwichtenberg [115] to give upper bounds for the length of reductions in simply typed λ-calculus. Whereas the bound h from Definition 1.3.25 implicitly reduces the term to normal form, i.e., $h((\lambda x.P)\ Q)$ is expressed in terms of $h(P\{x := Q\})$, the bounds for reductions of simply typed terms can be expressed in such a way that the bound for $(\lambda x.P)\ Q$ is expressed in terms of the bounds for P and Q. This technique uses implicitly a version of the fundamental lemma of maximality—see the proof of the main lemma [115, p.407]. Springintveld [123] applies the technique to the dependent system λP and to the weak version $\lambda\underline{\omega}$ of higher-order typed λ-calculus.

Xi [143] gives a proof of the standardization theorem which provides an upper bound on the length of the standard reduction path obtained from any given reduction path, and Xi uses this to provide upper bounds for the length of reduction paths in simply typed λ-calculus.

Van Daalen proves strong normalization of simply typed λ-calculus using induction on a certain triple—see [93, p.507]. Lévy [83] uses the technique to prove strong normalization of a labeled λ-calculus with a bounded predicate. This proof yields also that all developments are finite, and standardization, as reported in [28].

Capretta and Valentini [23] prove strong normalization of simply typed λ-calculus by showing strong normalization of an alternative formulation of simply typed λ-calculus which they prove is equivalent to the usual formulation; this latter part is the difficult part of the proof.

Klop [76] shows strong normalization of a labeled λ-calculus by an interpretation in Λ_I. Several of the above techniques also use translations from Λ_K to Λ_I. The technique by Klop was discovered independently from a similar technique by Nederpelt [92] and has been reinvented and extended by many researchers, e.g., Khasidashvili [69], Karr [62], de Groote [31], Kfoury and Wells [66], Xi [141, 144], and the author [121]—see Chapter 2.

1.6. Developments

The preceding section analyzed approaches based on the fundamental lemma of perpetuality, etc., to proving that all reductions of typed terms terminate. In the present section we give a similar analysis for reduction of *labeled terms*, i.e., for so-called developments.

The first subsection presents the fundamental lemma of perpetuality for developments along with two related characterizations due to van Raamsdonk and Severi and to Xi, respectively. The second subsection presents a new proof, due independently to van Raamsdonk and Severi and to Xi, of the finite developments theorem. Whereas the proof by van Raamsdonk and Severi and by Xi use their respective characterizations, the proof presented here uses the fundamental lemma of perpetuality for developments. Other proofs of the theorem are reviewed in less detail.

1.6.1. Developments

This subsection introduces developments in terms of labeled terms; we follow Barendregt [3, 11.1–2], with some insignificant deviations.

1.6.1. DEFINITION.

(i) The set $\underline{\Lambda}_K$ ($\underline{\lambda}$-*terms* or *labeled* λ-terms) is defined as follows.

$$\begin{array}{lcl} x \in \underline{\Lambda}_K & & \\ P \in \underline{\Lambda}_K & \Rightarrow & \lambda x.P \in \underline{\Lambda}_K \\ P, Q \in \underline{\Lambda}_K & \Rightarrow & P\,Q \in \underline{\Lambda}_K \\ P, Q \in \underline{\Lambda}_K & \Rightarrow & (\underline{\lambda} x.P)Q \in \underline{\Lambda}_K. \end{array}$$

In the last clause $(\underline{\lambda} x.P)Q$ is a *labeled redex*.

(ii) The notions of reduction $\underline{\beta}, \beta$ on $\underline{\Lambda}_K$ are defined by:

$$\begin{array}{lcl} (\underline{\lambda} x.P)Q & \underline{\beta} & P\{x := Q\} \\ (\lambda x.P)\,Q & \beta & P\{x := Q\}. \end{array}$$

(iii) The notion of reduction β^* is defined by:

$$\beta^* = \underline{\beta} \cup \beta.$$

1.6.2. Remark. As done for λ-terms in Section 1.2.1 we briefly fix the terminology and notation for some well-known concepts—see [3]. We assume familiarity with conventions for omitting parentheses, with the notions of free and bound variables, with the variable convention, and with substitution. Also, $\subseteq$ denotes the subterm relation,[4] $\equiv$ denotes syntactic equality up to renaming of bound variables. $\mathrm{FV}(M)$ denotes the set of variables that occur free in M. A $\underline{\lambda}$-context C is a $\underline{\lambda}$-term with a single occurrence of $[]$; $C[M]$ denotes the result of replacing the occurrence of $[]$ in C by M. $\|M\|$ denotes the number of occurrences of abstractions (labeled and unlabeled), applications, and variables in M. The set $\underline{\Lambda}_I$ is the subset of $\underline{\Lambda}_K$ where, for every $M \in \underline{\Lambda}_I$ and every $\lambda x.P \subseteq M$ and $(\underline{\lambda} x.P)\, Q \subseteq M$, $x \in \mathrm{FV}(P)$.[5]

1.6.3. Lemma.

(i) $M, N \in \underline{\Lambda}_K \;\Rightarrow\; M\{x := N\} \in \underline{\Lambda}_K$.

(ii) $M \in \underline{\Lambda}_K \;\&\; M \to_{\beta^*} N \;\Rightarrow\; N \in \underline{\Lambda}_K$.

Proof.

(i) By induction on M.

(ii) By induction on the derivation of $M \to_{\beta^*} N$, using (i). □

1.6.4. Definition.

(i) A *development* of $M \in \underline{\Lambda}_K$ is $\underline{\beta}$-reduction path from M.

(ii) A *complete* development of $M \in \underline{\Lambda}_K$ is one which ends in an $N \in \mathrm{NF}_{\underline{\beta}}$.

The *finiteness of developments theorem* states that all developments eventually terminate, i.e., that $M \in \mathrm{SN}_{\underline{\beta}}$ for all $M \in \underline{\Lambda}_K$. A stronger form asserts in addition that the $\underline{\beta}$-normal form of $M \in \underline{\Lambda}_K$ is unique.

1.6.2. Fundamental lemma of perpetuality and developments

The following is an analog of the fundamental lemma of perpetuality for developments. It is used implicitly in several proofs in the literature of finite developments.

[4] Recall that the subterms of $(\underline{\lambda} x.P)\, Q$ are the subterms of P and Q and the term $(\underline{\lambda} x.P)\, Q$ itself; that is, $\underline{\lambda} x.P$ is *not* a subterm.

[5] In other words, $\underline{\Lambda}_I$ is the set of all $M \in \underline{\Lambda}_K$ such that replacing every $\underline{\lambda}$ by λ yields an element of Λ_I.

1.6.5. LEMMA. *Assume* $N \in \mathrm{SN}_{\underline{\beta}}$ *if* $x \notin \mathrm{FV}(M)$. *Then*

$$M\{x := N\} \in \mathrm{SN}_{\underline{\beta}} \Rightarrow (\underline{\lambda} x.M)\, N \in \mathrm{SN}_{\underline{\beta}}.$$

PROOF. Suppose $M\{x := N\} \in \mathrm{SN}_{\underline{\beta}}$. If $x \notin \mathrm{FV}(M)$, then, by assumption, $N \in \mathrm{SN}_{\underline{\beta}}$. If $x \in \mathrm{FV}(M)$, then $N \subseteq M\{x := N\}$, so again $N \in \mathrm{SN}_{\underline{\beta}}$. Also $M \in \mathrm{SN}_{\underline{\beta}}$. If $(\underline{\lambda} x.M)\, N \in \infty_{\underline{\beta}}$, then any infinite reduction must have form

$$\begin{array}{lll} (\underline{\lambda} x.M)\, N & \twoheadrightarrow_{\underline{\beta}} & (\underline{\lambda} x.M')\, N' \\ & \rightarrow_{\underline{\beta}} & M'\{x := N'\} \\ & \rightarrow_{\underline{\beta}} & \ldots . \end{array}$$

Since

$$M \twoheadrightarrow_{\underline{\beta}} M' \;\&\; N \twoheadrightarrow_{\underline{\beta}} N' \Rightarrow M\{x := N\} \twoheadrightarrow_{\underline{\beta}} M'\{x := N'\},$$

there is an infinite reduction sequence

$$\begin{array}{lll} M\{x := N\} & \twoheadrightarrow_{\underline{\beta}} & M'\{x := N'\} \\ & \rightarrow_{\underline{\beta}} & \ldots , \end{array}$$

contradicting $M\{x := N\} \in \mathrm{SN}_{\underline{\beta}}$. □

1.6.6. COROLLARY. *If* $N \in \mathrm{SN}_{\underline{\beta}}$, *then*

$$M\{x := N\} \in \mathrm{SN}_{\underline{\beta}} \Rightarrow (\underline{\lambda} x.M)\, N \in \mathrm{SN}_{\underline{\beta}}.$$

PROOF. By Lemma 1.6.5. □

1.6.7. REMARK. Following van Raamsdonk and Severi [106] one can show that $\mathrm{SN}_{\underline{\beta}}$ is the smallest set closed under the rules:

(i) $x \in X$.

(ii) $P \in X \Rightarrow \lambda x.P \in X$.

(iii) $P \in X \;\&\; Q \in X \Rightarrow P\, Q \in X$.

(iv) $P\{x := Q\} \in X \;\&\; Q \in X \Rightarrow (\underline{\lambda} x.P)\, Q \in X$.

The proof of this uses two principles: induction on lexicographically ordered pairs $\langle l_{\underline{\beta}}(\bullet), \| \bullet \| \rangle$ and the fundamental lemma of perpetuality for developments. Proofs using the characterization correspond to direct proofs using the two principles, as was the case for β-reduction—see Remark 1.3.6.

1.6.8. REMARK. Another characterization of $\mathrm{SN}_{\underline{\beta}}$ is due to Xi [140], who considers a relation $\underline{\triangleright}$ on $\underline{\Lambda}_K$ defined by

$$\underline{\triangleright} \;=\; \underline{\sqsupset} \;\cup\; \rightarrow_{\underline{l}},$$

where $\rightarrow_{\underline{l}}$ denotes left-most $\underline{\beta}$-reduction and where $\sqsupseteq$ is the smallest relation closed under the rules:

$$\lambda x.M \sqsupseteq M \quad M\,N \sqsupseteq M \quad M\,N \sqsupseteq N \quad (\underline{\lambda} x.M)\,N \sqsupseteq M \quad (\underline{\lambda} x.M)\,N \sqsupseteq N.$$

Let $\underline{\mathcal{H}}(M_0) = \max\{n \mid M_0 \trianglerighteq M_1 \trianglerighteq \ldots \trianglerighteq M_n\} \in \mathbb{N}^*$. Then, for all $M \in \underline{\Lambda}_K$,

$$\mathrm{SN}_{\underline{\beta}} = \{M \in \underline{\Lambda}_K \mid \underline{\mathcal{H}}(M) < \infty\}.$$

The proof and uses of this characterization are very similar to those of the characterization in [106].

1.6.3. A new proof of the finite developments theorem

The following proof of the finite developments theorem is due to van Raamsdonk and Severi [106]; their proof uses their characterization of $\mathrm{SN}_{\underline{\beta}}$ whereas the following proof uses lexicographic induction and the fundamental lemma of perpetuality—see Remark 1.6.7.

1.6.9. LEMMA. $M, N \in \mathrm{SN}_{\underline{\beta}} \Rightarrow M\{x := N\} \in \mathrm{SN}_{\underline{\beta}}$.

PROOF. By induction on $\langle l_{\underline{\beta}}(M), \|M\|\rangle$. Let $L^* \equiv L\{x := N\}$.

1. $M \equiv x$. Then $M^* \equiv N \in \mathrm{SN}_{\underline{\beta}}$.
2. $M \equiv y$. Then $M^* \equiv y \in \mathrm{SN}_{\underline{\beta}}$.
3. $M \equiv \lambda x.P$. By the induction hypothesis, $P^* \in \mathrm{SN}_{\underline{\beta}}$. It follows that $M^* \equiv \lambda x.P^* \in \mathrm{SN}_{\underline{\beta}}$.
4. $M \equiv P\,Q$. Similar to the preceding case.
5. $M \equiv (\underline{\lambda} y.P)Q$. Then $P\{y := Q\} \in \mathrm{SN}_{\underline{\beta}}$ and $Q \in \mathrm{SN}_{\underline{\beta}}$. By the induction hypothesis $(P\{y := Q\})^* \equiv P^*\{y := Q^*\} \in \mathrm{SN}_{\underline{\beta}}$ and $Q^* \in \mathrm{SN}_{\underline{\beta}}$. By the fundamental lemma of perpetuality for developments it follows that $((\underline{\lambda} y.P)\,Q)^* \equiv (\underline{\lambda} y.P^*)\,Q^* \in \mathrm{SN}_{\underline{\beta}}$. □

1.6.10. THEOREM (Finite Developments). *For all* $M \in \underline{\Lambda}_K$, $M \in \mathrm{SN}_{\underline{\beta}}$.

PROOF. By induction on M.

1. $M \equiv x$. Then $M \in \mathrm{SN}_{\underline{\beta}}$.
2. $M \equiv \lambda x.P$. By the induction hypothesis, $P \in \mathrm{SN}_{\underline{\beta}}$, and therefore $M \in \mathrm{SN}_{\underline{\beta}}$.
3. $M \equiv P\,Q$. Similar to Case 2.
4. $M \equiv (\underline{\lambda} x.P)Q$. By the induction hypothesis $P, Q \in \mathrm{SN}_{\underline{\beta}}$. By Lemma 1.6.9 also $P\{x := Q\} \in \mathrm{SN}_{\underline{\beta}}$. By the fundamental lemma of perpetuality for developments, $M \in \mathrm{SN}_{\underline{\beta}}$. □

There are many proofs of the finite developments theorem in the literature; the following is an incomplete list. The theorem was first proved by Church and Rosser [24, 25] for Λ_I; they also sketch a proof for Λ_K.[6] Curry and Feys [29] and Schroer [113] give full proofs of the theorem for Λ_K. Other proofs were later given independently by Hyland [55] and Hindley [48]. Barendregt et al. [5] subsequently simplified Hyland's proof—see also [3].

Xi [140] gives a proof similar to the above using instead of the fundamental lemma of perpetuality for developments his characterization of $\mathrm{SN}_{\underline{\beta}}$—see Remark 1.6.8. Van Oostrom [95, 96] shows that Lemma 1.6.9 can be eliminated by proving in Theorem 1.6.10 the stronger assertion: for all substitutions σ with $\sigma(x) \in \mathrm{SN}_{\underline{\beta}}$ for all x, it holds that $M\sigma \in \mathrm{SN}_{\underline{\beta}}$.

Another proof due to van Oostrom [95] uses Klop's [76] technique for reducing strong normalization to weak normalization. Other proofs that work by translation into strongly normalizing typed λ-calculi are due to Parigot [98] (see also [80]), van Oostrom and van Raamsdonk [97], van Raamsdonk and Severi [106], Ghilezan [40], and Statman [124].

The theorem has also been proved in several ways for various notions of higher-order rewrite systems. Klop [76] proves it for orthogonal combinatory reduction systems by means of his technique to reduce weak normalization to strong normalization. Van Oostrom [94, 96] proves finiteness of developments for orthogonal higher-order rewriting systems and for pattern rewriting systems. Each of these two results implies finite developments for orthogonal combinatory reduction systems. Melliès [86] gives an axiomatic formulation of developments and shows finite developments for this formulation, which includes orthogonal combinatory reduction systems, but apparently not pattern rewriting systems—see [96]. Khasidashvili [69, 71] gives algorithms to compute longest developments and length of such developments in orthogonal expression reduction systems; these algorithms are special cases of methods to compute longest reductions and the length of such reductions in certain restricted orthogonal expression reduction systems.

One can formulate a version of the fundamental lemma of maximality for developments and use this to give a corresponding effective strategy $\underline{F}_\infty$ computing longest developments and a map $h : \mathrm{SN}_{\underline{\beta}} \to \mathbb{N}$ computing the length of longest developments, similarly to the development in Sections 1.3.5-1.3.6. However, de Vrijer [135] shows that in the case of developments one can do better; he gives a map $f : \underline{\Lambda}_K \to \mathbb{N}$ (called h in [135]) computing the length of longest developments where $f((\underline{\lambda}x.P)\,Q)$ is expressed in terms of $f(P)$ and $f(Q)$; this of course implies finiteness of developments. He also shows that $\underline{F}_\infty$ computes longest developments. In the last section of this chapter we apply to de Vrijer's technique a principle

[6] See the end of [25], or the beginning of Chapter V of [24].

of duality thereby arriving at a technique to compute shortest development as well as the length of such developments.

1.7. Maximal and perpetual redexes

Having applied the techniques related to perpetual and maximal β-reduction strategies from Section 1.3 to various strong normalization problems in Sections 1.4–1.6, we now return to study perpetual and maximal β-redexes. This leads to some *conservation theorems.*

The first subsection reviews some fundamental results relating reduction on terms with and without labels, which will be used in the rest of the section. In particular, a scheme employed in several proofs of conservation theorems in the literature is made explicit. The next three subsections prove the conservation theorem for Λ_I, the conservation theorem for Λ_K, and a related conservation theorem due to Bergstra and Klop, using this proof scheme. These results are used in the fifth subsection to characterize perpetual β-redexes (the notion of maximal β-redex turns out to be trivial). The sixth subsection gives a proof of the normalization theorem similar to the proofs of the conservation theorems, and the last subsection gives a very short proof of the conservation theorem for Λ_I using the normalization theorem.

1.7.1. Reduction on terms with and without labels

There are two important ways to move from a term with labels to one without: one can either *erase* all labels or *reduce* all labeled redexes. This is done by the two maps $|\bullet|, \varphi(\bullet) : \underline{\Lambda}_K \to \Lambda_K$, respectively, introduced below.

1.7.1. Definition. For $M \in \underline{\Lambda}_K$ define $|M| \in \Lambda_K$ as follows.

$$\begin{array}{lcl} |x| & = & x \\ |\lambda x.P| & = & \lambda x.|P| \\ |P\,Q| & = & |P|\,|Q| \\ |(\underline{\lambda} x.P)Q| & = & (\lambda x.|P|)\,|Q|. \end{array}$$

1.7.2. Lemma. *Let* $M, N \in \underline{\Lambda}_K$.

(i) $|M|\{x := |N|\} \equiv |M\{x := N\}|$.

(ii) (Projection.) $M \to_{\beta^*} N \Rightarrow |M| \to_\beta |N|$.

(iii) (Lifting.) $|M| \to_\beta K \Rightarrow \exists N \in \underline{\Lambda}_K : M \to_{\beta^*} N \;\&\; |N| \equiv K$.

Proof. (i): By induction on M. (ii): By induction on the derivation of $M \to_{\beta^*} N$. (iii): By induction on the derivation of $|M| \to_\beta K$. □

1.7.3. Corollary. *Let* $M \in \underline{\Lambda}_K$.

(i) $M \in \mathrm{SN}_{\beta^*} \Leftrightarrow |M| \in \mathrm{SN}_{\beta}$.

(ii) $M \in \mathrm{NF}_{\beta^*} \Leftrightarrow |M| \in \mathrm{NF}_{\beta}$.

The following map $\varphi(M)$ computes a complete *inside-out* development of $M \in \underline{\Lambda}_K$, whereas $M \twoheadrightarrow_{\underline{\beta}} N \in \mathrm{NF}_{\underline{\beta}}$ means that N is the result of an arbitrary complete development of M. In the last clause of the definition it is implicit that no previous clause applies.

1.7.4. Definition. Define $\varphi : \underline{\Lambda}_K \to \Lambda_K$ as follows.

$$
\begin{array}{lcl}
\varphi(x) & = & x \\
\varphi(\lambda x.Q) & = & \lambda x.\varphi(Q) \\
\varphi((\underline{\lambda} x.P)Q) & = & \varphi(P)\{x := \varphi(Q)\} \\
\varphi(P\,Q) & = & \varphi(P)\,\varphi(Q).
\end{array}
$$

1.7.5. Lemma. *For all $M, N \in \underline{\Lambda}_K$:*

(i) $\varphi(M\{x := N\}) = \varphi(M)\{x := \varphi(N)\}$.

(ii)

$$
\begin{tikzcd}
M \arrow[rr, "\to_{\underline{\beta}}"'] \arrow[dr, "\varphi"'] & & N \arrow[dl, "\varphi"] \\
& K &
\end{tikzcd}
$$

(iii)

$$
\begin{tikzcd}
M \arrow[r, "\to_{\beta}"'] \arrow[d, "\varphi"'] & N \arrow[d, "\varphi"] \\
K \arrow[r, "\twoheadrightarrow_{\beta}"'] & L
\end{tikzcd}
$$

Proof. (i): By induction on M. (ii): By induction on the derivation of $M \to_{\underline{\beta}} N$ using (i). (iii): By induction on the derivation of $M \to_{\beta} N$ using (i). $\Box$

The following expresses a relation between $|\bullet|$ and $\varphi(\bullet)$.

1.7.6. Lemma. *Let $M \equiv C[(\lambda x.P)\,Q] \in \underline{\Lambda}_K$, $N \equiv C[P\{x := Q\}] \in \Lambda_K$, and $L \equiv C[(\underline{\lambda} x.P)\,Q] \in \underline{\Lambda}_K$. Then*

$$
\begin{tikzcd}
M \arrow[dd, "\to_{\beta}"'] & \\
& L \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \\
N &
\end{tikzcd}
$$

Proof. By induction on the derivation of $M \to_\beta N$. $\square$

The following proposition expresses the core idea of several proofs of conservation theorems in the literature.

1.7.7. Proposition. *Let* $M \in \Lambda_K$ *and* $M \to_\beta N$. *Then*

$$M \in \infty_\beta \;\Rightarrow\; N \in \infty_\beta$$

if there is an $S \subseteq \underline{\Lambda}_K$ *and* $F^* : \infty_{\beta^*} \to \infty_{\beta^*}$ *with*

(i) $M \equiv C[(\lambda x.P)Q]$, $C[P\{x := Q\}] \equiv N$, $C[(\underline{\lambda} x.P)Q] \in S$ *for some* C, P, Q.

(ii) $L \in S \;\Rightarrow\; F^*(L) \in S$.

(iii) *For all* $L \in S : L \to_\beta F^*(L) \;\Rightarrow\; \varphi(L) \twoheadrightarrow^+_\beta \varphi(F^*(L))$.

Proof. Let $M \to_\beta N$ where $M \in \infty_\beta$, and let C, P, Q, S, and F^* be as required in (i)-(iii).

Let $L_0 \equiv C[(\underline{\lambda} x.P)\ Q]$, $N_0 \equiv N$, and $M_0 \equiv M$. By Corollary 1.7.3, $L_0 \in \infty_{\beta^*}$. Since F^* is perpetual,

$$L_0 \to_{\beta^*} L_1 \to_{\beta^*} L_2 \ldots$$

with $L_i = F^*(L_{i-1})$ is infinite.

By Lemma 1.7.5–1.7.6, and the assumptions we can erect the diagram:[7]

$$\begin{tikzcd}
M_0 \arrow[rr, "\to_\beta"'] \arrow[dd, "\to_\beta"'] & & M_1 \arrow[rr, "\to_\beta"'] & & M_2 \arrow[rr, "\to_\beta"'] & & \cdots \\
& L_0 \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_\beta"'] & & L_1 \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_{\underline{\beta}}"'] & & L_2 \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_\beta"'] & & \cdots \\
N_0 \arrow[rr, "\twoheadrightarrow^+_\beta"'] & & N_1 \arrow[rr, "\equiv"'] & & N_2 \arrow[rr, "\twoheadrightarrow^+_\beta"'] & & \cdots
\end{tikzcd}$$

Here

$$\begin{array}{lcl} L_i \to_\beta L_{i+1} & \Rightarrow & N_i \twoheadrightarrow^+_\beta N_{i+1} \\ L_i \to_{\underline{\beta}} L_{i+1} & \Rightarrow & N_i \equiv N_{i+1}. \end{array}$$

By finiteness of developments $L_i \to_\beta L_{i+1}$ for infinitely many i, giving an infinite β-reduction path from N_0. $\square$

1.7.8. Remark. The diagram used in the above proof is an infinite version of the diagram used by Barendregt [3, 11.1] to prove the *strip lemma,* the main lemma in his proof of the Church-Rosser property.

[7] The reduction $M_0 \to_\beta M_1 \to_\beta \ldots$ is constructed by projection of $L_0, L_1, \ldots$, but the former reduction path is not essential.

1.7.2. The conservation theorem for Λ_I

We now use Proposition 1.7.7 to prove the conservation theorem for Λ_I.

1.7.9. LEMMA. *For any $M \in \underline{\Lambda}_I$: $M \to_{\beta^*} N \Rightarrow N \in \underline{\Lambda}_I$.*[8]

PROOF. Show by induction on M that

$$M, N \in \underline{\Lambda}_I \Rightarrow M\{x := N\} \in \underline{\Lambda}_I, \tag{$*$}$$

and by induction on the derivation of $M \to_{\beta^*} N$ that

$$FV(M) \subseteq \mathrm{FV}(N). \tag{$+$}$$

Using $(*)$ and $(+)$ proceed by induction on the derivation of $M \to_{\beta^*} N$. □

1.7.10. LEMMA. *For any $M \in \underline{\Lambda}_I$: $M \to_\beta N \Rightarrow \varphi(M) \twoheadrightarrow_\beta^+ \varphi(N)$*

PROOF. Show by induction on M that for all $M \in \underline{\Lambda}_I$:

$$\mathrm{FV}(M) \subseteq \mathrm{FV}(\varphi(M)).$$

Using this property and Lemma 1.7.5(i), proceed by induction on the derivation of $M \to_\beta N$. □

1.7.11. THEOREM (Conservation for Λ_I). *If $M \in \Lambda_I$ and $M \to_\beta N$, then*

$$M \in \infty_\beta \Rightarrow N \in \infty_\beta.$$

PROOF. By the preceding two lemmas we can use Proposition 1.7.7 with $S = \underline{\Lambda}_I$ and any partial, perpetual β^*-reduction strategy in the role of F^*. □

1.7.12. REMARK. Since Λ_I is closed under β-reduction, we can view β as a notion of reduction on Λ_I, and we can view any β-reduction strategy on Λ_K as a β-reduction strategy on Λ_I. The conservation theorem for Λ_I states that in Λ_I, all β-redexes and β-reduction strategies are perpetual.

1.7.13. COROLLARY. *Let $M \in \Lambda_I$.*

(i) $M \in \mathrm{WN}_\beta \Rightarrow M \in \mathrm{SN}_\beta$.
(ii) $M \in \mathrm{WN}_\beta$ & $N \subseteq M \Rightarrow N \in \mathrm{WN}_\beta$.

PROOF.

(i) If $M \in \mathrm{WN}_\beta$, then $M \twoheadrightarrow_\beta N \in \mathrm{NF}_\beta$, for some N. If $M \in \infty_\beta$, then by the conservation theorem, $N \in \infty_\beta$, a contradiction.

[8] $\underline{\Lambda}_I$ is defined in Remark 1.6.2.

(ii) If $M \in \mathrm{WN}_\beta$ and $N \subseteq M$, then $M \in \mathrm{SN}_\beta$, and therefore $N \in \mathrm{SN}_\beta$, in particular $N \in \mathrm{WN}_\beta$. □

As mentioned in Remark 1.4.19 and at the end of Section 1.5, a number of techniques to prove strong normalization of typed λ-calculi use translations from Λ_K to Λ_I. Most of these techniques also use some variant of Corollary 1.7.13(i). For instance, the techniques by the author [121] and Xi [141, 144] use a translation $t : \Lambda_K \to \Lambda_I$ such that $t(M) \in \mathrm{SN}_\beta \Rightarrow M \in \mathrm{SN}_\beta$. By the corollary, it then suffices to show $t(M) \in \mathrm{WN}_\beta$ to infer $M \in \mathrm{SN}_\beta$.

The conservation theorem for Λ_I is due to Church and Rosser [24, 25], and was later proved by Curry and Feys [29]. A proof in the spirit of the former proof is given by Barendregt et al. [3, 5]. These proofs are all by syntactic methods; a semantic proof appears in [51]. Klop [76] proves a generalization of the theorem for orthogonal non-erasing combinatory reduction systems.

The above proof is a slight simplification of the proof by Barendregt et al.; our proof uses inside-out developments rather than arbitrary developments and avoids the explicit notions of redex occurrence and residual (similarly, Takahashi [127] proves Curry and Feys' *standardization theorem* using parallel reductions, arguing that these are more convenient than the arbitrary developments used in, e.g., Mitschke's proof [89]—see also [3]). A very short proof will be given in the last subsection.

1.7.3. The conservation theorem for Λ_K

We now use Proposition 1.7.7 to prove the conservation theorem for Λ_K.

1.7.14. Definition.

(i) An *I-redex* is a term $(\lambda x.P)\,Q \in \Lambda_K$ where $x \in \mathrm{FV}(P)$. A *K-redex* is a term $(\lambda x.P)\,Q \in \Lambda_K$ where $x \notin \mathrm{FV}(P)$.

(ii) We write $\mathbf{K}\,P\,Q$ for $(\lambda x.P)\,Q$ and $\underline{\mathbf{K}}\,P\,Q$ for $(\underline{\lambda} x.P)\,Q$ when $x \notin \mathrm{FV}(P)$ and call P and Q the *body* and *argument*, respectively, of the redex.

(iii) $\underline{\Lambda}^I$ is the subset of $\underline{\Lambda}_K$ where for each $M \in \underline{\Lambda}^I$ and each $(\underline{\lambda} x.P)Q \subseteq M$, it holds that $x \in \mathrm{FV}(P)$.

(iv) We write $M \equiv (\lambda^{\!*} x.P)\,Q$ if $M \equiv (\lambda x.P)\,Q$ or $M \equiv (\underline{\lambda} x.P)\,Q$.

1.7.15. Definition. Define $F_1^* : \infty_{\beta^*} \to \underline{\Lambda}_K$ by:

$$
\begin{array}{lcll}
F_1^*(x\,\vec{P}\,Q\,\vec{R}) & = & x\,\vec{P}\,F_1^*(Q)\,\vec{R} & \text{if } \vec{P} \in \mathrm{SN}_{\beta^*}, Q \notin \mathrm{SN}_{\beta^*} \\
F_1^*(\lambda x.P) & = & \lambda x.F_1^*(P) & \\
F_1^*((\lambda^{\!*} x.P)\,Q\,\vec{R}) & = & P\{x := Q\}\,\vec{R} & \text{if } Q \in \mathrm{SN}_{\beta^*} \\
F_1^*((\lambda^{\!*} x.P)\,Q\,\vec{R}) & = & (\lambda^{\!*} x.P)\,F_1^*(Q)\,\vec{R} & \text{if } Q \notin \mathrm{SN}_{\beta^*}.
\end{array}
$$

1.7.16. Lemma. *For all $M \in \infty_{\beta^*}$: $F_1^*(M) \in \infty_{\beta^*}$.*

Proof. First show that, for all $M \in \infty_{\beta^*}$:

$$|F_1^*(M)| = F_1(|M|) \tag{$*$}$$

by induction on M using Corollary 1.7.3. Since $M \in \infty_{\beta^*}$, $|M| \in \infty_\beta$ by Corollary 1.7.3. By $(*)$ and perpetuality of F_1, $|F_1^*(M)| = F_1(|M|) \in \infty_\beta$. Then by Corollary 1.7.3, $F_1^*(M) \in \infty_{\beta^*}$. □

1.7.17. Lemma. *For all $M \in \underline{\Lambda}^I$: $F_1^*(M) \in \underline{\Lambda}^I$.*

Proof. First prove by induction on M that

$$M, N \in \underline{\Lambda}^I \;\Rightarrow\; M\{x := N\} \in \underline{\Lambda}^I.$$

Using this show $F_1^*(M) \in \underline{\Lambda}^I$ by induction on M. □

1.7.18. Lemma. *For all $M \in \underline{\Lambda}^I$: $M \to_\beta F_1^*(M) \;\Rightarrow\; \varphi(M) \twoheadrightarrow_\beta^+ \varphi(F_1^*(M))$.*

Proof. By induction on M show that for all $M \in \underline{\Lambda}^I$: $\mathrm{FV}(M) \subseteq \mathrm{FV}(\varphi(M))$. Using this and Lemma 1.7.5 proceed by induction on M. □

1.7.19. Theorem (Conservation for Λ_K). *If $M \equiv C[\Delta] \to_\beta C[\Delta'] \equiv N$ where $M \in \Lambda_K$ and Δ is an I-redex, then*

$$M \in \infty_\beta \;\Rightarrow\; N \in \infty_\beta.$$

Proof. By the preceding three lemmas we can use Proposition 1.7.7 with $S = \underline{\Lambda}^I$ and $F^* = F_1^*$. □

1.7.20. Corollary. *Any I-redex is perpetual.*

1.7.21. Discussion (Barendregt et al. [3, 5]). The proof of the conservation theorem for Λ_I does not carry over to Λ_K, i.e., we cannot use Proposition 1.7.7 with $S = \underline{\Lambda}_K$ and F^* any partial, perpetual β^*-reduction strategy. For instance, $(\lambda x.\mathbf{K}\,\mathbf{I}\,x)\,\Omega$ is an I-redex, but the diagram in the proof of Proposition 1.7.7 is:

$$\begin{tikzcd}
(\lambda x.\mathbf{K}\,\mathbf{I}\,x)\,\Omega \arrow[rr, "\to_\beta"'] \arrow[dd, "\to_\beta"'] & & (\lambda x.\mathbf{I})\,\Omega \arrow[rr, "\to_\beta"'] & & (\lambda x.\mathbf{I})\,\Omega \arrow[rr, "\to_\beta"'] & & \cdots & \\
& (\underline{\lambda} x.\mathbf{K}\,\mathbf{I}\,x)\Omega \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_\beta"'] & & (\underline{\lambda} x.\mathbf{I})\Omega \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_\beta"'] & & (\underline{\lambda} x.\mathbf{I})\Omega \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_\beta"'] & & \cdots \\
\mathbf{K}\,\mathbf{I}\,\Omega \arrow[rr, "\to_\beta"'] & & \mathbf{I} \arrow[rr, "\equiv"'] & & \mathbf{I} \arrow[rr, "\equiv"'] & & \cdots &
\end{tikzcd}$$

After one step, no reductions occur in the lower sequence. The problem is that property (iii) in Proposition 1.7.7 fails for $S = \underline{\Lambda}_K$ if F^* is arbitrary. This is because in $M \to_\beta N$ the reduction may take place in the argument Q of a labeled K-redex $\underline{\mathbf{K}}\, P\, Q$, and then $\varphi(M) \equiv \varphi(N)$.

However, (iii) does hold for $S = \underline{\Lambda}^I$, i.e., when only I-redexes are labeled. The rescue then is that labeling an I-redex yields a term in $\underline{\Lambda}^I$, so (i) also holds. Moreover, to turn a $\underline{\Lambda}^I$ term into a term outside $\underline{\Lambda}^I$ would require a reduction step inside P of $(\underline{\lambda}x.P)Q$ which erased all occurrences of x, but F_1^* never reduces a redex inside P of a redex $(\underline{\lambda}x.P)\, Q$, so (ii) holds too.

The conservation theorem for Λ_K is due to Barendregt et al. [3, 5]. Khasidashvili [71] shows a version for orthogonal expression reduction systems, using perpetuality of his limit strategies mentioned earlier (see the end of Section 1.3.5). Our proof is a slight simplification of the proof by Barendregt et al.; apart from the simplifications mentioned in the preceding subsection, our proof uses a simpler perpetual reduction strategy than the proof by Barendregt et al.

1.7.4. Conservation under K-reduction

The preceding two subsections characterized perpetual I-redexes in Λ_I and Λ_K. Now we characterize perpetual K-redexes in Λ_K.

1.7.22. Definition.

(i) $\underline{\Lambda}^K$ is the subset of $\underline{\Lambda}_K$ such that for all $M \in \underline{\Lambda}_K$ and all $(\underline{\lambda}x.P)Q \subseteq M$, it holds that $x \notin \mathrm{FV}(P)$.

(ii) For $(L, R) = (\Lambda_K, \beta)$ and $(L, R) = (\underline{\Lambda}_K, \beta^*)$, an SN_R-*substitution* is a substitution σ such that $x\sigma \in \mathrm{SN}_R$ for every variable x. For $P, Q \in L$, we write $P \geq_\infty^R Q$ iff for all SN_R-substitutions σ:

$$P\sigma \in \infty_R \Leftarrow Q\sigma \in \infty_R.$$

For $Q \in \mathrm{SN}_R$, $\sigma + \{x := Q\}$ maps x to Q and acts as σ on any other variable. By projection and lifting $P \geq_\infty^\beta Q \Leftrightarrow P \geq_\infty^{\beta^*} Q$ for any $P, Q \in \Lambda_K$.

1.7.23. Definition. Define $F_2^* : \infty_{\beta^*} \to \underline{\Lambda}_K$ by:

$$\begin{array}{lcll}
F_2^*(x\, \vec{P}\, Q\, \vec{R}) & = & x\, \vec{P}\, F_2^*(Q)\, \vec{R} & \text{if } \vec{P} \in \mathrm{SN}_{\beta^*}, Q \notin \mathrm{SN}_{\beta^*} \\
F_2^*(\lambda x.P) & = & \lambda x.F_2^*(P) & \\
F_2^*((\lambda^{\!*} x.P)\, Q\, \vec{R}) & = & P\{x := Q\}\, \vec{R} & \text{if } P, Q \in \mathrm{SN}_{\beta^*} \\
F_2^*((\lambda^{\!*} x.P)\, Q\, \vec{R}) & = & (\lambda^{\!*} x.F_2^*(P))\, Q\, \vec{R} & \text{if } P \notin \mathrm{SN}_{\beta^*} \\
F_2^*((\lambda^{\!*} x.P)\, Q\, \vec{R}) & = & (\lambda^{\!*} x.P)\, F_2^*(Q)\, \vec{R} & \text{if } P \in \mathrm{SN}_{\beta^*}, Q \notin \mathrm{SN}_{\beta^*}.
\end{array}$$

1.7.24. LEMMA. *For all* $M \in \infty_{\beta^*}$*:* $F_2^*(M) \in \infty_{\beta^*}$.

PROOF. First show that, for all $M \in \infty_{\beta^*}$:

$$|F_2^*(M)| = F_2(|M|) \qquad (*)$$

by induction on M using Corollary 1.7.3. Since $M \in \infty_{\beta^*}$, $|M| \in \infty_\beta$ by Corollary 1.7.3. By $(*)$ and perpetuality of F_2, $|F_2^*(M)| = F_2(|M|) \in \infty_\beta$. Then by Corollary 1.7.3, $F_2^*(M) \in \infty_{\beta^*}$. □

1.7.25. LEMMA. *For all* $M \in \underline{\Lambda}^K$, $F_2^*(M) \in \underline{\Lambda}^K$.

PROOF. First prove by induction on M that

$$M, N \in \underline{\Lambda}^K \Rightarrow M\{x := N\} \in \underline{\Lambda}^K.$$

Using this property proceed by induction on M. □

1.7.26. DEFINITION. Let X be a set of variables.

(i) An SN_{β^*}-substitution σ is X*-neutral*, if $x\sigma = x$ for all $x \in X$.

(ii) M is X*-good* if, for all $\underline{\mathbf{K}}AB \subseteq M$ and X-neutral σ, $A\sigma \in \infty_{\beta^*} \Leftarrow B\sigma \in \infty_{\beta^*}$.

(iii) X *respects* M if $\mathrm{FV}(M) \subseteq X$ and $X \cap \mathrm{BV}(M) = \{\}$.

1.7.27. DEFINITION. For $M \in \infty_{\beta^*}$, define the set of variables $V(M)$ by:

$$\begin{array}{lcll}
V(x\,\vec{P}\,Q\,\vec{R}) & = & V(Q) & \text{if } \vec{P} \in \mathrm{SN}_{\beta^*}, Q \notin \mathrm{SN}_{\beta^*} \\
V(\lambda x.P) & = & \{x\} \cup V(P) & \\
V((\lambda^* x.P)\,Q\,\vec{R}) & = & \{\} & \text{if } P, Q \in \mathrm{SN}_{\beta^*} \\
V((\lambda^* x.P)\,Q\,\vec{R}) & = & \{x\} \cup V(P) & \text{if } P \notin \mathrm{SN}_{\beta^*} \\
V((\lambda^* x.P)\,Q\,\vec{R}) & = & V(Q) & \text{if } P \in \mathrm{SN}_{\beta^*}, Q \notin \mathrm{SN}_{\beta^*}.
\end{array}$$

1.7.28. LEMMA. *For all* $M \in \infty_{\beta^*}$*:* $V(M) \subseteq V(F_2^*(M))$.

PROOF. By induction on M using perpetuality of F_2^*. □

1.7.29. LEMMA. *Let* $M \in \infty_{\beta^*} \cap \underline{\Lambda}^K$, M *be* $X \cup V(M)$*-good,* X *respect* M.

(i) $F_2^*(M)$ *is* $X \cup V(F_2^*(M))$*-good, and* X *respects* $F_2^*(M)$.

(ii) $M \to_\beta F_2^*(M) \Rightarrow \varphi(M) \twoheadrightarrow_\beta^+ \varphi(F_2^*(M))$.

PROOF. Let $M \in \infty_{\beta^*} \cap \underline{\Lambda}^K$, M be $X \cup V(M)$-good, X respect M.

(i): Since reduction does not invent new free variables, and new bound variables are chosen fresh, X respects $F_2^*(M)$.

We show that $F_2^*(M)$ is $X \cup V(F_2^*(M))$-good by induction on M. Let $\underline{\mathbf{K}}\,A\,B \subseteq F_2^*(M)$ and let σ be an $X \cup V(F_2^*(M))$-neutral SN_{β^*}-substitution. We are to show that $A\sigma \in \infty_{\beta^*} \Leftarrow B\sigma \in \infty_{\beta^*}$.

1. $M \equiv x\,\vec{P}\,Q\,\vec{R}$, and $\vec{P} \in \mathrm{SN}_{\beta^*}, Q \notin \mathrm{SN}_{\beta^*}$. Then $F_2^*(M) = x\,\vec{P}\,F_2^*(Q)\,\vec{R}$.

   1.1. $\underline{\mathbf{K}}\,A\,B \subseteq S$, where $\vec{P} = \vec{P_1}, S, \vec{P_2}$ or $\vec{R} = \vec{R_1}, S, \vec{R_2}$. Then, by Lemma 1.7.28, $V(M) \subseteq V(F_2^*(M))$. Therefore, σ is $X \cup V(M)$-neutral. Since M is $X \cup V(M)$-good, $A\sigma \in \infty_{\beta^*} \Leftarrow B\sigma \in \infty_{\beta^*}$.

   1.2. $\underline{\mathbf{K}}\,A\,B \subseteq F_2^*(Q)$. Since $V(M) = V(Q)$, Q is $X \cup V(Q)$-good. By the induction hypothesis, $F_2^*(Q)$ is $X \cup V(F_2^*(Q))$-good. Since F_2^* is perpetual, $V(F_2^*(M)) = V(F_2^*(Q))$, so $F_2^*(Q)$ is $X \cup V(F_2^*(M))$-good. Therefore, $A\sigma \in \infty_{\beta^*} \Leftarrow B\sigma \in \infty_{\beta^*}$.

2. $M \equiv \lambda x.P$. Then $F_2^*(M) = \lambda x.F_2^*(P)$. Then $\underline{\mathbf{K}}\,A\,B \subseteq F_2^*(P)$. Since $V(M) = \{x\} \cup V(P)$, P is $X \cup \{x\} \cup V(P)$-good. Here $X \cup \{x\}$ respects P, so by the induction hypothesis, $F_2^*(P)$ is $X \cup \{x\} \cup V(F_2^*(P))$-good. Since $V(F_2^*(M)) = \{x\} \cup V(F_2^*(P))$, $F_2^*(P)$ is $X \cup V(F_2^*(M))$-good. Then $A\sigma \in \infty_{\beta^*} \Leftarrow B\sigma \in \infty_{\beta^*}$.

3. $M \equiv (\lambda^* x.P)\,Q\,\vec{R}$. We consider three subcases.

   3.1. $P \in \infty_{\beta^*}$. Then $F_2^*(M) = (\lambda^* x.F_2^*(P))\,Q\,\vec{R}$. There are, in turn, three cases to consider.

   3.1.1. $\underline{\mathbf{K}}\,A\,B \subseteq S$, where $S \equiv Q$ or $\vec{R} = \vec{R_1}, S, \vec{R_2}$. Similar to Case 1.1.

   3.1.2. $\underline{\mathbf{K}}\,A\,B \subseteq F_2^*(P)$. Similar to Case 2.

   3.1.3. $\underline{\mathbf{K}}\,A\,B \equiv (\lambda^* x.F_2^*(P))\,Q$. Since F_2^* is perpetual, $F_2^*(P) \in \infty_{\beta^*}$, i.e., $A \in \infty_{\beta^*}$. Thus $A\sigma \in \infty_{\beta^*}$, so $A\sigma \in \infty_{\beta^*} \Leftarrow B\sigma \in \infty_{\beta^*}$ trivially.

   3.2. $P \in \mathrm{SN}_{\beta^*}$, $Q \notin \mathrm{SN}_{\beta^*}$. As in Case 3.1, there are three subcases.

   3.2.1. $\underline{\mathbf{K}}\,A\,B \subseteq S$, where $S \equiv P$ or $\vec{R} = \vec{R_1}, S, \vec{R_2}$. Similar to Case 1.1.

   3.2.2. $\underline{\mathbf{K}}\,A\,B \subseteq F_2^*(Q)$. Similar to Case 1.2.

   3.2.3. $\underline{\mathbf{K}}\,A\,B \equiv (\lambda^* x.P)\,F_2^*(Q)$. This case is impossible. Indeed, suppose that $\underline{\mathbf{K}}AB \equiv (\lambda^* x.P)\,F_2^*(Q)$, so $\underline{\mathbf{K}}AB' \equiv (\lambda^* x.P)\,Q \subseteq M$. The identity substitution ι is clearly $X \cup V(M)$-neutral, but according to the above, $A\iota \notin \infty_{\beta^*}$ and $B'\iota \in \infty_{\beta^*}$, contradicting the assumption that M is $X \cup V(M)$-good.

   3.3. $P, Q \in \mathrm{SN}_{\beta^*}$. Then $F_2^*(M) = P\{x := Q\}\,\vec{R}$.

   3.3.1. $\underline{\mathbf{K}}\,A\,B \subseteq S$, where $S \in \vec{R}$. Similar to Case 1.1.

   3.3.2. $\underline{\mathbf{K}}\,A\,B \subseteq P\{x := Q\}$. We consider three subcases.

   (a) $\underline{\mathbf{K}}\,A\,B \subseteq Q$. Similar to Case 1.1.

   (b) $\underline{\mathbf{K}}\,A\,B \subseteq P$. Similar to Case 1.1.

   (c) $\underline{\mathbf{K}}\,A\,B \equiv \underline{\mathbf{K}}\,(I\{x := Q\})\,(J\{x := Q\})$, where $\underline{\mathbf{K}}\,I\,J \subseteq P$. Since $\mathrm{FV}(Q) \subseteq \mathrm{FV}(M) \subseteq X$, $y\sigma = y$, for all $y \in \mathrm{FV}(Q)$. Therefore, $\underline{\mathbf{K}}A\sigma B\sigma \equiv \underline{\mathbf{K}}I\sigma' J\sigma'$, where $\sigma' = \sigma + \{x := Q\}$.

Since $V(M) \subseteq V(F_2^*(M))$, σ is $X \cup V(M)$-neutral. Now $x \notin V(M)$, and $x \in \mathrm{BV}(M)$ so $x \notin X$. Therefore σ' is $X \cup V(M)$-neutral. Thus, since M is $X \cup V(M)$-good, $A\sigma \equiv I\sigma' \in \infty_{\beta^*} \Leftarrow B\sigma \equiv J\sigma' \in \infty_{\beta^*}$.

(ii): By induction on M. □

1.7.30. Theorem (Conservation of K-redexes). *Assume that* $P \geq_\infty^\beta Q$ *and* $M \equiv C[\mathbf{K}\, P\, Q] \to_\beta C[P] \equiv N$ *where* $M \in \Lambda_K$. *Then*

$$M \in \infty_\beta \Rightarrow N \in \infty_\beta.$$

Proof. Suppose $M \in \infty_\beta$ and $M \equiv C[\mathbf{K}\ P\ Q] \to_\beta C[P] \equiv N$, where $P \geq_\infty^\beta Q$. Let $F^* = F_2^*$ and

$$S = \{J \in \underline{\Lambda}^K \cap \infty_{\beta^*} \mid J \text{ is } \mathrm{FV}(M) \cup V(J) - \text{good } \& \ \mathrm{FV}(M) \text{ respects } J\}.$$

Then condition (i) of Proposition 1.7.7 is clearly satisfied, and by Lemmas 1.7.24, 1.7.25, and 1.7.29, conditions (ii) and (iii) are also satisfied. □

1.7.31. Corollary. *A K-redex* $\mathbf{K}\, P\, Q$ *is perpetual if* $P \geq_\infty^\beta Q$.

1.7.32. Corollary. *A K-redex* $\mathbf{K}\ P\ Q$ *is perpetual if one of the following conditions are satisfied:*

(i) $P \in \infty_\beta$.

(ii) $Q \in \mathrm{SN}_\beta$ *and* $\mathrm{FV}(Q) = \emptyset$.

1.7.33. Corollary. *A redex* $(\lambda x.P)\ Q$ *is perpetual if*

$$P\sigma\{x := Q\sigma\} \in \infty_\beta \Leftarrow Q\sigma \in \infty_\beta$$

for all SN_β*-substitutions* σ.

Proof. If $x \in \mathrm{FV}(P)$ then the redex is perpetual by the conservation theorem for Λ_K. If $x \notin \mathrm{FV}(P)$, then the condition of the theorem is equivalent to $P \geq_\infty^\beta Q$, so the redex is again perpetual by the preceding Corollary. □

1.7.34. Discussion. It is not true that $M \in \Lambda_K$ and $M \to_\beta N$ by contraction of any K-redex implies

$$M \in \infty_\beta \Rightarrow N \in \infty_\beta.$$

For instance, for the term $M \equiv \mathbf{K}\,\mathbf{I}\,\Omega$ and the reduction step $\mathbf{K}\,\mathbf{I}\,\Omega \to_\beta \mathbf{I}$ the assertion is wrong. The diagram from the proof of Proposition 1.7.7 is:

$$\begin{tikzcd}
\mathbf{K}\,\mathbf{I}\,\Omega \arrow[rr, "\to_\beta"'] \arrow[dd, "\to_\beta"'] & & \mathbf{K}\,\mathbf{I}\,\Omega \arrow[rr, "\to_\beta"'] & & \mathbf{K}\,\mathbf{I}\,\Omega \arrow[rr, "\to_\beta"'] & & \cdots \\
& \underline{\mathbf{K}}\,\mathbf{I}\,\Omega \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_\beta"'] & & \underline{\mathbf{K}}\,\mathbf{I}\,\Omega \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_\beta"'] & & \underline{\mathbf{K}}\,\mathbf{I}\,\Omega \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_\beta"'] & & \cdots \\
\mathbf{I} \arrow[rr, "\equiv"'] & & \mathbf{I} \arrow[rr, "\equiv"'] & & \mathbf{I} \arrow[rr, "\equiv"'] & & \cdots
\end{tikzcd}$$

In the lower sequence every term is identical to its successor, and the problem evidently is the same as earlier: (ii) of Proposition 1.7.7 fails for $S = \underline{\Lambda}^K$; that is, in $M \to_\beta N$ the reduction step may occur in the argument of a labeled K-redex, and then $\varphi(M) \equiv \varphi(N)$.

However, (ii) holds if the reduction step is not inside an argument of a labeled K-redex.[9] If the initial K-redex $\mathbf{K}\ P\ Q$ is such that $P \geq_\infty^\beta Q$ and we use F_2^* to compute the middle reduction path, then no reduction will be inside the argument of labeled K-redex. Indeed, when F_2^* contracts $(\lambda^* x.K)\ L$, $L \in \mathrm{SN}_{\beta^*}$. Since F_2^* computes standard reduction paths, this means, roughly, that every residual of the initial labeled K-redex $\underline{\mathbf{K}}\ P\ Q$ has form $\underline{\mathbf{K}}\ P\sigma\ Q\sigma$ where σ is an SN_{β^*}-substitution. Since $P \geq_\infty^{\beta^*} Q$, also $P\sigma \in \infty_{\beta^*} \Leftarrow Q\sigma \in \infty_{\beta^*}$. Therefore, F_2^* does not contract a redex inside $Q\sigma$. It may happen that F_2^* contracts a redex inside $P\sigma$. In this case, all the following reductions will also be inside $P\sigma$.

Theorem 1.7.30 is due to Bergstra and Klop [18]. Our proof above is a simplification of the proof of Bergstra and Klop. Xi [140] proves Corollary 1.7.33 directly, instead of proving conservation for Λ_K and the Bergstra-Klop theorem separately. Khasidashvili and Ogawa [74] independently prove Corollary 1.7.33, using a variant of the strategy F_2, and study applications to various restricted λ-calculi. Corollary 1.7.32(ii) is also taken from Khasidashvili and Ogawa [74].

1.7.5. Perpetual and maximal redexes

The following proposition shows that the converse of Theorem 1.7.30 also holds. The idea of the proof is that one can simulate the effect of substitutions by means of contexts and reductions.

1.7.35. Proposition (Bergstra and Klop [18]). *Assume*

$$C[\mathbf{K}\ P\ Q] \in \infty_\beta \ \Rightarrow\ C[P] \in \infty_\beta$$

[9] This observation generalizes the earlier observation that (ii) holds in $\underline{\Lambda}_K$ if all labeled redexes are I-redexes. In that case no reduction can take place inside the argument of a labeled K-redex.

for all contexts C. Then $P \geq^{\beta}_{\infty} Q$.

Proof. To show $P \geq^{\beta}_{\infty} Q$, let $\vec{R} \in \mathrm{SN}_\beta$, and suppose

$$Q\{\vec{x} := \vec{R}\} \in \infty_\beta.$$

Put $C \equiv (\lambda\vec{x}.[])\,\vec{R}$. Since

$$(\lambda\vec{x}.(\mathbf{K}\,P\,Q))\,\vec{R} \twoheadrightarrow_\beta \mathbf{K}\,(P\{\vec{x} := \vec{R}\})\,(Q\{\vec{x} := \vec{R}\}),$$

also

$$C[\mathbf{K}\,P\,Q] \in \infty_\beta.$$

By our assumptions, this implies $C[P] \in \infty_\beta$, i.e., $(\lambda\vec{x}.P)\,\vec{R} \in \infty_\beta$. Since $\vec{R} \in \mathrm{SN}_\beta$, for some n

$$F_1^n((\lambda\vec{x}.P)\,\vec{R}) = P\{\vec{x} := \vec{R}\},$$

and by perpetuality of F_1, $P\{\vec{x} := \vec{R}\} \in \infty_\beta$ as required. □

The following corollary, in which (i) is due to Barendregt et al. [3, 5] and (ii) is due to Bergstra and Klop [18], sums up the situation.

1.7.36. Corollary. *A redex $(\lambda x.P)\,Q$ is perpetual iff*

$(\lambda x.P)\,Q$ is an I-redex; or

$(\lambda x.P)\,Q$ is a K-redex with $P \geq^{\beta}_{\infty} Q$.

Proof. By Corollary 1.7.20, Corollary 1.7.31 and Proposition 1.7.35. □

We now proceed to characterize maximal redexes. The intuition is as follows. Given a redex Δ with contractum Δ', we can conceive a context C which is such that $C[\Delta]$ can duplicate Δ. Therefore the longest reduction path from $C[\Delta]$ is obtained only if we do not contract Δ until it has been duplicated. But then Δ is not maximal. The only escape is when the contractum of Δ has an infinite reduction path. Then $C[\Delta']$ has arbitrarily long reduction paths, so Δ is maximal.

1.7.37. Proposition. *Redex Δ with contractum Δ' is maximal iff $\Delta' \in \infty_\beta$.*

Proof.

$\Leftarrow$: If $\Delta' \in \infty_\beta$ then for any $n > 0$ and context C, $C[\Delta'] \in (n-1)_\beta$.

$\Rightarrow$: We assume $\Delta' \in \mathrm{SN}_\beta$ and prove that Δ is not maximal by finding an n such that $C[\Delta] \in n_\beta$ but not $C[\Delta'] \in (n-1)_\beta$.

Since $\Delta' \in \mathrm{SN}_\beta$ there is by König's Lemma an $m \in \mathbb{N}$ such that $\Delta' \in (m-1)_\beta$ and $\Delta' \notin m_\beta$. Then $\Delta \in m_\beta$. So for $C \equiv (\lambda x.\lambda y.y\, x\, x)\,[]$ we have for some Q

$$C[\Delta] \to_\beta \lambda y.y\;\Delta\;\Delta \twoheadrightarrow_\beta^{2m} \lambda y.y\;Q\;Q;$$

that is, $C[\Delta] \in (2m+1)_\beta$.

On the other hand, any reduction of $C[\Delta']$ has form

$$C[\Delta'] \twoheadrightarrow_\beta^{k} C[Q'] \to_\beta \lambda y.y\;Q'\;Q' \twoheadrightarrow_\beta^{2l} \lambda y.y\;Q''\;Q''$$

for some Q', Q'', where $k + l \leq m - 1$, and therefore $k + 1 + 2l < 2m$. So, $C[\Delta'] \notin (2m)_\beta$. □

1.7.6. The normalization theorem

In this subsection we prove the normalization theorem for Λ_K which states that repeated contraction of the left-most redex in a weakly normalizing term eventually leads to a normal-form. We use a technique very similar to that used to prove conservation theorems in the preceding subsections.

1.7.38. Definition. Define $F_l^* : \underline{\Lambda}_K \to \underline{\Lambda}_K$ a follows. If $M \in \mathrm{NF}_{\beta^*}$ then $F_l^*(M) = M$; otherwise,

$$\begin{array}{lcll}
F_l^*(x\,\vec{P}\,Q\,\vec{R}) & = & x\,\vec{P}\,F_l^*(Q)\,\vec{R} & \text{if } \vec{P} \in \mathrm{NF}_{\beta^*}, Q \notin \mathrm{NF}_{\beta^*} \\
F_l^*(\lambda x.P) & = & \lambda x.F_l^*(P) & \\
F_l^*((\lambda^* x.P)\,Q\,\vec{R}) & = & P\{x := Q\}\,\vec{R}. &
\end{array}$$

We write $M \to_{l^*} N$ if $M \notin \mathrm{NF}_{\beta^*}$ and $F_l^*(M) = N$. More specifically, if $M \equiv C[(\underline{\lambda}x.P)\;Q]$ and $C[P\{x := Q\}] \equiv N$ we write $M \to_{\underline{l}} N$, and if $M \equiv C[(\lambda x.P)\;Q]$ and $C[P\{x := Q\}] \equiv N$ we write $M \to_l N$.

1.7.39. Lemma. *For all $M \in \underline{\Lambda}_K$: $|F_l^*(M)| = F_l(|M|)$.*

Proof. By induction on M. □

1.7.40. Lemma. *Let $M \in \underline{\Lambda}_K$.*

$$\begin{tikzcd}
M \arrow[r, "\to_{\underline{l}}"'] \arrow[d, "\varphi"'] & N \arrow[d, "\varphi"] \\
K \arrow[r, dotted, "\to_l"'] & L
\end{tikzcd}$$

Proof. By induction on M. □

We prove the contrapositive of the normalization theorem: if the left-most reduction path from M does not terminate, then no reduction path does. For this it suffices to show the following result, very similar to the conservation theorems seen earlier—this explains why the technique of the previous subsections is useful.

1.7.41. Theorem. *If $M \in \Lambda_K$ and $M \to_\beta N$, then*

$$M \in \infty_l \;\Rightarrow\; N \in \infty_l.$$

Proof. Let $M \equiv C[(\lambda x.P)\ Q] \to_\beta C[P\{x := Q\}] \equiv N$. Suppose $M \in \infty_l$, i.e.

$$M \equiv M_0 \to_l M_1 \to_l M_2 \to_l \dots .$$

Let $L_0 = C[(\underline{\lambda} x.P)\ Q]$, and $N_0 \equiv N$. By Lemma 1.7.5, 1.7.6, 1.7.39, and 1.7.40, we can erect the diagram:

$$\begin{tikzcd}
M_0 \arrow[rr, "\to_l"'] \arrow[dd, "\to_\beta"'] & & M_1 \arrow[rr, "\to_l"'] & & M_2 \arrow[rr, "\to_l"'] & & \cdots \\
& L_0 \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_l"'] & & L_1 \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_{\underline{l}}"'] & & L_2 \arrow[ul, "|\bullet|"'] \arrow[dl, "\varphi"] \arrow[rr, "\to_l"'] & & \cdots \\
N_0 \arrow[rr, "\to_l"'] & & N_1 \arrow[rr, "\equiv"'] & & N_2 \arrow[rr, "\to_l"'] & & \cdots
\end{tikzcd}$$

where

$$\begin{array}{lcl} L_i \to_l L_{i+1} & \Rightarrow & N_i \to_l N_{i+1} \\ L_i \to_{\underline{l}} L_{i+1} & \Rightarrow & N_i \equiv N_{i+1}. \end{array}$$

By finiteness of developments, $L_i \to_l L_{i+1}$, for infinitely many i, giving an infinite left-most reduction path from N_0. □

1.7.42. Corollary (Normalization theorem). *F_l is normalizing.*

Proof. Suppose $M \in \mathrm{WN}_\beta$, i.e., $M \twoheadrightarrow_\beta N \in \mathrm{NF}_\beta$. If M had an infinite leftmost reduction, then by Theorem 1.7.41, so did N, a contradiction. □

1.7.43. Definition. Let $M \in \Lambda_K$. A finite or infinite reduction path

$$M_0 \to_\beta M_1 \to_\beta M_2 \to_\beta \dots$$

is *quasi-leftmost* if it is finite or for all $i \in \mathbb{N}$ there is $j > i$ with $M_j \to_l M_{j+1}$.

1.7.44. Corollary. *If $M \in \mathrm{WN}_\beta$, then any quasi-leftmost reduction from M is finite.*

Proof. First show as in Theorem 1.7.41 that if $M \to_\beta N$ and M has an infinite quasi-leftmost reduction, then so does N. Then proceed as in Corollary 1.7.42. □

The normalization theorem is due to Curry and Feys [29]. Barendregt [3] infers the normalization theorem from the standardization theorem, and uses both of these theorems to prove normalization of quasi-leftmost reductions.

Barendregt et al. [6] define a β-redex Δ to be *needed* in a term M, if Δ (or a residual of Δ) is contracted in every reduction of M to normal form. They then show that every term not in normal-form has at least one needed redex, and that a reduction strategy that contracts only needed redexes is normalizing. They also show that it is undecidable, in general, whether a redex is needed in a term; however, the left-most redex is always needed, and this yields another proof of the normalization theorem. Similar results were shown by Huet and Lévy [54] in their early study of neededness in the context of orthogonal term rewriting systems, and much has been done since in various contexts—see [73] for references to some paper‘s. Similar results were discovered independently by Khasidashvili [68] (see also [70, 72]); in particular, the proof of Theorem 1.7.41 can be viewed as a special case of a proof due to Khasidashvili [68].

For more on normalization, see [76, 105].

1.7.7. Conservation from normalization

In this last subsection we give a very short proof of the conservation theorem for Λ_I, using the fact that F_∞ is perpetual and F_l is normalizing.

1.7.45. Lemma (Regnier [108]). *For all $M \in \Lambda_I$, $F_l(M) = F_\infty(M)$.*

Proof. If $\lambda x.P \subseteq M \in \Lambda_I$, then $x \in \mathrm{FV}(P)$. □

1.7.46. Corollary.

(i) *For all $M \in \Lambda_I$, $M \in \mathrm{WN}_\beta \Leftrightarrow M \in \mathrm{SN}_\beta$.*

(ii) *For all $M \in \Lambda_I$, $M \in \infty_\beta$ & $M \to_\beta N \Rightarrow N \in \infty_\beta$.*

Proof.

(i) Since F_∞ is perpetual and F_l is normalizing, Lemma 1.7.45 implies:

$$M \in \mathrm{WN}_\beta \Leftrightarrow \exists n : F_l^n(M) \in \mathrm{NF}_\beta \Leftrightarrow \exists n : F_\infty^n(M) \in \mathrm{NF}_\beta \Leftrightarrow M \in \mathrm{SN}_\beta.$$

(ii) Suppose $M \to_\beta N$. If $M \in \infty_\beta$, then by (i), $M \notin \mathrm{WN}_\beta$. Hence $N \notin \mathrm{WN}_\beta$, in particular $N \in \infty_\beta$. □

1.7.47. Remark. The same technique can be used to prove that in Λ_ω (see Definition 1.4.3) all reduction paths have the same length: one proves directly that in Λ_ω, F_∞ is minimal. Since F_∞ is also maximal, the longest and shortest reduction path have the same length, and so *all* reduction paths have the same length.

1.7.48. REMARK. Not all strategies are maximal in Λ_I; for instance the strategy which always contracts the right-most redex is not maximal, as the example $(\lambda x.\lambda y.y\ x\ x)\ (\mathbf{I}\,\mathbf{I}) \twoheadrightarrow_l^3 \lambda y.y\ \mathbf{I}\ \mathbf{I}$ shows.

1.7.49. REMARK. A simpler proof of the above corollary, which does not use F_∞, can be obtained by proving directly that F_l is perpetual in Λ_I using the fundamental lemma of perpetuality, rather than inferring this from $F_l = F_\infty$ and perpetuality of F_∞. Slight variations of this technique are due to Curry and Feys [29] and to van Raamsdonk [105].

Barendregt et al. [6] show that leftmost reduction paths have maximal length among all reduction paths in which only needed redexes are contracted, and that in Λ_I all redexes are needed. This gives another proof that in Λ_I, F_l is maximal and thereby perpetual.

1.8. A note on shortest developments

As mentioned above, de Vrijer [135] presents a proof of the *finite developments* theorem which, in addition to showing that all developments are finite, gives an effective reduction strategy computing longest developments as well as a simple formula computing the length of such longest developments.

We now show that by applying a rather simple and intuitive principle of duality to de Vrijer's approach one arrives at a proof that *some* developments are finite which in addition yields an effective reduction strategy computing shortest developments as well as a simple formula computing the length of such shortest developments. The duality fails for general β-reduction.

Our results simplify previous work by Khasidashvili [68].

1.8.1. Shortest developments

We first present our technique for computing shortest developments and then explain the relation to de Vrijer's [135] technique afterwards.

1.8.1. DEFINITION.

(i) For all $x \in V$ define $m_x : \underline{\Lambda}_K \to \mathbb{N}$ by:[10]

$$
\begin{array}{lcll}
m_x(x) & = & 1 & \\
m_x(y) & = & 0 & \text{if } x \not\equiv y \\
m_x((\underline{\lambda} y.P)\ Q) & = & m_x(P) + m_x(Q)\lfloor m_y(P), 1 \rfloor & \\
m_x(P\ Q) & = & m_x(P) + m_x(Q) & \text{if } P \not\equiv \underline{\lambda} y.R \\
m_x(\lambda y.P) & = & m_x(P). &
\end{array}
$$

[10] $\lfloor m, n \rfloor$ and $\lceil m, n \rceil$ denote the minimum and maximum of m and n, respectively.

(ii) Define $h : \underline{\Lambda}_K \to \mathbb{N}$ by:

$$\begin{array}{lcll} h(x) & = & 0 & \\ h((\underline{\lambda} y.P)\, Q) & = & h(P) + h(Q)\lfloor m_y(P), 1 \rfloor + 1 & \\ h(P\, Q) & = & h(P) + h(Q) & \text{if } P \not\equiv \underline{\lambda} y.R \\ h(\lambda y.P) & = & h(P). & \end{array}$$

(iii) Define the strategy $H : \underline{\Lambda}_K \to \underline{\Lambda}_K$ by:

$$\begin{array}{lcl} H(x) & = & x \\ H((\underline{\lambda} y.P)\, Q) & = & \left\{ \begin{array}{ll} (\underline{\lambda} y.P)\, H(Q) & \text{if } \lfloor m_y(P), 1 \rfloor = 1 \ \&\ Q \notin \mathrm{NF}_{\underline{\beta}} \\ P\{y := Q\} & \text{otherwise} \end{array} \right. \\ H(P\, Q) & = & \left\{ \begin{array}{ll} H(P)\, Q & \text{if } P \not\equiv \underline{\lambda} y.R \ \&\ P \notin \mathrm{NF}_{\underline{\beta}} \\ P\, H(Q) & \text{if } P \not\equiv \underline{\lambda} y.R \ \&\ P \in \mathrm{NF}_{\underline{\beta}} \end{array} \right. \\ H(\lambda y.P) & = & \lambda y.H(P). \end{array}$$

As will be seen below, $M \to_{\underline{\beta}} H(M) \to_{\underline{\beta}} H(H(M)) \to_{\underline{\beta}} \ldots$ is a shortest complete development from M, and $h(M)$ is the length of this development. Corollary 1.8.8 expresses this succinctly as: $L_H(M) = s_{\underline{\beta}}(M) = h(M)$.

1.8.2. Remark.

(i) $x \notin \mathrm{FV}(M) \Rightarrow m_x(M) = 0$.

(ii) $M \in \mathrm{NF}_{\underline{\beta}} \Leftrightarrow h(M) = 0$.

(iii) $\lfloor m_y(P), 1 \rfloor \neq 1 \Rightarrow \lfloor m_y(P), 1 \rfloor = m_y(P)$.

1.8.3. Lemma. *Let $x \not\equiv y$. Then:*

(i) $m_y(M\{x := N\}) = m_y(M) + m_y(N) m_x(M)$.

(ii) $h(M\{x := N\}) = h(M) + h(N) m_x(M)$.

Proof. (i) is by induction on M. Let $L^* \equiv L\{x := N\}$.

1. $M \equiv z$.

   1.1. $z \equiv x$. Then

$$\begin{array}{rcl} m_y(x^*) & = & m_y(N) \\ & = & m_y(x) + m_y(N) m_x(x). \end{array}$$

   1.2. $z \not\equiv x$. Then

$$\begin{array}{rcl} m_y(z^*) & = & m_y(z) \\ & = & m_y(z) + m_y(N) m_x(z). \end{array}$$

2. $M \equiv (\underline{\lambda} z.P)\,Q$. Since $z \notin \mathrm{FV}(N)$, also $m_z(N) = 0$. Therefore, by the induction hypothesis,

$$\begin{aligned}
& m_y((\underline{\lambda} z.P^*)\,Q^*) \\
&= m_y(P^*) + m_y(Q^*)\lfloor m_z(P^*), 1\rfloor \\
&= m_y(P) + m_y(N)m_x(P) + (m_y(Q) + m_y(N)m_x(Q))\lfloor m_z(P), 1\rfloor \\
&= m_y(P) + m_y(N)m_x(P) + m_y(Q)\lfloor m_z(P), 1\rfloor + m_y(N)m_x(Q)\lfloor m_z(P), 1\rfloor \\
&= m_y(P) + m_y(Q)\lfloor m_z(P), 1\rfloor + m_y(N)(m_x(P) + m_x(Q)\lfloor m_z(P), 1\rfloor) \\
&= m_y((\underline{\lambda} z.P)\,Q) + m_y(N)m_x((\underline{\lambda} z.P)\,Q).
\end{aligned}$$

3. $M \equiv P\,Q$ where $P \not\equiv \underline{\lambda} y.R$. Then, by the induction hypothesis,

$$\begin{aligned}
m_y(P^*\,Q^*) &= m_y(P^*) + m_y(Q^*) \\
&= m_y(P) + m_y(N)m_x(P) + m_y(Q) + m_y(N)m_x(Q) \\
&= m_y(P\,Q) + m_y(N)m_x(P\,Q).
\end{aligned}$$

4. $M \equiv \lambda y.P$. Similar to Case 3.

This concludes the proof of (i); (ii) is also by induction on M.

1. $M \equiv z$.

   1.1. $z \equiv x$. Then

$$\begin{aligned}
h(x^*) &= h(N) \\
&= h(x) + h(N)m_x(x).
\end{aligned}$$

   1.2. $z \not\equiv x$. Then

$$\begin{aligned}
h(z^*) &= h(z) \\
&= h(z) + h(N)m_x(z).
\end{aligned}$$

2. $M \equiv (\underline{\lambda} z.P)\,Q$. Since $z \notin \mathrm{FV}(N)$, also $m_z(N) = 0$. Therefore, by the induction hypothesis and (i),

$$\begin{aligned}
& h((\underline{\lambda} z.P^*)\,Q^*) \\
= \;& h(P^*) + h(Q^*)\lfloor m_z(P^*), 1\rfloor + 1 \\
= \;& h(P) + h(N)m_x(P) + (h(Q) + h(N)m_x(Q))\lfloor m_z(P), 1\rfloor + 1 \\
= \;& h(P) + h(N)m_x(P) + h(Q)\lfloor m_z(P), 1\rfloor + h(N)m_x(Q)\lfloor m_z(P), 1\rfloor + 1 \\
= \;& h(P) + h(Q)\lfloor m_z(P), 1\rfloor + 1 + h(N)(m_x(P) + m_x(Q)\lfloor m_z(P), 1\rfloor) \\
= \;& h((\underline{\lambda} z.P)\,Q) + h(N)m_x((\underline{\lambda} z.P)\,Q).
\end{aligned}$$

3. $M \equiv P\,Q$ where $P \not\equiv \underline{\lambda} y.R$. Then, by the induction hypothesis,

$$\begin{aligned}
h(P^*\,Q^*) &= h(P^*) + h(Q^*) \\
&= h(P) + h(N)m_x(P) + h(Q) + h(N)m_x(Q) \\
&= h(P\,Q) + h(N)m_x(P\,Q).
\end{aligned}$$

4. $M \equiv \lambda y.P$. Similar to Case 3. $\square$

1.8.4. Lemma. *Suppose that $M \to_{\underline{\beta}} N$. Then*

(i) $m_x(M) \leq m_x(N)$.

(ii) $h(M) \leq h(N) + 1$.

Proof. (i) is by induction on $M \to_{\underline{\beta}} N$.

1. $M \equiv (\underline{\lambda} y.P)\, Q \to_{\underline{\beta}} P\{y := Q\} \equiv N$. By Lemma 1.8.3,

$$\begin{array}{rcl} m_x((\underline{\lambda} y.P)\, Q) & = & m_x(P) + m_x(Q)\lfloor m_y(P), 1\rfloor \\ & \leq & m_x(P) + m_x(Q) m_y(P) \\ & = & m_x(P\{y := Q\}). \end{array}$$

2. $M \equiv (\underline{\lambda} y.P)\, Q \to_{\underline{\beta}} (\underline{\lambda} y.P')\, Q' \equiv N$, where $P \to_{\underline{\beta}} P'$ and $Q \equiv Q'$, or vice versa. By the induction hypothesis,

$$\begin{array}{rcl} m_x((\underline{\lambda} y.P)\, Q) & = & m_x(P) + m_x(Q)\lfloor m_y(P), 1\rfloor \\ & \leq & m_x(P') + m_x(Q')\lfloor m_y(P'), 1\rfloor \\ & = & m_x((\underline{\lambda} y.P')\, Q'). \end{array}$$

3. $M \equiv P\, Q \to_{\underline{\beta}} P'\, Q' \equiv N$, where $P \not\equiv \underline{\lambda} y.R$, and where $P \to_{\underline{\beta}} P'$ and $Q \equiv Q'$, or vice versa. Similar to Case 2.

4. $M \equiv \lambda y.P \to_{\underline{\beta}} \lambda y.P' \equiv N$, where $P \to_{\underline{\beta}} P'$. Similar to Case 2.

This concludes (i); (ii) is also by induction on $M \to_{\underline{\beta}} N$.

1. $M \equiv (\underline{\lambda} y.P)\, Q \to_{\underline{\beta}} P\{y := Q\} \equiv N$. By Lemma 1.8.3

$$\begin{array}{rcl} h((\underline{\lambda} y.P)\, Q) & = & h(P) + h(Q)\lfloor m_y(P), 1\rfloor + 1 \\ & \leq & h(P) + h(Q) m_y(P) + 1 \\ & = & h(P\{y := Q\}) + 1. \end{array}$$

2. $M \equiv (\underline{\lambda} y.P)\, Q \to_{\underline{\beta}} (\underline{\lambda} y.P')\, Q' \equiv N$, where $P \to_{\underline{\beta}} P'$ and $Q \equiv Q'$, or vice versa. By the induction hypothesis and (i),

$$\begin{array}{rcl} h((\underline{\lambda} y.P)\, Q) & = & h(P) + h(Q)\lfloor m_y(P), 1\rfloor + 1 \\ & \leq & h(P') + h(Q')\lfloor m_y(P'), 1\rfloor + 2 \\ & = & h((\underline{\lambda} y.P')\, Q') + 1. \end{array}$$

3. $M \equiv P\, Q \to_{\underline{\beta}} P'\, Q' \equiv N$, where $P \not\equiv \underline{\lambda} y.R$, and where $P \to_{\underline{\beta}} P'$ and $Q \equiv Q'$, or vice versa. Similar to Case 2.

4. $M \equiv \lambda y.P \to_{\underline{\beta}} \lambda y.P' \equiv N$, where $P \to_{\underline{\beta}} P'$. Similar to Case 2. □

1.8.5. Corollary. *For all $M \in \underline{\Lambda}_K$: $h(M) \leq s_{\underline{\beta}}(M)$.*

Proof. By induction on $h(M)$.

1. $h(M) = 0$. Then $M \in \mathrm{NF}_{\underline{\beta}}$, and then $s_{\underline{\beta}}(M) = 0$.
2. $h(M) \neq 0$. Then $M \notin \mathrm{NF}_{\underline{\beta}}$. Let $M \to_{\underline{\beta}} N$ be such that $s_{\underline{\beta}}(M) = s_{\underline{\beta}}(N) + 1$. By Lemma 1.8.4(ii) and the induction hypothesis,

$$\begin{array}{rcl} h(M) & \leq & h(N) + 1 \\ & \leq & s_{\underline{\beta}}(N) + 1 \\ & = & s_{\underline{\beta}}(M). \end{array}$$

□

1.8.6. Lemma. *If $h(M) \neq 0$ then $M \to_{\underline{\beta}} H(M)$ and $h(M) = h(H(M)) + 1$.*

Proof. By induction on M. Assume $h(M) \neq 0$.

1. $M \equiv x$. This case is impossible since $h(x) = 0$.
2. $M \equiv (\underline{\lambda} y.P)\, Q$.

2.1. $\lfloor m_y(P), 1 \rfloor = 1$ and $Q \notin \mathrm{NF}_{\underline{\beta}}$. By the induction hypothesis,

$$\begin{array}{rcl} h((\underline{\lambda} y.P)\, Q) & = & h(P) + h(Q)\lfloor m_y(P), 1\rfloor + 1 \\ & = & h(P) + h(Q) + 1 \\ & = & h(P) + h(H(Q)) + 2 \\ & = & h(P) + h(H(Q))\lfloor m_y(P), 1\rfloor + 2 \\ & = & h((\underline{\lambda} y.P)\, H(Q)) + 1 \\ & = & h(H((\underline{\lambda} y.P)\, Q)) + 1. \end{array}$$

2.2. $\lfloor m_y(P), 1 \rfloor \neq 1$ or $Q \in \mathrm{NF}_{\underline{\beta}}$. By Lemma 1.8.3

$$\begin{array}{rcl} h((\underline{\lambda} y.P)\, Q) & = & h(P) + h(Q)\lfloor m_y(P), 1\rfloor + 1 \\ & = & h(P) + h(Q) m_y(P) + 1 \\ & = & h(P\{y := Q\}) + 1. \end{array}$$

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

$$\begin{array}{rcl} h(\lambda y.P) & = & h(P) \\ & = & h(H(P)) + 1 \\ & = & h(\lambda y.H(P)) + 1 \\ & = & h(H(\lambda y.P)) + 1. \end{array}$$

4. $M \equiv P\, Q$. Similar to Case 3. □

1.8.7. Corollary. *For all $M \in \underline{\Lambda}_K$: $h(M) = L_H(M)$.*

Proof. By induction on $h(M)$.

1. $h(M) = 0$. Then $M \in \mathrm{NF}_{\underline{\beta}}$, and then $L_H(M) = 0$.

2. $h(M) \neq 0$. Then $M \notin \mathrm{NF}_{\underline{\beta}}$, and then by Lemma 1.8.6 and the induction hypothesis,
$$\begin{array}{rcl} h(M) & = & h(H(M)) + 1 \\ & = & L_H(H(M)) + 1 \\ & = & L_H(M). \end{array}$$ □

1.8.8. COROLLARY. *For all $M \in \underline{\Lambda}_K$: $h(M) = s_{\underline{\beta}}(M) = L_H(M)$.*

PROOF. Let $M \in \underline{\Lambda}_K$. Obviously, $s_{\underline{\beta}}(M) \leq L_H(M)$. By Corollary 1.8.5 and 1.8.7,
$$s_{\underline{\beta}}(M) \leq L_H(M) = h(M) \leq s_{\underline{\beta}}(M).$$ □

1.8.2. Relation to Khasidashvili's technique

Khasidashvili [68] calls a redex Δ in M *essential*, notation $E(\Delta, M)$, if every complete development of M must reduce Δ (or a residual of Δ). He shows that any strategy which reduces in each step an inner-most essential redex yields shortest complete developments, and he gives a formula for the length of such developments: the number of essential redexes in the initial term. He also gives an algorithm to decide whether a redex in a term is essential; this makes the above strategy and formula effective, but the algorithm is—in our opinion—somewhat involved. The algorithm can be simpler formulated in terms of the map m_y as follows:

$$\begin{array}{lcl} E(\Delta, (\underline{\lambda} y.P)\, Q) & \Leftrightarrow & \Delta \equiv (\lambda y.P)\, Q \text{ or } E(\Delta, P) \text{ or } [E(\Delta, Q)\ \&\ m_y(P) > 0] \\ E(\Delta, P\, Q) & \Leftrightarrow & E(\Delta, P) \text{ or } E(\Delta, Q) \\ E(\Delta, \underline{\lambda} y.P) & \Leftrightarrow & E(\Delta, P). \end{array}$$

In this terminology, the map h counts the number of essential redexes in a term, and H reduces an essential redex that is not contained in the argument of another essential redex.

1.8.3. Relation to de Vrijer's technique

De Vrijer [135] studies the following maps n_x, g, and G, which arise from m_x, h, and H by replacing all minimum operators $\lfloor \bullet, \bullet \rfloor$ by maximum operators $\lceil \bullet, \bullet \rceil$; intuitively this makes sense since we now consider longest instead of shortest developments.

(i) For all $x \in V$ define $n_x : \underline{\Lambda}_K \to \mathbb{N}$ by:

$$\begin{array}{lcll} n_x(x) & = & 1 & \\ n_x(y) & = & 0 & \text{if } x \not\equiv y \\ n_x((\underline{\lambda} y.P)\, Q) & = & n_x(P) + n_x(Q) \lceil n_y(P), 1 \rceil & \\ n_x(P\, Q) & = & n_x(P) + n_x(Q) & \text{if } P \not\equiv \underline{\lambda} y.R \\ n_x(\lambda y.P) & = & n_x(P). & \end{array}$$

(ii) Define $g : \underline{\Lambda}_K \to \mathbb{N}$ by:

$$\begin{array}{lcll} g(x) & = & 0 & \\ g((\underline{\lambda} y.P)\, Q) & = & g(P) + g(Q)\lceil n_y(P), 1\rceil + 1 & \\ g(P\, Q) & = & g(P) + g(Q) & \text{if } P \not\equiv \underline{\lambda} y.R \\ g(\lambda y.P) & = & g(P). & \end{array}$$

(iii) Define the strategy $G : \underline{\Lambda}_K \to \underline{\Lambda}_K$ by:

$$\begin{array}{lcl} G(x) & = & x \\ G((\underline{\lambda} y.P)\, Q) & = & \begin{cases} (\underline{\lambda} y.P)\, G(Q) & \text{if } \lceil n_y(P), 1\rceil = 1 \ \&\ Q \notin \mathrm{NF}_{\underline{\beta}} \\ P\{y := Q\} & \text{otherwise.} \end{cases} \\ G(P\, Q) & = & \begin{cases} G(P)\, Q & \text{if } P \not\equiv \underline{\lambda} y.R \ \&\ P \notin \mathrm{NF}_{\underline{\beta}} \\ P\, G(Q) & \text{if } P \not\equiv \underline{\lambda} y.R \ \&\ P \in \mathrm{NF}_{\underline{\beta}} \end{cases} \\ G(\lambda y.P) & = & \lambda y.G(P). \end{array}$$

De Vrijer proves that $M \to_{\underline{\beta}} G(M) \to_{\underline{\beta}} G(G(M)) \to_{\underline{\beta}} \ldots$ is a longest complete development from M, and that $g(M)$ is the length of this development. This is expressed by the equations: $L_G(M) = l_{\underline{\beta}}(M) = g(M)$. The finite developments theorem is an immediate corollary.

The proof of these equations can be carried out *exactly* as in 1.8.2–1.8.8 by replacing $s_{\underline{\beta}}$, $\lfloor \bullet, \bullet \rfloor$, $\leq$, m_x, h, and L_H by $l_{\underline{\beta}}$, $\lceil \bullet, \bullet \rceil$, $\geq$, n_x, g, and L_G, respectively! This works because the properties used in 1.8.2–1.8.8 involving $s_{\underline{\beta}}, m_x$, etc. are invariant under the transformation, as the reader is encouraged to check.[11] For instance, the property $\lfloor m, n \rfloor \leq m$ becomes $\lceil m, n \rceil \geq m$.

1.8.4. Discussion

Although the general notions of longest and shortest complete β-reduction sequences are intuitively "opposite," they are, technically speaking, very different. For instance, there is an effective reduction strategy that computes longest complete β-reduction sequences (see [119] among others), but no effective reduction strategy that computes shortest complete β-reduction sequences [3]. In contrast, the above shows that one can effectively compute both shortest and longest complete developments, and the proofs reveal an amazing duality between the two concepts. It is natural to ask why the duality does not carry over to the general case of β-reduction.

The difference between the minimal strategy H and the maximal strategy G is revealed on terms of form $(\underline{\lambda} y.P)\, Q$ where $Q \notin \mathrm{NF}_{\underline{\beta}}$. The rationale behind the minimal strategy is that if any reduction of $(\underline{\lambda} y.P)\, Q$ to $\underline{\beta}$-normal form must reduce inside at least one residual of Q, then it is best to

[11] To obtain this result, a small change has been made to G as compared to de Vrijer's formulation; in his formulation the condition $\lceil n_y(P), 1\rceil = 1$ is $n_y(P) = 0$—see the last subsection.

perform reductions in Q first, to avoid proliferation. This is decidable for developments, but undecidable for β-reduction [6].

The rationale behind the maximal strategy is that if any reduction of $(\underline{\lambda} y.P)Q$ to $\underline{\beta}$-normal form may reduce inside at most one residual of Q, then it is best to perform reductions in Q first, to avoid erasing. An equivalent technique, used by de Vrijer [135], is to test whether reducing $(\underline{\lambda} y.P)\, Q$ one step would delete Q, and if so reduce Q to normal form first. This is decidable for developments as well as for β-reduction.

From the point of view of efficiency, a minimal strategy is clearly better than a maximal strategy. It is a remarkable fact that in general β-reductions we can effectively do the worst possible job, but not the best possible job.

CHAPTER 2

Weak and Strong Normalization in Type Theory

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 chapter 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 not only simplifies the task of proving strong normalization as compared to previous approaches, but also suggests an approach to an open problem in type theory, pursued in the next chapter.

2.1. Introduction

As mentioned in the 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 even when known to be true.

The classical proof of strong normalization for β-reduction in simply typed λ-calculus is by a method due to Tait [125]. It was generalized to second-order typed λ-calculus by Girard [41], and subsequently simplified by Tait [126]. It has since been generalized to a variety of λ-calculi—see [4, 34, 42, 50, 80, 132]. A version of the proof is also presented in Section 1.5.

[1] Reduction on terms in typed λ-calculi is closely related to reduction on derivations in natural deduction logics via the Curry-Howard isomorphism [29, 52]. This will be implicit in the rest of the chapter.

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 [35] 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 [104] independently uses the same technique to prove weak normalization for reduction of natural deduction derivations in predicate logic.

Nederpelt [92], Klop [76], Khasidashvili [69], Karr [62], de Groote [31], and Kfoury and Wells [66] 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 [67] 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 [4] weak normalization of β-reduction implies strong normalization of β-reduction. The conjecture has also been mentioned by Geuvers [38], and, in a less concrete form, by Klop.

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

Section 2.2 presents Klop's technique, which is based on the conservation theorem for λI and an interpretation of λK in λI. Section 2.3 analyzes the relationship to the similar techniques by Nederpelt and others. Section 2.4 presents our extension of Klop's technique, which is based on a continuation passing style translation. Section 2.5 shows that the continuation passing style translation is a special case of a class of translations, which we call *permutative inner interpretations,* each of which give rise to a similar extension of Klop's technique. The versatility of our approach is demonstrated by application to some typed λ-calculi in Section 2.6 and 2.7. These systems include *second-order λ-calculus* and the system of *positive, recursive types.* Section 2.8 concludes and reviews directions for further work.

2.1.1. Preliminaries

The following is explained in more detail in [3].

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, $\rightarrow_R$ 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. We use $\Rightarrow$, $\Leftrightarrow$, and & to denote the obvious connectives in the meta-language.

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

2.1.2. DEFINITION. A finite or infinite sequence

$$M_0 \rightarrow_R M_1 \rightarrow_R \ldots$$

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.

2.1.3. DEFINITION. Define the following subsets of Λ_K:

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

2.1.4. 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.

2.1.5. DEFINITION. For $M \in \mathrm{SN}_R \cap \mathrm{CR}_R$, $\mathrm{nf}_R(M)$ is the unique $N \in \mathrm{NF}_R$ satisfying $M \twoheadrightarrow_R N$.

2.2. Klop's technique

This section presents Klop's technique [76, 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 2.3. The second subsection proves a result that will be used in our extension in Section 2.4.

2.2.1. The idea: non-erasing reductions

2.2.1. 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.

2.2.2. Theorem (Klop [76]). *For all* $M \in \Lambda_K$,

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

2.2.3. Remark. Klop's proof of Theorem 2.2.2 is in two steps:

$$\iota(M) \in \mathrm{WN}_{\beta\pi} \;\Rightarrow\; \iota(M) \in \mathrm{SN}_{\beta\pi} \;\Rightarrow\; M \in \mathrm{SN}_\beta. \tag{2.1}$$

The first implication is a special case of Klop's conservation theorem [76] for *definable extensions* of Λ_I, and the second one is proved by the implications:

$$\iota(M) \in \mathrm{SN}_{\beta\pi} \;\Rightarrow\; \iota(M) \in \mathrm{SN}_{\beta\pi\kappa} \;\Rightarrow\; \iota(M) \in \mathrm{SN}_{\beta\kappa} \;\Rightarrow\; M \in \mathrm{SN}_\beta. \tag{2.2}$$

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 $\iota(M) \twoheadrightarrow_\kappa M$.

2.2.2. Proof of part of Klop's result

In Section 2.4 our extension uses the second implication of (2.1), which we therefore prove now. The proof follows the structure of (2.2).

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

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

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

$$M\{x := L\} \to_\kappa N\{x := L\} \tag{2.3}$$

and

$$L\{x := M\} \twoheadrightarrow_\kappa L\{x := N\} \tag{2.4}$$

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

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

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

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

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

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

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 (2.3)-(2.4),

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

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 where $Q_m \equiv R_m$ 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

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

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

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

This exhausts all possibilities. $\square$

2.2.5. 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 2.2.4 $k-(m-1)$ 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. □

2.2.6. LEMMA. *For all* $M \in \Lambda_K^\pi$,

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

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

2.2.7. MAIN LEMMA (Klop [76]). *For all* $M \in \Lambda_K$,

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

PROOF. By Lemmas 2.2.5 and 2.2.6. □

2.3. Variations on Klop's technique

Klop's technique [76] was inspired by Nederpelt's [92] technique, and is also related to the later techniques by Khasidashvili [69], Karr [62], de Groote [31], and Kfoury and Wells [66]. 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 [76, II.4]. For more on the relationship between de Groote's and Kfoury and Wells' technique, see [67]. The notions of reduction discussed in this section have been considered in a number of other contexts [2, 63, 64, 65, 90, 108, 112, 134]—see [67] for a survey.

2.3.1. The technique by de Groote

This subsection presents de Groote's [31] technique to reduce strong normalization for the systems in the λ-cube [4] 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.

2.3.1. 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.

2.3.2. THEOREM (de Groote [31]). *For all $M \in \Lambda_K$,*

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

2.3.3. REMARK. The proof of Theorem 2.3.2 by de Groote is in two parts:

$$M \in \mathrm{WN}_{\beta_I\beta_S} \;\Rightarrow\; M \in \mathrm{SN}_{\beta_I\beta_S} \;\Rightarrow\; M \in \mathrm{SN}_\beta. \tag{2.5}$$

The *first part* of (2.5) 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

$$M \equiv M_0 \to_{\beta_I\beta_S} M_1 \to_{\beta_I\beta_S} \dots,$$

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:

$$\mathrm{INC}_{\beta_I\beta_S} \text{ and } \mathrm{CR}_{\beta_I\beta_S} \text{ and } M \in \mathrm{WN}_{\beta_I\beta_S} \;\Rightarrow\; M \in \mathrm{SN}_{\beta_I\beta_S}. \tag{2.6}$$

The *second part* of (2.5) is proved by the implications:

$$M \in \mathrm{SN}_{\beta_I\beta_S} \;\Rightarrow\; M \in \mathrm{SN}_{\beta_I\beta_S\beta_K} \;\Rightarrow\; M \in \mathrm{SN}_{\beta_I\beta_K} \;\Rightarrow\; M \in \mathrm{SN}_\beta. \tag{2.7}$$

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.

2.3.2. 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 2.2.2 and the proof of de Groote's Theorem 2.3.2. In both cases, the overall proof consists of two implications—(2.1) and (2.5)—see Remarks 2.2.3 and 2.3.3. Klop and de Groote prove the first implication in (2.1) and (2.5) differently, but de Groote's proof can be adapted to Klop's setting. As for the second implication in (2.1) and (2.5), the proof consists in both cases of three implications—(2.2) and (2.7). The first two implications in (2.2) and (2.7) 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 $\beta = \beta_I\beta_K$.

2.3.3. Nederpelt's technique

Nederpelt [92] proves β-strong normalization of all terms in a typed λ-calculus from the Automath family [30], 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 [20, 58, 59, 60], and their relevance to explicit substitution calculi [57, 61].

2.3.4. 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]

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

[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.

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

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

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 $(\lambda y.[])\,Q$.

2.3.5. THEOREM (Nederpelt [92]). *For all $M \in \Lambda_K$,*

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

2.3.6. REMARK. The proof structure is as (2.5)-(2.7) in Remark 2.3.3 with β_1 in place of $\beta_I\beta_S$ and β_2 in place of β_K.

2.3.4. Karr's technique

Karr [62] 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.

2.3.7. 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 $C[z] \twoheadrightarrow_{\beta_K} z$.

2.3.8. THEOREM (Karr [62]). *For all $M \in \Lambda_K$,*

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

2.3.9. REMARK. The proof is as (2.7) in Remark 2.3.3 with β_γ in place of β_S.

2.3.5. Kfoury and Wells' technique

Kfoury and Wells [66] 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.

2.3.10. 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 $M \rightarrow_{*} N \Leftrightarrow M \rightarrow_{\beta_I} M' \twoheadrightarrow_{\gamma} N \in \mathrm{NF}_{\gamma}$.

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

$$(\lambda y.\lambda x.P)\, Q\, R \rightarrow_{\gamma} (\lambda x.(\lambda y.P)\, Q)\, R.$$

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

2.3.11. THEOREM (Kfoury and Wells [66]). *For all* $M \in \Lambda_K$,

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

2.3.12. REMARK. Instead of proceeding as in (2.5)-(2.7) 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.

2.3.6. More general techniques by Klop and Khasidashvili

Klop [76, II.4] generalizes the technique from Section 2.2 to *regular combinatory reduction systems* (such systems are described in the survey [78]). 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 (2.5)-(2.7) in Remark 2.3.3 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 [69] 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 a more recent paper, Khasidashvili formulates his technique for so-called orthogonal expression reduction system [71]. The proof of the result is similar to Nederpelt's. As applications he obtains several theorems in the theory of perpetual reductions—see Chapter 1.

2.4. Extensions of Klop's technique

This section presents the main contribution of the chapter: 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 [141].

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

2.4.1. The idea: simulation of π

Theorem 2.2.2 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.

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

$$L \to_\beta K \quad \Rightarrow \quad \phi(L) \twoheadrightarrow_\beta^+ \phi(K) \tag{2.8}$$

$$L \to_\pi K \quad \Rightarrow \quad \phi(L) \twoheadrightarrow_\beta \phi(K). \tag{2.9}$$

2.4.2. 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 2.2.7, $M \in \mathrm{SN}_\beta$. □

So, the problem is to find ϕ. One approach, mentioned by Klop [76, 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 2.6).

Fortunately another technique is available. It is well-known [26, 112] that one can simulate reductions like π by means of a *continuation passing style* (CPS) translation [109, 100]. More precisely, there is a CPS translation $\psi : \Lambda_I^\pi \to \Lambda_I^\pi$ and an "optimizing" 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 $y M N$ 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.

2.4.3. 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 2.4.2, $M \in \mathrm{SN}_\beta$. $\square$

2.4.2. 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 [100]; see also [109].

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

(i) Define $\underline{\bullet} : \Lambda_K^\pi \to \Lambda_K$ 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 $\bullet\!:\!\bullet : \Lambda_K^\pi \times \Lambda_K \to \Lambda_K$ by:

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

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

The idea is to use Proposition 2.4.3 with $\psi(M) = \underline{M}$ and $\phi(M) = \underline{\underline{M}}$.

2.4.5. Lemma. *For all $M, N \in \Lambda_K^\pi$ 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. □

2.4.6. Lemma. *For all $M, N \in \Lambda_K^\pi$ 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 2.4.5(i),

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

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,

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

4. $M \equiv P\ Q$. Then, by the induction hypothesis and Lemma 2.4.5(ii),

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

The remaining case is similar to Case 3. □

2.4.7. 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 2.4.5(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 2.4.5(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 2.4.5(i) and 2.4.6,

$$\begin{array}{rcl} M : K & \equiv & (\lambda m.m\ \underline{\underline{Q}}\ K)\ \lambda x.\underline{\underline{P}} \\ & \to_\beta & (\lambda x.\underline{\underline{P}})\ \underline{\underline{Q}}\ K \\ & \twoheadrightarrow_\beta & \underline{\underline{P\{x := Q\}}}\ K \\ & \twoheadrightarrow_\beta & (P\{x := Q\}) : K. \end{array}$$

2. $M \equiv P\,Q \to_\beta P\,Q' \equiv N$, where $Q \to_\beta Q'$. Then, by the induction hypothesis and Lemma 2.4.5(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,

$$\begin{array}{rcl} M : K & \equiv & y\ (P : (\lambda m.m\ \underline{R}\ K))\ \underline{\underline{Q}} \\ & \equiv & y\ ((P\ R) : K)\ \underline{\underline{Q}} \\ & \equiv & N : K. \end{array}$$

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. □

2.4.8. THEOREM. *For all* $M \in \Lambda_K$

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

PROOF. By Proposition 2.4.3 and Lemma 2.4.7, 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 the translation in the corollary omits some η-redexes.

2.4.9. 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. 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 2.4.8. □

Xi [141] independently discovers Corollary 2.4.9 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 chapter 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 2.4.9 is very short, but—in our opinion—less transparent.

2.4.10. REMARK. Recall from Chapter 1 that 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 technique expressed by Corollary 2.4.9: 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 technique in the corollary, one may choose freely which path to prove finite, whereas in the technique based on perpetual reductions, one must prove that the path computed by F is finite.

2.4.11. 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.

2.4.12. 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 [112] which generalizes β_S can be simulated by a CPS translation [112], as was also noted in [67]. 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 $\underline{\iota(M)} \in \mathrm{WN}_\beta$ (neither set 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

2.5. Simulation by permutative inner interpretation

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 [112] 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 2.4 is a special case. The fourth subsection gives another special case due to Loader [85]. The last subsection explains the relation to the notion of an inner model.

2.5.1. Permutative inner interpretations

2.5.1. 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:

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

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

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

(i) The language $\mathcal{L}(I) \subseteq \Lambda_K$ *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 *sound* if, for all $X, Y, Z \in \mathcal{L}(I)$, 1-2 hold, and

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

2.5.4. 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 that $M \in \mathrm{SN}_\beta$, for all $M \in \Lambda_K$.

2.5.2. Simulations from permutative inner interpretations

The following is a convenient auxiliary notion.

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

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

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

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

2.5.7. 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

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

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. □

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

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

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

Proof.

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

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

In the remaining cases, apply the induction hypothesis.

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

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

In the remaining cases, use the induction hypothesis. □

2.5.9. Definition.

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

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

(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 $M \in \Lambda_L$.

2.5.10. 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 $M \in \Lambda_L$:

$$M \to_{\beta_1} N \Rightarrow \|M\|_l = \|N\|_l. \tag{2.10}$$

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

$$P\{k := Q\} \in \Lambda_L. \tag{2.11}$$

Finally prove (i) by induction on $M \to_{\beta_1} N$ using (2.10)-(2.11):

1. $M \equiv (\lambda k.P)\ Q \to_{\beta_1} P\{k := Q\} \equiv N$. Then, by (2.11), $N \in \Lambda_L$.
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 (2.10), $\|Q\|_k = 1$. 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 $M \in \Lambda_L$:

$$M \to_{\beta_1} N \;\Rightarrow\; \|M\| > \|N\|. \tag{2.12}$$

Now (ii) follows by (i) and (2.12).

(iii) By the technique due to Tait and Martin-Löf—see [3]. □

2.5.11. 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 2.5.10(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 $\|K\|_k = 1$,

$$\text{If } M \to_{\beta_1} N \quad \text{then} \quad M\{x := L\} \to_{\beta_1} N\{x := L\} \tag{2.13}$$
$$\text{If } M \to_{\beta_1} N \quad \text{then} \quad L\{x := M\} \twoheadrightarrow_{\beta_1} L\{x := N\} \tag{2.14}$$
$$\text{If } M \to_{\beta_0} N \quad \text{then} \quad M\{k := L\} \to_{\beta_0} N\{k := L\} \tag{2.15}$$
$$\text{If } M \to_{\beta_0} N \quad \text{then} \quad K\{k := M\} \to_{\beta_0} K\{k := N\}. \tag{2.16}$$

Here (2.13),(2.15) are by induction on $M \to_\beta N$, (2.14),(2.16) by induction on L.

We now proceed by induction on $M \to_\beta N$ using (2.13)-(2.16):

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 (2.13)-(2.14).
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 (2.15)-(2.16).

In the remaining cases, use the induction hypothesis. □

2.5.12. 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

$$
\begin{tikzcd}
M \arrow[rrr, "\to_\beta"'] \arrow[d, "\bullet"'] & & & N \arrow[d, "\bullet"] \\
\underline{M} \arrow[r, dotted, no head, "=_{\beta_1}"] \arrow[d, "\mathrm{nf}_{\beta_1}"'] & L \arrow[r, dotted, no head, "\to_{\beta_0}"] \arrow[dl, dotted, "\twoheadrightarrow_{\beta_1}"] & L' \arrow[r, dotted, "=_{\beta_1}"] \arrow[dl, dotted, "\twoheadrightarrow_{\beta_1}"] & \underline{N} \arrow[d, "\mathrm{nf}_{\beta_1}"] \\
M' \arrow[r, dotted, "\to_{\beta_0}"'] & O \arrow[rr, dotted, "\twoheadrightarrow_{\beta_1}"'] & & N'
\end{tikzcd}
\qquad
\begin{tikzcd}
M \arrow[r, "\to_\pi"'] \arrow[d, "\bullet"'] & N \arrow[d, "\bullet"] \\
\underline{M} \arrow[r, dotted, no head, "=_{\beta_1}"] \arrow[d, "\mathrm{nf}_{\beta_1}"'] & \underline{N} \arrow[d, "\mathrm{nf}_{\beta_1}"] \\
M' \arrow[r, dotted, no head, "\equiv"'] & N'
\end{tikzcd}
$$

by Lemmas 2.5.8, and 2.5.11. □

2.5.13. 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 2.4.3, Remark 2.5.6, and Lemma 2.5.12. □

2.5.3. CPS translation as a permutative inner interpretation

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

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

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. □

2.5.4. Loader's permutative inner interpretation

Loader [85] 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.

2.5.5. 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., [7].

2.5.15. 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:

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

2.5.16. 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. □

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.

2.5.17. Remark. Inner models are related to *term models* of the untyped λ-calculus—see [3].

2.6. Application to typed λ-calculi à la Curry

In this section we use the CPS translation from Section 2.4 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 $\lambda\!\rightarrow$, positive recursive types $\lambda\mu^+$, and subtypes $\lambda\!\subseteq$; see, e.g., [4, 133, 88], 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.

2.6.1. Typed λ-calculi à la Curry

2.6.1. 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.

2.6.2. 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 2.4.9, $M \in \mathrm{SN}_\beta$. □

It is well-known that various CPS translations preserve typability in various typed λ-calculi—see, e.g., [27, 44, 46, 81, 87].

2.6.2. Simple types

2.6.3. DEFINITION. The simply typed λ-calculus $\lambda{\rightarrow} = (\mathrm{Type}(\lambda{\rightarrow}), \vdash)$ is:

(i) $\mathrm{Type}(\lambda{\rightarrow})$ is defined by the grammar:

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

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

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

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

2.6.4. DEFINITION. Let $\bot$ be a fixed type, and $\neg\sigma \equiv \sigma \rightarrow \bot$. Define maps

$$[\bullet], [\bullet]' : \mathrm{Type}(\lambda{\rightarrow}) \rightarrow \mathrm{Type}(\lambda{\rightarrow})$$

by:

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

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

2.6.5. Convention. From now on we assume that the translation $[\bullet]$ from Section 2.4 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.

2.6.6. Lemma. *For all $M \in \Lambda_K$, $\sigma \in$ Type($\lambda\!\to$), $\Gamma \in$ Context(Type($\lambda\!\to$)),*

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

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

Proof. Induction on $\Gamma \vdash M : \sigma$ using Convention 2.6.5. $\square$

2.6.7. Corollary. $\lambda\!\to\models \mathrm{WN}_\beta \;\Rightarrow\; \lambda\!\to\models \mathrm{SN}_\beta$.

2.6.3. Positive, recursive types

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

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

where α occurs only *positively* in σ—see, e.g., [133]—and with the extra rule:

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

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

2.6.9. Definition. Define $[\bullet], [\bullet]' :$ Type($\lambda\mu^+$) $\to$ Type($\lambda\mu^+$) as for $\lambda\!\to$ and:

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

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

2.6.10. Lemma.

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

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

(iii) *If* $\Gamma \vdash M : \sigma$ *then* $\Delta, [\Gamma] \vdash [M] : [\sigma]$, *for some context* 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). $\square$

2.6.11. COROLLARY. $\lambda\mu^+ \models \mathrm{WN} \Rightarrow \lambda\mu^+ \models \mathrm{SN}$.

2.6.4. Subtypes

2.6.12. 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:

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

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

$$[b]' \;=\; b.$$

2.6.14. LEMMA.

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

(ii) *If* $\Gamma \vdash M : \sigma$ *then* $\Delta, [\Gamma] \vdash [M] : [\sigma]$, *for some context* Δ *with* $\mathrm{dom}(\Delta) = \mathrm{FV}([M]) \backslash \mathrm{FV}(M)$.

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). $\square$

2.6.15. COROLLARY. $\lambda\subseteq \models \mathrm{WN} \Rightarrow \lambda\subseteq \models \mathrm{SN}$.

2.6.5. Inner type interpretations in $\lambda\!\rightarrow$

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 $\lambda\!\rightarrow$. Each linear permutative inner interpretation satisfying the condition hence gives a technique to prove that weak normalization implies strong normalization in $\lambda\!\rightarrow$; similar conditions can be derived for other systems.

2.6.16. Definition.

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

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

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

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

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 $\sigma, \tau, \rho \in \mathrm{Type}(\lambda\!\rightarrow)$, there is a Δ such that

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

2.6.17. Proposition. *If an inner interpretation I agrees with an inner type interpretation T, then*

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

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

Proof. By induction on $\Gamma \vdash M : \sigma$ using Convention 2.6.5. □

2.6.18. Remark. Inner interpretations agreeing with inner type interpretations resemble *Kleisli triples* and *monads*—see, e.g., [90, 91, 33, 111].

2.7. 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 [14] variants of these calculi in which abstractions have form $\lambda x.M$ rather than $\lambda x{:}\sigma\,.\,M$. In the next chapter we show how the technique can be modified to the usual formulations of $\lambda 2$ and $\lambda\omega$.

2.7.1. 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 $\lambda\!\!\to$ 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

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

2.7.1. Second-order types

2.7.2. 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 $\mathrm{Term}(\lambda 2)$ is:

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

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

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

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

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

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

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

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

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

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

PROOF. Like the proof of Theorem 2.5.13. □

2.7.6. LEMMA.

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

(ii) *If* $\Gamma \vdash M : \sigma$ *then* $\Delta, [\Gamma] \vdash [M] : [\sigma]$, *for some context* Δ *with* $\mathrm{dom}(\Delta) = \mathrm{FV}([M]) \backslash \mathrm{FV}(M)$.

PROOF.

(i) Induction on σ.

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

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.

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

[4] A small technical difficulty appears in 2.7.4 and 2.7.10. 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—see also Discussion 3.4.1.

2.7.2. Higher-order types

2.7.8. 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:

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

(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 [38, 2.2.16]; see also [46].

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

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

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

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

2.7.11. THEOREM. $[M] \in \mathrm{WN}_\beta \Rightarrow M \in \mathrm{SN}_\beta$.

PROOF. Like the proof of Theorem 2.5.13. □

2.7.12. LEMMA.

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

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

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

(iv) *If* $\Gamma \vdash M : \sigma$ *then* $\Delta, [\Gamma] \vdash [M] : [\sigma]$, *for some context* Δ *with* $\mathrm{dom}(\Delta) = \mathrm{FV}([M]) \backslash \mathrm{FV}(M)$.

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.

2.7.13. 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.

2.8. 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 $\lambda\mu^+$ [133] and for the Curry systems $\lambda 2$ and $\lambda\cap^-$ [4] the straight-forward technique fails, because generalization and intersection introduction do not work under double negations.

We have also applied our extension to versions of $\lambda 2$ and $\lambda\omega$ à la Church. In the next chapter we generalize the technique to a class of calculi which includes more traditional formulations of $\lambda 2$ and $\lambda\omega$.

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., [27, 139]. 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 2.5, the inner interpretation must be sound, not just permutative.

CHAPTER 3

Normalization in Pure Type Systems

The *Barendregt-Geuvers-Klop* conjecture states that every weakly normalizing *pure type system* is also strongly normalizing—pure type systems are a general formalism of which specific type theories can be viewed as special cases. In this chapter, we show that the conjecture is true for the class of *generalized non-dependent* pure type systems, a class which includes, e.g., the left hand side of Barendregt's λ-*cube* as well as the system λU studied in the literature. This seems to be the first result giving a positive answer to the conjecture not merely for some concrete systems for which strong normalization is known to hold, but for a uniform *class* of systems in which not all systems are strongly normalizing.

3.1. Introduction

In Chapter 2 we reduced strong normalization to weak normalization in simply and second-order typed λ-calculus and in certain systems with subtypes and recursive types. For a *domain-free* [14] version of higher-order typed λ-calculus we also showed that strong normalization of all legal *objects* follows from weak normalization of all legal *objects,* but stated nothing about *constructors.* As mentioned in Chapter 2, Xi [141] independently uses the same technique to reduce strong normalization of simply and second-order typed λ-calculus to weak normalization of the same systems extended with certain pairing operators and type constants.

Each of the systems mentioned above is known to be strongly normalizing. Thus, for these systems, weak normalization trivially implies strong normalization. In this chapter we generalize the technique to the class of *generalized non-dependent* pure type systems—including the left hand side of the λ-cube as well as λU—and show that, for any system in the class, weak normalization implies strong normalization, provided the system satisfies certain technical properties (which are satisfied in the systems mentioned above). This seems to be the first result stating that the Barendregt-Geuvers-Klop conjecture is true for a *class* of systems. An interesting aspect of our class is that it includes both systems that are strongly normalizing as well as systems that are not. This shows that the technique does not implicitly use strong normalization of the systems in question. Moreover, for the specific systems of simply, second-order, and higher-order typed λ-calculus the present results improve those from Chapter 2 and those by Xi by not relying on any extra pairing operators, by not requiring domain-free formulations of any of the systems, and by showing that weak normalization of *all* legal expressions implies strong normalization of *all* legal expressions in the system.

Section 3.2 reviews some fundamental definitions. This includes a generalization of Coquand and Herbelin's notion of *logical non-dependent* pure type system to what we call *generalized non-dependent* pure type systems. The section also presents a classification of legal expressions into terms, types, and sorts due to Berardi. Section 3.3 and 3.4 present continuation passing style translations on types and terms, generalizing similar translations of Coquand and Herbelin. Section 3.5 uses the translations to infer strong normalization from weak normalization as in the previous chapter. Section 3.6 assesses the scope of the technique and reviews directions for further work.

3.2. Pure type systems

This section presents some fundamental definitions. The first subsection reviews *pure type systems,* as presented by Barendregt, Geuvers, and Nederhof [4, 39, 38]. Throughout the chapter we use implicitly numerous well-known properties about pure type systems. The second subsection introduces some notation regarding *normalization.* The third subsection presents the new class of *generalized non-dependent* pure type systems, in which *types* do not depend on *terms,* as shown in the fourth subsection.

3.2.1. Pure type systems

In this subsection we introduce *pure type systems.*

3.2.1. Definition. A *pure type system* (PTS) is a triple $(\mathcal{S}, \mathcal{A}, \mathcal{R})$ where

(i) $\mathcal{S}$ is a set of *sorts.*

(ii) $\mathcal{A} \subseteq \mathcal{S} \times \mathcal{S}$ is a set of *axioms.*

(iii) $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S} \times \mathcal{S}$ is a set of *rules.*

We write $(s, s') \in \mathcal{R}$ for $(s, s', s') \in \mathcal{R}$.

3.2.2. Definition. Let $(\mathcal{S}, \mathcal{A}, \mathcal{R})$ be a PTS.

(i) For each $s \in \mathcal{S}$, let $\mathcal{V}_s$ denote a countably infinite set of *variables* such that $\mathcal{V}_s \cap \mathcal{V}_{s'} = \emptyset$ when $s \neq s'$, and let $\mathcal{V} = \cup_{s \in \mathcal{S}} \mathcal{V}_s$.

(ii) The set $\mathcal{E}$ of *expressions* is given by the abstract syntax:

$$\mathcal{E} = \mathcal{V} \mid \mathcal{S} \mid \mathcal{E}\mathcal{E} \mid \lambda\mathcal{V} : \mathcal{E}.\mathcal{E} \mid \Pi\mathcal{V} : \mathcal{E}.\mathcal{E}.$$

We assume familiarity with the *subexpression* relation $\subseteq$, with the set $\mathrm{FV}(M)$ of *free variables* of M, and with substitution $M\{x := N\}$ for $x \in \mathcal{V}$ and $M, N \in \mathcal{E}$. We write $A \to B$ for $\Pi d{:}A.\, B$ when $d \notin \mathrm{FV}(B)$. We use $\equiv$ to denote syntactic identity modulo α-conversion and adopt the usual hygiene conventions—see [3].

(iii) The relation $\to_\beta$ on $\mathcal{E}$ is the compatible closure of the rule

$$(\lambda x{:}A \,.\, M)\ N \quad \beta \quad M\{x := N\}.$$

Also, $\twoheadrightarrow_\beta$ and $=_\beta$ are the transitive, reflexive closure and the transitive, reflexive, symmetric closure of $\to_\beta$, respectively.

(iv) The set $\mathcal{C}$ of *contexts* is the set of all sequences

$$x_1{:}A_1, \ldots, x_n{:}A_n,$$

where $x_1, \ldots, x_n \in \mathcal{V}$, $A_1, \ldots, A_n \in \mathcal{E}$, and $x_i \not\equiv x_j$ when $i \neq j$. The empty sequence is $[]$, and the concatenation of Γ and Δ is Γ, Δ. We write $x : A \in \Gamma$ if $\Gamma \equiv \Gamma_1, x : A, \Gamma_2$, for some Γ_1, Γ_2, and we write $\Gamma \subseteq \Delta$ if, for every $x : A \in \Gamma$, also $x : A \in \Delta$. For $\Gamma \in \mathcal{C}$, $\mathrm{dom}(\Gamma) = \{x \mid x{:}A \in \Gamma, \text{ for some } A\}$.

(v) The relation $\vdash\, \subseteq \mathcal{C} \times \mathcal{E} \times \mathcal{E}$ is defined in Figure 3.1. If $\Gamma \vdash M : A$, then Γ is *legal* and M, A are *legal* (in Γ). We use the notation $\Gamma \vdash A : B : C$ meaning that $\Gamma \vdash A : B$ and $\Gamma \vdash B : C$.

3.2.3. Convention. To save notation we often consider in the remainder a PTS λS and say, e.g., that $s \in \mathcal{S}$ or $M \in \mathcal{E}$ with the understanding that $\lambda S = (\mathcal{S}, \mathcal{A}, \mathcal{R})$ and that $\mathcal{V}$, $\mathcal{E}$, $\mathcal{C}$, $\to_\beta$ and $\vdash$ are defined as in Definition 3.2.2.

3.2.4. Example. The λ-*cube* consists of the eight PTSs λS, where

(i) $\mathcal{S} = \{*, \Box\}$.

(ii) $\mathcal{A} = \{(*, \Box)\}$.

(iii) $\{(*, *)\} \subseteq \mathcal{R} \subseteq \{(*, *), (\Box, *), (*, \Box), (\Box, \Box)\}$.

The name of each system and its associated set of rules is given by the table:

$\lambda\!\to$	$(*, *)$			
$\lambda 2$	$(*, *)$	$(\Box, *)$		
$\lambda\underline{\omega}$	$(*, *)$		$(\Box, \Box)$	
$\lambda\omega = \lambda\underline{\omega}2$	$(*, *)$	$(\Box, *)$	$(\Box, \Box)$	
λP	$(*, *)$			$(*, \Box)$
$\lambda P2$	$(*, *)$	$(\Box, *)$		$(*, \Box)$
$\lambda P\underline{\omega}$	$(*, *)$		$(\Box, \Box)$	$(*, \Box)$
$\lambda C = \lambda P\omega$	$(*, *)$	$(\Box, *)$	$(\Box, \Box)$	$(*, \Box)$

(axiom)	$[] \vdash s_1 : s_2$	if $(s_1, s_2) \in \mathcal{A}$
(start)	$\dfrac{\Gamma \vdash A : s}{\Gamma, x{:}A \vdash x : A}$	if $x \in \mathcal{V}_s$ & $x \notin \mathrm{dom}(\Gamma)$
(weakening)	$\dfrac{\Gamma \vdash A : B \quad \Gamma \vdash C : s}{\Gamma, x{:}C \vdash A : B}$	if $x \in \mathcal{V}_s$ & $x \notin \mathrm{dom}(\Gamma)$
(product)	$\dfrac{\Gamma \vdash A : s_1 \quad \Gamma, x{:}A \vdash B : s_2}{\Gamma \vdash (\Pi x{:}A.\, B) : s_3}$	if $(s_1, s_2, s_3) \in \mathcal{R}$
(application)	$\dfrac{\Gamma \vdash F : (\Pi x{:}A.\, B) \quad \Gamma \vdash a : A}{\Gamma \vdash F\, a : B\{x := a\}}$	
(abstraction)	$\dfrac{\Gamma, x{:}A \vdash b : B \quad \Gamma \vdash (\Pi x{:}A.\, B) : s}{\Gamma \vdash \lambda x{:}A\,.\, b : \Pi x{:}A.\, B}$	
(conversion)	$\dfrac{\Gamma \vdash A : B \quad \Gamma \vdash B' : s}{\Gamma \vdash A : B'}$	if $B =_\beta B'$

Figure 3.1: PURE TYPE SYSTEMS

The λ-cube is depicted diagrammatically in Figure 3.2. That traditional formulations of some of the systems in the λ-cube are equivalent with the formulations in terms of pure type systems is explained in [16, 4, 38], where more information about following systems may also be found.

3.2.5. EXAMPLE. The following systems extend $\lambda\omega$ with sort $\triangle$, axiom $\Box : \triangle$, and some rules for the new sort. The system λHOL is defined by:

(i) $\mathcal{S} = \{*, \Box, \triangle\}$.

(ii) $\mathcal{A} = \{(*, \Box), (\Box, \triangle)\}$.

(iii) $\mathcal{R} = \{(*, *), (\Box, *), (\Box, \Box)\}$.

The system λU^- is defined by:

(i) $\mathcal{S} = \{*, \Box, \triangle\}$.

(ii) $\mathcal{A} = \{(*, \Box), (\Box, \triangle)\}$.

(iii) $\mathcal{R} = \{(*, *), (\Box, *), (\Box, \Box), (\triangle, \Box)\}$.

$$\begin{tikzcd}
& \lambda\omega \arrow[rr] & & \lambda C \\
\lambda 2 \arrow[ur] \arrow[rr] & & \lambda P2 \arrow[ur] & \\
& \lambda\underline{\omega} \arrow[uu] \arrow[rr] & & \lambda P\underline{\omega} \arrow[uu] \\
\lambda{\to} \arrow[uu] \arrow[ur] \arrow[rr] & & \lambda P \arrow[uu] \arrow[ur] &
\end{tikzcd}$$

Figure 3.2: The λ-cube

The system λU is defined by:

(i) $\mathcal{S} = \{*, \Box, \triangle\}$.

(ii) $\mathcal{A} = \{(*, \Box), (\Box, \triangle)\}$.

(iii) $\mathcal{R} = \{(*, *), (\Box, *), (\Box, \Box), (\triangle, *), (\triangle, \Box)\}$.

3.2.6. Example. The system $\lambda *$ is defined by:

(i) $\mathcal{S} = \{*\}$.

(ii) $\mathcal{A} = \{(*, *)\}$.

(iii) $\mathcal{R} = \{(*, *)\}$.

3.2.2. Normalization

In this subsection we introduce some notation pertaining to *normalization*.

3.2.7. Definition. Let λS be a PTS. A *β-reduction path* from an expression M_0 is a (possibly infinite) sequence $M_0 \to_\beta M_1 \to_\beta M_2 \to_\beta \ldots$ If the sequence is finite, it *ends* in the last expression M_n and has *length* n.

3.2.8. Definition. Let λS be a PTS, and M an expression.

(i) $M \in \infty_\beta \Leftrightarrow$ there is an infinite β-reduction path from M.

(ii) $M \in \mathrm{NF}_\beta \Leftrightarrow$ there is no β-reduction path of length 1 or more from M.

(iii) $M \in \mathrm{SN}_\beta \Leftrightarrow$ all β-reduction paths from M are finite.

(iv) $M \in \mathrm{WN}_\beta \Leftrightarrow$ there is a β-reduction from M ending in $N \in \mathrm{NF}_\beta$.

Elements of $\mathrm{NF}_\beta, \mathrm{SN}_\beta, \mathrm{WN}_\beta$ are *β-normal forms*, *β-strongly normalizing*, and *β-weakly normalizing*, respectively. We also write, e.g., $\infty_\beta(M)$ for $M \in \infty_\beta$.

3.2.9. Definition. λS is *weakly normalizing* if all legal expressions are weakly normalizing, and *strongly normalizing* if all legal expressions are strongly normalizing. In this case we write $\lambda S \models \mathrm{WN}_\beta$ and $\lambda S \models \mathrm{SN}_\beta$, respectively.

3.2.10. Example. All the systems of the λ-cube are strongly normalizing—see, e.g., [16, 4, 39, 38]. The system $\lambda*$ is the simplest PTS which is not strongly normalizing. The system λU is is a natural extension of $\lambda\omega$ which, surprisingly, is not strongly normalizing. This result shows that, apparently, the fact that $\lambda*$ fails to be strongly normalizing is not merely a consequence of the cyclicity in its axiom.

3.2.11. Conjecture (Barendregt, Geuvers, Klop). *For every PTS λS:*

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

We shall prove the conjecture for a certain class of PTSs—see Theorem 3.5.20.

3.2.3. Generalized non-dependent pure type systems

This subsection presents the new notion of a *generalized non-dependent* PTS in which *types* do not depend on *terms*, as explained in Subsection 3.2.4.

The following notion is from [16, 4, 39, 38].

3.2.12. Definition. A PTS λS is *functional* iff

(i) For all $(s_1, s_2), (s_1', s_2') \in \mathcal{A}$: $s_1 \equiv s_1' \;\Rightarrow\; s_2 \equiv s_2'$.

(ii) For all $(s_1, s_2, s_3), (s_1', s_2', s_3') \in \mathcal{R}$: $s_1 \equiv s_1' \;\&\; s_2 \equiv s_2' \;\Rightarrow\; s_3 \equiv s_3'$.

3.2.13. Definition. Let λS be a functional PTS. λS is *persistent* if

(i) For all $(s_1, s_2), (s_1', s_2') \in \mathcal{A}$: $s_2 \equiv s_2' \;\Rightarrow\; s_1 \equiv s_1'$.

(ii) For all $(s_1, s_2, s_3) \in \mathcal{R}$: $s_2 \equiv s_3$.

3.2.14. Remark. Condition (ii) together with functionality ensures that the legal expressions can be classified into mutually exclusive and together exhaustive categories which do not depend on contexts—see Proposition 3.2.26. Condition (i) is useful for classifying subexpressions—see Proposition 3.2.32.

3.2.15. Remark. Berardi [16] studies classification in functional systems satisfying (ii). Geuvers and Nederhof [39] study classification in functional systems satisfying both (i) and the following condition implied by (ii):

(ii') For all $(s_1, s_2, s_3), (s'_1, s'_2, s'_3) \in \mathcal{R}$: $s_1 \equiv s_2 \;\&\; s_3 \equiv s'_3 \Rightarrow s_2 \equiv s'_2$.

For the purposes of this chapter, (ii) turns out to be the simplest condition to work with. Most PTSs in the literature satisfy (i), and most of those which satisfy (ii') also satisfy (ii). Hence little generality is lost by our choice.

The following relation is also mentioned by Berardi [16].

3.2.16. DEFINITION. Let λS be a PTS.

(i) The relation $<_{\mathcal{A}}$ is the transitive closure of $\mathcal{A}$.

(ii) The relation $\leq_{\mathcal{A}}$ is the reflexive closure of $<_{\mathcal{A}}$.

We often omit $\mathcal{A}$ from $<_{\mathcal{A}}$ and $\leq_{\mathcal{A}}$ to avoid clutter.

3.2.17. DEFINITION. A PTS λS is *stratified* if

(i) There is no infinite sequence $s_1, s_2, \ldots \in \mathcal{S}$ such that $s_1 < s_2 < \ldots$.

(ii) For all $(s_1, s_2, s_3) \in \mathcal{R}$: $s_1 \geq s_2 \geq s_3$.

3.2.18. REMARK. Condition (i) gives rise to a useful induction principle—see Remark 3.5.19—which is used in the proof of Theorem 3.5.20. Condition (ii) ensures that *types* do not depend on *terms*—see Remark 3.2.37.

3.2.19. EXAMPLE. The systems in the left-hand side of the cube are stratified, those in the right hand side are not. λU and λHOL are stratified, $\lambda *$ is not.

3.2.20. LEMMA. *Let λS be stratified. Then $s < s' \Rightarrow s \not\equiv s'$.*

PROOF. Assume $s < s'$. If $s \equiv s'$ then $s, s, \ldots$ would be an infinite sequence with $s < s < \ldots$ which is a contradiction. □

3.2.21. DEFINITION. A PTS is *generalized non-dependent* if it is both stratified and persistent.

3.2.22. EXAMPLE. The left hand side of the cube as well as λU and λHOL are generalized non-dependent.

3.2.23. REMARK. Generalized non-dependent PTSs resemble Coquand and Herbelin's *logical non-dependent* PTSs [27]. A functional PTS λS with distinguished sorts $P, T \in \mathcal{S}$ is logical if

(i) $(P, T) \in \mathcal{A}$.

(ii) $(s, P) \notin \mathcal{A}$ for all $s \in \mathcal{S}$.

(iii) $(P, P) \in \mathcal{R}$.

λS is logical non-dependent if, in addition, the only rules concerning P have form (s, P). With $P = *$ and $T = \Box$, the systems in the left hand side of the cube are logical non-dependent, but those in the right hand side are not. Whether or not a PTS is logical non-dependent naturally depends on the choice of P and T, and the fact that a PTS is logical non-dependent allows us to conclude something about expressions involving P and T only. This is quite adequate in many situations, but if we wish to reason about *all* the legal expression in a PTS we must require a notion of non-dependence that concerns *all* sorts. This is what generalized non-dependence attempts.

3.2.4. Classification

Now we divide the set of legal expressions into certain *terms, types,* and *sorts*, and show that in generalized non-dependent PTSs, types do not depend on terms.

3.2.24. Definition. Let λS be a PTS and $s \in \mathcal{S}$.

(i) s is a *top-sort* if there is no $s' \in \mathcal{S}$ with $(s, s') \in \mathcal{A}$.

(ii) s is a *bot-sort* if there is no $s' \in \mathcal{S}$ with $(s', s) \in \mathcal{A}$.

(iii) s is an *isolated sort* if s is both a bot-sort and a top-sort.

$\mathcal{S}_\top, \mathcal{S}_\bot, \mathcal{S}_{\mathrm{I}}$ are the set of top-sorts, bot-sorts, and isolated sorts, respectively.

The following terminology is from [16].

3.2.25. Definition. Let λS be a PTS, $s \in \mathcal{S}$.

(i) $\mathsf{Type}^s_\Gamma = \{M \in \mathcal{E} \mid \Gamma \vdash M : s\}$; $\mathsf{Type}^s = \cup_{\Gamma \in \mathcal{C}} \mathsf{Type}^s_\Gamma$.

(ii) $\mathsf{Term}^s_\Gamma = \{M \in \mathcal{E} \mid \exists A \in \mathcal{E} : \Gamma \vdash M : A : s\}$; $\mathsf{Term}^s = \cup_{\Gamma \in \mathcal{C}} \mathsf{Term}^s_\Gamma$.

The members of Type^s and Term^s are *s-types* and *s-terms,* respectively.

The following fundamental property is proved by Berardi [16]. A related result is due to Geuvers and Nederhof [39].

3.2.26. Proposition (Classification). *Let λS be persistent, M legal.*

(i) *$M \in \mathsf{Term}^s$ for some $s \in \mathcal{S}$; or*

(ii) *$M \in \mathsf{Type}^s$ for some $s \in \mathcal{S}_\top$; or*

(iii) *$M \equiv s$ for some $s \in \mathcal{S}_\top$.*

Moreover, (i)-(iii) *are mutually exclusive and s is unique in* (i)-(iii).

3.2.27. Example. The table in Figure 3.3 shows the categories in λC. Each legal expression is an object, a constructor, a kind, or $\Box$. Figuratively speaking, the mutually exclusive and together exhaustive categories are obtained by taking the left-most column and the top-most row in Figure 3.3.

s-terms Term^s	s-types Type^s	sorts s
constructors	kinds	$\Box$
objects	types	$*$

Figure 3.3: CATEGORIES IN λC.

The rest of this subsection is devoted to classification of subexpressions of a given expression.

3.2.28. REMARK. Let λS be generalized non-dependent. For simplicity, assume $\mathcal{S} = \{s_1, \dots, s_n\}$ and $\mathcal{A} = \{(s_1, s_2), (s_2, s_3), \dots, (s_{n-1}, s_n)\}$. Consider Figure 3.4, which is an abstract version of Figure 3.3. By Proposition 3.2.26,

Term^{s_n}	Type^{s_n}	s_n
$\vdots$	$\vdots$	$\vdots$
Term^{s_1}	Type^{s_1}	s_1

Figure 3.4: CATEGORIES IN λS.

each legal expression M is in the left-most column or in the top-most row. In the former case one can show that every subexpression of M is

(i) in the same category as M; or

(ii) in a category higher in the left-most column, or in the category at the top of the middle column.[1]

The following notion collects the cases in (ii) in a single set.

[1]In the case of an arbitrary generalized non-dependent PTS, $\mathcal{S}$ consists of a (possibly infinite) set of disjoint subsets $\mathcal{S}_1, \mathcal{S}_2, \dots$ each of which is totally ordered and has a greatest (but not necessarily a least) element with respect to $\leq_{\mathcal{A}}$. That is, the diagram of categories consists of a (possibly infinite) number of copies of Figure 3.4, each of which may be infinite downwards, but not upwards. The preceding reasoning then applies to each of the copies.

3.2.29. Definition.

$$\mathsf{Neu}^s_\Gamma = \{M \in \mathcal{E} \mid M \in \mathsf{Term}^{s'}_\Gamma \ \& \ s < s' \text{ or } M \in \mathsf{Type}^{s'}_\Gamma \ \& \ s \leq s' \in \mathcal{S}_\top\}.$$

Also, $\mathsf{Neu}^s = \cup_{\Gamma \in \mathcal{C}} \mathsf{Neu}^s_\Gamma$. The members of Neu^s are called *s-neutral.*

All s-types are s-neutral, and s-neutral expressions are not s-terms.

3.2.30. Lemma. *Let λS be generalized non-dependent, $s \in \mathcal{S}$.*

(i) $M \in \mathsf{Type}^{s'} \ \& \ s' \geq s \ \Rightarrow \ M \in \mathsf{Neu}^s$.

(ii) $M \in \mathsf{Neu}^s \ \Rightarrow \ M \notin \mathsf{Term}^s$.

Proof.

(i) Suppose $M \in \mathsf{Type}^{s'}$, $s' \geq s$. If $s' \in \mathcal{S}_\top$, $M \in \mathsf{Neu}^s$, trivially. If $s' \notin \mathcal{S}_\top$, then $(s', s'') \in \mathcal{A}$, for some s''. Then $\Gamma \vdash M : s' : s''$, for some Γ, i.e., $M \in \mathsf{Term}^{s''}_\Gamma$ and $s'' > s' \geq s$, so $M \in \mathsf{Neu}^s$ again.

(ii) We show the contrapositive. Suppose $M \in \mathsf{Term}^s$. Suppose $s' > s$. By Lemma 3.2.20, $s' \not\equiv s$. Thus, by Proposition 3.2.26, $M \notin \mathsf{Term}^{s'}$. Now suppose $s' \geq s$ and $s' \in \mathcal{S}_\top$. By Propositions 3.2.26, $M \notin \mathsf{Type}^{s'}$. Hence, $M \notin \mathsf{Neu}^s$. □

3.2.31. Lemma. *Let λS be generalized non-dependent, $M \in \mathsf{Type}^s_\Gamma$, $s \in \mathcal{S}_\top$.*

(i) $M \not\equiv x$.

(ii) $M \equiv s' \ \Rightarrow \ (s', s) \in \mathcal{A}$.

(iii) $M \not\equiv \lambda x{:}A \,.\, B$.

(iv) $M \not\equiv B\ A$

(v) $M \equiv \Pi x{:}A.\, B \ \Rightarrow \ A \in \mathsf{Type}^s_\Gamma \ \& \ B \in \mathsf{Type}^s_{\Gamma, x:A}$.

Proof. Assume $\Gamma \vdash M : s$, where $s \in \mathcal{S}_\top$.

(i) If $M \equiv x$, then by generation, $x : A \in \Gamma$, for some A with $A =_\beta s$ and $\Gamma \vdash A : s'$, for some s'. By Church-Rosser and subject reduction, $\Gamma \vdash s : s'$. By generation, $(s, s') \in \mathcal{A}$, contradicting $s \in \mathcal{S}_\top$.

(ii) By generation.

(iii) If $M \equiv \lambda x{:}A \,.\, B$, then by generation, $s =_\beta \Pi x{:}E.\, F$, for some E, F. By Church-Rosser, this is impossible.

(iv) If $M \equiv B\ A$, then by generation, $\Gamma \vdash B : \Pi x{:}E.\, F$ and $\Gamma \vdash A : E$, where $s =_\beta F\{x := A\}$, for some E and F. By correctness of types, $\Gamma \vdash \Pi x{:}E.\, F : s_3$, for some sort s_3. By generation again, $\Gamma \vdash E : s_1$ and $\Gamma, x : E \vdash F : s_2$, for some $(s_1, s_2, s_3) \in \mathcal{R}$. By substitution, $\Gamma \vdash F\{x := A\} : s_2$. By Church-Rosser, $F\{x := A\} \twoheadrightarrow_\beta s$. By subject reduction, $\Gamma \vdash s : s_2$. By generation, $(s, s_2) \in \mathcal{A}$, contradicting $s \in \mathcal{S}_\top$.

(v) If $M \equiv \Pi x{:}A.B$, then by generation $\Gamma \vdash A : s_1$ and $\Gamma, x{:}A \vdash B : s_2$ for some $(s_1, s_2, s) \in \mathcal{R}$. Since λS is generalized non-dependent, it holds that $s_1 \geq s_2 \equiv s$. Since $s \in \mathcal{S}_\top$, $s_1 \equiv s_2 \equiv s$. □

3.2.32. Proposition. *Let λS be generalized non-dependent, $M \in \mathsf{Term}^s_\Gamma$.*

(i) $M \equiv x \Rightarrow x \in \mathcal{V}_s$.

(ii) $M \equiv s' \Rightarrow (s', s''), (s'', s) \in \mathcal{A}$.

(iii) $M \equiv \lambda x{:}A\,.\,B \Rightarrow B \in \mathsf{Term}^s_{\Gamma,x:A}\ \&\ A \in \mathsf{Neu}^s_\Gamma$.

(iv) $M \equiv B\,A \Rightarrow B \in \mathsf{Term}^s_\Gamma\ \&\ A \in \mathsf{Term}^s_\Gamma \cup \mathsf{Neu}^s_\Gamma$.

(v) $M \equiv \Pi x{:}A.\,B \Rightarrow B \in \mathsf{Term}^s_{\Gamma,x:A}\ \&\ A \in \mathsf{Term}^s_\Gamma \cup \mathsf{Neu}^s_\Gamma$.

Proof. Assume $\Gamma \vdash M : D : s$.

(i) If $M \equiv x$, then by generation, $x : B \in \Gamma$ for some B with $\Gamma \vdash B : s'$, $x \in \mathcal{V}_{s'}$, and $B =_\beta D$. By Church-Rosser, $D \twoheadrightarrow_\beta E$ and $B \twoheadrightarrow_\beta E$ for some E. By subject reduction and uniqueness of types $s \equiv s'$.

(ii) If $M \equiv s'$, then by generation, $(s', s'') \in \mathcal{A}$ for some $s'' =_\beta D$. By Church-Rosser, $D \twoheadrightarrow_\beta s''$. By subject reduction, $\Gamma \vdash s'' : s$. By generation $(s'', s) \in \mathcal{A}$.

(iii) If $M \equiv \lambda x{:}A\,.\,B$, by generation, $\Gamma, x{:}A \vdash B : C$ and $\Gamma \vdash \Pi x{:}A.\,C : s'$ for some s' and C with $D =_\beta \Pi x{:}A.\,C$. By Church-Rosser, $D \twoheadrightarrow_\beta E$ and $\Pi x{:}A.\,C \twoheadrightarrow_\beta E$, for some E. By subject reduction and uniqueness of types, $s \equiv s'$. By generation, $\Gamma \vdash A : s_1$ and $\Gamma, x{:}A \vdash C : s$ for some $(s_1, s) \in \mathcal{R}$ with $s_1 \geq s$. Then $A \in \mathsf{Type}^{s_1}_\Gamma \subseteq \mathsf{Neu}^s_\Gamma$ and $B \in \mathsf{Term}^s_{\Gamma,x:A}$.

(iv) If $M \equiv B\,A$, then by generation, $\Gamma \vdash B : \Pi x{:}C.\,E$ and $\Gamma \vdash A : C$ for some C, E with $D =_\beta E\{x := A\}$. Then, by correctness of types, $\Gamma \vdash \Pi x{:}C.\,E : s_3$. By generation, $\Gamma \vdash C : s_1$ and $\Gamma, x{:}C \vdash E : s_3$ for some $(s_1, s_3) \in \mathcal{R}$ with $s_1 \geq s_3$. By substitution, $\Gamma \vdash E\{x := A\} : s_3$. By Church-Rosser, $E\{x := A\} \twoheadrightarrow_\beta F$ and $D \twoheadrightarrow_\beta F$, for some F. By subject reduction and uniqueness of types, $s \equiv s_3$. Hence $B \in \mathsf{Term}^s_\Gamma$ and $A \in \mathsf{Term}^{s_1}_\Gamma \subseteq \mathsf{Term}^s_\Gamma \cup \mathsf{Neu}^s_\Gamma$.

(v) If $M \equiv \Pi x{:}A.\,B$, then, by generation, $\Gamma \vdash A : s_1$ and $\Gamma, x{:}A \vdash B : s_3$ for some $(s_1, s_3) \in \mathcal{R}$ with $s_1 \geq s_3$ and $s_3 =_\beta D$. By Church-Rosser, $D \twoheadrightarrow_\beta s_3$. By subject and predicate reduction, $\Gamma \vdash \Pi x{:}A.\,B : s_3 : s$, so $B \in \mathsf{Term}^s_{\Gamma,x:A}$. By generation, $(s_3, s) \in \mathcal{A}$. Now, either $s_1 \equiv s_3$ and then $A \in \mathsf{Term}^s_\Gamma$, or $s_1 > s_3$ and then $A \in \mathsf{Type}^{s_1}_\Gamma$, where by injectivity of $\mathcal{A}$, $s_1 \geq s$, so $\mathsf{Type}^{s_1}_\Gamma \subseteq \mathsf{Neu}^s_\Gamma$. □

3.2.33. Proposition. *Let λS be generalized non-dependent, $M \in \mathsf{Neu}^s_\Gamma$.*

(i) $M \equiv x \Rightarrow x \in \mathcal{V}_{s'}\ \&\ s' > s$.

(ii) $M \equiv s' \Rightarrow (s', s'') \in \mathcal{A}$ *for some* $s'' \geq s$.

(iii) $M \equiv \lambda x{:}A\,.\,B \;\Rightarrow\; B \in \mathsf{Neu}^s_{\Gamma,x:A} \;\&\; A \in \mathsf{Neu}^s_{\Gamma}$.

(iv) $M \equiv B\,A \;\Rightarrow\; B \in \mathsf{Neu}^s_{\Gamma} \;\&\; A \in \mathsf{Neu}^s_{\Gamma}$

(v) $M \equiv \Pi x{:}A.\,B \;\Rightarrow\; B \in \mathsf{Neu}^s_{\Gamma,x:A} \;\&\; A \in \mathsf{Neu}^s_{\Gamma}$.

Proof. By Lemma 3.2.31 and Proposition 3.2.32. □

3.2.34. Remark. Proposition 3.2.32 and 3.2.33 will be used to define separate continuation passing style translations on s-terms and s-neutral expressions.

3.2.35. Corollary. *Let λS be generalized non-dependent and $s \in \mathcal{S}$.*

(i) $M \in \mathsf{Neu}^s \;\&\; N \subseteq M \;\Rightarrow\; N \in \mathsf{Neu}^s$.

(ii) $M \in \mathsf{Term}^s \;\&\; N \subseteq M \;\Rightarrow\; N \in \mathsf{Term}^s \cup \mathsf{Neu}^s$.

Proof. By induction on M using Proposition 3.2.33 and 3.2.32. □

As a special case we have the following analysis of the sorts of variables that can occur in s-neutral expressions and in s-terms.

3.2.36. Corollary. *Let λS be generalized non-dependent, $s \in \mathcal{S}$.*

(i) $B \in \mathsf{Neu}^s \;\&\; x \in \mathrm{FV}(B) \;\&\; x \in \mathcal{V}_{s'} \;\Rightarrow\; s' > s$.

(ii) $B \in \mathsf{Term}^s \;\&\; x \in \mathrm{FV}(B) \;\&\; x \in \mathcal{V}_{s'} \;\Rightarrow\; s' \geq s$.

Proof. Suppose $x \in \mathrm{FV}(B)$ and $x \in \mathcal{V}_{s'}$.

(i) If $B \in \mathsf{Neu}^s$, then, by Corollary 3.2.35, $x \in \mathsf{Neu}^s$. By Lemma 3.2.31, $x \notin \mathsf{Type}^{s''}$ if $s \leq s'' \in \mathcal{S}_{\top}$, so $x \in \mathsf{Term}^{s''}$ for some $s'' > s$. By Proposition 3.2.32, $x \in \mathcal{V}_{s''}$, ie, $s' \equiv s''$.

(ii) If $B \in \mathsf{Term}^s$, then, by Corollary 3.2.35, $x \in \mathsf{Term}^s \cup \mathsf{Neu}^s$. If $x \in \mathsf{Term}^s$, $x \in \mathcal{V}_s$, by Proposition 3.2.32. If $x \in \mathsf{Neu}^s$, proceed as in (i). □

3.2.37. Remark. Let λS be generalized non-dependent, $s \in \mathcal{S}$. Also, suppose $M \in \mathsf{Type}^s$ and $N \subseteq M$. By Lemma 3.2.30 and Corollary 3.2.35, $N \notin \mathsf{Term}^s$. Thus s-types do not depend on s-terms.

3.3. CPS translation of types

In this section we present a continuation passing style (CPS) translation on s-types. More precisely, we introduce a CPS translation on s-neutral expressions; this is more convenient than working with s-types, since the former are closed under subexpressions. The first subsection introduces the CPS translation. The second and third subsection show that the translation preserves β-equivalence and legality, respectively.

The translation generalizes Coquand and Herbelin's [27] translation for logical non-dependent pure type systems—see Remark 3.2.23—and the results below are similar to those of Coquand and Herbelin. The main problem involved with the generalization has already been solved—to generalize Coquand and Herbelin's notion of logical non-dependence to deal with all sorts of a PTS. Another, smaller, problem is to find conditions ensuring that *negation* makes sense on s-types; this leads to the notions of *negatable sorts* and *negatable PTSs* in the first subsection.

3.3.1. Translation

This subsection introduces a CPS translation on s-neutral expressions. For the translation we need a notion of *negation;* more precisely, we would like to have an expression $\bot_s$ such that if A is an s-type, then so is $A \to \bot_s$. The following definition expresses a requirement on the sort s that allows the construction of this product.

3.3.1. Definition. Let λS be PTS. An $s \in \mathcal{S}$ is *negatable* if

(i) s is not isolated (see Definition 3.2.24).

(ii) $(s, s, s) \in \mathcal{R}$.

An $s \in \mathcal{S}$ is *relevant* if $(s_1, s_2, s) \in \mathcal{R}$, for some $s_1, s_2 \in \mathcal{S}$. A PTS is *negatable* if all its relevant sorts are negatable.

The following then shows how to define negation.

3.3.2. Definition. Let λS be generalized non-dependent, $s \in \mathcal{S}$ negatable. Define $(\bot_s\, ;\, \Delta_s)$ by:

$$(\bot_s\, ;\, \Delta_s) = \begin{cases} (s' \;\; ; \;\; I_s\!:\!s' \to s') & \text{if } (s', s) \in \mathcal{A} \\ (z \;\; ; \;\; z\!:\!s\,,\; I_s\!:\!z \to z) & \text{else, if } (s, s') \in \mathcal{A}. \end{cases}$$

(the choice of s' is unique) where $z \in \mathcal{V}_{s'}$ and $I_s \in \mathcal{V}_s$. Let $\neg_s A \equiv A \to \bot_s$.

3.3.3. Remark. The purpose of the variable I_s will become clear in Section 3.4.

3.3.4. Lemma. *Let λS be generalized non-dependent, $s \in \mathcal{S}$ negatable. Then Δ_s is legal, $\Delta_s \vdash I_s : \bot_s \to \bot_s$, $\Delta_s \vdash \bot_s : s$, and*

$$\Gamma \vdash A : s \Rightarrow \Delta_s, \Gamma \vdash \neg_s A : s.$$

Proof. We consider two cases.

1. $(s', s) \in \mathcal{A}$, where $\bot_s \equiv s'$. Then $[] \vdash \bot_s : s$ and so $d{:}\bot_s \vdash \bot_s : s$, where d is a fresh variable. Since s is negatable, $(s, s, s) \in \mathcal{R}$. Thus $[] \vdash \bot_s \to \bot_s : s$ and $I_s{:}\bot_s \to \bot_s \vdash I_s : \bot_s \to \bot_s$. Therefore Δ_s is legal, $\Delta_s \vdash I_s : \bot_s \to \bot_s$, and by thinning, $\Delta_s \vdash \bot_s : s$.
2. $(s, s') \in \mathcal{A}$, where $\bot_s \equiv z$. Then $z{:}s \vdash z : s$ and $z{:}s, d{:}z \vdash z : s$. Hence $z{:}s \vdash z \to z : s$ and $z{:}s, I_s{:}z \to z \vdash I_s : z \to z$. Therefore Δ_s is legal, $\Delta_s \vdash I_s : \bot_s \to \bot_s$, and by thinning, $\Delta_s \vdash \bot_s : s$.

Now suppose $\Delta_s, \Gamma \vdash A : s$. By start and thinning, $\Delta_s, \Gamma, d : A \vdash \bot_s : s$. Hence, in both cases, $\Delta_s, \Gamma \vdash \neg_s A : s$. □

3.3.5. Definition. Let λS be generalized non-dependent, $s \in \mathcal{S}$ negatable. Define $\langle\cdot\rangle^s, \langle\!|\cdot|\!\rangle^s : \mathsf{Neu}^s \to \mathcal{E}$ and $\langle\!|\cdot|\!\rangle^s : \mathcal{C} \to \mathcal{C}$ as in Figure 3.5.

$$
\begin{array}{lcl}
\langle x\rangle^s & = & x \\
\langle s'\rangle^s & = & s' \\
\langle \lambda x{:}A\,.\,M\rangle^s & = & \lambda x{:}\langle A\rangle^s\,.\,\langle M\rangle^s \\
\langle M\,N\rangle^s & = & \langle M\rangle^s\,\langle N\rangle^s \\
\langle \Pi x{:}A.\,B\rangle^s & = & \Pi x{:}\langle\!|A|\!\rangle^s.\,\langle\!|B|\!\rangle^s \\
 & & \\
\langle\!|M|\!\rangle^s & = & \begin{cases} \neg_s\neg_s\,\langle M\rangle^s & \textbf{if } M \in \mathsf{Type}^s \\ \langle M\rangle^s & \textbf{otherwise} \end{cases} \\
 & & \\
\langle\!|[]|\!\rangle^s & = & [] \\
\langle\!|\Gamma, x : A|\!\rangle^s & = & \begin{cases} \langle\!|\Gamma|\!\rangle^s, x : \langle\!|A|\!\rangle^s & \textbf{if } A \in \mathsf{Type}^{s'} \text{ for some } s' \geq s \\ \langle\!|\Gamma|\!\rangle^s & \textbf{otherwise}\ . \end{cases}
\end{array}
$$

Figure 3.5: CPS translation of types

3.3.2. Preservation of equality on neutral expressions

In this subsection we show that if $B_1 =_\beta B_2$ for $B_1, B_2 \in \mathsf{Neu}^s$, then $\langle B_1\rangle^s =_\beta \langle B_2\rangle^s$ and $\langle\!|B_1|\!\rangle^s =_\beta \langle\!|B_2|\!\rangle^s$, if the system is generalized non-dependent, and s is negatable.

First a couple of lemmas.

3.3.6. Lemma. *Let λS be generalized independent and M legal in Γ.*

(i) $M \in \mathsf{Type}^s \Rightarrow M \in \mathsf{Type}^s_\Gamma$ *for all* $s \in \mathcal{S}_\top$.

(ii) $M \in \mathsf{Term}^s \Rightarrow M \in \mathsf{Term}^s_\Gamma$ *for all* $s \in \mathcal{S}$.

PROOF.

(i) Assume $M \in \mathsf{Type}^s$, for some $s \in \mathcal{S}_\top$, i.e., $\Delta \vdash M : s$, for some Δ. Since M is legal in Γ, $\Gamma \vdash M : B$ for some B (either *that,* or $\Gamma \vdash C : M$, for some C; and since M is not a top-sort, correctness of types implies $\Gamma \vdash M : B$, for some B again). If $B \notin \mathcal{S}_\top$, then, by correctness of types, $\Gamma \vdash B : s'$ for some s', so $M \in \mathsf{Term}^{s'}$, contradicting Proposition 3.2.26. Hence, $B \in \mathcal{S}_\top$, and by Proposition 3.2.26, $B \equiv s$, i.e., $M \in \mathsf{Type}^s_\Gamma$.

(ii) Assume $M \in \mathsf{Term}^s$, for some $s \in \mathcal{S}$, i.e., $\Delta \vdash M : A : s$, for some Δ, A. Since M is legal in Γ, $\Gamma \vdash M : B$ for some B as in (i). If $B \in \mathcal{S}_\top$, then, $M \in \mathsf{Type}^s$ for an $s \in \mathcal{S}_\top$, contradicting Proposition 3.2.26. Hence $B \notin \mathcal{S}_\top$, and by correctness of types, $\Gamma \vdash B : s'$ for some s', so $M \in \mathsf{Term}^{s'}$, and by Proposition 3.2.26, $s' \equiv s$, i.e., $M \in \mathsf{Term}^s_\Gamma$. $\square$

3.3.7. PROPOSITION. *Let λS be generalized non-dependent, M be legal in $\Gamma, x{:}A, \Delta$, and assume $\Gamma \vdash N : A$.*

(i) $M \in \mathcal{S}_\top \Leftrightarrow M\{x := N\} \in \mathcal{S}_\top$.

(ii) $M \in \mathsf{Type}^s_{\Gamma,x:A,\Delta} \Leftrightarrow M\{x := N\} \in \mathsf{Type}^s_{\Gamma,\Delta\{x:=N\}}$ *for all* $s \in \mathcal{S}_\top$.

(iii) $M \in \mathsf{Term}^s_{\Gamma,x:A,\Delta} \Leftrightarrow M\{x := N\} \in \mathsf{Term}^s_{\Gamma,\Delta\{x:=N\}}$ *for all* $s \in \mathcal{S}$.

PROOF. (i)-(iii)"$\Rightarrow$": by substitution.

(i)-(iii)"$\Leftarrow$": we show (i); (ii)-(iii) are similar. Assume $M\{x := N\} \in \mathcal{S}_\top$. Since M is legal in $\Gamma, x : A, \Delta$, by Proposition 3.2.26 and Lemma 3.3.6, exactly one of the following situations arise:

1. $M \in \mathcal{S}_\top$.
2. $M \in \mathsf{Type}^s_{\Gamma,x:A,\Delta}$ for some $s \in \mathcal{S}_\top$.
3. $M \in \mathsf{Term}^s_{\Gamma,x:A,\Delta}$ for some $s \in \mathcal{S}$.

Suppose, for the sake of contradiction, that $M \in \mathsf{Type}^s_{\Gamma,x:A,\Delta}$. By (ii)"$\Rightarrow$", $M\{x := N\} \in \mathsf{Type}^s_{\Gamma,\Delta\{x:=N\}}$. This contradicts $M\{x := N\} \in \mathcal{S}_\top$, by Proposition 3.2.26. Thus $M \notin \mathsf{Type}^s_{\Gamma,x:A,\Delta}$. Similarly, $M \notin \mathsf{Term}^s_{\Gamma,x:A,\Delta}$. Hence, $M \in \mathcal{S}_\top$. $\square$

3.3.8. LEMMA. *Let λS be generalized non-dependent, $B_1, B_2 \in \mathsf{Neu}^s_\Gamma$, and $B_1 =_\beta B_2$. Then*

$$B_1 \in \mathsf{Type}^s \Leftrightarrow B_2 \in \mathsf{Type}^s.$$

PROOF. Assume $B_1 \in \mathsf{Type}^s$. By Church-Rosser, $B_1 \twoheadrightarrow_\beta C$ and $B_2 \twoheadrightarrow_\beta C$, for some C. By subject reduction $C \in \mathsf{Type}^s$. We consider two cases.

1. $s \in \mathcal{S}_\top$. If $B_2 \notin \mathsf{Type}^s$, then by Proposition 3.2.26, $B_2 \in \mathcal{S}_\top \cup \mathsf{Term}^{s'}$, for some s', and then by subject reduction $C \in \mathcal{S}_\top \cup \mathsf{Term}^{s'}$, contradicting Proposition 3.2.26. Hence $B_2 \in \mathsf{Type}^s$.

2. $s \notin \mathcal{S}_\top$. Then $(s, s') \in \mathcal{A}$, for some s', i.e., $B_1, C \in \mathsf{Term}^{s'}$. Now $B_2 \notin \mathsf{Term}^{s'}$, yields a contradiction as in (i), so $B_2 \in \mathsf{Term}^{s'}$. By Lemma 3.3.6, $\Gamma \vdash B_1 : s : s'$ and $\Gamma \vdash B_2 : D : s'$. By uniqueness of types, Church-Rosser, and subject reduction, $\Gamma \vdash B_2 : s : s'$, so $B_2 \in \mathsf{Type}^s$. □

3.3.9. Lemma. *Let λS be generalized non-dependent, $s \in \mathcal{S}$ negatable. Assume $M \in \mathsf{Neu}^s_{\Gamma,x:A,\Delta}$ and $\Gamma \vdash N : A$. Then*

(i) $\langle M \rangle^s \{x := \langle N \rangle^s\} \equiv \langle M\{x := N\} \rangle^s$.

(ii) $\langle\!\langle M \rangle\!\rangle^s \{x := \langle N \rangle^s\} \equiv \langle\!\langle M\{x := N\} \rangle\!\rangle^s$.

Proof. Let $K^* \equiv K\{x := N\}$ for $K \in \mathcal{C} \cup \mathcal{E}$. $M^* \in \mathsf{Neu}^s_{\Gamma,\Delta^*}$, by substitution.

(i) By induction on M.

1. $M \equiv x$. Then,
$$\begin{array}{rcl} \langle x \rangle^s \{x := \langle N \rangle^s\} & \equiv & \langle N \rangle^s \\ & \equiv & \langle x^* \rangle^s. \end{array}$$

2. $M \equiv y \not\equiv x$. Then
$$\begin{array}{rcl} \langle y \rangle^s \{x := \langle N \rangle^s\} & \equiv & y \\ & \equiv & \langle y \rangle^s \\ & \equiv & \langle y^* \rangle^s. \end{array}$$

3. $M \equiv s'$. Similar to Case 2.

4. $M \equiv \lambda y{:}D \,.\, P$. By Proposition 3.2.33, $D \in \mathsf{Neu}^s_{\Gamma,x:A,\Delta}$ and also $P \in \mathsf{Neu}^s_{\Gamma,x:A,\Delta,y:D}$. Hence, by the induction hypothesis,
$$\begin{array}{rcl} (\langle \lambda y{:}D \,.\, P \rangle^s)\{x := \langle N \rangle^s\} & \equiv & \lambda y{:}\langle D^* \rangle^s \,.\, \langle P^* \rangle^s \\ & \equiv & \langle \lambda y{:}D^* \,.\, P^* \rangle^s \\ & \equiv & \langle (\lambda y{:}D \,.\, P)^* \rangle^s. \end{array}$$

5. $M \equiv M_1\, M_2$. Similar to Case 4.

6. $M \equiv \Pi y{:}A_1.\, A_2$. Similar to Case 4, using Proposition 3.3.7.

(ii) By (i) and Proposition 3.3.7. □

3.3.10. Lemma. *Let λS be generalized non-dependent, $s \in \mathcal{S}$ negatable, and $B_1 \in \mathsf{Neu}^s$. Then*
$$B_1 \to_\beta B_2 \;\Rightarrow\; \langle B_1 \rangle^s \to_\beta \langle B_2 \rangle^s.$$

Proof. By induction on $B_1 \to_\beta B_2$. By subject reduction, $B_2 \in \mathsf{Neu}^s$.

1. $B_1 \equiv (\lambda x{:}A\,.\,M)\,N \to_\beta M\{x := N\} \equiv B_2$. By assumption, $B_1 \in \mathsf{Neu}^s_\Gamma$, for some Γ. By a few steps of generation, $\Gamma \vdash N : E$, where $A =_\beta E$ and $\Gamma \vdash A : s'$, so by conversion $\Gamma \vdash N : A$. By Proposition 3.2.33, $M \in \mathsf{Neu}^s_{\Gamma,x:A}$. Then, by Lemma 3.3.9,

$$\begin{array}{rcl} \langle(\lambda x{:}A\,.\,M)\,N\rangle^s & \equiv & (\lambda x{:}\langle A\rangle^s\,.\,\langle M\rangle^s)\,\langle N\rangle^s \\ & \to_\beta & \langle M\rangle^s\{x := \langle N\rangle^s\} \\ & \equiv & \langle M\{x := N\}\rangle^s. \end{array}$$

2. $B_1 \equiv \Pi x{:}A.\,B \to_\beta \Pi x{:}A'.\,B' \equiv B_2$, where $A \to_\beta A'$ and $B \equiv B'$, or vice versa. Then, by the induction hypothesis and Proposition 3.2.33, $\langle A\rangle^s \to_\beta \langle A'\rangle^s$ and $\langle B\rangle^s \equiv \langle B'\rangle^s$, or vice versa. Then, by Lemma 3.3.8, $A \in \mathsf{Type}^s \Leftrightarrow A' \in \mathsf{Type}^s$ and $B \in \mathsf{Type}^s \Leftrightarrow B' \in \mathsf{Type}^s$. Therefore, $\langle\!\langle A\rangle\!\rangle^s \to_\beta \langle\!\langle A'\rangle\!\rangle^s$ and $\langle\!\langle B\rangle\!\rangle^s \equiv \langle\!\langle B'\rangle\!\rangle^s$, or vice versa. Thus,

$$\begin{array}{rcl} \langle \Pi x{:}A.\,B\rangle^s & \equiv & \Pi x{:}\langle\!\langle A\rangle\!\rangle^s.\,\langle\!\langle B\rangle\!\rangle^s \\ & \to_\beta & \Pi x{:}\langle\!\langle A'\rangle\!\rangle^s.\,\langle\!\langle B'\rangle\!\rangle^s \\ & \equiv & \langle \Pi x{:}A.\,B\rangle^s. \end{array}$$

3. $B_1 \equiv \lambda x{:}A\,.\,B \to_\beta \lambda x{:}A'\,.\,B' \equiv B_2$, where $A \to_\beta A'$ and $B \equiv B'$, or vice versa. Similar to Case 2.

4. $B_1 \equiv A\,B \to_\beta A'\,B' \equiv B_2$, where $A \to_\beta A'$ and $B \equiv B'$, or vice versa. Similar to Case 2. $\square$

3.3.11. Lemma. *Let λS be generalized non-dependent, $s \in \mathcal{S}$ negatable, and $B_1 \in \mathsf{Neu}^s$. Then*

$$B_1 \twoheadrightarrow_\beta B_2 \;\Rightarrow\; \langle B_1\rangle^s \twoheadrightarrow_\beta \langle B_2\rangle^s.$$

Proof. By Lemma 3.3.10, using transitivity and subject reduction. $\square$

3.3.12. Proposition. *Let λS be generalized non-dependent, $s \in \mathcal{S}$ negatable, $B_1, B_2 \in \mathsf{Neu}^s$, and $B_1 =_\beta B_2$. Then*

(i) $\langle B_1\rangle^s =_\beta \langle B_2\rangle^s$.

(ii) $\langle\!\langle B_1\rangle\!\rangle^s =_\beta \langle\!\langle B_2\rangle\!\rangle^s$.

Proof.

(i) By Church-Rosser, $B_1 \twoheadrightarrow_\beta C$ and $B_2 \twoheadrightarrow_\beta C$, for some C. By Lemma 3.3.11, $\langle B_1\rangle^s \twoheadrightarrow_\beta \langle C\rangle^s$ and $\langle B_2\rangle^s \twoheadrightarrow_\beta \langle C\rangle^s$. Hence, $\langle B_1\rangle^s =_\beta \langle B_2\rangle^s$.

(ii) By (i) and Lemma 3.3.8. $\square$

3.3.3. Embedding of types

In this subsection we show that, if $M \in \mathsf{Neu}^s$, then $\langle M\rangle^s \in \mathsf{Neu}^s$, provided the system is generalized non-dependent and s is negatable.

3.3.13. PROPOSITION. *Let λS be generalized non-dependent, and assume that $s \in \mathcal{S}$ is negatable. Then*

(i) *For all $s' \geq s$:* $\quad \Gamma \vdash A : s' \quad \Rightarrow \quad \Delta_s, \langle\!\langle\Gamma\rangle\!\rangle^s \vdash \langle A\rangle^s : s'$, *if $s' \in \mathcal{S}_\top$.*

(ii) *For all $s' > s$:* $\quad \Gamma \vdash M : A : s' \quad \Rightarrow \quad \Delta_s, \langle\!\langle\Gamma\rangle\!\rangle^s \vdash \langle M\rangle^s : \langle A\rangle^s : s'$.

PROOF. We prove simultaneously by induction on $\Gamma \vdash E : F$ that

(i) $F \equiv s' \in \mathcal{S}_\top \;\&\; s' \geq s \Rightarrow \Delta_s, \langle\!\langle\Gamma\rangle\!\rangle^s \vdash \langle E\rangle^s : s'$.

(ii) $\Gamma \vdash F : s' \;\&\; s' > s \Rightarrow \Delta_s, \langle\!\langle\Gamma\rangle\!\rangle^s \vdash \langle E\rangle^s : \langle F\rangle^s : s'$.

Note that $E, F \in \mathsf{Neu}^s$. We first check the cases of (i).

1. The derivation is
$$\vdash s_1 : s' \qquad (s_1, s') \in \mathcal{A}.$$
Since $\langle s_1\rangle^s = s_1$, $\langle\!\langle[]\rangle\!\rangle^s \equiv []$, and Δ_s is legal, start implies
$$\Delta_s, \langle\!\langle[]\rangle\!\rangle^s \vdash \langle s_1\rangle^s : s'.$$

2. The derivation ends in
$$\frac{\Gamma \vdash s' : s''}{\Gamma, x : s' \vdash x : s'}.$$
This contradicts Lemma 3.2.31.

3. The derivation ends in
$$\frac{\Gamma \vdash M : s' \qquad \Gamma \vdash C : s''}{\Gamma, x : C \vdash M : s'}.$$
By the induction hypothesis,
$$\Delta_s, \langle\!\langle\Gamma\rangle\!\rangle^s \vdash \langle M\rangle^s : s'.$$
We consider two cases.

3.1. $s'' \geq s$. If $s'' \in \mathcal{S}_\top$ then by the induction hypothesis
$$\Delta_s, \langle\!\langle\Gamma\rangle\!\rangle^s \vdash \langle C\rangle^s : s'' \qquad (*).$$
If $s'' \notin \mathcal{S}_\top$ then $(s'', s''') \in \mathcal{A}$, for some s'''. Then $\Gamma \vdash s'' : s'''$, where $s''' > s'' \geq s$, and $\langle s''\rangle^s \equiv s''$. Therefore, by the induction hypothesis (ii), $(*)$ holds also in this case.[2]

[2]In the remainder this type of step will be left implicit.

By Lemma 3.3.4 (if $s'' \equiv s$) and Proposition 3.2.26 (if $s'' > s$),

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \langle\!\langle C \rangle\!\rangle^s : s''.$$

Hence,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, x : \langle\!\langle C \rangle\!\rangle^s \vdash \langle M \rangle^s : s'.$$

Since $s'' \geq s$, $\langle\!\langle \Gamma, x : C \rangle\!\rangle^s = \langle\!\langle \Gamma \rangle\!\rangle^s, x : \langle\!\langle C \rangle\!\rangle^s$. Thus,[3]

$$\Delta_s, \langle\!\langle \Gamma, x : C \rangle\!\rangle^s \vdash \langle M \rangle^s : s'.$$

3.2. $s'' \not\geq s$. By Proposition 3.2.26, $\langle\!\langle \Gamma, x : C \rangle\!\rangle^s = \langle\!\langle \Gamma \rangle\!\rangle^s$. Thus,

$$\langle\!\langle \Gamma, x : C \rangle\!\rangle^s \vdash \langle M \rangle^s : s'.$$

4. The derivation ends in

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s'}{\Gamma \vdash \Pi x{:}A.\, B : s'} \quad (s_1, s') \in \mathcal{R}$$

where $s_1 \geq s'$. Since $s' \in \mathcal{S}_\top$, $s_1 \equiv s'$. By the induction hypothesis,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \langle A \rangle^s : s_1 \;\&\; \Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, x : \langle\!\langle A \rangle\!\rangle^s \vdash \langle B \rangle^s : s'.$$

By Lemma 3.3.4 and Proposition 3.2.26,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \langle\!\langle A \rangle\!\rangle^s : s_1 \;\&\; \Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, x : \langle\!\langle A \rangle\!\rangle^s \vdash \langle\!\langle B \rangle\!\rangle^s : s'.$$

Hence

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \Pi x{:}\langle\!\langle A \rangle\!\rangle^s.\, \langle\!\langle B \rangle\!\rangle^s : s'$$

Thus,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \langle \Pi x{:}A.\, B \rangle^s : s'.$$

5. The derivation ends in

$$\Gamma \vdash \lambda x{:}A\,.\,M : s'$$

where $s' \equiv \Pi x{:}A.\, B$. This case is impossible.

6. The derivation ends in

$$\frac{\Gamma \vdash M : \Pi x{:}A.\, B \quad \Gamma \vdash N : A}{\Gamma \vdash M\, N : s'}$$

where $s' \equiv B\{x := N\}$. This contradicts Lemma 3.2.31

[3] In the remainder this type of step will be left implicit.

7. The derivation ends in

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash s' : s''}{\Gamma \vdash M : s'} \quad A =_\beta s'$$

By generation $(s', s'') \in \mathcal{A}$, contradicting $s' \in \mathcal{S}_\top$.

This concludes the cases of (i). We proceed with the cases of (ii).

1. The derivation is

$$\vdash s_1 : s_2 \quad (s_1, s_2) \in \mathcal{A}$$

Since $\langle s_2 \rangle^s = s_2$, $\langle s_1 \rangle^s = s_1$, $\langle\!\!\langle [] \rangle\!\!\rangle^s = []$, and Δ_s is legal,

$$\Delta_s, \langle\!\!\langle [] \rangle\!\!\rangle^s \vdash \langle s_1 \rangle^s : \langle s_2 \rangle^s : s'.$$

2. The derivation ends in

$$\frac{\Gamma \vdash A : s''}{\Gamma, x : A \vdash x : A}$$

Then $\Gamma, x : A \vdash A : s''$. By uniqueness of types $s' \equiv s''$. By the induction hypothesis (i)-(ii),

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle A \rangle^s : s'.$$

By Proposition 3.2.26, $\langle\!\!\langle A \rangle\!\!\rangle^s \equiv \langle A \rangle^s$. Thus,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle\!\!\langle A \rangle\!\!\rangle^s : s'.$$

Hence,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s, x : \langle\!\!\langle A \rangle\!\!\rangle^s \vdash x : \langle A \rangle^s.$$

Also, $\langle x \rangle^s = x$. Thus,

$$\Delta_s, \langle\!\!\langle \Gamma, x : A \rangle\!\!\rangle^s \vdash \langle x \rangle^s : \langle A \rangle^s : s'.$$

3. The derivation ends in

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash C : s''}{\Gamma, x : C \vdash M : A}$$

Since $\Gamma \vdash M : A$, $x \notin \mathrm{FV}(M) \cup \mathrm{FV}(M)$. Hence, by strengthening, $\Gamma \vdash A : s'$. By the induction hypothesis,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle M \rangle^s : \langle A \rangle^s : s'.$$

Now proceed as in Case 3 in (i).

4. The derivation ends in

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s_3}{\Gamma \vdash \Pi x{:}A.\, B : s_3} \qquad (s_1, s_3) \in \mathcal{R}$$

where $s_1 \geq s_3$. By generation, $(s_3, s') \in \mathcal{A}$, injectivity of $\mathcal{A}$ implies $s_3 \geq s$. Hence, by the induction hypothesis (i)-(ii),

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \langle A \rangle^s : s_1 \ \& \ \Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, x : \langle\!\langle A \rangle\!\rangle^s \vdash \langle B \rangle^s : s_3.$$

By Lemma 3.3.4 and Proposition 3.2.26,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \langle\!\langle A \rangle\!\rangle^s : s_1 \ \& \ \Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, x : \langle\!\langle A \rangle\!\rangle^s \vdash \langle\!\langle B \rangle\!\rangle^s : s_3.$$

Hence

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \Pi x{:}\langle\!\langle A \rangle\!\rangle^s.\, \langle\!\langle B \rangle\!\rangle^s : s_3$$

Since $\langle s_3 \rangle^s \equiv s_3$,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \langle \Pi x{:}A.\, B \rangle^s : \langle s_3 \rangle^s : s'.$$

5. The derivation ends in

$$\frac{\Gamma, x : A \vdash M : B \quad \Gamma \vdash \Pi x{:}A.\, B : s''}{\Gamma \vdash \lambda x{:}A\,.\, M : \Pi x{:}A.\, B}$$

By functionality $s' \equiv s''$. By generation,

$$\Gamma \vdash A : s_1 \ \& \ \Gamma, x : A \vdash B : s' \ \& \ (s_1, s') \in \mathcal{R},$$

where $s_1 \geq s' > s$. By the induction hypothesis (i)-(ii),

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, x : \langle\!\langle A \rangle\!\rangle^s \vdash \langle M \rangle^s : \langle B \rangle^s \ \& \ \Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \Pi x{:}\langle\!\langle A \rangle\!\rangle^s.\, \langle\!\langle B \rangle\!\rangle^s : s'.$$

By Proposition 3.2.26, $\langle\!\langle A \rangle\!\rangle^s \equiv \langle A \rangle^s$ and $\langle\!\langle B \rangle\!\rangle^s \equiv \langle B \rangle^s$. Therefore,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \lambda x{:}\langle A \rangle^s\,.\, \langle M \rangle^s : \Pi x{:}\langle\!\langle A \rangle\!\rangle^s.\, \langle\!\langle B \rangle\!\rangle^s : s'.$$

Thus,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \langle \lambda x{:}A\,.\, M \rangle^s : \langle \Pi x{:}A.\, B \rangle^s : s'.$$

6. The derivation ends in

$$\frac{\Gamma \vdash M : \Pi x{:}A.\, B \quad \Gamma \vdash N : A}{\Gamma \vdash M\, N : B\{x := N\}}.$$

By correctness of types,

$$\Gamma \vdash \Pi x{:}A.\, B : s_3,$$

for some $s_3 \in \mathcal{S}$. By generation,

$$\Gamma \vdash A : s_1 \ \&\ \Gamma, x : A \vdash B : s_3 \quad (s_1, s_3) \in \mathcal{R},$$

where $s_1 \geq s_3$. By substitution,

$$\Gamma \vdash B\{x := N\} : s_3.$$

By uniqueness of types, $s_3 \equiv s'$. By the induction hypothesis,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle M \rangle^s : \Pi x{:}\langle\!\!\langle A \rangle\!\!\rangle^s.\, \langle\!\!\langle B \rangle\!\!\rangle^s : s' \ \&\ \Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle N \rangle^s : \langle A \rangle^s : s_1.$$

By generation,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s, x : \langle\!\!\langle A \rangle\!\!\rangle^s \vdash \langle\!\!\langle B \rangle\!\!\rangle^s : s'.$$

By Proposition 3.2.26, $\langle\!\!\langle A \rangle\!\!\rangle^s \equiv \langle A \rangle^s$ and $\langle\!\!\langle B \rangle\!\!\rangle^s \equiv \langle B \rangle^s$. Then, by substitution,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle B \rangle^s\{x := \langle N \rangle^s\} : s'.$$

Hence,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle M \rangle^s\, \langle N \rangle^s : \langle B \rangle^s\{x := \langle N \rangle^s\} : s'.$$

By Lemma 3.3.9, $\langle B \rangle^s\{x := \langle N \rangle^s\} \equiv \langle B\{x := N\} \rangle^s$. Thus,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle M\, N \rangle^s : \langle B\{x := N\} \rangle^s : s'.$$

7. The derivation ends in

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash B : s''}{\Gamma \vdash M : B} \quad A =_\beta B$$

As usual,

$$\Gamma \vdash A : s''.$$

By uniqueness of types $s' \equiv s''$. By the induction hypothesis (i)-(ii),

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle M \rangle^s : \langle A \rangle^s : s' \ \&\ \Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle B \rangle^s : s'.$$

By Proposition 3.3.12, $\langle A \rangle^s =_\beta \langle B \rangle^s$. Thus,

$$\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle M \rangle^s : \langle B \rangle^s : s'.$$

This concludes the proof. □

3.3.14. Corollary. *Let λS be generalized non-dependent, $s \in \mathcal{S}$ negatable.*

$$\Gamma \vdash A : s' \ \&\ s' \geq s \Rightarrow \Delta_s, \Gamma \vdash \langle A \rangle^s : s'.$$

Proof. Assume $\Gamma \vdash A : s'$. We consider two cases.

1. $s' \in \mathcal{S}_\top$. Then, by Proposition 3.3.13(i), $\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle A \rangle^s : s'$.
2. $s' \notin \mathcal{S}_\top$. Then $(s', s'') \in \mathcal{A}$, for some s''. By Proposition 3.3.13(ii), $\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle A \rangle^s : \langle s' \rangle^s$, i.e., $\Delta_s, \langle\!\!\langle \Gamma \rangle\!\!\rangle^s \vdash \langle A \rangle^s : s'$. □

3.4. CPS translation of terms

This section presents a CPS translation on s-terms. The first subsection discusses certain difficulties with the translation. The second subsection presents the translation, and the last subsection shows that the translation preserves legality.

The translation generalizes Coquand and Herbelin's [27] translation for logical non-dependent pure type systems—see Remark 3.2.23—and the results below are similar to those of Coquand and Herbelin. The generalization involves mainly two problems. First, for technical reasons our translation introduces some free variables which entail certain typing problems. In fact, it turns out that we are able to translate s-terms only when s has certain properties. The second problem, which is also encountered in the case of logical non-dependent systems, concerns the typing of certain bound variables introduced by the translation. These problems are discussed in the first subsection below.

3.4.1. Problems

This subsection discusses two problems involved with formulating a CPS translation on s-terms.

3.4.1. Discussion. A main difficulty with the CPS translation on s-terms, to be introduced below, stems from the introduction of fresh variables. For instance, consider in $\lambda 2$ the expression $\lambda\alpha{:}* \,.\, \lambda x{:}\alpha \,.\, x$ which is legal in the empty context. It will be translated into the expression

$$\lambda\kappa.\kappa\ \lambda\alpha{:}* \,.\, \lambda\eta.\ \bullet_\alpha\ \{(\lambda k.k\ \lambda x{:}\overset{\neg}{*}\overset{\neg}{*}\alpha\ .\lambda h.\bullet_x\ \overset{\neg}{*}\overset{\neg}{*}\alpha\ \ [(\lambda l.x\ l)\ h]\ x)\ \eta\}\ \alpha,$$

where $\bullet_\alpha$ and $\bullet_x$ are fresh variables, and where we have left out domains on some abstractions for brevity.

To show that the translation preserves legality we must type the translated expression in a context with bindings for the fresh variables $\bullet_\alpha$ and $\bullet_x$. Let us first consider how to type the subexpression $\lambda\eta.\cdots$. It turns out, ignoring the argument $\overset{\neg}{*}\overset{\neg}{*}\alpha$ to $\bullet_x$ for the moment, that this expression is legal in the context

$$\bot_*{:}*,\quad \bullet_\alpha{:}\bot_* \to * \to \bot_*,\quad \alpha{:}*,\quad \bullet_x{:}\bot_* \to \overset{\neg}{*}\overset{\neg}{*}\alpha\ \to \bot_*.$$

However, we cannot type $\lambda\alpha{:}*\,.\,\lambda\eta.\cdots$. The natural attempt to use the abstraction rule fails because we cannot remove $\alpha{:}*$ from the context. The problem is that the type $\bot_* \to \overset{\neg}{*}\overset{\neg}{*}\alpha\ \to \bot_*$ makes sense only in context $\alpha{:}*$.

The way out is to use instead the context

$$\bot_*{:}*,\quad \bullet_\alpha{:}\bot_* \to * \to \bot_*,\quad \alpha{:}*,\quad \bullet_x{:}\forall\beta : *.\bot_* \to \beta \to \bot_*$$

and use an explicit type application $\bullet_x \ \overset{\neg\neg}{**}\alpha$. The generalized type for $\bullet_x$ makes sense also after removal of $\alpha : *$ from the context.

On the other hand, the correct type for $\bullet_\alpha$ turns out to be $\bot_* \to * \to \bot_*$. The type system is not powerful enough to abstract $*$ analogously to the way $\overset{\neg\neg}{**}\alpha$ was abstracted. Fortunately, $*$ contains no free variables, so there is no need to abstract it.

In the general case of CPS translation of an s-term in some PTS λS, one must distinguish between those abstractions $\lambda x\!:A\,.\ \cdots$ under which the fresh variable $\bullet_x$ must be accompanied by a type application, and those abstractions $\lambda\alpha\!:A.\cdots$ under which the variable $\bullet_\alpha$ must not be accompanied by a type application.

Each fresh variable is introduced in the following situation

$$\frac{\Gamma \vdash A : s_1 \qquad \Gamma, x : A \vdash M : C : s}{\Gamma \vdash \lambda x\!:\!A\,.\,M : \Pi x\!:\!A.\,C : s} \qquad (s_1, s) \in \mathcal{R}$$

The CPS translation will introduce a fresh variable $\bullet_x$ under $\lambda x : A$. The type of this variable should be $\Pi B\!:\!s_1.\,\bot_s \to B \to \bot_s$ or $\bot_s \to (\!|A|\!)^s \to \bot_s$.

It is simplest to choose the former. This can be done whenever formation of the product in question is allowed, i.e., when there is $s_2 \in \mathcal{S}$ with

$$(s_1, s_2) \in \mathcal{A} \ \ \& \ \ (s_2, s) \in \mathcal{R}.$$

When the product is disallowed, we must choose the latter type and make sure that no binding $y : D$ for a free variable y of $(\!|A|\!)^s$ can subsequently be removed from the context. The free variables of $(\!|A|\!)^s$ are the same as those of A, and

$$y \in \mathrm{FV}(A) \ \Rightarrow \ y \in \mathcal{V}_{s_1'} \text{ for some } s_1' > s_1.$$

There are two ways such a variable y can be removed from the context:

$$\frac{\Delta, y\!:\!D\!:\!s_1' \vdash M : E : s}{\Delta \vdash \lambda y\!:\!D\,.\,M : \Pi y\!:\!D.\,E : s} \qquad (s_1', s) \in \mathcal{R}$$

$$\frac{\Delta, y\!:\!D\!:\!s_1' \vdash M : s_0 : s}{\Delta \vdash \Pi y\!:\!D.\,M : s_0 : s} \qquad (s_0, s) \in \mathcal{A} \ \ \& \ \ (s_1', s_0) \in \mathcal{R}$$

These two situations can be prevented by assuming for all $s_1' > s_1$:

(i) $(s_1', s) \notin \mathcal{R}$; and

(ii) $(s_0, s) \in \mathcal{A} \Rightarrow (s_1', s_0) \notin \mathcal{R}$.

This motivates the following definition.

3.4.2. Definition. Let λS be generalized non-dependent, $s \in \mathcal{S}$.

(i) $s_1 \in \mathcal{S}$ is *generalizable* in s, notation $s_1 \uparrow s$, if there is $s_2 \in \mathcal{S}$ such that $(s_1, s_2) \in \mathcal{A}$ and $(s_2, s) \in \mathcal{R}$.

(ii) $s_1 \in \mathcal{S}$ is *harmless* in s, notation $s_1 \downarrow s$, if for all $s_1' > s_1$: $(s_1', s) \notin \mathcal{R}$, and if $(s_0, s) \in \mathcal{A}$ then $(s_1', s_0) \notin \mathcal{R}$.

(iii) A rule $(s_1, s) \in \mathcal{R}$ is *clean* if $s_1 \uparrow s$ or $s_1 \downarrow s$.

(iv) λS is *clean* if all $(s_1, s) \in \mathcal{R}$ are clean.

3.4.3. Remark. Let λS be persistent, $(s_1, s) \in \mathcal{R}$. Each of the following conditions imply that $s_1 \downarrow s$ and therefore that (s_1, s) is clean.

(i) $s_1 \in \mathcal{S}_\top$.

(ii) for all $s_1' > s_1$: no rule has form (s_1', s').

(iii) $s \in \mathcal{S}_\perp$ and the only rule of form (s', s) is (s_1, s).

Consider the systems of the left hand side of the cube. By (i), all rules of form $(\Box, s)$ are clean. By (iii) the rule $(*, *)$ is clean in $\lambda{\to}$ and $\lambda\underline{\omega}$; in $\lambda 2$ and $\lambda\omega$ the rules is clean since $* \uparrow *$.

In λHOL, $(*, *)$ is clean since $* \uparrow *$, and $(\Box, *)$, $(\Box, \Box)$ are clean by (ii).

In λU^-, $(*, *)$ and $(\Box, \Box)$ are clean because $* \uparrow *$ and $\Box \uparrow \Box$, $(\Box, *)$ is clean because $\Box \downarrow *$ (none of (i)-(iii) apply), and $(\triangle, \Box)$ is clean by (i).

In λU, $(*, *)$, $(\Box, *)$, and $(\Box, \Box)$ are clean because because the first sort is generalizable in the second, and $(\triangle, *)$ and $(\triangle, \Box)$ are clean by (i).

The following gives a supply of fresh variables.

3.4.4. Definition. Let λS be a PTS.

(i) For each $s \in \mathcal{S}$, let $\mathcal{U}_s$ denote a countably infinite set of *variables* such that $\mathcal{U}_s \cap \mathcal{U}_{s'} = \emptyset$ when $s \neq s'$ and $\mathcal{U} \cap \mathcal{V} = \emptyset$, where $\mathcal{U} = \cup_{s \in \mathcal{S}} \mathcal{U}_s$.

(ii) For each $x \in \mathcal{V}_s$, let $\bullet_x \in \mathcal{U}_s$ be such that $\bullet_x \neq \bullet_y$ when $x \not\equiv y$.

The following shows how to choose fresh variables and typings for them.

3.4.5. Definition. Let λS be generalized non-dependent and clean, and $s \in \mathcal{S}$ negatable. For $M \in \mathsf{Term}^s$, define $\Delta^s(M)$ a in Figure 3.6.

The following lemma will be used to show that, indeed, free variables of the types in $\Delta^s(M)$ cannot be removed from context.

3.4.6. Lemma. *Let λS be generalized non-dependent and clean, $s \in \mathcal{S}$ negatable, and $M \in \mathsf{Term}^s$. Let $z \in \mathcal{V}_{s_1'}$ and $x \in \mathcal{V}_{s_1}$. Then*

$$\bullet_z : E \in \Delta^s(M) \;\; \& \;\; x \in \mathrm{FV}(E) \backslash \{\perp_s\} \quad \Rightarrow \quad s_1 > s_1' \;\; \& \;\; s_1' \downarrow s.$$

Proof. By induction on M.

$$
\begin{array}{lcl}
\Delta^s(x) & = & [] \\
\Delta^s(s') & = & [] \\
\Delta^s(\lambda x{:}A\,.\,B) & = & \begin{cases} \bullet_x : \Pi B{:}s_1.\,\bot_s \to B \to \bot_s,\ \ \Delta^s(B) & \textbf{if } x \in \mathcal{V}_{s_1}\ \ \&\ \ s_1 \uparrow s \\ \bullet_x : \bot_s \to \langle\!\langle A \rangle\!\rangle^s \to \bot_s,\ \ \Delta^s(B) & \textbf{if } x \in \mathcal{V}_{s_1}\ \ \&\ \ s_1 \downarrow s \end{cases} \\
\Delta^s(B\ A) & = & \begin{cases} \Delta^s(A), \Delta^s(B) & \textbf{if } A \in \mathsf{Term}^s \\ \Delta^s(B) & \textbf{else} \end{cases} \\
\Delta^s(\Pi x{:}A.\,B) & = & \begin{cases} \Delta^s(A), \Delta^s(B) & \textbf{if } A \in \mathsf{Term}^s \\ \Delta^s(B) & \textbf{else}\ . \end{cases}
\end{array}
$$

Figure 3.6: Choice of fresh variables

1. $M \equiv y$. Then the property trivially holds.
2. $M \equiv s'$. Similar to Case 1.
3. $M \equiv \lambda y{:}A\,.\,M$, where $y \in \mathcal{V}_{s_1''}$. By generation, $A \in \mathsf{Neu}_\Gamma^{s_1''}$.
   3.1. $s_1'' \uparrow s$. Then

   $$\Delta^s(\lambda y{:}A\,.\,M) \equiv \bullet_y : \Pi B{:}s_1.\,\bot_s \to B \to \bot_s,\ \ \Delta^s(M).$$

   Then $\bullet_z : E \in \Delta^s(M)$. Now use the induction hypothesis.

   3.2. $s_1'' \downarrow s$. Then

   $$\Delta^s(\lambda y{:}A\,.\,M) \equiv \bullet_y : \bot_s \to \langle\!\langle A \rangle\!\rangle^s \to \bot_s,\ \ \Delta^s(M).$$

   If $\bullet_z : E \in \Delta^s(M)$ use the induction hypothesis. If $\bullet_z$ is $\bullet_y$, then $z \equiv y$, so $s_1'' \equiv s_1'$. Since $x \in \mathrm{FV}(\langle\!\langle A \rangle\!\rangle^s) = \mathrm{FV}(A)$, Corollary 3.2.36 implies $s_1 > s_1'$.
4. $M \equiv B\ A$. Similar to Case 3.1.
5. $M \equiv \Pi x{:}A.\,B$. Similar to Case 3.1. □

3.4.7. Discussion. Another difficulty with our CPS translation on terms is that it introduces some new bound variables whose types depend on the type of the term we are translating. For instance, consider again the term

$$\lambda \alpha{:}*\,.\,\lambda x{:}\alpha\,.\,x.$$

If we supply the missing domains in the translated version

$$\lambda\kappa.\kappa\ \lambda\alpha{:}*\,.\,\lambda\eta.\,\bullet_\alpha\ \{(\lambda k.k\ \lambda x{:}\overset{\neg}{*}\overset{\neg}{*}\alpha\ .\lambda h.\bullet_x\ \overset{\neg}{*}\overset{\neg}{*}\alpha\ [(\lambda l.x\ l)\ h]\ x)\ \eta\}\ \alpha.$$

it turns out that the type of κ should be $\overset{\neg}{*}\,\langle D\rangle^*$ where D is the type of $\lambda\alpha : *\,.\,\lambda x : \alpha\,.\,x$. Our solution to this problem, following Coquand and

Herbelin [27], is to define the CPS translation of a term relative to the context in which the terms is considered. Another possibility [46] is to define the translation relative to derivations. These issues are discussed further in [10].

However, even in a fixed context, the type of a term is unique only up to β-equality. This ambiguity is resolved by choosing types in normal form; this is possible since we are working under the hypothesis that the system we are dealing with is weakly normalizing.

This motivates the following lemma and definition.

3.4.8. Lemma. *Let λS be functional and weakly normalizing.*

$$M \in \mathsf{Term}^s_\Gamma \Rightarrow \textit{there is exactly one } D \in \mathrm{NF}_\beta \textit{ with } \Gamma \vdash M : D : s\ .$$

Proof. Assume $M \in \mathsf{Term}^s_\Gamma$. We show that such a D exists. By assumption, $\Gamma \vdash M : C : s$ for some C. Since λS is weakly normalizing, $C \twoheadrightarrow_\beta D$ for some $D \in \mathrm{NF}_\beta$. By subject and predicate reduction, $\Gamma \vdash M : D : s$.

To show uniqueness of D, suppose that also $\Gamma \vdash M : D' : s$ for some $D' \in \mathrm{NF}_\beta$. By uniqueness of types, $D =_\beta D'$. Since $D, D \in \mathrm{NF}_\beta$, Church-Rosser implies $D \equiv D'$. $\square$

3.4.9. Definition. Let λS be functional and weakly normalizing. For any $M \in \mathsf{Term}^s_\Gamma$, $\mathsf{Type}^s_\Gamma(M)$ is the unique $D \in \mathrm{NF}_\beta$ with $\Gamma \vdash M : D : s$.

3.4.2. Translation

This subsection defines the translation on s-terms.

3.4.10. Definition. Let λS be generalized non-dependent, clean, and weakly normalizing, and $s \in \mathcal{S}$ negatable. For $M \in \mathsf{Term}^s_\Gamma$ define $[M]^s_\Gamma \in \mathcal{E}$ as in Figure 3.7.

3.4.11. Remark. $[M]^s_\Gamma$ is defined by induction on $M \in \mathsf{Term}^s_\Gamma$. The expressions D, E, and F which occur in the clauses for, e.g., $\lambda x{:}\,A\,.\,B$ are not necessarily smaller than $\lambda x{:}\,A\,.\,B$, but this does not matter since $\langle\cdot\rangle^s$, not $[\cdot]^s_\Gamma$, is applied to D, E, and F. The idea of using two distinct translations in this way also appears in [27] and [46].

3.4.3. Embedding of terms

Now we show that, if $M \in \mathsf{Term}^s_\Gamma$, then $[M]^s_\Gamma \in \mathsf{Term}^s_{\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, \Delta^s(M)}$.

First a lemma.

3.4.12. Lemma. *Let λS be weakly normalizing, generalized non-dependent, and clean, $s \in \mathcal{S}$ negatable. Let $\Gamma \subseteq \Delta$ both be legal, $M \in \mathsf{Term}^s_\Gamma$.*

$$
\begin{array}{lll}
[x]^s_\Gamma & = \lambda k{:}D\,.\,x\;k \\
[s']^s_\Gamma & = \lambda k{:}D\,.\,k\;s' \\
[\lambda x{:}A\,.\,B]^s_\Gamma & = \begin{cases} \lambda k{:}D\,.\,k\;\lambda x{:}\langle\!| A |\!\rangle^s\,.\,\lambda h{:}E\,.\;\bullet_x\;\langle\!| A |\!\rangle^s\;([B]^s_{(\Gamma,x:A)}\;h)\;x & \textbf{if } s_1 \uparrow s \\ \lambda k{:}D\,.\,k\;\lambda x{:}\langle\!| A |\!\rangle^s\,.\,\lambda h{:}E\,.\;\bullet_x\;([B]^s_{(\Gamma,x:A)}\;h)\;x & \textbf{else} \end{cases} \\
[B\;A]^s_\Gamma & = \begin{cases} \lambda k{:}D\,.\,[B]^s_\Gamma\;\lambda j{:}F\,.\,j\;[A]^s_\Gamma\;k & \textbf{if } A \in \mathsf{Term}^s \\ \lambda k{:}D\,.\,[B]^s_\Gamma\;\lambda j{:}F\,.\,j\;\langle A\rangle^s\;k & \textbf{else} \end{cases} \\
[\Pi x{:}A.\,B]^s_\Gamma & = \begin{cases} \lambda k{:}D\,.\,k\;\;\Pi x{:}([A]^s_\Gamma\;I_s).\,([B]^s_{\Gamma,x:A}\;I_s) & \textbf{if } A \in \mathsf{Term}^s \\ \lambda k{:}D\,.\,k\;\;\Pi x{:}\langle\!| A |\!\rangle^s.\,([B]^s_{\Gamma,x:A}\;I_s) & \textbf{else} \end{cases}
\end{array}
$$

where $D \equiv \,\bar{s}\langle \mathsf{Type}^s_\Gamma(M)\rangle^s$ in each clause for $[M]^s_\Gamma$.
$E \equiv \,\bar{s}\langle \mathsf{Type}^s_{\Gamma,x:A}(B)\rangle^s$ and $x \in \mathcal{V}_{s_1}$ in the clause for $[\lambda x{:}A\,.\,B]^s_\Gamma$
$F \equiv \langle \mathsf{Type}^s_\Gamma(B)\rangle^s$ in the clause for $[B\;A]^s_\Gamma$.

Figure 3.7: Non-standard CPS translation of terms

(i) $\mathsf{Type}^s_\Gamma(M) \equiv \mathsf{Type}^s_\Delta(M)$.

(ii) $[M]^s_\Gamma \equiv [M]^s_\Delta$.

Proof.

(i) Since $M \in \mathsf{Term}^s_\Gamma$, also $M \in \mathsf{Term}^s_\Delta$ by thinning. Let $A \equiv \mathsf{Type}^s_\Gamma(M)$ and $A' \equiv \mathsf{Type}^s_\Delta(M)$. Then $\Gamma \vdash M : A : s$ and $\Delta \vdash M : A' : s$. By thinning, $\Delta \vdash M : A : s$. By uniqueness of types, $A =_\beta A'$. Since $A, A' \in \mathrm{NF}$, Church-Rosser implies $A \equiv A'$.

(ii) Let $D \equiv \,\bar{s}\langle \mathsf{Type}^s_\Gamma(M)\rangle^s$, $D' \equiv \,\bar{s}\langle \mathsf{Type}^s_\Delta(M)\rangle^s$. By (i), $D \equiv D'$. Now proceed by induction on M.

1. $M \equiv x$. Then
$$[x]^s_\Gamma \equiv \lambda k{:}D\,.\,x\;k \equiv [x]^s_\Delta.$$
2. $M \equiv s'$. Similar to Case 1.
3. $M \equiv \lambda x : A\,.\,B$. By Proposition 3.2.32, $B \in \mathsf{Term}^s_{\Gamma,x:A}$ and $A \in \mathsf{Neu}^s$. Also, $\mathsf{Type}^s_\Gamma(M) \in \mathsf{Neu}^s$ and $\mathsf{Type}^s_{\Gamma,x:A}(B) \in \mathsf{Neu}^s_{\Gamma,x:A}$. Moreover, $\Gamma, x : A \subseteq \Delta, x : A$ are both legal. Finally, let $x \in \mathcal{V}_{s_1}$, and $E \equiv \,\bar{s}\langle \mathsf{Type}^s_{\Gamma,x:A}(B)\rangle^s$, $F \equiv \,\bar{s}\langle \mathsf{Type}^s_{\Delta,x:A}(B)\rangle^s$ By (i), $E \equiv F$. We consider two cases.

   3.1. $s_1 \uparrow s$. Then, by the induction hypothesis,
$$
\begin{array}{rcl}
[\lambda x{:}A\,.\,B]^s_\Gamma & = & \lambda k{:}D\,.\,k\;\lambda x{:}\langle\!| A |\!\rangle^s\,.\,\lambda h{:}E\,.\bullet_x\langle\!| A |\!\rangle^s\;([B]^s_{(\Gamma,x:A)}\;h)\;x \\
& = & \lambda k{:}D'.k\;\lambda x{:}\langle\!| A |\!\rangle^s\,.\,\lambda h{:}F\,.\bullet_x\langle\!| A |\!\rangle^s\;([B]^s_{(\Delta,x:A)}\;h)\;x \\
& = & [\lambda x{:}A\,.\,B]^s_\Delta.
\end{array}
$$

3.2. $s' \not\sim s$. Similar to 3.1.

4. $M \equiv B\,A$. Similar to Case 3.
5. $M \equiv \Pi x{:}A.\,B$. Similar to Case 3. □

3.4.13. PROPOSITION. *Let λS be weakly normalizing, generalized non-dependent, and clean, and $s \in \mathcal{S}$ negatable. Then*

$$\Gamma \vdash M : A : s \Rightarrow \Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, \Delta^s(M) \vdash [M]^s_\Gamma : \langle\!\langle A \rangle\!\rangle^s : s.$$

PROOF. By induction on $\Gamma \vdash M : A$. Before proceeding with the individual cases it is useful to make some general observations.

Let $D \equiv \overline{s}\langle \mathsf{Type}^s_\Gamma(M)\rangle^s$. By definition, $A \twoheadrightarrow_\beta \mathsf{Type}^s_\Gamma(M)$. By Lemma 3.3.11, $\overline{s}\langle A\rangle^s \twoheadrightarrow_\beta D$. By Corollary 3.3.14 and Lemma 3.3.4 $\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \overline{s}\langle A\rangle^s : s$ and $\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \overline{\overline{s}s}\,\langle A\rangle^s : s$. By subject reduction also $\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash D : s$ and $\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s \vdash \overline{s}D : s$.

We now proceed with the individual cases.

1. The derivation is
$$\vdash s_1 : s_2 \quad (s_1, s_2) \in \mathcal{A}.$$
By a few steps,
$$\Delta_s \vdash \lambda k{:}D\,.\,k\,s_1 : \overline{\overline{s}s}\,s_2.$$
That is,
$$\Delta_s, \langle\!\langle [] \rangle\!\rangle^s, \Delta^s(s_1) \vdash [s_1]^s_{[]} : \langle\!\langle s_2 \rangle\!\rangle^s.$$
2. The derivation ends in
$$\frac{\Gamma \vdash A : s'}{\Gamma, x : A \vdash x : A}$$
Then $\Gamma, x : A \vdash A : s'$. By uniqueness of types $s' \equiv s$. By a few steps,
$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, x : \overline{\overline{s}s}\,\langle A\rangle^s \vdash \lambda k{:}D\,.\,x\,k : \overline{\overline{s}s}\,\langle A\rangle^s.$$
That is,
$$\Delta_s, \langle\!\langle \Gamma, x : A \rangle\!\rangle^s, \Delta^s(x) \vdash [x]^s_{(\Gamma, x:A)} : \langle\!\langle A \rangle\!\rangle^s.$$
3. The derivation ends in
$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash C : s'}{\Gamma, x : C \vdash M : A}$$
Since $\Gamma \vdash M : A$, $x \notin \mathrm{FV}(M) \cup \mathrm{FV}(A)$. Hence, by strengthening, $\Gamma \vdash A : s$. By the induction hypothesis,
$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, \Delta^s(M) \vdash [M]^s_\Gamma : \langle\!\langle A \rangle\!\rangle^s.$$
By Lemma 3.4.12, $[M]^s_\Gamma = [M]^s_{(\Gamma, x:C)}$. Hence
$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, \Delta^s(M) \vdash [M]^s_{(\Gamma, x:C)} : \langle\!\langle A \rangle\!\rangle^s.$$
We consider two cases.

3.1. $s' \geq s$. By Corollary 3.3.14,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s \vdash \langle C \rangle^s : s'.$$

By Proposition 3.2.26 and Lemma 3.3.4,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s \vdash \langle\!\![C]\!\!\rangle^s : s'.$$

By thinning

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(M) \vdash \langle\!\![C]\!\!\rangle^s : s'.$$

Hence,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(M), x : \langle\!\![C]\!\!\rangle^s \vdash [M]^s_{(\Gamma, x:C)} : \langle\!\![A]\!\!\rangle^s.$$

Since $\langle\!\![C]\!\!\rangle^s$ is legal in $\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s$, it holds that $z \notin \mathrm{FV}(\langle\!\![C]\!\!\rangle^s)$, for all $z : E \in \Delta^s(M)$. By permutation,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, x : \langle\!\![C]\!\!\rangle^s, \Delta^s(M) \vdash [M]^s_{(\Gamma, x:C)} : \langle\!\![A]\!\!\rangle^s.$$

Thus,

$$\Delta_s, \langle\!\![\Gamma, x : C]\!\!\rangle^s, \Delta^s(M) \vdash [M]^s_{(\Gamma, x:C)} : \langle\!\![A]\!\!\rangle^s.$$

3.2. $s' \not\geq s$. By Proposition 3.2.26, $\langle\!\![\Gamma, x : C]\!\!\rangle^s = \langle\!\![\Gamma]\!\!\rangle^s$. Thus,

$$\Delta_s, \langle\!\![\Gamma, x : C]\!\!\rangle^s, \Delta^s(M) \vdash [M]^s_{(\Gamma, x:C)} : \langle\!\![A]\!\!\rangle^s.$$

4. The derivation ends in

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s_3}{\Gamma \vdash \Pi x{:}A.\, B : s_3} \qquad (s_1, s_3) \in \mathcal{R},$$

where $s_1 \geq s_3$ and $x \in \mathcal{V}_{s_1}$. Since $(s_3, s) \in \mathcal{A}$, $\bot_s \equiv s_3$.

4.1. $A \in \mathsf{Term}^s$, i.e., $\Gamma \vdash A : E : s$, for some E. By uniqueness of types, Church-Rosser, and subject reduction, $\Gamma \vdash s_1 : s$. By injectivity of $\mathcal{A}$, $s_1 \equiv s_3 \equiv \bot_s$. By the induction hypothesis,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(A) \vdash [A]^s_\Gamma :\overset{\neg}{s}\overset{\neg}{s}\ s_1 \& \Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(B) \vdash [B]^s_{(\Gamma, x:A)} :\overset{\neg}{s}\overset{\neg}{s}\ s_3.$$

By convention, $\mathrm{dom}(\Delta^s(A)) \cap \mathrm{dom}(\Delta^s(B)) = \emptyset$. Therefore, we can replace $\Delta^s(A)$ and $\Delta^s(B)$ by $\Delta^s(\Pi x{:}\, A.\, B)$. Therefore, in a few steps,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(\Pi x{:} A. B) \vdash \lambda k{:} D\,.\, k\; \Pi x{:} ([A]^s_\Gamma\, I_s).\, ([B]^s_{\Gamma, x:A}\, I_s) :\overset{\neg}{s}\overset{\neg}{s}\ s_3.$$

That is,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(\Pi x{:} A.\, B) \vdash [\Pi x{:} A.\, B]^s_\Gamma : \langle\!\![s_3]\!\!\rangle^s.$$

4.2. $A \notin \mathsf{Term}^s$. By injectivity of $\mathcal{A}$, $s_1 \equiv s_3$ or $s_1 \geq s$. Since $A \notin \mathsf{Term}^s$, the former is impossible. By Corollary 3.3.14 and the induction hypothesis,

$$\Delta_s, \langle\![\Gamma]\!\rangle^s \vdash \langle\![A]\!\rangle^s : s_1 \& \Delta_s, \langle\![\Gamma]\!\rangle^s, x : \langle\![A]\!\rangle^s, \Delta^s(B) \vdash [B]^s_{(\Gamma,x:A)} : \overset{\neg}{s}\overset{\neg}{s}\; s_3.$$

We must now move $x : \langle\![A]\!\rangle^s$ across $\Delta^s(B)$. Suppose $x \in \mathrm{FV}(E)$ for some $\bullet_z : E \in \Delta^s(B)\Gamma, x : A$. By Lemma 3.4.6, $z \in \mathcal{V}_{s_1'}$, $s_1 > s_1'$, and $s_1' \downarrow s$. This contradicts $(s_3, s) \in \mathcal{A}$ and $(s_1, s_3) \in \mathcal{R}$.
Hence by permutation,

$$\Delta_s, \langle\![\Gamma]\!\rangle^s, \Delta^s(B), x : \langle\![A]\!\rangle^s \ \vdash\ [B]^s_{(\Gamma,x:A)} : \overset{\neg}{s}\overset{\neg}{s}\; s_2.$$

Therefore, in a few steps,

$$\Delta_s, \langle\![\Gamma]\!\rangle^s, \Delta^s(B) \ \vdash\ \lambda k{:}D\,.\, k\, \Pi x{:}\langle\![A]\!\rangle^s.\ ([B]^s_{\Gamma,x:A}\, I_s) : \overset{\neg}{s}\overset{\neg}{s}\; s_3.$$

That is,

$$\Delta_s, \langle\![\Gamma]\!\rangle^s, \Delta^s(\Pi x{:}A.\, B) \ \vdash\ [\Pi x{:}A.\, B]^s_\Gamma : \langle\![s_3]\!\rangle^s.$$

5. The derivation ends in

$$\frac{\Gamma, x : A \ \vdash\ M : B \qquad \Gamma \ \vdash\ \Pi x{:}A.\, B : s'}{\Gamma \ \vdash\ \lambda x{:}A\,.\, M : \Pi x{:}A.\, B}.$$

By functionality $s' \equiv s$. By generation,

$$\Gamma \ \vdash\ A : s_1 \ \&\ \Gamma, x : A \ \vdash\ B : s \ \&\ (s_1, s) \in \mathcal{R},$$

where $s_1 \geq s$. Hence, by the induction hypothesis and thinning,

$$\Delta_s, \langle\![\Gamma]\!\rangle^s, x : \langle\![A]\!\rangle^s, \Delta^s(M) \ \vdash\ [M]^s_{(\Gamma,x:A)} : \langle\![B]\!\rangle^s.$$

We must now move $x : \langle\![A]\!\rangle^s$ across $\Delta^s(M)$. Suppose $x \in \mathrm{FV}(E)$ for some $\bullet_z : E \in \Delta^s(M)$. By Lemma 3.4.6, $z \in \mathcal{V}_{s_1'}$, $s_1 > s_1'$ and $s_1' \downarrow s$. This contradicts $(s_1, s) \in \mathcal{R}$. Hence by transitivity

$$\Delta_s, \langle\![\Gamma]\!\rangle^s, \Delta^s(M), x : \langle\![A]\!\rangle^s \ \vdash\ [M]^s_{(\Gamma,x:A)} : \langle\![B]\!\rangle^s.$$

We now consider two cases.

5.1. $s_1 \uparrow s$. In a few steps

$$\Delta_s, \langle\![\Gamma]\!\rangle^s, \Delta^s(M) \vdash \langle\![A]\!\rangle^s : s_1 \& \Delta_s, \langle\![\Gamma]\!\rangle^s, \Delta^s(M), x : \langle\![A]\!\rangle^s \vdash \overset{\neg}{s}\langle B\rangle^s : .$$

Therefore, after a few more steps,

$$\Delta_s, \langle\![\Gamma]\!\rangle^s, \bullet_x : \Pi B{:}s_1.\bot_s \to B \to \bot_s, \Delta^s(M) \vdash [\lambda x{:}A\,.\, M]^s_\Gamma : \langle\![\Pi x{:}A.B]\!\rangle^s,$$

i.e.,

$$\Delta_s, \langle\![\Gamma]\!\rangle^s, \Delta^s(\lambda x{:}A\,.\, M) \ \vdash\ [\lambda x{:}A\,.\, M]^s_\Gamma : \langle\![\Pi x{:}A.\, B]\!\rangle^s.$$

5.2. $s_1 \downarrow s$. Similar.

6. The derivation ends in

$$\frac{\Gamma \vdash M : \Pi x{:}A.\,B \qquad \Gamma \vdash N : A}{\Gamma \vdash M\,N : B\{x := N\}}\,.$$

By correctness of types,

$$\Gamma \vdash \Pi x{:}A.\,B : s_3.$$

for some $s_3 \in \mathcal{S}$. By generation,

$$\Gamma \vdash A : s_1 \;\&\; \Gamma, x : A \vdash B : s_3 \quad (s_1, s_3) \in \mathcal{R}.$$

where $s_1 \geq s_3$. By substitution,

$$\Gamma \vdash B\{x := N\} : s_3.$$

By uniqueness of types $s_3 \equiv s$. By the induction hypothesis,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(M) \vdash \langle M\rangle^s :_{\overset{\neg}{s}\overset{\neg}{s}} \Pi x{:}\langle\!\![A]\!\!\rangle^s.\, \langle\!\![B]\!\!\rangle^s.$$

By generation,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(M) \vdash \Pi x{:}\langle\!\![A]\!\!\rangle^s.\, \langle\!\![B]\!\!\rangle^s : s''.$$

for some s''. We now consider two cases.

6.1. $N \in \mathsf{Term}^s$. By the induction hypothesis,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(N) \vdash [N]^s_\Gamma : \langle\!\![A]\!\!\rangle^s.$$

Therefore, by a few simple steps,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(M\,N) \vdash \langle M\,N\rangle^s : \langle\!\![B]\!\!\rangle^s\{x := [N]^s_\Gamma\}.$$

By Corollary 3.2.36, $x \notin \mathrm{FV}(B) = \mathrm{FV}(\langle\!\![B]\!\!\rangle^s)$. Thus,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(M\,N) \vdash \langle M\,N\rangle^s : \langle\!\![B\{x := N\}]\!\!\rangle^s.$$

6.2. $N \notin \mathsf{Term}^s$. By Proposition 3.3.13,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s \vdash \langle N\rangle^s : \langle\!\![A]\!\!\rangle^s.$$

Therefore, by a few simple steps,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(M\,N) \vdash \langle M\,N\rangle^s : \langle\!\![B]\!\!\rangle^s\{x := \langle N\rangle^s\}.$$

By Lemma 3.3.9,

$$\Delta_s, \langle\!\![\Gamma]\!\!\rangle^s, \Delta^s(M\,N) \vdash \langle M\,N\rangle^s : \langle\!\![B\{x := N\}]\!\!\rangle^s.$$

7. The derivation ends in

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash B : s'}{\Gamma \vdash M : B} \quad A =_\beta B.$$

As usual,

$$\Gamma \vdash A : s'.$$

By uniqueness of types $s' \equiv s$. By the induction hypothesis,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, \Delta^s(M) \vdash [M]^s_\Gamma : \langle\!\langle A \rangle\!\rangle^s.$$

By Corollary 3.3.14, Lemma 3.3.4 and thinning,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, \Delta^s(M) \vdash \langle\!\langle A \rangle\!\rangle^s : s.$$

By Proposition 3.3.12, $\langle\!\langle A \rangle\!\rangle^s =_\beta \langle\!\langle B \rangle\!\rangle^s$. Thus,

$$\Delta_s, \langle\!\langle \Gamma \rangle\!\rangle^s, \Delta \vdash [M]^s_\Gamma : \langle\!\langle B \rangle\!\rangle^s.$$

This concludes the proof. □

3.5. Strong normalization from weak normalization

In this section we use the CPS translations of the two preceding sections to show that in all generalized non-dependent pure type systems—that are also negatable and clean—weak normalization implies strong normalization. The first subsection shows that our CPS translation on s-terms preserves infinite reductions. The second subsection proves a conservation result which is useful for relating weak and strong normalization, and the last subsection puts all the pieces together.

3.5.1. Preservation of infinite reductions

In this subsection we show that, for every $M \in \mathsf{Term}^s_\Gamma$,

$$[M]^s_\Gamma \in \mathrm{SN}_\beta \Rightarrow M \in \mathrm{SN}_\beta$$

when λS is generalized non-dependent, weakly normalizing and clean, and s is negatable. The proof technique, due to Xi [141], uses a variant of Plotkin's [100] colon translation. Other proofs are discussed in Chapter 2.

3.5.1. Definition. Let λS be generalized non-dependent, weakly normalizing, and clean, and $s \in \mathcal{S}$ be negatable. For $K \in \mathcal{E}$ and $M \in \mathsf{Term}^s_\Gamma$, define $M :^s_\Gamma K \in \mathcal{E}$ and $M ;^s_\Gamma \in \mathcal{E}$ as in Figure 3.8.

$$
\begin{array}{lcl}
x :^s_\Gamma K & = & x\, K \\
s' :^s_\Gamma K & = & K\, s' \\
(\lambda x{:}A\,.\,B) :^s_\Gamma K & = & \begin{cases} K\ \lambda x{:}\langle\!|A|\!\rangle^s\,.\,\lambda h{:}E\,.\ \bullet_x\ \langle\!|A|\!\rangle^s\ (B :^s_{(\Gamma,x:A)} h)\ x & \textbf{if } s_1 \uparrow s \\ K\ \lambda x{:}\langle\!|A|\!\rangle^s\,.\,\lambda h{:}E\,.\ \bullet_x\ (B :^s_{(\Gamma,x:A)} h)\ x & \textbf{if } s_1 \downarrow s \end{cases} \\
(B\ A) :^s_\Gamma K & = & \begin{cases} B :^s_\Gamma \lambda j{:}F\,.\,j\ A;^s_\Gamma\ K & \textbf{if } A \in \mathsf{Term}^s \\ B :^s_\Gamma \lambda j{:}F\,.\,j\ \langle A\rangle^s\ K & \textbf{else} \end{cases} \\
(\Pi x{:}A.\,B) :^s_\Gamma K & = & \begin{cases} K\ \Pi x{:}(A :^s_\Gamma I_s).\,(B :^s_{\Gamma,x:A} I_s) & \textbf{if } A \in \mathsf{Term}^s \\ K\ \Pi x{:}\langle\!|A|\!\rangle^s.\,(B :^s_{\Gamma,x:A} I_s) & \textbf{else} \end{cases} \\
 & & \\
M;^s_\Gamma & = & \lambda h{:}D\,.\,M :^s_\Gamma h
\end{array}
$$

where $E \equiv \neg s\langle\mathsf{Type}^s_{\Gamma,x:A}(B)\rangle^s$, $x \in \mathcal{V}_{s_1}$ in the clause for $(\lambda x{:}A\,.\,B) :^s_\Gamma K$.
$F \equiv \langle\mathsf{Type}^s_\Gamma(B)\rangle^s$ in the clause for $(B\ A) :^s_\Gamma K$.
$D \equiv \neg s\langle\mathsf{Type}^s_\Gamma(M)\rangle^s$ in the definition for $M;^s_\Gamma$.

Figure 3.8: COLON TRANSLATION OF TERMS

3.5.2. LEMMA. *Let λS be generalized non-dependent, weakly normalizing, and clean, and $s \in \mathcal{S}$ be negatable, and let $\Gamma \subseteq \Delta$ both be legal. For all $K \in \mathcal{E}$ and $M \in \mathsf{Term}^s_\Gamma$:*

$$M :^s_\Gamma K \equiv M :^s_\Delta K.$$

PROOF. By induction on M. Note that $M \in \mathsf{Term}^s_\Delta$ by thinning.

1. $M \equiv x$. Then
$$\begin{array}{rcl} x :^s_\Gamma K & \equiv & x\ K \\ & \equiv & x :^s_\Delta K. \end{array}$$
2. $M \equiv s'$. Similar to Case 1.
3. $M \equiv \lambda x{:}A\,.\,B$. Then $B \in \mathsf{Term}^s_{\Gamma,x:A}$ and $A \in \mathsf{Neu}^s$. Suppose first that $s_1 \uparrow s$, where $x \in \mathcal{V}_{s_1}$. Let $E_\Gamma \equiv \neg s\langle\mathsf{Type}^s_{\Gamma,x:A}(B)\rangle^s$ and let also $E_\Delta \equiv \neg s$ $\langle\mathsf{Type}^s_{\Delta,x:A}(B)\rangle^s$. By Lemma 3.4.12 and the induction hypothesis,
$$\begin{array}{rcl} (\lambda x{:}A\,.\,B) :^s_\Gamma K & \equiv & K\ \lambda x{:}\langle\!|A|\!\rangle^s\,.\,\lambda h{:}E_\Gamma\,.\ \bullet_x\ \langle\!|A|\!\rangle^s\ (B :^s_{(\Gamma,x:A)} h)\ x \\ & \equiv & K\ \lambda x{:}\langle\!|A|\!\rangle^s\,.\,\lambda h{:}E_\Delta\,.\ \bullet_x\ \langle\!|A|\!\rangle^s\ (B :^s_{(\Delta,x:A)} h)\ x \\ & \equiv & (\lambda x{:}A\,.\,B) :^s_\Delta K. \end{array}$$
The case where $s_1 \downarrow s$ is similar.
4. $M \equiv \Pi x{:}A.\,B$. Similar to Case 3.

5. $M \equiv A\, B$. Similar to Case 3. $\square$

3.5.3. LEMMA. *Let λS be generalized non-dependent, weakly normalizing, and clean, and $s \in \mathcal{S}$ be negatable. For all $M \in \mathsf{Term}^s_\Gamma$:*

(i) $k \notin \mathrm{dom}(\Gamma) \Rightarrow (M :^s_\Gamma K)\{k := L\} = M :^s_\Gamma (K\{k := L\})$.

(ii) $K \twoheadrightarrow_\beta L \Rightarrow M :^s_\Gamma K \twoheadrightarrow_\beta M :^s_\Gamma L$.

PROOF. By induction on M. $\square$

3.5.4. LEMMA. *Let λS be generalized non-dependent, weakly normalizing, and clean, and $s \in \mathcal{S}$ be negatable. Let $M \in \mathsf{Term}^s_{\Gamma,x:A,\Delta}$ and $\Gamma \vdash N : A$. Let $L^* \equiv L\{x := N\}$ for $L \in \mathcal{C} \cup \mathcal{E}$.*

(i) $N \in \mathsf{Term}^s_\Gamma \;\&\; L^+ \equiv L\{x := N;^s_\Gamma\} \Rightarrow (M :^s_{\Gamma,x:A,\Delta} K)^+ \twoheadrightarrow_\beta M^* :^s_{\Gamma,\Delta^*} K^+$.

(ii) $N \in \mathsf{Neu}^s_\Gamma \;\&\; L^\# \equiv L\{x := \langle N\rangle^s\} \Rightarrow (M :^s_{\Gamma,x:A,\Delta} K)^\# \twoheadrightarrow_\beta M^* :^s_{\Gamma,\Delta^*} K^\#$.

PROOF. (i) is by induction on M.

1. $M \equiv x$. By substitution, Γ, Δ^* is legal. Then, by Lemma 3.5.3(i) and Lemma 3.5.2,

$$\begin{array}{rcl} (x :^s_{\Gamma,x:A,\Delta} K)^+ & \equiv & (x\, K)^+ \\ & \equiv & N;^s_\Gamma\, K^+ \\ & \to_\beta & (N :^s_\Gamma h)\{h := K^+\} \\ & \equiv & N :^s_\Gamma K^+ \\ & \equiv & N :^s_{\Gamma,\Delta^*} K^+ \\ & \equiv & x^* :^s_{\Gamma,\Delta^*} K^+. \end{array}$$

2. $M \equiv y \not\equiv x$. By substitution, $y \in \mathsf{Term}^s_{\Gamma,\Delta^*}$, and

$$\begin{array}{rcl} (y :^s_{\Gamma,x:A,\Delta} K)^+ & \equiv & (y\, K)^+ \\ & \equiv & y\, K^+ \\ & \equiv & y :^s_{\Gamma,\Delta^*} K^+ \\ & \equiv & y^* :^s_{\Gamma,\Delta^*} K^+. \end{array}$$

3. $M \equiv s'$. Similar to the previous case.

4. $M \equiv \lambda y : B\, .\, C$. Then $C \in \mathsf{Term}^s_{\Gamma,x:A,\Delta,y:B}$ and $B \in \mathsf{Neu}^s$. Since $\Gamma \vdash N : A$ and $N \in \mathsf{Term}^s_\Gamma$, $A \in \mathsf{Type}^s_\Gamma$ and $x \in \mathcal{V}_s$.
Let $T \equiv \mathsf{Type}^s_{\Gamma,x:A,\Delta,y:B}(C)$ and $E \equiv \neg s\langle T\rangle^s$. Also, $T' \equiv \mathsf{Type}^s_{\Gamma,\Delta^*,y:B}(C^*)$ and $E' \equiv \neg s\langle T'\rangle^s$. Since $T \in \mathsf{Type}^s$, Corollary 3.2.36 implies $x \notin \mathrm{FV}(T) = \mathrm{FV}(\neg s\langle T\rangle^s)$. Since $\Gamma, x : A, \Delta, y : B \vdash C : T$, also $\Gamma, \Delta^*, y : B \vdash C^* : T^*$.

Therefore, $T^* \twoheadrightarrow_\beta T'$. By Lemma 3.3.11,

$$\begin{array}{rcl} E^+ & \equiv & (\,\bar{s}\langle T\rangle^s)^+ \\ & \equiv & \bar{s}\langle T\rangle^s \\ & \equiv & \bar{s}\langle T^*\rangle^s \\ & \twoheadrightarrow_\beta & \bar{s}\langle T'\rangle^s \\ & \equiv & E'. \end{array}$$

If $s \uparrow s$, then

$$\begin{array}{ll} \multicolumn{2}{l}{((\lambda y{:}B\,.\,C) :^s_{\Gamma,x:A,\Delta} K)^+} \\ \equiv & (K\ \lambda y{:}\langle\!|B|\!\rangle^s\,.\,\lambda h{:}E\,.\ \bullet_x\ \langle\!|B|\!\rangle^s\ (C :^s_{(\Gamma,x:A,\Delta,y:B)} h)\ y)^+ \\ \twoheadrightarrow_\beta & K^+\ \lambda y{:}\langle\!|B|\!\rangle^s\,.\,\lambda h{:}E'\,.\ \bullet_x\ \langle\!|B|\!\rangle^s\ (C^* :^s_{(\Gamma,\Delta^*,y:B)} h)\ y \\ \equiv & (\lambda y{:}B\,.\,C^*) :^s_{\Gamma,\Delta^*} K^+ \\ \equiv & (\lambda y{:}B\,.\,C)^* :^s_{\Gamma,\Delta^*} K^+. \end{array}$$

The case where $s \downarrow s$ is similar.

5. $M \equiv B\,C$. Similar to the preceding case.
6. $M \equiv \Pi y{:}B.\,C$.

This concludes the proof of (i). The proof of (ii) is by induction on M.

1. $M \equiv x$. This case is impossible: since $x \in \mathsf{Term}^s_{\Gamma,x:A,\Delta}$, it follows that $A \in \mathsf{Type}^s_{\Gamma,x:A,\Delta}$, hence $N \in \mathsf{Term}^s_{\Gamma,x:A,\Delta}$ contradicting $N \in \mathsf{Neu}^s$.
2. $M \equiv y \not\equiv x$. Then, by substitution, $y \in \mathsf{Term}^s_{\Gamma,\Delta^*}$, and

$$\begin{array}{rcl} (y :^s_{\Gamma,x:A,\Delta} K)^\# & \equiv & (y\ K)^\# \\ & \equiv & y\ K^\# \\ & \equiv & y :^s_{\Gamma,\Delta^*} K^\# \\ & \equiv & y^* :^s_{\Gamma,\Delta^*} K^\#. \end{array}$$

3. $M \equiv s'$. Similar to the previous case.
4. $M \equiv \lambda y{:}B\,.\,C$. Then $C \in \mathsf{Term}^s_{\Gamma,x:A,\Delta,y:B}$ and $B \in \mathsf{Neu}^s$.
   Let $T \equiv \mathsf{Type}^s_{\Gamma,x:A,\Delta,y:B}(C)$ and $E \equiv \bar{s}\langle T\rangle^s$. Also, $T' \equiv \mathsf{Type}^s_{\Gamma,\Delta^*,y:B}(C^*)$ and $E' \equiv \bar{s}\langle T'\rangle^s$. Since $\Gamma, x : A, \Delta, y : B \vdash C : T$, it also follows that $\Gamma, \Delta^*, y : B^* \vdash C^* : T^*$. Therefore, $T^* \twoheadrightarrow_\beta T'$. By Lemma 3.3.11 and 3.3.9,

$$\begin{array}{rcl} E^\# & \equiv & (\,\bar{s}\langle T\rangle^s)^\# \\ & \equiv & \bar{s}\langle T^*\rangle^s \\ & \twoheadrightarrow_\beta & \bar{s}\langle T'\rangle^s \\ & \equiv & E'. \end{array}$$

If $s \uparrow s$, then

$$\begin{array}{ll} \multicolumn{2}{l}{((\lambda y{:}B\,.\,C) :^s_{\Gamma,x:A,\Delta} K)^\#} \\ \equiv & (K\,\lambda y{:}\langle\!| B |\!\rangle^s\,.\,\lambda h{:}E\,.\,\bullet_x\ \langle\!| B |\!\rangle^s\,(C :^s_{(\Gamma,x:A,\Delta,y:B)} h)\,y)^\# \\ \twoheadrightarrow_\beta & K^\#\,\lambda y{:}\langle\!| B^* |\!\rangle^s\,.\,\lambda h{:}E'\,.\,\bullet_x\ \langle\!| B^* |\!\rangle^s\,(C^* :^s_{(\Gamma,\Delta^*,y:B^*)} h)\,y \\ \equiv & (\lambda y{:}B^*\,.\,C^*) :^s_{\Gamma,\Delta^*} K^\# \\ \equiv & (\lambda y{:}B\,.\,C)^* :^s_{\Gamma,\Delta^*} K^\#. \end{array}$$

The case where $s \downarrow s$ is similar.

5. $M \equiv B\,C$. Similar to the preceding case.
6. $M \equiv \Pi y{:}B.\,C$. Similar to the preceding case.

This concludes the proof of (ii). □

The following lemma, related to certain results in the theory of perpetual reductions (see Chapter 1), gives a sufficient condition for strong normalization of terms of a certain form.

3.5.5. Lemma. *Let $(\lambda x{:}A\,.\,M_0)\,M_1 \ldots M_n \in \mathcal{E}$ for some $n \geq 1$.*

$$A,\ M_1,\ M_0\{x := M_1\}\,M_2 \ldots M_n \in \mathrm{SN}_\beta \Rightarrow (\lambda x{:}A\,.\,M_0)\,M_1 \ldots M_n \in \mathrm{SN}_\beta.$$

Proof. Suppose $A,\ M_1,\ M_0\{x := M_1\}\,M_2 \ldots M_n \in \mathrm{SN}_\beta$. Clearly also $M_0, M_2, \ldots, M_n \in \mathrm{SN}_\beta$. If $(\lambda x{:}A\,.\,M_0)\,M_1 \ldots M_n \in \infty_\beta$, then any infinite reduction must therefore have form

$$\begin{array}{lll} (\lambda x{:}A\,.\,M_0)\,M_1 \ldots M_n & \twoheadrightarrow_\beta & (\lambda x{:}A'\,.\,M_0')\,M_1' \ldots M_n' \\ & \to_\beta & M_0'\{x := M_1'\}\,M_2' \ldots M_n' \\ & \to_\beta & \ldots\,. \end{array}$$

But then also

$$\begin{array}{lll} M_0\{x := M_1\}\,M_2 \ldots M_n & \twoheadrightarrow_\beta & M_0'\{x := M_1'\}\,M_2' \ldots M_n' \\ & \to_\beta & \ldots\,, \end{array}$$

contradicting $M_0\{x := M_1\}\,M_2 \ldots M_n \in \mathrm{SN}_\beta$. □

The following lemma summarizes the syntactic form of legal expressions.

3.5.6. Lemma. *Let λS be a PTS. If M is legal, then*

(i) $M \equiv x\,M_1 \ldots M_n$, *where* $n \geq 0$*; or*

(ii) $M \equiv s$*; or*

(iii) $M \equiv \Pi x{:}A.\,M_0$*; or*

(iv) $M \equiv (\lambda x{:}A\,.\,M_0)\,M_1 \ldots M_n$, *where* $n \geq 0$.

PROOF. Any $M \in \mathcal{E}$ has form (i), (ii'), (iii'), or (iv), where (ii'),(iii') are

(ii') $M \equiv s\, M_1 \ldots M_n$, where $n \geq 0$.

(iii') $M \equiv (\Pi x{:}A.\, M_0)\, M_1 \ldots M_n$, where $n \geq 0$.

The job then is to show that $n = 0$ in (ii') and (iii').

For (ii') let $s\, M_1 \ldots M_n$ be legal and assume $n > 0$. Then $s\, M_1$ is legal and, by correctness of types, $\Gamma \vdash s M_1 : s'$, for some Γ and s'. By generation, $\Gamma \vdash s : \Pi x{:}A.\, B$, for some $\Pi x{:}A.\, B$. By generation again, $\Pi x{:}A.\, B =_\beta s''$, for some s'', contradicting Church-Rosser. Thus $n = 0$.

For (iii') let $(\Pi x{:}\, A.\, M_0)\, M_1 \ldots M_n$ be legal and assume $n > 0$. Then $(\Pi x{:}A.\, B)\, M_1$ is legal and, by correctness of types, $\Gamma \vdash (\Pi x{:}A.\, M_0)\, M_1 : s'$, for some Γ and s'. By generation $\Gamma \vdash \Pi x{:}\, A.\, M_0 : \Pi y{:}\, E.\, F$, for some $\Pi y{:}E.\, F$. By generation again, $\Pi y{:}E.\, F =_\beta s''$, for some s'', contradicting Church-Rosser. Thus $n = 0$. $\square$

3.5.7. LEMMA. *Let λS be generalized non-dependent, weakly normalizing, and clean, and $s \in \mathcal{S}$ be negatable. For all $k \in \mathcal{V}$ and $M \in \mathsf{Term}^s_\Gamma$:*

$$M :^s_\Gamma k \in \mathrm{SN}_\beta \;\Rightarrow\; M \in \mathrm{SN}_\beta.$$

PROOF. By lexicographic induction on $\langle i, j\rangle$, where i is the length of the longest reduction from $M :^s_\Gamma k$ and j is the size of M. We split into cases according to the structure of M.

1. $M \equiv x\, M_1 \ldots M_n$, where $n \geq 0$. If $n = 0$, $M \equiv x \in \mathrm{SN}_\beta$. If $n \geq 1$, let

$$M'_i = \begin{cases} M_i ;^s_\Gamma & \textbf{if } M_i \in \mathsf{Term}^s \\ \langle M_i \rangle^s & \textbf{if } M_i \in \mathsf{Neu}^s. \end{cases}$$

   Also, for certain $F_1, \ldots, F_n$, let

$$\begin{array}{lcll} K_{n+1} & \equiv & k & \\ K_i & \equiv & \lambda j_i{:}F_i\, .\, j_i\, M'_i\, K_{i+1} & 1 \leq i \leq n. \end{array}$$

   Then

$$M :^s_\Gamma k \equiv x\, K_1.$$

   Since $M :^s_\Gamma k \in \mathrm{SN}_\beta$, also $M'_i \in \mathrm{SN}_\beta$ for all i. By Lemma 3.3.11 and the induction hypothesis, $M_i \in \mathrm{SN}_\beta$. Therefore $M \in \mathrm{SN}_\beta$.

2. $M \equiv s'$. Then $s' \in \mathrm{SN}_\beta$.
3. $M \equiv \Pi x{:}A.\, B$. Similar to Case 1.
4. $M \equiv \lambda x{:}A\, .\, B$. Similar to Case 1.

5. $M \equiv (\lambda x\colon A\,.\,B)\ M_1 \dots M_n$, where $n \geq 1$. Let M_i' and K_i be as in Case 1. Then

$$\begin{array}{rcl} M :^s_\Gamma k & \equiv & K_1\ \lambda x\colon \langle\!\langle A \rangle\!\rangle^s\,.\,\lambda h\colon E\,.\ \bullet_x\ \langle\!\langle A \rangle\!\rangle^s\ (B :^s_{(\Gamma,x:A)} h)\ x \\ & \twoheadrightarrow_\beta & \bullet_x\ \langle\!\langle A \rangle\!\rangle^s\ (B :^s_{(\Gamma,x:A)} K_2)\{x := M_1'\}\ M_1' \\ & \twoheadrightarrow_\beta & \bullet_x\ \langle\!\langle A \rangle\!\rangle^s\ (B\{x := M_1\} :^s_\Gamma K_2)\ M_1' \\ & \equiv & \bullet_x\ \langle\!\langle A \rangle\!\rangle^s\ (B\{x := M_1\}\ M_2 \dots M_n :^s_\Gamma k)\ M_1'. \end{array}$$

By Lemma 3.3.11, $A \in \mathrm{SN}_\beta$. Also, $B\{x := M_1\}\ M_2 \dots M_n :^s_\Gamma k \in \mathrm{SN}_\beta$ and $M_1' \in \mathrm{SN}_\beta$. Moreover, by Lemma 3.3.11 and the induction hypothesis $B\{x := M_1\}\ M_2 \dots M_n \in \mathrm{SN}_\beta$ and $M_1 \in \mathrm{SN}_\beta$. Therefore, $(\lambda x\colon A\,.\,B)\ M_1 \dots M_n \in \mathrm{SN}_\beta$, by Lemma 3.5.5. □

3.5.8. Lemma. *Let λS be generalized non-dependent, weakly normalizing, and clean, and $s \in \mathcal{S}$ be negatable. For all $M \in \mathsf{Term}^s_\Gamma$:*

$$[M]^s_\Gamma \twoheadrightarrow_\beta M ;^s_\Gamma\,.$$

Proof. By induction on M using Lemma 3.5.3(i). □

3.5.9. Proposition. *Let λS be generalized non-dependent, weakly normalizing, and clean, and $s \in \mathcal{S}$ be negatable. For all $M \in \mathsf{Term}^s_\Gamma$:*

$$[M]^s_\Gamma \in \mathrm{SN}_\beta \ \Rightarrow\ M \in \mathrm{SN}_\beta.$$

Proof. By Lemma 3.5.7 and Lemma 3.5.8. □

3.5.2. A conservation result

In this subsection we prove a version of the conservation theorem for expressions (see Chapter 1).

3.5.10. Definition. Let $K \to_\ell L$ mean that $K \to_\beta L$ by a *left-most* reduction.

3.5.11. Definition. Let λS be generalized non-dependent. An $s \in \mathcal{S}$ is *secure* if, for all $N \in \mathsf{Neu}^s$, $N \in \mathrm{SN}_\beta$.

3.5.12. Lemma. *Let λS be generalized non-dependent, $s \in \mathcal{S}$ be secure, and $M \in \mathsf{Term}^s$. Then there is an N such that:*

$$M \in \mathrm{WN}_\beta \ \Rightarrow\ M \twoheadrightarrow_\ell N \in \mathrm{NF}_\beta.$$

Proof. Rather than derive the result by the usual technique for untyped λ-terms we use erasing to infer the result from the one for untyped λ-terms.

Let L be the language generated by the grammar:

$$L ::= \mathcal{V} \mid \mathcal{S} \mid \lambda\mathcal{V}.L \mid L\ L \mid \Pi\mathcal{V}\colon L.\ L,$$

and let $|\cdot| : \mathcal{E} \to L$ be the forgetful map:

$$\begin{array}{rcl} |x| & = & x \\ |s| & = & s \\ |t\ u| & = & |t|\ |u| \\ |\lambda x : A.t| & = & \lambda x.|t| \\ |\Pi x : A.B| & = & \Pi x : |A|.|B| \end{array}$$

In terms of reduction, L is isomorphic to the set of untyped λ-terms—we can view $\Pi x{:}A.B$ as $x\,A\,B$. The relation $\to_{\beta^*}$ on L is the compatible closure of the rule

$$(\lambda x.b)\ a \quad \beta^* \quad b\{x := a\}.$$

For every $K \in \mathcal{E}$ show by induction on K that

$$K \to_\beta L \;\Rightarrow\; |K| \to_{\beta^*} |L| \qquad (\triangle).$$

For every $K \in L$, show by induction on K that

$$K \in \mathrm{NF}_\beta \;\Rightarrow\; |K| \in \mathrm{NF}_{\beta^*}. \qquad (+)$$

In the converse direction, show for all $N \in \mathsf{Term}^s$, by induction on N,

$$|N| \in \mathrm{NF}_{\beta^*} \;\Rightarrow\; N \in \mathrm{SN}_\beta. \qquad (*)$$

We write $K \to_{\ell^*} L$ if $K \to_{\beta^*} L$ by a left-most reduction. Finally, prove for all $N \in \mathsf{Term}^s$,

$$|N| \to_{\ell^*} K \;\Rightarrow\; \exists N' : N \twoheadrightarrow_\ell N' \;\&\; |N'| \equiv K. \qquad (\Box)$$

by induction on N using $(*)$, splitting into cases according to Lemma 3.5.6.

Since $M \in \mathrm{WN}_\beta$, also $|M| \in \mathrm{WN}_{\beta^*}$ by $(\triangle)$ and $(+)$. This result implies that left-most β^*-reduction of $|M|$ terminates in a normal form, i.e., that $|M| \twoheadrightarrow_{\ell^*} N \in \mathrm{NF}_{\beta^*}$—see Section 1.7.6. By $(\Box)$, $M \twoheadrightarrow_\ell M'$ & $|M'| \equiv N$ for some $M' \in \mathrm{SN}_\beta$. Hence $M \twoheadrightarrow_\ell M' \twoheadrightarrow_\ell M'' \in \mathrm{NF}_\beta$, by $(*)$. $\Box$

3.5.13. Remark. The idea in the proof of Lemma 3.5.12 of studying *domain-free* expressions (elements of L) to prove properties about expressions (elements of $\mathcal{E}$) appears also in [38] and [14]. In the latter paper, so-called *domain-free* pure type systems are introduced, allowing properties about *legal* expressions to be inferred from properties about *legal* domain-free expressions.

3.5.14. Definition. Let λS be generalized non-dependent and $s \in \mathcal{S}$.

$$\mathsf{I\text{-}Term}^s = \{M \in \mathsf{Term}^s \cup \mathsf{Neu}^s \mid M \supseteq \lambda x{:}A\,.\,B \in \mathsf{Term}^s \;\Rightarrow\; x \in \mathrm{FV}(B)\}.$$

3.5.15. Proposition. *Let λS be generalized non-dependent, $s \in \mathcal{S}$ secure. For all $M \in$ I-Term^s:*

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

Proof. By Lemma 3.5.12 we may proceed by induction on lexicographically ordered pairs $\langle m, M\rangle$, where m is the length of the left-most reduction sequence to normal-form of M.

1. $M \equiv x\, M_1 \dots M_n$. Then $M_1, \dots, M_n \in \mathrm{WN}_\beta$. $M_1, \dots, M_n \in \text{I-}\mathsf{Term}^s$, so by the induction hypothesis, $M_1, \dots, M_n \in \mathrm{SN}_\beta$, so $M \in \mathrm{SN}_\beta$.
2. $M \equiv' s$. Then $M \in \mathrm{SN}_\beta$.
3. $M \equiv \Pi x{:}A.\, B$. Similar to Case 1.
4. $M \equiv (\lambda x : A\, .\, M_0)\, M_1 \dots M_n$. If $n = 0$, proceed as in Case 1. Now assume $n > 0$. If $M \in \mathsf{Neu}^s$, then $M \in \mathrm{SN}_\beta$, so assume $M \in \mathsf{Term}^s$. Then, $M \to_\ell M_0\{x := M_1\}\, M_2 \dots M_n \in \mathsf{Term}^s \cap \text{I-}\mathsf{Term}^s \cap \mathrm{WN}_\beta$. By the induction hypothesis, $M_0\{x := M_1\}\, M_2 \dots M_n \in \mathrm{SN}_\beta$. Also, since $\lambda x : A\, .\, M_0 \in \mathsf{Term}^s$, $x \in \mathrm{FV}(M_0)$, so $M_1 \in \mathrm{SN}_\beta$. Then $M \in \mathrm{SN}_\beta$ by Lemma 3.5.5. □

3.5.3. Strong normalization from weak normalization

In this subsection we finally show that

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

provided λS is generalized non-dependent, clean and negatable.

3.5.16. Lemma. *Let λS be generalized non-dependent, weakly normalizing and clean, and $s \in \mathcal{S}$ secure and negatable. For $M \in \mathsf{Term}^s_\Gamma$:*

$$[M]^s_\Gamma \in \mathrm{WN}_\beta \ \Rightarrow\ M \in \mathrm{SN}_\beta.$$

Proof. By Proposition 3.5.15 and 3.5.9, noting that $[M]^s_\Gamma \in \text{I-}\mathsf{Term}^s$. □

3.5.17. Lemma. *Let λS be generalized non-dependent.*

(i) *For all $s \in \mathcal{S}_\top$, $s \in \mathrm{SN}_\beta$.*

(ii) *For all $s \in \mathcal{S}_\top$, and $M \in \mathsf{Type}^s$, $M \in \mathrm{SN}_\beta$.*

Proof. (i) is trivial. (ii) is by induction on M using Lemma 3.2.31. □

3.5.18. Lemma. *Let λS be generalized non-dependent and weakly normalizing, and $s \in \mathcal{S}$ secure. If $s \in \mathcal{S}$ is irrelevant then*

$$M \in \mathsf{Term}^s \ \Rightarrow\ M \in \mathrm{SN}_\beta.$$

Proof. Assume that $M \in \mathsf{Term}^s$ and s is irrelevant. We show that then M is not an application. The result then follows by induction on M using Proposition 3.2.32.

So, suppose $\Gamma \vdash K\ L : C : s$ for some K, L, C. Then, by generation $\Gamma \vdash K : \Pi x{:}A.\,B : s$ for some $\Pi x{:}A.\,B$. By generation again, there is some $(s_1, s) \in \mathcal{R}$, contradicting irrelevance of s. $\square$

3.5.19. Remark. Let λS be generalized non-dependent, $s \in \mathcal{S}$. There is no infinite sequence $s \equiv s_0 : s_1 : s_2 \ldots$ with $(s_0, s_1), (s_1, s_2), \ldots \in \mathcal{A}$ since λS is stratified. In fact, by functionality it easily follows that there is an n such that for any sequence

$$s \equiv s_0 : s_1 : \ldots : s_{m-1} : s_m \qquad (*)$$

with $(s_0, s_1), (s_1, s_2), \ldots, (s_{m-1}, s_m) \in \mathcal{A}$, $m \leq n$.

Let $l(s)$ denote the least n such that for any sequence of form $(*)$, $m \leq n$.

3.5.20. Theorem. *Let λS belong to the (infinite) class of generalized non-dependent, clean, and negatable.*

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

Proof. Suppose $\lambda S \models \mathrm{WN}_\beta$. We prove that for any legal expression M, $M \in \mathrm{SN}_\beta$. If $M \in \mathcal{S}_\top$ or $M \in \mathsf{Type}^s$ for some $s \in \mathcal{S}_\top$, then $M \in \mathrm{SN}_\beta$, by Lemma 3.5.17. By Proposition 3.2.26 it suffices to show for all $s \in \mathcal{S}$:

$$M \in \mathsf{Term}^s \ \Rightarrow\ M \in \mathrm{SN}_\beta.$$

We proceed by induction on $l(s)$.

1. $l(s) = 0$. Then $s \in \mathcal{S}_\top$. If $N \in \mathsf{Neu}^s$, then $N \in \mathsf{Type}^s$, so $N \in \mathrm{SN}_\beta$, by Lemma 3.5.17. Thus, s is secure. Now let $M \in \mathsf{Term}^s$, i.e., $M \in \mathsf{Term}^s_\Gamma$ for some Γ. If s is irrelevant, then $M \in \mathrm{SN}_\beta$, by Lemma 3.5.18. If s is relevant, then s is also negatable. By Proposition 3.4.13, $[M]^s_\Gamma \in \mathsf{Term}^s$, so $[M]^s_\Gamma \in \mathrm{WN}_\beta$ by assumption. Then $M \in \mathrm{SN}_\beta$, by Lemma 3.5.16.
2. $l(s) > 0$. If $N \in \mathsf{Neu}^s$, then $N \in \mathsf{Type}^{s'}$ for some $s \leq s' \in \mathcal{S}_\top$ and then $N \in \mathrm{SN}_\beta$, or $N \in \mathsf{Term}^{s'}$ where $s < s'$, and then $N \in \mathrm{SN}_\beta$ by the induction hypothesis. Thus s is secure. Now proceed as in Case 1. $\square$

3.5.21. Corollary. *If λS is any of $\lambda{\to}, \lambda 2, \lambda\underline{\omega}, \lambda\omega, \lambda HOL, \lambda U^-, \lambda U$, then*

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

3.6. Conclusion

We have shown that for any generalized non-dependent (see 3.2.21) PTS that is also clean (see 3.4.2) and negatable (see 3.3.1), weak normalization implies strong normalization. For dependent systems the technique runs into difficulties due to its use of the CPS translation—see [12]. In a nut-shell, the CPS-translation of a term involves the CPS-translation of a type of the term. If types may not contain terms, then we can define CPS-translation of types first. However, if types may contain terms, we must use a single translation working on both forms of objects, and—unfortunately—there is no guarantee that our definition is "well-founded," since a term may be smaller than some of its types.

It is possible to generalize further the notion of non-dependence. In this chapter we have considered the order $\leq_{\mathcal{A}}$ and made certain requirements relative to that. We might consider an order $\leq$ which extends $\leq_{\mathcal{A}}$ by relating sorts that are incomparable with respect to $\leq_{\mathcal{A}}$. For instance, Berardi's [16] formulation of the logic cube consists of the eight PTSs λS, where

(i) $\mathcal{S} = \{*^p, \Box^p, *^s, \Box^s\}$.

(ii) $\mathcal{A} = \{(*^s, \Box^s), (*^p, \Box^p)\}$.

(iii) $\mathcal{R}$ is given for each system in the table:

λPROP	$(*^p, *^p)$				
λPROP2	$(*^p, *^p)$	$(\Box^p, *^p)$			
λPROP$\underline{\omega}$	$(*^p, *^p)$		$(\Box^p, \Box^p)$		
λPROPω	$(*^p, *^p)$	$(\Box^p, *^p)$	$(\Box^p, \Box^p)$		
λPRED	$(*^p, *^p)$			$(*^s, *^p)$	$(*^s, \Box^p)$
λPRED2	$(*^p, *^p)$	$(\Box^p, *^p)$		$(*^s, *^p)$	$(*^s, \Box^p)$
λPRED$\underline{\omega}$	$(*^p, *^p)$		$(\Box^p, \Box^p)$	$(*^s, *^p)$	$(*^s, \Box^p)$
λPREDω	$(*^p, *^p)$	$(\Box^p, *^p)$	$(\Box^p, \Box^p)$	$(*^s, *^p)$	$(*^s, \Box^p)$

For these systems, one might define $s_1 < s_2$ for $s_1 \in \{*^p, \Box^p\}$, $s_2 \in \{*^s, \Box^s\}$. Note that with this understanding of the relation $<$, all of the above systems become stratified. With a slight modification of the notion of cleanliness and the associated technique for choosing types for fresh variables, one can use this idea to show that weak normalization implies strong normalization also for the systems λPRED$\underline{\omega}$ and λPREDω of Berardi's logic cube.

However, the extended technique does not work for the two systems λPRED and λPRED2: the sort $\Box^p$ is not negatable. Moreover, the extended technique does not work in any of the systems in the right hand side of Barendregt's [4] or Geuvers' [38, 37] version of the logic cube.[4] Finally, the

[4]Barendregt's version differs from Berardi's in two ways: in the first four systems the axiom $(*^s, \Box^s)$ is omitted—this is not an essential difference—and the last four systems have extra sort $*^f$ and rules $(*^s, *^s, *^f)$, $(*^s, *^f, *^f)$ violating persistence. Geuvers' version

extended technique does not apply to λC or the other systems in the right hand side of the λ-cube: stratification still fails.

Another way to extend the class of systems for which the Barendregt-Geuvers-Klop conjecture is true is to attack the problem from the other side: instead of extending our technique to prove

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

for increasingly large systems, we can show that $(*)$ for some systems follows from $(*)$ of smaller systems. Translations which eliminate dependent types, but preserve reductions [45, 39], might be generalized to classes of pure type systems with such applications in mind.

A problem related to the Barendregt-Geuvers-Klop conjecture is the so-called *K-conjecture* [8]. It states that for any PTS λS,

$$\lambda S \models \mathrm{SN}_\beta \;\Rightarrow\; \lambda^\kappa S \models \mathrm{SN}_{\beta\kappa},$$

where $\lambda^\kappa S$ is the system arising by addition of the rules

$$\frac{\Gamma \vdash_K A : B \qquad \Gamma \vdash_K C : D}{\Gamma \vdash_K \mathsf{K}\,A\,C : B}$$

$$\frac{\Gamma \vdash_K A : B \qquad \Gamma \vdash_K B' : s}{\Gamma \vdash_K A : B'} \qquad \textbf{if } B =_\kappa B'.$$

where K is a constant and $\to_\kappa$ and $=_\kappa$ are the obvious closures of the rule

$$\mathsf{K}\,A\,B \quad \kappa \quad A.$$

It seems that the techniques in this chapter can be used to solve the K-conjecture for the generalized non-dependent systems which are also clean and negatable. This will be addressed elsewhere.

differs from Berardi's in that the last two systems have the additional rule $(\Box^p, *^s)$ violating stratification. For more on the correspondence between traditional formulations of logics and formulations as pure type systems, and between the λ-cube and the logic-cube, see [4, 16, 131, 38, 37].

Bibliography

[1] S. Abramsky, D.M. Gabbay, and T.S.E. Maibaum, editors. *Handbook of Logic in Computer Science*, volume II. Oxford University Press, 1992.

[2] 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.

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

[4] H.P. Barendregt. Lambda calculi with types. In Abramsky et al. [1], pages 117–309.

[5] 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.

[6] H.P Barendregt, J.R. Kennaway, J.W. Klop, and M.R. Sleep. Needed reduction and spine strategies for the lambda calculus. *Information and Computation*, 75(3):191–231, 1987.

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

[8] G. Barthe. Extensions of pure type systems. In Dezani-Ciancaglini and Plotkin [32], pages 16–31.

[9] G. Barthe, J. Hatcliff, and M.H. Sørensen. Classical pure type systems. In S. Brookes, M. Main, A. Melton, and M. Mislove, editors, *Mathematical Foundations of Programming Semantics*, volume 6 of *Electronic Notes in Computer Science*. Elsevier, 1997.

[10] G. Barthe, J. Hatcliff, and M.H. Sørensen. CPS translations and applications: the cube and beyond. In O. Danvy, editor, *ACM SIGPLAN Workshop on Continuations*, number NS-96-13 in BRICS Notes Series, pages 4:1–31, 1997.

[11] G. Barthe, J. Hatcliff, and M.H. Sørensen. Weak normalization implies strong normalization in generalized non-dependent pure type systems. Submitted for publication, 1997.

[12] G. Barthe, J. Hatcliff, and M.H. Sørensen. CPS translations and applications: the cube and beyond. Submitted for publication, 1998. Extended version of [10].

[13] G. Barthe, J. Hatcliff, and M.H. Sørensen. An induction principle for pure type systems. Submitted for publication, 1998.

[14] 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.

[15] G. Barthe and M.H. Sørensen. Domain-free pure type systems. Submitted for publication, 1998. Extended version of [14].

[16] S. Berardi. *Type Dependence and Constructive Mathematics*. PhD thesis, Universita di Torino, 1990.

[17] J.A. Bergstra and J.W. Klop. Church-Rosser strategies in the lambda calculus. *Theoretical Computer Science*, 9:27–38, 1979.

[18] 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.

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

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

[21] C. Böhm, M. Coppo, and M. Dezani-Ciancaglini. Termination tests inside λ-calculus. In A. Salomaa and M. Steinby, editors, *λ-Calculus and Computer Science Theory*, volume 37 of *Lecture Notes in Computer Science*, pages 95–110. Springer-Verlag, 1975.

[22] C. Böhm and M. Dezani-Ciancaglini. λ-terms as total or partial functions on normal forms. In C. Böhm, editor, *λ-Calculus and Computer Science Theory*, volume 52 of *Lecture Notes in Computer Science*, pages 96–121. Springer-Verlag, 1975.

[23] V. Capretta and S. Valentini. A general method to prove the normalization theorem for first and second order typed λ-calculi. To appear in *Mathematical Structures in Computer Science.*

[24] A. Church. *The Calculi of Lambda-Conversion.* Princeton University Press, Princeton, N. J., 1941.

[25] A. Church and J.B. Rosser. Some properties of conversion. *Transactions of the American Mathematical Society*, 39:11–21, 1936.

[26] 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.

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

[28] Pierre-Louis Curien. Algèbre universelle, introduction au λ-calcul et aux logiques combinatoires (notes de cours). Technical Report LIENS-95-30, Ecole Normale Supériere, 1995.

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

[30] N.G. de Bruijn. A survey of the project AUTOMATH. In Seldin and Hindley [116], pages 579–606.

[31] P. de Groote. The conservation theorem revisited. In Bezem and Groote [19], pages 163–178.

[32] M. Dezani-Ciancaglini and G. Plotkin, editors. *Typed Lambda Calculus and Applications*, volume 902 of *Lecture Notes in Computer Science.* Springer-Verlag, 1995.

[33] 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.

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

[35] R.O. Gandy. An early proof of normalization by A.M. Turing. In Seldin and Hindley [116], pages 453–455.

[36] R.O. Gandy. Proofs of strong normalization. In Seldin and Hindley [116], pages 457–477.

[37] J.H. Geuvers. Conservativity between logics and typed lambda-calculi. In H. Barendregt and T. Nipkow, editors, *Types for Proofs and Programs*, volume 806 of *Lecture Notes in Computer Science*, pages 79–107. Springer-Verlag, 1993.

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

[39] J.H. Geuvers and M.J. Nederhof. A modular proof of strong normalization for the calculus of constructions. *Journal of Functional Programming*, 1(2):155–189, 1991.

[40] S. Ghilezan. Application of typed lambda calculi in the untyped lambda calculus. In A. Nerode and Yu.V. Matiyasevich, editors, *Symposium on Logical Foundations of Computer Science*, volume 813 of *Lecture Notes in Computer Science*, pages 129–139. Springer-Verlag, 1994.

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

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

[43] B. Gramlich. *Termination and Confluence Properties of Structured Rewrite Systems.* PhD thesis, Fachbereich Informatik der Universität Kaiserslautern, 1996.

[44] 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.

[45] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. In *Logic in Computer Science*, pages 194–204, 1987.

[46] 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.

[47] J. Hatcliff and O. Danvy. Thunks and the lambda calculus. Technical Report 95/3, Department of Computer Science, University of Copenhagen, 1995.

[48] J.R. Hindley. Reductions of residuals are finite. *Transactions of the American Mathematical Society*, 240:345–361, 1978.

[49] J.R. Hindley. BCK-combinators and linear λ-terms have types. *Theoretical Computer Science*, 64:97–105, 1989.

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

[51] F. Honsell and M. Lenisa. Some results on the full abstraction problem for restricted lambda calculi. In A.M. Borzyszkowski and S. Sokolowski, editors, *Symposium on Mathematical Foundations of Computer Science*, volume 711 of *Lecture Notes in Computer Science*, pages 84–104. Springer-Verlag, 1993.

[52] W. Howard. The formulae-as-types notion of construction. In Seldin and Hindley [116], pages 479–490.

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

[54] G. Huet and J.-J. Lévy. Call by need computations in non-ambiguous linear term rewriting systems. Preprint 359, INRIA, 1979.

[55] J.M.E. Hyland. A simple proof of the Church-Rosser theorem. Oxford University, 1973.

[56] B. Jacobs. Semantics of lambda-I and of other substructure calculi. In Bezem and Groote [19], pages 195–208.

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

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

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

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

[61] 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.

[62] 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.

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

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

[65] 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.

[66] 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.

[67] 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.

[68] Z. Khasidashvili. β-reductions and β-developments with the least number of steps. In P. Martin-Löf and G. Mints, editors, *International Conference on Computer Logic*, volume 417 of *Lecture Notes in Computer Science*, pages 105–111. Springer-Verlag, 1988.

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

[70] Z. Khasidashvili. Optimal normalization in orthogonal term rewriting systems. In C. Kirchner, editor, *Rewriting Techniques and Applications*, volume 690 of *Lecture Notes in Computer Science*, pages 243–258. Springer-Verlag, 1993.

[71] 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.

[72] Z. Khasidashvili. On higher order recursive program schemes. In S. Tison, editor, *Colloquium on Trees in Algebra and Programming*, volume 787 of *Lecture Notes in Computer Science*, pages 172–186. Springer-Verlag, 1994.

[73] Z. Khasidashvili and J. Glauert. Discrete normalization and standardization in deterministic residual structures. In S. Tison, editor, *Algebraic and Logic Programming*, volume 1139 of *Lecture Notes in Computer Science*, pages 135–149. Springer-Verlag, 1996.

[74] Z. Khasidashvili and M. Ogawa. Perpetuality and uniform normalization. In M. Hanus and J. Heering, editors, *Algebraic and Logic Programming*, volume 1298 of *Lecture Notes in Computer Science*, pages 240–255. Springer-Verlag, 1997.

[75] S.C. Kleene. Origins of recursive function theory. *Annals of the History of Computing*, 3(1):52–67, 1981.

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

[77] J.W. Klop. Term rewriting systems. In Abramsky et al. [1], pages 1–116.

[78] 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.

[79] Y. Komori. BCK-algebras and lambda calculus. In *Proceedings of the 10th Symposium on Semigroups*, pages 5–11. Josai University, Sakado, 1987.

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

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

[82] B. Lercher. Lambda-calculus terms that reduce to themselves. *Notre Dame Journal of Formal Logic*, XVII(2):291–292, 1976.

[83] J.-J. Lévy. *Réductions correctes et optimales dans le lambda-calcul.* PhD thesis, Université de Paris VII, 1978.

[84] J.-J. Lévy. Optimal reductions in the lambda-calculus. In Seldin and Hindley [116], pages 159–191.

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

[86] P.-A. Melliès. *Description Abstraite des Systèmes de Réécriture.* PhD thesis, Université Paris VII, 1996. Thèse de doctorat.

[87] 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.

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

[89] G. Mitschke. The standardization theorem in λ-calculus. *Zeitschrift für Mathematischen Logik und Grundlagen der Mathematik*, 25:29–31, 1979.

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

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

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

[93] R. Nederpelt, J.H. Geuvers, and R.C. de Vrijer, editors. *Selected Papers on Automath.* Elsevier Science B.V., 1994.

[94] V. van Oostrom. *Confluence for Abstract and Higher-Order Rewriting.* PhD thesis, Vrije Universiteit Amsterdam, 1994.

[95] V. van Oostrom. Take five. IR-406, Vrije Universiteit Amsterdam, 1996.

[96] V. van Oostrom. Finite family developments. In H. Comon, editor, *Rewriting Techniques and Applications*, volume 1232 of *Lecture Notes in Computer Science*, pages 308–322. Springer-Verlag, 1997.

[97] V. van Oostrom and F. van Raamsdonk. Comparing combinatory reduction systems and higher-order rewrite systems. In J. Heering, K. Meinke, B. Möller, and T. Nipkow, editors, *Higher Order Algebra,*

Logic and Term Rewriting, volume 816 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.

[98] M. Parigot. Internal labellings in lambda-calculus. In B. Rovan, editor, *Symposium on Mathematical Foundations of Computer Science*, volume 452 of *Lecture Notes in Computer Science*, pages 439–445. Springer-Verlag, 1990.

[99] D.A. Plaisted. Polynomial time termination and constraint satisfaction tests. In C. Kirchner, editor, *Rewriting Techniques and Applications*, volume 690 of *Lecture Notes in Computer Science*, pages 405–420. Springer-Verlag, 1993.

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

[101] 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.

[102] J. van de Pol. *Termination of Higher-Order Rewrite Systems.* PhD thesis, University of Utrecht, 1996. Volume 16 of Questiones Infinitae.

[103] J. van de Pol and H. Schwichtenberg. Strict functionals for termination proofs. In Dezani-Ciancaglini and Plotkin [32], pages 350–364.

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

[105] F. van Raamsdonk. *Confluence and Normalisation for Higher-Order Rewriting.* PhD thesis, Vrije Universiteit Amsterdam, 1996.

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

[107] F. van Raamsdonk, P. Severi, M.H.B. Sørensen, and H. Xi. Perpetual reductions in λ-calculus. *Information and Computation*, 1998. To appear.

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

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

[110] J.B. Rosser. Highlights of the history of the lambda-calculus. *Annals of the History of Computing*, 6(4):337–349, 1984.

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

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

[113] D.E. Schroer. *The Church-Rosser Theorem.* PhD thesis, Cornell University, 1965.

[114] 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.

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

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

[117] P. Severi. *Normalisation in Lambda Calculus and its relation to Type Inference.* PhD thesis, Eindhoven University of Technology, 1996.

[118] M.H. Sørensen. Embeddings and infinite reduction paths in untyped λ-calculus. Presented at the second International Workshop on Termination, La Bresse, France, 1995.

[119] 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.

[120] M.H. Sørensen. Properties of infinite reduction paths in untyped λ-calculus. In J. Ginzburg, Z. Khasidashvili, J.J. Lévy, E. Vogel, and E. Vallduví, editors, *Proceedings of the Tbilisi Symposium on Language, Logic, and Computation*, CLSI Lecture Notes, 1996. To appear.

[121] M.H. Sørensen. Strong normalization from weak normalization in typed λ-calculi. *Information and Computation*, 133(1):35–71, 1997.

[122] M.H. Sørensen. A note on shortest developments. Submitted for publication, 1998.

[123] J. Springintveld. Lower and upper bounds for reductions of types in $\lambda\underline{\omega}$ and λP. In Bezem and Groote [19], pages 391–405.

[124] R. Statman. A local translation of untyped λ calculus into simply typed λ calculus. Research Report 91-134, Carnegie-Mellon University, 1991.

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

[126] 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.

[127] M. Takahashi. Parallel reductions in λ-calculus. *Information and Computation*, 118:120–127, 1995.

[128] J. Terlouw. Reduction of higher type levels by means of an ordinal analysis of finite terms. *Annals of Pure and Applied Logic*, 28:73–102, 1985.

[129] J. Terlouw. Strong normalization in type systems: a model theoretical approach. *Annals of Pure and Applied Logic*, 73:53–78, 1995.

[130] J. Terlouw. A proof of strong normalization for generalized labeled β-reduction by means of Howard's successor relation method. Manuscript, Rijksuniversiteit Groningen, the Netherlands, February 1998.

[131] T. Tonino and K.-E. Fujita. On the adequacy of representing higher order intuitionistic logic as a pure type system. *Annals of Pure and Applied Logic*, 57:251–276, 1992.

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

[133] 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.

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

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

[136] R.C. de Vrijer. Exactly estimating functionals and strong normalization. *Koninklijke Nederlandse Akademie van wetenschappen*, 90(4), 1987. Also appeared as [137].

[137] R.C. de Vrijer. Exactly estimating functionals and strong normalization. *Indagationes Mathematicae*, 49:479–493, 1987.

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

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

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

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

[142] H. Xi. Separating developments. Manuscript, 1996.

[143] H. Xi. Upper bounds for standardization and an application. To appear in the *Journal of Symbolic Logic*. An earlier version appeared in the *Proceedings of the 5th Kurt Gödel Colloquium,* volume 1289 of *Lecture Notes in Computer Science,* pages 335–348, Springer-Verlag, 1997.

[144] H. Xi. Weak and strong beta normalisations in typed λ-calculi. In P. de Groote and J.R. Hindley, editors, *Typed Lambda Calculus and Applications*, volume 1210 of *Lecture Notes in Computer Science*, pages 390–404. Springer-Verlag, 1997.

Index