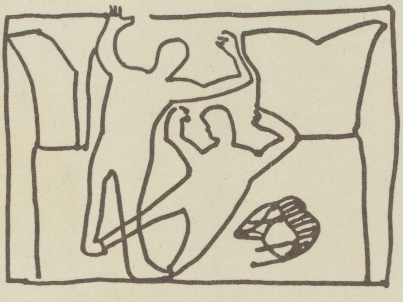
SURJECTIVE PAIRING AND STRONG NORMALIZATION:

TWO THEMES IN LAMBDA CALCULUS

by ROEL DE VRIJER



SURJECTIVE PAIRING AND STRONG NORMALIZATION:

TWO THEMES IN LAMBDA CALCULUS

SURJECTIVE PAIRING AND STRONG NORMALIZATION:

TWO THEMES IN LAMBDA CALCULUS

ACADEMISCH PROEFSCHRIFT

TER VERKRIJGING VAN DE GRAAD VAN DOCTOR IN DE WISKUNDE EN NATUURWETENSCHAPPEN AAN DE UNIVERSITEIT VAN AMSTERDAM, OP GEZAG VAN DE RECTOR MAGNIFICUS, DR S.K. THODEN VAN VELZEN, HOOGLERAAR IN DE FACULTEIT DER GENEESKUNDE, IN HET OPENBAAR TE VERDEDIGEN IN DE AULA DER UNIVERSITEIT (OUDE LUTHERSE KERK, SINGEL 411, HOEK SPUI) OP WOENSDAG 7 JANUARI 1987 TE 15.00 UUR PRECIES

DOOR

ROELOF CORNELIS DE VRIJER

GEBOREN TE EINDHOVEN

PROMOTOREN:	Prof.dr H.P. Barendregt
	Prof.dr W. Peremans
CO-PROMOTOR:	Dr D.H.J. de Jongh

druk: Offset Kanters, Alblasserdam

Uitgeverij Eburon
Postbus 2867
2601 CW Delft

ISBN 90-70879-63-8

CONTENTS

INTRODUCTION

This thesis consists of five independent syntactic studies, treating of various systems of typed and untyped λ-calculus. They have been arranged into five chapters in the reverse order in which they were written. Each of these chapters has its own introduction, with all the essential information, and its own references; so they can be read independently of this general introduction.

This introduction is divided into two parts. First, in order to set the scene, we make a few general remarks on the relation between Church-Rosser and strong normalization results, and in particular on the use of strong normalization in Church-Rosser proofs. Thereafter the respective papers are briefly sketched and provided with some additional comment.

For the discussion we need some terminology. Suppose we have a set T of terms with a one step reduction relation $\rightarrow$. Following Klop [1980] the structure $(T, \rightarrow)$ may be called an Abstract Reduction System. By $\twoheadrightarrow$ we denote the reflexive transitive closure of $\rightarrow$, and by $=$ the generated equivalence relation, called *conversion*. A finite or infinite sequence of terms $t_0 \rightarrow t_1 \rightarrow t_2 \rightarrow \ldots$, is called a *reduction sequence* (of t_0). If there exist no infinite reduction sequences of t, then t is called *strongly normalizing* (SN). If all terms are SN, then the reduction relation $\rightarrow$ itself is called strongly normalizing too. Finally, $\rightarrow$ is called *Church-Rosser* (CR) if

$$t \twoheadrightarrow s \ \& \ t \twoheadrightarrow r \Rightarrow \ (\exists t')(s \twoheadrightarrow t' \ \& \ r \twoheadrightarrow t'),$$

and *weakly Church-Rosser* (WCR) if

$$t \rightarrow s \ \& \ t \rightarrow r \Rightarrow \ (\exists t')(s \twoheadrightarrow t' \ \& \ r \twoheadrightarrow t').$$

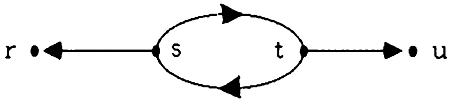
In syntactic work on the λ-calculus, Church-Rosser and strong normalization are the best known type of result. The original Church-Rosser theorem was the first method to establish the consistency of the *type free* λ-calculus. As a matter of fact, it remained essentially the only way to prove consistency for several decades, until the discovery of the lattice-theoretic models by Scott. For the *typed* systems there is a natural set-theoretic semantics and therefore the consistency is a priori unproblematic. In this case the importance of the Church-Rosser theorem lies rather in the fact that, when combined with a strong normalization result, it yields a decision method for testing the convertibility of two typed terms.

We now turn to the employment of SN in Church-Rosser proofs. Let us start considering those systems for which SN is already established (as it will be the case with most of the typed systems). In general their Church-Rosser problem is trivial, since it can be reduced to weak Church-Rosser by the implication

$$\mathrm{SN} + \mathrm{WCR} \Rightarrow \mathrm{CR}.$$

And usually WCR can be straightforwardly verified. The proof of the implication is easy and standard; it can be found e.g. in 3.1.8 of capter 1 below.

Note by the way that without SN the Church-Rosser property does not follow from WCR. The following simple counterexample is by Staples [1975].



Maybe a little surprisingly, the implication SN+WCR ⇒ CR can be exploited also for type free systems, where SN generally does not hold. We describe the procedure in an abstract setting.

LICENSED REDUCTION

The main step is devising some sort of *licensing* system that regulates reduction. In effect, a license ℓ attached to a term t has to contain the information:

(a) which reduction steps are allowed from t;

(b) if the step $t \to s$ is allowed by ℓ, which is the license that s inherits.

If $t \to s$ is allowed by ℓ and ℓ' is the license that s inherits under the indicated reduction step, we denote this by $(t,\ell) \to' (s,\ell')$. Furthermore we let $\twoheadrightarrow'$ denote the reflexive transitive closure of $\to'$ and define $\to_1$ by the clause:

(i) $t \to_1 s \;\Leftrightarrow\; (\exists\ell)(\exists\ell')((t,\ell) \twoheadrightarrow' (s,\ell'))$.

Note that by these definitions the following implications are obvious.

(ii) $(t,\ell) \to' (s,\ell') \;\Rightarrow t \to s$;

(iii) $t \to_1 s \;\Rightarrow t \twoheadrightarrow s$.

Now, in order that the licensing system be appropriate for the purpose of proving Church-Rosser for the original system, it is sufficient that the following additional requirements are met.

(iv) $t \to s \Rightarrow (\exists\ell)(\exists\ell')((t,\ell) \to' (s,\ell'))$;

(v) $(t,\ell_0) \twoheadrightarrow' (s,\ell'_0) \;\&\; (t,\ell_1) \twoheadrightarrow' (r,\ell'_1) \;\Rightarrow$

$$(\exists\ell)(\exists\ell')(\exists\ell'')((t,\ell) \twoheadrightarrow' (s,\ell') \;\&\; (t,\ell) \twoheadrightarrow' (r,\ell''));$$

(vi) $\to'$ is WCR;

(vii) $\to'$ is SN.

Finding the appropriate licenses to satisfy (iv) and (v) is called "lifting", forgetting about the licenses in (ii) "projecting".

CLAIM. If for $\rightarrow$ a licensing system can be found such that (iv) to (vii) are valid, then $\rightarrow$ is Church-Rosser.

PROOF. (The argument is standard; it can be found e.g. in section 3.1 of chapter 1, where the above method is actually applied.) First we establish the implication:

$(*)\qquad t \rightarrow_1 s \;\&\; t \rightarrow_1 r \Rightarrow (\exists t')(s \rightarrow_1 t' \;\&\; r \rightarrow_1 t')$.

Assume $t \rightarrow_1 s$ and $t \rightarrow_1 r$. Let ℓ_0, ℓ'_0 and ℓ_1, ℓ'_1 be such that $(t, \ell_0) \twoheadrightarrow' (s, \ell'_0)$ and $(t, \ell_1) \twoheadrightarrow' (r, \ell'_1)$ (by (i)). Then by (v) there are ℓ, ℓ', ℓ'', such that $(t, \ell) \twoheadrightarrow' (s, \ell')$ and $(t, \ell) \twoheadrightarrow' (r, \ell'')$. But by (vi) and (vii) the relation $\rightarrow'$ is CR, and hence there must be a common $\twoheadrightarrow'$-reduct (t', ℓ^*) of (s, ℓ') and (r, ℓ''). Consequently by (i) again we have $s \rightarrow_1 t'$ and $r \rightarrow_1 t'$. Thereby $(*)$ is established. Now from $(*)$ it follows immediately that $\rightarrow_1$ is CR. From this CR for $\rightarrow$ can be derived, since it follows by (i), (iii) and (iv) that the reflexive transitive closures of $\rightarrow$ and $\rightarrow_1$ coincide. □

APPLICATIONS

We mention three well-known cases of licensing systems for the λ-calculus to which the above applies.

A. DEVELOPMENTS. The oldest example of such a system uses the sets of redex occurrences in a term t as the licenses that can be attached to it. A reduction step is allowed by ℓ if it results from the contraction of a redex in ℓ. The inherited license is the set of residuals of the redexes in ℓ. The projections of the $\rightarrow'$-reduction sequences are usually called developments and the required strong normalization result is known as the Finite Developments theorem. We make use of the concept of developments in chapter 1. In chapter 2 a new proof of the Finite Developments theorem is given.

B. LABELLED REDUCTION. More recently, Hyland and Wadsworth, and Lévy have defined systems of so called labelled reduction, which have, apart from their use in proving CR, many other interesting applications. Here we only give a cursory definition of the Hyland/Wadsworth calculus, which was designed as a tool for studying the lattice-theoretic models. A Hyland/Wadsworth license for a term t can be represented as an assigment of natural numbers to all subterms of t. Then the contraction rule for licensed β-reduction is

$(\lambda x.t)^{n+1} s \rightarrow' ((x := s^n)t)^n$.

In addition an improper reduction rule is required, to get rid of multiple labels: $(t^n)^m \rightarrow t^{\min(n,m)}$. (More details can be found e.g. in Barendregt [1981].)

C. TYPES. One may even view the typed λ-calculus $\boldsymbol{\lambda}^{\tau}$ as a licensing system for the type free λ-calculus. Its licenses would consist of the typings and $\rightarrow'$ becomes just reduction within $\boldsymbol{\lambda}^{\tau}$. But, as reduction steps from an untypable term cannot be lifted, we do not have (iv). On the other hand, for typable terms t condition (iv) can be strengthened to:

(iv^{τ}) $\ t \twoheadrightarrow s \Rightarrow (\exists \ell)(\exists \ell')((t,\ell) \twoheadrightarrow' (s,\ell'))$.

The other requirements of licensed reduction that were formulated above do go through.

DEVIATING CASES

The above abstract method for proving Church-Rosser does not work in all situations, and as a matter of fact, two of the exceptions do play a prominent role in this thesis.

A. AUTOMATH. In the first place there are the Automath systems with η-reduction, of which the system $\boldsymbol{\lambda\lambda}$ in chapter 5 is an example. The problem is that whereas they are strongly normalizing all right, there is no easy way to establish WCR. This is caused by the fact that in these systems the types are themselves λ-expressions, also liable to reduction. Thus we have the following problematic example (in the un-Automathlike usual notation).

$\lambda x{:}\alpha.\,((\lambda y{:}\beta.\,y)\,x) \rightarrow_{\beta} \lambda x{:}\alpha.\,x$, and
$\lambda x{:}\alpha.\,((\lambda y{:}\beta.\,y)\,x) \rightarrow_{\eta} \lambda y{:}\beta.\,y$.

By the assumption that the terms are well-formed it follows that α and β are convertible into each other. But from this information a common reduct cannot be derived (unless we know CR already of course).

The Church-Rosser problem for Automath systems with η-reduction was solved by van Daalen [1980]. The proof makes essential use of strong normalization.

B. SURJECTIVE PAIRING. The second deviating case is the λ-calculus extended with surjective pairing, treated in chapter 1. In this case the usual reduction rules are WCR. As a consequence it has proved to be tempting to search for an appropriate licensing system. Such a system cannot be found however, as it has been shown by Klop that the λ-calculus with surjective pairing is not Church-Rosser.

SUMMARIES OF THE CHAPTERS

CHAPTER 1. Surjective pairing in connection with the λ-calculus was first studied by Barendregt. He showed that in the λ-calculus surjective pairing is not definable (see appendix 1 of chapter 1). However, it was Colin Mann who discovered in 1972 that there were difficulties in proving the Church-Rosser theorem for the extension of the λ-calculus with surjective pairing. What is meant here is the system (called $\boldsymbol{\lambda\pi^c}$ in chapter 1) that is obtained by adding to the λ-calculus constants π, π_0 and π_1 and the extra contraction rules

$$\begin{aligned}\pi_0(\pi XY) &\to X,\\ \pi_1(\pi XY) &\to Y, \text{ and}\\ \pi(\pi_0 X)(\pi_1 X) &\to X.\end{aligned}$$

The Church-Rosser problem for $\boldsymbol{\lambda\pi^c}$ remained the foremost open problem in λ-calculus, till in 1977 Klop found a counterexample. In the meantime it had become clear that a Church-Rosser proof would not be needed in order to establish the consistency: it is not difficult to find within the Pω-model of the λ-calculus elements that satisfy the requisite equations (see appendix 2 of chapter 1).

Yet the situation was not fully clarified. For, a syntactic consistency proof was in no way deemed impossible by the failure of CR. Moreover, it was not settled by the model construction whether surjective pairing is conservative over the λ-calculus. And another question arises: why stick to the reduction rules of $\boldsymbol{\lambda\pi^c}$? Do they have any intrinsic significance?

In chapter 1 we hope to put an end to all these questions by proving the conservativity of surjective pairing over the λ-calculus by purely syntactic means. In the proof we do not employ the reduction rules of $\boldsymbol{\lambda\pi^c}$. Instead we make use of several other auxiliary reduction relations. These were devised simply with the aim of making our proof work and we do not claim any a priori computational significance for them.

For an outline of the conservativity proof we refer to chapter 1, which has an extensive introduction of its own. Here we only point out some further connections.

It appears that interest in surjective pairing has for a good deal been inspired by category theory. The original motivation of Mann was the connection between category theory and proof theory. He didn't need the CR result he sought for after all, since he worked with typed systems. In their recent monograph, Lambek & Scott [1986] again study the connection between category theory and proof theory, and their C-monoids do correspond exactly to the type free λη-calculus with surjective pairing. Finally we should mention the work of Curien [1986] on a system called "Strong Categorical Combinatory Logic", which is designed for the implementation of functional programming languages. The system is also inspired by category theory and contains λ-calculus with surjective pairing.

CHAPTER 2. The Finite Developments theorem was already mentioned above. The proof which we present in this chapter is based on a seemingly naive attempt to produce an expression for the number of steps in a development of maximal length of a term with respect to a specified set of redexes. One finds such an expression by trying to imagine a reduction strategy that is as uneconomic as possible, never allowing a short cut if a longer route could be taken as well. The resulting strategy turns out to correspond to the so called "perpetual strategy", which in the λ-calculus with the usual β-reduction always finds an infinite reduction path if there is one. Besides that, there is not much to say about the proof and that might well be its merit. (This chapter has been published as de Vrijer [1985].)

CHAPTER 3. We prove strong normalization for typed λ-calculus and combinatory logic by a method similar to that of chapter 2, again by employing a maximally uneconomic reduction strategy. In our expression for the height $\mathrm{h}(t)$ of the reduction tree of a term t we employ a new type of so called "labeled" functionals. Let t and s be terms such that also the applicative term ts is well-formed. Then the labeled functional which is attached to t contains—apart from an expression for the height of t—all the information that is required to calculate the labeled functional of ts given that for s.

The expression for $\mathrm{h}(t)$ is presented as a valuation of the terms of the typed λ-calculus $\boldsymbol{\lambda}^{\boldsymbol{\tau}}$. In fact two such valuations are given. The first one only provides an upper bound for the height. But the corresponding proof of strong normalizaton—called the "quick proof"—is simple and transparent. Certainly more transparent than the ones involving a computability predicate. As a matter of fact, the quick proof provides some extra insight into the nature of the computability proofs. The computability predicate can be analyzed as abstracting from all the information—still explicit in the labeled functionals—that is not needed for a successful induction on the terms establishing strong normalization.

The second valuation yields the exact estimates for the height of reduction trees. Although the valuation itself is easily grasped, the proof that it actually does its job is rather complicated. This is compensated, however, by the fact that it gives more information than the known strong normalization proofs: viz. the uniformity result that is described in 2.3.4 and 2.3.5 of chapter 3.

It should be pointed out here that there exists a connection between our "quick proof", and the proof in Gandy [1980] which proceeds by way of an evaluation of the terms of $\boldsymbol{\lambda}^{\boldsymbol{\tau}}$ into a strictly monotonic fragment (called $\boldsymbol{\lambda}$-$\mathbf{I}^{+}$) of an extension of the calculus $\boldsymbol{\lambda}^{\boldsymbol{\tau}}$ itself. Our proof seems more perspicuous, since the valuation that is used in the quick proof is immediately recognizable as estimating an upper bound and the labeled functionals are tailor-made for that purpose. Another difference in the proofs is more superficial than it may appear at first sight, however. Instead of considering the monotonic functionals as semantic objects, as we do, Gandy presents his evaluation as a syntactic coding in the λ-I^{+}-calculus. In order to make sure that the terms of type 0 in that calculus actually convert to numerals, he then assumes a normalization result. This difference in the proof is not a substantial one though. The expressions we employ for labeled functionals can be regarded as syntactic objects as well. We would then have to define the resulting calculus explicitly, but that is straightforward. Conversely, one could conceive Gandy's λ-I^{+}-calculus as just a notational system for the class of strictly monotonic functionals of finite type—instead of as a system of syntactic objects on its own.

CHAPTER 4. We prove that the system $\mathbf{N\text{-}HA}^{\omega}{}_{\mathbf{p}}$ (i.e., typed λ-calculus or combinatory logic with a recursion operator and surjective pairing) is strongly normalizing. The proof is by a computability argument. This chapter is based on a privately circulated note, written in 1982 to answer a question posed by Lambek & Scott. They needed the result for their monograph [1986].

CHAPTER 5. This paper, published earlier as de Vrijer [1975], is a contribution to the proof theory ("language theory" in Automath jargon) of Automath, the family of languages for checking mathematical proofs originated by de Bruijn [1970]. Before commenting on the paper itself, we first make a few remarks on its background.

In the opinion of the author, the Automath languages should be viewed in the first place as an analysis—aiming at a formalization—of reasoning, more in particular: of the reasoning employed by mathematicians to convince themselves and others of their results. In this respect there is no big difference with similar projects of Leibniz, Frege, and Russell and Whitehead.

In the Automath project much stress has always been laid on two related aspects of feasibility:

(i) feasibility of the actual coding of reasoning;
(ii) feasibility of the checking of the code.

It is in realizing these requirements to a considerable extent that Automath distinguishes itself from other formalizations of logic.
Its achievements in this respect seem to depend on two basic features:

(i) the employment of a concept that has become known under the name "formulas as types";
(ii) incorporating in the systems a mechanism for handling abbreviations.

The formulas-as-types notion is explained in chapter 5. Anyway, it has become quite familiar by now (the notion also forms the basis of the systems of Martin-Löf [1975]). So there is no need to go into details here.

We only comment on one distinctive feature of some of those Automath languages that exploit the formulas/types analogy most fully. An example is AUT-QE, the language that was employed by van Benthem Jutting [1977] for his translation of Landau's "Grundlagen der

Analysis". We use Martin-Löf's set up as a comparison. (The notation used is that of chapter 5.)

By contrast with AUT-QE, in Martin-Löf's theories there is no complete parallel between objects and types: objects can be in applicative position, but types cannot. Accordingly, the rule of Martin-Löf that introduces applicative terms can be formulated as:

$t \in \alpha,\ f \in [x,\alpha]\beta(x) \ \vdash\ \ <t>f \in \beta(t)$.

In Automath one reckons with the possibility that the type γ of f is available only in the form of an abbreviation, or even that it was introduced as a mere parameter. In those cases there is no reason to suppose that the functional character of f is reflected in the actual syntactical form of γ. Therefore the corresponding rule should be given as:

$t \in \alpha,\ f \in \gamma,\ \gamma \in [x,\alpha]\underline{\text{type}} \ \vdash\ \ <t>f \in <t>\gamma$.

Here the statement $\gamma \in [x,\alpha]\underline{\text{type}}$ is to be interpreted as "γ is a type of functions with domain α".

As a matter of fact, the system **λλ** of chapter 5 has an even less restrictive rule:

$t \in \alpha,\ f \in \gamma \ \vdash\ \ <t>f \in <t>\gamma$.

Then the extra requirement on the functional character of γ is included in the definition of the legitimate fragment **λλ**-ℓ of **λλ** (cf. ch.V, 3.2.1 and 3.4.3).

Now, for a discussion of the results of chapter 5, we must say something more on the language theory of Automath. Its principle aims are:

(i) obtaining correct and tractable definitions of the various Automath languages;
(ii) establishing their essential proof-theoretical properties, such as SN, CR, closure of the well-formed expressions under the reduction rules of the system ("closure" for short) and decidability.

Historically, a first proof of normalization for an Automath system was given in 1971 by van Benthem Jutting. The first comprehensive treatment of an Automath system is given by Nederpelt [1973]. He defines a system **Λ** and proves for it SN, CR and decidability. Two questions were left open:

(i) closure for **Λ**;
(ii) CR for **Λ** extended with η-reduction.

It was already mentioned above that the second problem has been solved by van Daalen [1980]. It turned out that the closure problem for **Λ** could be reduced to another problem, viz. the well-foundedness of the so called "big trees". The big tree of a term is derived from its reduction tree by allowing, apart from the usual ones, as (improper) reduction steps also the following two operations:

(i) passing to the type of a term;
(ii) passing to an arbitrary subterm.

A system is said to satisfy the property BT, if for all its terms the big tree is well-founded (cf. section 1.3 of ch.5). The above mentioned reduction of the closure problem for $\mathbf{\Lambda}$ can now be rendered as:

$$\text{BT for } \mathbf{\Lambda} \Rightarrow \text{closure for } \mathbf{\Lambda}.$$

The system $\boldsymbol{\lambda\lambda}$ introduced in the paper in chapter 5 is a variant of AUT-QE, which was devised for the purpose of proof-theoretical study. Strangely enough, by the very definition of $\boldsymbol{\lambda\lambda}$, its closure problem is trivial. In spite of that BT is needed again, now for the decidability of $\boldsymbol{\lambda\lambda}$ and also to establish the soundness of the language definition employed. This is an intriguing aspect of the language theory of Automath: depending on how a system is defined, the problematic aspects shift to a different place. (The "instability of language theory" is discussed in van Daalen [1980]; there also different methods for defining Automath languages are compared.) In consequence, the problems of a particular Automath system cannot be considered to be solved to full satisfaction, until all the desirable results, SN, CR, closure and decidability, are established *together*. In chapter 5 this aim is achieved for the system $\boldsymbol{\lambda\lambda}$. The essential technical problem that had to be solved was BT.

It is claimed in chapter 5 that the method of proof employed for BT, using bookkeeping pairs, can also be applied to Nederpelt's system $\mathbf{\Lambda}$, thus solving its closure problem. This is actually accomplished in van Daalen [1980], ch.VII. There he gives also another proof of BT.

ACKNOWLEDGEMENT. I would like to thank H.P. Barendregt, L.S. van Benthem Jutting, D.H.J. de Jongh, W. Peremans and M. Pigge for the indispensable support they gave during the later stages of the preparation of this thesis.

REFERENCES

Barendregt, H.P. [1981], *The Lambda Calculus, its Syntax and Semantics* (North Holland).

Benthem Jutting, L.S. van [1977], *Checking Landau's "Grundlagen" in the Automath System*, Math. Centre Tracts 83, Amsterdam.

Bruijn, N.G. de [1970], 'The mathematical language Automath, its usage and some of its extensions', in: *Symposium on Automatic Demonstration*, IRIA, Versailles 1968, Springer Lecture Notes in Mathematics 125, p. 29-61.

Curien, P.-L. [1986], *Categorial Combinators, Sequential Algorithms and Functional Programming* (Pitman).

Daalen, D.T. van [1980], *The language theory of Automath*, dissertation, Technische Hogeschool Eindhoven.

Gandy, R.O. [1980], 'Proofs of Strong Normalization', in: *To H.B. Curry* (eds. J.P. Seldin and J.R Hindley), p. 457–477.

Klop, J.W. [1980], *Combinatory reduction systems*, Mathematical Center Tracts 129, Amsterdam.

Lambek, J. and P. Scott [1986], *Introduction to higher order categorical logic* (Cambridge Univərsity Press).

Mann, C.R. [1973], *Connections Between Proof Theory and Category Theory*, dissertation, Oxford University.

Martin-Löf, P. [1975], 'An intuitionistic theory of types, predicative part', in: *Logic Colloquium 1973* (eds. Rose and Sheperdson).

Nederpelt, R.P. [1973], *Strong normalization for a typed lambda calculus with lambda structured types*, dissertation Technische Hogeschool Eindhoven.

Staples, J. [1975], 'Church-Rosser Theorems For Replacement Systems', in: *Algebra and Logic*, Springer Lecture Notes in Mathematics 450, p.291-307.

Vrijer, R.C. de [1975], 'Big trees in a λ-calculus with λ-expressions as types'. In C. Böhm (ed.), *λ-Calculus and Computer Science Theory*, Lecture Notes in Computer Science 37 (Springer).

Vrijer, R.C. de [1985], 'A Direct Proof of the Finite Developments Theorem', *Journal of Symbolic Logic* 50, p.339-343.

1
EXTENDING THE LAMBDA CALCULUS WITH SURJECTIVE PAIRING IS CONSERVATIVE

§0. Preliminaries

This is a paper on extensions of pure λβ-calculus. The reader is assumed to have at least some elementary knowledge of the λ-calculus and some versedness in its basic syntactic techniques. As a reference Barendregt [1981] may be used, especially chapters 3 and 11 and the section on reduction diagrams in chapter 12. We will adopt the notations and terminology which are used there whenever possible. Acquaintance with the general theory of Combinatory Reduction Systems (CRS's) as developed in Klop [1980] will be useful.

In this preliminary section we briefly survey some basic notions and terminology, and stipulate some further notations and conventions to be used in the sequel.

0.1. The set of pure λ-terms, Λ, is constructed from a set of variables $x, y, z, \ldots$ by the term-forming operations abstraction $(\lambda x.M)$ and application (MN). In extensions certain constants or function symbols of positive arity may be added.

Terms are considered modulo α-equivalence. The symbol ≡ is used for (syntactic) identity of terms (modulo α-equivalence). The result of substituting M for x in N is denoted by $(x:=M)N$. We shall always assume the bound variables of N to be chosen such that no free variable of M becomes bound after the substitution. $\mathrm{FV}(M)$ denotes the set of variables in M.

0.2. A *context* C[ ] is a term with an open place (colloquially: "the hole"). $C[M]$ is the result of filling that open place with M.

With the help of contexts an exact representation can be given of occurrences. If M occurs somewhere in N, then $N \equiv C[M]$ for some context C[ ], where the hole in C[ ] indicates the position of this particular occurrence of M in N (there may be more). Formally we identify this occurrence with the pair $\Sigma = <M, C[\ ]>$. In informal discussion the second coordinate is most often suppressed and then one speaks loosely of "the occurrence M". Vice versa, we will sometimes say Σ when we mean the occurring term M instead of the occurrence. If $N \equiv C[M]$ for some M, then C[ ] is called a *subcontext* of N. And $C[M']$ is called the result of replacing the occurrence M in N by M'. It should be noted that there is a contrast here with substitution: free variables in M' may become bound in $C[M']$.

The *depth* of the occurrence $\Sigma = \langle M, C[\ ]\rangle$ is defined as the number of symbols in $C[\ ]$.

0.3. Let $\mathbf{r}$: $M \to N$ be a *contraction rule*, i.e. a relation between terms. The *one step reduction relation* corresponding to $\mathbf{r}$ (usual notation $\to_{\mathbf{r}}$) is the *compatible closure* of $\mathbf{r}$:

$$(M, N) \in \mathbf{r} \Rightarrow C[M] \to_{\mathbf{r}} C[N].$$

Our starting point in this paper is the system $\boldsymbol{\lambda}$ of pure λβ-calculus. There is but one contraction rule, namely that of β-reduction:

$$\boldsymbol{\beta}\colon\ (\lambda x. M)N \to (x := N)M.$$

If $M \to_{\mathbf{r}} N$ by the contraction of a redex Σ in M, then we may indicate this by the notation Σ: $M \to_{\mathbf{r}} N$ or $M \,{}^{(\Sigma)}\!\!\to_{\mathbf{r}} N$. The reduction sequence consisting of this single step is denoted by (Σ). In case Σ coincides with the whole term M itself, Σ is called a *main redex*, and the reduction step (Σ) a *main reduction*.

The *reduction relation* corresponding to $\mathbf{r}$ (denoted by $\twoheadrightarrow_{\mathbf{r}}$) is the reflexive and transitive closure of $\to_{\mathbf{r}}$. Many kinds of, mostly auxiliary, reduction relations will be defined in the sequel, with correspondingly more and less fancy notations. Always a version of $\overset{=}{\to}$ will denote the reflexive closure of the corresponding version of $\to$, and $\twoheadrightarrow$ the reflexive and transitive closure. As usual we write σ: $M \twoheadrightarrow N$ to indicate that σ is a $\to$-reduction sequence leading from M to N. Finally, if $\to_{\mathrm{a}}$ and $\to_{\mathrm{b}}$ are relations on terms, then $\to_{\mathrm{a}} + \to_{\mathrm{b}}$ will denote the concatenated relation:

$$M \to_{\mathrm{a}} + \to_{\mathrm{b}} N \Leftrightarrow (\exists P)(M \to_{\mathrm{a}} P \;\&\; P \to_{\mathrm{b}} N)$$

The +-sign is also used for concatenation of reduction sequences; $\sigma + \rho$ is defined, if the first term of ρ and the end term of σ are the same, as first σ and then ρ.

0.4. Just as in Combinatory Reduction Systems, our contraction rules induce ancestor/descendant relations between occurrences in an original and a reduced term, which are taken to be understood without further notice. *Residuals* are descendants of redexes.

Unfortunately not all extensions of $\boldsymbol{\lambda}$ that will turn up in this paper are CRS's, and so the general theory of CRS's cannot be applied without further ado. There seems to be place here for yet some more generalization.

The set of residuals of the redex Σ after reduction sequence ρ is denoted as usual by Σ/ρ, and if ρ consists only of the contraction of one redex Δ simply by Σ/Δ. If Σ and Δ are redexes in the same term M, then the elementary diagram of Σ and Δ exists if the reductions $(\Sigma)+(\Delta/\Sigma)$ and $(\Delta)+(\Sigma/\Delta)$ have the same end term— with the same residuals of possible other redexes in M. (This is not quite accurate, (Δ/Σ) may consist of several steps, the order of which is irrelevant, however.)

0.5. To facilitate reading we will as much as possible make use of diagram language. A diagram which consists partly of drawn and partly of interrupted lines stands for the conditional statement that if some terms satisfy the drawn part, then there exist terms that stand in such and such relations to these and to each other, as it is pictured in the whole diagram. Often it is unnecessary to name all terms figuring in a diagram. In such cases we prefer the variant without parameters, or with only those displayed which are actually referred to in the text.

0.6. Let ≥ be a reduction relation and = the convertibility relation which is generated by ≥. Then there are two versions of the Church-Rosser theorem which are easily seen to be equivalent.

0.6.1. CHURCH-ROSSER THEOREM VERSION 1. If $M \geq N$ and $M \geq P$, then there exists a Q, such that $N \geq Q$ and $P \geq Q$.

0.6.2. CHURCH-ROSSER THEOREM VERSION 2. If $M = N$, then there exists a Q, such that $M \geq Q$ and $N \geq Q$.

In diagram language:

0.6.1.

$$\begin{tikzcd} {} \arrow[r, dash, "\geq"] \arrow[d, dash, "\geq"'] & {} \arrow[d, dash, dotted, "\geq"] \\ {} \arrow[r, dash, dotted, "\geq"'] & {} \end{tikzcd}$$

0.6.2.

$$\begin{tikzcd} {} \arrow[rr, dash, "="] \arrow[dr, dash, dotted, "\geq"'] & & {} \arrow[dl, dash, dotted, "\geq"] \\ & {} & \end{tikzcd}$$

When referring to the Church-Rosser Theorem we will in general mean the second version.

A set of contraction rules or a (one step) reduction relation are called *Church-Rosser* if their compatible, reflexive and transitive closure satisfies the Church-Rosser theorem.

0.7. Two relations $\to_a$ and $\to_b$ are said to *commute* if the diagram

$$\begin{tikzcd} {} \arrow[r, dash, "\to_a"] \arrow[d, dash, "\to_b"'] & {} \arrow[d, dash, dashed, "\to_b"] \\ {} \arrow[r, dash, dashed, "\to_a"'] & {} \end{tikzcd}$$

holds. $\to_a$ has the *diamond property* if it commutes with itself. Notice that in this terminology the Church-Rosser property for a relation $\to_a$

boils down to the diamond property of its reflexive transitive closure $\twoheadrightarrow_a$.

It appears that proofs of commutativity, Church-Rosser and the like do often depend on a few elementary properties of relations. Some, e.g. the Hindley-Rosen lemma and the "strip" lemma, have become real λ-calculus classics. We find it convenient to have the following variant at our disposal.

0.7.1. Lemma. Let $\rightarrow_a$ and $\rightarrow_b$ be relations. Assume $\rightarrow'_a$ and $\rightarrow'_b$ to be extensions of $\rightarrow_a$ and of $\rightarrow_b$ respectively, such that $\rightarrow'_a \subseteq \twoheadrightarrow_a$ and $\rightarrow'_b \subseteq \twoheadrightarrow_b$. Moreover suppose that

$$\begin{tikzcd} {} \arrow[r, dash, "\rightarrow'_a"] \arrow[d, dash, "\rightarrow'_b"'] & {} \arrow[d, dash, dotted, "\twoheadrightarrow'_b"] \\ {} \arrow[r, dash, dotted, "\underset{=}{\rightarrow}'_a"'] & {} \end{tikzcd}$$

Then $\twoheadrightarrow_a$ and $\twoheadrightarrow_b$ commute.

Proof. First establish by successive inductions on $\twoheadrightarrow'_b$ and on $\twoheadrightarrow'_a$, that $\twoheadrightarrow'_b$ and $\twoheadrightarrow'_a$ commute. Then notice that $\twoheadrightarrow_a = \twoheadrightarrow'_a$ and $\twoheadrightarrow_b = \twoheadrightarrow'_b$. □

Noteworthy special cases are found by taking $\rightarrow_b$ itself for $\rightarrow'_b$ and either $\rightarrow_a$ or $\twoheadrightarrow_a$ for $\rightarrow'_a$.

0.7.2. Lemma. If one of the diagrams (i) or (ii) holds, then $\twoheadrightarrow_a$ and $\twoheadrightarrow_b$ commute.

(i)

$$\begin{tikzcd} {} \arrow[r, dash, "\rightarrow_a"] \arrow[d, dash, "\rightarrow_b"'] & {} \arrow[d, dash, dotted, "\twoheadrightarrow_b"] \\ {} \arrow[r, dash, dotted, "\underset{=}{\rightarrow}_a"'] & {} \end{tikzcd}$$

(ii)

$$\begin{tikzcd} {} \arrow[r, dash, "\twoheadrightarrow_a"] \arrow[d, dash, "\rightarrow_b"'] & {} \arrow[d, dash, dotted, "\twoheadrightarrow_b"] \\ {} \arrow[r, dash, dotted, "\twoheadrightarrow_a"'] & {} \end{tikzcd}$$

Proof. Immediate by 0.7.1. □

(i) is lemma 3.3.6 in Barendregt [1981], (ii) with $\rightarrow_a = \rightarrow_b$ is the so-called strip lemma. But most typically lemma 0.7.1 will be used in the sequel with $\rightarrow'_a$ some sort of simultaneous $\rightarrow_a$-reduction.

§1. Introduction

1.1. Let $\boldsymbol{\lambda\pi}$ be the extension of the pure λ-calculus, $\boldsymbol{\lambda}$ (cf. 0.3), with the constants π, π_0 and π_1 and with the following axioms, which express that π, with the projections π_0 and π_1, is a surjective pairing:

$$\pi_0(\pi XY) = X$$
$$\pi_1(\pi XY) = Y$$
$$\textbf{sur}:\quad \pi(\pi_0 X)(\pi_1 X) = X$$

The set of $\lambda\pi$-terms will be denoted by $\Lambda\boldsymbol{\pi}$, the pure λ-terms by Λ.

Reading the surjectivity axiom (**sur**) from right to left, it asserts that (any) X can be written as a pair. It is equivalent to the rule of π-extensionality

$$\boldsymbol{\pi}\textbf{-ext}:\ \pi_0 X = \pi_0 Y,\ \pi_1 X = \pi_1 Y \ \vdash\ X = Y,$$

saying roughly that any term is determined by its respective projections. The relation between axiom and rule is to be compared to that between the axiom of η-conversion

$$\boldsymbol{\eta}:\quad \lambda x.Mx = M, \text{ provided } x \text{ does not occur free in } M,$$

and the rule of (functional) extensionality:

$$\textbf{ext}:\ Mx = Nx \ \vdash\ M = N \quad (\text{x not free in } M \text{ and } N).$$

It has been stressed by Scott (e.g. in [1975b]) that $\boldsymbol{\eta}$ does not primarily express extensionality, but rather something that might be called "functionality": everything is a function. This is analogous to the meaning of **sur**: everything is a pair.

In pure λ-calculus no surjective pairing is definable. That is, there do not exist P, P_0 and P_1 in Λ, such that the above equations for P, P_0 and P_1 instead of π, π_0 and π_1 are derivable in $\boldsymbol{\lambda}$. This result is due to Barendregt [1974]. We give a short proof in appendix 1.

Nevertheless, $\boldsymbol{\lambda\pi}$ is consistent, as can be shown by the construction of models. The very straightforward definition of a graph model will be presented in appendix 2.

The question remains then, whether the consistency of $\boldsymbol{\lambda\pi}$ cannot also be established by purely syntactic means. In the case of $\boldsymbol{\lambda}$, the consistency follows from the Church-Rosser theorem. This road seems here to be blocked, however, because the reduction relation on $\Lambda\boldsymbol{\pi}$ which results if one adds to usual β-reduction the contraction rules:

$$\boldsymbol{\pi_0}:\quad \pi_0(\pi XY) \to X,$$
$$\boldsymbol{\pi_1}:\quad \pi_1(\pi XY) \to Y, \text{ and}$$
$$\boldsymbol{\pi^c}:\ \pi(\pi_0 X)(\pi_1 X) \to X,$$

is not Church-Rosser. An ingeneous counterexample was provided by Klop [1980]. I baptize the system with this reduction relation $\boldsymbol{\lambda\pi^c}$, for "classical" $\boldsymbol{\lambda\pi}$, as it seems to be taken for granted in most of the literature that $\boldsymbol{\pi_0}$, $\boldsymbol{\pi_1}$ and $\boldsymbol{\pi^c}$ are the natural derivatives of the axioms for surjective pairing. ($\boldsymbol{\lambda\pi^c}$ is called $\boldsymbol{\lambda}$+**SP** in Klop [1980].) We come back to this point in section 1.3.

1.2. Results. The present paper aims at a syntactic consistency proof for $\boldsymbol{\lambda\pi}$. In [1980] Scott remarks that proof theoretic results can be very sensitive to the exact formulation of the rules of the system under scrutiny. And with respect to $\boldsymbol{\lambda\pi^c}$: is there not some modified property of reduction relations that will imply that not all terms are interconvertible, and so do the job for the failing Church-Rosser property?

It may be noticed by the way that a candidate for such a property would be that of two normal forms being convertible only if they are syntactical identical. (In the presence of the Church-Rosser property a triviality.) This property is called UN—for Unicity of Normal forms—in Klop [1980]. Several systems that are not Church-Rosser, for example the system $\boldsymbol{\lambda\delta}$ which is to be discussed in 1.3 below, are proved there yet to satisfy UN. For $\boldsymbol{\lambda\pi^c}$, however, UN is still an open problem.

Anyhow, we will in fact supply $\boldsymbol{\lambda\pi}$ with a somewhat modified reduction relation, satisfying a different form of the Church-Rosser property. The resulting syntactic consistency proof is of course of a quite different nature than the model theoretic one. As one might expect, the syntactic proof gives also some extra information. Thus it will be an immediate corollary that $\boldsymbol{\lambda\pi}$ is a conservative extension of $\boldsymbol{\lambda}$.

The modified reduction relation we equip $\boldsymbol{\lambda\pi}$ with, is denoted by $\geq$. To contrast it with $\boldsymbol{\lambda\pi^c}$, the resulting system $(\Lambda\boldsymbol{\pi},\geq)$ will be called $\boldsymbol{\lambda\pi^{\geq}}$. In the defining rules the convertibility relation =, which was already defined in the first lines of section 1.1, is assumed.

1.2.1. DEFINITION. The reduction relation $\geq$ of $\boldsymbol{\lambda\pi^{\geq}}$ is the least compatible, reflexive and transitive relation on $\Lambda\boldsymbol{\pi}$, satisfying:

$\boldsymbol{\beta}$: $(\lambda x.M)N \geq (x:=N)M$;

$\boldsymbol{\pi_0}$: $\pi_0(\pi X_0 X_1) \geq X_0$;

$\boldsymbol{\pi_1}$: $\pi_1(\pi X_0 X_1) \geq X_1$;

$\mathbf{l}$: $\pi(\pi_0 X)Y \geq X$, provided that $\pi_1 X = Y$;

$\mathbf{r}$: $\pi Y(\pi_1 X) \geq X$, provided that $\pi_0 X = Y$.

l and **r** stand for "left" and "right". The conditions to which the rules **l** and **r** are subjected are given in terms of = and so independent of $\geq$.

One readily verifies that the equivalence relation generated by $\geq$ coincides with the convertibility relation = of $\boldsymbol{\lambda\pi}$. So there is in effect no need to distinguish conversion in $\boldsymbol{\lambda\pi^{\geq}}$ (or $\boldsymbol{\lambda\pi^c}$) from conversion in $\boldsymbol{\lambda\pi}$. Note that the rules **l** and **r** both imply the rule $\boldsymbol{\pi^c}$: $\pi(\pi_0 X)(\pi_1 X) \geq X$.

1.2.2. DEFINITION. By $\approx$ we denote the least compatible equivalence relation on $\Lambda\boldsymbol{\pi}$, satisfying the clause

$$X_0 = Y_0,\ \ X_1 = Y_1 \ \Rightarrow\ \pi X_0 X_1 \approx \pi Y_0 Y_1 .$$

EXAMPLES. One has e.g. $\pi_0(\pi((\lambda x.x)y)z) \approx \pi_0(\pi yz)$, and $\lambda y.\pi((\lambda x.x)y)z \approx \lambda y.\pi yz$, but not $(\lambda x.\pi xz)y \approx \pi yz$.

In effect, $\approx$ disregards replacement of occurrences of subterms under the influence of a π by convertible ones. Since there are no π's there, on Λ the relation $\approx$ is just syntactic identity ($\equiv$).

Now the Church-Rosser property for $\boldsymbol{\lambda\pi}^{\geq}$ will be established modulo $\approx$, that is, in the following form.

1.2.3. CLAIM. (CR/$\approx$). If $\boldsymbol{\lambda\pi} \vdash M = N$, then there exist $\approx$-equivalent Q_0 and Q_1, such that $M \geq Q_0$ and $N \geq Q_1$. (See the diagram.)

$$\begin{tikzcd}
M \arrow[r, dash, "="] \arrow[d, dotted, dash, "\geq"'] & N \arrow[d, dotted, dash, "\geq"] \\
Q_0 \arrow[r, dotted, dash, "\approx"'] & Q_1
\end{tikzcd}$$

The proof is complicated and it occupies the bulk of this paper. Our main results, the conservativity and hence the consistency of $\boldsymbol{\lambda\pi}$ then follow at once. For assume $\boldsymbol{\lambda\pi} \vdash M = N$ for $M, N \in \Lambda$. Find Q_0 and Q_1 as in the theorem above. Then, as $\geq$-reduction cannot introduce constants which were not already present, all terms on the reduction sequences $M \geq Q_0$ and $N \geq Q_1$ must be in Λ, in particular $Q_0, Q_1 \in \Lambda$. Hence $Q_0 \equiv Q_1$ and the reductions $M \geq Q_0$ and $N \geq Q_1$ can only use $\boldsymbol{\beta}$. So M and N are convertible in $\boldsymbol{\lambda}$ as well.

Thus we established as a corollary to 1.2.3:

1.2.4. MAIN RESULTS. (i) $\boldsymbol{\lambda\pi}$ is a conservative extension of $\boldsymbol{\lambda}$, i.e. if $M, N \in \Lambda$, then $\boldsymbol{\lambda\pi} \vdash M = N \Rightarrow \boldsymbol{\lambda} \vdash M = N$.
(ii) $\boldsymbol{\lambda\pi}$ is consistent.

The proof of theorem 1.2.3 is not the straightforward type of Church-Rosser proof, as e.g. one which proceeds by defining and then glueing together elementary diagrams. The traditional techniques do play an indirect role though, in the analysis of some auxiliary reduction systems.

The remainder of this introduction is meant to give some intuitive insight into the ideas behind the actual proof, which then further occupies the §§2, 3 and 4. The reader who prefers to do so may skip the informal part and move on directly to §2, keeping this introduction for casual reference. An outline of §§2 to 4 is presented in 1.9.

1.3. Digression. It was already mentioned in section 1.1 that the system $\boldsymbol{\lambda\pi^c}$ is not Church-Rosser. One of the complications that arises in an attempted proof stems from the fact that the metavariable X occurs twice in the π^c-redex $\pi(\pi_0 X)(\pi_1 X)$, thus causing the redex to be unstable under reduction in one of the X's. In CRS jargon: $\boldsymbol{\pi^c}$ is not left linear. Another complication lies in the ambiguity of the rules of $\boldsymbol{\lambda\pi^c}$: the rules $\boldsymbol{\pi_0}$ and $\boldsymbol{\pi^c}$, and $\boldsymbol{\pi_1}$ and $\boldsymbol{\pi^c}$ overlap.

The factor of non-left linearity is diagnosed to be the most serious one. In order to isolate this phenomenon, Hindley proposed in 1973 (cf. the sections on open problems in Böhm [1975] or Staples [1975]) to study the system $\boldsymbol{\lambda\delta}$ which results by extending $\boldsymbol{\lambda}$ with a single constant δ and the following simplified form of the π^c-rule:

$\boldsymbol{\delta}$: $\delta XX \to X$.

This system was proved by Klop to be not Church-Rosser. As a matter of fact the counterexample for Church-Rosser in $\boldsymbol{\lambda\pi^c}$ is a direct translation of that for $\boldsymbol{\lambda\delta}$.

1.3.1. All the same, $\boldsymbol{\lambda\delta}$ can very well serve as a toy system for illustrating some of the ideas which lie behind our main proof. Observe that the contraction rule $\boldsymbol{\delta}$: $\delta XX \to X$ may be conceived of as a restricted form of the more liberal rule:

$\mathbf{l}^{\boldsymbol{\delta}}$: $X = Y \vdash \delta XY \to X$,

which, in contrast to $\boldsymbol{\delta}$, is stable under reduction (i.e., a descendant of a $\mathbf{l}^{\boldsymbol{\delta}}$-redex is still a $\mathbf{l}^{\boldsymbol{\delta}}$-redex). It is easy to prove that $\mathbf{l}^{\boldsymbol{\delta}}$, in combination with $\boldsymbol{\beta}$, does satisfy the Church-Rosser property. Now, somewhat surprisingly, $\to$ can, without the Church-Rosser property being spoiled, be extended further by the rule,

$\mathbf{r}^{\boldsymbol{\delta}}$: $X = Y \vdash \delta XY \to Y$.

For under this further extension of $\to$ the convertibility relation generated remains the same. Hence the Church-Rosser result for $\boldsymbol{\lambda}\mathbf{l}^{\boldsymbol{\delta}}$ carries over immediately to $\boldsymbol{\lambda}\mathbf{l}^{\boldsymbol{\delta}}\mathbf{r}^{\boldsymbol{\delta}}$: a common reduct of $\lambda l^\delta r^\delta$-convertible terms can be found already by using only β- and l^δ-reduction.

There is a general principle at stake here:

1.3.2. Principle. The Church-Rosser problem for a more extended reduction relation can be reduced to a more restricted one, as long as the restricted reduction is strong enough to generate the original convertibility relation. Conversely: if addition of rules for reduction does not result in an extended convertibility relation, then by this extension the Church-Rosser property is conserved.

Our play system shows how this principle can be applied to an actual Church-Rosser problem. The a priori unclear case of $\boldsymbol{\lambda}\mathbf{l}^{\boldsymbol{\delta}}\mathbf{r}^{\boldsymbol{\delta}}$ (how to find a common reduct of X and Y under the in this respect rather uninforming assumption that $X = Y$?) could be reduced to the trivial one of $\boldsymbol{\lambda}\mathbf{l}^{\boldsymbol{\delta}}$. Furthermore notice that we have here an illustration of what can be attained by varying the rules of reduction. The systems $\boldsymbol{\lambda\delta}$, $\boldsymbol{\lambda}\mathbf{l}^{\boldsymbol{\delta}}$, and $\boldsymbol{\lambda}\mathbf{l}^{\boldsymbol{\delta}}\mathbf{r}^{\boldsymbol{\delta}}$ all have the same convertibility relation. Yet the latter two are Church-Rosser, while the first is not.

1.4. Back to $\boldsymbol{\lambda\pi}$. Cannot the same method be applied to $\boldsymbol{\lambda\pi}$? Indeed the rules $\mathbf{l}$ and $\mathbf{r}$ of $\boldsymbol{\lambda\pi}^{\boldsymbol{\geq}}$ are to a certain extent stabilized versions of the trouble causing contraction rule $\boldsymbol{\pi}^{\mathbf{c}}$: $\pi(\pi_0 X)(\pi_1 X) \geq X$. Left linearity is restored and the not quite unproblematic kind of overlap which the π_1-redex $\pi_1(\pi\, YZ)$ still has with the π^c-redex $\pi(\pi_0(\pi\, YZ))(\pi_1(\pi\, YZ))$ no longer exists in $\pi(\pi_0(\pi\, YZ))(\pi_1(\pi\, YZ))$ considered as an l-redex. (Or, for that matter, in $\pi(\pi_0 X))(\pi_1(\pi\, YZ))$ in case $X = \pi\, YZ$. The conditions on the contraction rules $\mathbf{l}$ and $\mathbf{r}$ guarantee that reduction does not lead across the borders of convertibility classes, these being determined independently of reduction by the definition of = in 1.1.)

In trying to establish Church-Rosser for $\boldsymbol{\lambda\pi}^{\boldsymbol{\geq}}$ minus $\mathbf{r}$, however, there is still a problem, owing to another case of overlap. For assume $\pi_1 X = Y$ and consider the following diagram.

$$\begin{tikzcd}
\pi_1(\pi(\pi_0 X)Y) \arrow[r, dash, "\geq"] \arrow[d, dash, "\geq"'] & \pi_1 X \arrow[d, dash, dotted] \\
Y \arrow[r, dash, dotted] & ?
\end{tikzcd}$$

Indeed $\pi_1(\pi(\pi_0 X)\, Y) \geq \pi_1 X$ by $\mathbf{l}$ and $\pi_1(\pi(\pi_0 X)\, Y) \geq Y$ by $\boldsymbol{\pi}_{\mathbf{1}}$. But now it is in general not at all clear how to find a common reduct of $\pi_1 X$ and Y. If we started by leaving out $\mathbf{l}$ instead of $\mathbf{r}$, then the same problem would arise of course, now with an r-redex under the influence of a π_0-projection. As a matter of fact the problem we just encountered forms a serious obstacle to the proof strategy we intend to follow—which is based on the ideas indicated in the digression under 1.3.1 and 1.3.2. How it will be solved is described below in the sections 1.5 ff.

It may be already pointed out here that the same sort of overlap with the other projection rule, as in $\pi_0(\pi(\pi_0 X)\, Y)$, is not quite so problematic. Whichever reduction rule is used, $\boldsymbol{\pi}_{\mathbf{0}}$ or $\mathbf{l}$, the result is X anyway. See further 1.8 below.

1.5. Tentative heuristics of the main proof. Recall that the rules $\mathbf{l}$ and $\mathbf{r}$ are both liberalized variants of $\boldsymbol{\pi}^{\mathbf{c}}$ and that therefore both would do to generate the intended convertibility relation = of $\boldsymbol{\lambda\pi}$. Even $\boldsymbol{\pi}^{\mathbf{c}}$ alone does that (with $\boldsymbol{\beta}$ and the projection rules of course). The only reason why $\boldsymbol{\pi}^{\mathbf{c}}$ can not altogether be replaced by say $\mathbf{l}$, is the possibility of clashes with $\boldsymbol{\pi_1}$ of the kind described above. But can we not both have the cake and eat it, by as a rule liberalizing $\boldsymbol{\pi^c}$ to $\mathbf{l}$, but in all cases that a π_1-clash threatens reinterpreting it as $\mathbf{r}$?

This rough proposal leads to a first approximation of our proof of the Church-Rosser property for $\geq$ of $\boldsymbol{\lambda\pi^{\geq}}$. Assume that M and N are convertible. Then there exists a conversion of M and N with, apart from $\boldsymbol{\beta}$, $\boldsymbol{\pi_0}$ and $\boldsymbol{\pi_1}$, only applications of the surjectivity rule $\boldsymbol{\pi^c}$. (That is, a conversion in $\boldsymbol{\lambda\pi^c}$.) Now, for the purpose of finding a common reduct of M and N, interpret applications of the surjectivity rule to redexes of which the residuals can be predicted to come under the influence of a π_1, as instances of $\mathbf{r}$, other ones as instances of $\mathbf{l}$.

Matters are slightly more complicated, though. For, why would the requisite choice between the rules $\mathbf{l}$ and $\mathbf{r}$ be uniform in the different residuals of one and the same redex? It is even clear that in general this is not the case; redexes and their residuals can be dispersed under reduction, some ending under the influence of a π_0, some under a π_1, some remaining "free" forever. To cope with this possibility of underdeterminedness of required information, we introduce in an extension of $\Lambda\boldsymbol{\pi}$ a new device (viz. that of bookkeeping pair), that will allow us to handle occurrences of redexes and their residuals simultaneously under different assumptions on the order in which projections will eventually act on them. A brief description of this device will be given in section 1.7. But first we need a formal tool that will enable us to manage the kind of information at issue.

1.6. Labels. In the auxiliary modifications of $\boldsymbol{\lambda\pi}$ that will be used in the proof of the Church-Rosser property, each subterm occurrence (and hence all redexes), Σ, of a given term M will be supplied with a label, $\ell(\Sigma)$, which consists of a sequence of zeros and ones. Roughly, a label represents partial information on the order in which the projection constants π_0 and π_1 act on subterm occurrences, or can be predicted to act on their descendants under reduction of M. Since the kind of information we have in mind can be derived from the context of the occurrence, the label of $\Sigma = \langle N, C[\ ]\rangle$ (i.e., the label of N in $C[N]$) will be defined completely in terms of $C[\ ]$ alone.

The partial information wich is coded in a label should be interpreted according to the following heuristic principle:

1.6.1. HEURISTIC PRINCIPLE. If the n'th digit of $\ell(\Sigma)$ is i, then the n'th element of a sequence of projections, of length at least n, acting on (any descendant of) Σ, will allways be π_i.

(NB. In $C[\pi_0(\pi_0(\pi_1 N))]$ three projections—or a sequence of projections of length three—are said to act on N, of which the first one is π_1 and the second and third ones are π_0.)

In case no sequence of projections of length n or more acts on Σ or any of its descendants, it depends only on the first n−1 digits of Σ's label whether it fulfills 1.6.1. This leaves room for some arbitrariness in the actual definition of labels, while adhering to the heuristic principle of interpretation. (E.g. for technical reasons we shall adopt the convention to assign the label 000... to any occurrence which is in applicative position.)

With this kind of information stored in the labels of subterm occurrences, restrictions on the rules **l** and **r** will be stipulated with the effect that problematic situations such as the one described in 1.4 above are avoided. E.g. an application of **r** will only be allowed to occurrences of $\pi Y(\pi_1 X)$ with a label of the form $\alpha = 1\alpha'$, this label being a guarantee that no π_0 will act on the actual occurrence or on one of its descendants. (As a consequence of the above mentioned convention, π-redexes in applicative position are only liable to contraction according to **l**.)

1.7. Bookkeeping pairs. Three kinds of contexts can be distinguished according to the above, namely those admitting of rule **l** (label of the form 0α), those admitting of rule **r** (label 1α), and those which do not (yet) carry enough information to settle the issue. Now in order to make sure that reduction is not necessarily obstructed in the latter case, the systems **λπp** and **λπp*** will be defined, incorporating as a formal device pairs of the form $\ulcorner M_0, M_1 \urcorner$, with the stipulation that M_0 and M_1 are treated as if their context provided them respectively with label 0 and with label 1. Formally: $\ell(M_0)=0$ and $\ell(M_1)=1$. In this manner e.g. a contraction of the π-redex $\pi(\pi_0 X)(\pi_1 X)$ can be simultaneously dealt with as an instance of **l** and of **r** in the respective left and right components of the "bookkeeping pair" $\ulcorner \pi(\pi_0 X)(\pi_1 X), \pi(\pi_0 X)(\pi_1 X) \urcorner$. And when in the process of reduction descendants of this pair end up in a context which is more determined, the bookkeeping pair can be cancelled: only that component is kept, of which the label is consistent with the extended information carried by the new context.

For technical reasons the formation of bookkeeping pairs is still to be generalized somewhat: in an α-context (label α) the indexed bookkeeping pair $\ulcorner M_0, M_1 \urcorner_\alpha$ is admitted, with the stipulations that $\ell(M_0)=\alpha 0$ and $\ell(M_1)=\alpha 1$.

Now the system **λπp** can be roughly characterized as based on the extension of **Λπ** with bookkeeping pairs, and incorporating the reduction rules **β**, $\boldsymbol{\pi_0}$, $\boldsymbol{\pi_1}$, **l** and **r**, along with a rule **p** which in an α-context allows an occurrence M to be replaced by the bookkeeping pair $\ulcorner M, M \urcorner_\alpha$. The system **λπp*** is derived from **λπp** by adding an appropriate mechanism for the cancellation of bookkeeping pairs in certain positions where they stand in the way rather than being of use.

Thereby we will have internalized, at a purely syntactic level, the possibility to deal with occurrences simultaneously under the alternative hypotheses of any finite amount of information concerning their context, as it is coded into the binary tree of labels <L, ≤>. All this luxury has its price though, consisting in a considerable growth of the complexity of the systems of reduction that must be investigated. More in particular, we will be confronted with the problem that the bookkeeping pairs, once allowed into our system, may pop up in places where they stand in the way rather than that they are much of a help. Such is the case e.g. in a term like $\ulcorner \lambda x.x, \lambda x.x \urcorner y$, in which the potential redex $(\lambda x.x)y$ is obstructed by the presence of the bookkeeping pair.

1.8. Some further technicalities. In the systems **λπp** and **λπp*** two kinds of overlap between the intricately interwoven contraction rules **l** and **r** on the one hand and $\boldsymbol{\pi_0}$ and $\boldsymbol{\pi_1}$ on the other, remain. These are illustrated in the following two diagrams. The first one under the provision that $\pi_1 X = Y$, the second by the restrictions on **l** only in a 0α-context and under the provision that $\pi_1(\pi XY) = Z$, that is, $Y = Z$.

(i)
$$\begin{tikzcd}
\pi_0(\pi(\pi_0 X)Y) \arrow[r, "\rightarrow_1"] \arrow[d, "\rightarrow_{\pi_0}"'] & \pi_0 X \arrow[d, dotted, no head] \\
\pi_0 X \arrow[r, dotted, no head] & ?
\end{tikzcd}$$

(ii)
$$\begin{tikzcd}
\pi(\pi_0(\pi XY))Z \arrow[r, "\rightarrow_{\pi_0}"] \arrow[d, "\rightarrow_1"'] & \pi XZ \arrow[d, dotted, no head] \\
\pi XY \arrow[r, dotted, no head] & ?
\end{tikzcd}$$

In both diagrams the indicated reductions result by contractions of two overlapping redexes, the contraction of the one leaving no residual of the other in the respective end terms. We now briefly comment on the diagrams seperately.

(i) This type of overlap is quite unproblematic. For reducing $\pi_0(\pi(\pi_0 X)\, Y)$ either with **l** or with $\boldsymbol{\pi_0}$, each time the reduct is $\pi_0 X$. In the theory of CRS's, systems in which there is only this harmless kind of overlap are called weakly non-ambiguous.

(ii) Here the situation seems to be more serious. How to find a common reduct of the respective one step reducts πXY and πXZ under the single assumption that Y and Z are convertible? We are in the same kind of deadlock that we met in 1.3.

Our solution is so drastic that it may look a bit too cheap: we declare πXY and πXZ to be equivalent (under the given assumptions) and thereby just stop worrying. In section 2.3 the equivalence relation ~ will be defined, which disregards up to convertibility the component X_i of the pair $\pi X_0 X_1$ in contexts with label jα, j≠i. Then in **λπp** and **λπp*** terms will be considered modulo ~.

Let it be just ascertained here that, just as in the case of **l** and **r**, the context restrictions give protection against the possibility of interference of a wrong projection rule. (Matters would not work out all right if we declared $\pi_1(\pi XY)$ to be $\sim$-equivalent to $\pi_1(\pi XZ)$ on the condition that $Y=Z$, for reasons similar to those described in 1.4 and 1.5 above with respect to $\mathbf{l}$ and $\mathbf{r}$.) Notice that $\sim$ can be seen as a refinement of the equivalence relation $\approx$ on $\Lambda\pi$ defined in 1.2.2. There is a parallel with the rules $\mathbf{l}$ and $\mathbf{r}$ again: in $\boldsymbol{\lambda\pi}$ we do not care so much about contexts.

The decision to work modulo $\sim$ could be debated. By considering terms as equivalence classes they certainly become, so to speak, heavier to carry around. On the other hand, one is to a certain extent already accustomed to this kind of proceeding, as one normally deals with α-equivalence in exactly the same way. The alternative would be to incorporate $\sim$ in the reduction rules. As a matter of fact that is what one did with α-equivalence in some of the early λ-calculus papers. It had its drawbacks too. Anyway, by our choice the reduction relations—already cumbersome enough—are not needlessly complicated still further and kept as much as possible in line with those of the more common systems (the paradigm being that of the regular, or at least weakly regular, Combinatory Reduction Systems).

1.9. Outline. It will have become clear by now that two auxiliary systems play an important role in this paper. Viz.

$\boldsymbol{\lambda\pi}^{\geq} = (\Lambda\pi, \geq)$, and
$\boldsymbol{\lambda\pi}\mathbf{p}^{*} = (\Lambda\pi\mathbf{p}^{*}, \rightarrow *)$.

Of these, the function of $\boldsymbol{\lambda\pi}^{\geq}$ has been pointed out already in section 1.2. There it was shown how the Church-Rosser theorem $(/\approx)$ for $\boldsymbol{\lambda\pi}^{\geq}$ (claim 1.2.3) would enable us to derive our main results, the conservativity of $\boldsymbol{\lambda\pi}$ over $\boldsymbol{\lambda}$ and the consistency of $\boldsymbol{\lambda\pi}$ (cf. 1.2.4). The system $\boldsymbol{\lambda\pi}\mathbf{p}^{*}$ can best be accounted for by referring to our digression in 1.3. In the above sections $\boldsymbol{\lambda\pi}\mathbf{p}^{*}$ was arrived at in an attempt to design a variant of $\boldsymbol{\lambda\pi}^{\geq}$ such that

(i) the modified reduction relation would be a restriction of that of $\boldsymbol{\lambda\pi}^{\geq}$;
(ii) the conversion relation of $\boldsymbol{\lambda\pi}^{\geq}$ would be retained;
(iii) we would be able to prove Church-Rosser for the restricted reduction relation.

This in order to be able to use the principle formulated in 1.3.2 for deriving the Church-Rosser theorem for $\boldsymbol{\lambda\pi}^{\geq}$. Now, since we had to introduce labels and bookkeeping pairs in $\boldsymbol{\lambda\pi}\mathbf{p}^{*}$, matters have become more complicated than they were in the case of $\boldsymbol{\lambda}\mathbf{l}^{\delta}\mathbf{r}^{\delta}$ and $\boldsymbol{\lambda}\mathbf{l}^{\delta}$. Yet it will turn out that the pattern of reasoning that was illustrated in 1.3.1 can be used for $\boldsymbol{\lambda\pi}^{\geq}$ and $\boldsymbol{\lambda\pi}\mathbf{p}^{*}$ as well.

In order to translate terms from $\Lambda\pi$ into $\Lambda\pi\mathbf{p}^{*}$ and back we make use of an imbedding ψ: $\Lambda\pi \rightarrow \Lambda\pi\mathbf{p}^{*}$ and a projection ϕ: $\Lambda\pi\mathbf{p}^{*} \rightarrow \Lambda\pi$. Both can be defined in a straightforward and natural way. Further we let $=^*$ denote conversion in $\boldsymbol{\lambda\pi p^*}$. Then in the remainder of this paper the following three statements will be established, matching in that order with the requirements (i), (ii) and (iii) that were formulated above.

1.9.1. CLAIM.
(i) If $M \in \Lambda\boldsymbol{\pi}$, then
$\boldsymbol{\lambda\pi p^*} \vdash \psi(M) \to^* N \;\Rightarrow\; (\exists N' \in \Lambda\boldsymbol{\pi})(\boldsymbol{\lambda\pi^{\geq}} \vdash M \geq N' \;\&\; N' \approx \phi(N))$;
(ii) $\boldsymbol{\lambda\pi} \vdash M = N \;\Rightarrow\; \boldsymbol{\lambda\pi p^*} \vdash \psi(M) =^* \psi(N)$;
(iii) $\boldsymbol{\lambda\pi p^*}$ is Church-Rosser.

From these facts one easily derives CR/$\approx$ for $\boldsymbol{\lambda\pi^{\geq}}$ (claim 1.2.3) as follows.

1.9.2. COROLLARY. $\boldsymbol{\lambda\pi} \vdash M = N \;\Rightarrow\; (\exists K' \approx K'')(M \geq K' \;\&\; N \geq K'')$.
PROOF. Suppose that $\boldsymbol{\lambda\pi} \vdash M = N$. Then by 1.9.1(ii) also $\boldsymbol{\lambda\pi p^*} \vdash \psi(M) =^* \psi(N)$ and it follows by 1.9.1(iii) that $\psi(M)$ and $\psi(N)$ must have a common $\to^*$-reduct K. We can then apply 1.9.1(i) (twice) to find $K', K'' \in \Lambda\boldsymbol{\pi}$, such that $M \geq K'$, $N \geq K''$ and $K' \approx K'' \approx \phi(K)$. □

A global outline of the actual proof of 1.2.3 can now be given, as each of the §§2, 3 and 4 corresponds roughly with one of the three propositions that together make up 1.9.1.

The principle object of §2 is defining the systems $\boldsymbol{\lambda\pi p}$ and $\boldsymbol{\lambda\pi p^*}$. After some preperation in the sections 2.1/4, these definitions are finally accomplished in 2.5. Proposition 1.9.1(ii) will be an immediate consequence (lemma 2.5.4).

§3 is entirely devoted to proving the Church-Rosser theorem for $\boldsymbol{\lambda\pi p^*}$ (proposition 1.9.1(iii)). The structure of the proof is pointed out at the beginning of that section.

In §4 proposition 1.9.1(i) is established. Roughly, what has to be done is eliminating the bookkeeping pairs from the $\to^*$-reduction sequence. This involves essentially a postponement argument: we need the fact that reduction steps that consist in the contraction of a redex lying completely within a bookkeeping pair, can be moved to the end of a reduction sequence.

§2. The systems with bookkeeping pairs: $\boldsymbol{\lambda\pi}\mathbf{p}$ and $\boldsymbol{\lambda\pi}\mathbf{p}^*$

The definition of the sets of labelled terms $\Lambda\boldsymbol{\pi}\mathbf{p}$ and $\Lambda\boldsymbol{\pi}\mathbf{p}^*$, corresponding to the theories $\boldsymbol{\lambda\pi}\mathbf{p}$ and $\boldsymbol{\lambda\pi}\mathbf{p}^*$, proceeds in a number of consecutive steps, departing from $\Lambda\boldsymbol{\pi}$. We first point out the several stages of the definition, by summing up the notions and the intermediate sets of terms that will be met on the way.

(i) The set of pre-terms $\Lambda^\circ\boldsymbol{\pi}\mathbf{q}$ is defined as the extension of $\Lambda\boldsymbol{\pi}$ with bookkeeping pairs and labels.

(ii) $\Lambda\boldsymbol{\pi}\mathbf{q}$ is defined as $\Lambda^\circ\boldsymbol{\pi}\mathbf{q}/{\sim}$, where the equivalence relation $\sim$ is the refinement of $\approx$ that was mentioned in 1.8.

(iii) Then, finally, $\Lambda\boldsymbol{\pi}\mathbf{p}$ and $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ can be obtained as subsets of $\Lambda\boldsymbol{\pi}\mathbf{q}$, by putting restrictions on the positions where bookkeeping pairs are allowed to occur, depending on their index and the label of the context of occurrence.

As a matter of fact, the restrictions to $\Lambda\boldsymbol{\pi}\mathbf{p}$ and $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ will turn out not to be affected by $\sim$. That is, the situation that will have been attained at the end of section 2.4 can be pictured in the following commuting diagram.

$$\begin{tikzcd}
L\times\Lambda\boldsymbol{\pi} \arrow[r, phantom, "\subseteq"] & \Lambda^\circ\boldsymbol{\pi}\mathbf{p}^* \arrow[r, phantom, "\subseteq"] \arrow[d, phantom, "/\sim"] & \Lambda^\circ\boldsymbol{\pi}\mathbf{p} \arrow[r, phantom, "\subseteq"] \arrow[d, phantom, "/\sim"] & \Lambda^\circ\boldsymbol{\pi}\mathbf{q} \arrow[d, phantom, "/\sim"] \\
 & \Lambda\boldsymbol{\pi}\mathbf{p}^* \arrow[r, phantom, "\subseteq"] & \Lambda\boldsymbol{\pi}\mathbf{p} \arrow[r, phantom, "\subseteq"] & \Lambda\boldsymbol{\pi}\mathbf{q}
\end{tikzcd}$$

2.1. Pre-terms and labels

2.1.1. DEFINITION. The set L of *labels* consists of all finite sequences of the symbols 0 and 1 (including the empty sequence), and the infinite sequences that become eventually constant 0.

The empty sequence is denoted by $<>$, the symbol ∞ is used for the infinite constant sequence 000..., satisfying the recursion equation $\infty = 0\infty$.

On L a partial order is defined by:

2.1.2. DEFINITION. $\alpha \lesssim \beta \Leftrightarrow \exists\gamma.\ \beta = \alpha\gamma$.

$\alpha \lesssim \beta$ will sometimes be expressed by saying that β *extends* α. The relation $\lesssim$ is obviously reflexive and transitive and moreover one easily verifies:

(i) $<> \lesssim \alpha$, for every α

(ii) $\beta_1 \lesssim \beta_2 \Rightarrow \alpha\beta_1 \lesssim \alpha\beta_2$

The purpose of labels, coding partial information on contexts, was explained in 1.6. The finite elements of L will also serve as indexes for the bookkeeping pairs. These are in:roduced in the following definition.

2.1.3. DEFINITION. Consider the extension of $\Lambda\boldsymbol{\pi}$ which is obtained by adding to its rules of term formation, λ-abstraction and application, the extra rule to construct from M_0 and M_1 for each finite label α (here called the *index*), the *bookkeeping pair* $\ulcorner M_0, M_1 \urcorner_\alpha$. The empty index is called *neutral*. We agree to use for neutral bookkeeping pairs the notation $\ulcorner M_0, M_1 \urcorner$ instead of $\ulcorner M_0, M_1 \urcorner_{<>}$.

Define the function ϕ, which maps terms with bookkeeping pairs back to $\Lambda\boldsymbol{\pi}$ by deleting all second coordinates, by induction:

$$\begin{aligned} \phi(M) &\equiv M, \text{ if } M \text{ is a variable or a constant} \\ \phi(\lambda x.M) &\equiv \lambda x.\phi(M) \\ \phi(MN) &\equiv \phi(M)\phi(N) \\ \phi(\ulcorner M_0, M_1 \urcorner_\alpha) &\equiv \phi(M_0) \end{aligned}$$

Then:

(i) the set of *pre-terms* is obtained by restricting the bookkeeping pair forming rule to the condition that $\boldsymbol{\lambda\pi} \vdash \phi(M_0) = \phi(M_1)$;

(ii) the set $\Lambda^{\circ}\boldsymbol{\pi q}$ of *labelled pre-terms* consists of the pairs $<\alpha, M>$, where $\alpha \in \mathrm{L}$ is a label and M a pre-term.
Notation: $\alpha.M$.

EXAMPLES. $\lambda x.\ulcorner x, x \urcorner_\alpha$, $\lambda x.\ulcorner x, (\lambda y.y)x \urcorner_\alpha$, $\ulcorner \ulcorner x, x \urcorner_\alpha, (\lambda y.y)x \urcorner_\alpha$ and $\lambda x.\ulcorner x, (\lambda y.y)\ulcorner x, x \urcorner_\beta \urcorner_\alpha$ are pre-terms. But $\ulcorner x, y \urcorner_\alpha$ and $(\lambda y.\ulcorner x, y \urcorner_\alpha)x$ are not.

Note that ϕ is idempotent and that $\Lambda\boldsymbol{\pi}$ is the set of its fixed points. ϕ can also be extended to the set of labelled pre-terms $\Lambda^{\circ}\boldsymbol{\pi q}$ by simply putting $\phi(\alpha.M) = \phi(M)$. Conversely the function ψ: $\Lambda\boldsymbol{\pi} \to \Lambda^{\circ}\boldsymbol{\pi q}$ defined by $\psi(M) = <>.M$ is a natural embedding of $\Lambda\boldsymbol{\pi}$ into $\Lambda^{\circ}\boldsymbol{\pi q}$.

2.1.4. DEFINITION. (i) The convertibility relation of $\boldsymbol{\lambda\pi}$ is extended to the set of pre-terms by adding the rule:

$\ulcorner M_0, M_1 \urcorner_\alpha = M_0$ (of course provided that $\ulcorner M_0, M_1 \urcorner_\alpha$ is a well-formed pre-term).

(ii) Conversion in $\boldsymbol{\lambda^{\bullet}\pi q}$ just neglects the labels, i.e., we define

$\boldsymbol{\lambda^{\bullet}\pi q} \vdash \alpha.M = \beta.N \Leftrightarrow M = N$ according to (i).

It is important to be aware that conversion is independent of labels. Moreover we have for $M, N \in \Lambda\boldsymbol{\pi}$ that

$\boldsymbol{\lambda^{\bullet}\pi q} \vdash \alpha.M = \beta.N \Leftrightarrow \boldsymbol{\lambda\pi} \vdash M = //$.

In the sequel convertibility either in $\boldsymbol{\lambda\pi}$ or according to one of the clauses of definition 2.1.4 will be indicated by a mere equation $M = N$. Here M and N can be terms either with or without bookkeeping pairs or labels.

From the substitutivity of **λπ** follows that Λ°**πq** is closed under substitution and that also conversion in **λ•πq** is substitutive.

EXAMPLES. We have $\lambda x.\lceil x, x\rceil_\alpha = \lambda x.\lceil x, (\lambda y.y)x\rceil_\alpha = \lambda x.x$. Also $\lceil y, y\rceil_\alpha = y$; but not $(\lambda x.\lceil y, x\rceil_\alpha)y = y$, since the lefthand term is not well-formed.

If C[ ] is a subcontext of M, then as a subcontext of $\alpha.M$ it is officially denoted by α.C[ ]. However, we adopt the convention to omit labels whenever this is possible without causing confusion, that is, both when the label α is already known and when it is not relevant for the discussion. So, we can without danger of being misunderstood refer to α.C[ ] as the subcontext C[ ] of $\alpha.M$.

The label $\ell(\Sigma)$ of the occurrence $\Sigma = \langle N, \mathrm{C}[\ ]\rangle$ of N in $\alpha.M$ will depend on α and C[ ] (not on N). Therefore we first define the function ℓ which assigns to each subcontext C[ ] of $\alpha.M$ a label $\ell(\mathrm{C}[\ ])$, indicating the position of the hole. Then $\ell(\Sigma)$ is simply identified with $\ell(\mathrm{C}[\ ])$. The notation $\mathrm{C}[\ ]_\beta$ is used if we want to indicate (implicitly) that $\ell(\mathrm{C}[\ ]) = \beta$. Such a context is called a β-context. (So $\alpha.\mathrm{C}[\ ]_\beta$ is a β- and not an α-context.)

It has been pointed out already that in a following step towards the definition of **λπp**-terms, equivalence classes of pre-terms will be formed by disregarding certain occurrences, depending on their position. For these occurrences the label will remain undefined, and naturally the same applies to the labels of the contexts related to the positions in question. Undefined labels are indicated by the symbol ↑. Occurrences without a defined label are called *transient*.

2.1.5. DEFINITION. The label of the hole in α.C[ ], notation $\ell(\mathrm{C}[\ ])$, is defined by induction on (the number of symbols in) C[ ]. In advance we stipulate that subcontexts of contexts with undefined label also have their label undefined: if $\ell(\mathrm{D}[\ ]) = \uparrow$, then $\ell(\mathrm{D}[\mathrm{E}[\ ]]) = \uparrow$ as well. Other cases are taken care of by the following clauses:

$$\begin{array}{rll}
\ell(\alpha.[\ ]) & = \alpha & \\
\ell(\mathrm{D}[\lambda x.[\ ]]) & = \langle\rangle & \\
\ell(\mathrm{D}[[\ ]Q]) & = \infty & \\
\ell(\mathrm{D}[\pi_i[\ ]]_\beta) & = i\beta, & \text{for } i = 0 \text{ or } 1 \\
\ell(\mathrm{D}[\pi[\ ]Q]_\beta) & = \gamma, & \text{if } \beta = 0\gamma \\
 & \ \uparrow, & \text{if } \beta = 1\gamma \\
 & \ \langle\rangle, & \text{if } \beta = \langle\rangle
\end{array}$$

$$
\begin{aligned}
\ell(\mathrm{D}[\pi Q[\;]]_\beta) &= \gamma, && \text{if } \beta = 1\gamma \\
&\ \uparrow, && \text{if } \beta = 0\gamma \\
&\ \langle\rangle, && \text{if } \beta = \langle\rangle \\
\ell(\mathrm{D}[Q[\;]]) &= \langle\rangle, && \text{in other cases with the hole in argument position} \\
\ell(\mathrm{D}[\lceil[\;], Q\rceil_\beta]) &= \beta 0 \\
\ell(\mathrm{D}[\lceil Q, [\;]\rceil_\beta]) &= \beta 1
\end{aligned}
$$

2.1.6. DEFINITION. Let $\Sigma = \langle P, \mathrm{C}[\;]\rangle$ be an occurrence of P in $\alpha.M$. The label of Σ is defined by $\ell(\Sigma) = \ell(\mathrm{C}[\;])$.

If P occurs in $\alpha.M$ with label $\ell(P) = \beta$, then $\beta.P$ can be viewed as a labelled subterm of $\alpha.M$ and by abuse of language we will speak of the occurrence $\beta.P$.

With so many brackets definition 2.1.5 may look rather unattractive. It can be rephrased in a direct definition of the label of an occurrence by induction on its depth as follows.

2.1.7. ALTERNATIVE DEFINITION. In $\alpha.M$ the occurrence M itself of course has label $\ell(M) = \alpha$. Other occurrences are always the direct suboccurrence of an intermediate occurrence N of smaller depth, of which the label can be assumed to be determined according to the induction. In the table the relevant cases as to the form of N and $\ell(N)$ are distinguished. The occurrences of which the label is defined are called P and, if two suboccurrences of N can be covered at once, Q. (It is assumed that $\beta \neq \uparrow$.)

N	$\ell(N)$	$\ell(P)$	$\ell(Q)$
$\mathrm{C}[P]$	$\uparrow$	$\uparrow$	
$\lambda x.P$	β	$\langle\rangle$	
$\pi_i P$	β	$i\beta$	
πPQ	0β	β	$\uparrow$
	1β	$\uparrow$	β
	$\langle\rangle$	$\langle\rangle$	$\langle\rangle$
PR	β	∞	
RP (not one of the cases above)	β	$\langle\rangle$	
$\lceil P, Q\rceil_\gamma$	β	$\gamma 0$	$\gamma 1$

EXAMPLE. In $\alpha.(\lambda x.\pi_0\ulcorner\pi_1 X,\pi YZ\urcorner_{11})P$ one computes successively: $\ell(\alpha.-)=\alpha$, $\ell(P)=<>$, $\ell(\lambda x.-)=\infty$, $\ell(\pi_0\ulcorner -,-\urcorner_{11})=<>$, $\ell(\pi_0)=\infty$, $\ell(\ulcorner -,-\urcorner_{11})=0$, $\ell(\pi_1 X)=110$, $\ell(X)=1110$, $\ell(\pi YZ)=111$, $\ell(\pi)=\ell(\pi Y)=\infty$, $\ell(Y)=\uparrow$ and $\ell(Z)=11$.

CONVENTION. All relations $\rightarrow_{\mathrm{a}}$ on labelled terms that will be met in the sequel respect labels. That is, they satisfy te implication

$$\alpha.M\rightarrow_{\mathrm{a}}\beta.N \ \Rightarrow\ \alpha=\beta$$

For convenience we will further write $\alpha.M\rightarrow_{\mathrm{a}}N$ instead of $\alpha.M\rightarrow_{\mathrm{a}}\alpha.N$. This is in accordance with our policy to omit labels whenever possible without danger of confusion. (In contrast the indexes of bookkeeping pairs are always explicitly mentioned. The notation $\ulcorner M_0,M_1\urcorner$ is used only as short for the neutral bookkeeping pair $\ulcorner M_0,M_1\urcorner_{<>}$, cf. 2.1.3.)

The notion of compatibility which was mentioned in 0.3, has to be adapted to the presence of labels. Since our new notion of compatibility will in the sequel be used mainly for relations defined on the restrictions $\Lambda\boldsymbol{\pi}\mathbf{p}$ and $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ of $\Lambda\boldsymbol{\pi}\mathbf{q}$, it has a parameter A for subsets of $\Lambda^{\circ}\boldsymbol{\pi}\mathbf{q}$.

2.1.8. DEFINITION. (i) Let A be a subset of $\Lambda^{\circ}\boldsymbol{\pi}\mathbf{q}$. The relation $\rightarrow_{\mathrm{a}}$ on A is called *compatible (with respect to A)*, if

$$\mathrm{C}[M]_\alpha\in\mathrm{A}\ \&\ \alpha.M\rightarrow_{\mathrm{a}}N \ \Rightarrow\ \mathrm{C}[M]_\alpha\rightarrow_{\mathrm{a}}\mathrm{C}[N]_\alpha$$

(ii) Moreover, $\rightarrow_{\mathrm{a}}$ is called *monotone*, if

$$\alpha.M\rightarrow_{\mathrm{a}}N\ \&\ \alpha\leq\beta \ \Rightarrow\ \beta.M\rightarrow_{\mathrm{a}}N$$

The following condition for the soundness of inductive definitions is straightforward.

2.1.9. CLOSURE CONDITION. Let $\rightarrow_{\mathrm{a}}$ be a relation on A. A necessary and sufficient condition for the existence of a compatible relation on A extending $\rightarrow_{\mathrm{a}}$ is given by the implication:

$$\mathrm{C}[M]_\alpha\in\mathrm{A}\ \&\ \alpha.M\rightarrow_{\mathrm{a}}N \ \Rightarrow\ \mathrm{C}[N]_\alpha\in\mathrm{A}.$$

Strictly speaking, whenever a relation is defined as the compatible closure of $\rightarrow_{\mathrm{a}}$ (or "the compatible relation generated by $\rightarrow_{\mathrm{a}}$") this condition should be verified. In most cases it will be obvious though.

2.2. Labels under replacement and substitution. The stability of the several notions of reduction which are studied in the sequel, will turn out to depend on the fact that labels of occurrences do not diminish in passing from ancestor to descendant. In the following lemmas the ground is prepared by showing that labels are non-diminishing under a number of operations involving replacement and substitution.

It is here convenient to treat the value undefined ($\uparrow$) as if it were itself a label. The partial order $\leq$ may then be regarded as incorporating $\uparrow$ as maximal element: $\alpha \leq \uparrow$ for every α.

2.2.1. LEMMA. (i) $\alpha \leq \beta \Rightarrow \ell(\alpha.C[\ ]) \leq \ell(\beta.C[\ ])$.
(ii) $\ell(\alpha.C[\ ]) \leq \ell(D[C[\ ]]_\alpha)$.
(iii) With the one exception that both $C[\ ] \equiv \pi C'[\ ]$ and $D_1[\ ] \equiv D'[[\ ]\mathcal{X}]$, we have

$$\alpha \leq \beta \Rightarrow \ell(D_1[C[\ ]]_\alpha) \leq \ell(D_2[C[\ ]]_\beta).$$

PROOF. Induction on $C[\ ]$. For (i) and (ii) it is all very straightforward, one just follows definition 2.1.5. Here we only touch on a few cases of (iii).
- $C[\ ] \equiv \lambda x.C'[\ ]$. Then, as $\ell(D_1[\lambda x.[\ ]]) = \ell(D_2[\lambda x.[\ ]]) = <>$, the induction hypothesis can be used for $C'[\ ]$ with $D_i[\lambda x.[\ ]]$, $(i=1, 2)$. The exception does not interfere: regardless of the form of $D_1[\ ]$, there cannot exist a $D'[\ ]$ such that $D_1[\lambda x.[\ ]] \equiv D'[[\ ]\mathcal{X}]$.
- $C[\ ] \equiv C''[\ ]\mathcal{X}$, $C''[\ ] \not\equiv \pi C'[\ ]$. Then the induction hypothesis can be used for $C''[\ ]$ with $D_i[[\ ]_\infty \mathcal{X}]$, $(i = 1,2)$. The exception is taken care of by the assumption on $C''[\ ]$.
- $C[\ ] \equiv \pi C'[\ ]\mathcal{X}$. With definition 2.1.5 one easily checks that $\ell(D_1[\pi[\ ]\mathcal{X}]_\alpha) \leq \ell(D_2[\pi[\ ]\mathcal{X}]_\beta)$. Then the induction hypothesis can be used for $C'[\ ]$. □

EXAMPLE. An illustration of the exception is given by $D_1[\ ] \equiv 00.[\ ]\mathcal{X}$, $D_2[\ ] = \infty.[\ ]$ and $C[\ ] \equiv \pi[\ ]$. Then $\ell(D_1[\ ]) = \ell(D_2[\ ]) = \infty$, but $\ell(D_1[C[\ ]]) = \ell(00.\pi[\ ]\mathcal{X}) = 0$ and $\ell(D_2[C[\ ]]) = \ell(\infty.\pi[\ ]) = <>$. And $0 \leq <>$ does not hold.

An immediate consequence of the fact that $\ell(P)$ depends only on the context of occurrence of P and not on any of the characteristics of P itself, is that changing P in some way or another does not affect its label. More precisely:

2.2.2. LEMMA. Let P occur in $\alpha.M$ with $\ell(P)=\beta$ and let M' be the result of replacing P by Q in M. Then in $\alpha.M'$ we have still $\ell(Q)=\beta$.

The reader may have noticed that in dealing with occurrences and their labels we speak two languages. The official one, in terms of contexts, was employed in definitions 2.1.5 and 2.1.6 and in lemma

2.2.1 above; the loose one, in terms of subterms, in the alternative definition 2.1.7 and in lemma 2.2.2. The former has the advantage of precision; the latter, on the other hand, has greater flexibility and is more in line with common usage. In opting for both advantages, we accept the lack of uniformity that ensues.

In this line, parts (i) and (ii) of the following lemma are essentially the same. But (i) is easier to prove, whereas (ii) is the form we prefer for applications.

2.2.3. LEMMA. (i) Let Q be an occurrence in $C[\ ]_\alpha$, disjoint with the hole. Let $C'[\ ]_\beta$ be the result of replacing Q by some Q'. Assume further that Q is not one of the terms π, πX, π_0 or π_1 in applicative position. Then $\alpha \leq \beta$.
(ii) Let P and Q be disjoint occurrences in M, M' the result of replacing Q by some Q'. Assume further that Q is not one of the terms π, πX, π_0 or π_1 in applicative position. Then $\ell(P)$ in M $\leq \ell(P)$ in M'.
(iii) The same as (ii), only now several disjoint Q's replaced, all satisfying the condition.
(iv) As (iii), but now also P changed as in 2.2.2.

PROOF. (i) Induction on $C[\ ]$. Distinguish cases according to the shape of $C[\ ]$, following 2.1.5. The only interesting cases are $C[\ ] \equiv D[\pi_i[\ ]]$, $D[\pi X[\ ]]$ or $D[\pi[\ ]X]$. Then if $Q \subseteq X$ the label of the hole is not affected by the replacement. Otherwise the induction hypothesis can be used with $D[\ ]$.
(ii) M is of the form $C[P]_\alpha$, with $\ell(P) = \alpha$ and M' of the form $C'[P]_\beta$ as in (i). Apply (i).
(iii) Just repeat (ii).
(iv) Combine (iii) and 2.2.2. □

Notice that an increase of $\ell(P)$ results e.g. if x in $C[xP]$ is replaced by π_0.

2.2.4. LEMMA. (i) Let P be an occurrence in M, and P' the corresponding occurrence in $(x:=N)M$. Then $\ell(P) \leq \ell(P')$.
(ii) Let P occur in $<>.N$, and P' be a corresponding occurrence in $(x:=N)M$, then $\ell(P) \leq \ell(P')$.

PROOF. (i) follows from 2.2.3(iv) with $Q \equiv x$, not falling under the exceptions; (ii) follows from 2.2.1(i) and (ii). □

We conclude with two more immediate consequences of 2.1.5/7. 2.2.5(ii) is important for the next section.

2.2.5. LEMMA. (i) Replacement of an occurrence that is not in applicative position ($\ell \neq \infty$) does not affect the label of any disjoint occurrence P.

(ii) Replacement of a transient occurrence Q does not affect the label of any occurrence P that either ccntains or is disjoint with Q.

PROOF. (i) can easily be verified by inspection of the table in 2.1.7, using induction on the depth of P.
(ii) If P contains Q, then use 2.2.2. For the case that P and Q are disjoint, let R be the maximal transient occurrence containing Q. As only the clause for $\ell(\pi XY)$ can introduce the value $\uparrow$ (for X or Y), R is not in applicative position. Then, if R contains P, P is transient by that very fact, and if not, then (i) can be applied with R instead of Q as the occurrence being replaced. □

2.3. ~-equivalence, the system $\boldsymbol{\lambda\pi}$q. In this section we move on from the stage of pre-terms, by abstracting from the transient occurrences in Λ°**πq**. This is done by first defining the equivalence relation ~, the refined version of ≈ which was announced in 1.8, and then defining Λ**πq** as Λ°**πq**/~. (After this, still one more step is required to get at Λ**πp**. The **q** in Λ**πq** may be taken as mnemonic for "quasi".)

2.3.1. DEFINITION. ~ is the monotone and compatible equivalence relation on Λ°**πq** which is generated by the clauses:

$$Y = Y' \Rightarrow 0.\pi XY \sim \pi XY',$$
$$Y = Y' \Rightarrow 1.\pi YX \sim \pi Y'X.$$

An equivalent characterization is: $M \sim N \Leftrightarrow N$ can be obtained from M by replacing a number of transient occurrences by terms that are convertible to the original ones. (Cf. 2.1.8 for the definitions of monotone and compatible.)

EXAMPLES. - $\alpha.\pi_0(\pi X((\lambda y.y)Z)) \sim \pi_0(\pi XZ)$;
- $<>.\pi X((\lambda y.y)\ Y)Z \sim \pi XYZ$, since $\infty = 0\infty$;
- not $\alpha.\pi_0(\pi((\lambda y.y)\ Y)Z) \sim \pi_0(\pi YZ)$;
- $0.\pi X\lceil\pi(\pi_0 X)((\lambda y.y)(\pi_1 X)), (\lambda y.y)X\rceil \sim \pi X\lceil X, (\lambda y.y)X\rceil$.

2.3.2. LEMMA. Let $M, N \in \Lambda^\circ$**πq**, $M \sim N$. Then:
(i) M and N have the same form, in the sense that
- $M \equiv x \Rightarrow N \equiv x$,
- $M \equiv \lambda x.M_0 \Rightarrow N \equiv \lambda x.N_0\ \&\ <>.M_0 \sim N_0$, etc.
(but note especially the case:
- $M \equiv 0\alpha.\pi M_0 M_1 \Rightarrow N \equiv 0\alpha.\pi N_0 N_1,\ \alpha.M_0 \sim N_0\ \&\ M_1 = N_1$).
(ii) To each non-transient occurrence P in M there corresponds naturally a unique ~-equivalent one, Q, in N of the same form and with the same label.

(iii) Let P and Q be corresponding non-transient occurrences in M and N, not falling under the exceptions of 2.2.3 and with label β. Then, if $\beta.R \sim S$, the results of replacing P by R in M and Q by S in N respectively, are $\sim$-equivalent terms again.
(iv) $\alpha.M_0 \sim M_1$ & $\langle\rangle.N_0 \sim N_1 \Rightarrow \alpha.(x{:=}N_0)M_0 \sim (x{:=}N_1)M_1$.

PROOF. (i) can easily be verified by induction on M. For (ii) the corresponding occurrence Q can then be defined by induction on the depth of P, the label taken care of by 2.2.5(ii). As to (iii) it should be observed that 2.2.3(ii) guarantees that the transient occurrences which are disjoint with the ones being replaced remain transient after the replacement. (iv) follows by repeated application of (iii). □

These invariances make it possible to operate with $\sim$-equivalence in the same way as it is normally done with α-equivalence, that is, use elements of $\Lambda^\circ\boldsymbol{\pi}\mathbf{q}$ as representitives for their respective equivalence classes. The latter will constitute the elements of $\Lambda\boldsymbol{\pi}\mathbf{q}$.

2.3.3. DEFINITION. $\Lambda\boldsymbol{\pi}\mathbf{q}$, the set of quasi terms, is defined by:

$$\Lambda\boldsymbol{\pi}\mathbf{q} = \Lambda^\circ\boldsymbol{\pi}\mathbf{q}/\sim.$$

It remains then to be checked, that all relevant predicates and operations on $\Lambda^\circ\boldsymbol{\pi}\mathbf{q}$ are respected by $\sim$. As this is obviously not the case with length (of the term), and with depth (of an occurrence), these notions are adjusted to the new situation by simply not longer counting the symbols within transient occurrences.

But besides that everything is all right so far. Convertibility is respected because of the premisses $Y=Z$ in definition 2.3.1, as = is a compatible relation. Non-transient occurrences and their labels were covered already in 2.3.2. They will be called just occurrences, as of course transient occurrences do not exist any more in $\Lambda\boldsymbol{\pi}\mathbf{q}$: they were the ones abstracted from. Accordingly the contexts can in $\Lambda\boldsymbol{\pi}\mathbf{q}$ only be used to indicate positions with a defined label $(\ell(C[\ ]) \neq \uparrow)$. Substitution was already taken care of by 2.3.2(iv).

2.3.4. DEFINITION. $\boldsymbol{\lambda\pi}\mathbf{q}$ is the system based on $\Lambda\boldsymbol{\pi}\mathbf{q}$ and with its convertibility relation = derived from that of $\boldsymbol{\lambda}^\circ\boldsymbol{\pi}\mathbf{q}$ as indicated above.

Observe once more that in passing from system to system, conversion is each time merely adapted to the latest version of terms, staying essentially the same. Conversion in $\boldsymbol{\lambda}^\circ\boldsymbol{\pi}\mathbf{q}$, from which $\boldsymbol{\lambda\pi}\mathbf{q}$ is derived, was just that of $\boldsymbol{\lambda\pi}$ extended to terms with labels and bookkeeping pairs. This line is pursued in the sequel. The systems $\boldsymbol{\lambda\pi}\mathbf{p}$ and $\boldsymbol{\lambda\pi}\mathbf{p}^*$ will be obtained by putting restrictions on the set of terms $\Lambda\boldsymbol{\pi}\mathbf{q}$ and by defining new reduction relations, but as to convertibility we will just keep with $\boldsymbol{\lambda\pi}\mathbf{q}$.

2.4. $\Lambda\pi\mathbf{p}$ and $\Lambda\pi\mathbf{p}^*$. Our final step towards the generation of a manageable system of terms which includes bookkeeping pairs, consists in effectuating the restrictions on the positions which a bookkeeping pair with a given index is allowed to occupy. As explained in 1.7, the pair $\lceil M_0, M_1\rceil_\alpha$ belongs in an α-context.

2.4.1. DEFINITION. The set of *canonical* terms, denoted by $\Lambda\pi\mathbf{p}^*$, is defined as the subset of $\Lambda\pi\mathbf{q}$ consisting of all terms which meet the requirement that α-bookkeeping pairs occur only in α-contexts:

$$\Sigma = <\lceil M_0, M_1\rceil_\alpha, C[\ ]> \ \Rightarrow\ \ell(\Sigma) = \alpha.$$

EXAMPLES. The terms $\alpha.\lambda x.\pi_0\lceil\lceil\pi_0 y, \pi_0 y\rceil_{00}, \pi_0\lceil y, y\rceil_{001}\rceil_0$ and $\infty.x\lceil y, y\rceil$ are canonical. On the other hand e.g. $\alpha.\pi_1\lceil y, y\rceil_{0\beta}$ and $<>.\lceil\lceil y, y\rceil, y\rceil$ are not. Note in particular that, although $<>.(\lambda y.\,yx)\lceil y, y\rceil \in \Lambda\pi\mathbf{p}^*$, we have $<>.\lceil y, y\rceil x \notin \Lambda\pi\mathbf{p}^*$, the latter term being a β-reduct of the former one.

From the example we see that the set $\Lambda\pi\mathbf{p}^*$ will not be closed under the reduction rules which are intended for the system $\boldsymbol{\lambda\pi}\mathbf{p}^*$. (Of which β-reduction is one.) Therefore we need yet another intermediate system $\boldsymbol{\lambda\pi}\mathbf{p}$ over a broader set of terms $\Lambda\pi\mathbf{p}$, being closed under reduction all right.

2.4.2. DEFINITION. The set $\Lambda\pi\mathbf{p}$ is the subset of $\Lambda\pi\mathbf{q}$ wich results if one weakens the condition on occurrences of bookkeeping pairs for $\Lambda\pi\mathbf{p}^*$ to:

$$\Sigma = <\lceil M_0, M_1\rceil_\alpha, C[\ ]> \ \Rightarrow\ \alpha \leq \ell(\Sigma).$$

EXAMPLES. We have $\Lambda\pi\mathbf{p}^* \subseteq \Lambda\pi\mathbf{p} \subseteq \Lambda\pi\mathbf{q}$. As to the non-canonical terms mentioned above, we still have $\alpha.\pi_1\lceil y, y\rceil_{0\beta} \notin \Lambda\pi\mathbf{p}$, but $<>.\lceil\lceil y, y\rceil, y\rceil$ and $<>.\lceil y, y\rceil x \in \Lambda\pi\mathbf{p}$.

The counterparts of $\Lambda\pi\mathbf{p}^*$ and $\Lambda\pi\mathbf{p}$ in $\Lambda^\circ\pi\mathbf{q}$ are called $\Lambda^\circ\pi\mathbf{p}^*$ and $\Lambda^\circ\pi\mathbf{p}$ respectively. It is obvious by 2.3.2(ii) that $\Lambda^\circ\pi\mathbf{p}^*$ and $\Lambda^\circ\pi\mathbf{p}$ are closed under $\sim$ and hence that definitions 2.4.1 and 2.4.2 are in order in this respect (cf. table 2.0.1).

Although, as said, performing a contraction in a $\Lambda\pi\mathbf{p}^*$-term may lead outside of $\Lambda\pi\mathbf{p}^*$, the result will always remain within $\Lambda\pi\mathbf{p}$. (That is, the worst thing that can happen is that an α-bookkeeping pair shows up in a β-context with $\beta > \alpha$) From there it can be projected back to $\Lambda\pi\mathbf{p}^*$ by $(\ )^*$. This operation will be defined via the auxiliary reduction relation $\hookrightarrow$ on $\Lambda\pi\mathbf{p}$. $\hookrightarrow$ is called cancelling, as it cancels superfluous bookkeeping pairs by selecting that component which is still consistent with the information presented by the actual context.

2.4.3. DEFINITION. The one step reduction relation $\hookrightarrow$ on $\Lambda\pi\mathbf{p}$, called *cancelling*, is defined as the monotone and compatible closure of the contraction rule:

$$\mathbf{can}\colon\ \alpha\mathrm{i}.\ulcorner M_0, M_1\urcorner_\alpha \hookrightarrow M_\mathrm{i},\ \text{for } \mathrm{i}=0 \text{ or } 1.$$

EXAMPLES. $<>.\ulcorner\ulcorner y,y\urcorner,y\urcorner \hookrightarrow \ulcorner y,y\urcorner$; $<>.\ulcorner y,y\urcorner x \hookrightarrow yx$; $\alpha.\lambda x.\pi_1(\pi_0\ulcorner\ulcorner\pi_0 y,\pi_0 y\urcorner_{00},\pi_0\ulcorner y,y\urcorner_{001}\urcorner_0) \hookrightarrow \lambda x.\pi_1(\pi_0(\pi_0\ulcorner y,y\urcorner_{001})$. The monotonicity and compatibility amount to the same as the general formulation $\mathrm{C}[\ulcorner M_0, M_1\urcorner_\alpha]_{\alpha\mathrm{i}\beta} \hookrightarrow \mathrm{C}[M_\mathrm{i}]$.

Conceived of as a relation on $\Lambda^\circ\pi\mathbf{p}$, it follows by lemma 2.3.2 (iii) that $\hookrightarrow$ is respected by $\sim$, as obviously

$$\alpha\mathrm{i}.\ulcorner M_0, M_1\urcorner_\alpha \sim \ulcorner N_0, N_1\urcorner_\alpha \ \Rightarrow\ \alpha\mathrm{i}.\, M_\mathrm{i} \sim N_\mathrm{i}.$$

Moreover, it is important to note that $\hookrightarrow$ preserves convertibility: $\hookrightarrow\ \subseteq\ =$. Finally, that $\Lambda\pi\mathbf{p}$ is closed under $\hookrightarrow$ and under substitution is established, along with some other closure properties, in the following lemmas. Observe, however, that $\Lambda\pi\mathbf{p}^*$ is not closed under substitution, as can be illustrated by the example: $M \equiv \pi_0 x$ and $N \equiv <>.\ulcorner N_0, N_1\urcorner \in \Lambda\pi\mathbf{p}^*$, but $(x:=N)M \equiv \pi_0\ulcorner N_0, N_1\urcorner \notin \Lambda\pi\mathbf{p}^*$.

2.4.4. LEMMA. (i) $\alpha.M\in\Lambda\pi\mathbf{p}$ & $\alpha\leq\beta \Rightarrow \beta.M\in\Lambda\pi\mathbf{p}$,
(ii) If $\mathrm{C}[P]_\beta\in\Lambda\pi\mathbf{p}$ & $\alpha.Q\in\Lambda\pi\mathbf{p}$ & $\alpha\leq\beta$ & $P\not\equiv\pi$, πX, π_0 or π_1 in applicative position, $\mathrm{C}[Q]_\beta\in\Lambda\pi\mathbf{p}$,
(iii) $\alpha.M\in\Lambda\pi\mathbf{p}$ & $<>.N\in\Lambda\pi\mathbf{p} \Rightarrow \alpha.(x:=N)M\in\Lambda\pi\mathbf{p}$,
(iv) If P is not of the form πX in applicative position, then $\mathrm{C}[P]_\alpha\in\Lambda\pi\mathbf{p} \Rightarrow \alpha.P\in\Lambda\pi\mathbf{p}$.

PROOF. (i) is an immediate corollary to 2.2.1(i). For (ii) assume $\ulcorner X_0, X_1\urcorner_\delta$ occurs in $\mathrm{C}[Q]_\beta$ with label γ. We have to check that $\delta\leq\gamma$. Distinguish cases as to the position of $\ulcorner X_0, X_1\urcorner_\delta$. If $\ulcorner X_0, X_1\urcorner_\delta$ occurs within Q, then $\delta\leq\gamma$ follows from the assumptions that $\alpha.Q\in\Lambda\pi\mathbf{p}$ and $\alpha\leq\beta$ by 2.2.1(ii). If $\ulcorner X_0, X_1\urcorner_\delta$ and Q are disjoint, then 2.2.3(ii) can be used. If $Q\subseteq\ulcorner X_0, X_1\urcorner_\delta$, then the replacement did not affect the context of the bookkeeping pair at all. (iii) follows from (ii), and (iv) from 2.2.1(iii). (The indicated exception is the only case that occurrences in P do not automatically obtain the same labels in $\mathrm{C}[P]_\alpha$ and in $\alpha.P$). □

2.4.5. LEMMA. $\Lambda\pi\mathbf{p}$ is closed under $\hookrightarrow$.

PROOF. Assume $\mathrm{C}[\ulcorner X_0, X_1\urcorner_\alpha]_{\alpha\mathrm{i}\beta}\in\Lambda\pi\mathbf{p}$. Then by 2.4.4(iv) also $\alpha\mathrm{i}.X_\mathrm{i}\in\Lambda\pi\mathbf{p}$; since $\alpha\mathrm{i}\leq\alpha\mathrm{i}\beta$, it follows then by 2.4.4(ii) that $\mathrm{C}[X_\mathrm{i}]_{\alpha\mathrm{i}\beta}\in\Lambda\pi\mathbf{p}$. □

As each $\hookrightarrow$-step reduces the length of the term, it is obvious that $\hookrightarrow$ is strongly normalizing. Also the Church-Rosser property for $\hookrightarrow$ is easily verified and hence each $\Lambda\boldsymbol{\pi}\mathbf{p}$-term $\alpha.M$ has a unique $\hookrightarrow$-normal form.

2.4.6. DEFINITION. (i) The *canonical form* of a term $\alpha.M\in\Lambda\boldsymbol{\pi}\mathbf{p}$, notation $(\alpha.M)^*$, is the (unique) $\hookrightarrow$-normal form of $\alpha.M$.
(ii) The relation on $\Lambda\boldsymbol{\pi}\mathbf{p}$ which transforms M into M^* in one step is denoted by $\hookrightarrow^*$: $M\hookrightarrow^* M^*$.

NOTATION. We slightly ambiguously use the notations $\alpha.M^*$ and M^* for $(\alpha.M)^*$. This is convenient, but it must be observed that this way * becomes context sensitive. E.g. we get $\pi_0(\ulcorner M_0, M_1\urcorner)^*\equiv\pi_0 M_0{}^*$, whereas $\pi_1(\ulcorner M_0, M_1\urcorner)^*\equiv\pi_1 M_1{}^*$. And even nastier, the X^*-s in $\ulcorner X^*, X^*\urcorner_\alpha$ are in general not the same. Sometimes we prefer to be more precise, and then use the notation $\alpha{*}M$ to denote M^* in an α-context.

EXAMPLES. $\alpha.(\lambda x.\ulcorner\ulcorner y,(\lambda x.x)y\urcorner, y\urcorner)^*\equiv\lambda x.\ulcorner y, y\urcorner$;
$<>.(\ulcorner y,\ulcorner y,(\lambda x.x)y\urcorner\urcorner)^*\equiv\ulcorner y,(\lambda x.x)y\urcorner$; $<>.(\ulcorner y, y\urcorner x)^*\equiv yx$.

An α-bookkeeping pair in a β-context with $\beta>\alpha$, can always be cancelled by $\hookrightarrow$, so the $\hookrightarrow$-normal forms must be all in $\Lambda\boldsymbol{\pi}\mathbf{p}^*$. Conversely $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ is defined such as not to contain any superfluous bookkeeping pairs, so that in $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ there is nothing to cancel. Therefore $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ can be characterized as the set of canonical forms and $\hookrightarrow^*$ ($\subseteq\Lambda\boldsymbol{\pi}\mathbf{p}\times\Lambda\boldsymbol{\pi}\mathbf{p}^*$) behaves as the identity relation on $\Lambda\boldsymbol{\pi}\mathbf{p}^*$. Put more concisely: * is a projection operation from $\Lambda\boldsymbol{\pi}\mathbf{p}$ onto $\Lambda\boldsymbol{\pi}\mathbf{p}^*$.

This section is concluded with some lemmas and definitions which will prove to be of use in the sequel. It may be pointed out first, that although it is shown in lemma 2.4.8 that $\hookrightarrow$ is substitutive, $\hookrightarrow^*$ is not. This is illustrated by the following trivial counterexample:

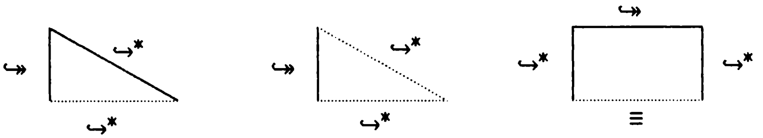
$x\ulcorner y, y\urcorner\hookrightarrow^* x\ulcorner y, y\urcorner$, as $x\ulcorner y, y\urcorner$ is itself canonical.

But, substituting π_0 for x, we do not have

$\pi_0\ulcorner y, y\urcorner\hookrightarrow^*\pi_0\ulcorner y, y\urcorner$, as $\pi_0\ulcorner y, y\urcorner$ is not canonical.

In 2.4.7 some properties of $\hookrightarrow$ and $\hookrightarrow^*$ are summed up which should be obvious by the above.

2.4.7. LEMMA. (i) $\hookrightarrow^*\subseteq\hookrightarrow\!\!\!\!\rightarrow$
(ii) $\hookrightarrow\!\!\!\!\rightarrow+\hookrightarrow^* = \hookrightarrow^*$
(iii) The following diagrams hold.



(iv) $(\mathrm{C}[Q])^* \equiv (\mathrm{C}[Q^*])^*$,

PROOF. Straightforward. □

2.4.8. LEMMA.

(i) $\alpha.\, M_0 \hookrightarrow M_1$ & $\langle\rangle.\, N \in \Lambda\pi\mathbf{p} \Rightarrow \alpha.\, (x:=N)\, M_0 \hookrightarrow (x:=N)\, M_1$.

(ii) $\langle\rangle.\, N_0 \hookrightarrow N_1$ & $\alpha.\, M \in \Lambda\pi\mathbf{p} \Rightarrow \alpha.\, (x:=N_0)\, M \hookrightarrow\!\!\!\twoheadrightarrow (x:=N_1)\, M$.

(iii) $\alpha.\, M_0 \hookrightarrow\!\!\!\twoheadrightarrow M_1$ & $\langle\rangle.\, N_0 \hookrightarrow\!\!\!\twoheadrightarrow N_1 \Rightarrow \alpha.\, (x:=N_0)\, M_0 \hookrightarrow\!\!\!\twoheadrightarrow (x:=N_1)\, M_1$.

(iv) $((x:=N)\, M)^* \equiv ((x:=N)\, M^*)^* \equiv ((x:=N^*)\, M)^* \equiv ((x:=N^*)\, M^*)^*$.

PROOF. For (i) use 2.2.4(i) on the bookkeeping pair in M_0 which is cancelled. For (ii) proceed in the same way with 2.2.4(ii). (iii) follows from (i) and (ii), (iv) from (iii). □

2.4.9. DEFINITION. For a relation $\rightarrow_a$ on $\Lambda\pi\mathbf{p}$ we introduce the notations $\rightarrow_a{}^*$ and $\rightarrow_a{}^{1/2}$, by defining:

(i) $\rightarrow_a{}^*$ denotes the restriction to $\Lambda\pi\mathbf{p}^*$ of $\rightarrow_a + \hookrightarrow^*$;

(ii) $\rightarrow_a{}^{1/2}$ denotes $\rightarrow_a + \hookrightarrow\!\!\!\twoheadrightarrow$.

NOTATION. The suggestive notation $\rightarrow_a{}^{1/2}$ indicates that though some cancellation may have been performed, the job is not necessarily finished, thus leaving us somewhere halfway the canonical form.

In accordance with 0.3 the double arrow ($\twoheadrightarrow$) with sub- and superscripts is always meant to signify the reflexive transitive closure of the whole. (The same can be said with respect to $\mathrel{=\!\!\!\twoheadrightarrow}$.) In particular $\twoheadrightarrow_a{}^*$ denotes always the reflexive transitive closure of $\rightarrow_a{}^*$. Note that as a consequence we do not have a special notation for the restriction to $\Lambda\pi\mathbf{p}^*$ of $\twoheadrightarrow_a + \hookrightarrow^*$.

2.4.10. LEMMA. (i) $\rightarrow_a{}^{**} = \rightarrow_a{}^*$;

(ii) $\rightarrow_a{}^{1/2*} = \rightarrow_a{}^*$.

PROOF. Straightforward. □

2.4.11. DEFINITION. Let $\rightarrow_a$ be a relation on $\Lambda\pi\mathbf{p}$. Then $\rightarrow_a$ is called

(i) *$*$-projectable* if $\alpha.\, M \rightarrow_a N \Rightarrow \alpha.\, M^* \mathrel{=\!\!\!\twoheadrightarrow}_a{}^* N^*$;

(ii) *$*$-monotone* if $\alpha.\, M \rightarrow_a N$ & $\alpha \leq \beta \Rightarrow \beta.\, M^* \mathrel{=\!\!\!\twoheadrightarrow}_a{}^* N^*$;

(iii) *$*$-compatible* if $\alpha.\, M \rightarrow_a N \Rightarrow \mathrm{C}[M]_\alpha{}^* \mathrel{=\!\!\!\twoheadrightarrow}_a{}^* \mathrm{C}[N]^*$;

(iv) *½-projectable* if the following diagram holds

$$\begin{tikzcd} {} \arrow[r, "\rightarrow_a"] \arrow[d, "\hookrightarrow"'] & {} \arrow[d, dotted, "\hookrightarrow\!\!\!\twoheadrightarrow"] \\ {} \arrow[r, dotted, "\overset{=}{\rightarrow}_a"'] & {} \end{tikzcd}$$

(v) *½-monotone* if
$\alpha.\, M \rightarrow_a N \;\&\; \alpha \leq \beta \Rightarrow (\exists N')(\beta.\, M \overset{=}{\rightarrow}_a N' \;\&\; \beta.\, N \hookrightarrow\!\!\!\twoheadrightarrow N')$;
(vi) *½-compatible* if
$\alpha.\, M \rightarrow_a N \;\&\; \ell(C[\;]) = \alpha \Rightarrow (\exists Q)(C[M] \overset{=}{\rightarrow}_a Q \;\&\; C[N] \hookrightarrow\!\!\!\twoheadrightarrow Q)$.

All relations $\rightarrow_a \subseteq \Lambda\pi\mathbf{p}^* \times \Lambda\pi\mathbf{p}^*$ are trivially $*$-projectable. As to the connection between the different kinds of monotonicity and compatibility, observe that ½-monotonicity and compatibility are weaker properties than plain monotonicity and compatibility. Furthermore it is obvious that for $*$-projectable relations monotonicity implies $*$-monotonicity and compatibility $*$-compatibility. The following lemma provides a means for proving $*$-projectability etc.

2.4.12. LEMMA. (i) Let $\rightarrow_a$ be ½-projectable. Then both $\rightarrow_a$ and $\twoheadrightarrow_a$ are $*$-projectable.
(ii) If $\rightarrow_a$ is ½-projectable and ½-monotone, then $\rightarrow_a$ and $\twoheadrightarrow_a$ are $*$-monotone as well.
(iii) $\rightarrow_a$ is ½-projectable and ½-compatible $\Rightarrow$ $\rightarrow_a$ and $\twoheadrightarrow_a$ are $*$-compatible.

PROOF. (i) First establish the diagrams a and b.

a
$$\begin{tikzcd} {} \arrow[r, "\rightarrow_a"] \arrow[d, "\hookrightarrow\!\!\!\twoheadrightarrow"'] & {} \arrow[d, dotted, "\hookrightarrow\!\!\!\twoheadrightarrow"] \\ {} \arrow[r, dotted, "\overset{=}{\rightarrow}_a"'] & {} \end{tikzcd}$$

b
$$\begin{tikzcd} {} \arrow[r, "\rightarrow_a"] \arrow[d, "\hookrightarrow^*"'] & {} \arrow[d, dotted, "\hookrightarrow\!\!\!\twoheadrightarrow"] \\ {} \arrow[r, dotted, "\overset{=}{\rightarrow}_a"'] & {} \end{tikzcd}$$

a just by repeated application of the assumption, b as a consequence of a because $\hookrightarrow^* \subseteq \hookrightarrow\!\!\!\twoheadrightarrow$. Then the $*$-projectability of $\rightarrow_a$ follows by combining b and 2.4.7(iii):

$$\begin{tikzcd} {} \arrow[r, "\rightarrow_a"] \arrow[d, "\hookrightarrow^*"'] & {} \arrow[d, dotted, "\hookrightarrow\!\!\!\twoheadrightarrow"] \arrow[dr, "\hookrightarrow^*"] & \\ {} \arrow[r, dotted, "\overset{=}{\rightarrow}_a"'] & {} \arrow[r, dotted, "\hookrightarrow^*"'] & {} \end{tikzcd}$$

The property of $\rightarrow_a$ being ½-projectable carries over to $\twoheadrightarrow_a$, since $\rightarrow_a \subseteq \twoheadrightarrow_a$, and a series of diagrams of the form a can be linked

together to the right. As a consequence also $\twoheadrightarrow_a$ is $*$-projectable.
(ii) Let $\alpha . M \to_a N$ and $\alpha \leq \beta$. Then by the assumption that $\to_a$ is ½-monotone and because $\to_a$ is already $*$-projectable by (i), in a β-context the following diagram can be constructed.

$$\begin{tikzcd}
 & N \arrow[d, dash, "\hookrightarrow\!\!\!\!\twoheadrightarrow"] \\
M \arrow[r, dash, "\Rrightarrow_a"] \arrow[d, dash, "\hookrightarrow^*"'] & N' \arrow[d, dash, dotted, "\hookrightarrow^*"] \\
M^* \arrow[r, dash, dotted, "\Rrightarrow_a^*"'] & N'^*
\end{tikzcd}$$

The $*$-monotonicity of $\to_a$ can be concluded, since $N \hookrightarrow\!\!\!\!\twoheadrightarrow N'$ implies that $N'^* \equiv N^*$. Regarding $\twoheadrightarrow_a$ it now clearly suffices to verify that it is itself ½-monotone too—the ½-projectability following just as in (i). We leave it as an exercise to derive the ½-monotonicity of $\twoheadrightarrow_a$ from the ½-projectability and ½-monotonicity of $\to_a$.
(iii) The reasoning is completely analogous to that of (ii). □

2.4.13. Lemma. (i) $\to_a$ is $*$-projectable $\Rightarrow$ $\to_a^{\frac{1}{2}}$ is $*$-projectable.
(ii) If $\to_a$ is $*$-monotone (or $*$-compatible), then so is $\to_a^*$.
(iii) If $\to_a^*$ is $*$-monotone (or $*$-compatible), then so is $\twoheadrightarrow_a^*$.
(iv) If $\to_a$ is $*$-projectable, then

$$M \in \Lambda\pi\mathbf{p}^* \;\&\; M \twoheadrightarrow_a + \hookrightarrow^* N \;\Rightarrow\; M \twoheadrightarrow_a^* N.$$

Proof. By now straightforward. For the $*$-compatibility part of (ii) lemma 2.4.7(iv) can be used. □

In the sequel the property of being both ½-monotone and ½-compatible will be referred to efficiently by the short "½+½". The following fundamental fact is a consequence of definition 2.4.11.

2.4.14. Lemma. If $\to_a$ is ½+½, then

$$\alpha . M \to_a N \;\&\; \alpha \leq \beta \Rightarrow (\exists Q)(C[M]_\beta \Rrightarrow_a Q \;\&\; C[N] \hookrightarrow\!\!\!\!\twoheadrightarrow Q).$$

Proof. First use definition 2.4.11(v) to find a N' such that $\beta . M \Rrightarrow_a N' \;\&\; \beta . N \hookrightarrow\!\!\!\!\twoheadrightarrow N'$. Subsequently 2.4.11(vi) yields a Q such that $C[M] \Rrightarrow_a Q \;\&\; C[N'] \hookrightarrow\!\!\!\!\twoheadrightarrow Q$. This Q suffices because $\hookrightarrow\!\!\!\!\twoheadrightarrow$ is compatible and transitive. □

2.5. λπp and λπp*. We now first present the contraction rules of the systems **λπp** and **λπp***. The one step reduction relation $\rightarrow$ of **λπp** then results at once. It is composed of two parts: $\rightarrow_\pi$ an $\rightarrow_p$. The reader is warned in advance that the component $\rightarrow_p$ will deviate from the customary, by being not at all monotone, and compatible only in a restricted sense.

2.5.1. DEFINITION. (i) The one step reduction relation $\rightarrow_\pi$ is the compatible and monotone closure, in $\Lambda\pi\mathbf{p}$, of the contraction rules:

$\boldsymbol{\beta}$: $<>.(\lambda x.M)N \rightarrow (x:=N)M$;
$\boldsymbol{\pi_0}$: $<>.\pi_0(\pi X_0 X_1) \rightarrow X_0$;
$\boldsymbol{\pi_1}$: $<>.\pi_1(\pi X_0 X_1) \rightarrow X_1$;
$\mathbf{l}$: $0.\pi(\pi_0 X)\,Y \rightarrow X$, provided that $\pi_1 X = Y$;
$\mathbf{r}$: $1.\pi Y(\pi_1 X) \rightarrow X$, provided that $\pi_0 X = Y$.

(ii) The one step reduction relation $\rightarrow_p$ is the least relation on $\Lambda\pi\mathbf{p}$ that satisfies the "contraction" rule:

$\mathbf{p}$: $\mathrm{C}[X]_\alpha \rightarrow \mathrm{C}[\lceil X, X\rceil_\alpha]$, provided that X is not already a bookkeeping pair itself and α is finite.

(iii) Then $\rightarrow$, the one step reduction relation of **λπp**, is defined as the union:

$$\rightarrow \;=\; \rightarrow_\pi \cup \rightarrow_p.$$

EXAMPLES. Provided that $\pi_1 X = Y$, an application of $\mathbf{l}$ yields
$<>.\lceil \pi(\pi_0 X)\,Y, X\rceil \rightarrow_\pi \lceil X, X\rceil$, but not
$<>.\lceil X, \pi(\pi_0 X)\,Y\rceil \rightarrow_\pi \lceil X, X\rceil$.

The relation $\rightarrow_p$ is not monotone: although
$01.\pi_1 X \rightarrow_p \pi_1\lceil X, X\rceil_{101}$ as well as
$<>.\pi_1 X \rightarrow_p \pi_1\lceil X, X\rceil_1$, we do not have
$01.\pi_1 X \rightarrow_p \pi_1\lceil X, X\rceil_1$.

The relation $\rightarrow_p$ is not even quite compatible, for although
$\infty.\pi X \rightarrow_p \pi\lceil X, X\rceil$, not $00.\pi XY \rightarrow_p \pi\lceil X, X\rceil Y$ (cf. the exception to 2.2.1(iii)).

COMMENT. In accordance with what was said on the subject of bookkeeping pairs in section 1.7, the purpose of the bookkeeping pair creating rule $\mathbf{p}$ is to make it possible to generate extra (hypothetical) information on the context of an occurrence X. Our motivation for not making $\rightarrow_p$ monotone should be clear from that purpose. For suppose e.g. that a step like

$<>.\pi_1(\pi_0(\pi_1 X)) \rightarrow_p \pi_1(\pi_0(\pi_1\lceil X, X\rceil_1))$

would be admitted. The labels of the two descendants X of X would be respectively 10 and 11. Compare these to the label 101 of X in the original. It is then clear that the proposed p-step would not result in growth of information: passing from the label 101 to 10 means a loss of information, while the label 11 represents information that is even inconsistent with the original 101.

The third cluster of examples above illustrates why we could not make the relation $\rightarrow_{\mathrm{p}}$ compatible either. It is almost compatible though, in a sense that will be made precise in the next section (cf. lemma 2.6.4).

The = in the conditions on **l** and **r** is just that of **λπq**. So the definition is not circular (compare 1.2.1). Furthermore notice that $\rightarrow \subseteq =$, implying the stability of the conditions under reduction. That $\rightarrow$ behaves well with respect to ~ is proved in the following lemma.

2.5.2. LEMMA. Considering $\rightarrow$ as a relation on $\Lambda^\circ\boldsymbol{\pi}\mathbf{p}$, we have the diagram:

$$\begin{tikzcd} \cdot \arrow[r, dash, "\sim"] \arrow[d, dash, "\rightarrow"'] & \cdot \arrow[d, dash, dotted, "\rightarrow"] \\ \cdot \arrow[r, dash, dotted, "\sim"'] & \cdot \end{tikzcd}$$

PROOF. With 2.3.2(ii) and (iii) in mind, it suffices to verify that if $\alpha.P$ is an $\rightarrow$-redex and $\alpha.P \sim Q$, and if R and S are the respective contracta of P and Q, then $\alpha.R \sim S$. Distinguish cases according to the contraction rule used. Then the result follows from 2.3.2(ii) for the rules $\boldsymbol{\pi_0}$,$\boldsymbol{\pi_1}$, **l** and **r** (= is respected by ~), with in addition 2.3.2(iv) for **β**. For **p** the monotonicity of ~ is needed, as the context of X is extended to $\alpha 0$ and to $\alpha 1$ in the respective components of the new bookkeeping pair. □

It is important to observe that $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ is not closed under $\rightarrow$. Three examples of reductions which start in $\Lambda\boldsymbol{\pi}\mathbf{p}^*$, but with a non-canonical result are (with X, Y and Z appropriately chosen):

$$<>.\pi_1(\pi x \pi_0)\ulcorner \pi ZZ, X \urcorner \rightarrow_{\pi} \pi_0 \ulcorner \pi ZZ, X \urcorner,$$
$$<>.(\lambda x.x)\ulcorner \lambda y.y,\ Y \urcorner Z \rightarrow_{\pi} \ulcorner \lambda y.y,\ Y \urcorner Z, \text{ and}$$
$$<>.\pi_0 \ulcorner \pi ZZ, X \urcorner_0 \rightarrow_{\mathrm{p}} \ulcorner \pi_0 \ulcorner \pi ZZ, X \urcorner_0 , \pi_0 \ulcorner \pi ZZ, X \urcorner_0 \urcorner.$$

In each of these examples the canonical form of the end term would contain (or be itself) a redex, which is here, however, still obstructed by the presence of a bookkeeping pair. These pairs appear to be superfluous though, as they somewhat misleadingly suggest an indeterminateness of context that does not really exist. In the system **λπp*** all such disturbing and useless bookkeeping pairs will be automatically cancelled.

2.5.3. DEFINITION. **λπp*** is defined as the reduction system over the set of terms Λ**πp*** which has $\rightarrow^*$ as its one step reduction relation. (Recall that according to 2.4.9, $\rightarrow^* = \rightarrow + \hookrightarrow^*$).

EXAMPLES. The three examples of $\rightarrow$-reductions above give rise to the $\rightarrow^*$-steps:

$$\begin{aligned}
&<>.\pi_1(\pi x \pi_0)\lceil \pi ZZ, X\rceil \rightarrow_\pi{}^* \pi_0(\pi ZZ),\\
&<>.(\lambda x.x)\lceil \lambda y.y, Y\rceil Z \rightarrow_\pi{}^* (\lambda y.y)Z, \text{ and}\\
&<>.\pi_0\lceil \pi ZZ, X\rceil_0 \rightarrow_p{}^* \lceil \pi_0(\pi ZZ), \pi_0 X\rceil.
\end{aligned}$$

(Of course the X, Y, and Z must be assumed to be chosen such that actually everything is in Λ**πp***.)

The usefulness of **λπp*** for our original problem concerning **λπ** rests on the fact that **λπ** can be seen as a subsystem of **λπp***. First, via the embedding ψ which was defined after 2.1.3, Λ**π** is included in Λ**πp***: terms without bookkeeping pairs are always canonical. Moreover, conversion in **λπ** can be carried out, via the detour of $\rightarrow^*$, in **λπp*** as well. This in spite of the weakening of the reduction relation by the context restrictions on the rules. The latter was in fact the content of proposition 1.9.2(ii). We prove it now. The equivalence relation generated by $\rightarrow^*$ is denoted by $=^*$.

2.5.4. THEOREM. $M, N \in$ Λ**π** & **λπ** $\vdash M = N \Rightarrow\ <>.M =^* N$.

PROOF. Induction on deductions in **λπ** (cf. 1.1). The only interesting deduction step is an application of the surjectivity axiom $\pi(\pi_0 X)(\pi_1 X) = X$. The other axioms are already included in **λπp*** as the rules $\boldsymbol{\pi_0}$, $\boldsymbol{\pi_1}$ and **β**, independent of context. So it suffices to prove $<>.\mathrm{C}[\pi(\pi_0 X)(\pi_1 X)]_\alpha =^* \mathrm{C}[X]$ for any C[ ] and X (in Λ**π**). Now if $\alpha \neq <>$, this can be concluded simply by an application of **l** (if $\alpha = 0\alpha'$) or **r** (if $\alpha = 1\alpha'$). In case of $\alpha = <>$, the $=^*$-equivalence can be established via the introduction of a bookkeeping pair:

$$\begin{aligned}
\mathrm{C}[\pi(\pi_0 X)(\pi_1 X)] &\rightarrow_p{}^* \mathrm{C}[\lceil \pi(\pi_0 X)(\pi_1 X), \pi(\pi_0 X)(\pi_1 X)\rceil]\\
&\rightarrow_\pi{}^* \mathrm{C}[\lceil X, \pi(\pi_0 X)(\pi_1 X)\rceil]\\
&\rightarrow_\pi{}^* \mathrm{C}[\lceil X, X\rceil]\\
&{}^*_p\!\leftarrow \mathrm{C}[X].
\end{aligned}$$

The rules used were respectively **p**, **l**, **r**, and again **p**. □

This is a good point to return for a moment to the sketch of proof which was given in the ouline 1.9. After some more preliminaries on **λπp** and **λπp*** in 2.6, in §3 the Church-Rosser theorem for **λπp*** will be proved (cf. 1.9.1(iii)). By theorem 2.5.4 this implies that λπ-convertible terms have a common $\rightarrow^*$-reduct in **λπp***. It will then be our final task to translate a $\rightarrow^*$-reduction sequence starting from $<>.M$ with $M \in$ Λ**π** into a $\geq$-sequence from M, in such a way that the end term of the $\geq$-sequence depends uniformly on the end term of the $\to^*$-sequence (cf. 1.9.2(i)).

This is accomplished in §4. Via a postponement result it is shown there, that if $\mathbf{\lambda\pi p^*} \vdash \langle\rangle.M \to^* N$ for terms $M \in \Lambda\pi$ and $N \in \Lambda\mathbf{\pi p^*}$, then $\mathbf{\lambda\pi^{\geq}} \vdash M \geq \phi(N)$. ($\phi$ was defined in 2.1.3). As a matter of fact, since in $\mathbf{\lambda\pi p^*}$ we work modulo ~, the end term $\phi(N)$ is obtained only up to ~. But by the obvious implication $X \sim Y \Rightarrow \phi(X) \approx \phi(Y)$, claim 1.9.1(i) then follows; and as it was pointed out in 1.9, that suffices for establishing CR/$\approx$ for $\mathbf{\lambda\pi}$ and thereby as a corollary the conservativity of $\mathbf{\lambda\pi}$ over $\mathbf{\lambda}$.

2.6. Monotonicity and compatibility in $\mathbf{\lambda\pi p^*}$. In this section we establish some elementary facts concerning the compatibility and monotonicity properties of the reduction relations of the systems $\mathbf{\lambda\pi p}$ and $\mathbf{\lambda\pi p^*}$. In effect it will be found that $\to^*$ is both $*$-monotone and $*$-compatible. What it all amounts to, is investigating the behaviour of $\to$ under cancelling. We start considering $\to_\pi$ seperately.

Recall that $\to_\pi$ and $\hookrightarrow$ are compatible and monotone relations on $\Lambda\mathbf{\pi p}$ by definition. But $\hookrightarrow^*$ lacks these properties, as witnessed by the examples:

- $\infty.\pi_0 \hookrightarrow^* \pi_0$, but not $\langle\rangle.\pi_0\lceil x, x\rceil \hookrightarrow^* \pi_0\lceil x, x\rceil$;
- $\langle\rangle.\lceil x, x\rceil \hookrightarrow^* \lceil x, x\rceil$, but not $0.\lceil x, x\rceil \hookrightarrow^* \lceil x, x\rceil$.

Moreover, the following examples show that also $\to_\pi^*$ is neither compatible nor monotone, even when considered with respect to $\Lambda\mathbf{\pi p^*}$.

- $\infty.(\lambda x.x)\pi_0 \to_\pi^* \pi_0$, but not $\langle\rangle.(\lambda x.x)\pi_0\lceil x, x\rceil \to_\pi^* \pi_0\lceil x, x\rceil$;
- $\langle\rangle.(\lambda x.x)\lceil x, x\rceil \to_\pi^* \lceil x, x\rceil$, but not $0.(\lambda x.x)\lceil x, x\rceil \to_\pi^* \lceil x, x\rceil$.

Nevertheless, we shall prove that $\to_\pi$ is $*$-projectable, from which it then follows by the monotonicity/compatibility of $\to_\pi$ that both $\to_\pi$ and $\to_\pi^*$ are $*$-monotone and $*$-compatible. The proof makes use of lemma 2.4.12. The following lemma yields the requisite diagram.

2.6.1. Lemma.

$$\begin{tikzcd} M \arrow[r, "(\Sigma)\to_\pi"] \arrow[d, "(\Delta)\hookrightarrow"'] & N \arrow[d, dotted, "\hookrightarrow\!\!\!\!\twoheadrightarrow"] \\ P \arrow[r, dotted, "\overset{=}{\to}_\pi"'] & Q \end{tikzcd}$$

PROOF. Distinguish cases according to the relative positions of Σ and Δ. If they are disjoint then there is no problem. Lemma 2.2.3(ii) guarantees that the residual of Σ in P and that of Δ in N are still redexes. Contracting them yields (the same) Q. Distinguish further:
- $\Sigma \subseteq \Delta$. Without loss of generality assume $M \equiv C[\lceil X_0, X_1 \rceil_\alpha]_{\alpha 0 \beta}$ and $P \equiv C[X_0]$. Then if $\Sigma \subseteq X_1$, just take $Q \equiv P$. If $\Sigma \subseteq X_0$ and Y is the result of contracting Σ in X_0, then let $Q \equiv C[Y]$. This is correct because the label of X_0 is not less in P than in M $(\alpha 0 \leq \alpha 0 \beta)$.
- $\Delta \subseteq \Sigma$. Further cases have to be distinguished, now according to the rule of 2.5.1 applied (i.e. the character of Σ).
$\underline{\beta}$ Use 2.4.8 and the monotonicity/compatibility of $\hookrightarrow\!\!\!\!\twoheadrightarrow$.
$\underline{\pi_0}$ $M \equiv C[\pi_0(\pi XY)]_\alpha$, $N \equiv C[X]$. If $\Delta \subseteq Y$ then just take $Q \equiv N$.
If $\Delta \subseteq X$ and Z is the result of cancelling Δ in X, then let $Q \equiv C[Z]$. This is correct because, by 2.2.2, the label of X in N remains α (and by the compatibility of $\hookrightarrow$ of course).
This should suffice. □

Observe by the way that the only non-routine argument in the proof was the checking of the labels of residuals. We can subsume this under a more general statement.

2.6.2. LEMMA. If $M \rightarrow N$ or $M \hookrightarrow N$ and Σ' is a descendant in N of an occurrence Σ in M, then $\ell(\Sigma') \geq \ell(\Sigma)$.

PROOF. Checking the different cases is a now routine matter. □

With this lemma and the monotonicity/compatibility of $\rightarrow_\pi$ and $\hookrightarrow$, 2.6.1 becomes just an application of well-known theory on CRS's: the construction of elementary diagrams for left linear rules without overlap (Klop [1980]). (NB: The rules of $\boldsymbol{\lambda\pi}\mathbf{p}$ do have overlap mutually, but not with $\hookrightarrow$.)

Reduction relations which satisfy lemma 2.6.2 will be called *non-diminishing*.

2.6.3. LEMMA. (i) $\rightarrow_\pi$ and $\twoheadrightarrow_\pi$ are $*$-projectable.
(ii) $\rightarrow_\pi{}^*$ and $\twoheadrightarrow_\pi{}^*$ are $*$-monotone and $*$-compatible.
(iii) $M \in \Lambda\boldsymbol{\pi}\mathbf{p}^*$ & $M \twoheadrightarrow_\pi N \Rightarrow M \twoheadrightarrow_\pi{}^* N^*$.

PROOF. (i) $\rightarrow_\pi$ is ½-projectable by lemma 2.6.1. So lemma 2.4.12(i) applies.
(ii) $\rightarrow_\pi$ and $\twoheadrightarrow_\pi$ are monotone, compatible and $*$-projectable. In order to derive that they are $*$-monotone and $*$-compatible we do not even need 2.4.12(ii) or (iii). For $\rightarrow_\pi{}^*$ and $\twoheadrightarrow_\pi{}^*$ apply 2.4.13(ii).
(iii) Since in $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ always $M \equiv M^*$, this is an immediate consequence of (i) and lemma 2.4.13(iv). □

We turn to p-reduction. Recall that a $\rightarrow_{\mathrm{p}}{}^*$-reduction step is composed of a $\rightarrow_{\mathrm{p}}$-step starting with a term in $\Lambda\boldsymbol{\pi}\mathbf{p}^*$, followed by a $\hookrightarrow^*$-step which takes the result back into $\Lambda\boldsymbol{\pi}\mathbf{p}^*$. By the restrictions on $\mathbf{p}$ (cf. 2.5.1), this rule can only introduce bookkeeping pairs in contexts with finite label, that is, not in applicative position. It then follows by lemma 2.2.5(i) that in order to get the canonical result no cancelling is required outside the new bookkeeping pair. To bring out this feature more clearly we recast the relation $\rightarrow_{\mathrm{p}}{}^*$ in a form analogous to that of $\mathbf{p}$, recapitulated here from definition 2.5.1(ii).

$$\mathbf{p}:\quad \mathrm{C}[X]_\alpha \rightarrow_{\mathrm{p}} \mathrm{C}[\ulcorner X, X\urcorner_\alpha],$$
$$\mathbf{p}^*:\quad \mathrm{C}[X]_\alpha \rightarrow_{\mathrm{p}}{}^* \mathrm{C}[\ulcorner \alpha 0 * X, \alpha 1 * X\urcorner_\alpha],$$

(α a finite label and X not already a bookkeeping pair itself). Note the contrast with $\rightarrow_{\pi}{}^*$. Contracting a π-redex can very well increase the label of a disjoint occurrence, as a result of which some extra cancelling is then required.

With the help of the above characterization of p- and p^*-reduction it is possible to indicate to which extent these relations are compatible. We do it for $\rightarrow_{\mathrm{p}}{}^*$.

2.6.4. LEMMA. Suppose we have M and $\mathrm{D}[\ ]$ with $\mathrm{D}[M] \in \Lambda\boldsymbol{\pi}\mathbf{p}^*$ and that there do not exist terms M' and X and a context $\mathrm{D}'[\ ]$ such that both $M \equiv \pi M'$ and $\mathrm{D}[\ ] \equiv \mathrm{D}'[[\ ]X]$. Then:

(i) Σ: $M \rightarrow_{\mathrm{p}}{}^* N \Rightarrow \mathrm{D}[M] \rightarrow_{\mathrm{p}}{}^* \mathrm{D}[N]$;

(ii) $M \twoheadrightarrow_{\mathrm{p}}{}^* N \Rightarrow \mathrm{D}[M] \twoheadrightarrow_{\mathrm{p}}{}^* \mathrm{D}[N]$.

PROOF. (i) By the characterization of $\mathbf{p}^*$ given above it suffices to check that the label of the occurrence Σ that is doubled is the same in M and in $\mathrm{D}[M]$. This is immediate by lemma 2.2.1(iii).
(ii) If M and $\mathrm{D}[\ ]$ satisfy the conditions of the lemma, then after the p^*-step $\mathrm{D}[M] \rightarrow_{\mathrm{p}}{}^* \mathrm{D}[N]$ this is still the case with N and $\mathrm{D}[\ ]$. So (i) can be repeated. □

TERMINOLOGY. In satifying lemma 2.6.4 we call the relations $\rightarrow_{\mathrm{p}}$ and $\rightarrow_{\mathrm{p}}{}^*$ *quasi compatible* with respect to $\Lambda\boldsymbol{\pi}\mathbf{p}^*$.

It should be kept in mind that neither p- nor and p^*-reduction are monotone. An α-bookkeeping pair is not allowed to be created by $\mathbf{p}$ in a β-context with $\beta > \alpha$. This is, however, less of a restriction than it may seem. In $\boldsymbol{\lambda\pi}\mathbf{p}^*$ the α-bookkeeping pair that is introduced in such a β-context would anyhow be cancelled at once by $\hookrightarrow^*$. As a matter of fact, it will be shown (cf. lemma 2.6.6) that $\rightarrow_{\mathrm{p}}{}^*$ is $*$-monotone. We prove this along with the $*$-compatibility of $\rightarrow_{\mathrm{p}}{}^*$ with the help of lemma 2.4.12.

2.6.5. LEMMA. (i) $\rightarrow_p$ is ½-monotone.
(ii) $\rightarrow_p$ is ½-compatible.
(iii)

$$\begin{tikzcd}
M \arrow[r, "(\Sigma)\rightarrow_p"] \arrow[d, "(\Delta)\hookrightarrow"'] & N \arrow[d, dashed, "\hookrightarrow\!\!\!\twoheadrightarrow"] \\
P \arrow[r, dashed, "\Rightarrow_p"'] & Q
\end{tikzcd}$$

PROOF. (i) Assume $\alpha. M \equiv C[\Sigma]_\beta \rightarrow_p C[\lceil\Sigma, \Sigma\rceil_\beta] \equiv N$, and $\alpha \leq \alpha'$. Then by 2.2.1(i) we have for $\beta' = \ell(\alpha'.C[\ ])$ that $\beta \leq \beta'$. Now it is easy to specify an N' such that $\alpha'. M \Rightarrow_p N'$, distinguishing two subcases.
a. $\beta' > \beta$. Take $N' \equiv M$, the new bookkeeping pair can be cancelled.
b. $\beta' = \beta$. Take just $N' \equiv N$.
(ii) This proof is similar, now with the help of lemma 2.2.1(ii).
(iii) As Σ cannot itself be a bookkeeping pair by the restriction on $\rightarrow_p$, the possibility of $\Sigma \equiv \Delta$ is ruled out. The remaining cases are:
- $\Delta \subsetneq \Sigma$. Then, if $M \equiv C[\Sigma]_\alpha$ and Δ: $\alpha.\Sigma \hookrightarrow \Sigma'$, by the monotonocity of $\hookrightarrow$ the diagram

$$\begin{tikzcd}
C[\Sigma] \arrow[r, "\rightarrow_p"] \arrow[d, "\hookrightarrow"'] & C[\lceil\Sigma, \Sigma\rceil_\alpha] \arrow[d, dashed, "\hookrightarrow\!\!\!\twoheadrightarrow"] \\
C[\Sigma'] \arrow[r, dashed, "\rightarrow_p"'] & C[\lceil\Sigma', \Sigma'\rceil_\alpha]
\end{tikzcd}$$

holds.
- $\Sigma \subsetneq \Delta$. Assume without loss of generality that $M \equiv C[\lceil X_0, X_1\rceil_\beta]_{\beta 0 \beta'} \hookrightarrow C[X_0] \equiv P$, and (the case $\Sigma \subsetneq X_1$ being trivial) assume $\Sigma \subsetneq X_0$. That is, $N \equiv C[\lceil Y_0, X_1\rceil_\beta]_{\beta 0 \beta'}$ and $\beta 0. X_0 \rightarrow_p Y_0$. Then, since $\rightarrow_p$ is already ½+½ by (i) and (ii), there exists by 2.4.14 a Q such that $C[X_0] \Rightarrow_p Q$ and $C[Y_0] \hookrightarrow\!\!\!\twoheadrightarrow Q$. But then $N \hookrightarrow C[Y_0] \hookrightarrow\!\!\!\twoheadrightarrow Q$.
- $\Sigma \cap \Delta = \emptyset$. As the p-redex Σ cannot be in applicative position, by 2.2.5(i) the label of Δ is invariant under contraction of Σ. Moreover, now by 2.2.3(ii), the label of Σ does not diminish under (Δ). So we can reason analogous to the foregoing case, using the fact that $\rightarrow_p$ is ½+½. □

2.6.6. LEMMA. (i) $\rightarrow_p{}^*$ and $\twoheadrightarrow_p{}^*$ are $*$-monotone and $*$-compatible.
(ii) $\rightarrow^*$ and $\twoheadrightarrow^*$ are both $*$-monotone and $*$-compatible.

PROOF. (i) The $*$-monotonicity of $\rightarrow_p$ and $\twoheadrightarrow_p$ can by lemma 2.6.5 be concluded from 2.4.12(ii); the $*$-compatibility from 2.4.12(iii). In order to derive the result for the starred versions, use 2.4.13(ii).
(ii) The results of clause (i) and lemma 2.6.3(ii) can be combined, using 2.4.13(iii) for the transitive closure. □

§3. The Church-Rosser theorem for the system $\lambda\pi p^*$

All well known proofs of the Church-Rosser theorem have the same global structure. An auxiliary one step reduction relation $\to_1$ is defined, which consists, instead of just contracting a single redex, in an immediate jump to the complete development with respect to an arbitrary set of redexes. One then proves that $\to_1$ satisfies the diamond property. From that the Church-Rosser property for $\to$ can be deduced at once. By lemma 0.7.1 it suffices to verify the obvious inclusions $\to \subseteq \to_1$ and $\to_1 \subseteq \twoheadrightarrow$.

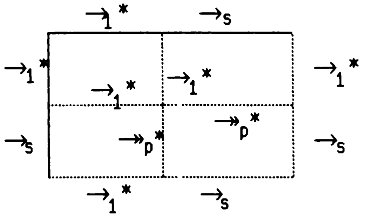
The differences between the various proofs lie mainly in the way $\to_1$ is arrived at—by a Tait/Martin-Löf type direct definition for example, or via the finite developments theorem—and, correspondingly, in the proof of the diamond property for $\to_1$.

It is essential in this kind of set up, that residuals of redexes are redexes again, of the same type as the ancestor. We will see in section 3.2, however, that under $\to^*$ this is not always the case. The constants involved in an existing π-redex may become separated by the bookkeeping pair which is introduced in a $\to_p{}^*$-step.

We deal with this problem by segregating $\to_p{}^*$-reduction from the other, substantial, reduction rules. Thus we exploit two complementary concepts of fast one step reduction. In the first place there is $\to_1{}^*$, derived from $\to_\pi{}^*$ in a more or less standard way. In addition, in section 3.2 a notion $\to_s$ of "simplifying" p^*-reduction will be defined. It is the restriction of $\twoheadrightarrow_p{}^*$ obtained by requiring that in the end term no redexes are disturbed by occurrences of bookkeeping pairs.

Then the role of the "one step" reduction relations in the traditional Church-Rosser proofs is played here by the relation $\to_+$ defined as the sum $\to_1{}^* + \to_s$. Accordingly we shall prove the diamond property for $\to_+$. The structure of the proof is best described by way of the following diagram.

3.0.1. DIAGRAM.



Establishing 3.0.1 is our main task for this section. It can be divided into three parts, corresponding to the different rectangles in the diagram. The first, left upper rectangle asks for a more or less standard treatment, using the finite developments theorem. Some

complications are caused by the ambiguity of the contraction rules (cf. section 3.1). The fourth, the right downmost one, will be dealt with in 3.2. Moreover, in that section the treatment of the identical second and third rectangles is prepared by the introduction of the special species of $\twoheadrightarrow_p^*$-reduction $\rightarrow_s$. Subsequently these rectangles are attained in section 3.3.

In 3.4 Church-Rosser for $\rightarrow^*$ can then be concluded. For, though we do not have $\rightarrow^* \subseteq \rightarrow_+$, it will turn out that the convertibility relation $=_+$ which is generated by $\rightarrow_+$, coincides with $=^*$. As $\rightarrow_+ \subseteq \twoheadrightarrow^*$ does hold all right, the second version of the Church-Rosser theorem, 0.6.2, follows.

3.1. Marked $\rightarrow_\pi$-reduction. In this section we prove $\rightarrow_\pi^*$-Church-Rosser by the method which uses the finite developments theorem to arrive at $\rightarrow_1$, and marked reduction in order to encode developments with respect to sets of redexes (compare Barendregt [1981], Ch.11, § 2). Due to the ambiguity of the rules some adjustments have to be made, however.

As a matter of fact, by considering terms in $\Lambda\pi\mathbf{p}$ modulo ~-equivalence, we tailored $\boldsymbol{\lambda\pi\mathrm{p}}$ minus p-reduction as a weakly regular CRS with (stable) conditions. Church-Rosser for regular Term Rewriting Systems with conditions of a certain kind (not the ones here encountered) is proved in Bergstra & Klop [1982]. The authors express the belief that their results will carry over to weakly regular TRS's as well. Quite in general, the opinion seems to prevail that the Church-Rosser theorem and related results for regular CRS's generalize easily to the weakly regular case. Nevertheless, there appears to exist no actual treatment of the weakly regular systems in print. Accordingly, it may be worthwhile to call attention to the complications described in 3.1.1 below in defining a coherent notion of residual.

We sketch our proof of $\rightarrow_\pi$-Church-Rosser as an intermediate step towards $\rightarrow_\pi^*$-Church-Rosser. The key notion in the proof, that of marked reduction, is needed also in section 3.3 treating of the interference of $\rightarrow_\pi^*$ and $\rightarrow_p^*$, and in the final §4.

Generally the purpose of using some notion of marked (or underlined) reduction is to save the trouble of having to be very precise on residuals. Indeed, we could do without that also here. But, the definition of $\rightarrow_{\pi'}$ on its own being rather cumbersome, we will try to facilitate understanding by relating it explicitly to the matter it is meant to encode.

3.1.1. Recall the diagrams of section 1.8, which served to illustrate the cases of overlap between the contraction rules $\boldsymbol{\pi_0}$ and $\mathbf{l}$ (and $\boldsymbol{\pi_1}$ and $\mathbf{r}$). Everything seemed all right there, as the end terms were either identical (in (i)), or at least ~-equivalent (in (ii)). Looking closer though, one observes that the π_0's in the respective reducts $\pi_0 X$ in (i) descend from a different ancestor in the original term. The same is true of the π's in πXY and πXZ in diagram (ii). This awkward subtlety gives rise to a serious problem in tracing a possible third redex in which one of these constants is involved.

We sketch the situation in the following two diagrams, closely related to the ones in 1.8. (Only $\boldsymbol{\pi_0}$ and **l** are covered, again, but the case of $\boldsymbol{\pi_1}$ and **r** is completely analogous.)

(i)

$$\begin{tikzcd}
 & C[\pi_0^1(\pi^2(\pi_0^3(\pi^4XY))Z)] \arrow[dl, "(23)\to_{l}"'] \arrow[dr, "(12)\to_{\pi_0}"] & \\
C[\pi_0^1(\pi^4XY)] & & C[\pi_0^3(\pi^4XY)] \arrow[d, "(34)\to_{\pi_0}"'] \\
 & & C[X]
\end{tikzcd}$$

(ii)

$$\begin{tikzcd}
 & C[\pi^1(\pi_0^2(\pi^3(\pi_0^4X)Y))Z] \arrow[dl, "(23)\to_{\pi_0}"'] \arrow[dr, "(12)\to_{l}"] & \\
C[\pi^1(\pi_0^4X)Z] & & C[\pi^3(\pi_0^4X)Y] \arrow[d, "(34)\to_{l}"'] \\
 & & C[X]
\end{tikzcd}$$

It must be assumed of course in (i) that $Y = Z$, and in (ii) that $\pi_1 X = Y = Z$.

We have attached numerals to the constants involved in the reductions and for this occasion indicate a redex by the combination of numerals attached to the constants that constitute the redex in question. (That is, e.g. the redex $\pi_0{}^3(\pi^4 XY)$ is indicated as 34.) So in the original terms of both diagrams we can distinguish redexes 12, 23 and 34.

Now just concentrate on diagram (i). It is clear that in the result $C[\pi_0{}^3(\pi^4 XY)]$ of reduction step (12) the redex 34 is residual of the redex 34 in the original term, whereas in the result $C[\pi_0{}^1(\pi^4 XY)]$ of (23) no such residual exists. In the notation which was pointed out in §0: $34/(12) = 34$ and $34/(23) = \emptyset$. Hence it appears that the reduction sequences (23) and (12)+(34) are both complete developments with respect to the set of redexes $\{12, 23, 34\}$. But the end terms $C[\pi_0(\pi XY)]$ and $C[X]$ are not the same and as a consequence it becomes apparent that FD! for $\to_\pi$ fails.

Can it not be repaired? To this end we declare $\pi_0{}^1(\pi^4 XY)$ to be a *virtual residual* of the redexes 12 and 34 under (23). In the same way in diagram (ii) the redex 14 in $C[\pi^1(\pi^4 X)Z]$ is to be considered as the virtual residual of 12 and 34 under (23).

3.1.2. TERMINOLOGY. Call the occurrences of constants that are required to constitute a redex the *critical constants* of that redex. (E.g. in $\pi_0(\pi\, \pi_1\pi_1)$ the critical constants are π_0 and π; in general the critical constants are the ones that are displayed in the contraction rules, cf. 2.5.1.) Notice that the redexes Σ and Δ in M *overlap* if they share one critical constant. Now, given a set $\mathfrak{R}$ of redex occurrences in M, we define an $\mathfrak{R}$-*chain* to be a maximal set $\{\Sigma_1, \ldots, \Sigma_n\} \subseteq \mathfrak{R}$, $n > 0$, such that for each $i < n$, Σ_i and Σ_{i+1} overlap. The $\mathfrak{R}$-chains form a partition of $\mathfrak{R}$. Since β-redexes have no overlap, they constitute an $\mathfrak{R}$-chain each on their own. An *inner redex* of $\mathfrak{R}$ is one which overlaps with two other redexes in $\mathfrak{R}$; these redexes will then belong to the same $\mathfrak{R}$-chain. As explicated in 3.1.1, contraction of an inner redex of $\mathfrak{R}$ leaves only a *virtual residual* of its immediate neighbour redexes.

3.1.3. DEFINITION. (i) Given a pair $<M, \mathfrak{R}>$, $M \in \Lambda\boldsymbol{\pi}\mathbf{p}$, $\mathfrak{R}$ a set of $\rightarrow_\pi$-redexes in M, and a term N such that Σ: $M \rightarrow_\pi N$ for a $\Sigma \in \mathfrak{R}$, the set $\mathfrak{R}/(\Sigma)$ of residuals of $\mathfrak{R}$ in N is defined to consist of:

a The residuals of the elements of $\mathfrak{R}$, that is, for each $\Delta \in \mathfrak{R}$ the set $\Delta/(\Sigma)$,

and, if Σ was an inner redex of $\mathfrak{R}$,

b the virtual residual of the immediate neighbours of Σ.

(ii) Let $M \in \Lambda\boldsymbol{\pi}\mathbf{p}$ and $\mathfrak{R}$ a set of $\rightarrow_\pi$-redex occurrences in M. Then a *development* of $<M, \mathfrak{R}>$ is a sequence $<M_0, \mathfrak{R}_0>, <M_1, \mathfrak{R}_1>, \ldots$ $<M_n, \mathfrak{R}_n>$, with $M_i \in \Lambda\boldsymbol{\pi}\mathbf{p}$ and $\mathfrak{R}_i$ a set of redex occurrences in M_i, such that

$$(\forall i<n)((\exists \Sigma_i \in \mathfrak{R}_i)(M_i \, {}^{(\Sigma_i)}{\rightarrow_\pi} \, M_{i+1}) \;\&\; \mathfrak{R}_{i+1} = \mathfrak{R}_i/(\Sigma_i)).$$

The sequence $M_0, M_1, \ldots M_n$ is called a *development* of M with respect to $\mathfrak{R}$. If moreover $\mathfrak{R}_n = \emptyset$, then the development is a *complete* one.

Note that as a result of this definition, "set of residuals of" is no longer a distributive notion. The identity $\mathfrak{R}/(\Sigma) = \cup\{\Delta/(\Sigma) \mid \Delta \in \mathfrak{R}\}$ does not hold in general.

We now give a formalization of all this by way of appropriate concepts of marked term and marked reduction.

3.1.4. DEFINITION. The pair $<M, \mathfrak{R}>$ is represented by the marked term which is obtained by attaching primes ('), apostrophes (’) and inverse apostrophes (‘) to λ's and critical constants occurring in M in the following way.

(i) the initial λ of a redex $(\lambda x. N_0) N_1$ in $\mathfrak{R}$ is primed (result: $(\lambda' x. N_0) N_1$),

(ii) the leftmost critical constant of each $\mathfrak{R}$-chain gets ‘ (result: $\ldots \pi_{(i)}$‘$\ldots$),

(iii) the rightmost critical constant of each $\mathfrak{R}$-chain gets an ’ (result: $\ldots \pi_{(i)}$’$\ldots$),

(iv) all other critical constants of π-redexes in $\mathfrak{R}$ are primed
$(\ldots\pi_{(i)}'\ldots)$.

The marked terms that are thus obtained as representing pairs $<M,\mathfrak{R}>$, constitute the set $\Lambda'\boldsymbol{\pi}\mathbf{p}$. The restriction of $\Lambda'\boldsymbol{\pi}\mathbf{p}$ to canonical M's is denoted by $\Lambda'\boldsymbol{\pi}\mathbf{p}^*$.

COMMENT. An $\mathfrak{R}$-chain in M can be recognized in the representing marked variant of M in $\Lambda'\boldsymbol{\pi}\mathbf{p}$ because all its critical constants are marked. Such a chain of marked $\pi‘_{(i)}$'s is called a π-chain. The apostrophes play the role of begin (‘) and end (’) markers for π-chains. This feature is necessary for marking the kind of difference that exists e.g. between the marked terms $\pi_0‘(\pi'(\pi_0'(\pi’XY))Z)$ and $\pi_0‘(\pi’(\pi_0‘(\pi’XY))Z)$, the first one representing one $\mathfrak{R}$-chain of length 3, the second two $\mathfrak{R}$-chains, each o: length 1.

Observe that the "represent" relation between $\Lambda'\boldsymbol{\pi}\mathbf{p}$ and the pairs $<M,\mathfrak{R}>$ is one-one. Hence the pair $<M,\mathfrak{R}>$ and the marked variant of M representing it can be identified.

The set $\Lambda'\boldsymbol{\pi}\mathbf{p}$ could be given a direct inductive definition, without reference to sets of redexes, in the following way.

3.1.5. ALTERNATIVE DEFINITION. $\Lambda'\boldsymbol{\pi}\mathbf{p}$ is defined as the set of the *marked variants* of terms $M\in\Lambda\boldsymbol{\pi}\mathbf{p}$ The marked variants of M are defined by induction on the number of marks, according to the following clauses.

(i) $M\in\Lambda\boldsymbol{\pi}\mathbf{p} \Rightarrow M\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

(ii) $\underline{a}$. $C[(\lambda x.N_0)N_1]\in\Lambda'\boldsymbol{\pi}\mathbf{p} \Rightarrow C[(\lambda' x.N_0)N_1]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

$\underline{b}_1$. $C[\pi_i.\pi XY]\in\Lambda'\boldsymbol{\pi}\mathbf{p} \Rightarrow C[\pi_i‘.\pi’XY]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

$_2$. $C[\pi_i.\pi‘XY]\in\Lambda'\boldsymbol{\pi}\mathbf{p} \Rightarrow C[\pi_i‘.\pi'XY]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

$\underline{c}_1$. $C[\pi(\pi_0X)\ Y]_{0\alpha}\in\Lambda'\boldsymbol{\pi}\mathbf{p}\ \&\ \pi_1X=Y \Rightarrow C[\pi‘(\pi_0’X)\ Y]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

$_2$. $C[\pi(\pi_0‘X)\ Y]_{0\alpha}\in\Lambda'\boldsymbol{\pi}\mathbf{p}\ \&\ \pi_1X=Y \Rightarrow C[\pi‘(\pi_0'X)\ Y]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

$\underline{d}_1$. $C[\pi\ Y(\pi_1X)]_{1\alpha}\in\Lambda'\boldsymbol{\pi}\mathbf{p}\ \&\ \pi_0X=Y \Rightarrow C[\pi‘\ Y(\pi_1’X)]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

$_2$. $C[\pi\ Y(\pi_1‘X)]_{1\alpha}\in\Lambda'\boldsymbol{\pi}\mathbf{p}\ \&\ \pi_0X=Y \Rightarrow C[\pi‘\ Y(\pi_1'X)]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

The rules $\underline{b}_3$, $\underline{c}_3$ and $\underline{d}_3$ are now redundant:

$\underline{b}_3$. $C[\pi_i’.\pi XY]\in\Lambda'\boldsymbol{\pi}\mathbf{p} \Rightarrow C[\pi_i'.\pi’XY]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

$\underline{c}_3$. $C[\pi’(\pi_0X)\ Y]_{0\alpha}\in\Lambda'\boldsymbol{\pi}\mathbf{p}\ \&\ \pi_1X=Y \Rightarrow C[\pi'(\pi_0’X)\ Y]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

$\underline{d}_3$. $C[\pi’\ Y(\pi_1X)]_{1\alpha}\in\Lambda'\boldsymbol{\pi}\mathbf{p}\ \&\ \pi_0X=Y \Rightarrow C[\pi'\ Y(\pi_1’X)]\in\Lambda'\boldsymbol{\pi}\mathbf{p}$

If $N\in\Lambda'\boldsymbol{\pi}\mathbf{p}$, then $|N|$ is the corresponding term in $\Lambda\boldsymbol{\pi}\mathbf{p}$ which is obtained by deleting all marks. $\Lambda'\boldsymbol{\pi}\mathbf{p}^* = \{N\in\Lambda'\boldsymbol{\pi}\mathbf{p} \mid |N|\in\Lambda\boldsymbol{\pi}\mathbf{p}^*\}$.

Finally, developments are covered in $\Lambda'\boldsymbol{\pi}\mathbf{p}$ by the concept of marked reduction $\rightarrow_{\pi'}$. It can be def:ned informally as follows. (In the line of 3.1.5 a less verbose formal inductive definition could be given. We leave this to the diligent reader.)

3.1.6. DEFINITION. $\to_{\pi'}$ is the one step reduction relation on $\Lambda'\boldsymbol{\pi}\mathbf{p}$ which is derived from $\to_{\pi}$ by restricting:

- β-contraction to redexes of which the initial λ is primed: $(\lambda' x.N_0)\,N_1 \to_{\pi'} (x{:=}N_1)\,N_0$, and
- the rules $\boldsymbol{\pi}_\mathbf{0}$, $\boldsymbol{\pi}_\mathbf{1}$, **l** and **r** to redexes of which the critical constants are marked in that order by either ‘ and ', ' and ', ' and ’, or ‘ and ’.

If a leftmost (or rightmost) critical constant of a π-chain of length at least four is involved in the contraction, its mark ‘ (or ’) is passed on to the leftmost (or rightmost) critical constant in the residual π-chain. If the original π-chain contained only one or two redexes (two or three critical constans respectively) no residuals remain. Hence in the case of three critical constants, the mark of the single critical constant that is not involved in the contraction (it must be either ‘ or ’) is cancelled in the reduct.

EXAMPLES. $\pi_0\text{‘}(\pi'(\pi_0'(\pi\text{’}XY))\;Y) \to_{\pi'} \pi_0\text{‘}(\pi\text{’}XY)$ (in three ways); $\pi_0\text{‘}(\pi\text{’}(\pi_0\text{‘}(\pi\text{’}XY))\;Y) \to_{\pi'} \pi_0\text{‘}(\pi\text{’}XY)$ (in two ways); $\pi_0\text{‘}(\pi'(\pi_0\text{’}X)(\pi_1 X)) \to_{\pi'} \pi_0 X$ (in two ways).

3.1.7. LEMMA. (i) $\Lambda'\boldsymbol{\pi}\mathbf{p}$ is closed under $\to_{\pi'}$.
(ii) $\Lambda'\boldsymbol{\pi}\mathbf{p}$ is closed under $\hookrightarrow, \hookrightarrow\!\!\!\!\twoheadrightarrow$ and $\hookrightarrow^*$.

PROOF. Straightforward, the system is so designed. For β-reduction check first that $\Lambda'\boldsymbol{\pi}\mathbf{p}$ is closed under substitution. □

Any notion of reduction on $\Lambda\boldsymbol{\pi}\mathbf{p}$ or $\Lambda\boldsymbol{\pi}\mathbf{p}^*$ we met so far can be considered to be extended to marked terms, by allowing the metavariables that occur in the contraction rules to carry marks. Thus, if Δ: $M \to_a N$, for $M, N \in \Lambda\boldsymbol{\pi}\mathbf{p}$, then if $|M_0| \equiv M$, there exists an N_0 with $|N_0| \equiv N$ such that Δ_0: $M_0 \to_a N_0$, where Δ_0 is the redex occurrence in N_0 that corresponds to Δ. Passing from (Δ) to (Δ_0) is called *lifting*. Conversely, passing from (Δ_0) to (Δ) is called *projecting* (cf. Barendregt [1981], pp. 279/80). It should be noted though, that $\Lambda'\boldsymbol{\pi}\mathbf{p}$ is not a priori closed under any kind of reduction. (This is why I speak here, somewhat loosely, of "marked terms", rather than $\Lambda'\boldsymbol{\pi}\mathbf{p}$.) As a matter of fact, it is easily seen that $\Lambda'\boldsymbol{\pi}\mathbf{p}$ is not closed under $\to_p$. We return to this point in section 3.3.

If $\mathfrak{R}_1 \subseteq \mathfrak{R}_2$, then each development of $<M, \mathfrak{R}_1>$ with result $<N, \mathfrak{R}_3>$ can be lifted to a development of $<M, \mathfrak{R}_2>$, consisting of the same reduction steps. For the result $<N, \mathfrak{R}_4>$ we have $\mathfrak{R}_3 \subseteq \mathfrak{R}_4$. In particular all developments of $<M, \mathfrak{R}_1>$ and $<M, \mathfrak{R}_2>$, with $\mathfrak{R}_1$ and $\mathfrak{R}_2$ arbitrary sets of redex occurrences in M, can be lifted to developments of $<M,\mathfrak{R}_1\cup\mathfrak{R}_2>$.

If M_1 and $M_2\in\Lambda'\boldsymbol{\pi}\mathbf{p}$ represent respectively $<M,\mathfrak{R}_1>$ and $<M,\mathfrak{R}_2>$, then the element of $\Lambda'\boldsymbol{\pi}\mathbf{p}$ which represents $<M,\mathfrak{R}_1\cup\mathfrak{R}_2>$, denoted by M_1+M_2, can be found as follows:

- the λ's that are primed in at least one of M_1 and M_2 are primed in M_1+M_2 as well,
- the π's having the same mark in M_1 and in M_2 get that mark in M_1+M_2 as well,
- the π's that are marked both in M_1 and M_2, but with a different symbol in each, are primed in M_1+M_2.

3.1.8. THEOREM (FD!). For each $M\in\Lambda'\boldsymbol{\pi}\mathbf{p}$:
(i) the number of steps in an $\to_{\pi'}$-sequence is finite;
(ii) there exists a unique $\to_{\pi'}$-normal form N such that $M\twoheadrightarrow_{\pi'}N$.

PROOF. (i) It is not difficult to adapt the proof of the finite developments theorem in de Vrijer [1985]. Primarily the definitions of height (h) and multiplicity (m_x) must be modified such that

- h counts also the reduction steps originating in a π-chain (= the entier of half the length of the π-chain);
- both h and m_x neglect transient occurrences.

The circumstance that some of the contraction rules are subject to conditions can a priori only shorten complete reductions. In fact the effect is nihil, since the conditions are stable.
(ii) This is an immediate consequence of $\to_{\pi'}$-Church-Rosser, which we now prove by a standard argument. Suppose $M\twoheadrightarrow_{\pi'}N$ and $M\twoheadrightarrow_{\pi'}P$. By induction on $h(M)$, i.e. the maximal number of $\to_{\pi'}$-steps from M, we show that N and P have a common $\twoheadrightarrow_{\pi'}$-reduct Q. If one of the $\twoheadrightarrow_{\pi'}$-reductions is empty, then the result follows trivially. So assume terms N_1 and P_1 as indicated in the drawn part of the diagram below. Then the diagram can be completed as shown.

$$\begin{tikzcd}
M \arrow[r, "\pi'"] \arrow[d, "\pi'"'] \arrow[dr, phantom, "A"] & N_1 \arrow[r, two heads, "\pi'"] \arrow[d, dotted, two heads, "\pi'"] & N \arrow[d, dotted, two heads, "\pi'"] \\
P_1 \arrow[r, dotted, two heads, "\pi'"] \arrow[d, two heads, "\pi'"'] & {} \arrow[r, dotted, two heads, "\pi'"] \arrow[d, dotted, two heads, "\pi'"] & {} \arrow[d, dotted, two heads, "\pi'"] \\
P \arrow[r, dotted, two heads, "\pi'"'] & {} \arrow[r, dotted, two heads, "\pi'"'] & Q
\end{tikzcd}$$

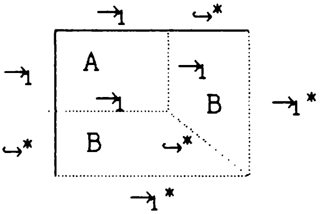
For, since the conditions on $\mathbf{l}$ and $\mathbf{r}$ are stable, A is just the routine construction of an elementary diagram. The other squares are derived by the induction hypothesis. □

3.1.9. Definition. (i) For $M \in \Lambda'\boldsymbol{\pi\mathrm{p}}$, the *height* $\mathrm{h}(M)$ is the number of reduction steps in a $\to_{\pi'}$-reduction sequence from M of maximal length.
(ii) Denote the unique $\to_{\pi'}$-normal form of $M \in \Lambda'\boldsymbol{\pi\mathrm{p}}$ by $\mathrm{CD}(M)$. Then the one step reduction relation $\to_1$ on $\Lambda\boldsymbol{\pi\mathrm{p}}$ is defined by

$$M \to_1 N \iff (\exists M' \in \Lambda'\boldsymbol{\pi\mathrm{p}})(|M'| \equiv M \ \&\ \mathrm{CD}(M') \equiv N).$$

3.1.10 Lemma. (i) $\to_1$ is self commuting.
(ii) $\to_1{}^*$ is self commuting.

Proof. (i) Standard again. Suppose that $M \to_1 N$ and $M \to_1 P$. Then there are $M_0, M_1 \in \Lambda'\boldsymbol{\pi\mathrm{p}}$ such that $|M_0| \equiv |M_1| \equiv M$, and complete developments σ_0: $M_0 \twoheadrightarrow_{\pi'} N$ and σ_1: $M_1 \twoheadrightarrow_{\pi'} P$. Let $M' \equiv M_0 + M_1$. Then σ_0 and σ_1 can be lifted to developments σ_0': $M' \twoheadrightarrow_{\pi'} N'$ and σ_1': $M' \twoheadrightarrow_{\pi'} P'$, with $|N'| \equiv N$ and $|P'| \equiv P$. By FD! it follows that $\mathrm{CD}(M') \equiv \mathrm{CD}(N') \equiv \mathrm{CD}(P')$ and hence $\mathrm{CD}(M')$ is a common $\to_1$-reduct of N and P.
(ii) We have the following diagram



A is an application of (i). As to the rectangles marked B it is sufficient to note that the complete development underlying a $\to_1$-step is $*$-projectable, by the same token as $\twoheadrightarrow_\pi$ is $*$-projectable. □

3.1.11 Corollary. $\to_\pi{}^*$ satisfies the Church-Rosser property.

Proof. In order to conclude the Church-Rosser property for $\to_\pi{}^*$ from lemma 0.7.1 with 3.1.10(ii), it suffices to verify the inclusions $\to_\pi{}^* \subseteq \to_1{}^*$ and $\to_1{}^* \subseteq \twoheadrightarrow_\pi{}^*$. Well, if $M \,{}^{(\Sigma)}\!\!\to_\pi{}^* N$, then $M \to_1{}^* N$, because $N \equiv \mathrm{CD}(M')$ for the M' which is obtained from M by marking only Σ. On the other hand, if $M \to_1{}^* N$, then of course $M \twoheadrightarrow_\pi + \hookrightarrow^* N$, as in general any $\to_{\pi'}$-sequence can be projected to a $\to_\pi$-sequence by just deleting all the marks. But then by 2.6.3(iii) also $M \twoheadrightarrow_\pi{}^* N$. □

For use in section 4 still note the following. It is by now a routine matter to extend FD! to $\boldsymbol{\lambda\pi\mathrm{p}^*}$. Thus we obtain the height function h^*, assigning to each $M \in \Lambda'\boldsymbol{\pi\mathrm{p}^*}$ the length of the longest $\to_{\pi'}{}^*$-sequence from M and the function CD^*, assigning to $M \in \Lambda'\boldsymbol{\pi\mathrm{p}^*}$ its unique $\to_{\pi'}{}^*$-normal form. Of course one has $\mathrm{h}^*(M) \leq \mathrm{h}(M)$ and $\mathrm{CD}^*(M) \equiv \mathrm{CD}(M)^*$.

3.2. The p-part. The Church-Rosser property for $\rightarrow_{p}^{*}$ is relatively easy to prove. It is done by the method indicated in section 0.7, using an auxiliary relation $\rightarrow_{p}^{*''}$ of simultaneous p*-reduction, which must first be defined. In order to finish the fourth rectangle, we then still need the concept of p-simplification. It is defined in the second part of this section. There also some further properties of p-reduction will be discussed as a preparation to section 3.3.

3.2.1. Definition. $\rightarrow_{p}^{*''}$ is the relation on $\Lambda\pi p^{*}$ which is generated by the inductive clauses:

<u>a</u>. $\alpha. M \underline{\rightarrow}_{p}^{*} N \Rightarrow M \rightarrow_{p}^{*''} N$,

<u>b</u>. $\alpha i. X_i \rightarrow_{p}^{*''} Y_i$, for $i=0,1 \Rightarrow C[\ulcorner X_0, X_1 \urcorner_\alpha]_\alpha \rightarrow_{p}^{*''} C[\ulcorner Y_0, Y_1 \urcorner_\alpha]$.

The definition satisfies the closure condition 2.1.9, because neither introducing a bookkeeping pair nor changing the content of a bookkeeping pair occurrence does affect the label and hence the “cancellability” of disjoint bookkeeping pairs (cf. lemma 2.2.3(ii)). One easily sees that $\rightarrow_{p}^{*''}$ is quasi compatible for the same reasons as $\rightarrow_{p}^{*}$ (cf. 2.6.4).

3.2.2. Lemma. (i) $\rightarrow_{p}^{*''}$ is $*$-monotone;
(ii) $\rightarrow_{p}^{*} \subseteq \rightarrow_{p}^{*''}$, and $\rightarrow_{p}^{*''} \subseteq \twoheadrightarrow_{p}^{*}$.

Proof. (i) The proof is by induction on definition 3.2.1. Clause <u>a</u> is taken care of by the $*$-monotonicity of $\rightarrow_{p}^{*}$. As to <u>b</u>, suppose that $\beta. C[\ulcorner X_0, X_1 \urcorner_\alpha]_\alpha \rightarrow_{p}^{*''} C[\ulcorner Y_0, Y_1 \urcorner_\alpha]$ follows from $\alpha i. X_i \rightarrow_{p}^{*''} Y_i$. Let $\beta' > \beta$. By the kind of reasoning we practiced in section 2.2, it is easily seen that the context C[ ] is of such a character that the following holds: there exists a context $\beta'. D[\ ]_{\alpha'}$ (with $\alpha \leq \alpha'$) such that regardless of P one has $\beta' * C[P]_\alpha \equiv D[\alpha' * P]_{\alpha'}$. (E.g. lemma 2.2.5(i) can be used, since in $\Lambda\pi p^{*}$ bookkeeping pairs never occur in applicative position.) By this fact—derived from the special form of C[ ]—an application of the induction hypothesis is straightforward: for example if $\alpha' = \alpha 0 \alpha''$, then $\beta'. D[\alpha' * X_0]_{\alpha'} \rightarrow_{p}^{*''} D[\alpha' * Y_0]_{\alpha'}$ results at once from $\alpha' * X_0 \rightarrow_{p}^{*''} \alpha' * Y_0$.
(ii) Easy. □

The diagram that is required for an applicaton of lemma 0.7.1 is provided for by the next lemma.

3.2.3. **Lemma**.

$$\begin{tikzcd}[column sep=huge, row sep=huge]
M \arrow[r, "\rightarrow_p{}^{*'}"] \arrow[d, "(\Delta)\rightarrow_p{}^{*}"'] & N \arrow[d, dashed, two heads, "\twoheadrightarrow_p{}^{*}"] \\
P \arrow[r, dashed, "\rightarrow_p{}^{*'}"'] & Q
\end{tikzcd}$$

Proof. Induction on the definition of $\rightarrow_p{}^{*''}$ (definition 3.2.1). Let Σ be the occurrence in M that is being doubled in N in case $M \rightarrow_p{}^{*''} N$ is a consequence of the base clause $\underline{a}$ of 3.2.1. (The case that $M \rightarrow_p{}^{*''} N$ consists of an empty step is trivial and can therefore be neglected.) Otherwise let Σ be the maximal bookkeeping pair in M to which the inductive clause $\underline{b}$ was applied. Distinguish cases as to the relative position of Σ and Δ.
(i) $\Sigma \cap \Delta = \emptyset$. This case is trivial by 2.2.5(i). The labels of Σ and Δ are not at all affected by the disjoint reduction step.
(ii) $\Delta \subseteq \Sigma$. Depending on which clause of 3.2.1 was applied, there are two subcases.
$\underline{a}$. We have the drawn part of the diagram

$$\begin{tikzcd}[column sep=huge, row sep=huge]
\mathrm{C}[\Sigma]_\alpha \arrow[r, "\rightarrow_p{}^{*}"] \arrow[d, "\rightarrow_p{}^{*}"'] & \mathrm{C}[\lceil\Sigma^*,\Sigma^*\rceil_\alpha] \arrow[d, dashed, two heads, "\mathrm{A}"', "\twoheadrightarrow_p{}^{*}"] \\
\mathrm{C}[\Sigma'] \arrow[r, dashed, "\mathrm{B}", "\rightarrow_p{}^{*}"'] & \mathrm{C}[\lceil\Sigma'^*,\Sigma'^*\rceil_\alpha]
\end{tikzcd}$$

with $\alpha.\Sigma \rightarrow_p{}^* \Sigma'$. Then $\alpha i.\alpha i{*}\Sigma \Rrightarrow_p{}^* \alpha i{*}\Sigma'$ by the $*$-monotonicity of $\rightarrow_p{}^*$ (2.6.6(i)), and so A holds by the quasi compatibility of $\rightarrow_p{}^*$. B is a straightforward $\rightarrow_p{}^*$-reduction step.
$\underline{b}$. $M \equiv \mathrm{C}[\lceil M_0, M_1\rceil_\alpha] \rightarrow_p{}^{*''} \mathrm{C}[\lceil N_0, N_1\rceil_\alpha] \equiv N$, with $\alpha i.\ M_i \rightarrow_p{}^{*''} N_i$. Assume, without loss of generality, that $\Delta \subseteq M_0$ and Δ: $\alpha 0.\ M_0 \rightarrow_p{}^* P_0$. Apply the induction hypothesis to find a Q_0 such that

$$\begin{tikzcd}[column sep=huge, row sep=huge]
M_0 \arrow[r, "\rightarrow_p{}^{*'}"] \arrow[d, "\rightarrow_p{}^{*}"'] & N_0 \arrow[d, dashed, two heads, "\twoheadrightarrow_p{}^{*}"] \\
P_0 \arrow[r, dashed, "\rightarrow_p{}^{*'}"'] & Q_0
\end{tikzcd}$$

Then it follows by the quasi compatibility of $\rightarrow_p^*$ and $\rightarrow_p^{*"}$ that $C[\ulcorner Q_0, N_1\urcorner_\alpha]$ can be taken for the requisite common reduct Q.
(iii) $\Sigma \subseteq \Delta$. Let Σ: $\alpha.\Delta \rightarrow_p^{*"} \Delta_0$. The result is given by the diagram

$$\begin{tikzcd}
C[\Delta] \arrow[r, "\rightarrow_p^{*"}"] \arrow[d, "\rightarrow_p^*"'] & C[\Delta_0] \arrow[d, dotted, "B", "\rightarrow_p^*"'] \\
C[\ulcorner \Delta^*, \Delta^* \urcorner_\alpha] \arrow[r, dotted, "A", "\rightarrow_p^{*"}"'] & C[\ulcorner \Delta_0^*, \Delta_0^* \urcorner_\alpha]
\end{tikzcd}$$

Here A follows by the $*$-monotonicity and the quasi compatibility of $\rightarrow_p^{*"}$. B is just a single $\rightarrow_p^*$-step.
(iv) $\Sigma \equiv \Delta$. Then Σ must be the result of clause $\underline{a}$ of 3.2.1, because Δ can not be a bookkeeping pair. Hence either $N \equiv P$ or $N \equiv M$, and matters are trivialized. □

3.2.4. COROLLARY. $\rightarrow_p^*$ is Church-Rosser.

PROOF. Lemma 0.7.1 can be used to deduce that $\twoheadrightarrow_p^*$ is self commuting. The conditions $\rightarrow_p^* \subseteq \rightarrow_p^{*"}$ and $\rightarrow_p^{*"} \subseteq \twoheadrightarrow_p^*$ were already verified in 3.2.2(ii). And the diagram needed is that of 3.2.3. □

Even in canonical terms it is still possible that bookkeeping pairs stand in the way and obstruct reduction. The pertinent cases are covered in the following lemma, which shows in each case how to clean up.

3.2.5. LEMMA. The following are derived rules in $\boldsymbol{\lambda\pi p^*}$.

$$C[\pi\ulcorner X_0, X_1\urcorner_\alpha Y]_{0\alpha} \rightarrow_{p*} C[\ulcorner \pi X_0 Y, \pi X_1 Y\urcorner_{0\alpha}],$$
$$C[\pi Y\ulcorner X_0, X_1\urcorner_\alpha]_{1\alpha} \rightarrow_{p*} C[\ulcorner \pi Y X_0, \pi Y X_1\urcorner_{1\alpha}],$$
$$C[\pi_i\ulcorner X_0, X_1\urcorner_{i\alpha}]_\alpha \rightarrow_{p*} C[\ulcorner \pi_i X_0, \pi_i X_1\urcorner_\alpha].$$

PROOF. Each of the rules follows by just doubling the whole occurrence displayed and consequently cancelling the descendants of the original bookkeeping pair. □

The effect of these reduction steps is that the bookkeeping pair is as it were opened to that part of the expression which acts upon it as a function. That way a redex may be constituted of which the ingredients were still separated before the simplification was performed. (An example of this would be a simplifying p^*-step

$$<>.\pi_0\ulcorner \pi X_0 X_1, Y\urcorner_0 \rightarrow_p^* \ulcorner \pi_0(\pi X_0 X_1), \pi_0 Y\urcorner.)$$

Indeed the components of the bookkeeping pairs in the end terms are all potential π-redexes, which were blocked in the original.

Normal forms under the derived rules of 3.2.5 are called p-*simple*.

In p-simple terms no π-redexes are blocked anymore.

It is easy to verify that for each $M \in \Lambda\pi\mathbf{p}^*$ a p-simple form can be reached by performing a finite number of p*-steps of the above kind. As a matter of fact this normal form is unique. This does not interest us here, however. Rather, for the purpose of establishing Church-Rosser for $\rightarrow^*$, the following auxiliary one step reduction relation of non-unique p-simplification will turn out to be very useful.

3.2.6. Definition. The one step reduction relation $\rightarrow_s$ on $\Lambda\pi\mathbf{p}^*$ is defined by:

$$M \rightarrow_s N \;\Leftrightarrow\; M \twoheadrightarrow_p{}^* N \;\&\; N \text{ is p-simple.}$$

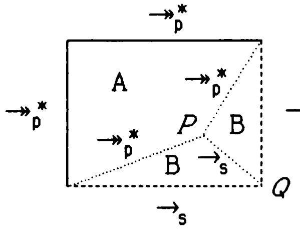
3.2.7. Lemma. For all $M \in \Lambda\pi\mathbf{p}^*$ there exists a term N, such that $M \rightarrow_s N$. This N is in general not unique.

Proof. Easy. □

3.2.8. Lemma.

$$\begin{tikzcd} {} \arrow[r, dash, "\twoheadrightarrow_p^*"] \arrow[d, dash, "\twoheadrightarrow_p^*"'] & {} \arrow[d, dash, dotted, "\rightarrow_s"] \\ {} \arrow[r, dash, dotted, "\rightarrow_s"'] & {} \end{tikzcd}$$

Proof. The proof is given by the diagram



A is $\rightarrow_p{}^*$-Church-Rosser. Q is found from P by the existence of p-simple forms. Then B follows (twice) because $\twoheadrightarrow_p{}^* + \rightarrow_s = \rightarrow_s$. □

The separate treatment of the "p-part" of the system **λπp*** in this section is concluded with a few further technical lemmas on p-reduction. The notions of "internal" and "external" occurrence, which we first define, play an important role in §4.

3.2.9. Definition. An occurrence is called *internal*, if it lies completely inside a bookkeeping pair, otherwise *external*.

3.2.10. LEMMA. Let Δ: $M \rightarrow_p{}^* N$ and suppose that $\Sigma = \langle P, \mathrm{C}[\ ]_\beta \rangle$ is an external occurrence in N. Then Σ's (unique) ancestor Q in M is external too and either
a $\Delta \subseteq Q$, in which case $M \equiv \mathrm{C}[Q]_\beta$ and $\beta.\, Q \rightarrow_p{}^* P$, or
b $\Delta \cap Q = \emptyset$, in which case $P \equiv Q$ and $\ell(Q) = \beta$.

PROOF. Since Σ is external the bookkeeping pair that is created by (Δ) must be either part of or disjoint with Σ. In the first case we have for some $\mathrm{D}[\ ]_\gamma$ that $P \equiv \mathrm{D}[\lceil \Delta^*, \Delta^* \rceil_\gamma]$ and $Q \equiv \mathrm{D}[\Delta]$. For the second case remind that by 2.2.5(i) the label $\ell(Q)$ is not affected by (Δ).
□

3.2.11. LEMMA. Let σ: $\alpha.\, M \twoheadrightarrow_p{}^* N$. Then we have one of the two following cases.
a N contains no occurrences of bookkeeping pairs and $M \equiv N$.
b N can be written in the form

$$\ldots \lceil X_1, Y_1 \rceil_{\alpha_1} \ldots \lceil X_2, Y_2 \rceil_{\alpha_2} \ldots\ldots \lceil X_n, Y_n \rceil_{\alpha n} \ldots,$$

with $n \geq 1$ and each external occurrence of a bookkeeping pair displayed as one of the $\lceil X_i, Y_i \rceil_{\alpha i}$'s, in which case M is of the form

$$\ldots Z_1 \ldots Z_2 \ldots\ldots Z_n \ldots,$$

coinciding with N on the dots, and such that for each of the i's:

$$\ell(Z_i) = \alpha_i \text{ and } \alpha_i.\, Z_i \twoheadrightarrow_p{}^* \lceil X_i, Y_i \rceil_{\alpha i}.$$

PROOF. With the help of the foregoing lemma it is not difficult to verify by induction on the length of σ the slightly more general statement that if

$$N \equiv \ldots P_1 \ldots P_2 \ldots\ldots P_n \ldots,$$

all the P_i's being external occurrences and $\ell(P_i) = \alpha_i$, then

$$M \equiv \ldots Z_1 \ldots Z_2 \ldots\ldots Z_n \ldots,$$

coinciding with N on the dots, and such that for each i:

$$\ell(Z_i) = \alpha_i \text{ and } \alpha_i.\, Z_i \twoheadrightarrow_p{}^* P_i. \quad □$$

The content of the next lemma is that a so called "main" p*-reduction, that is, a p*-step that doubles the whole term (cf. 0.3) can always be moved to the front of a reduction sequence. The lemma will be instrumental in the translation of $\rightarrow^*$-sequences into $\geq$-sequences in §4.

3.2.12. LEMMA (on main p-reduction).
(i) Let $\alpha.\, M \twoheadrightarrow_p{}^* \lceil N_0, N_1 \rceil_\alpha$. Then either
a. $M \equiv \lceil M_0, M_1 \rceil_\alpha$ and $\alpha i.\, M_i \twoheadrightarrow_p{}^* N_i$, or
b. $\alpha.\, M \rightarrow_p{}^* \lceil M_0, M_1 \rceil_\alpha$ with $M_i \equiv \alpha i * M$ and $\alpha i.\, M_i \twoheadrightarrow_p{}^* N_i$.

(ii) Let $\alpha . M \twoheadrightarrow^* \lceil N_0, N_1 \rceil_\alpha$. Then either

$\underline{a}$. $M \equiv \lceil M_0, M_1 \rceil_\alpha$ and $\alpha i . M_i \twoheadrightarrow^* N_i$, or

$\underline{b}$. $\alpha . M \rightarrow_{\mathrm{p}}{}^* \lceil M_0, M_1 \rceil_\alpha$ as in (i) and $\alpha i . M_i \twoheadrightarrow^* N_i$.

PROOF. (i) If M is a bookkeeping pair, case $\underline{a}$ trivially applies. If not, then of course $\alpha . M \rightarrow_{\mathrm{p}}{}^* \lceil M^*, M^* \rceil_\alpha$ by a main p^*-step and $\alpha i . M^* \twoheadrightarrow^* (\lceil N_0, N_1 \rceil_\alpha)^* \equiv N_i$ follows from $\alpha . M \twoheadrightarrow_{\mathrm{p}}{}^* \lceil N_0, N_1 \rceil_\alpha$ by the $*$-monotonicity of $\twoheadrightarrow_{\mathrm{p}}{}^*$ (cf. 2.6.6(i)) and because $\alpha i . N_i$ is canonical, as it occurs in an αi-context in the canonical $\lceil N_0, N_1 \rceil_\alpha$.
(ii) The same reasoning, now with 2.6.6(ii). □

3.3. Weaving $\rightarrow_*$ and $\rightarrow_{\mathrm{p}}$. For the completion of the second and third rectangles in the target diagram 3.0.1, we make use of a notion $\rightarrow_{\pi''}$ of simultaneous π'-reduction. It may be compared to the relation $\rightarrow_{\mathrm{p}}{}^{*''}$, defined in 3.2.1.

3.3.1. DEFINITION. $\rightarrow_{\pi''}$ is the least monotone and compatible relation on $\Lambda'\boldsymbol{\pi}\mathbf{p}$ which satisfies the inductive clauses:

$\underline{a}$. $\alpha . M \mathrel{\underline{\underline{\rightarrow}}}_{\pi'} N \Rightarrow M \rightarrow_{\pi''} N$;

$\underline{b}$. $\alpha i . X_i \rightarrow_{\pi''} Y_i$, for $i=0,1$ $\Rightarrow$ $\alpha . \lceil X_0, X_1 \rceil_\alpha \rightarrow_{\pi''} \lceil Y_0, Y_1 \rceil_\alpha$.

3.3.2. LEMMA. (i) $\rightarrow_{\pi''}$ is $*$-projectable;
(ii) clause $\underline{b}$ of 3.3.1 is also correct for $\rightarrow_{\pi''}{}^{½}$, that is, that we have:

$$\alpha i . X_i \rightarrow_{\pi''}{}^{½} Y_i, \text{ for } i=0,1 \;\Rightarrow\; \alpha . \lceil X_0, X_1 \rceil_\alpha \rightarrow_{\pi''}{}^{½} \lceil Y_0, Y_1 \rceil_\alpha;$$

(iii) $\rightarrow_{\pi'}{}^* \subseteq \rightarrow_{\pi''}{}^*$ and $\rightarrow_{\pi''}{}^* \subseteq \twoheadrightarrow_{\pi'}{}^*$.

PROOF. (i) Use lemma 2.4.12(i). That $\rightarrow_{\pi''}$ is ½-projectable can be easily verified by induction on definition 3.3.1.
(ii) is straightforward.
(iii) $\rightarrow_{\pi'}{}^* \subseteq \rightarrow_{\pi''}{}^*$ by clause $\underline{a}$ of 3.3.1. We derive $\rightarrow_{\pi''}{}^* \subseteq \twoheadrightarrow_{\pi'}{}^*$ from the more obvious $\rightarrow_{\pi''} \subseteq \twoheadrightarrow_{\pi'}$. Assume $M \rightarrow_{\pi''}{}^* N$, i.e., $M \in \Lambda\boldsymbol{\pi}\mathbf{p}^*$ and $M \rightarrow_{\pi''} N' \hookrightarrow^* N$ for some N'. Then $M \twoheadrightarrow_{\pi'}{}^* N$ follows from $M \twoheadrightarrow_{\pi'} N' \hookrightarrow^* N$ since $\twoheadrightarrow_{\pi'}$ is $*$-projectable (cf. lemma 2.6.3(iii)). □

Observe that $\Lambda'\boldsymbol{\pi}\mathbf{p}$ is not closed under the rule of bookkeeping pair introduction $\mathbf{p}$ (cf. 2.5.1(ii)). A new bookkeeping pair might break a π-chain and thereby disturb a marked redex. An example would be the p-step

$$\alpha . \pi_0{}'(\pi' XY) \rightarrow_{\mathrm{p}} \pi_0{}' \lceil \pi' XY, \pi' XY \rceil_{0\alpha}.$$

Therefore we work on $\Lambda'\boldsymbol{\pi}\mathbf{p}$ with a restricted version $\rightarrow_{\mathrm{p}'}$ of $\rightarrow_{\mathrm{p}}$. It must be kept in mind then, that if $|M'| \equiv M$, not just any p-reduction sequence from M can be lifted to a p'-sequence from M'.

3.3.3. Definition. For $M \in \Lambda'\boldsymbol{\pi}\mathbf{p}$ we define:

$$M \to_{p'} N \Leftrightarrow M \to_{p} N \ \&\ N \in \Lambda'\boldsymbol{\pi}\mathbf{p}.$$

Now the completion of the second and third rectangles of diagram 3.0.1 will be obtained in a series o: lemmas. The first one establishes what might be called the "½-substitutivity" of $\to_{p'}$.

3.3.4. Lemma. (i) Let $M_0 \to_{p'} M_1$ and $<>.N \in \Lambda'\boldsymbol{\pi}\mathbf{p}$. Then there exists a Q such that

$$\begin{tikzcd}[column sep=small]
(x:=N)M_0 \arrow[dr, dotted, "\overset{\equiv}{\to}_{p'}"'] & & (x:=N)M_1 \arrow[dl, dotted, "\hookrightarrow\!\!\!\!\to"] \\
& Q &
\end{tikzcd}$$

(ii) Let $<>.N_0 \to_{p'} N_1$, $M \in \Lambda'\boldsymbol{\pi}\mathbf{p}$. Then for some Q:

$$\begin{tikzcd}[column sep=small]
(x:=N_0)M \arrow[dr, dotted, "\twoheadrightarrow_{p'}"'] & & (x:=N_1)M \arrow[dl, dotted, "\hookrightarrow\!\!\!\!\to"] \\
& Q &
\end{tikzcd}$$

Proof. (i) Suppose that M_1 is obtained from $M_0 \equiv C[P]_\alpha$ by doubling P. If $(x:=N)C[\ ]_\alpha \equiv D[\ ]_\beta$ and $(x:=N)P \equiv R$, then $(x:=N)M_0 \equiv D[R]_\beta$ and $(x:=N)M_1 \equiv D[\lceil R, R\rceil_\alpha]_\beta$. Now there are (according to 2.2.4(i)) two possibil:ties for β:

- $\beta=\alpha$; then $D[R]_\beta \to_{p'} D[\lceil R, R\rceil_\alpha]_\beta$; take $Q \equiv (x:=N)M_1$
- $\beta>\alpha$; then $D[\lceil R, R\rceil_\alpha]_\beta \hookrightarrow D[R]_\beta$; take $Q \equiv (x:=N)M_0$.

(ii) Again, the occurrences of P ir. $(x:=N_0)M$ which stem from the occurrence of P in N_0 (label $\ell(P)=\alpha$) doubled in $N_0 \to_{p'} N_1$, receive in N_1 a label equal or greater than α (lemma 2.2.4(ii)). In all places where the new label is greater, Q has P; in the places where $\ell(P)$ is invariant under the substitution, Q has $\lceil P, P\rceil_\alpha$. □

3.3.5. Lemma.

(i)
$$\begin{tikzcd}
M \arrow[r, "\to_{\pi''}"] \arrow[d, "(\Delta)\to_{p'}"'] & N \arrow[d, dotted, "\twoheadrightarrow_{p'}"] \\
P \arrow[r, dotted, "\to_{\pi''}{}^{1/2}"'] & Q
\end{tikzcd}$$

(ii)
$$\begin{tikzcd}
{} \arrow[r, "\to_{\pi''}^{*}"] \arrow[d, "\to_{p'}^{*}"'] & {} \arrow[d, dotted, "\twoheadrightarrow_{p'}^{*}"] \\
{} \arrow[r, dotted, "\to_{\pi''}^{*}"'] & {}
\end{tikzcd}$$

Proof. (i) We proceed as in the proof of 3.2.3, this time by induction on the definition of $\to_{\pi''}$. Assume the step $M \to_{\pi''} N$ to be non-empty and choose Σ in the same way as in 3.2.3 as either the maximal bookkeeping pair to which clause b of definition 3.3.1 was applied or the π-redex in M which was contracted to obtain N. Distinguish cases as to the relative positions of Δ and Σ.

- $\Sigma \cap \Delta = \emptyset$. Since $\rightarrow_p$ is non-diminishing (cf. 2.6.2) and $\rightarrow_{\pi''}$ monotone, the step (Σ) can also be performed from P. Let the result be Q_1. Now, if $M \equiv C[\Delta]_\alpha$, then $Q_1 \equiv C_1[\ulcorner\Delta,\Delta\urcorner_\alpha]_\beta$ and $N \equiv C_1[\Delta]_\beta$, where $C_1[\ ]$ results from $C[\ ]$ by (Σ). Of course also $\rightarrow_{\pi''}$ is non-diminishing, so we can distinguish two subcases.

<u>a</u> $\beta=\alpha$; then $N \twoheadrightarrow_{p'} Q_1$; take $Q \equiv Q_1$

<u>b</u> $\beta>\alpha$; then $Q_1 \hookrightarrow N$ and consequently $P \rightarrow_{\pi''}^{1/2} N$; take $Q \equiv N$.

- $\Sigma \subsetneq \Delta$. If $M \equiv C[\Delta]_\alpha \ {}^{(\Sigma)}\!\!\rightarrow_{\pi''} C[\Delta_0]$, by the montonicity of $\rightarrow_{\pi''}$ the diagram

$$\begin{tikzcd}
C[\Delta]_\alpha \arrow[r, "\rightarrow_{\pi''}"] \arrow[d, "\rightarrow_{p'}"'] & C[\Delta_0] \arrow[d, dotted, "\rightarrow_{p'}"] \\
C[\ulcorner\Delta,\Delta\urcorner_\alpha] \arrow[r, dotted, "\rightarrow_{\pi''}"'] & C[\ulcorner\Delta_0,\Delta_0\urcorner_\alpha]
\end{tikzcd}$$

follows. That the p-step $C[\Delta_0]_\alpha \rightarrow_p C[\ulcorner\Delta_0,\Delta_0\urcorner_\alpha]$ is $\rightarrow_{p'}$, is an immediate consequence of the assumption that $C[\Delta] \rightarrow_p C[\ulcorner\Delta,\Delta\urcorner_\alpha]$ is so.

- $\Sigma \equiv \Delta$. Just use the monotonicity of $\rightarrow_{\pi''}$ again.

- $\Delta \subsetneq \Sigma$. Let $M \equiv C[\Sigma]_\alpha$. A further division in subcases must be made, according to the character of Σ. First we treat some cases where Σ is itself a $\rightarrow_{\pi'}$-redex.

<u>β_1</u> $\Sigma \equiv (\lambda' x.M_0)N$, $\Delta \subseteq M_0$ and $M_0 \ {}^{(\Delta)}\!\!\rightarrow_{p'} M_1$. Then according to lemma 3.3.4(i) a Q' can be found such that $(x:=N)M_0 \Rrightarrow_{p'} Q'$ and $(x:=N)M_1 \hookrightarrow\!\!\!\!\twoheadrightarrow_{p'} Q'$. The diagram is completed by the ½-compatibility of $\rightarrow_{p'}$.

$$\begin{tikzcd}
C[(\lambda' x.M_0)N] \arrow[rr, "\rightarrow_{\beta'}"] \arrow[d, "\rightarrow_{p'}"'] & & C[(x:=N)M_0] \arrow[dr, dotted, "\Rrightarrow_{p'}"] & \\
C[(\lambda' x.M_1)N] \arrow[r, dotted, "\rightarrow_{\beta'}"] & C[(x:=N)M_1] \arrow[r, dotted, "\hookrightarrow\!\!\!\!\twoheadrightarrow"] & C[Q'] \arrow[r, dotted, "\hookrightarrow\!\!\!\!\twoheadrightarrow"] & Q
\end{tikzcd}$$

<u>β_2</u> $\Sigma \equiv (\lambda' x.M_0)N_0$, $\Delta \subseteq N_0$. This case is analogous to the foregoing one, now using 3.3.4(ii).

<u>π_0</u> $\Sigma \equiv \pi_0'(\pi' X_0\, Y)$, $\Delta \subseteq X_0$. (NB: $\Delta \subseteq Y$ is not allowed, because Y is transient; $\Delta \equiv \pi' X_0\, Y$ is not allowed, because the doubling

would disturb a marked redex.) Now, since $\ell(\Delta)$ does not diminish under contraction of Σ, we get either diagram (i) or diagram (ii); (i) if $\ell(\Delta)$ is the same in $C[X_0]$ as it is in $C[\pi_0'(\pi' X_0\, Y)]$, (ii) if it is greater.

(i)

$$\begin{tikzcd}
C[\pi_0'(\pi' X_0 Y)] \arrow[r, "\pi_0'"] \arrow[d, "p'"'] & C[X_0] \arrow[d, dashed, "p'"] \\
C[\pi_0'(\pi' X_1 Y)] \arrow[r, dashed, "\pi_0'"] & C[X_1]
\end{tikzcd}$$

(ii)

$$\begin{tikzcd}
C[\pi_0'(\pi' X_0 Y)] \arrow[rr, "\pi_0'"] \arrow[d, "p'"'] & & C[X_0] \arrow[d, dashed, "\equiv"] \\
C[\pi_0'(\pi' X_1 Y)] \arrow[r, dashed, "\pi_0'"] & C[X_1] \arrow[r, dashed, hook] & C[X_0]
\end{tikzcd}$$

If Σ is a π_1-, l- or r-redex, the reasoning is the same. That leaves us with the possibility that Σ is not itself a π-redex, but a bookkeeping pair $\Sigma \equiv \ulcorner X_0, X_1 \urcorner_\beta$, with $\beta i.\, X_i \rightarrow_{\pi''} Y_i$ (for $i = 0,1$) and $N \equiv C[\ulcorner Y_0, Y_1 \urcorner_\beta]_\alpha$. Suppose without loss of generality that $\Delta \subseteq X_0$, (Δ): $X_0 \rightarrow_{p'} X_0'$. The induction hypothesis yields the following diagram.

$$\begin{tikzcd}
\beta 0.\, X_0 \arrow[r, "\pi''"] \arrow[d, "p'"'] & Y_0 \arrow[d, dashed, two heads, "p'"] \\
X_0' \arrow[r, dashed, "\pi''\frac{1}{2}"'] & Y_0'
\end{tikzcd}$$

Define $Q \equiv C[\ulcorner Y_0', Y_1 \urcorner_\beta]$. Then $N \twoheadrightarrow_{p'} Q$ because $\twoheadrightarrow_{p'}$ is quasi compatible. Moreover $P \rightarrow_{\pi''}{}^{\frac{1}{2}} Q$ follows from $X_0' \rightarrow_{\pi''}{}^{\frac{1}{2}} Y_0'$ and $X_1 \rightarrow_{\pi''} Y_1$ by lemma 3.3.2(ii).

(ii) The diagram

$$\begin{tikzcd}
\cdot \arrow[r, "\pi''"] \arrow[d, "p'"'] & \cdot \arrow[r, hook, "*"] \arrow[d, dashed, two heads, "p'"] & \cdot \arrow[dd, dashed, two heads, "p'^{*}"] \\
\cdot \arrow[r, dashed, "\pi''\frac{1}{2}"] \arrow[d, hook, "*"'] & \cdot \arrow[dr, dotted, hook, "*"'] & \\
\cdot \arrow[rr, dashed, "\pi''^{*}"'] & & \cdot
\end{tikzcd}$$

can be constructed using (i) and the $*$-projectability of both $\twoheadrightarrow_{p'}$ and $\twoheadrightarrow_{\pi''}{}^{\frac{1}{2}}$ (the latter follows by lemma 2.4.13(i) from 3.3.2(i)). □

3.3.6. Lemma. $\twoheadrightarrow_{\pi'}{}^{*}$ and $\twoheadrightarrow_{p'}{}^{*}$ commute.

Proof. Recall that by lemma 3.3.2(iii) we have the inclusions $\rightarrow_{\pi'}{}^{*} \subseteq \rightarrow_{\pi''}{}^{*}$ and $\rightarrow_{\pi''}{}^{*} \subseteq \twoheadrightarrow_{\pi'}{}^{*}$. Then the lemma can be concluded from 3.3.5(ii) by an application of 0.7.1. □

3.3.7. LEMMA. On $\Lambda'\boldsymbol{\pi}\mathbf{p}^*$ we have $\rightarrow_s \subseteq \twoheadrightarrow_{p'}{}^*$.

PROOF. The proof will make use of the lemmas 3.2.11 and 3.2.12. Let $\alpha. M \rightarrow_s N$ (i.e. $\alpha. M \twoheadrightarrow_p{}^* N$ with N p-simple). We proceed by induction on N. If there are no occurrences of bookkeeping pairs in N at all, then $N \equiv M$ (case a of 3.2.11) and $M \twoheadrightarrow_{p'}{}^* N$ holds trivially. So suppose that the situation is as described in 3.2.11b. Then on each of the reductions σ_i: $\alpha_i. Z_i \twoheadrightarrow_p{}^* \ulcorner X_i, Y_i \urcorner_{\alpha i}$ lemma 3.2.12(i) can be applied, yielding reductions $\sigma_i{}^0$: $\alpha_i 0. Z_i{}^0 \twoheadrightarrow_p{}^* X_i$ and $\sigma_i{}^1$: $\alpha_i 1. Z_i{}^1 \twoheadrightarrow_p{}^* Y_i$, with either $Z_i \equiv \ulcorner Z_i{}^0, Z_i{}^1 \urcorner_{\alpha i}$ or (Z_i): $Z_i \rightarrow_p{}^* \ulcorner Z_i{}^0, Z_i{}^1 \urcorner_{\alpha i}$. Since X_i and Y_i—both being subterms of the p-simple term N—are p-simple, by the induction hypothesis we may assume $\sigma_i{}^0$ and $\sigma_i{}^1$ to be in fact $\rightarrow_{p'}{}^*$-reduction sequences. Also, if Z_i is not itself a bookkeeping pair, the p*-reduction step (Z_i) is in fact $\rightarrow_{p'}{}^*$. For if it would break up a marked redex Σ, then the (external) residual of Σ in N would still be a broken redex, contradicting the p-simplicity of N. The steps (Z_i) and the sequences $\sigma_i{}^0$ and $\sigma_i{}^1$ can be combined to constitute a $\rightarrow_p{}^*$-reduction sequence from M to N, which is easily seen to be in fact a $\rightarrow_{p'}{}^*$-reduction sequence since in the above we already checked that the constituents were. □

3.3.8. LEMMA.

$$\begin{tikzcd}
M \arrow[r, "\rightarrow_1{}^*"] \arrow[d, "\rightarrow_s"'] & N \arrow[d, dotted, "\twoheadrightarrow_p{}^*"] \\
P \arrow[r, dotted, "\rightarrow_1{}^*"'] & Q
\end{tikzcd}$$

PROOF. If $M \rightarrow_1{}^* N$, an M_0 can be chosen such that $|M_0| \equiv M$ and $M_0 \twoheadrightarrow_{\pi'}{}^* N$ (cf. the proof of 3.1.11). Lift $M \rightarrow_s P$ to $M_0 \rightarrow_s P_0$ (with $|P_0| \equiv P$). By 3.3.7 then $M_0 \twoheadrightarrow_{p'}{}^* P_0$ holds too. So 3.3.6 can be applied to find a Q such that the diagram

$$\begin{tikzcd}
M_0 \arrow[r, "\twoheadrightarrow_{\pi'}{}^*"] \arrow[d, "\twoheadrightarrow_{p'}{}^*"'] & N \arrow[d, dotted, "\twoheadrightarrow_{p'}{}^*"] \\
P_0 \arrow[r, dotted, "\twoheadrightarrow_{\pi'}{}^*"'] & Q
\end{tikzcd}$$

holds. Since N has no marks ($\in \Lambda\boldsymbol{\pi}\mathbf{p}^*$), neither has Q. So $P_0 \twoheadrightarrow_{\pi'}{}^* Q$ is a complete development and hence $P \rightarrow_1{}^* Q$. □

3.4. Completing the diagram. Ey now we have gathered all the ingredients which are required to finish diagram 3.0.1. We proceed as it was already indicated in the first paragraphs of §3.

3.4.1. DEFINITION. The one step reduction relation $\rightarrow_+$ on $\mathbf{\Lambda\pi p^*}$ is defined by

$$\rightarrow_+ \; = \; \rightarrow_1{}^* + \rightarrow_s .$$

$=_+$ is the equivalence relation which is generated by $\rightarrow_+$.

3.4.2. LEMMA. (i) $\rightarrow_+ \subseteq \twoheadrightarrow^*$.
(ii) $=_+$ and $=^*$ coincide.

PROOF. (i) is trivial, since $\rightarrow_1{}^* \subseteq \twoheadrightarrow_\pi{}^*$ and $\rightarrow_s \subseteq \twoheadrightarrow_p{}^*$. One half of (ii), namely $=_+ \subseteq =^*$, is immediate by (i). Since $=^*$ is generated by $\rightarrow_p{}^*$ and $\rightarrow_\pi{}^*$, it suffices for the other half to establish $\underline{a}$ and $\underline{b}$ below.
$\underline{a}$. $\rightarrow_p{}^* \subseteq =_+$. For suppose $M \rightarrow_p{}^* N$. By 3.2.7 there exists a p-simple form N' of N, that is, $N \rightarrow_s N'$. Since $\rightarrow_p{}^* + \rightarrow_s = \rightarrow_s$, we have also $M \rightarrow_s N'$. Moreover $\rightarrow_s \subseteq \rightarrow_+$, because $\rightarrow_1{}^*$ is reflexive. Then $M =_+ N$ follows (via N').
$\underline{b}$. $\rightarrow_\pi{}^* \subseteq =_+$. If $M \rightarrow_\pi{}^* N$ then also $M \rightarrow_1{}^* N$, and $M =_+ N$ can be established, with the same reasoninz as in $\underline{a}$, via an N' such that $N \rightarrow_s N'$. □

3.4.3. THEOREM. The system $\mathbf{\lambda\pi p^*}$ (i.e. $(\mathbf{\Lambda\pi p^*}, \rightarrow^*)$) is Church-Rosser.

PROOF. By the lemmas 3.1.10(ii), 3.3.8 and 3.2.8, dealing respectively with the first, the second and third, and the fourth rectangles of diagram 3.0.1, it follows that $\rightarrow_+$ is self commuting and hence certainly Church-Rosser. So if $M =_+ N$, then M and N have a common $\rightarrow_+$-reduct. But, since $=^*$ is the same as $=_+$ (by 3.4.2(ii)), and each $\rightarrow_+$-sequence can be transformed into a $\rightarrow^*$-sequence (by 3.4.2(i)), we have then also the diagram

$$\begin{tikzcd}
{} \arrow[rr, dash, "=^*"] \arrow[dr, dotted, two heads, "\twoheadrightarrow^*"'] & & {} \arrow[dl, dotted, two heads, "\twoheadrightarrow^*"] \\
& {} &
\end{tikzcd}$$

This is the Curch-Rosser theorem for $\rightarrow^*$ in the second version (0.6.2). □

With theorem 3.4.3 we have established claim 1.9.1(iii) and thereby accomplished the first part of the program that was set out at the end of section 2.5.

§4. The Translation

Combining the results of the previous sections we know that if $\boldsymbol{\lambda\pi} \vdash M = N$ for expressions $M, N \in \Lambda\boldsymbol{\pi}$, then it follows by theorem 2.5.4 that also $\boldsymbol{\lambda\pi p^*} \vdash M =^* N$. Hence by the Church-Rosser theorem for $\boldsymbol{\lambda\pi p^*}$ (theorem 3.3.3) M and N have a common $\twoheadrightarrow^*$-reduct in $\Lambda\boldsymbol{\pi p^*}$. I.e. there exists a term $K \in \Lambda\boldsymbol{\pi p^*}$ such that $M \twoheadrightarrow^* K$ and $N \twoheadrightarrow^* K$.

In this concluding section we will show that if $\boldsymbol{\lambda\pi p^*} \vdash M \twoheadrightarrow^* K$ for the terms $M \in \Lambda\boldsymbol{\pi}$ and $K \in \Lambda\boldsymbol{\pi p^*}$, then $\boldsymbol{\lambda\pi} \vdash M \geq \phi(K)$ (cf. 2.1.3 for ϕ). This result applied twice in the above situation yields $\phi(K)$ as a common reduct of M and N in $\boldsymbol{\lambda\pi}$. (As a matter of fact the end term $\phi(K)$ is obtained only up to $\approx$. But this is sufficient for CR/$\approx$ as it was claimed in 1.2.3.)

4.1. Internal and external reduction. In order to translate reduction sequences of $\boldsymbol{\lambda\pi p^*}$ into $\boldsymbol{\lambda\pi}$, we first bring them in a special form, reached by postponement of the reduction steps which consist in the contraction of a redex that occurs within a bookkeeping pair. The resulting notion of e/i-reduction sequence (definition 4.1.2) has some resemblance with the concept of semi standardization, sometimes used in proofs of the standardization theorem in combinatory logic and pure λ-calculus (cf. Curry et al. [1972] and Mitschke [1979]). This method of proving standardization, originating with Rosser [1935], inspired our proceeding in this section.

4.1.1. DEFINITION. Recall that by definition 3.2.9 a redex occurrence Σ in a term $M \in \Lambda\boldsymbol{\pi p}$ lying completely inside a bookkeeping pair is called internal. Accordingly, a reduction step (Σ): $M \to N$ is called *internal* if the contracted redex Σ is internal; notation $M \to^{i} N$. Other reduction steps are called *external* $(\to^{e})$. Formally the relation $\to^{i}$ on $\Lambda\boldsymbol{\pi p}$ can be defined as the monotone and compatible closure of the reduction rule:

i: $\ulcorner M_0, M_1 \urcorner_\alpha \to^{i} N$, if $\ulcorner M_0, M_1 \urcorner_\alpha \to N$,

and $\to^{e}$ by $\to^{e} = \to \backslash \to^{i}$.

Derived notations such as $\to^{i*}$, $\twoheadrightarrow_{\pi}{}^{e*}$, $\to_{\pi'}{}^{i*}$, etc. are used in accordance with established conventions. Moreover we use the notation $\to^{(-e)}$ for $\to \backslash \to_{\pi}{}^{e}$. (Note that $\to^{(-e)}$ can be conceived of either as $\to_{p} \cup \to_{\pi}{}^{i}$ or as $\to^{i} \cup \to_{p}{}^{e}$ at will.)

A marked term $M \in \Lambda'\boldsymbol{\pi p}$ is called *internal* if all its marked redexes are internal. Obviously, if M is internal and $M \to_{\pi'}{}^{*} N$, then $M \to_{\pi'}{}^{i*} N$ and also N is internal. We say that $M \to_{1}{}^{i} N$, if N is the complete development of M with respect to a set of internal redexes.

EXAMPLES. - $(\lambda x.\, xy)\ulcorner\pi_0(\pi xy), x\urcorner \to^{e} \ulcorner\pi_0(\pi xy), x\urcorner y \to^{i} \ulcorner x, x\urcorner y$;
- $(\lambda x.\, xy)\ulcorner\pi_0(\pi xy), x\urcorner \to^{i*} (\lambda x.\, xy)\ulcorner x, x\urcorner \to^{e*} xy$;
- $(\lambda x.\, xy)\ulcorner\pi_0(\pi xy), x\urcorner \to^{e*} \pi_0(\pi xy)y \to^{e*} xy$;
- 0. $(\lambda x.x)x \to_{p}^{e} (\lambda x.x)\ulcorner x, x\urcorner \to_{\pi}^{e*} x$.

Observe that by changing the order of reduction, an internal step may become an external one (second and third example).

4.1.2. DEFINITION. A reduction sequence in $\boldsymbol{\lambda\pi p^*}$ is called *e/i*, if it consists of a number of $\to_{\pi}^{e*}$-reduction steps, followed by a number of $\to^{(-e)*}$-steps (i.e. $\to_{p}^{e*}$- and $\to^{i*}$-steps).

The rest of this section is devoted to proving that any $\to^*$-reduction sequence can be transformed into one that is e/i. It will become clear in 4.2 that e/i sequences in $\boldsymbol{\lambda\pi p^*}$ can be easily translated into $\boldsymbol{\lambda\pi}$. Then the main results of this paper, announced in section 1.2, follow at once.

4.1.3. LEMMA. $M \to_1^* N \Rightarrow (\exists L)(M \twoheadrightarrow_{\pi}^{e*} L \to_1^{i*} N)$.

PROOF. Let $M_0 \in \Lambda'\boldsymbol{\pi p^*}$ such that $|M_0| \equiv M$ and $\mathrm{CD}^*(M_0) \equiv N$. We use induction on $\mathrm{h}^*(M_0)$. If M_0 is internal, then $M \to_1^{i*} N$ and $L \equiv M$ will do. Otherwise suppose $M_0 \to_{\pi'}^{e*} M_1$. By the induction hypothesis there exists an L such that $|M_1| \twoheadrightarrow_{\pi}^{e*} L \to_{\pi}^{i*} N$. Since of course $M \to_{\pi}^{e*} |M_1|$ holds too, this L suffices. □

4.1.4. LEMMA.

(i)

$$\begin{tikzcd} & M \arrow[dl, "\to_1^{i*}"'] \arrow[dr, dotted, "\twoheadrightarrow_{\pi}^{e*}"] & \\ K \arrow[dr, "(\Sigma)\to_{\pi}^{e*}"'] & & L \arrow[dl, dotted, "\to_1^{i*}"] \\ & N & \end{tikzcd}$$

(ii)

$$\begin{tikzcd} & {} \arrow[dl, "\to_1^{i*}"'] \arrow[dr, dotted, "\twoheadrightarrow_{\pi}^{e*}"] & \\ {} \arrow[dr, "\twoheadrightarrow_{\pi}^{e*}"'] & & {} \arrow[dl, dotted, "\to_1^{i*}"] \\ & {} & \end{tikzcd}$$

PROOF. (i) As one easily sees, Σ is the residual of a unique external redex occurrence Σ_1 in M. (The situation may be compared to that of 3.2.10.) Let $M_0 \in \Lambda'\boldsymbol{\pi p^*}$ such that $|M_0| \equiv M$ and $\mathrm{CD}^*(M_0) \equiv K$. Then by FD! we obtain $\mathrm{CD}^*(M_0 + \langle M, \Sigma_1\rangle) \equiv N$. Therefore $M \to_1^* N$. Apply lemma 4.1.3.

(ii) Repeat (i). □

4.1.5. LEMMA

(i)
$$\begin{tikzcd}
 & M \arrow[dl, "(\Delta)\to_p^*"'] \arrow[dr, dotted, "\to_\pi^{e*}"] & \\
K \arrow[dr, "(\Sigma)\to_\pi^{e*}"'] & & L \arrow[dl, dotted, "\twoheadrightarrow_p^*"] \\
 & N &
\end{tikzcd}$$

(ii)
$$\begin{tikzcd}
 & {} \arrow[dl, "\twoheadrightarrow_p^*"'] \arrow[dr, dotted, "\twoheadrightarrow_\pi^{e*}"] & \\
{} \arrow[dr, "\twoheadrightarrow_\pi^{e*}"'] & & {} \arrow[dl, dotted, "\twoheadrightarrow_p^*"] \\
 & {} &
\end{tikzcd}$$

PROOF. (i) This time lemma 3.2.10 exactly applies. Again let Σ_1 be the external redex occurrence in M from which Σ descends and suppose (Σ_1): $M \to_\pi^{e*} L$. Assume furthermore that $\ell(\Delta) = \alpha$ in M, that is, in the p*-step $M \to_p^* K$ the occurence Δ is replaced by $\ulcorner\Delta^*, \Delta^*\urcorner_\alpha$. Now, as Σ is external (and according to 3.2.10), either $\Delta \cap \Sigma_1 = \emptyset$ or $\Delta \subseteq \Sigma_1$. In both cases the descendants of Δ in L have—due to the fact that $\to_\pi^{e*}$ is non-diminishing—a label which extends to α. In those positions (in L) where $\ell(\Delta)$ increased in passing from M to L, the same happens with $\ell(\ulcorner\Delta^*, \Delta^*\urcorner_\alpha)$ under (Σ), thus causing the bookkeeping pair to be cancelled in N. So N can be obtained from L by doubling the descendants of Δ that in L still have label α.

(ii) Straightforward inductions from (i). □

4.1.6. THEOREM. If σ: $M \twoheadrightarrow^* N$, then there exists an e/i-reduction sequence from M to N.

PROOF. Induction on the number of $\to^*$-steps in σ. If $M \equiv N$ there is not much to prove. So assume an M_0 which satisfies the drawn part of the following diagram $(\sigma = (\Sigma) + \sigma_1)$.

$$\begin{tikzcd}
M \arrow[r, "(\Sigma)\to^*"] \arrow[d, dotted, "\twoheadrightarrow_\pi^{e*}"'] \arrow[dr, phantom, "B"] & M_0 \arrow[r, "\sigma_1: \twoheadrightarrow^*"] \arrow[d, dotted, "\twoheadrightarrow_\pi^{e*}"] \arrow[r, phantom, shift right=4, "A"'] & N \arrow[dl, dotted, "\twoheadrightarrow(-e)*"] \\
{} \arrow[r, dotted, "\twoheadrightarrow(-e)*"'] & {} &
\end{tikzcd}$$

Then the triangle A can be found by the induction hypothesis. If Σ is an external redex that suffices. Otherwise B can be constructed using either lemma 4.1.4 or 4.1.5, dependent on the character of (Σ) (either $\to_\pi^{i*}$ or $\to_p^*$). □

4.2. $\boldsymbol{\lambda\pi}$ is Church-Rosser

4.2.1 Lemma. Suppose $M \in \Lambda\boldsymbol{\pi}$ and $\mathbf{\lambda\pi p}^* \vdash \alpha. M \twoheadrightarrow_\pi^* N$. Then $\boldsymbol{\lambda\pi}^{\geq} \vdash M \geq K$ for some $K \in \Lambda\boldsymbol{\pi}$ such that $\alpha. K \sim N$.

Proof. Let $\mathbf{\lambda^\circ\pi p}^*$ be the system with as terms the set $\Lambda^\circ\mathbf{\pi p}^*$ and as one step reduction relation $\rightarrow^*$ considered as a relation on $\Lambda^\circ\mathbf{\pi p}^*$ (the rules of 2.5.1 restricted to non-transient redex occurrences). There obviously exists a $K \in \Lambda^\circ\mathbf{\pi p}^*$ such that $\alpha. K \sim N$ and $\mathbf{\lambda^\circ\pi p}^* \vdash \alpha. M \twoheadrightarrow_\pi^* K$. As $\rightarrow_\pi$ does not create bookkeeping pairs, there will be nothing to cancel for $\hookrightarrow^*$ during this reduction, and we have $\alpha. M \twoheadrightarrow_\pi K$ and $K \in \Lambda\boldsymbol{\pi}$. It then suffices to remind that each of the rules for $\rightarrow_\pi$ (cf. 2.5.1(i)) is covered by one for $\geq$ (cf. 1.2.1). In particular both $\mathbf{l}$ and $\mathbf{r}$ are included in $\boldsymbol{\lambda\pi}^{\geq}$ without context restrictions. □

4.2.2. Theorem. Suppose $M \in \Lambda\boldsymbol{\pi}$ and $\mathbf{\lambda\pi p}^* \vdash \alpha. M \twoheadrightarrow^* N$. Then $\boldsymbol{\lambda\pi} \vdash M \geq K$ for some $K \in \Lambda\boldsymbol{\pi}$ such that $K \approx \phi(N)$.

Proof. The proof is by induction on N (considered as an element of $\Lambda\mathbf{\pi p}^*$). By theorem 4.1.6 there is a L such that $\mathbf{\lambda\pi p}^* \vdash \alpha. M \twoheadrightarrow_\pi^{e*} L$ and $\mathbf{\lambda\pi p}^* \vdash \alpha. L \twoheadrightarrow^{(-e)*} N$. Consequently lemma 4.2.1 can be used to supply us with a $K_0 \in \Lambda\boldsymbol{\pi}$ such that $\alpha. K_0 \sim L$ and $\boldsymbol{\lambda\pi} \vdash M \geq K_0$. Now, if N is already an element of $\Lambda\boldsymbol{\pi}$ itself, then the (-e)-part of the e/i sequence is empty and we have $\phi(N) \equiv N \equiv L$. So in this case K can be taken just K_0, as on $\Lambda\boldsymbol{\pi}$ one has of course $\sim \subsetneq \approx$. Otherwise N contains one or more bookkeeping pairs and can thus be assumed to be written as

$$\ldots\ulcorner X_1, Y_1 \urcorner_{\alpha_1} \ldots \ulcorner X_2, Y_2 \urcorner_{\alpha_2} \ldots\ldots \ulcorner X_n, Y_n \urcorner_{\alpha n} \ldots,$$

where each maximal occurrence of a bookkeeping pair is displayed as one of the $\ulcorner X_i, Y_i \urcorner_{\alpha i}$'s. Since $\alpha. K_0 \sim L$ the second part of the e/i-sequence from M to N may be rendered as well as $\mathbf{\lambda\pi p}^* \vdash \alpha. K_0 \twoheadrightarrow^{(-e)*} N$. From the fact that this reduction proceeds completely without external reduction steps one easily infers (analogous to 2.7.11) that K_0 must have a shape similar to that of N, that is

$$K_0 \equiv \ldots Z_1 \ldots Z_2 \ldots\ldots Z_n \ldots,$$

coinciding with N on the dots, and such that for each of the i's:

$$\ell(Z_i) = \alpha_i \text{ and } \alpha_i. Z_i \twoheadrightarrow^{(-e)*} \ulcorner X_i, Y_i \urcorner_{\alpha i}.$$

A subterm Z_i of $K_0 \in \Lambda\boldsymbol{\pi}$ being bookkeeping pairless, the lemma on main p-reduction (3.2.12(ii)) can bə applied, yielding for each i with $1 \leq i \leq n$ a reduction $\alpha_i 0. Z_i \twoheadrightarrow^* X_i$. On these we can use the induction hypothesis, and thereby obtain reductions $Z_i \geq X_i'$ in $\boldsymbol{\lambda\pi}$, with $X_i' \approx \phi(X_i)$.

It remains to combine the reductions we established so far. By the compatibility of $\geq$ it follows that

$$\boldsymbol{\lambda\pi} \vdash K_0 \geq \ldots X_1' \ldots X_2' \ldots\ldots X_n' \ldots .$$

Define $K \equiv \ldots X_1' \ldots X_2' \ldots\ldots X_n' \ldots$. Then $K \approx \phi(N)$ is an immediate consequence of the above by the the definition of ϕ (cf. 2.1.3) and the compatibility of $\approx$. Moreover, $M \geq K$ via K_0. So K satisfies the requirements of the theorem. □

At last everything is set to prove the Church-Rosser theorem (modulo $\approx$) for $\geq$, already announced in 1.2.3. With $\psi(M) \equiv M$, the proof of theorem 4.2.3 is the one already given for 1.9.2.

4.2.3. THEOREM. In $\boldsymbol{\lambda\pi}$ the following diagram holds.

$$\begin{tikzcd}
M \arrow[r, dash, "="] \arrow[d, dash, dotted, "\geq"'] & N \arrow[d, dash, dotted, "\geq"] \\
K_0 \arrow[r, dash, dotted, "\approx"'] & K_1
\end{tikzcd}$$

PROOF. By theorem 2.5.4 we know that if $\boldsymbol{\lambda\pi} \vdash M = N$, then also $\boldsymbol{\lambda\pi\mathrm{p}}^* \vdash \langle\rangle . M =^* N$. The Church-Rosser theorem for $\boldsymbol{\lambda\pi\mathrm{p}}^*$ (theorem 3.4.3) then implies the existence of a $K \in \Lambda\boldsymbol{\pi\mathrm{p}}^*$ such that in $\boldsymbol{\lambda\pi\mathrm{p}}^*$ both $\langle\rangle . M \twoheadrightarrow^* K$ and $\langle\rangle . N \twoheadrightarrow^* K$ hold. Then by applying theorem 4.2.2 to these reductions, terms K_0 and $K_1 \in \Lambda\boldsymbol{\pi}$ are found such that $K_0 \approx \phi(K) \approx K_1$. □

Now the main results of this paper, the conservativity and consistency of $\boldsymbol{\lambda\pi}$, follow in the way which was indicated in 1.2.

4.2.4. COROLLARY. (i) $\boldsymbol{\lambda\pi}$ is a conservative extension of $\boldsymbol{\lambda}$.
(ii) $\boldsymbol{\lambda\pi}$ is consistent.

PROOF. The proof which was given in section 1.2, showing how to derive 1.2.4 from CR/$\approx$ (claim 1.2.3), can be employed. □

APPENDIX 1: The undefinability of surjective pairing

A.1.1. It was proved by Barendregt [1974] that in $\boldsymbol{\lambda}$ a surjective pairing is not definable. I.e. in Λ one can not find closed expressions P, P_0 and P_1 such that for arbitrary $M, N \in \Lambda$ the equations $P_0(PMN) = M$, $P_1(PMN) = N$ and $P(P_0N)(P_1N) = M$ are derivable in $\boldsymbol{\lambda}$. (Throughout this section by stating an equation or a reduction ϕ we mean $\boldsymbol{\lambda} \vdash \phi$.) Here we give a short proof of this fact, using theorem A.1.2. In fact we prove the non-existence under a slightly weaker condition: the equations are only required to hold for closed M and N.

A.1.2. THEOREM ("Barendregt's lemma"). Let $FM \geq N$. Then there exists an expression N' such that $Fx \geq N'$ and N can be obtained by replacing in N' certain (disjoint) subexpressions $N_1, \ldots, N_p$ $(p \geq 0)$ —respectively of the form $N_i \equiv xN_1{}^i \ldots N_{j_i}{}^i$ $(j_i \geq 0)$ —by respective reducts of $(x:=M)N_i$ $(1 \leq i \leq p)$. I.e. each N_i is replaced in N' by an expression M_i such that $(x:=M)N_i \geq M_i)$.

This is not a very well-known, though quite useful theorem. It was stated in a weaker form in an unpublished manuscript [1972] of Barendregt, namely for weak combinatory logic and with the extra condition that N is a normal form. A proof for the λ-calculus and for arbitrary N was given in de Boer [1975]. In his thesis [1980], van Daalen provides the formal background for a more precise formulation of the theorem, thus making possible a short and elegant proof. It is included as exercise 15.4.8 in Barendregt [1981].

A.1.3. Let $\Omega \in \Lambda$ be defined by $\Omega \equiv (\lambda x.xx)(\lambda x.xx)$. Then it is an easy exercise to verify the following noteworthy "freezing" property of Ω.

If $M_1, \ldots, M_p, Z \in \Lambda$ and $\Omega M_1 \ldots M_p \geq Z$, then $Z \equiv \Omega M_1{}' \ldots M_p{}'$ for certain $M_i{}' \in \Lambda$ such that $M_i \geq M_i{}'$ $(p \geq 0,\ 1 \leq i \leq p)$. That is, Ω has order 0 (cf. Curry-Feys [1958], chapter 4F).

A.1.4. THEOREM. In Λ there do not exist closed expressions P, P_0 and P_1 which satisfy for arbitrary closed $M, N \in \Lambda$ the equations $P_0(PMN) = M$, $P_1(PMN) = N$ and $P(P_0N)(P_1N) = M$.

PROOF. Assume there were such P, P_0 and P_1. Define $F \equiv \lambda x.P(P_0\Omega)(P_1x)$. Then $F\Omega = P(P_0\Omega)(P_1\Omega) = \Omega$ and hence, by the Church-Rosser theorem for $\boldsymbol{\lambda}$, the terms $F\Omega$ and Ω have a common $\geq$-reduct. By A.1.3 this can only be Ω itself. So $F\Omega \geq \Omega$, and we can apply theorem A.1.2 to yield an N' with the there ascribed properties (taking $M \equiv N \equiv \Omega$). Then again by A.1.3 one easily verifies that there are but two possibilities for N', namely either $N' \equiv \Omega$ or $N' \equiv x$. We investigate both cases.

CASE 1: $N' \equiv \Omega$. Then $Fx \geq \Omega$ and so $Fx = \Omega$ and by substitutivity of conversion $FX = \Omega$ for any $X \in \Lambda$. So in particular for any closed $X \in \Lambda$ we have $P_1 X = P_1(P(P_0\Omega)(P_1 X) = P_1(FX) = P_1\Omega$. Now let $X_0 \equiv P(\lambda x.x)(\lambda x.x)$ and $X_1 \equiv P(\lambda xy.x)(\lambda xy.x)$. Then $P_1(X_0) = P_1(X_1) = P_1(\Omega)$, and by projection we arrive at the conclusion that $\lambda x.x = \lambda xy.x$, contradicting Church-Rosser.

CASE 2: $N' \equiv x$. Then $Fx \geq x$ and so $Fx = x$ and $FX = X$ for any $X \in \Lambda$. So for any closed $X \in \Lambda$ we obtain the string of equations $P_0 X = P_0(FX) = P_0(P(P_0\Omega)(P_1 X)) = P_0\Omega$. Now proceed further as in case 1 to arrive at a contradiction. □

APPENDIX 2. The consistency of $\boldsymbol{\lambda\pi}$ by a graph model

We now briefly indicate how to prove the consistency of $\boldsymbol{\lambda\pi}$, using a construction conceived by Scott [1975a] (p. 178). In the discussion familiarity is assumed with the *graph model* for $\boldsymbol{\lambda}$. For details we refer to Scott [1975a] and Barendregt [1981].

A.2.1. THE GRAPH MODEL.

Let ω be the set of natural numbers $\{0,1,2,\ldots\}$ and let $\mathrm{P}\omega$ be the powerset of ω. Let $\mathrm{e}_0, \mathrm{e}_1, \ldots$ be some standard enumeration of the finite subsets of ω, such that $\mathrm{e}_0 = \emptyset$. Finally assume a bijective pairing function from $\omega \times \omega$ onto ω. In ω the pair of n and m is denoted by (n,m). Then the application of two elements A and B of $\mathrm{P}\omega$ is defined as follows:

$$A(B) = \{m \mid (\exists \mathrm{e}_n \subseteq B)((n,m) \in A)\}.$$

It turns out that using this definition of application an interpretation of the closed terms of Λ into $\mathrm{P}\omega$ can be defined, such that equality in $\boldsymbol{\lambda}$ is respected. (In fact, in $\mathrm{P}\omega$ much more terms are equated than in $\boldsymbol{\lambda}$.) Here we do not further describe this construction, but now immediately turn to the introduction of a pairing in $\mathrm{P}\omega$.

A.2.2. PAIRING IN $\mathrm{P}\omega$.

For A, B in $\mathrm{P}\omega$ we define the pair $[A,B]$ as follows:

$$[A,B] = \{2m \mid m \in A\} \cup \{2n+1 \mid n \in B\}.$$

Notice that this is a rather obvious procedure: coding two sets A and B into one set C, by storing up A into the even members of C and B into the odd ones. Furthermore, clearly any set C decomposes uniquely into two components according to this procedure. Viz.

$$(C)_0 = \{n \mid 2n \in C\} \text{ and}$$
$$(C)_1 = \{n \mid 2n+1 \in C\}.$$

Then one easily verifies the equations

$$([A,B])_0 = A, \quad ([A,B])_1 = B \text{ and } [(A)_0,(A)_1] = A.$$

The interesting point is now, that these pairing and projection operations are themselves definable as elements of Pω (in agreement with the above definition of application).

A.2.3. DEFINITION. We define P, P_0 and $P_1 \in \mathrm{P}\omega$ as follows:

$$P = \{(k,(0,2m)) \mid m \in e_k\} \cup \{(0,(\ell,2n+1)) \mid n \in e_\ell\};$$
$$P_0 = \{(m,n) \mid 2n \in e_m\};$$
$$P_1 = \{(m,n) \mid 2n+1 \in e_m\}.$$

One easily verifies by the definition of application in Pω that for any $A, B \in \mathrm{P}\omega$,

$$P(A)(B) = [A,B], \quad P_0(C) = (C)_0 \text{ and } P_1(C) = (C)_1$$

and hence by the equations in A.2.2:

$$P_0(P(A)(B)) = A, \quad P_1(P(A)(B)) = B \text{ and}$$
$$P(P_0(A))(P_1(A)) = A.$$

Using these facts a model for **λπ**, extending the standard interpretation of **λ** in Pω, can be constructed, by interpreting π_0 as P_0, π_1 as P_1 and π as P. Then the consistency of **λπ** follows.

References

Barendregt, H.P. [1972], *The undefinability of Church's δ* (Unpublished).

Barendregt, H.P. [1974], 'Pairing without conventional restraints', *Zeitschr. für Math. Logik und Grundl. der Math.* 20, p.289-306.

Barendregt, H.P. [1981], *The Lambda Calculus* (North Holland).

Barendregt, H.P., J. Bergstra, J.W. Klop and H. Volken [1976], *Degrees, Reductions and Representability in the λ-Calculus*, Preprint nr. 22, Department of Mathematics, Utrecht University.

Bergstra, J.A. and J.W. Klop [1986], 'Conditional Rewrite Rules: Confluence and Termination', *Journal of Computer and System Sciences* 32, p.323-362.

Boer, S. de [1975], *De ondefinieerbaarheid van Church's δ-funktie in de λ-calculus en Barendregt's lemma*, stageverslag, Technische Hogeschool Eindhoven.

Böhm, C. (ed.) [1975], 'λ-Calculus and Computer Science Theory'. *Springer Lecture Notes in Computer Science* 37.

Curry, H.B. and R. Feys [1958], *Combinatory Logic Vol. I* (North Holland).

Curry, H.B., J.R. Hindley and J.P. Seldin [1972], *Combinatory Logic Vol. II* (North Holland).

Daalen, D.T. van [1980], *The language theory of Automath*, dissertation, Technische Hogeschool Eindhoven.

Klop, J.W. [1980], 'Combinatory reduction systems', *Mathematical Center Tracts* 129, Amsterdam.

Mann, C.R. [1973], *Connections Between Proof Theory and Category Theory*, dissertation, Oxford University.

Mitschke, G. [1979], 'The standardization theorem for the λ-calculus'. *Zeitschr. für Math. Logik und Grundl. der Math.* 25, p.29-31.

Rosser, J.B., [1935], 'A mathematical logic without variables I', *Annals of Mathematics* 36, p.127-150.

Scott, D. [1975a], 'Lambda Calculus and Recursion Theory'. *Proc. of t the third Scandinavian Logic Symposium*, ed. S. Kanger (North Holland), p.154-193.

Scott, D. [1975b], 'Some Philosophical Issues Concerning Theories of Combinators', in: *λ-Calculus and Computer Science Theory*, ed. C. Böhm, Springer Lect.Notes in Computer Science 37, p.346-366.

Scott, D. [1980], 'Lambda Calculus: Some Models, Some Philosophy', in: J. Barwise et al. (ed.), *The Kleene Symposium* (North Holland).

Staples, J. [1975], 'Church-Rosser Theorems For Replacement Systems', in: *Algebra and Logic*, Springer Lecture Notes in Mathematics 450, p.291-307.

Vrijer, R.C. de [1985], 'A Direct Proof of the Finite Developments Theorem', *Journal of Symbolic Logic* 50, p.339-343.

THE JOURNAL OF SYMBOLIC LOGIC
Volume 50, Number 2, June 1985

2

A DIRECT PROOF OF THE FINITE DEVELOPMENTS THEOREM

§0. Introduction. Let M be a term of the type free λ-calculus and let $\mathscr{R}$ be a set of occurrences of redexes in M. A reduction sequence from M which first contracts a member of $\mathscr{R}$ and afterwards only residuals of $\mathscr{R}$ is called a *development* (of M with respect to $\mathscr{R}$). The finite developments tneorem says that developments are always finite.

There are several proofs of this theorem in the literature. A plausible strategy is to define some kind of measure for pairs $(M,\mathscr{R})$, which—if M' results from M by contracting a redex occurrence in $\mathscr{R}$ and $\mathscr{R}'$ is the set of residuals of $\mathscr{R}$ in M'—decreases in passing from $(M,\mathscr{R})$ to $(M',\mathscr{R}')$. This procedure is followed as a matter of fact in the proofs in Hyland [4] and in Barendregt [1] (both are covered in Klop [5]). If, as in the latter proof, the natural numbers are used as measures, then the measure of $(M,\mathscr{R})$ will actually denote an upper bound of the number of reduction steps in a development of M with respect to $\mathscr{R}$.

In the present proof we straightforwardly define for each pair $(M,\mathscr{R})$ a natural number, which can easily be seen to indicate the exact number of reduction steps in a development of maximal length of M with respect to $\mathscr{R}$.

Following Barendregt [1] we represent the pair $(M,\mathscr{R})$ by priming the initial λ of each redex occurrence of M that is in $\mathscr{R}$. Then after any number of reduction steps, the residuals of $\mathscr{R}$ are exactly the marked redex occurrences (λ'-redexes) in the reduced term. Hence a development of M with respect to $\mathscr{R}$ is a reduction sequence in which only λ'-redexes are contracted. This approach leads to a short and accurate formulation of FD: it boils down to strong normalization for λ'-reduction.

§1. λ'-reduction. We restrict our attention here to the pure $\lambda\beta$-calculus.

1.1. The set Λ of pure λ-terms is defined inductively from a set of variables x, y, etc., by the clauses:

(i) x is a variable $\Rightarrow x \in \Lambda$,

(ii) $M, N \in \Lambda \Rightarrow (MN) \in \Lambda$,

(iii) $M \in \Lambda$, x is a variable $\Rightarrow (\lambda x.M) \in \Lambda$.

The set Λ' of λ'-terms is an extension of Λ, which is obtained by adding to (i)–(iii) (with Λ' for Λ):

(iv) $M, N \in \Lambda'$, x is a variable $\Rightarrow ((\lambda' x.M)N) \in \Lambda'$.

Parentheses will be omitted in accordance with common usage. Terms are considered modulo α-equivalence (i.e. change of bound variables).

By $(x := M)N$ we denote the result of substituting M for x in N. We shall always assume the bound variables to be chosen in such a way that no free variable of M becomes bound after the substitution.

$\mathrm{FV}(M)$ denotes the set of free variables of M.

1.2. Note on subterms. Not every subterm of a term in Λ' is itself in Λ'. The implications $\lambda x.M \in \Lambda' \Rightarrow M \in \Lambda'$, $MN \in \Lambda' \Rightarrow N \in \Lambda'$ and $(\lambda' x.M)N \in \Lambda' \Rightarrow M \in \Lambda'$ hold as one might expect; but $MN \in \Lambda' \Rightarrow M \in \Lambda'$ does not. The one exception to this implication ıs the case that M is of the form $\lambda' x.P$.

1.3. Λ' is provided with a restricted reduction relation $\rightarrow'$, which allows only λ'-redexes, i.e. subterm occurrences of the form $(\lambda' x.P)Q$, to be contracted. So we define:

$$M \rightarrow' N \Leftrightarrow N \text{ results from } M \text{ by replacing a part of the form } (\lambda' x.P)Q \text{ by its contraction } (x := Q)P.$$

A λ'-reduction sĕquence ($\rightarrow'$-sequence) from M is a finite or infinite sequence of terms $M_0, M_1, \ldots$ such that $M_0 = M$ and $M_i \rightarrow' M_{i+1}$.

The following lemma indicates that Λ' is closed under substitution and under λ'-reduction and that by substitution one cannot "create" new λ'-redexes.

1.4. Lemma. (i) $M, N \in \Lambda' \Rightarrow (x := N)M \in \Lambda'$.

(ii) $M \in \Lambda'$, $M \rightarrow' N \Rightarrow N \in \Lambda'$.

(iii) $M, N \in \Lambda'$, $(x := N)\,M$ *is a* λ'*-redex* $\Rightarrow$ *either* $M = x$ *or* M *is already itself a* λ'*-redex.*

Proof. The proofs of (i) and (ii) are by induction on M; (i) is used in the proof of (ii).

For (iii) assume that $(x := N)M = (\lambda' y.P_0)P_1$. It is clearly sufficient to show that if $M = M_0 M_1$, then $M_0 = \lambda' y.Q$ for some Q. Suppose not. Then, according to 1.2, $M_0 \in \Lambda'$ and $(x := N)M_0 \notin \Lambda'$, contradicting (i).

§2. The proof of FD. In this section we prove that $\rightarrow'$ is strongly normalizable, i.e. that for each $M \in \Lambda'$ the length of its $\rightarrow'$-sequences is bounded.

First we define for each variable x the multiplicity function $\mathrm{m}_x\colon \Lambda' \rightarrow \omega$, and subsequently the function $\mathrm{h}\colon \Lambda' \rightarrow \omega$, depending on the multiplicities. Then $\mathrm{h}(M)$ is shown to be the length of the longest $\rightarrow'$-sequence from M (i.e. the number of reduction steps).

2.1. Definition. For each $M \in \Lambda'$ and variable x the *multiplicity* of x in M, $\mathrm{m}_x(M)$, is defined by induction on M. (In the definition the bound variables of M are assumed to be chosen different from x.)

$$\begin{aligned}
\mathrm{m}_x(x) &= 1,\\
\mathrm{m}_x(y) &= 0 \quad \text{if } y \neq x,\\
\mathrm{m}_x((\lambda' y.M)N) &= \mathrm{m}_x(M) + \mathrm{m}_x(N)\,.\max(\mathrm{m}_y(M), 1),\\
\mathrm{m}_x(MN) &= \mathrm{m}_x(M) + \mathrm{m}_x(N) \quad \text{if } MN \text{ is not a } \lambda'\text{-redex},\\
\mathrm{m}_x(\lambda y.M) &= \mathrm{m}_x(M).
\end{aligned}$$

2.2. Definition. For $M \in \Lambda'$ the *height* $\mathrm{h}(M)$ is defined inductively:

$$\begin{aligned} \mathrm{h}(x) &= 0, \\ \mathrm{h}((\lambda' x\,.M)N) &= \mathrm{h}(M) + \mathrm{h}(N)\,.\max(\mathrm{m}_x(M), 1) + 1, \\ \mathrm{h}(MN) &= \mathrm{h}(M) + \mathrm{h}(N) \quad \text{if } MN \text{ is not a } \lambda'\text{-redex}, \\ \mathrm{h}(\lambda x\,.M) &= \mathrm{h}(M). \end{aligned}$$

2.3. Lemma. $\mathrm{m}_x(M) > 0 \Leftrightarrow x \in \mathrm{FV}(M)$.

Proof. Trivial induction on M.

2.4. Lemma. *If* $x \neq y$, *then* $\mathrm{m}_y((x := N)M) = \mathrm{m}_y(M) + \mathrm{m}_y(N)\,.\mathrm{m}_x(M)$.

Proof. (Induction on M). Let M be a variable z. One easily verifies that the left-hand and the right-hand side both equal respectively $\mathrm{m}_y(N)$ if $z = x$, 1 if $z = y$ and 0 if $z \neq x, y$.

Let $M = (\lambda' z\,.P)Q$ (z chosen such that $z \neq x, y$ and $z \notin \mathrm{FV}(N)$). Using the induction hypothesis for P (twice!) and Q and the fact that $\mathrm{m}_z(N) = 0$, one simply calculates

$$\begin{aligned} &\mathrm{m}_y((x := N)M) = \mathrm{m}_y((\lambda' z.(x := N)P)(x := N)Q) \\ &= \mathrm{m}_y((x := N)P) + \mathrm{m}_y((x := N)Q)\,.\max(\mathrm{m}_z((x := N)P), 1) \\ &= \mathrm{m}_y(P) + \mathrm{m}_y(N)\,.\mathrm{m}_x(P) + (\mathrm{m}_y(Q) + \mathrm{m}_y(N)\,.\mathrm{m}_x(Q))\,.\max(\mathrm{m}_z(P), 1) \\ &= \mathrm{m}_y(M) + \mathrm{m}_y(N)\,.\mathrm{m}_x(M). \end{aligned}$$

Let $M = PQ$, not a λ'-redex. Then by 1.4(iii) neither is $(x := N)M$ a λ'-redex. So

$$\mathrm{m}_y((x := N)M) = \mathrm{m}_y((x := N)P) + \mathrm{m}_y((x := N)Q),$$

from which, using the induction hypothesis for P and Q, one simply calculates the result.

The remaining case, $M = \lambda z\,.P$, we leave to the reader.

2.5. Lemma. $\mathrm{h}((x := N)M) = \mathrm{h}(M) + \mathrm{h}(N)\,.\mathrm{m}_x(M)$.

Proof. Again induction on M and some calculation.

2.6. Lemma. $M \to' N \Rightarrow \mathrm{m}_x(N) \leq \mathrm{m}_x(M)$.

Proof (Induction on M). Distinguish cases as to the form of the reduction step. Lemma 2.4 is used in the case that $M = (\lambda' y\,.P)Q$ and $N = (y := Q)P$. If, with the same M, $P \to' P_0$ and $N = (\lambda' y\,.P_0)Q$, the induction hypothesis yields both $\mathrm{m}_x(P_0) \leq \mathrm{m}_x(P)$ and $\mathrm{m}_y(P_0) \leq \mathrm{m}_y(P)$, by which the result follows. Other cases are as simple.

2.7. Lemma. $M \to' N \Rightarrow \mathrm{h}(N) < \mathrm{h}(M)$.

Proof (Induction on M, again). The crucial case is $M = (\lambda' x\,.P)Q$ and $N = (x := Q)P$. Then by 2.5, $\mathrm{h}(N) = \mathrm{h}(P) + \mathrm{h}(Q)\,.\mathrm{m}_x(P) \leq \mathrm{h}(M) - 1$. The previous lemma is used for $M = (\lambda' x\,.P)Q$, $P \to' P_0$, $N = (\lambda' x\,.P_0)Q$. Then $\mathrm{m}_x(P_0) \leq \mathrm{m}_x(P)$ and hence, by the induction hypothesis for P_0,

$$\mathrm{h}(N) = \mathrm{h}(P_0) + \mathrm{h}(Q)\,.\max(\mathrm{m}_x(P_0), 1) + 1 < \mathrm{h}(M).$$

2.8. Theorem. *No* $\to'$*-sequence from M has more than* $\mathrm{h}(M)$ *steps.*

Proof. Induction on $\mathrm{h}(M)$, using 2.7.

2.9. Corollary (FD). *All developments are finite.*

Now, in order to show that $\mathrm{h}(M)$ is an exact bound, we produce for each M with $\mathrm{h}(M) > 0$ a term M^* such that $M \to' M^*$ and $\mathrm{h}(M) = \mathrm{h}(M^*) + 1$. The definition of M^* is reminiscent of the so-called perpetual reduction strategy in [1].

2.10. Definition. For each $M \in \Lambda'$ with $\mathrm{h}(M) > 0$ we define M^*, by induction on M, as follows. (A clause for variables is not needed, since always $\mathrm{h}(x) = 0$.)

$$\begin{aligned} ((\lambda' x.P)Q)^* &= (\lambda' x.P)Q^* && \text{if } \mathrm{m}_x(P) = 0 \text{ and } \mathrm{h}(Q) > 0,\\ &= (x := Q)P && \text{otherwise.}\end{aligned}$$

If PQ is not a λ'-redex, then

$$\begin{aligned} (PQ)^* &= P^*Q && \text{if } \mathrm{h}(P) > 0,\\ &= PQ^* && \text{otherwise.}\end{aligned}$$

(NB: $\mathrm{h}(PQ) > 0 \Rightarrow \mathrm{h}(P) > 0$ or $\mathrm{h}(Q) > 0$ or both.) Finally,

$$(\lambda x.P)^* = \lambda x.P^*.$$

(NB: $\mathrm{h}(\lambda x.P) > 0 \Rightarrow \mathrm{h}(P) > 0$.)

2.11. Lemma. *If* $\mathrm{h}(M) > 0$ *then* $M \to' M^*$ *and* $\mathrm{h}(M) = \mathrm{h}(M^*) + 1$.

Proof. Easy induction on M.

2.12. Theorem. $\mathrm{h}(M)$ *is exactly the number of reduction steps in a* $\to'$*-sequence from* M *of maximal length.*

Proof. Combine 2.8 and 2.11.

§3. Connection with Hindley's proof of FD. Our proof was inspired by a proof of strong normalization for the theory of abbreviations LSP, due to N. G. de Bruijn (presented in van Daalen [2]). There is also a proof of FD by R. Hindley [3], to which the present one bears a casual connection. Hindley defines by induction on $M \in \Lambda$ a number $\mathrm{p}(M)$, which is simultaneously proved to bound the length of M's developments (with respect to the set of all redex occurrences in M). The definition of p runs parallel to our definition of h, except for the clause for $(\lambda x.M)N$, where an extra term $\mathrm{m}_x(M).\mathrm{p}(M).\mathrm{p}(N)$ is added to the sum (our terminology). Accordingly, the reasoning is along different lines and more involved than ours.

§4. Combinatory weak reduction. In [3] FD is generalized to systems incorporating a rather broad class of operators with defined reductions. Though it is not mentioned there, for these operators the definition of p does yield an exact bound. As such it can be added to our definition of h.

By way of example we consider the special case of combinatory logic with weak reduction (CL). Its symbols are the variables and the constants $\mathbf{S}$ and $\mathbf{K}$; the only term-forming operation is application. In CL$'$ the constants $\mathbf{S}$ and $\mathbf{K}$ can be primed when occurring as the "head" of a redex; that is to say, in the combinations $\mathbf{S}MNP$ and $\mathbf{K}MN$ respectively. Then $M \to' N$ holds if and only if N results from M by replacing either a part of the form $\mathbf{S}'PQR$ by $PR(QR)$ or a part $\mathbf{K}'PQ$ by P.

Define the function h by:

$$\mathrm{h}(M) = 0 \quad \text{if } M \text{ is a variable or a constant;}$$

$$\begin{aligned}\mathrm{h}(MN) &= \mathrm{h}(M)+\mathrm{h}(N)+1 && \text{if } M=\mathbf{K}'P,\\ &\quad \mathrm{h}(M)+2.\mathrm{h}(N)+1 && \text{if } M=\mathbf{S}'PQ,\\ &\quad \mathrm{h}(M)+\mathrm{h}(N) && \text{otherwise.}\end{aligned}$$

It is not difficult to show that h(M) is an exact bound on the length of the $\rightarrow'$-sequences from M.

REFERENCES

[1] H. P. Barendregt, ***The lambda calculus***, North-Holland, Amsterdam, 1981.

[2] D. T. van Daalen, ***The language theory of Automath***, Dissertation, Technological University of Eindhoven, Eindhoven, 1980.

[3] R. Hindley, *Reductions of residuals are finite*, ***Transactions of the American Mathematical Society***, vol. 240 (1978), pp. 345–361.

[4] M. Hyland, *A simple proof of the Church-Rosser theorem*, Typescript, Oxford University, Oxford, 1973.

[5] J. W. Klop, ***Combinatory reduction systems***, Dissertation, Mathematical Centre Tracts 127, Mathematisch Centrum, Amsterdam, 1980.

3 EXACTLY ESTIMATING FUNCTIONALS AND STRONG NORMALIZATION

§0. INTRODUCTION

It is our aim in this paper to prove strong normalization for the typed λ-calculus (λ^τ) by giving directly an expression which for each term t determines the number h(t), indicating the exact height of its reduction tree (i.e. the number of reduction steps in a reduction sequence from t of maximal length). Such an expression is also obtained in our proof of the finite developments theorem for the pure λ-calculus [A]. There it is easier to define, however, in virtue of the fact that in a complete development all contracted redexes are residuals of redexes already present in the original term. In contrast, in a λ^τ-reduction sequence redexes may be contracted which are created at an earlier stage of that sequence.

0.1. To obtain the expression for the height we proceed roughly as follows. By induction on its length, to each term t of type $(\alpha)\beta$ we attach a pair <f,m> (denoted by [t]), consisting of a functional f and a number m. The number m will turn out to be the height: m = h(t); the functional f will be such that it yields the pair attached to ts, when it is applied to a term s of type α: f[s] = [ts]. To terms of the ground type o just the number indicating their height is attached, since they cannot be applied.

The pairs we speak of here form a collection, much resembling the well-known functionals of finite type. We call them hereditarily labeled functionals of finite type or for short labeled functionals.

0.2. Because of the additional task of providing the exact estimates, the proof of strong normalization which results is more complicated than some of the existing proofs, in particular those using a computability argument (see e.g. Troelstra ['73]) and Van Daalen's proof in ['80] (taken over in Barendregt ['81]). In exchange some extra transparency seems to result: the expression for the height follows directly from the idea explained in 0.1 and an easily understandable reduction strategy for producing maximal reduction sequences.

Moreover the proof may be considerably simplified by no longer requesting exactness. This is the 'quick' proof, which provides a looser upper bound for the height. It is reminiscent of the use of finite type functionals in Gandy ['80].[1)]

0.3. OUTLINE. The hierarchy of labeled functionals L is introduced in section 1. There only some elementary definitions are given. More of the theory of the labeled functionals then follows in sections 3 and 4, when it is needed in the proofs.

In section 2 we first give a short sketch of the typed λ-calculus. Then we define and assess two valuations of terms into L. The first, the exact valuation, yields the expression $[t]*$, which is claimed to be exactly $h(t)$. As a consequence of this claim a uniformity result concerning $h(t)$ is stated.

The second valuation, the so-called loose one, is used in section 3 for the quick proof of strong normalization. There we show that the valuations belong to a subhierarchy of L, the collection of hereditarily <-monotone labeled functionals M. It is to be compared to the hereditarily strictly monotonic functionals used by Gandy.

In section 4 a more restricted subhierarchy C is used in the proof of the claim $h(t) = [t]*$.

Finally in section 5 we make some observations on the extension of the method to other systems.

0.4. ACKNOWLEDGEMENTS. I would like to thank D.H.J. de Jongh and P.H. Rodenburg for their helpful suggestions for improving the text.

§1. LABELED FUNCTIONALS.

Types are as usual built up from the ground type o by finite iterations of a single inductive clause: if α and β are types, then (α)β is also a type. Then by induction on the type α, we define the collection of labeled functionals of type α, L_α.

$L_o = \omega$,

$L_{(\beta)\gamma} = (L_\beta \to L_\gamma) \times \omega$.

(Here ω denotes the set of natural numbers, × cartesian product and → function space.)

1.1. NOTATION. We will use the same kind of metavariables (f, g, etc.) for members of L_α (pairs generally) and of $L_\beta \to L_\gamma$ (functionals). . Let $f = \langle f', m\rangle \in L_{(\beta)\gamma}$, $g \in L_\beta$. We write $f* = m$ and - par abus de langage - $fg = f'g$. For reasons of uniformity the $*$-notation is extended to type o: $n* = n$ $(n \in L_o)$. As a rule we suppress type subscripts whenever it is possible without giving rise to confusion. So $f \in L$ means that $f \in L_\alpha$ for some type α. Of course L may be thought of more precisely as defined by $L = {}^{\cup}_{\alpha}L_\alpha$. The same convention we shall adopt tacitly for all type-labeled predicates and operations yet to be defined. Also, in writing down an expression like fg, we assume the types to fit, that is: for certain types α and β, $f \in L_{(\alpha)\beta}$ and $g \in L_\alpha$.

1.2. THE +-OPERATION. Let $n \in \omega$, for $f \in L_\alpha$ we define $f +_\alpha n \in L_\alpha$ by induction on α.

$$m +_o n = m + n$$

$$f +_{(\beta)\gamma} n = \langle \Lambda g \in L_\beta . fg +_\gamma n,\ f* + n\rangle.$$

Here the symbol Λ is used for functional abstraction in the metalanguage. So the definition of addition is recapitulated in the equations $(f+n)g = fg+n$ and $(f+n)* = f*+n$.

Notice that $+ = {}^{\cup}_{\alpha}+_\alpha$ extends standard addition on ω. By induction over the types it is easily checked that

(i) $f+0 = f$ and (ii) $(f+m)+n = f+(m+n)$.

1.3. MINIMALLY CUMULATIVE FUNCTIONALS. By induction on α we define for every $n \in \omega$ the *minimally cumulative functional* $c^\alpha_n \in L_\alpha$.

$$c^o_n = n$$

$$c^{(\beta)\gamma}_n = \langle \Lambda f \in L_\beta . c_{n+f*}, n\rangle$$

These functionals are cumulative in the sense that (type subscripts omitted): (i) $c_n f_1 \ldots f_m = c_{n+f_1*+\ldots+f_m*}$ and hence in particular (ii) $(c_0 f_1 \ldots f_m)* = f_1*+\ldots+f_m*$. Below, in section 4, a wider class of so-called cumulative functionals will be defined and discussed.

For the moment we conclude with the characteristic equation: (iii) $c^\alpha_n + m = c^\alpha_{n+m}$. It is readily verified by induction on α. For

$c_n^o+m = n+m = c_{n+m}^o$ and $(c_n^{(\beta)\gamma}+m)* = c_n^{(\beta)\gamma}*+m = n+m = c_{n+m}^{(\beta)\gamma}*$ follow immediately from the definition. And if $f \in L_\beta$, then $(c_n^{(\beta)\gamma}+m)f = c_n^{(\beta)\gamma}f+m = c_{n+f*}^{\gamma}+m$, which by the induction hypothesis equals $c_{n+f*+m}^{\gamma} = c_{n+m}^{(\beta)\gamma}f$.

§2. THE SYSTEM λ^τ OF TYPED λ-CALCULUS AND THE VALUATIONS.

2.1. TERMS. λ^τ-terms are built up from typed variables x^α, y^α, etc. (for each type α) by the inductive clauses: (i) x^α is a term of type α, (ii) if t and s are terms of type $(\alpha)\beta$ and α respectively, then (ts) is a term of type β, (iii) if t is a term of type β, then $(\lambda x^\alpha.t)$ is a term of type $(\alpha)\beta$. We write $t \in \alpha$ for t has type α.

We presuppose some acquaintance with λ-calculus conventions, e.g. concerning parentheses. The type superscripts of the variables will be suppressed where possible (viz. whenever the type is either clear from the context or not essential). Terms are regarded modulo α-equivalence (i.e. change of bound variables).

$(x:=t)s$ denotes the result of substituting t for x in s; here the bound variables in s are tacitly assumed to be chosen in such a way, that no free variable of t becomes bound after the substitution. The notation $x \in t$ is used to express that x has at least one free occurrence in t.

2.2. REDUCTION. We restrict attention to β-reduction. A term t *reduces* to s (notation $t \to s$), if s is the result of replacing a part of t of the form $(\lambda x.t_0)t_1$ (a *redex*) by $(x:=t_1)t_0$. If t does not contain any redex, it is said to be a *normal form*.

A *reduction sequence* of t is a, finite or infinite, sequence of terms $t_0,t_1,\dots$, such that $t_0 = t$ and $t_i \to t_{i+1}$. A term t is called *weakly normalizable* if at least one reduction sequence of t terminates in a normal form; t is called *strongly normalizable* if all of t's reduction sequences are finite. In the latter case, by König's lemma, the number of reduction steps in a reduction sequence of t is bounded; the maximum is denoted by $h(t)$ (the *height* of the reduction tree).

2.3. THE EXACT VALUATION. Now in order to obtain the expression for the height, terms are evaluated in L starting from an *assignment* v, which gives a value $\mathrm{v}(x^{\alpha})$ in L_{α} to each variable x^{α}. As is customary we write $\mathrm{v}(x/\mathrm{f})$ for the assignment which corresponds with v everywhere except at x: $\mathrm{v}(x/\mathrm{f})(x) = \mathrm{f}$, $\mathrm{v}(x/\mathrm{f})(y) = \mathrm{v}(y)$ if $y \neq x$.

2.3.1. DEFINITION. Let t be a term of type α. The *exact valuation* $[t]_{\mathrm{v}} \in L$ is defined for any assignment v, by induction on t.

(i) $[x]_{\mathrm{v}} = \mathrm{v}(x)$

(ii) $[t_0 t_1]_{\mathrm{v}} = [t_0]_{\mathrm{v}}[t_1]_{\mathrm{v}}$

(iii) $[\lambda x^{\alpha}.t_0]_{\mathrm{v}} = \langle \Lambda \mathrm{f} \in L_{\alpha}.[t_0]_{\mathrm{v}} + \mathrm{f}^{\ast} + 1, [t_0]_{\mathrm{v}}{}^{\ast} \rangle$ if $x \notin t_0$, and

$\langle \Lambda \mathrm{f} \in L_{\alpha}.[t_0]_{\mathrm{v}(x/\mathrm{f})} + 1, [t_0]_{\mathrm{v}(x/c_0^{\alpha})}{}^{\ast} \rangle$ if $x \in t_0$.

Notice that if $x \notin t$, then $[t]_{\mathrm{v}} = [t]_{\mathrm{v}(x/\mathrm{f})}$, as can easily be verified by induction on t. Let c be the assignment defined by $\mathrm{c}(x^{\alpha}) = c_0^{\alpha}$ and put $[t] = [t]_{\mathrm{c}}$.

It may be instructive to calculate the following examples:

$[\lambda x^{o}.x] = [\lambda x^{o}.y^{o}] = \langle \Lambda \mathrm{m}.\mathrm{m}+1, 0 \rangle$,

$[\lambda x^{(o)o} y^{o}.x(xy)] = \langle \Lambda \mathrm{f}.\langle \Lambda \mathrm{m}.\mathrm{f}(\mathrm{fm})+2, \mathrm{f}(\mathrm{f}0)+1 \rangle, 0 \rangle$,

$[(\lambda x^{(o)o} y^{o}.x(xy))\lambda z^{o}.z] = \langle \Lambda \mathrm{m}.\mathrm{m}+4, 3 \rangle$.

2.3.2. CLAIM. For any term t, $\mathrm{h}(t) = [t]^{\ast}$.
This will be proved in section 4. Here we first comment on the definition of $[t]$ and then call attention to some consequences of 2.3.2.

2.3.3. COMMENTS ON 2.3.1. By the account given in the introduction (0.1), clause (ii) is sufficiently motivated.
ad i. Observe that if $t_1, \ldots, t_m$ are strongly normalizable (of the appropriate types), then so is $xt_1 \ldots t_m$ and moreover the height is given by the equation $\mathrm{h}(xt_1 \ldots t_m) = \mathrm{h}(t_1) + \ldots \mathrm{h}(t_m)$. This squares with the fact that $(c_0 \mathrm{f}_1 \ldots \mathrm{f}_m)^{\ast} = \mathrm{f}_1{}^{\ast} + \ldots \mathrm{f}_m{}^{\ast}$ (1.3 (iii)).
ad iii. To get a grasp of this clause the reader should try to

invent a strategy for constructing a reduction sequence from t which is as long as possible. Clearly if $x \notin t_0$, then in order to spoil no potential reduction steps a redex $(\lambda x.t_0)t_1$ should not be contracted until t_1 is in normal form. On the other hand, if $x \in t_0$, it is better not to perform reductions inside t_1 before contracting $(\lambda x.t_0)t_1$, for t_1 might get multiplied in t_0. As a matter of fact it turns out that the strategy we have in mind here corresponds to the 'perpetual' reduction strategy for the pure λ-calculus, described in Barendregt ['81]. This strategy will be implicit in the proof of 4.9.

2.3.4. CONSEQUENCES OF 2.3.2. Apart from establishing strong normalization, the expression for h(t) in 2.3.2 allows us to infer some extra information. Roughly it can be put as follows: the height of a term depends uniformly on the valuations of its constituent parts.

A typical consequence, immediate by using definition 2.3.1(ii), is that for a fixed term $t \in (\alpha)\beta$, the height of ts, for $s \in \alpha$, depends uniformly on $[s]$. In particular, given a fixed $t \in (o)\beta$, h(ts) is determined by h(s) alone.

We now give an accurate and quite general formulation of this uniformity result (from which the abovementioned consequence is obtained by substituting tx for t).

2.3.5. RESULT. Let $[s_1] = [s_2]$. Then $[(x:=s_1)t] = [(x:=s_2)t]$ and hence $\mathrm{h}((x:=s_1)t) = \mathrm{h}((x:=s_2)t)$.

PROOF. It is an immediate corollary to the following technical lemma. □

2.3.6. SUBSTITUTION LEMMA. $[(x:=s)t]_{\mathrm{v}} = [t]_{\mathrm{v}(x/[s]_{\mathrm{v}})}$.

PROOF. Induction on t. The only case which is not quite trivial is $t = \lambda y^{\alpha}.t_0$. We may assume that $y \neq x$ and $y \notin s$.

Suppose $y \in t_0$. Then $[(x:=s)t]_{\mathrm{v}} = [\lambda y.(x:=s)t_0]_{\mathrm{v}} =$ $\langle \Lambda \mathrm{f} \in \mathrm{L}_{\alpha}.[(x:=s)t_0]_{\mathrm{v}(y/\mathrm{f})} + 1, [(x:=s)t_0]_{\mathrm{v}(y/\mathrm{c}_0)}{}^{*}\rangle$ which by the induction hypothesis and because $[s]_{\mathrm{v}(y/\mathrm{f})} = [s]_{\mathrm{v}}$ is equal to $\langle \Lambda \mathrm{f} \in \mathrm{L}_{\alpha}.[t_0]_{\mathrm{v}(y/\mathrm{f})(x/[s]_{\mathrm{v}})} + 1, [t_0]_{\mathrm{v}(y/\mathrm{c}_0)(x/[s]_{\mathrm{v}})}{}^{*}\rangle =$

$= [\lambda y.t_0]_{v(x/[s]_v)}$. For the case that $y \notin t_0$ notice that then also $y \notin (x:=s)t_0$. □

2.4. THE LOOSE VALUATION. If one is just interested in the strong normalization result and not in the exact estimates, then one may simplify the meticulus clause (iii) of definition 2.3.1.

2.4.1. DEFINITION. Let t be a term of type α. The *loose valuation* $\{t\}_v \in L_\alpha$ is defined for any assignment v, by induction on t.

(i) $\{x\}_v = v(x)$

(ii) $\{t_0 t_1\}_v = \{t_0\}_v \{t_1\}_v$

(iii) $\{\lambda x^\alpha.t_0\}_v = \langle \Lambda f \in L_\alpha.\{t_0\}_{v(x/f)} + f* + 1, \{t_0\}_{v(x/c_0^\alpha)}*\rangle$

Once more $\{t\}$ is defined by $\{t\} = \{t\}_c$. Clearly this valuation is only intended to yield an upper bound for $h(t)$. So our new claim is less pretentious.

2.4.2. CLAIM. For any term t, $h(t) \leqq \{t\}*$.
This will be the object of the quick proof of strong normalization in section 3. In the proof the content of lemma 2.4.3 will be used.

2.4.3. LEMMA. (i) If $x \notin t$, then $\{t\}_v = \{t\}_{v(x/f)}$

(ii) $\{(x:=s)t\}_v = \{t\}_{v(x/\{s\}_v)}$

PROOF. As in 2.3.6. □

§3. THE QUICK PROOF OF STRONG NORMALIZATION.

3.1. HEREDITARILY <-MONOTONE FUNCTIONALS. By induction on α we define simultaneously the classes $M_\alpha \subset L_\alpha$ and the relations $<_\alpha$ on M_α.

(i) $M_0 = L_0$; $m <_0 n \Leftrightarrow m < n$.

(ii) $M_{(\alpha)\beta} = \{f \in L_{(\alpha)\beta} \mid$ conditions (a) and (b) below are fulfilled$\}$.

(a) $(\forall g \in M_\alpha)(fg \in M_\beta)$.

(b) $(\forall g,g' \in M_\alpha)(g <_\alpha g' \Rightarrow fg <_\beta fg')$ (f *<-monotone)*

(iii) For $f,g \in M_{(\alpha)\beta}$, $f <_{(\alpha)\beta} g \Leftrightarrow (\forall h \in M_\alpha)(fh <_\beta gh) \,\&\, f* < g*$.

$M = \bigcup_\alpha M_\alpha$ is the class of hereditarily $<$-monotone functionals.

Evidently, by clause (iia), M is closed under application. That $<$ is transitive is verified by induction on the types. By the following lemma, M is also closed under +.

3.2. LEMMA. (i) $f \in M_\alpha \Rightarrow f + m \in M_\alpha$, (ii) $f <_\alpha g \Rightarrow f + m <_\alpha g + m$, (iii) $m < n \Rightarrow f + m < f + n$, (iv) $m > 0 \Rightarrow f < f + m$.

PROOF. (i) and (ii) are simultaneously proved, by induction on α. For $\alpha = o$ it is obvious. So let $\alpha = (\beta)\gamma$ and suppose $f \in M_\alpha$. For (i) we must check clauses (iia) and (iib) of 3.1. Well, if $h, h' \in M_\beta$ and $h <_\beta h'$, then $(f + m)h = fh + m \in M_\beta$ by induction hypothesis (i), and $(f + m)h = fh + m < fh' + m = (f + m)h'$ by induction hypothesis (ii). So $f + m \in M_\alpha$ (i).

If moreover $f <_\alpha g$, then $(f + m)h < (g + m)h$ follows by induction hypothesis (i) and (ii), and $(f + m)* < (g + m)*$ by mere calculation. So $f + m <_\alpha g + m$ (ii).

(iii) is also easy to check by induction on the type of f, and (iv) follows from (iii) because $f = f + 0$. □

3.3. LEMMA. (i) $c_n^\alpha \in M$ for any α and n.
(ii) $m < n \Rightarrow c_m^\alpha < c_m^\alpha$.

PROOF. (i) and (ii) are proved by simultaneous induction on α. For $\alpha = o$ it is trivial. So let $\alpha = (\beta)\gamma$.

If h, $h' \in M_\beta$, then $c_n^\alpha h = c_{n+h*}^\gamma \in M_\gamma$ by induction hypothesis (i). If moreover $h <_\beta h'$, then $h* < h'*$ and hence $c_n^\alpha h = c_{n+h*}^\gamma < c_{n+h'*}^\gamma = c_n^\alpha h'$ by induction hypothesis (ii). This proves (i).

For (ii) suppose $m < n$ and let $h \in M_\beta$. Then $c_m^\alpha h < c_n^\alpha h$ follows because $m + h* < n + h*$ and hence $c_{m+h*}^\gamma < c_{n+h*}^\gamma$ (induction hypothesis (ii)), and $c_m^\alpha * < c_n^\alpha *$ is immediate $(m < n)$. This proves (ii). □

If the assignment v takes only values in M, v is called an M-assignment. In particular c is an M-assignment by 3.3(i).

3.4. MONOTONICITY LEMMA. (i) $\{t\}_v \in M$, for any M-assignment v. (ii) If $x^\alpha \in t$, then $\{t\}_{v(x/f_1)} < \{t\}_{v(x/f_2)}$, for any M-assignment v and $f_1, f_2 \in M_\alpha$ such that $f_1 <_\alpha f_2$.

PROOF. (i) and (ii) are proved by simultaneous induction on t.

- If t is a variable then (i) and (ii) clearly hold.

- Let $t = t_0 t_1$ and assume (i) and (ii) to hold for t_0 and t_1. Then $\{t\}_v \in M$ because M is closed under application. To establish (ii) observe that if $x \in t$, then x must occur in t_0 or in t_1 or in both. Distinguish cases accordingly.

Case 1. $x \in t_0$, $x \notin t_1$. Then $\{t\}_{v(x/f_i)} = \{t_0\}_{v(x/f_i)} \{t_1\}_v$, and (ii) follows from $\{t_0\}_{v(x/f_1)} < \{t_0\}_{v(x/f_2)}$.

Case 2. $x \notin t_0$, $x \in t_1$. Similar. Now (ii) follows because $\{t_0\}_v$ is <-monotone and $\{t_1\}_{v(x/f_1)} < \{t_1\}_{v(x/f_2)}$.

Case 3. $x \in t_0$, $x \in t_1$.
Apply induction hypothesis (ii) to t_0 and t_1 in turn and use the transitivity of <.

- Let $t = \lambda y^\beta . t_0$. (i) If $h \in M_\beta$, then $v(y/h)$ is an M-assignment; so $\{t_0\}_{v(y/h)} \in M$ by induction hypothesis (i), and hence $\{t\}_v h = \{t_0\}_{v(y/h)} + h* + 1 \in M$ because M is closed under +. If moreover $h < h'$, then $\{t\}_v h < \{t\}_v h'$ by applications of induction hypothesis (ii) and lemma 3.2(ii) and (iii). So $\{t\}_v \in M$.

To establish (ii), suppose $x^\alpha \in t$, $f_1 <_\alpha f_2$ and let $h \in M_\beta$. We have to check that (a) $\{t\}_{v(x/f_1)} h < \{t\}_{v(x/f_2)} h$ and (b) $\{t\}_{v(x/f_1)}* < \{t\}_{v(x/f_2)}*$. Notice that if $x \in t$, then also $x \in t_0$. Then (a) follows by an application of induction hypothesis (ii) to t_0 with $v(y/h)$ and lemma 3.2(ii), and (b) by an application of induction hypothesis (ii) to t_0 with $v(y/c_0)$ (by lemma 3.3(i) this is an M-assignment). □

3.5. REDUCTION LEMMA. If $t \to s$, then $\{s\}_{\mathrm{v}} < \{t\}_{\mathrm{v}}$ for any M-assignment v.

PROOF. Induction on t.

- If t is a variable there is nothing to prove.
- Let $t = (\lambda x . t_0) t_1$, $s = (x := t_1) t_0$. Then by the substitution lemma (2.4.3(ii)) $\{s\}_{\mathrm{v}} = \{t_0\}_{\mathrm{v}(x/\{t_1\}_{\mathrm{v}})}$ and by 3.2(iv) $\{t_0\}_{\mathrm{v}(x/\{t_1\}_{\mathrm{v}})} < \{t_0\}_{\mathrm{v}(x/\{t_1\}_{\mathrm{v}})} + \{t_1\}_{\mathrm{v}}{*} + 1$. But this is $\{t\}_{\mathrm{v}}$.
- Let $t = t_0 t_1$, $t_0 \to s_0$ and $s = s_0 t_1$. Then $\{s\}_{\mathrm{v}} < \{t\}_{\mathrm{v}}$ follows directly from $\{s_0\}_{\mathrm{v}} < \{t_0\}_{\mathrm{v}}$.
- Let $t = t_0 t_1$, $t_1 \to s_1$ and $s = t_0 s_1$. Then $\{s\}_{\mathrm{v}} < \{t\}_{\mathrm{v}}$ follows from $\{s_1\}_{\mathrm{v}} < \{t_1\}_{\mathrm{v}}$ because $\{t_0\}_{\mathrm{v}} \in \mathrm{M}$.
- Let $t = \lambda x^{\alpha} . t_0$, $t_0 \to s_0$ and $s = \lambda x . s_0$. For $\mathrm{f} \in \mathrm{M}_{\alpha}$, $\{s_0\}_{\mathrm{v}(x/\mathrm{f})} + \mathrm{f}{*} + 1 < \{t_0\}_{\mathrm{v}(x/\mathrm{f})} + \mathrm{f}{*} + 1$ follows from $\{s_0\}_{\mathrm{v}(x/\mathrm{f})} < \{t_0\}_{\mathrm{v}(x/\mathrm{f})}$ by lemma 3.2(ii); $\{s_0\}_{\mathrm{v}(x/\mathrm{c}_0)}{*} < \{t_0\}_{\mathrm{v}(x/\mathrm{c}_0)}$ is immediate from $\{s_0\}_{\mathrm{v}(x/\mathrm{c}_0)} < \{t_0\}_{\mathrm{v}(x/\mathrm{c}_0)}$.

So $\{s\}_{\mathrm{v}}\mathrm{f} < \{t\}_{\mathrm{v}}\mathrm{f}$ and $\{s\}_{\mathrm{v}}{*} < \{t\}_{\mathrm{v}}{*}$ and hence $\{s\}_{\mathrm{v}} < \{t\}_{\mathrm{v}}$.

The mentioned cases exhaust all possibilities for $t \to s$. □

3.6. THEOREM. No reduction sequence of t has more than $\{t\}{*}$ steps.

PROOF. Trivial induction on $\{t\}{*}$, using 3.5. □

3.7. COROLLARY. All terms t of λ^{τ} are strongly normalizable and $\mathrm{h}(t) \leqq \{t\}{*}$.

This finishes the quick proof of strong normalization.

§4. THE MAIN PROOF

4.1. HEREDITARILY CUMULATIVE, MONOTONE FUNCTIONALS.

To establish the claim 2.3.2 we shall make use of the fact that the functionals occurring as valuations of λ^{τ}-terms belong to an even more restricted class than M.

DEFINITION. The collections C_α are defined simultaneously with the relations $<_\alpha$ and $\leqq_\alpha$ by induction on α.

(i) $C_o = L_o$; $m <_o n \Leftrightarrow m < n$; $m \leqq_o n \Leftrightarrow m \leqq n$.

(ii) $C_{(\alpha)\beta} = \{f \in L_{(\alpha)\beta} \mid$ conditions (a), (b), (c) and (d) below are fulfilled$\}$

(a) $(\forall g \in C_\alpha \; fg \in C_\beta)$

(b) $(\forall g,g' \in C_\alpha)(g <_\alpha g' \Rightarrow fg <_\beta fg')$ (f <-*monotone*)

(c) $(\forall g,g' \in C_\alpha)(g \leqq_\alpha g' \Rightarrow fg \leqq_\beta fg')$ (f $\leqq$-*monotone*)

(d) $(\forall g \in C_\alpha)((fg)_* \geqq f_* + g_*)$ (f *cumulative*)

(iii) For $f,g \in C_{(\alpha)\beta}$,

(a) $f <_{(\alpha)\beta} g \Leftrightarrow (\forall h \in C_\alpha)(fh <_\beta gh) \;\&\; f_* < g_*$

(b) $f \leqq_{(\alpha)\beta} g \Leftrightarrow (\forall h \in C_\alpha)(fh \leqq_\beta gh) \;\&\; f_* \leqq g_*$

REMARKS. (1) $C = \bigcup_\alpha C_\alpha$ is, just like M, closed under application (condition (ii)(a)). $<_\alpha$ and $\leqq_\alpha$ are transitive relations on C_α.

(2) Notice that a new interpretation is given here to the sign $<_\alpha$, though in intention it is the same as that of the preceding section.

(3) One might be tempted to think that $f \leqq g$ could be defined simply as $f < g$ *or* $f = g$. This is of course the case on the ground type, but not on the higher types. For a counterexample in (o)o consider the functions $\Lambda n.n$ and $\Lambda n.n^2$. (Note that both $\Lambda n.n$ and $\Lambda n.n^2$ are minimal with respect to $<$.)

(4) <-monotonicity and $\leqq$-monotonicity are independent predicates, neither implies the other. For counterexamples: constant functions are $\leqq$-monotone but not <-monotone; a functional of type ((o)o)(o)o which is $\leqq$-monotone but not <-monotone can be defined by $f(\Lambda n.n^2) = \Lambda n.n^2$, $fg = \Lambda n.gn + 1$ if $g \neq \Lambda n.n^2$.

(5) We will often write $f \geqq g$ for $g \leqq f$, and $f > g$ for $g < f$.

(6) Many properties of the structure (M,<) hold also in (C,<) and can be extended to $\leqq$ as well, as is shown in the following lemma.

In the sequel we shall in general no longer explicitly indicate applications of this lemma, as the reader is assumed to have gained some experience in the calculus of labeled functionals by now.

4.2. LEMMA. (i) C is closed under $+$, (ii) $f < g \Rightarrow f + m < g + m$, $f \leqq g \Rightarrow f + m \leqq g + m$, (iii) $m < n \Rightarrow f + m < f + n$, $m \leqq n \Rightarrow f + m \leqq f + n$, (iv) $m > 0 \Rightarrow f < f + m$, $f \leqq f + m$, (v) $f < g \Rightarrow f \leqq g$, $f \leqq f$, (vi) $f < g \leqq h \Rightarrow f < h$, $f \leqq g < h \Rightarrow f < h$, (vii) $f > g \Rightarrow f \geqq g + 1$, (viii) $f(g + n) \geqq fg + n$.
PROOF. The reasoning for $<$ in 3.2 remains valid and it applies without change to $\geqq$. So (ii) to (iv) just carry over and for (i) we only have to check additionally that $f + n$ is cumulative in case f is. Suppose $(fh)* \geqq f* - h*$.
Then $((f + n)h)* = (fh)* + n \geqq f* + h* + n = (f + n)* + h*$ results by simple calculation.

(v) to (vii) are straightforwardly proved by induction over the types. (viii) follows from (vii) by induction on n. For $n = 0$ it is trivially true. Assume as induction hypothesis $f(g + n) \geqq fg + n$. Then, since $g + n + 1 > g + n$ and f is $<$-monotone $f(g + n + 1) > f(g + n)$ and hence $f(g + n + 1) > fg + n$. Then (vii) yields $f(g + n + 1) \geqq fg + n + 1$. □

4.3. LEMMA. (i) $c_n^\alpha \in C$ for any α and n, (ii) $m < n \Rightarrow c_m^\alpha < c_n^\alpha$, (iii) $m \leqq n \Rightarrow c_m^\alpha \leqq c_n^\alpha$.
PROOF. (i), (ii) and (iii) are simultaneously proved by induction on α, just as in 3.3. We only have to check additionally that c_n^α is cumulative. This is trivially the case: $(c_n f)* = (c_{n+f*})* = n + f* = c_n* + f*$. □

4.4. LEMMA. $f \in C_\alpha \Rightarrow f \geqq c_{f*}^\alpha$.
PROOF. Induction on α. For $\alpha = o$ it is trivial: $n = c_n^o$ (and $n* = n$). Let $\alpha = (\beta)\gamma$ and suppose $g \in C_\beta$. Then $fg \geqq c_{(fg)*} \geqq c_{f*+g*} = c_{f*}g$ follows from the induction hypothesis and by lemma 4.3(iii), because $(fg)* \geqq f* + g*$. From this $f \geqq c_{f*}$ can be directly concluded, since $f* = c_{f*}*$ by definition. □

It is in this sense that the c_n's are called minimally cumulative. The really minimal elements are the c_0's of course: $f \geqq c_0^\alpha$ for every $f \in C_\alpha$. (Note that $f \geqq c_{f*}$ does not generally hold for M.)

An assignment with all its values in C is called a C-assignment. The assignment c defined in 2.3.1 ($c(x^\alpha) = c_0^\alpha$) is a C-assignment by lemma 4.3.

4.5. C-LEMMA. (i) $[t]_v \in C$ for any C-assignment v.
(ii) If $x^\alpha \in t$, and $f, f' \in C_\alpha$, then

(a) $[t]_{v(x/f)} \geqq [t]_{v(x/c_0)} + f*$

(b) $f < f' \Rightarrow [t]_{v(x/f)} < [t]_{v(x/f')}$

(c) $f \leqq f' \Rightarrow [t]_{v(x/f)} \leqq [t]_{v(x/f')}$

PROOF. (i) and (ii) are proved by simultaneous induction on t. Since the reasoning for (ii)(b) and (c) and for the monotonicity part of (i) (i.e. checking (ii)(b) and (c) of definition 4.1) is completely similar to that of 3.4, we here concentrate on verifying (ii)(a) and for (i) on proving that $[t]_v$ is cumulative (clause (ii)(d) of 4.1). Clause (ii)(a) of 4.1. is left to the reader. Notice that in the cases $t = x$ and $t = t_0 t_1$, (i) is trivial anyway, since v is a C-assignment and C is closed under application.

- Let $t = x^\alpha$. Then application of lemma 4.4 yields:
$[t]_{v(x/f)} = f \geqq c_{f*}^\alpha = c_0^\alpha + f* = [t]_{v(x/c_0)} + f*$, establishing (ii)(a).
- Let $t = t_0 t_1$, $x^\alpha \in t$.

To establish (ii)(a) we have to distinguish between three cases as in 3.4.

Case 1. $x \in t_0$, $x \notin t_1$. Then $[t]_{v(x/f)} \geqq [t]_{v(x/c_0)} + f*$ follows from $[t_0]_{v(x/f)} \geqq [t_0]_{v(x/c_0)} + f*$ (induction hypothesis) by the definitions of $\geqq$ and $+$.

Case 2. $x \notin t_0$, $x \in t_1$. Apply the induction hypothesis to t_1 and use lemma 4.2(viii).

Case 3. $x \in t_0$, $x \in t_1$. Combination of cases 1 and 2, use the transitivity of $\leqq$.

- Let $t = \lambda y^\beta . t_0$, $y \notin t_0$. We can directly calculate
$([t]_v g)* = ([t_0]_v + g* + 1)* = [t_0]_v* + g* + 1 = [t]_v* + g* + 1$.
So $[t]_v$ is cumulative. For (ii)(a) suppose $x^\alpha \in t$ and $f \in C_\alpha$.
$[t]_{v(x/f)} = \langle \Lambda g \in L_\beta . [t_0]_{v(x/f)} + g* + 1, [t_0]_{v(x/f)}* \rangle$. As also $x \in t_0$ application of the induction hypothesis gives
$[t_0]_{v(x/f)} \geqq [t_0]_{v(x/c_0)} + f*$, and consequently

$[t]_{v(x/f)} \geq \langle \Lambda g \in L_{\beta}.\, [t_0]_{v(x/c_0)} + f* + g* + 1, ([t_0]_{v(x/c_o)} + f*)*\rangle =$
$[t]_{v(x/c_0)} + f*$.

- Let $t = \lambda y^{\beta}.t_0$, $y \in t_0$. Now $([t]_v g)* = ([t_0]_{v(y/g)} + 1)* \geq$ $([t_0]_{v(y/c_0)} + g* + 1)* = [t_0]_{v(y/c_0)}* + g* + 1 = [t]_v* + g* + 1$ follows by induction hypothesis (ii)(a). So $[t]_v$ is cumulative.

For (ii)(a) again suppose $x^{\alpha} \in t$ and $f \in C_{\alpha}$.
Then $[t]_{v(x/f)} = \langle \Lambda g \in L_{\beta}.\, [t_0]_{v(x/f)(y/g)} + 1, [t_0]_{v(x/f)(y/c_0)}*\rangle$.
Induction hypothesis (ii)(a) yields both

$[t_0]_{v(x/f)(y/g)} \geq [t_0]_{v(x/c_0)(y/g)} + f*$ and
$[t_0]_{v(x/f)(y/c_0)} \geq [t_0]_{v(x/c_0)(y/c_0)} + f*$, and consequently
$[t]_{v(x/f)} \geq \langle \Lambda g \in L_{\beta}.[t_0]_{v(x/c_0)(y/g)} + f* + 1,$
$([t_0]_{v(x/c_0)(y/c_0)} + f*)*\rangle = [t]_{v(x/c_0)} + f*$. □

4.6. DEFINITION. Let t, s be terms, v an assignment. Then $|t|_v \in \omega$ and $|t,s|_v \in \omega$ are defined by:

$$|t|_v = \sum_{x \in t} v(x)*$$

$$|t,s|_v = \sum_{x \in t \& x \notin s} v(x)*$$

4.6.1. Properties. We mention without proof some obvious properties of $|t|_v$ and $|t,s|_v$.
(i) $|x|_v = v(x)*$, (ii) $|t_0 t_1|_v \leq |t_0|_v + |t_1|_v$,
(iii) $x \notin t \Rightarrow |\lambda x.t|_v = |t|_v$, (iv) $x \in t \Rightarrow |\lambda x.t|_v + v(x)* = |t|_v$,
(v) $|t,s|_v \leq |t|_v$, (vi) $t \to s \Rightarrow |t|_v = |s|_v + |t,s|_v$.
With respect to (vi) notice that if $t \to s$, then all free variables of s are already free in t.

4.7. LEMMA. $[t]_v \geq c_{|t|_v}$ for any t and C-assignment v.

PROOF. Induction on t.

- Let $t = x$. According to 4.6.1(i) we have to check $v(x) \geq c_{v(x)*}$, which is the case by lemma 4.4.

- Let $t = t_0 t_1$. By the induction hypothesis $[t]_v \geq c_{|t_0|_v} c_{|t_1|_v}$ and some manipulation with the properties of the c_n's gives

$c_{|t_0|_v} c_{|t_1|_v} = c_{|t_0|_v + |t_1|_v}$. Hence $[t]_v \geq c_{|t|_v}$ follows since $|t_0|_v + |t_1|_v \geqq |t|_v$ (4.6.1(ii)).

- Let $t = \lambda x^{\alpha}.t_0$, $x \notin t_0$. By 4.6.1(iii) it is sufficient to show that $[t]_v \geq c_{|t_0|_v}$. This easily follows from $[t_0]_v \geq c_{|t_0|_v}$ (induction hypothesis) and some calculation:

  . For $h \in C_{\alpha}$, $[t]_v h = [t_0]_v + h_* + 1 \geq c_{|t_0|_v} + h* + 1 > c_{|t_0|_v} + h* = c_{|t_0|_v + h*} = c_{|t_0|_v} h$.

  . $[t]_v * = [t_0]_v * \geq c_{|t_0|_v} *$.

- Let $t = \lambda x^{\alpha}.t_0$, $x \in t_0$. By 4.6.1(iv) $|t_0|_{v(x/h)} = |t|_v + h*$ and $|t_0|_{v(x/c_0)} = |t|_v + c_0* = |t|_v$.

This together with applications of the induction hypothesis yields:

  . For $h \in C_{\alpha}$, $[t]_v h = [t_0]_{v(x/h)} + 1 > [t_0]_{v(x/h)} \geqq c_{|t_0|_{v(x/h)}} = c_{|t|_v + h*} = c_{|t|_v} h$.

  . $[t]_v * = [t_0]_{v(x/c_0)} * \geqq c_{|t_0|_{v(x/c_0)}} * = c_{|t|_v} *$. □

4.7.1. COROLLARY. $[t]_v * \geqq |t|_v$.

4.8. REDUCTION LEMMA. If $t \to s$, then for any C-assignment v, $[t]_v > [s]_v + |t,s|_v$.

PROOF. Induction on t. Distinguish cases according to the actual form of the reduction.

- Let $t = (\lambda x.s)t_1$, $x \notin s$.

Obviously $|t,s|_v \leqq |t_1|_v$ (only variables in t_1 possibly disappear) and hence $[t_1]_v * \geqq |t,s|_v$ by corollary 4.7.1. Then calculate: $[t]_v = [s]_v + [t_1]_v * + 1 > [s]_v + [t_1]_v * \geqq [s]_v + |t,s|_v$.

- Let $t = (\lambda x.t_0)t_1$, $x \in t_0$, $s = (x{:=}t_1)t_0$. Now no variables disappear at all, so $|t,s|_v = 0$. So it is sufficient to prove that $[t]_v > [s]_v$. Now $[t]_v = [t_0]_{v(x/[t_1]_v)} + 1 > [t_0]_{v(x/[t_1]_v)}$, but this is $[s]_v$ by the substitution lemma (2.3.6).

- $t = t_0 t_1$, $t_0 \to s_0$, $s = s_0 t_1$. Then $|t,s|_V \leq |t_0,s_0|_V$, for a variable can only disappear from t because of the reduction $t_0 \to s_0$. This and the induction hypothesis $[t_0]_V > [s_0]_V + |t_0,s_0|_V$ are used in the calculation: $[t]_V > ([s_0]_V + |t_0,s_0|_V)[t_1]_V = [s_0]_V[t_1]_V + |t_0,s_0|_V = [s]_V + |t_0,s_0|_V \geq [s]_V + |t,s|_V$.
- $t = t_0 t_1$, $t_1 \to s_1$, $s = t_0 s_1$. This case is like the preceding one. Use 4.2(viii).
- $t = \lambda x^{\alpha}.t_0$, $t_0 \to s_0$, $s = \lambda x.s_0$. There are three subcases. Case 1. $x \notin t_0$, $x \notin s_0$. Then $|t,s|_V = |t_0,s_0|_V$, and $[t]_V > [s]_V + |t,s|_V$ follows by easy application of the induction hypothesis.

Case 2. $x \in t_0$, $x \in s_0$. Then $|t,s|_V = |t_0,s_0|_V = |t_0,s_0|_{V(x/f)}$ for any f, and from this the result follows easily by the induction hypothesis.

Case 3. $x \in t_0$, $x \notin s_0$. Then $|t_0 \cdot s_0|_{V(x/f)} = |t,s|_V + f_*$ for any f, and hence by the induction hypothesis one obtains:

$[t]_V = \langle \Lambda f \in L_\alpha . [t_0]_{V(x/f)} + 1,\ [t_0]_{V(x/c_0)}{}^* \rangle >$

$\langle \Lambda f \in L_\alpha . [s_0]_{V(x/f)} + |t_0,s_0|_{V(x/f)} + 1,\ ([s_0]_{V(x/c_0)} + |t_0,s_0|_{V(x/c_0)})^* \rangle =$

$\langle \Lambda f \in L_\alpha . [s_0]_V + |t,s|_V + f^* + 1,\ [s_0]_V{}^* + |t,s|_V + 0 \rangle =$

$[\lambda x.s_0]_V + |t,s|_V$. □

4.8.1. COROLLARY. $h(t) \leq [t]^*$.

4.9. THEOREM. If $[t]^* > 0$, then there exists an s with $t \to s$ and such that

(i) if t is not of the form $\lambda x.t_0$, then $[t] = [s] + 1$,

(ii) if $t = \lambda x.t_0$, then $[t]^* = [s]^* + 1$.

PROOF. Induction on t. Distinguish cases as to the form of t.
- $t = x$. This case is trivial, as $[x]^* = c_0{}^* = 0$.
- $t = (\lambda x.t_0)t_1$, $x \notin t_0$. Then $[t] = [t_0] + [t_1]^* + 1$.

If $[t_1]^* = 0$, then let $s = t_0$.

Otherwise by the induction hypothesis for some s_1, $t_1 \to s_1$ and

$[t_1]^* = [s_1]^* + 1$. Let $s = (\lambda x.t_0)s_1$.

- $t = (\lambda x.t_0)t_1$, $x \in t_0$. Then $[t] = [t_0]_{c(x/[t_1])} + 1 = [(x{:=}t_1)t_0] + 1$ (substitution lemma).

Let $s = (x{:=}t_1)t_0$.

- $t = t_0 t_1$, t_0 not of the form $\lambda x.-$. There are two subcases.

Case 1. $[t_0]^* > 0$. Then by the induction hypothesis $t_0 \to s_0$ for some s_0 such that $[t_0] = [s_0] + 1$. Let $s = s_0 t_1$; $[t] = ([s_0] + 1)[t_1] = [s] + 1$.

Case 2. $[t_0]^* = 0$. Then, by 4.8.1., t_0 is in normal form and consequently must have the form $xt_2 \ldots t_n$ (with $t_2, \ldots, t_n$ in normal form). As $[x] = c_0^\alpha$ for a suitable α, $[t_0] = c_n$ for some n, and therefore, since $[t_0]^* = 0$, $[t_0] = c_0$. But then, as $([t_0][t_1])^* > 0$, certainly $[t_1]^* > 0$ and hence, by the induction hypothesis, $t_1 \to s_1$ for a certain s_1 such that $[t_1]^* = [s_1]^* + 1$. Let $s = t_0 s_1$, and calculate: $[t] = c_0[t_1] = c_{[t_1]^*} = c_{[s_1]^*+1} = c_0[s_1] + 1 = [s] + 1$.

- $t = \lambda x.t_0$. Notice that, regardless of whether $x \in t_0$ or not, $[t]^* = [t_0]^*$. By the induction hypothesis $t_0 \to s_0$ for a certain s_0 such that $[t_0]^* = [s_0]^* + 1$. Let $s = \lambda x.s_0$. □

4.9.1. COROLLARY. $h(t) = [t]^*$.

PROOF. By 4.9. $h(t) \geqq [t]^*$ and by 4.8.1. $h(t) \leqq [t]^*$. □

4.9.2. REMARK. One may observe that the proof of 4.9 implicitly contains a strategy for constructing a reduction sequence of t of maximal length. The same reduction strategy is defined in Barendregt ['81] (from Barendregt et al. ['76]), where it is called the perpetual strategy F_∞. It yields for any term t in the type free $\lambda\beta$-calculus an infinite reduction sequence, if there is one.

§5. EVALUATING THE COMBINATORS.

5.1. In the combinatory variant of the typed λ-calculus, the system CL^τ (typed combinatory logic), the role of λ-abstraction is taken over by constants $K_{\alpha,\beta} \in (\alpha)(\beta)\alpha$ and $S_{\alpha,\beta,\gamma} \in ((\alpha)(\beta)\gamma)((\alpha)\beta)(\alpha)\gamma$, for all types α, β and γ. From these, terms are built up with application as sole term forming operation. In CL^τ, $t \to s$ holds if s results from t by replacing a part of the form $\mathbf{K}t_0t_1$ by t_0 or a part $\mathbf{S}t_0t_1t_2$ by $t_0t_2(t_1t_2)$.

Let the exact valuations for **K** and **S** be given by:

$[\mathbf{K}_{\alpha,\beta}] = \langle \Lambda f \in L_{\alpha}.\langle \Lambda g \in L_{\beta}.f + g* + 1,\ f*\rangle, 0\rangle$

$[\mathbf{S}_{\alpha,\beta,\gamma}] = \langle \Lambda f \in L_{(\alpha)(\beta)\gamma}.\langle \Lambda g \in L_{(\alpha)\beta}.\langle \Lambda h \in L_{\alpha}.fh(qh) + 1,\ f* + g*\rangle, f*\rangle, 0\rangle$

(NB: Quantification over assignments is not needed; if variables are included one simply puts $[x^{\alpha}] = c^{\alpha}_{0}$.)

Then it can be established by the method of this paper that $[t]$ is exactly estimating again: $h(t) = [t]*$.

As a matter of fact it suffices to show that the valuations of CL^{τ}-terms are in M, so that the proof becomes as simple as the 'quick' proof in section 3.

5.2. Extension of our method to systems incorporating a recursion operator, however, does complicate matters. The uniformity result in 2.3.4 for example, will no longer hold as stated there. In particular, given a fixed $t \in (o)\beta$, $h(ts)$ will depend not only on $h(s)$, but on the numerical value of s as well. As a consequence a more involved type structure is needed, built up from $\omega\times\omega$.

We intend to investigate this in another paper.

FOOTNOTE

[1] The proof in Gandy ['80] is supposed to rest on the assumption of weak normalization. This assumption is used to ensure that every closed term of type o in the λ-I^{+} calculus reduces to a numerical term, denoting in a trivial way a natural number. (As a matter of fact, for uniqueness a Church-Rosser result would be required as well.) It seems, however, that the assumption can be eliminated, because under the standard interpretation each closed term of type α already denotes an element of T_{α}: and in particular terms of type o denote natural numbers.

REFERENCES

H.P. Barendregt [1981], The Lambda Calculus. Amsterdam etc. (North-Holland).

H.P. Barendregt, J. Bergstra, J.W. Klop, H. Volken [1976], Degrees, reductions and representability in the lambda calculus. Preprint no. 22, University of Utrecht, Department of Mathematics.

D.T. van Daalen [1980], The Language Theory of Automath (Dissertation), Technological University Eindhoven.

R.O. Gandy [1980], Proofs of Strong Normalization, in: To H.B. Curry (eds. J.P. Seldin and J.R. Hindley), pp. 457-477. London etc. (Academic Press).

A.S. Troelstra [1973], Metamathematical Investigation of Intuitionistic Arithmetic and Analysis; Lecture Notes in Mathematics 344. Berlin etc. (Springer Verlag).

R.C. de Vrijer [1985], A direct proof of the Finite Developments Theorem. Journal of Symbolic Logic, Vol. 50, nr. 2, pp. 339-343.

4
STRONG NORMALIZATION IN N-HA$^{\omega}{}_{p}$

Introduction. The purpose of this note is to amend the notion of computability employed in proofs of strong normalization, so as to make it work also for systems which include a surjective pairing. To facilitate comparison with existing proofs (i.e. for systems without surjective pairing), we work with the system **N-HA**$^{\omega}{}_{p}$ as it is formulated in Troelstra [1973], and roughly follow the set up of the proof of strong normalization of **N-HA**$^{\omega}$ given there in the §§2.2.12-2.2.19 and 2.2.30-2.2.31. Notations, terminology, etc. not defined here are taken from Troelstra [1973] (further referred to by [Tr]).

The new definition of computability has an extra clause for the product types of course. But apart from that, it deviates from the usual definitions in two more respects. First, the computable terms are explicitly required to be strongly normalizing. Secondly, by the very form of the computability conditions it is enforced at once that computability is preserved under reduction.

The first deviation is not an essential feature, but merely a matter of taste. The proof would not change much if one adopted the usual strategy of requiring strong normalization only at the ground type 0, and then seperately proving the implication

computable ⇒ strongly normalizable

for the higher types.

As to incorporating strong normalizability and closure under reduction in the computability conditions, the notion of computability employed here resembles that in de Vrijer [1975]. For the λ-based theory with surjective pairing, but without recursion, the method used there can be adapted to yield a proof of strong normalization that is simpler than the one given here. In an Automath setting such a proof is presented in van Daalen [1980], p.294 ff.

We give our proof for the combinatory and the λ-based versions of **N-HA**$^{\omega}{}_{p}$ together. Those who are only interested in one of both versions can just leave either the combinators or the λ's out.

The system N-HA$^{\omega}{}_{p}$. The types of **N-HA**$^{\omega}{}_{p}$ are built up from the ground type 0 by finite iterations of the operations of forming the *function type* $(\alpha)\beta$ and the *product type* $\alpha\times\beta$ from the types α and β. The terms of **N-HA**$^{\omega}{}_{p}$ all have a unique type. The notation $t\in\sigma$ is used to indicate that t has type σ. The atomic terms are (with ρ,σ,τ arbitrary types):

- the variables $x^\sigma, y^\sigma, z^\sigma, \ldots \in \sigma$;
- the combinators $\Pi_{\rho,\sigma} \in (\rho)(\sigma)\rho$ and $\Sigma_{\rho,\sigma,\tau} \in ((\rho)(\sigma)\tau)((\rho)\sigma)(\rho)\tau$;
- the constants 0 (zero) $\in 0$, S (successor) $\in (0)0$ and the recursor $R_\sigma \in (\sigma)((\sigma)(0)\sigma)(0)\sigma$;
- the pairing and projection constants $D_{\sigma,\tau} \in (\sigma)(\tau)\sigma\times\tau$, $D'_{\sigma,\tau} \in (\sigma\times\tau)\sigma$ and $D''_{\sigma,\tau} \in (\sigma\times\tau)\tau$.

The term forming rules are:

- application: if $t \in (\sigma)\tau$ and $s \in \sigma$, then $ts \in \tau$;
- abstraction: if $t[x^\sigma] \in \tau$, then $\lambda x^\sigma.t[x^\sigma] \in (\sigma)\tau$.

Then **N-HA$^\omega$$_p$** has the following contraction rules:

- $\Pi_{\rho,\sigma}t_1t_2$ contr t_1; $\Sigma_{\rho,\sigma,\tau}t_1t_2t_3$ contr $t_1t_3(t_2t_3)$;
- $R_\sigma t_1t_2 0$ contr t_1; $R_\sigma t_1t_2(St_3)$ contr $t_2(R_\sigma t_1t_2t_3)t_3$;
- $D'_{\sigma,\tau}(D_{\sigma,\tau}t_1t_2)$ contr t_1; $D''_{\sigma,\tau}(D_{\sigma,\tau}t_1t_2)$ contr t_2; $D_{\sigma,\tau}(D'_{\sigma,\tau}t)(D''_{\sigma,\tau}t)$ contr t;
- $(\lambda x^\sigma.t[x^\sigma])s$ contr $t[s]$.

As in [Tr] we use the notations $>_1$ and $\geq$ for the one step reduction relation which is generated by contr and for the reflexive and transitive closure of $>_1$ respectively.

Some extra terminology: a term t is called strongly normalizing (SN) if all its reduction sequences are finite. If so, by h(t) is denoted the length of a reduction sequence of t of maximal length (= the height of the reduction tree); and by $\nu(t)$ the maximum of the set $\{i \in N \mid (\exists t')(t \geq S^i t')\}$.

Computability and Strong Normalization. A notion of computability, C, is defined for terms of **N-HA$^\omega$$_p$** by induction on the typestructure. C_σ denotes the computable terms of type σ.

DEFINITION 1. $C_\sigma(t)$ iff the following three clauses are satisfied.

(i) t is SN.

(ii) If $\sigma = (\alpha)\beta$ and $t \geq t'$ and $s \in C_\alpha$, then $t's \in C_\beta$.

(iii) If $\sigma = \alpha\times\beta$ and $t \geq Dt't''$, then $t' \in C_\alpha$ and $t'' \in C_\beta$.

In a series of lemmas we now establish some obvious closure properties of C and list a few basic implications that can be used in proving that certain terms are computable.

LEMMAS. (i) $C(t) \,\&\, t \geq t' \Rightarrow C(t')$.

(ii) Any term which is formed by repeated application from computable terms is computable.

(iii) If $t \in 0$, then $SN(t) \Rightarrow C(t)$.

(iv) If $t \equiv xt_1\ldots t_n$ $(n \geq 0)$, then $SN(t) \Rightarrow C(t)$ $(\Leftrightarrow (\forall i \leq n)SN(t_i))$.

(v) $(\forall\sigma)(x^\sigma \in C_\sigma)$.

(vi) If $t \in (\sigma_1)\ldots(\sigma_n)\tau$, then
$$(\forall t_1 \in C_{\sigma 1})\ldots(\forall t_n \in C_{\sigma n})C(tt_1\ldots t_n) \Rightarrow C(t).$$

(vii) Let either $t \in 0$ or $t \in \alpha\times\beta$. Then

(a) if $t \equiv Dt't''$, then $C(t') \& C(t'') \& (\forall s)(t >_1 s \Rightarrow C(s)) \Rightarrow C(t)$.

(b) if t not of the form Dt't", then $(\forall s)(t >_1 s \Rightarrow C(s)) \Rightarrow C(t)$.

(viii) Let $t \in \alpha\times\beta$, then $C(D't) \& C(D't) \Rightarrow C(t)$.

PROOFS. (i), (ii) and (iii) are obvious (for (iii) just observe that the clauses (ii) and (iii) of the definition of C do not apply).

(iv) Assume $t \in \tau$ and apply induction on τ. We need only verify the clauses (ii) and (iii) of definition 1. Observe that if $t \geq t'$, then t' will have the form $x t'_1\ldots t'_n$ with $t_i \geq t'_i$ $(1 \leq i \leq n)$. So if $\tau = \alpha\times\beta$, then clause (iii) of the computability definition does not apply. That leaves the case that $\tau = (\alpha)\beta$. Let $s \in C_\alpha$. Then, as $t'_1\ldots t'_n$ and s are all SN and $t's \equiv x t'_1\ldots t'_n s$, also t's is SN and hence $C(t's)$ follows by the induction hypothesis.

(v) For each type σ we have $x^\sigma \in C_\sigma$.

(vi) This is proved by induction on n. Note that t is SN since $t x_1^{\sigma 1}\ldots x_n^{\sigma n}$ is, as $C(t x_1^{\sigma 1}\ldots x_n^{\sigma n})$ follows by (v) and the assumption. For the case $n=1$, assume $(\forall t_1 \in C_{\sigma 1})C(tt_1)$ and let $t \geq t'$ and $s \in C_{\sigma 1}$. Then also $ts \geq t's$ and hence the computability of t follows since C is closed under reduction by (i). The induction step is trivial.

(vii) Of course t is SN iff each s such that $t >_1 s$ is SN. Furthermore $t \geq Ds's''$ iff either $t \equiv Ds's''$ or $s \geq Ds's''$ for some s such that $t >_1 s$.

(viii) SN(t) follows from SN(D't). Assume $t \geq Dt't''$. Then $D't \geq D'(Dt't'') >_1 t'$ and C(t') follows from C(D't) becase C is closed under application (lemma (ii)). Similarly C(t") follows from D"t. □

We now first prove that all terms that can be formed without employing the rule of λ-abstraction are computable. The terms that include λ-abstraction will be treated afterwards.

THEOREM 1. Any term t of $\mathbf{N\text{-}HA}^{\omega}{}_{\mathbf{p}}$ which does not contain λ-abstraction is computable.

PROOF. We proceed as in [Tr] §2.2.19, by proving that all atomic terms are computable. The theorem is then implied since C is closed under application by lemma (ii).

(i)-(ii) C(0) and C(S) are immediate.

(iii) $C(\Pi_{\sigma,\tau})$. Let $t_1\ldots t_n$ be computable terms such that either $\Pi_{\sigma,\tau}t_1\ldots t_n \in 0$ or $\Pi_{\sigma,\tau}t_1\ldots t_n \in \alpha\times\beta$. By lemma (vi) it is sufficient to prove $C(\Pi_{\sigma,\tau}t_1\ldots t_n)$. This we do by induction on $h(t_1)+\ldots+h(t_n)$. Assume that $\Pi_{\sigma,\tau}t_1\ldots t_n >_1 s$. Again, by lemma (vii)(b), it is sufficient to prove C(s). There are two cases to consider.

– $s \equiv t_1 t_3 \ldots t_n$. Then C(s) follows by lemma (ii).
– $s \equiv \Pi_{\sigma,\tau} s_1 \ldots s_n$ and $(\exists i)(t_i >_1 s_i \,\&\, (\forall j \leq n)(j \neq i \Rightarrow s_j \equiv t_j))$. Then C(s) follows from the induction hypothesis.
(iv) $C(\Sigma_{\rho,\sigma,\pi})$ is proved similarly.
(v) $C(x^\sigma)$ was proved in lemma (iv).
(vi) By induction with respect to the ordered pair, $<\nu(t_3), h(t_1)+\ldots+h(t_n)>$ we prove that if $t_1 \ldots t_n$ are computable terms such that either $R_\sigma t_1 \ldots t_n \in 0$ or $R_\sigma t_1 \ldots t_n \in \alpha \times \beta$, then $C(R_\sigma t_1 \ldots t_n)$. From this $C(R_\sigma)$ follows by lemma (vi). Again it is, by lemma (vii)(b), enough to prove that C(s) if $R_\sigma t_1 \ldots t_n >_1 s$. We distinguish three cases.
– $t_3 \equiv 0$ and $s \equiv t_1 t_4 \ldots t_n$. Then C(s) follows by lemma (ii).
– $t_3 \equiv St_0$ and $s \equiv t_2(Rt_1t_2t_0)t_4 \ldots t_n$. As $t_2, t_4, \ldots, t_n$ are computable by assumption, it will by lemma (ii) suffice to show $C(Rt_1t_2t_0)$. To that purpose observe observe first that $C(t_3)$ implies $SN(St_0)$, hence $SN(t_0)$, hence $C(t_0)$. Then, since $\nu(t_0) < \nu(St_0)$, the induction hypothesis can be applied to $R_\sigma t_1 t_2 t_0 t'_4 \ldots t'_n$ with $t'_4 \ldots t'_n$ any computable terms of the correct types, yielding $C(R_\sigma t_1 t_2 t_0 t'_4 \ldots t'_n)$. From this $C(R_\sigma t_1 t_2 t_0)$ follows by lemma (vi).
– $s \equiv R_\sigma s_1 \ldots s_n$ and $(\exists i)(t_i >_1 s_i \,\&\, (\forall j \leq n)(j \neq i \Rightarrow s_j \equiv t_j))$. Then, regarding the fact that $\nu(t_3)$ cannot increase under reduction of t_3, C(s) follows from the induction hypothesis.
(vii) We show that $C(D_{\sigma,\tau})$. By lemma (vi) it is sufficient to prove C(Dt't") for $t' \in C_\sigma$ and $t'' \in C_\tau$. This is done by induction on h(t')+h(t"). Now, as C(t') and C(t") already hold, by lemma (vii)(a) we need only verify that C(s) if $Dt't'' >_1 s$. There are two cases.
– $t' \equiv D's$ and $t'' \equiv D''s$. Then C(s) follows by lemma (viii).
– $s \equiv Ds's''$ and either $s' \equiv t'$ and $t'' >_1 s''$ or $t' >_1 s'$ and $s'' \equiv t''$. Then C(s) follows by the induction hypothesis.
(viii) $C(D'_{\sigma,\tau})$. For let $t_1, \ldots, t_n$ be computable terms such that either $D'_{\sigma,\tau} t_1 \ldots t_n \in 0$ or $D'_{\sigma,\tau} t_1 \ldots t_n \in \alpha \times \beta$. The computability of $D'_{\sigma,\tau}$ can then be concluded from $C(D'_{\sigma,\tau} t_1 \ldots t_n)$, which we prove by induction on $h(t_1)+\ldots h(t_n)$. Assume $D'_{\sigma,\tau} t_1 \ldots t_n >_1 s$ and distinguish two cases.
– $t_1 \equiv Dt't''$ and $s \equiv t't_2 \ldots t_n$. Then C(t') follows from $C(t_1)$ by clause (iii) of the computability definition and hence C(s) by lemma (ii).
– $s \equiv D'_{\sigma,\tau} s_1 \ldots s_n$ etc. (see above).
(ix) $C(D''_{\sigma,\tau})$ is proved similarly. □

Now for the treatment of λ-terms we need the stronger notion of computability under subsitution, C*.

DEFINITION 2. $C^*(t) \Leftrightarrow$ for each substitution of computable terms of the appropriate types for occurrences cf variables free in t, the resulting term is computable. (Cf. [Tr] §2.2.30.)

THEOREM 2. $C^*(t)$ for each term t of **N-HA**$^{\omega}{}_{\mathbf{p}}$.

PROOF. Again it is appropriate to prove the theorem by induction on the structure of t. It is obvious that $C^*(t)$ is closed under application, since C is.
(i) For constants C^* and C mean the same, because there are no variables for which to substitute. So we already know that 0, S, $\Pi_{\sigma,\tau,\rho}$, $\Sigma_{\sigma,\tau,\rho}$, R_σ, $D_{\sigma,\tau}$, $D'_{\sigma,\tau}$ and $D''_{\sigma,\tau}$ are all C^*.
(ii) Substituting a computable term t for the variable x^σ in x^σ results in the computable term t. Substituting nothing for x^σ in x^σ, the result is x^σ, already shown to be computable. Hence also variables are C^*.
(iii) Since, as we pointed out above, C^* is closed under application, it remains to be shown that $\lambda x.t[x] \in C^*$ can be derived from the assumption that $t[x] \in C^*$. Let $\lambda x.t'[x]$ be obtained from $\lambda x.t[x]$ by substituting some computable terms; and let $t_1, \ldots, t_n$ be computable terms such that either $(\lambda x.t'[x])t_1 \ldots t_n \in 0$ or $(\lambda x.t'[x])t_1 \ldots t_n \in \alpha \times \beta$. Notice that $t'[x] \in C$ follows from our assumption that $t[x] \in C^*$. So we are allowed to apply induction on $h(t'[x])+h(t_1)+\ldots+h(t_n)$ in order to prove that $(\lambda x.t'[x])t_1 \ldots t_n \in C$. By lemma (vii)(b) it is sufficient to prove that C(s) if $(\lambda x.t'[x])t_1 \ldots t_n >_1 s$. There are three cases to be distinguished.
– $s \equiv t'[t_1]t_2 \ldots t_n$. Now observe that $t'[t_1]$ can be considered as the result of substituting some computable terms for free variables in $t[x]$ (namely those which were substituted to obtain $t'[x]$, plus t_1). So our assumption that $t[x] \in C^*$ implies $t'[t_1] \in C$. Consequently C(s) follows by lemma (ii).
– $t'[x] \equiv t''x$, x does not occur free in t" and $s \equiv t''t_1 \ldots t_n$. Observe that $t''t_1$ can be obtained by substituting t_1 for x in $t'[x]$, so this case reduces to the previous one.
– $s \equiv (\lambda x.t''[x])t'_1 \ldots t'_n$, obtained from $(\lambda x.t'[x])t_1 \ldots t_n$ by performing one reduction step either within $t'[x]$ or within one of the t_i's. Then C(s) follows by the induction hypothesis. □

COROLLARY. In **N-HA**$^{\omega}{}_{\mathbf{p}}$ all terms are strongly normalizing.

PROOF. Immediate by theorem 2. □

NOTE. In Troelstra [A] this result is obtained in a different way, viz. by reducing it to strong normalization for **N-HA**$^{\omega}$ (i.e. **N-HA**$^{\omega}{}_{\mathbf{p}}$ without pairing).

References.

Daalen, D.T. van [1980], *The language theory of Automath*, dissertation, Technische Hogeschool Eindhoven.

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

Troelstra, A.S. [A], Strong Normalization for typed terms with surjective pairing. To appear in *Notre Dame Journal of Formal Logic*.

Vrijer, R.C. de [1975], Big trees in a λ-calculus with λ-expressions as types. In C. Böhm (ed.), λ-Calculus and Computer Science Theory, *Lecture Notes in Computer Science* 37 (Springer).

5
BIG TREES IN A λ-CALCULUS WITH λ-EXPRESSIONS AS TYPES

§0. OUTLINE

The abstract term system λλ studied in this paper is a close relative of the Automath family of languages. In the investigation of normalization and decidability properties of these languages, λλ came up as a natural generalization of AUT-QE, the language currently in use for mechanical proof checking at the Automath project in Eindhoven. For introductory reference, see Van Daalen [4].
The introduction, section 1, is an informal account of the system λλ and its relation to other systems.
The formal description of λλ is given in the sections 2 and 3.
In 4 the main results are stated, mostly without proof.
Section 5 is devoted to proving that the big trees are well founded (BT).

§1. INTRODUCTION

1.1. Heuristic description

Before describing the main results of the paper we make a few heuristic comments, especially on the generalized typestructure involved. Here we use the "formulas-as-types" notion for interpreting mathematical statements and proofs, originated independently by De Bruijn [3] and Howard [6] (the term comes from Howard). Further references are given in 1.4.

1.1.1. Type structure

To illustrate the transition from the type structure of traditional type theory, e.g. the typed λ-calculus exhibited in Hindley et al. [5], to the types we have here, we consider constructive versions of propositional and predicate logic respectively. If we identify a proposition α with the type of its constructions (or proofs), then the implication α → β will be the type of constructions that map constructions of α to constructions of β. That is, α → β corresponds essentially to the cartesian power β^{α}.
In predicate logic a construction c of ∀x.P(x) will map any object t from the domain of quantification α to a construction of the proposition P(t). Hence the type of c(t) depends on the choice of t. The notion of power doesn't suffice any longer; we need that of cartesian product: $\prod_{x\epsilon\alpha} P(x)$.

1.1.2. Abstraction and application, two interpretations

Automath exploits the formal similarity between two kinds of abstraction: functional abstraction to form the functionlike construction $\lambda x\varepsilon\alpha.c(x)$ and the product construction $\prod_{x\varepsilon\alpha} P(x)$. It is convenient to unify these principles in the notations $[x,\alpha]c(x)$ and $[x,\alpha]P(x)$, respectively. Observe that now functional application in the former case corresponds to specification of coordinate axis in the latter. Also here we use the same notations: $<t>[x,\alpha]c(x)$ and $<t>[x,\alpha]P(x)$, which reduce to $c(t)$ and $P(t)$, respectively. Now this uniform syntactical treatment of both kinds of abstraction, very convenient for our purposes, may cause some confusion in interpretation. For example, vis à vis the formula-type analogy it amounts to using the same notation for both the predicate, i.e. "propositional function", $\lambda x\varepsilon\alpha.P(x)$ and its universal quantification $\forall x\varepsilon\alpha.P(x)$.

1.1.3. Supertypes, type inclusion

We further introduce the constant $\underline{\text{type}}$ as a "supertype" of types. Then, e.g. $[x,\alpha]\underline{\text{type}}$ will be the supertype of all those types β, such that whenever $t\ \varepsilon\ \alpha$, $<t>\beta$ is a meaningful type. Hence, carrying on the example from 1.1.2, we have $[x,\alpha]P(x)\ \varepsilon\ [x,\alpha]\underline{\text{type}}$. Moreover, because of the possibility of interpreting $[x,\alpha]P(x)$ as a proposition $\forall x\varepsilon\alpha.P(x)$, we require that $[x,\alpha]P(x)\ \varepsilon\ \underline{\text{type}}$. This motivates the facility in $\lambda\lambda$ (and in AUT-QE) to pass here from $[x,\alpha]\underline{\text{type}}$ to $\underline{\text{type}}$, known as the principle of type inclusion: $[x,\alpha]\underline{\text{type}} \subseteq \underline{\text{type}}$ (cf. [3], [4] and 3.5.2 below).

In order to clarify this slightly ambiguous situation one could for the product construction introduce the Π's again, and obtain $\Pi[x,\alpha]P(x)\ \varepsilon\ \underline{\text{type}}$ for the product and $[x,\alpha]P(x)\ \varepsilon\ \Pi[x,\alpha]\underline{\text{type}}$ for the type-valued function, respectively (cf. [12]).

1.1.4. $\lambda\lambda$-theories

Expressions are built up by using the principles of abstraction and application mentioned above, starting from variables, parameters and constants. A particular choice of the constants and their (super) type assignments will depend on the interpretation one has in mind. Such a choice is formally fixed by a base (cf. 3.1). Each base determines a specific $\lambda\lambda$-theory.

In informal mathematics new notions are always introduced in a context, possibly indicated by the presence of certain parameters and assumptions. This observation is reflected in $\lambda\lambda$ by the fact that constants are allowed to depend on parameters. We now illustrate the treatment of constants in $\lambda\lambda$ and the parameter mechanism involved.

Let $C_1(\alpha,\beta)$ be a type constant, to be interpreted as the proposition $\exists x\varepsilon\alpha.<x>\beta$, where β is supposed to represent a predicate on the type α. Informally introducing $C_1(\alpha,\beta)$ one might stipulate:

(1) "Let P be a type, Q be a predicate on P. Then we will consider $C_1(P,Q)$ as a proposition."

In $\lambda\lambda$ the (super) types of parameters are indicated by superscripts and hence the corresponding axiom reads:

$$(2)\quad C_1(P^{\underline{type}}, Q^{[x,P]\underline{type}})\ \varepsilon\ \underline{type}.$$

The rule of existence introduction can now be formalized by adding another constant $C_2(\alpha,\beta)$ together with the axiom

$$(3)\quad C_2(P^{\underline{type}}, Q^{[x,P]\underline{type}})\ \varepsilon\ [x,P][y,<x>Q]C_1(P,Q).$$

When actually given $\alpha\ \varepsilon\ \underline{type}$ and $\beta\ \varepsilon\ [x,\alpha]\underline{type}$, $C_1(\alpha,\beta)\ \varepsilon\ \underline{type}$ and $C_2(\alpha,\beta)\ \varepsilon\ [x,\alpha][y,<x>\beta]C_1(\alpha,\beta)$ can now be obtained as instances of (2) and (3), respectively. Moreover, for objects $t\ \varepsilon\ \alpha$ and $s\ \varepsilon\ <t>\beta$ application and β-reduction yield: $<s><t>C_2(\alpha,\beta)\ \varepsilon\ C_1(\alpha,\beta)$.

For further explanation on the subject of interpretation we refer to the treatment of AUT-QE in [4] and to Van Benthem Jutting [2].

1.2. Applicability

Usually, in type theory as in the Automath languages, term application is subjected to the applicability condition: $<t>f$ is a term iff there are types α and β such that $t\ \varepsilon\ \alpha$ and $f\ \varepsilon\ [x,\alpha]\beta$. Now in typed λ-calculus this condition is easy to formulate. The type structure and the assignments of types to terms are given in advance, i.e. all of the syntax precedes the generation of theorems. In our case however, types depend on objects and the type assignments are themselves treated as theorems in $\lambda\lambda$. Hence here the applicability condition would make derivability interfere with term formation. A common way of dealing with this complication (cf. Automath, Martin-Löf, etc.) is to generate the terms (including the types) simultaneously with the theorems.

By contrast we take the approach of allowing unrestricted application in $\lambda\lambda$, but instead now subjecting the rule of β-reduction

$$(4)\quad <t>[x,\alpha]\Sigma >_1 [x/t]\Sigma$$

to the condition $t\ \varepsilon\ \alpha$. We can then formulate an applicability condition by referring to derivability in $\lambda\lambda$ and so define the set of legitimate terms. The legitimate fragment $\lambda\lambda - \ell$ of $\lambda\lambda$ is the system one obtains by restricting $\lambda\lambda$ to the language containing only legitimate terms. Hence $\lambda\lambda - \ell$ may be considered as the part of $\lambda\lambda$ that is significant for interpretation. (Though, of course, the illegitimate terms do have a computational interpretation in the term model.) The justification for the above sketched procedure lies in the following result:

(5) $\lambda\lambda$ is a conservative extension of $\lambda\lambda - \ell$.

This property may be regarded as a soundness criterion for the notion of legitimacy as defined above, and hence for $\lambda\lambda$: if the equality of two significant (read: legitimate) terms can be proved in $\lambda\lambda$, it can be done using only significant terms. The proof of (5) uses the result on "big trees" described below.

1.3. Decidability, big trees

We now turn to a second desirable property of the systems:

(6) $\lambda\lambda$ and hence $\lambda\lambda - \ell$ are decidable.

Decidability of the typed λ-calculus is an easy corollary of the strong normalizability property (SN) and the Church-Rosser property (CR). Every term reduces effectively to its normal form (nf) and two terms are equal iff their nf's are identical. However, although both SN and CR go through for $\lambda\lambda$, they are not sufficient for the decidability, as we will now explain.

In the discussion we make use of an effective function τ, which assigns canonically to every object a type such that $t \in \tau(t)$. Then, since we have uniqueness of types (cf. 3.3.5):

(7) $t \in \alpha \Leftrightarrow \tau(t) = \alpha$

(where by CR, = is equivalent to having the same nf).

So, in order to see if (4) holds, we must first determine if $\tau(t)$ and α have the same nf (by (7)). Then in the process of reducing these terms questions of the form (4) may arise again, and so on.

To deal with this problem, we proceed as follows. Let $\rightarrow$ be the improper reduction relation generated by

(i) usual $\beta\eta$-reduction,

(ii) applying τ,

(iii) taking proper subterms.

Call the tree of $\rightarrow$-reduction sequences of a term Σ the "big tree" of Σ. Then we prove instead of SN the stronger property:

(BT) big trees of terms in $\lambda\lambda$ are well founded.

Together with CR this result easily implies the decidability. Further, as mentioned above, it is also used in the proof of (5).

In his thesis Nederpelt [8] stated as a conjecture for his system Λ, the closure property:

(8) Legitimate terms reduce to legitimate terms.

It turns out that BT (for Λ) implies (8). Further it seems that BT can be proved for Λ by a method, similar to the one used here. (Note that by contrast (8) for $\lambda\lambda$ is a simple consequence of the formulation of the system and its proof does not require BT.)

We feel that, apart from the applications described, BT may have some interest on its own.

1.4. Historical remarks

The first proof of normalization of an Automath system was given in Van Benthem Jutting [1]. Nederpelt [8] proved strong normalization for his system Λ. He made two conjectures: the above mentioned closure property for Λ' and CR for the system with η-reduction. The latter conjecture was proved by Van Daalen (to appear in his thesis). The result is assumed in this paper.

Scott [10] suggested to use the ideas of De Bruijn [3] for the formalization of an intuitionistic theory of constructions. At about the same time Howard [6] came up with similar ideas. The line is pursued by Martin-Löf [7]. His theory of types is claimed to be a natural formal framework for intuitionistic mathematics. The different accents in motivation - Automath more practical, Martin-Löf more philosophical - might be responsible for some of the differences in the investigated systems.

§2. THE LANGUAGE, EXPRESSIONS

In this paragraph we specify the language of a $\lambda\lambda$-theory. This language is affected by the choice of a base (cf. 3.1). A similarity type (defined below) codes the information, which is relevant for the formation of expressions. Hence for each similarity type s we define the language $\mathcal{L}_s$.

2.1. Alphabet

All formal symbols used are from the *alphabet* consisting of the symbols for

variables	$x, y, z, \ldots$
parameters	$P, Q, R, \ldots$
constants	$C_1, C_2, C_3, \ldots$ and <u>type</u>
binary relations	$=, \geq, >, >_1, \varepsilon$,

and the auxiliary symbols [,], <,>, (,), ,.

Variable symbols will be indexed by types to become (object-) variables, parameter symbols by types and supertypes to become object- and type-parameters, respectively. The set of variables is assumed to be such that whenever needed, we are able to choose uniquely a "new" variable of the desired type, not yet occurring in the context. The enumeration of the constant symbols is meant to show the order in which they can be introduced in a particular interpretation (cf. 1.1.4 and the notions of date and base). In Automath this would be the order in which they appear in a "book" (cf. [4]).

2.2. Similarity type

A *similarity type* s is a triple $<S_0,S_1,\sigma>$, where S_0 and S_1 are disjoint sets of natural numbers and σ is a function from $S_0 \cup S_1$ to $\{0,1\}^*$, the set of finite (possibly empty) sequences of zeros and ones.

Here S_0 indicates the set of constant symbols used for object-constants, S_1 the set of constant symbols used for type-constants and if $i \in S_0 \cup S_1$, then $\sigma(i)$ determines the positions of object- and type-parameters of C_i (cf. 2.3.1 (ii)).

2.3. Expressions

The expressions fall into three *sorts*: objects, types and supertypes. These are simultaneously defined in 2.3.1. In the definition we use already the notion of *closed* expression, to be defined in 2.3.4. However, it is clear that the definitions could have been given simultaneously.

2.3.1. Definition. Given a similarity type s, the sets of *variables*, *parameters*, *constants*, *objects*, *types* and *supertypes*, building together the set E_s of *expressions* is defined by simultaneous induction.

(i) If x is a variable symbol, P a parameter symbol, α a type, β a closed (cf. 2.3.4) type and β^* a closed supertype, then x^α is a variable, P^β is an object-parameter and P^{β^*} is a type-parameter.

(ii) Let $\sigma(i) = \delta_1,\ldots,\delta_n$ and $\Sigma_1,\ldots,\Sigma_n$ be expressions such that Σ_j is an object if $\delta_j = 0$ and Σ_j is a type if $\delta_j = 1$, then $C_i(\Sigma_1,\ldots,\Sigma_n)$ is an object-constant if $i \in S_0$ and a type-constant if $i \in S_1$.

(iii) Variables, object-parameters and object-constants are *atomic* objects. Type-parameters and type-constants are atomic types and type is the only atomic supertype.

(iv) If f and t are objects, α and β are types, α^* is a supertype and x^α a variable, then $<t>f$ and $[x^\alpha,\alpha]t$ are objects, $<t>\beta$ and $[x^\alpha,\alpha]\beta$ are types and $<t>\alpha^*$ and $[x^\alpha,\alpha]\alpha^*$ are supertypes.

2.3.2. Conventions. As syntactical variables we use $\Sigma, \Gamma, \ldots$ for expressions in general, $f, g, t, s, \ldots$ for objects, $\alpha, \beta, \ldots$ for types and $\alpha^*, \beta^*, \ldots$ for supertypes. The symbols for variables, parameters and constants are used themselves as syntactical variables for their respective categories as well.
As long as no confusion arises we will freely add and omit indexes. In particular the superscripts of variables and parameters are suppressed where possible, e.g. we write $[x,\alpha]x$ instead of $[x^\alpha,\alpha]x^\alpha$.
Vectorial notation is introduced for sequences of expressions; e.g. $\vec{\alpha}$ is short for the sequence $\alpha_1, \ldots, \alpha_n$, where the number n is either known or not essential.
As = is a symbol of the language we use $\equiv$ for syntactic equality between expressions.

Now follow some more technical and notational definitions concerning expressions.

2.3.3. Complexity, length and date
According to definition 2.3.1 each expression has a construction, easily seen to be unique, consisting of a finite number of applications of the rules (i) to (iv).
The *complexity* $c(\Sigma)$ of an expression Σ is the number of steps in its construction.
By induction on $c(\Sigma)$ we define two more measures on Σ: its *length* $\ell(\Sigma)$ and *date* $d(\Sigma)$.

$\ell(\Sigma) = 1$ if Σ is either a variable, a parameter or $\underline{\text{type}}$;

$\ell(C(\Sigma_1, \ldots, \Sigma_n)) = \max(\ell(\Sigma_1), \ldots, \ell(\Sigma_n)) + 1$; $\ell(\langle t \rangle \Gamma) = \max(\ell(t), \ell(\Gamma)) + 1$;

$\ell([x,\alpha]\Gamma) = \max(\ell(\alpha), \ell(\Gamma)) + 1$.

$d(\underline{\text{type}}) = 0$; $d(x^\alpha) = d(\alpha)$; $d(P^\Gamma) = d(\Gamma)$; $d(C_i(\Sigma_1, \ldots, \Sigma_n)) =$

$= \max(i, d(\Sigma_1), \ldots, d(\Sigma_n))$; $d(\langle t \rangle \Gamma) = \max(d(t), d(\Gamma))$; $d([x,\alpha]\Gamma) =$

$= \max(d(\alpha), d(\Gamma))$.

Notice that $d(\Sigma)$ is the greatest natural number i, such that C_i appears in the construction of Σ.

2.3.4. Free variables, parameters, special variables
By induction of $\ell(\Sigma)$ we define the sets $FV(\Sigma)$ of *free variables* and $Par(\Sigma)$ of *parameters* of Σ.

$FV(\underline{\text{type}}) = \emptyset$; $Par(\underline{\text{type}}) = \emptyset$

$FV(x) = \{x\}$; $Par(x) = \emptyset$

$FV(P) = \emptyset$; $Par(P) = \{P\}$

$$FV(C(\Sigma_1,\ldots,\Sigma_n)) = \bigcup_{i\leq n} FV(\Sigma_i) \quad ; \quad Par(C(\Sigma_1,\ldots,\Sigma_n)) = \bigcup_{i\leq n} Par(\Sigma_i)$$

$$FV(<t>\Gamma) = FV(t) \cup FV(\Gamma) \quad ; \quad Par(<t>\Gamma) = Par(t) \cup Par(\Gamma)$$

$$FV([x,\alpha]\Gamma) = FV(\alpha) \cup (FV(\Gamma) \setminus \{x\}) \quad ; \quad Par([x,\alpha]\Gamma) = Par(\alpha) \cup Par(\Gamma) \ .$$

The set $SV(\Sigma)$ of *special variables* of Σ is defined as

$$SV(\Sigma) = \bigcup_{x^\alpha \in FV(\Sigma)} FV(\alpha) \ .$$

An expression Σ is called *closed* if $FV(\Sigma) = \emptyset$.
For a sequence $\Sigma_1,\ldots,\Sigma_n$ we introduce the notation $F(\vec{\Sigma}) = \bigcup_{i\leq n} F(\Sigma_i)$, F is FV, SV or Par.

2.3.5. Proper subexpressions

The relation $\succ$ (contains as a proper subexpression) between expressions is the smallest transitive relation such that $C(\Sigma_1,\ldots,\Sigma_n) \succ \Sigma_i$; $<t>\Gamma \succ t$; $<t>\Gamma \succ \Gamma$; $[x,\alpha]\Gamma \succ \alpha$ and $[x,\alpha]\Gamma \succ \Gamma$.

2.3.6. Simultaneous substitution

Let $P_1,\ldots,P_m$ and $x_1,\ldots,x_n$ be sequences of distinct parameters and variables. And let $t_1,\ldots,t_n$ be a sequence of objects and $\Sigma_1,\ldots,\Sigma_m$ a sequence of expressions, such that Σ_i is of the same sort (i.e. object or type) as P_i. Then the result $[\vec{P},\vec{x}/\vec{\Sigma},\vec{t}]\Sigma$ of *simultaneous substitution* of $\vec{\Sigma}$, $\vec{t}$ for $\vec{P}$, $\vec{x}$ is defined by induction on $\ell(\Sigma)$. In the definition we abbreviate $[\vec{P},\vec{x}/\vec{\Sigma},\vec{t}]\Gamma$ by Γ'.
$x_i' \equiv t_i$ $(1 \leq i \leq n)$ and $x' \equiv x$ if $x \notin \{x_1,\ldots,x_n\}$.
$P_i' \equiv \Sigma_i$ $(1 \leq i \leq m)$ and $P' \equiv P$ if $P \notin \{P_1,\ldots,P_m\}$.
$(C(\Sigma_1,\ldots,\Sigma_k))' \equiv C(\Sigma_1',\ldots,\Sigma_k')$, $\underline{type}' \equiv \underline{type}$.
$(<t>\Gamma)' \equiv <t'>\Gamma'$.
$([x,\alpha]\Gamma)' \equiv [y,\alpha']([x/y]\Gamma)'$, where y is a new variable.
By $[\vec{P},\vec{x}/\vec{\Sigma},\vec{t}]\vec{\Gamma}$ we denote the sequence $\Gamma_1',\ldots,\Gamma_k'$.

2.3.7. α-equivalence

The relation $\approx$ of *α-equivalence* between expressions is the smallest equivalence relation such that, if $\Sigma \approx \Gamma$, $t \approx s$ and $\alpha \approx \beta$, then also $x^\alpha \approx x^\beta$, $P^\Sigma \approx P^\Gamma$, $C(\Lambda_1,\ldots,\Sigma,\ldots,\Lambda_n) \approx C(\Lambda_1,\ldots,\Gamma,\ldots,\Lambda_n)$, $<t>\Sigma \approx <s>\Gamma$ and if $y \notin FV(\Gamma)$ also $[x,\alpha]\Sigma \approx [y,\alpha][x/y]\Gamma$.

In the sequel we shall simply identify α-equivalent expressions. Formally one might pass to α-equivalence classes, considering an expression as merely a name, denoting the class it belongs to, and show that the preceding definitions behave well with respect to $\approx$.

In some places names of bound variables will be tacitly assumed to be chosen such that no "clashes" arise.

2.3.8. Lemma. Let $\{Q_1,\dots,Q_m\} \cap \mathrm{Par}(\vec{\Sigma},\vec{t}) = \{y_1,\dots,y_n\} \cap \mathrm{FV}(\vec{\Sigma},\vec{t}) = \emptyset$.
Then $\llbracket\vec{P},\vec{x}/\vec{\Sigma},\vec{t}\rrbracket\llbracket\vec{Q},\vec{y}/\vec{\Gamma},\vec{s}\rrbracket\Sigma \equiv \llbracket\vec{Q},\vec{y}/\llbracket\vec{P},\vec{x}/\vec{\Sigma},\vec{t}\rrbracket(\vec{\Gamma},\vec{s})\rrbracket\llbracket\vec{P},\vec{x}/\vec{\Sigma},\vec{t}\rrbracket\Sigma$.

2.3.9. Lemma. Let Σ be closed and $\mathrm{Par}(\Sigma) \subseteq \{Q_1,\dots,Q_n\}$.
Then $\llbracket\vec{P},\vec{x}/\vec{\Sigma},\vec{t}\rrbracket\llbracket\vec{Q}/\vec{\Gamma}\rrbracket\Sigma \equiv \llbracket\vec{Q}/\llbracket\vec{P},\vec{x}/\vec{\Sigma},\vec{t}\rrbracket\vec{\Gamma}\rrbracket\Sigma$.

2.4. Formulas, language

Let s be a similarity type. Then the *language* $\mathcal{L}_s$ consists of the *formulas*: $\Sigma = \Gamma$ (equals), $\Sigma \geq \Gamma$ (reduces to), $\Sigma > \Gamma$ (properly reduces to), $\Sigma >_1 \Gamma$ (reduces in one step to) and $\Sigma \in \Gamma$ (has type or has supertype), where Σ and Γ are expressions in E_s.
If R is a relation symbol we write $\Sigma_1,\dots,\Sigma_n\ R\ \Gamma_1,\dots,\Gamma_n$ $(\vec{\Sigma}\ R\ \vec{\Gamma})$ for the sequence of formulas $\Sigma_1\ R\ \Gamma_1,\dots,\Sigma_n\ R\ \Gamma_n$.

§3. λλ-THEORIES

3.1. Base

According to what has been said in 1.1.4, the set of axioms and rules of a λλ-theory can be divided into two parts.

(i) A set, characterizing the underlying system (the same for any λλ-theory).

(ii) In addition the assignments of types and supertypes to the relevant constants, determined by a base (defined below).

The situation may be compared to e.g. predicate logic, where one adds for each particular theory a set of mathematical axioms to the fixed framework of logical axioms and rules.
Now recall the example in 1.1.4. It involves an instance (1) of the general assumption scheme:

(*) "Let P_1 be a Σ_1, let P_2 be a Σ_2, ... and P_n be a Σ_n. Then ... ."

In such a scheme it is assumed that the Σ_i's are well defined in the given context, which leads to the requirement that Σ_{i+1} should not contain free variables or parameters other than $P_1,\dots,P_i$. This observation motivates the following definition.

3.1.1. Definition. A *regular sequence of parameters* (rsop) is a sequence $P_1^{\Sigma_1},\dots,P_n^{\Sigma_n}$ of distinct parameters, where the Σ_i's are closed types or supertypes and for $0 \leq i < n$, $\mathrm{Par}(\Sigma_{i+1}) \subseteq \{P_1,\dots,P_i\}$.

We now proceed to the definition of a base. Notice the requirements on dates, motivated by the remark made in 2.1.

3.1.2. Definition. A base Ω is a triple $\langle s,\rho,\tau'\rangle$, where

(i) s is a similarity type $\langle S_0,S_1,\sigma\rangle$.

(ii) ρ is an effective function from $S_0 \cup S_1$ to rsop's, such that for all $i \in S_0 \cup S_1$, if $\rho(i) = P_1^{\Sigma_1},\ldots,P_n^{\Sigma_n}$, then $C_i(P_1^{\Sigma_1},\ldots,P_n^{\Sigma_n}) \in E_s$ and $\max(d(\Sigma_1),\ldots,d(\Sigma_n)) < i$.

(iii) τ' is an effective function from S_0 to closed types in E_s and from S_1 to closed supertypes in E_s, such that for $i \in S_0 \cup S_1$, if $\rho(i) = P_1,\ldots,P_n$, then $\mathrm{Par}(\tau'(i)) \subseteq \{P_1,\ldots,P_n\}$ and $d(\tau'(i)) < i$.

3.2. Axioms and rules of $\lambda\lambda[\Omega]$

3.2.1. Given a base Ω, the $\lambda\lambda$-theory $\lambda\lambda[\Omega]$ is formulated in the language $\mathcal{L}_s$. The axioms and rules of $\lambda\lambda[\Omega]$ are the following.

I type assignment.

a) $x^\alpha \in \alpha$; $P^\Sigma \in \Sigma$.

b) $C_i(\Sigma_1,\ldots,\Sigma_n) \in [P_1,\ldots,P_n/\Sigma_1,\ldots,\Sigma_n]\tau'(i)$ if $i \in S_0 \cup S_1$ and $\rho(i) = P_1,\ldots,P_n$.

c) $\alpha \in \underline{\text{type}}$ (type inclusion).

d) $\Sigma \in \Gamma \vdash [x,\alpha]\Sigma \in [x,\alpha]\Gamma$, provided $x \notin SV(\Sigma)$.

e) $\Sigma \in \Gamma \vdash \langle t\rangle\Sigma \in \langle t\rangle\Gamma$.

II one step reduction.

β-reduction: $t \in \alpha \vdash \langle t\rangle[x,\alpha]\Sigma >_1 [x/t]\Sigma$.

η-reduction: $f \in \beta$, $\beta \in [x,\alpha]\alpha^* \vdash [x,\alpha]\langle x\rangle f >_1 f$, provided $x \notin FV(f)$,

$\beta \in [x,\alpha]\alpha^* \vdash [x,\alpha]\langle x\rangle\beta >_1 \beta$, provided $x \notin FV(\beta)$.

monotonicity rules:

a) $\Sigma >_1 \Gamma \vdash C(\Sigma_1,\ldots,\Sigma,\ldots,\Sigma_n) >_1 C(\Sigma_1,\ldots,\Gamma,\ldots,\Sigma_n)$.

b) $\Sigma >_1 \Gamma \vdash \langle t\rangle\Sigma >_1 \langle t\rangle\Gamma$; $t >_1 s \vdash \langle t\rangle\Sigma >_1 \langle s\rangle\Sigma$.

c) $\Sigma >_1 \Gamma \vdash [x,\alpha]\Sigma >_1 [x,\alpha]\Gamma$, provided $x \notin SV(\Sigma)$.

d) $\alpha >_1 \beta \vdash [x,\alpha]\Sigma >_1 [y,\beta][x/y]\Sigma$, provided $y \notin FV(\Sigma)$.

III proper reduction, reduction and equality.

a) $\Sigma >_1 \Gamma \vdash \Sigma > \Gamma$; $\Sigma > \Gamma$, $\Gamma > \Lambda \vdash \Sigma > \Lambda$.

b) $\Sigma > \Gamma \vdash \Sigma \geq \Gamma$; $\Sigma \geq \Sigma$.

c) $\Sigma \geq \Gamma \vdash \Sigma = \Gamma$; $\Sigma = \Gamma \vdash \Gamma = \Sigma$; $\Sigma = \Gamma$, $\Gamma = \Lambda \vdash \Sigma = \Lambda$.

d) $\Sigma = \Gamma$, $\Lambda \in \Sigma \vdash \Lambda \in \Gamma$; $\Sigma = \Gamma$, $\Sigma \in \Lambda \vdash \Gamma \in \Lambda$.

3.2.2. **Remarks**

(i) Ic) amounts to the principle of type inclusion (cf. 1.1.3 and 3.5).

(ii) The motivation of the restriction in Id) is clear from following example. Suppose one had $[x,\alpha]y^{C(x)} \in [x,\alpha]C(x)$. Then for arbitrary $t \in \alpha$ by application and β-reduction $y^{C(x)} \in C(t)$, which is obviously not intended.

(iii) The restriction in IIc) excludes the possibility of both $\langle t\rangle[x,\alpha]\langle y^{C(x)}\rangle[z,C(x)]z >_1 \langle t\rangle[x,\alpha]y^{C(x)} >_1 y^{C(x)}$ and $\langle t\rangle[x,\alpha]\langle y^{C(x)}\rangle[z,C(x)]z >_1 \langle y^{C(x)}\rangle[z,C(t)]z$, both in nf, violating CR.

In the sequel we assume an arbitrary base Ω to be fixed. By just stating a formula we mean that it is derivable in $\lambda\lambda[\Omega]$, for convenience further referred to as $\lambda\lambda$.

Syntactical variables for expressions are supposed to range over E_s.

3.2.3. Lemma. The monotonicity rules IIa)-d) hold also with $>_1$ replaced by $>$, $\geq$ or $=$.

3.2.4. Now follows a, rather technical, definition, auxiliary to the important substitution lemma 3.2.5. Compare also definition 3.1.1 (rsop's).

Definition. Given an expression Σ, a sequence $P_1^{\Sigma_1},\dots,P_m^{\Sigma_m},x_1^{\alpha_1},\dots,x_n^{\alpha_n}/\Gamma_1,\dots,\Gamma_m,t_1,\dots,t_n$ is called a *regular substitution sequence* (rss) *for* Σ, if the following conditions are satisfied:

(i) $\Gamma_i \in [\![\vec{P}/\vec{\Gamma}]\!]\Sigma_i$ $(1 \leq i \leq m)$.

(ii) $t_i \in [\![\vec{P},\vec{x}/\vec{\Gamma},\vec{t}]\!]\alpha_i$ $(1 \leq i \leq n)$.

(iii) If $Q^{\Lambda} \in \mathrm{Par}(\Sigma) \setminus \{P_1,\dots,P_m\}$, then $\mathrm{Par}(\Lambda) \cap \{P_1,\dots,P_m\} = \emptyset$.

(iv) If $y^{\beta} \in \mathrm{FV}(\Sigma) \setminus \{x_1,\dots,x_n\}$, then $\mathrm{FV}(\beta) \cap \{x_1,\dots,x_n\} = \mathrm{Par}(\beta) \cap \{P_1,\dots,P_m\} = \emptyset$.

It is easily verified that the conditions (iii) and (iv) are fulfilled if in particular:

- $m = 0$ and $\{x_1,\dots,x_n\} \cap \mathrm{SV}(\Sigma) = \emptyset$, or
- $\mathrm{Par}(\Sigma) \subseteq \{P_1,\dots,P_m\}$ and $\mathrm{FV}(\Sigma) \subseteq \{x_1,\dots,x_n\}$, and hence if
- $\mathrm{Par}(\Sigma) \subseteq \{P_1,\dots,P_m\}$ and Σ is closed.

3.2.5. Lemma. Let $\vec{P},\vec{x}/\vec{\Sigma},\vec{t}$ be both an rss for Σ and for Γ, and let $\Sigma\ R\ \Gamma$, where R is $>_1$, $>$, $\geq$, $=$ or $\in$. Then also $[\![\vec{P},\vec{x}/\vec{\Sigma},\vec{t}]\!]\Sigma\ R\ [\![\vec{P},\vec{x}/\vec{\Sigma},\vec{t}]\!]\Gamma$.

Proof. Simultaneous induction on the length of deduction of $\Sigma\ R\ \Gamma$.

3.3. Canonical type assignment, uniqueness of types

The assignment function τ' generates a function τ, which assigns canonically to each object a type and to each type a supertype, such that always $\Sigma \in \tau(\Sigma)$.

3.3.1. Definition. $\tau(\Sigma)$ is defined by induction on $\ell(\Sigma)$.
$\tau(x^{\alpha}) \equiv \alpha$; $\tau(P^{\Gamma}) \equiv \Gamma$.
$\tau(C_i(\Sigma_1,\ldots,\Sigma_n)) \equiv \llbracket\vec{P}/\vec{\Sigma}\rrbracket\tau'(i)$ for $i \in S_0 \cup S_1$, where $\rho(i) = P_1,\ldots,P_n$.
$\tau(\langle t\rangle\Gamma) \equiv \langle t\rangle\tau(\Gamma)$; $\tau([x,\alpha]\Gamma) \equiv [x,\alpha]\tau(\Gamma)$, where x is chosen such that $x \notin SV(\Gamma)$.

3.3.2. Lemma. $\Sigma \varepsilon \tau(\Sigma)$ holds for any object or type Σ.
Proof. Induction on $\ell(\Sigma)$.

3.3.3. Lemma. $\llbracket\vec{x}/\vec{t}\rrbracket\tau(C(\vec{\Sigma})) \equiv \tau(C(\llbracket\vec{x}/\vec{t}\rrbracket\vec{\Sigma}))$.
Proof. Immediate by lemma 2.3.9 and the definition of τ.

3.3.4. Lemma. Let $\vec{x}/\vec{t}$ be an rss for Σ, then $\tau(\llbracket\vec{x}/\vec{t}\rrbracket\Sigma) = \llbracket\vec{x}/\vec{t}\rrbracket\tau(\Sigma)$.
Proof. Induction on $\ell(\Sigma)$. Use lemma 3.3.3 in case Σ is a constant.

3.3.5. Theorem (uniqueness of types). $t \varepsilon \alpha \leftrightarrow \alpha = \tau(t)$.

Proof. One side is implied by lemma 3.3.2. For the other side, prove by simultaneous induction on the length of deduction of $t \varepsilon \alpha$ and $t = s$, respectively, the two statements $t \varepsilon \alpha \Rightarrow \alpha = \tau(t)$ and $t = s \Rightarrow \tau(t) = \tau(s)$. The proof makes use of the previous lemma.

3.3.6. Remark. The analogous result for supertypes does not hold (cf. 3.5). However, in $\lambda\lambda$ without rule Ic) one would obtain theorem 3.3.5 for supertypes as well.

3.4. Legitimacy

In this section we define the set L of legitimate expressions. Then the legitimate fragment $\lambda\lambda - \ell$ of $\lambda\lambda$ is the theory obtained by restricting the axioms and rules of $\lambda\lambda$, to use only expressions from L.

3.4.1. Remark that L depends on the choice of Ω. We might call Ω a *legitimate base* if $\{C_i(\rho(i)) \mid i \in S_0 \cup S_1\} \cup \{\tau'(i) \mid i \in S_0 \cup S_1\} \subseteq L$.

3.4.2. For the sake of the characterization of the legitimate expressions we now introduce a function τ^*, assigning canonically to each expression a supertype.

Definition.
$\tau^*(\alpha^*) \equiv \alpha^*$ for supertypes α^*.
$\tau^*(\alpha) \equiv \tau(\alpha)$ for types α.
$\tau^*(t) \equiv \tau(\tau(t))$ for objects t.

Remark. τ^* may be compared to Typ^* in Nederpelt [8].

3.4.3. Definition. The set L of *legitimate expressions* is specified by defining by induction on $(d(\Sigma),c(\Sigma))$ (i.e. $\omega.d(\Sigma) + c(\Sigma)$, cf. 5.2), what it means for an expression Σ to be *legitimate*.

$x^{\alpha} \in L$ iff $\alpha \in L$; $P^{\Sigma} \in L$ iff $\Sigma \in L$.

$C_i(\vec{\Sigma}) \in L$ iff $\Sigma_1,\ldots,\Sigma_n$, $\tau'(i) \in L$ and $\rho(i) / \vec{\Sigma}$ is an rss for $\tau'(i)$.

$<t>\Gamma \in L$ iff $t,\Gamma \in L$ and for some α, α^*: $t \varepsilon \alpha$ and $\tau^*(\Gamma) = [x,\alpha]\alpha^*$.

$[x,\alpha]\Gamma \in L$ iff $\alpha,\Gamma \in L$, provided $x \notin SV(\Gamma)$.

3.4.4. Lemma. Let $\vec{P},\vec{x}/\vec{\Gamma},\vec{t}$ be an rss for $\Sigma_1,\ldots,\Sigma_n$, respectively, and let $\vec{Q}/\vec{\Sigma}$ be an rss for the closed expression Σ. Then also $\vec{Q}/[\vec{P},\vec{x}/\vec{\Gamma},\vec{t}]\vec{\Sigma}$ is an rss for Σ.

Proof. Apply lemma 3.2.5.

3.4.5. Lemma. Let $\Sigma,\Gamma_1,\ldots,\Gamma_m,t_1,\ldots,t_n \in L$ and let $\vec{P},\vec{x}/\vec{\Gamma},\vec{t}$ be an rss for Σ, then $[\vec{P},\vec{x}/\vec{\Gamma},\vec{t}]\Sigma \in L$.

Proof. Induction on $\ell(\Sigma)$. Use lemmas 3.2.5 and 3.4.4.

3.4.6. Theorem (Extended Closure). Let $\Sigma \in L$ and let either $\Sigma \geq \Gamma$, or $\Sigma \not> \Gamma$ or $\tau(\Sigma) \equiv \Gamma$. Then also $\Gamma \in L$ (i.e. $\Sigma \in L$ and $\Sigma \rightarrow \Gamma \Rightarrow \Gamma \in L$).

3.5. Type inclusion, uniqueness of domains

The analogue of the uniqueness of types theorem for supertypes does not hold. E.g. we have both $[x,\alpha]\beta \varepsilon [x,\alpha]$type and $[x,\alpha]\beta \varepsilon$ type (cf. 1.1.3). However, one does obtain a weaker result, viz. uniqueness of domains:

$$\alpha \varepsilon [x,\beta]\beta^* \text{ and } \alpha \varepsilon [x,\gamma]\gamma^* \Rightarrow \beta = \gamma .$$

This property is important as a justification for the above characterization of legitimate expressions.

We state here without proof:

3.5.1. Theorem. $\alpha \in [x,\beta]\beta^*$ iff for some supertype λ^*, $\tau(\alpha) = [x,\beta]\lambda^*$.

In order to say something more on the structure of supertypes in $\lambda\lambda - \ell$, we define the relation $\leq$ of *type inclusion* between supertypes in L.

3.5.2. Definition. First define the relation $\subset$ between supertypes in L inductively by

(i) $\alpha^* \subset$ type for any supertype α^*.

(ii) If $\alpha^* \subset \beta^*$, then also $[x,\alpha]\alpha^* \subset [x,\alpha]\beta^*$ and $<t>\alpha^* \subset <t>\beta^*$.

Then $\leq$ is the smallest transitive relation in L extending $\equiv$ and $\subset$.

3.5.3. Theorem. Let $\alpha,\beta,\alpha^*,\beta^* \in L$. Then

(i) $\alpha \,\varepsilon\, \alpha^*$ and $\alpha \,\varepsilon\, \beta^* \Rightarrow \alpha^* \leq \beta^*$ or $\beta^* \leq \alpha^*$.

(ii) $\alpha \,\varepsilon\, \alpha^* \Rightarrow \tau(\alpha) \leq \alpha^*$.

So τ assigns to a legitimate type its minimal legitimate supertype. Note that a supertype in L, which is in nf, is always of the form $[x_1,\alpha_1] \ldots [x_k,\alpha_k]\underline{type}$.

§4. DECIDABILITY AND CONSERVATIVITY

4.1. Sequences, trees

We use $\sigma,\rho,\ldots$ to range over, finite or infinite, sequences of expressions. We define $\ell h(\sigma)$ to be the length of σ if σ is finite, $\ell h(\sigma) = \infty$ if σ is infinite. Σ will also stand for the sequence of length one, consisting of Σ only. If $\ell h(\sigma) < \infty$, then σ,ρ stands for the concatenation of σ and ρ. We define: $\sigma < \rho$ (ρ extends σ) iff there exists a sequence τ, such that $\sigma,\tau = \rho$.

4.1.1. Definition. A sequence $\Sigma_0,\Sigma_1,\ldots$ is called a

(i) *reduction sequence* of Σ_0 iff $\Sigma_i >_1 \Sigma_{i+1}$,

(ii) *rs-sequence* of Σ_0 iff either $\Sigma_i >_1 \Sigma_{i+1}$ or $\Sigma_i \succ \Sigma_{i+1}$,

(iii) *$\rightarrow$-sequence* of Σ_0 iff either $\Sigma_i >_1 \Sigma_{i+1}$ or $\Sigma_i \succ \Sigma_{i+1}$ or $\tau(\Sigma_i) \equiv \Sigma_{i+1}$.

4.1.2. Definition. The finite reduction sequences of a term Σ form under the partial order $<$ a tree, the *reduction tree* of Σ. Analogously we have the *rs-tree* and the $\rightarrow$-tree of Σ. The latter is called the *big tree* of Σ. The set of $\rightarrow$-sequences of Σ is denoted by $S(\Sigma)$. $B(\Sigma) = \{\Gamma \mid \Sigma \rightarrow \Gamma\}$.

4.1.3. Definition. $h(\Sigma)$ will be the *height* of the reduction tree of Σ:

$h(\Sigma) = \max(\{\ell h(\sigma) \mid \sigma \text{ is a reduction sequence of } \Sigma\})$.

Analogously, $b(\Sigma) = \max(\{\ell h(\sigma) \mid \sigma \in S(\Sigma)\})$ is the height of the big tree of Σ.

4.2. Normal forms, strong normalization

An expression Σ is in *normal form* (nf) if there does not exist an expression Γ such that $\Sigma >_1 \Gamma$.

An expression Σ is called *strongly normalizable* if $h(\Sigma) < \infty$, i.e., if the reduction tree of Σ is well founded.

4.3. Results

We now state the main results of the paper. The details of proofs are generally omitted. However, section 5 will be devoted to sketching the proof of BT (theorem 4.3.2).

4.3.1. Theorem (CR). If $\Sigma = \Gamma$, then there exists an expression Λ, such that $\Sigma \geq \Lambda$ and $\Gamma \geq \Lambda$.

A proof shall not be given here. Let it suffice to remark that in $\lambda\lambda$ without the rule of η-reduction the property follows easily from the strong normalizability of $\lambda\lambda$. In the present situation, where η-reduction is included, the proof is more complicated. It was proved by Van Daalen (cf. 1.4).

4.3.2. Theorem (BT). For every expression Σ, $b(\Sigma) < \infty$. I.e., big trees in $\lambda\lambda$ are well founded.

This result implies that every expression is strongly normalizable (SN). Moreover, by CR one obtains that for each Σ, there exists a unique nf Γ, such that $\Sigma = \Gamma$. (In contrast to its use in "uniqueness of types", uniqueness is here to be understood with respect to $\equiv$.) This unique expression will be denoted by $nf(\Sigma)$.

4.3.3. Corollary. Given an expression Σ, its big tree can be effectively constructed. Proof. Given the big trees of an object t and a type α, one can decide if $t \in \alpha$; viz. by merely checking if $nf(\tau(t)) \equiv nf(\alpha)$. By this observation it is easy to devise an algorithm, which, when applied to an expression Σ, constructs the big tree of Σ, and which can be proved to be correct by induction on $b(\Sigma)$.

4.3.4. Corollary. $\lambda\lambda$ is decidable.

4.3.5. Let $(\Sigma,\Gamma) \vdash \Lambda \; R \; \Lambda'$ assert the existence of a deduction of $\Lambda \; R \; \Lambda'$ in $\lambda\lambda$, in which occur only expressions from $B(\Sigma) \cup B(\Gamma)$.

Lemma (transitivity). If $\Sigma',\Gamma' \in B(\Sigma) \cup B(\Gamma)$ and $(\Sigma',\Gamma') \vdash \Lambda = \Lambda'$, then $(\Sigma,\Gamma) \vdash \Lambda = \Lambda'$.

4.3.6. Definition. A new measure $n(\Gamma)$ is defined by induction on $b(\Gamma)$:

$$n(\Gamma) = \Big(\sum_{(\sigma,\Lambda)\in S'(\Gamma)} n(\Lambda)\Big) + 1, \text{ where } S'(\Gamma) = \{\rho \in S(\Gamma) \mid \ell h(\rho) > 1\}.$$

4.3.7. Theorem. Let $\Sigma \; R \; \Gamma$, where R is $=$, $\geq$, $>$, $>_1$, or ε.
Then $(\Sigma,\Gamma) \vdash \Sigma \; R \; \Gamma$.

Proof. Induction on $n(\Sigma) + n(\Gamma)$. Let us restrict attention to equalities. If Σ and Γ are both in nf, then by CR, $\Sigma \equiv \Gamma$ and we are done. So assume that $\Sigma >_1 \Sigma'$.

Then by the induction hypothesis and transitivity, $(\Sigma,\Gamma) \vdash \Sigma' = \Gamma$. Hence it is enough to show that $(\Sigma,\Gamma) \vdash \Sigma = \Sigma'$. Now distinguish cases as to the last rule applied in a deduction of $\Sigma >_1 \Sigma'$. We treat only one case.
Let $\Sigma \equiv [x,\alpha]<x>f >_1 f \equiv \Sigma'$ and $\tau^*(f) = [x,\alpha]\alpha^*$. It must be shown that $(\Sigma,\Gamma) \vdash \tau^*(f) = [x,\alpha]\beta^*$ for some β^* (cf. the rule of η-reduction and theorem 3.5.1). By CR, $\tau^*(f)$ and $[x,\alpha]\alpha^*$ have a common reduct $[y,\gamma]\gamma^*$. Now $n(\alpha) + n(\gamma) < n(\Sigma)$ and $n(\tau^*(f)) + n([y,\gamma]\gamma^*) < n(\Sigma)$ imply that $(\Sigma,\Gamma) \vdash \alpha = \gamma$ and $(\Sigma,\Gamma) \vdash \tau^*(f) = [y,\gamma]\gamma^*$, respectively, and consequently $(\Sigma,\Gamma) \vdash \tau^*(f) =$ $= [x,\alpha]\gamma^*$.

4.3.8. Corollary. λλ is a conservative extension of λλ - ℓ.
Proof. By theorem 4.3.7 and the closure theorem 3.4.6.

§5. PROOF OF THE BIG TREE THEOREM

The strategy of the proof of BT (theorem 4.3.2) will be to define an extension λλ - *p* of λλ, by adding an extra rule of term formation for ordered pairs: if $\tau(\Sigma) = \Gamma$, then $\ulcorner\Sigma,\Gamma\urcorner$ is an expression. A pair $\ulcorner\Sigma,\Gamma\urcorner$ may be considered as just a copy of Σ, Γ being present only for bookkeeping reasons. The reduction relation is extended to include the projections $\ulcorner\Sigma,\Gamma\urcorner >_1 \Sigma$ and $\ulcorner\Sigma,\Gamma\urcorner >_1 \Gamma$. Strong normalization of expressions in λλ - *p* is proved by using a computability argument. Subsequently a map φ is defined, embedding λλ in λλ - *p* such that →-sequences in λλ give rise to longer rs-sequences in λλ - *p*. Termination of rs-sequences is an easy corollary of SN. Hence we may conclude that →-sequences in λλ do terminate.

5.1. Introduction of λλ - *p*

The base Ω, which was fixed under 3.2.2, is still assumed here. So λλ - *p* will be in fact an extension of λλ[Ω].
The definition of the set E - *p* of expressions of λλ - *p* involves a "forget function" p from expressions of λλ - *p* to expressions of λλ, consistently deleting the second coordinates of pairs. (Hence p acts as the identity on expressions of λλ.) The next two definitions should be taken as simultaneously defining the set E - *p* and the function p.

5.1.1. Definition. For the definition of E - *p* take clauses (i) to (iv) of the inductive definition of E (2.3.1.) and add a fifth clause:

(v) If Σ and Γ are in E - *p* and $\tau(p(\Sigma)) = p(\Gamma)$ is deducible in λλ, then $\ulcorner\Sigma,\Gamma\urcorner$ is an object if Σ is an object and a type if Σ is a type, reespectively.

5.1.2. Definition. The function $p\colon E - p \to E$ is defined inductively.

$p(\underline{type}) \equiv \underline{type}$; $p(P^{\Sigma}) \equiv P^{p(\Sigma)}$; $p(x^{\alpha}) \equiv x^{p(\alpha)}$;

$P(C(\Sigma_1,\ldots,\Sigma_n)) \equiv C(p(\Sigma_1),\ldots,p(\Sigma_n))$.

$p(\langle t\rangle\Sigma) \equiv \langle p(t)\rangle p(\Sigma)$; $p([x,\alpha]\Sigma) \equiv [x,p(\alpha)]p(\Sigma)$.

$p(\ulcorner\Sigma,\Gamma\urcorner) \equiv p(\Sigma)$.

5.1.3. The definitions, notations and conventions from section 2.3. are generalized to $E - p$. In particular, $\ell(\ulcorner\Sigma,\Gamma\urcorner) = \max(\ell(\Sigma),\ell(\Gamma)) + 1$; $d(\ulcorner\Sigma,\Gamma\urcorner) = \max(d(\Sigma),d(\Gamma))$; $\mathrm{Par}(\ulcorner\Sigma,\Gamma\urcorner) = \mathrm{Par}(\Sigma) \cup \mathrm{Par}(\Gamma)$; $\mathrm{FV}(\ulcorner\Sigma,\Gamma\urcorner) = \mathrm{FV}(\Sigma) \cup \mathrm{FV}(\Gamma)$; $\ulcorner\Sigma,\Gamma\urcorner \rhd \Sigma$, $\ulcorner\Sigma,\Gamma\urcorner \rhd \Gamma$. $[\![\vec{P},\vec{x}/\vec{\Lambda},\vec{t}]\!]\ulcorner\Sigma,\Gamma\urcorner \equiv \ulcorner[\![\vec{P},\vec{x}/\vec{\Lambda},\vec{t}]\!]\Sigma\,,\,[\![\vec{P},\vec{x}/\vec{\Lambda},\vec{t}]\!]\Gamma\urcorner$. Substitution is only admitted if the substitution result is in $E - p$ again, i.e., if the substitution does not violate the restriction in 5.1.1 (v). A sufficient condition for this requirement is given in 5.1.6 below.

5.1.4. The formulas of $\lambda\lambda - p$ are defined as in 2.4.

5.1.5. The axioms and rules of $\lambda\lambda - p$ are those of $\lambda\lambda$ (cf. 3.2.1) and additionally

II projection: $\ulcorner\Sigma,\Gamma\urcorner >_1 \Sigma$; $\ulcorner\Sigma,\Gamma\urcorner >_1 \Gamma$.

e) $\Sigma >_1 \Lambda \vdash \ulcorner\Sigma,\Gamma\urcorner >_1 \ulcorner\Lambda,\Gamma\urcorner$; $\Gamma >_1 \Lambda \vdash \ulcorner\Sigma,\Gamma\urcorner >_1 \ulcorner\Sigma,\Lambda\urcorner$.

Remark that now, by projection, an expression may reduce to an expression of a different sort, i.e. an object to a type and a type to a supertype, respectively. For that reason a few obvious restrictions are to be made in some of the rules. In IIa) and IIId) we require Σ and Γ to be of the same sort. In IIb), t and s have to be both objects; in IId), α and β have to be both types.

5.1.6. The definitions and results of sections 3.2 and 3.3 are generalized to $\lambda\lambda - p$. Remark in particular that by lemma 3.2.5 we obtain: If $\vec{P},\vec{x}/\vec{\Gamma},\vec{t}$ is an rss for Σ in $E - p$, then $[\![\vec{P},\vec{x}/\vec{\Gamma},\vec{t}]\!]\Sigma$ is in $E - p$ again, and hence an admitted substitution. Add to definition 3.3.1 the clause: $\tau(\ulcorner\Sigma,\Gamma\urcorner) \equiv \tau(\Sigma)$.

5.2. Norms

The proof of SN for $\lambda\lambda - p$ is essentially based on the method of proof orginated by Tait [11], and used e.g. by Prawitz [9, Appendix A] for a system of natural deduction. The key notion of this method, computability (alternative terminologies: convertability, validity, réductibilité), could be defined by induction on the length of type in [11] and on the length of the end formula of a deduction in [9]. Here it is essential that the type of a term and the end formula of a deduction do not change under reduction of the term and the deduction, respectively. In our proof their task will be fulfilled by a norm on expressions $\gamma(\Sigma)$. Auxiliary to its definition we first introduce the measure $m(\Sigma)$.

<u>Note</u>. Pairs of natural numbers are supposed to be ordered lexicographically.

5.2.1. <u>Definition</u>. $m(\Sigma)$ is defined by induction on $(d(\Sigma),c(\Sigma))$.
$m(\underline{type}) = 0$; $m(P^\Gamma) = m(\Gamma) + 1$; $m(x^\alpha) = m(\alpha) + 1$;
$m(C_i(\Sigma_1,\dots,\Sigma_n)) = \max(m(\Sigma_1),\dots,m(\Sigma_n)) + m(\tau'(i)) + 1$;
$m(\langle t\rangle\Gamma) = \max(m(t),m(\Gamma))$; $m([x,\alpha]\Gamma) = \max(m(\alpha),m(\Gamma))$ and
$m(\ulcorner\Gamma,\Lambda\urcorner) = \max(m(\Gamma),m(\Lambda))$.

5.2.2. <u>Lemma</u>.
(i) If Σ is an atomic expression (not $\underline{type}$), then $m(\tau(\Sigma)) < m(\Sigma)$.
(ii) For all objects and types Σ, $m(\tau(\Sigma)) \le m(\Sigma)$.
(iii) If $\Sigma \geq\!\!\!> \Gamma$, then $m(\Gamma) \le m(\Sigma)$.

5.2.3. The norm $\gamma(\Sigma)$ is going to be a, possibly empty, string of the brackets [and]. Let G,H,... range over such strings. They are well ordered by <: G < H iff the number of brackets in G is less then the number of brackets in H. λ denotes the empty string.

<u>Definition</u>. $\gamma(\Sigma)$ is defined by induction on $(m(\Sigma),\ell(\Sigma))$.
$\gamma(\underline{type}) = \lambda$; $\gamma(\Sigma) = \gamma(\tau(\Sigma))$ for other atomic Σ's;
$\gamma([x,\alpha]\Gamma) = [\gamma(\alpha)]\gamma(\Gamma)$; $\gamma(\ulcorner\Gamma,\Lambda\urcorner) = \gamma(\Gamma)$;

$$\gamma(\langle t\rangle\Gamma) = \begin{cases} G \text{ if } \gamma(\Gamma) = [\gamma(t)]G, \\ \lambda \text{ otherwise.} \end{cases}$$

5.2.4. <u>Lemma</u>.
If $\gamma(t_i) = \gamma(\alpha_i)$ $(1 \le i \le n)$, then $\gamma([x_1^{\alpha_1},\dots,x_n^{\alpha_n}/t_1,\dots,t_n]\Sigma = \gamma(\Sigma)$.

5.2.5. <u>Lemma</u>.
(i) If $t \in \alpha$, then $\gamma(t) = \gamma(\alpha)$.
(ii) If $\Sigma = \Gamma$, then $\gamma(\Sigma) = \gamma(\Gamma)$.
Proof. Prove (i) and (ii) simultaneously by induction on the length of deduction in $\lambda\lambda - p$. Use lemma 5.2.4.

5.3. Computability

The notion of computability can now be defined by induction on $\gamma(\Sigma)$.

5.3.1. <u>Definition</u>. An expression Σ is *computable* if both

(i) Σ is strongly normalizable;
(ii) whenever $\Sigma \ge [x,\alpha]\Gamma$ and $t \in \alpha$ and t is comp, then also $[x/t]\Gamma$ is comp.

The definition is correct. For if $\Sigma \geq [x,\alpha]\Gamma$ and $t \in \alpha$, then $\gamma(t) = \gamma(\alpha) < [\gamma(\alpha)]\gamma(\Gamma) = \gamma(\Sigma)$ and $\gamma([x/t]\Gamma) = \gamma(\Gamma) < \gamma(\Sigma)$.

5.3.2. Lemma.

(i) If $\Sigma \geq \Gamma$ and Σ is comp, then so is Γ.

(ii) Let Σ not have the form $[x,\alpha]\Gamma$. Then Σ is comp iff all Σ_1 such that $\Sigma >_1 \Sigma_1$ are comp.

Proof. Immediate by inspection of the definition.

5.3.3. Lemma. If $\Sigma_1,\ldots,\Sigma_n$ are comp, then so is $C(\vec{\Sigma})$.

Proof. Induction on $h(\Sigma_1) + \ldots + h(\Sigma_n)$.

5.3.4. Lemma. If both Σ and t are comp, then so is $<t>\Sigma$.

Proof. Induction on $h(\Sigma) + h(t)$. Assuming that $<t>\Sigma >_1 \Gamma$, prove that Γ is comp, and apply lemma 5.3.2 (ii). Distinguish two cases:

(i) Either $t >_1 t_1$ and $\Gamma \equiv <t_1>\Sigma$ or $\Sigma >_1 \Sigma_1$ and $\Gamma \equiv <t>\Sigma_1$. Then Γ is comp by the induction hypothesis.

(ii) $\Sigma \equiv [x,\alpha]\Lambda$ and $\Gamma \equiv [x/t]\Lambda$. Then Γ is comp by clause (ii) of the computability definition 5.3.1.

5.3.5. Lemma. If Σ and Γ are comp, then so is $\ulcorner\Sigma,\Gamma\urcorner$.

Proof. Again prove by induction on $h(\Sigma) + h(\Gamma)$, that $\ulcorner\Sigma,\Gamma\urcorner >_1 \Lambda$ implies that Λ is comp.

5.3.6. Definition. Σ is called *computable under substitution*, (cus) if for all comp expressions $t_1,\ldots,t_n$ and variables $x_1,\ldots,x_n$, such that $\vec{x}/\vec{t}$ is an rss for Σ, $[\vec{x}/\vec{t}]\Sigma$ is comp.

5.3.7. Theorem. All expressions Σ of $E - p$ are cus.

Proof. Induction on $\ell(\Sigma)$. Let $\vec{x}/\vec{t}$ be an arbitrary rss for Σ, such that $t_1,\ldots,t_n$ are comp. Throughout the proof we abbreviate $\Lambda' \equiv [\vec{x}/\vec{t}]\Lambda$.

The only case which is not immediate by the lemmas 5.3.3-5 and the induction hypothesis is $\Sigma \equiv [x,\alpha]\Gamma$, $\Sigma' \equiv [x,\alpha']\Gamma'$, where α' and Γ' are comp by the induction hypothesis. We check (i) and (ii) of definition 5.3.1.

(i) Suppose $\sigma = \Sigma_0,\Sigma_1,\ldots$ is a nonterminating reduction sequence of Σ'. Distinguish two cases:

a) There exist finite reduction sequences $\sigma_0,[x,\alpha_1]<x>f_0$ of Σ', and σ_1,α_1 of α', and $\sigma_2,<x>f_0$ of Γ', such that $\sigma = \sigma_0,[x,\alpha_1]<x>f_0,f_0,f_1,\ldots$. Then $\sigma_2,<x>f_0,<x>f_1,\ldots$ would be a nonterminating reduction sequence of Γ', contradicting the computability of Γ'.

b) Case a) does not apply, i.e., no outer η-reductions are performed in σ. Then σ would induce reduction sequences σ_0 of α' and σ_1 of Γ', such that either σ_0 or σ_1 or both are nonterminating, contradicting the fact that α' and Γ' are both comp.

(ii) Suppose $\Sigma' \geq [x,\alpha_1]\Gamma_1$. Again distinguish two cases:

a) $\alpha' \geq \alpha_2$, $\Gamma' \geq \langle x\rangle f$, $x \notin FV(f)$, and so $\Sigma' \geq [x,\alpha_2]\langle x\rangle f >_1 f$ and $f \geq [x,\alpha_1]\Gamma_1$. Let $t \in \alpha_1$ be comp. Then $[\![x/t]\!]\Gamma' \geq [\![x/t]\!](\langle x\rangle f) \equiv \langle t\rangle f \geq \langle t\rangle[x,\alpha_1]\Gamma_1 >_1 [\![x/t]\!]\Gamma_1$. Further $\vec{x},x/\vec{t},t$ is an rss for Γ and $[\![x/t]\!]\Gamma' \equiv [\![\vec{x},x/\vec{t},t]\!]\Gamma$. Hence by the induction hypothesis $[\![x/t]\!]\Gamma'$ is comp and by lemma 5.3.2 (i) so is $[\![x/t]\!]\Gamma_1$.

b) Case a) does not apply. Then $\alpha' \geq \alpha_1$ and $\Gamma' \geq [\![x^{\alpha_1}/x^{\alpha'}]\!]\Gamma_1$. Hence, if $t \in \alpha_1$, also $[\![x/t]\!]\Gamma' \geq [\![x/t]\!]\Gamma_1$ and repeating the argument in a) we find that for comp $t \in \alpha_1$, $[\![x/t]\!]\Gamma_1$ is comp.

5.3.8. Corollary. All expressions of $E-p$ are strongly normalizable.

5.3.9. Corollary. If Σ is an expression in $E-p$, then every rs-sequence of Σ terminates.

Proof. Induction on $(h(\Sigma),\ell(\Sigma))$, observing that if $\Sigma \succ \Gamma$, then $h(\Gamma) \leq h(\Sigma)$.

5.4. Embedding $\lambda\lambda$ in $\lambda\lambda - p$.

We now define a map $\varphi\colon E \to E-p$, such that to each $\to$-sequence of an expression Σ in $\lambda\lambda$ corresponds a longer rs-sequence of $\varphi(\Sigma)$ in $\lambda\lambda - p$. Then corollary 5.3.9 guarantees the well foundedness of big trees in $\lambda\lambda$.

5.4.1. Definition. $\varphi(\Sigma)$ is defined by induction on $(m(\Sigma),\ell(\Sigma))$.

$\varphi(x^\alpha) \equiv \ulcorner x^\alpha,\varphi(\alpha)\urcorner$; $\varphi(P^\Sigma) \equiv \ulcorner P^\Sigma,\varphi(\Sigma)\urcorner$;

$\varphi(C(\Sigma_1,\dots,\Sigma_n)) \equiv \ulcorner C(\varphi(\Sigma_1),\dots,\varphi(\Sigma_n)),\varphi(\tau(C(\vec{\Sigma})))\urcorner$;

$\varphi(\langle t\rangle\Gamma) \equiv \langle\varphi(t)\rangle\varphi(\Gamma)$ and $\varphi([x,\alpha]\Gamma) \equiv [y,\varphi(\alpha)][\![x/y]\!]\varphi(\Gamma)$.

5.4.2. Lemma. If $\Sigma \in E$, then $\Sigma = \varphi(\Sigma)$ (in $\lambda\lambda - p$).

Proof. By induction on $\ell(\Sigma)$, check that $\varphi(\Sigma) \geq \Sigma$.

5.4.3. Corollary. If $\Sigma \in \Gamma$ in $\lambda\lambda$, then $\varphi(\Sigma) \in \varphi(\Gamma)$.

5.4.4. Lemma. If $t \in \alpha$ in $\lambda\lambda$, $\Sigma \in E$, then $[\![x^\alpha/\varphi(t)]\!]\varphi(\Sigma) \geq \varphi([\![x/t]\!]\Sigma)$.

Proof. Induction on $(m(\Sigma),\ell(\Sigma))$. We show only three cases.

(i) $[\![x/\varphi(t)]\!]\varphi(x) \equiv [\![x/\varphi(t)]\!](\ulcorner x,\varphi(\alpha)\urcorner) \equiv \ulcorner\varphi(t),\varphi(\alpha)\urcorner >_1 \varphi(t)$.

(ii) $[x/\varphi(t)]\ \varphi(C(\vec{\Gamma})) \equiv \ulcorner C([x/\varphi(t)]\varphi(\vec{\Gamma})),[x/\varphi(t)]\varphi(\tau(C(\vec{\Gamma})))\urcorner \geq$
$\geq \ulcorner C(\varphi([x/t]\vec{\Gamma})),\varphi([x/t]\tau(C(\vec{\Gamma}))\urcorner \equiv \varphi([x/t]C(\vec{\Gamma}))$. Here we applied the induction hypothesis on $\Gamma_1,\ldots,\Gamma_n$ and $\tau(C(\vec{\Gamma}))$ and we used lemma 3.3.3.

(iii) $[x/\varphi(t)]\varphi([y,\beta]\Gamma) \equiv [z,[x/\varphi(t)]\varphi(\beta)][y/z][x/\varphi(t)]\varphi(\Gamma) \geq$
$\geq [u,\varphi([x/t]\beta)][y/u]\varphi([x/t]\Gamma) \equiv \varphi([x/t][y,\beta]\Gamma)$. (Apply induction hypothesis on β and Γ.)

5.4.5. Lemma. If $\Sigma >_1 \Gamma$ in $\lambda\lambda$, then $\varphi(\Sigma) > \varphi(\Gamma)$.

Proof. Induction on the length of deduction of $\Sigma >_1 \Gamma$. We show only one case. Let $\Sigma \equiv \langle t\rangle[x,\alpha]\Lambda >_1 [x/t]\Lambda \equiv \Gamma$ and $t \in \alpha$ (β-reduction). Then
$\varphi(\Sigma) \equiv \langle\varphi(t)\rangle[y,\varphi(\alpha)][x/y]\varphi(\Lambda) >_1 [x/\varphi(t)]\varphi(\Lambda) \geq \varphi(\Gamma)$, by lemmas 5.4.3 and 5.4.4.

5.4.6. Lemma. If $\Sigma \in E$, then $\varphi(\Sigma) > \varphi(\tau(\Sigma))$ (Σ either object or type).

Proof. Induction on $\ell(\Sigma)$. Two examples are:

(i) $\varphi(x^\alpha) \equiv \ulcorner x^\alpha,\varphi(\alpha)\urcorner >_1 \varphi(\alpha) \equiv \varphi(\tau(x^\alpha))$;

(ii) $\varphi(\langle t\rangle\Gamma) \equiv \langle\varphi(t)\rangle\varphi(\Gamma) > \langle\varphi(t)\rangle\varphi(\tau(\Gamma)) \equiv \varphi(\tau(\langle t\rangle\Gamma))$, by the induction hypothesis for Γ.

5.4.7. Lemma. If $\Sigma \rightarrow \Gamma$ in $\lambda\lambda$, then $\varphi(\Sigma) \rightarrow \varphi(\Gamma)$ in $\lambda\lambda - p$.

5.4.8. Corollary. If $\Sigma_0,\ldots,\Sigma_n$ is a →-sequence in $\lambda\lambda$, then there exists an rs-sequence from $\varphi(\Sigma_0)$ to $\varphi(\Sigma_n)$ in $\lambda\lambda - p$ of equal or greater length.
Proof. Induction on n, using the lemmas 5.4.5-7.

5.4.9. Theorem. If $\Sigma \in E$, then every →-sequence of Σ terminates.
Proof. Immediate from the corollaries 5.3.9. and 5.4.8.

References to chapter 5

[1] Benthem Jutting, L.S. van, On normal forms in AUTOMATH; Unpublished, 1971.

[2] Benthem Jutting, L.S. van, Checking Landau's "Grundlagen" in the AUTOMATH system; Mathematical Centre Tracts 83, Amsterdam 1979.

[3] Bruijn, N.G. de, The mathematical language AUTOMATH, its usage, and some of its extensions; Symposium on Automatic Demonstration (Versailles, December 1968), Springer Lecture Notes in Mathematics, Vol. 125 (1970), 29-61.

[4] Daalen, D.T. van, A description of AUTOMATH and some aspects of its language theory; proceedings of the Symposium APLASM (Orsay, December 1973), ed. P. Brafford (1975). Reprinted as appendix 1 in [2].

[5] Hindley, J.R., B. Lercher and J.P. Seldin, Introduction to Combinatory Logic, Cambridge University Press (1972).

[6] Howard, W.A., The formulae-as-types notion of construction; Unpublished (1969).

[7] Martin-Löf, P., An intuitionistic theory of types; Unpublished (1972).

[8] Nederpelt, R.P., Strong normalization in a typed lambda calculus with lambda structured types; Doctoral dissertation, Technological University Eindhoven (1973).

[9] Prawitz, D., Ideas and results in proof theory; Proc. of the second Scandinavian Logic Symposium, ed. J.E. Fenstad, North Holland 1971.

[10] Scott, D., Constructive validity; Symposium on Automatic Demonstration (Versailles, December 1968), Springer Lecture Notes in Mathematics, vol. 125 (1970), 237-275.

[11] Tait, W.W., Intensional interpretations of functionals of finite type I. J. of Symbolic Logic 32 (1967), 198-212.

[12] Zucker, J., Formalization of classical mathematics in AUTOMATH; Colloque international de logique (Clermont-Ferrand, 1975), ed. M. Guillaume, pp. 135-145, Paris 1977.

SAMENVATTING

In dit proefschrift worden een aantal verschillende formele systemen onderzocht, die alle opgevat kunnen worden als varianten van de λ-calculus. Het bestaat uit vijf onafhankelijke artikelen, voorafgegaan door een algemene inleiding. De artikelen zijn opgenomen in de omgekeerde volgorde van die waarin ze werden geschreven.

In de inleiding wordt de zogenaamde sterke normalisatie-eigenschap, één van de twee in de titel van dit proefschrift genoemde thema's, besproken met het oog op mogelijke toepassingen in Church-Rosser bewijzen.

Het eerste en meest omvangrijke hoofdstuk gaat over het systeem **λπ**, de uitbreiding van de zuivere ongetypeerde λ-calculus met constanten en axioma's voor surjectieve paring. Volgens het zogenaamde surjectiviteitsaxioma kunnen alle termen van de λ-calculus worden opgevat als een paar. Hoewel door het aangeven van een model de consistentie van **λπ** eenvoudig is vast te stellen, was tot dusverre niet bekend of er door deze uitbreiding aan de bewijsbare betrekkingen tussen de pure λ-termen nieuwe toegevoegd worden. Deze vraag wordt in hoofdstuk 1 ontkennend beantwoord.

In hoofdstuk 2 wordt een klassiek sterk normaliseringsresultaat uit de ongetypeerde λ-calculus, waarvan de betekenis in de algemene inleiding is toegelicht, van een nieuw en eenvoudig bewijs voorzien.

Het idee uit hoofdstuk 2 om voor het afschatten van een reductieboom gebruik te maken van een "maximaal oneconomische" reductiestrategie wordt in hoofdstuk 3 toegepast in de getypeerde λ-calculus. Er worden twee bewijzen van sterke normalisatie gegeven, een kort bewijs dat zeer inzichtelijk is en een gecompliceerder bewijs, dat als compensatie echter extra informatie oplevert over de structurele eigenschappen van reductiebomen.

In hoofdstuk 4 wordt in een concreet geval aangetoond dat onder de toevoeging van surjectieve paring aan een getypeerd systeem sterke normalisatie behouden blijft. Hieruit blijkt volgens een in de algemene inleiding aangegeven methode dat in een zo verkregen systeem in het algemeen ook de Church-Rosser eigenschap zal gelden. Hierdoor is het in een dergelijk getypeerd systeem niet nodig gebruik te maken van de gecompliceerde methoden van hoofdstuk 1.

Hoofdstuk 5 is een gepubliceerd artikel, dat een essentiële bijdrage levert aan de bewijstheorie (de zogenaamde "taaltheorie") van Automath-systemen. Deze systemen werden door de Nederlandse wiskundige N.G. de Bruijn ontworpen met het doel tot een zodanige formalisering van de wiskundige betoogtrant te komen dat

(i) het daadwerkelijk formaliseren van wiskunde uivoerbaar is;
(ii) de gecodeerde redeneringen mechanisch gecontroleerd kunnen worden.

Aan dit project ligt een natuurlijke, logische analyse van redeneren ten grondslag, die gebruik maakt van principes uit de λ-calculus.

STELLINGEN behorende bij het proefschrift 'Surjective pairing and strong normalization: two themes in lambda calculus'.

Roel de Vrijer, 7 januari 1987.

I

Laat → de eenstapsreductie in de gelabelde λ-calculus van Hyland/Wadsworth zijn (in de versie van Barendregt, zie ook blz. 3 van dit proefschrift), uitgebreid met de reductieregel

$<$: $M^p \to M^q$ als $q<p$.

Definieer nu met inductie naar n: M^n is *computable* als aan de hieronder geformuleerde voorwaarden (i) en (ii) is voldaan.

(i) M^n is sterk normaliserend;

(ii) wanneer $M^n \twoheadrightarrow (\lambda x.M_0)^{k+1}$ en N^k computable, dan is ook $((x:=N^k)M_0)^k$ computable.

(Vergelijk definitie 5.3.1 op blz. 123 van dit proefschrift.)
Analoog aan het bewijs van stelling 5.3.7 in hoofdstuk 5 van dit proefschrift kan worden bewezen dat alle termen van de vorm M^n computable zijn. Dat dan alle termen sterk normaliseren volgt hieruit met een eenvoudige inductie naar de opbouw. Sterke normalisatie voor Hyland/Wadsworth gelabelde reductie is een onmiddellijk gevolg.

Lit.: H.P. Barendregt, *The Lambda Calculus*, North Holland 1981.

II

Laat $\to_a$ de eenstapsreductie in de gelabelde λ-calculus van Hyland/Wadsworth zijn (in de versie van Barendregt), echter zonder de regel **label**. Wanneer men $\hookrightarrow$ interpreteert als $\to_{\text{label}}$, overal $\Lambda\bot^N$ leest voor $\Lambda\pi\mathbf{p}$, en vocr $\Lambda\pi\mathbf{p}^*$ de verzameling $\to_{\text{label}}$-normaalvormen in $\Lambda\bot^N$ neemt, dan gaan onder deze nieuwe interpretatie de definities en lemma's 2.4.5-10 en 2.4.11(i) en (iv) uit hoofdstuk 1 van dit proefschrift zondermeer door. De relatie $\to_a$ is dan ½- en derhalve, vanwege lemma 2.4.12(i), ook $*$-projecteerbaar. De geprojecteerde relatie $\to_a{}^*$ is precies de gelabelde reductie van Hyland en Wadsworth in de oorspronkelijke versie.

Lit.: H.P. Barendregt, *The Lambda Calculus*, North Holland 1981.
J.M.E. Hyland, 'A syntactic characterization of the equality in some models of the λ-calculus', *J. London Math. Soc.* 12 (1976), p. 361-370.
C.P. Wadsworth, 'The relation between computational and denotational properties for Scott's D∞-models of the lambda-calculus', *SIAM J. Comput.* 5 (1976) p. 488-521.

III

Een term t in de λ-calculus heeft de *range property* als één van de volgende uitspraken geldt (= tussen termen staat voor conversie).

(a) $tX = tY$ voor alle gesloten termen X en Y;

(b) Er zijn oneindig veel gesloten termen $X_0, X_1, \ldots$, zodanig dat $tX_i \neq tX_j$ als $i \neq j$.

Voor termen met een normaalvorm is de range property te bewijzen met een eenvoudige inductie over normaalvormen. Dezelfde redenering kan worden toegepast om aan te tonen dat in de zuivere getypeerde λ-calculus over een basistype met oneindig veel constanten alle termen de range property bezitten.

IV

De procedure die gevolgd wordt bij het verifiëren van Automath-teksten maakt essentieel gebruik van een principe dat—overigens geheel onafhankelijk—door Yessenin Volpin is beschreven als *structurele identificatie*. Door de wijze waarop dit gebeurt lijkt de theoretische beslisbaarheid van Automath-talen van relatief ondergeschikt belang.

Lit.: D.T. van Daalen, *The language theory of Automath*, dissertation Technische Hogeschool Eindhoven (1980), p. 13,14.
A.S. Yessenin Volpin, 'The ultra-intuitionistic criticism and the anti-traditional program for the foundations of mathematics'. In *Intuitionism and proof theory*, ed. Kino, Myhill and Vesley, North Holland (1970).

V

In de recursietheoretische literatuur is het niet ongebruikelijk om ter verkorting van redeneringen een beroep te doen op het intuïtieve inzicht van de lezer dat een bepaald algoritme geformaliseerd zal kunnen worden. Daar is op zichzelf niets tegen. Het is echter misleidend dit voor te stellen als een beroep op de these van Church. De strenge opbouw van de recursietheorie maakt een dergelijk *gebruik* van Church' these—met zijn verderstrekkende filosofische betekenis—overbodig.

VI

Het gelijkheidsbegrip is fundamenteel en onafhankelijk van de aard van de objecten waarop het wordt toegepast. Uit dit oogpunt is het onwenselijk dit begrip in een specifieke context (bijvoorbeeld voor verzamelingen, functies of keuzerijen) te definiëren. De opvatting van ondermeer Bishop en Martin-Löf dat een verzameling pas is gegeven door de elementen *en* een bijbehorende "gelijkheidsrelatie" is filosofisch twijfelachtig.

Lit.: E. Bishop, *Foundations of constructive analysis*, McGraw-Hill (1967).
P. Martin-Löf, *Intuitionistic type theory*, Bibliopolis, Napels (1984).

VII

De sorites-paradox leert dat uit de aanwezigheid van recursiemechanismen in natuurlijke taal niet dwingend volgt dat een adequate grammatica voor natuurlijke taal een oneindig aantal zinnen moet genereren.

VIII

Het conversationeel minimum bestaat niet.

Lit.: F.H. van Eemeren en R. Grotendorst, *Regels voor redelijke discussies*, Foris (1982).
R.C. de Vrijer, 'Ontbrekende premissen', in *Taalbeheersing in theorie en praktijk*, red. W. Koning, Foris (1985).

IX

Het dilemma van Protagoras, ook wel bekend onder de naam *Euathlos*, kan worden opgevat als een variant van de leugenaarsparadox die is toegesneden op een procedureel waarheidsbegrip.

X

Een intuïtie kan worden gedacht als opgebouwd uit twee elementen. Het eerste is een niet volledig gearticuleerde voorstelling van hoe het zit. Het tweede kan negatief worden omschreven als een onvermogen je voor te stellen hoe de werkelijkheid toch nog àf zou kunnen wijken. In de subjectieve zekerheidsbeleving speelt het tweede element een belangrijke rol. Voor een objectieve bepaling van de waarde van het intuïtieve inzicht gaat het om het eerste.

Ontwerp omslag: Renée Wijnen
Illustratie omslag: Karin Lugtigheid

ISBN 90-70879-63-8