LAMBDA CALCULUS EXTENDED WITH SEGMENTS

LAMBDA CALCULUS EXTENDED WITH SEGMENTS

PROEFSCHRIFT

TER VERKRIJGING VAN DE GRAAD VAN DOCTOR IN DE TECHNISCHE WETENSCHAPPEN AAN DE TECHNISCHE HOGESCHOOL EINDHOVEN, OP GEZAG VAN DE RECTOR MAGNIFICUS, PROF. DR. F. N. HOOGE, VOOR EEN COMMISSIE AANGEWEZEN DOOR HET COLLEGE VAN DEKANEN IN HET OPENBAAR TE VERDEDIGEN OP DINSDAG 4 MAART 1986 TE 16.00 UUR.

DOOR

HARMANNUS BALSTERS

GEBOREN TE GRONINGEN

Druk: Dissertatiedrukkerij Wibro, Helmond

Dit proefschrift is goedgekeurd
door de promotoren

Prof.dr. N.G. de Bruijn

en

Prof.dr. W. Peremans

CONTENTS

1. INTRODUCTION

The λ-calculus is concerned with axiomatizing the mathematical concept of function and the rules governing the application of functions to values of their arguments. In the λ-calculus a function is seen as a rule for calculating values; this is a view which differs from the one held in set theory, where a function is to be a set of ordered pairs and is identified with its graph. In axiomatizing the concepts of function and application we define (i) a syntax, consisting of a set of grammar rules, and (ii) inference rules. The λ-calculus to be described in this section, called $\lambda\sigma$, is an extension of the ordinary type free λ-calculus (cf. Barendregt [81]) and was originally conceived by N.G. de Bruijn (cf. de Bruijn[78b]). The main feature of $\lambda\sigma$ is the incorporation of a new class of terms called *segments*. These segments were originally devised in order to provide for certain abbreviational facilities in the mathematical language AUTOMATH. AUTOMATH is a typed λ-calculus in which it is possible to code mathematical texts in such a way that the correctness of each proof written in AUTOMATH can be verified mechanically (i.e. by a computer). There is much to say about the AUTOMATH system, much more than the topic of this thesis aims to cover. We shall mainly treat $\lambda\sigma$ as an interesting extension of the λ-calculus in its own right and not pay very much attention to connections with AUTOMATH. This thesis will be a rather technical treatise of the syntax and axiomatics of $\lambda\sigma$-theory. For an introduction to the AUTOMATH project we refer to de Bruijn [80] and Jutting [81]; the latter paper offers an excellent introduction to a fundamental AUTOMATH-language called AUT-68. For a detailed treatise of the language theory of the AUTOMATH-languages we refer to van Daalen [80].

This introduction consists of three sub-sections. In Section 1.1 we shall give an informal description of the $\lambda\sigma$-system and pinpoint major differences with ordinary type free λ-calculus (for a very complete and up-to-date description of type free λ-calculus we refer to Barendregt [81]). Section 1.2 contains an informal description of the $\lambda_T\sigma$-system ($\lambda\sigma$ extended with types). The types in $\lambda_T\sigma$ are an extension of the types in Church's Theory of simple types (cf. Church [40]), the extension being that simple types are constructed for segments and segment variables. Section 1.3, titled "Reduction and related proper-

ties", differs from Sections 1.1 and 1.2 in that it is completely formal. We have included this formal section in our introduction because it provides an abstract framework in which reduction can be discussed for term-rewriting systems in general.

1.1. An informal introduction to the λσ-system

In this section we shall give an informal description of a system called λσ. We shall offer some explanation for the motivation behind the system and show in which way λσ is an actual extension of ordinary type free λ-calculus. We start with a simple system called λV.

1.1.1. The system λV

The system λV is the well-known type free λ-calculus as described in Barendregt [81], although there are some slight deviations in notation. Type free λ-calculus has formulas like

$$xy \tag{1}$$

$$\lambda_x \cdot xy \tag{2}$$

$$\lambda_x \cdot (\lambda_y \cdot xy) \tag{3}$$

$$(\lambda_x \cdot (\lambda_y \cdot xy))(z) \tag{4}$$

The corresponding formulas in λV are written as

$$\delta yx \tag{1'}$$

$$\lambda_x \delta yx \tag{2'}$$

$$\lambda_x \lambda_y \delta yx \tag{3'}$$

$$\delta z \lambda_x \lambda_y \delta yx \tag{4'}$$

In λV functional abstraction is denoted by $\lambda_x(...)$ (i.e. the function that assigns (...) to the variable x, where x may occur in (...)), and functional application is denoted by δAB (i.e. the function B applied to its argument A, where A and B are λV-terms). Note that in λV arguments are written in front of functions, this in contrast with ordinary type free λ-calculus where application of a function B to its argument

A is usually written as B(A). The syntax of V is very simple and is given below.

Definition 1.1.1.

(1) λV-terms are words over the following alphabet

$v_1, v_2, v_3, \ldots$	variables
λ	abstractor
δ	applicator

(2) The set of λV-terms is the smallest set X satisfying

(i) $x \in X$, for every variable x

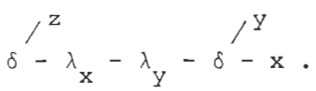
(ii) $A \in X \Rightarrow \lambda_x A \in X$, for every variable x

(iii) $A,B \in X \Rightarrow \delta AB \in X$ □

As will be clear, λV-terms are written in prefix notation: each variable has arity 0, each abstractor λ_x has arity 1 and the applicator δ has arity 2. Each term can be represented as a rooted tree. As an example we consider the term

$$\delta z \lambda_x \lambda_y \delta y x \qquad (4')$$

which we write in tree form as



(4")

The correspondence between terms like (4') and trees like (4") is one-to-one. It certainly helps to think of λV-terms as such trees, and in particular to see operations on terms as operations on their corresponding trees; especially when long terms are involved it is often useful to consider tree representations of terms.

1.1.2. Beta-reduction

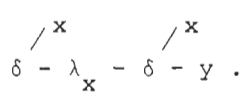
In λ-calculus we have the fundamental notion of application. The application of a function B to an argument A is written as δAB. Apart from functional application we have the notion of functional abstraction. As said before, the intuitive meaning of $\lambda_x(\ldots)$ is "the function that assigns (...) to the variable x". This is illustrated in the following

example (not a λV-term by the way)

$$\delta\, 3\, \lambda_x(2 \cdot x + 1) = 2 \cdot 3 + 1$$

i.e., we substitute the number 3 for the variable x in $2 \cdot x + 1$. A formula of the form $\delta\, A\, \lambda_x B$ is called a *redex*. Substitution of A for the free occurrences of x in B is denoted by $\Sigma_x(A,B)$. The transition from $\delta\, A\, \lambda_x B$ to $\Sigma_x(A,B)$ is called *β-reduction*. We now proceed by giving a more formal description of substitution.

We recall that an occurrence of a variable x in a term A is called *bound* in A if this occurrence of x lies in the scope of some abstractor λ_x in A; otherwise this occurrence of x is called *free* in A. Note that a variable can occur both free and bound in the same term; as an example consider the two occurrences of the variable x in the following term written in tree form



Definition 1.1.2.

If A is a term and x is a variable and y is a variable with $y \neq x$ then we define $\Sigma_x(A,B)$ inductively for terms B by

(1) $\Sigma_x(A,x) = A$

$\Sigma_x(A,y) = y$

(2) $\Sigma_x(A,\lambda_x C) = \lambda_x C$

(3) $\Sigma_x(A,\lambda_y C) =$

$\lambda_y \Sigma_x(A,C)$, if x does not occur free in C, or: y does not occur free in A

$\lambda_z \Sigma_x(A,C')$, otherwise - where C' is obtained by renaming of all free occurrences of y in C by some variable z which does not occur free in A,C.

(4) $\Sigma_x(A,\delta CD) = \delta\, \Sigma_x(A,C)\, \Sigma_x(A,D)$. □

Most of the four clauses in the definition given above are self-evident, with the possible exception of clause (3). Clause (3) is necessary in order to avoid that free occurrences of the variable y in A get bound by the λ_y of $\lambda_y C$ after substitution, which would otherwise lead to

inconsistencies. This renaming of bound variables is known as α-*reduction*. In our case it is said that $\lambda_y C$ α-*reduces* to $\lambda_z C'$. Usually α-reduction is considered unessential. If α-reduction transforms a term A into A' then A and A' are considered to be equivalent in an informal way. This convention implies that the name of a bound variable is unessential; the "meaning" of a term is considered unaltered after performing an α-reduction on that term. Actually, in the definition of substitution given above, clause (3) does not introduce a proper term but rather an α-equivalence class of terms.

1.1.3. Name-free notation

Renaming of bound variables can sometimes be very cumbersome; proofs involving α-reduction are notoriously tedious. But apart from this we have our own intrinsic reasons to avoid α-reduction. Later on we shall introduce the full λσ-system, an extension of λV. The main feature of λσ is the incorporation of a new class of terms called *segments*. Segments are discussed in Section 1.1.4. Substitution of segments for their corresponding variables can give rise to a large number of α-reductions, especially when the formulas are long. There is, however, a very simple way to avoid α-reduction. In de Bruijn [72], N.G. de Bruijn introduced the concept of *nameless dummies*; he invented a λ-calculus notation that makes α-reduction superfluous. The idea is that we just write λ instead of $\lambda_x, \lambda_y, \ldots$ and every variable is replaced by a term of the form ξ(n), where n is some positive integer. Each ξ(n) is called a *name-free variable* and n is called a *reference number*. The reference number n of a name-free variable ξ(n) determines the λ that binds a specific occurrence of ξ(n) in some term. The procedure is as follows. If the name-free variable ξ(n) occurs in some term t, we first form the tree representation of t. We then descend from ξ(n) towards the root of the tree and the n-th λ encountered is the λ that binds ξ(n). As an example consider the following name-carrying term in tree representation

$$\lambda_x - \lambda_y - \delta \overset{y}{/} - \lambda_z - \delta \overset{z}{/} - \lambda_w - x \ .$$

The name-free equivalent of this term is

$$\lambda - \lambda - \delta \overset{\xi(1)}{-} \lambda - \delta \overset{\xi(1)}{-} \lambda - \xi(4)$$

Remark. If a reference number n is larger than the number of λ's lying on the path from an occurrence of $\xi(n)$ to the root of the tree in which it occurs then we can interpret that occurrence as being free.

The use of name-free notation has certain consequences for substitution of $\lambda\xi$-terms (λV-terms written in name-free form), which we now shortly describe. Substitution in a $\lambda\xi$-term t results in the replacement of free occurrences of a certain variable in t by some term u. We could also describe this situation in terms of trees by saying that certain end-points $\xi(n)$ of the tree equivalent $\hat{t}$ of t have been replaced by some tree $\hat{u}$. Consider the following example of such a substitution in a $\lambda\xi$-tree.

Let t be the $\lambda\xi$-term

$$\lambda\,\lambda\,\delta\,\delta\,\xi(2)\,\xi(1)\,\lambda\,\lambda\,\lambda\,\xi(3)$$

which has the following tree-representation $\hat{t}$

$$\lambda - \lambda - \delta - \lambda - \lambda - \lambda - \xi(3)\ .$$
$$\qquad\qquad\quad \delta - \xi(1)$$
$$\qquad\qquad\qquad\quad \xi(2)$$

This tree contains a redex, namely

$$\delta - \lambda - \lambda - \lambda - \xi(3)$$
$$\quad\ \delta - \xi(1)$$
$$\qquad\ \xi(2)$$

and we can therefore perform a β-reduction on $\hat{t}$. By β-reducing $\hat{t}$, the end-point $\xi(3)$ is a candidate for substitution of the sub-tree

$$\delta - \xi(1)\ .$$
$$\quad\ \xi(2)$$

Should we, however, simply replace $\xi(3)$ by this sub-tree, as would have

been the case if $\hat{t}$ had been written in name-carrying form, then this would result in the following tree $\hat{t}'$

$$\lambda - \lambda - \lambda - \lambda - \delta - \begin{array}{l} \xi(2) \\ \xi(1) \end{array} .$$

It is immediately clear that the variables $\xi(1)$ and $\xi(2)$ in $\hat{t}'$ refer to completely different λ's than in $\hat{t}$. This inconsistency is due to the fact that

(i) $\xi(1)$ and $\xi(2)$ are external references in $\hat{t}$ (i.e., references to λ's to the left of the subterm $\delta\,\xi(2)\,\xi(1)$);
(ii) after replacement, the variables $\xi(1)$ and $\xi(2)$ in $\hat{t}'$ have two extra λ's on their left.

There is, however, a simple way to resolve this inconsistency: by raising the reference numbers 1 and 2 in $\xi(1)$ and $\xi(2)$ by 2 in $\hat{t}'$, these variables refer to the same λ's that they originally referred to in $\hat{t}$. This example demonstrates that certain measures have to be taken in order to ensure that external references remain intact when we substitute a $\lambda\xi$-term. In Section 2, where we give a formal definition of substitution of name-free terms, we shall introduce so-called *reference mappings*, which see to it that reference numbers are suitably updated in order to avoid inconsistencies as described above. We refrain from further discussion of these reference mappings here; they shall be described extensively, both informally and formally, in Section 2.

In the following sections of this chapter we shall first stick to name-carrying notation of formulas. The major reason for this is to point out that name-carrying notation can possibly be maintained in $\lambda\sigma$-calculus (λV-calculus extended with segments and segment variables), but we also want to show how awkward things can get in $\lambda\sigma$-calculus by employing name-carrying notation. In the case of λV-calculus the name-free notation might seem exaggerated in preciseness, and we can imagine reservations towards this notation as far as readability of formulas is concerned. In the case of $\lambda\sigma$-calculus we shall try to show that the name-free notation has advantages over name-carrying notation, both in preciseness and readability.

1.1.4. Segments and abbreviations

We may consider a variable as an abbreviation of a certain term if this variable can be replaced by that term by means of some suitable β-reduction. For example, consider the following term written in tree form

$$\lambda_w - \delta - \begin{matrix} /\lambda_x - x \\ \lambda_z - \delta - z \end{matrix} \quad \begin{matrix} \\ /w \end{matrix} \quad . \tag{5}$$

By β-reducing (5) we obtain the term

$$\lambda_w - \delta - \begin{matrix} /w \\ \lambda_x - x \end{matrix} \quad , \tag{5'}$$

i.e. a term in which the variable z has been replaced by the term $\lambda_x x$ and the redex has vanished. If we would have more occurrences of the variable z, each bound by the λ_z of the redex, then each of these occurrences serves as a kind of abbreviation of the term $\lambda_x x$.

In λσ there are, however, still quite different things that we want to abbreviate. One such thing is a so-called δ-string like

$$\begin{matrix} /A \quad /B \quad /C \\ \delta - \delta - \delta \end{matrix} \quad . \tag{6}$$

If it occurs more than once in a certain term, we may wish to abbreviate it. Yet (6) is not a term, in the sense of a λV-term, but only *part* of a term; it becomes a λV-term if we place an arbitrary λV-term behind it. Such parts of λV-terms are called *segments*. Another example of a segment is a so-called λ-string like

$$\lambda_x - \lambda_y - \lambda_z \quad . \tag{7}$$

In AUTOMATH we have many cases where we would like to abbreviate segments. In this respect we mention an interesting AUTOMATH-language, namely Nederpelt's language Λ (cf. Nederpelt [73]). The original idea of introducing such a language as Λ stems from N.G. de Bruijn who devised a language called AUT-SL (from AUTOMATH-Single Line) in which AUTOMATH texts can be represented as one single formula. The language Λ was devised as a fundamental and simple AUTOMATH-language which is very well suited for language-theoretical investigations. In typical

codings of AUTOMATH texts in Λ we encounter very many copies of certain δ-strings and λ-strings, copies which we would like to abbreviate. As a consequence, segments like δ-strings and λ-strings will be treated as separate independent entities in $\lambda\sigma$. In $\lambda\sigma$ we shall even take a broader approach and allow for segments of a much more general form than δ-strings or λ-strings alone. In the following section we shall give examples of such segments of a more general form.

1.1.5. Segment variables and substitution

Segments are terms with a kind of open end on the extreme right. From now on we shall use the symbol ω to indicate the open end on the right. So

$$\delta - \overset{A}{\delta} - \overset{B}{\delta} - \overset{C}{\delta} - \omega$$

is a segment as well as

$$\lambda_x - \lambda_y - \lambda_z - \omega \ .$$

As said before, segments are not λV-terms; a segment becomes a λV-term if we replace the ω by an arbitrary λV-term. According to this scheme the following formulas can also be considered as segments:

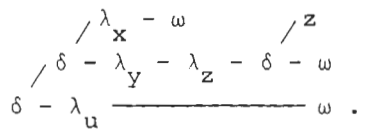
$$\delta - \overset{A}{\lambda_x} - \lambda_y - \omega$$

$$\lambda_x - \overset{A}{\delta} - \lambda_y - \overset{B}{\delta} - \omega \ .$$

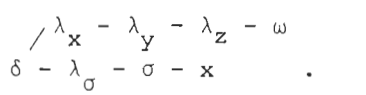
By replacing the ω in both of these formulas by some λV-term we obtain a λV-term (provided, of course, that A and B are λV-terms). In $\lambda\sigma$ we will go even one step further by allowing recursive nesting of segments, and as a consequence ω's can occur in other branches as well, like in

$$\delta - \overset{\lambda_x - \omega}{\lambda_y} - \lambda_z - \overset{z}{\delta} - \omega$$

or



All these occurrences of ω in the foregoing formulas act as a kind of "holes", which - once replaced by a λV-term - yield again a λV-term. All formulas having an ω on the extreme right are called segments in λσ. Along with segments we also add to our system a new kind of variables for which segments can be substituted. These variables are represented by unary prefix symbols and are denoted, in name-carrying form, by σ,σ',σ",.... An example of a λσ-term containing a segment and a segment variable is



(8)

This term is in redex form, where the segment variable σ is bound by the λ_σ of the redex. Performing a β-reduction on this redex results in

$$\lambda_x - \lambda_y - \lambda_z - x \qquad (8')$$

i.e., the prefix symbol σ is replaced by the segment $\lambda_x \lambda_y \lambda_z$ (where the ω has been dropped). In λσ, segment variables can serve as a means to abbreviate segments, just like variables in λV can serve as a means to abbreviate λV-terms. When using segment variables to abbreviate segments we must be careful, though. Consider for example the λσ-term (8). The variable x in that term refers to the abstractor λ_x hidden inside the segment variable σ, as seen in (8') where x gets bound by λ_x after β-reduction of (8). This is an intended feature which we always have to take into account in λσ-calculus. If a segment variable σ occurs in some λσ-term then after replacement of σ by the segment s that σ abbreviates in t, it can happen, as most often will be the case, that certain variables occurring in t get captured by abstractors lying on the main branch of the tree representation of s. This is to say that each occurrence of a segment variable σ in a λσ-term t can contain abstractors - *hidden inside* σ - which will capture certain variables in t after performing a β-reduction in t resulting in the replacement of σ by the segment that σ abbreviates in t.

We now wish to discuss a situation in which there are more occurrences of the same segment variable σ in some $\lambda\sigma$-term. Consider the following $\lambda\sigma$-term in tree representation

$$\begin{array}{l} \diagup\, \lambda_x - \lambda_y - \omega \qquad \diagup\, x \\ \delta - \lambda_\sigma - \sigma - \sigma - \delta - y \ . \end{array} \tag{9}$$

Performing a β-reduction on this term results in

$$\lambda_x - \lambda_y - \lambda_x - \lambda_y - \delta \overset{\diagup\, x}{-} y \tag{9'}$$

where both instances of σ have been replaced by the segment $\lambda_x \lambda_y$. The variables x and y in (9') are bound by the last two abstractors λ_x and λ_y as indicated by the arrows in (9") shown below

$$\lambda_x - \lambda_y - \lambda_x - \lambda_y - \delta \overset{\diagup\, x}{-} y \ . \tag{9"}$$

Suppose, however, that we would want x and y to be bound by other occurrences of the abstractors λ_x and λ_y as indicated in

$$\lambda_x - \lambda_y - \lambda_x - \lambda_y - \delta \overset{\diagup\, x}{-} y \ . \tag{9'''}$$

In $\lambda\sigma$ we want to have the freedom to allow for such deviations in priority of binding power of λ's, which appear when we have more than one occurrence of some segment variable σ in a $\lambda\sigma$-term. One way of doing this is by renaming the abstractors in (9') in a suitable way; consider for example the following term

$$\lambda_x - \lambda_y - \lambda_{x_1} - \lambda_{y_1} - \delta \overset{\diagup\, x}{-} y \qquad (x \neq x_1,\ y \neq y_1) \ . \tag{9""}$$

It is clear that the variables x and y are bound by the first two abstractors λ_x and λ_y, just as we intended them to be bound in (9'''). This renaming, however, is done after substitution has taken place; i.e. the renaming has taken place after β-reduction of (9) to (9'). What we would like is that it can be seen beforehand (i.e. before β-reduction takes place) how the abstractors inside segments shall be

renamed. We would like to have a means systematically indicating beforehand how this renaming of bound variables shall take place, instead of more or less arbitrarily renaming bound variables in segments after β-reduction. One way of doing this is by replacing the first, respectively the second, occurrence of σ in (9) by $\sigma(x,y)$, respectively $\sigma(x_1,y_1)$. These parameter lists (x,y) and (x_1,y_1) serve as instructions indicating that the abstractors λ_x and λ_y are to be renamed by λ_x,λ_y and $\lambda_{x_1},\lambda_{y_1}$ in the first, respectively the second occurrence of σ in (9) (actually only in the second occurrence of σ real renaming takes place). In general if a segment has n ($n \geq 0$) λ's lying on the main branch of its tree, say $\lambda_{x_1},\ldots,\lambda_{x_n}$, and σ is a segment variable referring to that segment then by adding a parameter list $(y_1,\ldots,y_n)$ to σ we have an instruction indicating that the *n* abstractors $\lambda_{x_1},\ldots,\lambda_{x_n}$ are to be renamed by $\lambda_{y_1},\ldots,\lambda_{y_n}$ and in that order. Also the occurrence of the variables $x_1,\ldots,x_n$ in the segment which were bound by $\lambda_{x_1},\ldots,\lambda_{x_n}$ are to be renamed by $y_1,\ldots,y_n$. We note that it is important that the parameter list added to a segment variable σ has as its length: the number of λ's lying on the main branch of the segment s that σ refers to (this number is called the *weight* of s).

By adding parameter lists to segment variables we have a means to bind occurrences of variables referring to a λ hidden inside a segment exactly as we desire. There is still one problem, though, that we have to resolve. When performing a β-reduction inside a segment we are sometimes dealing with redices which, in the substitutional process involved, have an effect on the ω on the extreme right of that segment. Consider, for example, the following segment

$$\lambda_x - \delta \overset{A}{/} - \lambda_y - \lambda_z - \lambda_w - \omega \ . \tag{10}$$

By β-reducing the redex $\delta\, A\, \lambda_y\, \lambda_z\, \lambda_w\, \omega$ occurring in (10) we are faced with evaluating $\Sigma_y(A,\lambda_z\,\lambda_w\,\omega)$. By the clauses given in Definition 1.1.1 we know how to "shift" the Σ_y-operator past the two abstractors λ_z and λ_w, but then we arrive at the ω and have to decide how to evaluate $\Sigma_y(A,\omega)$. We could simply define $\Sigma_y(A,\omega)$ as ω, but then certain vital information would get lost; a situation which we now explain. Suppose that (10) occurs as a segment in some term t and that (10)

is referred to by some segment variable $\sigma(y_1,y_2,y_3,y_4)$ occurring in t. Suppose also that there is an occurrence of the variable y_2 in t which refers to the abstractor λ_{y_2} hidden inside $\sigma(y_1,y_2,y_3,y_4)$. By β-reducing (10) and defining $\Sigma_y(A,\omega)$ as ω, this occurrence of y_2 is no longer a candidate for substitution of the term A (which would have been the case prior to this β-reduction of (10)), simply because the abstractor λ_y (or better: λ_{y_2}) has vanished. In order to avoid inconsistencies and to keep this candidate-role of substitution for such occurrences of variables y_2 intact, we shall define such substitutions of a term A at an end-point ω of a segment by

$$\Sigma_y(A,\omega) = \delta \not{-}^{A} \lambda_y - \omega \ .$$

In this way it remains possible to refer to the λ_y of the original redex in (10), and occurrences of variables which referred indirectly to that lambda by means of a reference to a lambda hidden inside some segment variable remain candidates for substitution of the term A. There is still a problem, though, because the order of the λ's in the reduced segment is different from the order in which they appeared in the original segment. In our example, β-reduction of (10) results in

$$\lambda_x - \lambda_{z'} - \lambda_{w'} - \delta \not{-}^{A} \lambda_y - \omega \qquad (10')$$

where z and w have possibly been replaced by new variables z' and w', this in case that free occurrences of z or w in A would otherwise have been captured. The abstractors in (10) appear in the order $\lambda_x,\lambda_y,\lambda_z,\lambda_w$ and in (10') the order is $\lambda_x,\lambda_{z'},\lambda_{w'},\lambda_y$. This difference has consequences when these segments are substituted for some occurrence of a variable $\sigma(y_1,y_2,y_3,y_4)$. Consider, for example, the following two terms in which the segments (10), respectively (10'), occur

$$\begin{array}{l} \not{\lambda}^{A}_{x} - \delta - \lambda_y - \lambda_z - \lambda_w - \omega \\ \delta - \lambda_\sigma - \sigma(y_1,y_2,y_3,y_4) - y_2 \end{array} \qquad (11)$$

and

$$/^{A}\ \lambda_x - \lambda_{z'} - \lambda_{w'} - \delta - \lambda_y - \omega$$
$$\delta - \lambda_\sigma - \sigma(y_1,y_2,y_3,y_4) - y_2 \quad . \tag{11'}$$

These terms β-reduce to

$$\lambda_{y_1} - \delta - /^{A'}\ \lambda_{y_2} - \lambda_{y_3} - \lambda_{y_4} - y_2 \tag{12}$$

and

$$\lambda_{y_1} - \lambda_{y_2} - \lambda_{y_3} - /^{A'}\ \delta - \lambda_{y_4} - y_2 \tag{12'}$$

where A' is obtained from A by renaming all free occurrences of x by y_1. In (12) we see that A' can be substituted for y_2 by performing one more β-reduction; this is, however, not the case in (12'). So by changing the order of the λ's in some segment s by performing a β-reduction inside s we can get the situation that occurrences of variables that originally (i.e. prior to this β-reduction of s) referred to a certain λ hidden inside some parameter-listed segment variable, afterwards refer to a completely different λ. There is a way, however, in which such inconsistencies can be resolved. By adding an extra parameter, called a *segment mapping* (or *segmap* for short) to an ω we can safely β-reduce a segment prior to substitution of that segment. A segmap is a permutation of some interval [1..n] of $\mathbb{N}$ ($n \geq 0$), and tells us how to restore the original order of the λ's occurring in a segment; i.e. by adding a segmap to the ω on the extreme right of a segment we can determine the order in which the abstractors occurred before β-reduction of the original segment. Instead of writing ω we now write ω(ψ), where ψ is some segmap. In our example we replace the ω on the extreme right of (11') by ω(ψ), where ψ is a permutation of [1..4] defined by

$$\psi(1) = 1$$
$$\psi(2) = 3$$
$$\psi(3) = 4$$
$$\psi(4) = 2 \ .$$

Let us denote this modification of (11') by (11"). If we rearrange the order of the parameter list (y_1,y_2,y_3,y_4) in accordance to ψ (i.e. the first parameter remains first in the list, the second becomes the third, the third becomes the fourth and - most importantly - the fourth parameter becomes the second in the list) then we obtain a new parameter list (y_1,y_3,y_4,y_2). By replacing the parameter list (y_1,y_2,y_3,y_4) in (11') by this new parameter list (y_1,y_3,y_4,y_2) we obtain the following modified version of (11")

$$\begin{array}{l} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad /^{A'} \\ /\lambda_x - \lambda_{z'} - \lambda_{w'} - \delta - \lambda_y - \omega \\ \delta - \lambda_\sigma - \sigma(y_1,y_3,y_4,y_2) - y_2 \end{array} \qquad (11")$$

which β-reduces to

$$\lambda_{y_1} - \lambda_{y_3} - \lambda_{y_4} - \delta - \lambda_{y_2} - y_2 \qquad (12")$$

and we see that all occurrences of variables in (12) and (12") refer to the same λ's, just as we wanted.

By adding parameter lists and segmaps we can take care of problems concerning references to λ's hidden inside segment variables in a suitable way. We shall now attempt to give a more formal description of substitution of a segment for a segment variable.

We shall present this definition in name-carrying form, this in order to show that name-carrying notation can be maintained in principle but that employment of name-free notation provides for a more natural (and certainly more concise) means for dealing with substitution of segments for segment variables.

Definition 1.1.3.
Let $A\,\omega(\psi)$ be a segment with weight n ($n \in \mathbb{N} \cup \{0\}$), ψ be a permutation of $[1..n]$ and B be a term. Substitution of $A\,\omega(\psi)$ for $\sigma(y_1,\ldots,y_n)$ in $\sigma(y_1,\ldots,y_n)B$ is defined by

$$\Sigma_{\sigma(y_1,\ldots,y_n)}\,(A\,\omega(\psi),\sigma(y_1,\ldots,y_n)B) = \qquad (i)$$

$$= \Sigma_{\sigma(y'_1,\ldots,y'_n)}\,(A\,\omega(id(n)),\sigma(y'_1,\ldots,y'_n)B) = \qquad (ii)$$

$$= A' \; \Sigma_{\sigma(y_1,\ldots,y_n)} \; (A\,\omega(\psi),B) \qquad \text{(iii)}$$

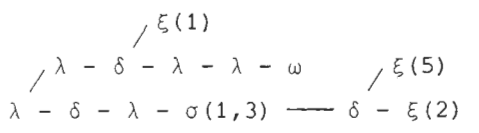
where id(n) denotes the identity map on [1..n], $(y'_1,\ldots,y'_n)$ is the result of rearranging $(y_1,\ldots,y_n)$ as indicated by ψ and A' is the result of suitable renaming of bound variables in A as indicated by $(y'_1,\ldots,y'_n)$. □

This definition is still rather vague since we have not defined $\Sigma_{\sigma(y_1,\ldots,y_n)}\,(A\,\omega(\psi),B)$, and also because such descriptions as "rearrangement of a parameter list as indicated by a segmap" and "suitable renaming of bound variables in a term as indicated by a parameter list" can hardly be considered as descriptions with formal status. The transition from (ii) to (iii) is also a bit strange, since it is not clear from (ii) alone how the segmap ψ in (iii) suddenly turns up again. Apparently, this is not a very good definition since it is too vague; but, as mentioned before, this definition was only intended as an attempt towards a formal definition. A precise formal definition of substitution for segment variables can of course be given, but such a definition would be rather involved. There is a more elegant and shorter way to define substitution for segment variables, namely by employing name-free notation for segments and segment variables. This notation is described in the following section.

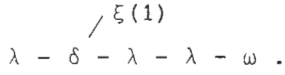
1.1.6. Name-free notation for segments and segment variables

There is another way of dealing with references to λ's hidden inside segment variables than attaching parameter lists to segment variables, namely by employing name-free notation. What we shall do is the following. Segment variables are written in name-free form as σ(n,m), where n denotes the reference number of σ (which, like in ξ(n), determines the λ that some specific occurrence of σ(n,m) refers to) and m (m ≥ 0) denotes the number of λ's lying on the main branch of the tree representation of the segment that σ(n,m) intends to abbreviate (the number m is also called the *weight* of σ(n,m)). The number m in σ(n,m) is to play the role of a parameter list in name-carrying notation; i.e. m indicates that there are m λ's hidden inside σ(n,m). As an example of a term in name-free notation containing a segment

and a segment variable consider the following term written in tree form



In this term we see that $\sigma(1,3)$ abbreviates a segment with three λ's lying on the main branch of its tree; so when determining the λ that $\xi(5)$ refers to we descend from $\xi(5)$ towards the root of the tree, subtract 3 from 5, subsequently subtract 1 and see that $\xi(5)$ refers to the first λ (from the left) of the tree. The variable $\xi(2)$ refers to the second λ (from the right) hidden inside $\sigma(1,3)$; $\xi(2)$ is thus bound by the second λ (from the right) of the segment



By employing name-free notation we get a concise way of denoting segment variables and can do without attaching (potentially long) parameter lists to these variables. There is still one problem, though; a problem which we discussed earlier on in the name-carrying version of $\lambda\sigma$-calculus, which dealt with the performance of certain β-reductions inside segments prior to substitution of those segments for their respective segment variables. By performing a β-reduction inside a segment, the order in which certain λ's originally occurred in that segment can be disturbed and, as we have seen earlier, this can lead to problems when we substitute the reduced segment for certain occurrences of segment variables in a term in which that segment occurs. We solved those problems by adding segmaps to the ω's on the extreme right of the segments involved and we shall do so again in the name-free version of $\lambda\sigma$.

We now shortly describe substitution of segments for segment variables and we shall give this description in an informal manner in terms of trees. The tree representation of a segment has an $\omega(\psi)$ - where ψ is some segmap - on the extreme right of its main branch. When we substitute a segment we remove the $\omega(\psi)$ and put the remaining tree fragment in the place of some occurrence of a segment variable in a $\lambda\sigma$-tree. Segment variables occur in $\lambda\sigma$-trees as unary nodes and substi-

tution of segments for segment variables thus gives rise to replacements at unary nodes inside a $\lambda\sigma$-tree (which differs completely from $\lambda\xi$-substitutions, where we could only perform replacements at end-nodes of trees). When such a substitution is performed, we again - as in the case of $\lambda\xi$-substitutions - have to be careful and update external references in segments in order to ensure that these references remain intact after substitution. But not only do we have to update external references when we substitute a segment for a corresponding segment variable, we also have to take into account the effect of the segmap ψ attached to the end-point ω of the segment involved, since such a segmap reallocates references to λ's lying on the main branch of the segment which we want to substitute. We now give an example to demonstrate both of these features. Consider the following example of a $\lambda\sigma$-tree containing a segment and a segment variable

$$\begin{array}{l} \qquad\qquad\qquad\qquad\quad / \xi(3) \\ \qquad\qquad\quad / \lambda - \lambda - \delta - \omega(\psi) \\ \lambda - \delta - \lambda - \lambda - \lambda - \sigma(3,2) - \xi(1) \end{array}$$

where ψ is the permutation of $[1..2]$ defined by $\psi(1) = 2$ and $\psi(2) = 1$. This tree, which we shall refer to as $\hat{t}$, contains a redex, namely

$$\begin{array}{l} \qquad\qquad\quad / \xi(3) \\ \qquad / \lambda - \lambda - \delta - \omega(\psi) \\ \delta - \lambda - \lambda - \lambda - \sigma(3,2) - \xi(1) \end{array}$$

and we can therefore perform a β-reduction on $\hat{t}$. By β-reducing $\hat{t}$, the unary node $\sigma(3,2)$ is a candidate for substitution of the sub-tree

$$\begin{array}{l} \qquad / \xi(3) \\ \lambda - \lambda - \delta - \omega(\psi) \ . \end{array}$$

Should we simply replace $\sigma(3,2)$ by the tree fragment

$$\begin{array}{l} \qquad / \xi(3) \\ \lambda - \lambda - \delta \end{array}$$

then this would result in the following tree $\hat{t}'$

$$\begin{array}{l} \qquad\qquad\qquad\qquad / \xi(3) \\ \lambda - \lambda - \lambda - \lambda - \lambda - \delta - \xi(1) \ . \end{array}$$

It is immediately clear that the variables $\xi(1)$ and $\xi(3)$ refer to different λ's than they originally referred to in $\hat{t}$. The variable $\xi(3)$ is an external reference in $\hat{t}$ and, as in the case of $\lambda\xi$-substitutions, has to be suitably updated whenever the segment in which $\xi(3)$ occurs is substituted for some segment variable. The variable $\xi(1)$ in $\hat{t}$ refers to one of the two λ's hidden inside $\sigma(2,3)$; it seems to refer to the first λ (from the right) lying on the main branch of the segment involved, but the segmap ψ reallocates this reference to the second λ (from the right). This means that correct β-reduction of $\hat{t}$ would result in the following tree $\hat{t}''$

$$\lambda - \lambda - \lambda - \lambda - \lambda - \delta \begin{array}{l} - \xi(2) \\ \diagup\ \xi(5) \end{array} .$$

In Section 2 we shall give a formal definition of substitution of $\lambda\sigma$-terms. In this definition we shall use so-called *reference mappings* which see to it that reference numbers are suitably updated, like in our example in the transition from $\hat{t}$ to $\hat{t}''$. These reference mappings (or *refmaps* for short) and their interaction with $\lambda\sigma$-terms are described extensively in Section 2, and we refrain from further discussion of refmaps here.

The employment of name-free notation and segmaps makes it possible to give a formal definition of substitution of segments for segment variables in a very concise way, as we shall see in Section 2. In previous examples describing how substitution of segments for segment variables can take place we have restricted ourselves to rather simple situations. Our formal treatment of such substitutions, however, will take much more involved situations into account. Our formal definition of substitution will take into consideration certain accumulative effects which can occur when segments contain references to other segments, or even λ's which bind segment variables.

1.2. An introduction to the typed system $\lambda_T\sigma$

In this section we shall give a description of the $\lambda\sigma$-system extended with types for terms. The types in $\lambda_T\sigma$ are a generalization of the types described in Church's Theory of simple types (cf. Church [40]),

the extension being that simple types are constructed for segments and that the description is given in name-free notation. The basic ideas for our description are taken from de Bruijn [78b]. We shall start from a name-carrying calculus without segments - which, basically, is Church's system of simple types - called $\Lambda_T V$. We then gradually move on to a system in which operations on types are made more explicit and in which the name-free notation is incorporated. Finally, we shall describe the full $\lambda_T \sigma$-system by offering, in name-free notation, a typing of segments. The definitions offered in this section will be followed by explanatory remarks.

Definition 1.2.1 ($\Lambda_T V$).

(1) Type symbols (T)

The set of type symbols T is the smallest set X such that

(i) $e, \otimes \in X$;

(ii) $\alpha, \beta \in X \setminus \{\otimes\} \Rightarrow (\alpha\beta) \in X$.

(2) Primitive symbols

The set of primitive symbols consists of

(i) variables: $x_\alpha, y_\alpha, z_\alpha, \ldots \quad \alpha \in T \setminus \{\otimes\}$;

(ii) the symbols λ (abstractor)
and δ (applicator) .

(3) Terms ($\lambda_T V$)

The set of terms $\lambda_T V$ is the smallest set X such that

(i) $x_\alpha \in X$, for every variable x_α ;

(ii) $t \in X \Rightarrow \lambda x_\alpha t \in X$, for every variable x_α ;

(iii) $u, v \in X \Rightarrow \delta u v \in X$.

(4) Types of terms

The function typ on $\lambda_T V$ is defined inductively for terms t by

(i) $typ(x_\alpha) = \alpha$;

(ii) $typ(\lambda x_\alpha u) = \begin{cases} (\alpha\beta) , & \text{if } typ(u) = \beta \neq \otimes \\ \otimes , & \text{otherwise} \end{cases}$;

(iii) $typ(\delta u v) = \begin{cases} \beta , & \text{if } typ(u) = \alpha \neq \otimes \text{ and } typ(v) = (\alpha\beta) \\ \otimes , & \text{otherwise} \end{cases}$.

(5) The set of correct terms ($\Lambda_T V$)

$\Lambda_T V = \{t \in \lambda_T V \mid typ(t) \neq \otimes\}$. □

Remarks.

(1) e is some ground type, ⊗ is to be interpreted as the type of terms which are "incorrectly" typed.

(2) $(\alpha\beta)$ is to be interpreted as the type of those terms which map terms of type α to terms of type β.

(3) If $typ(t) = \alpha$ then α is generally of the form $(\alpha_1(\alpha_2(\alpha_3 \ldots (\alpha_n \alpha_{n+1}) \ldots)))$, where $\alpha_1,\ldots,\alpha_{n+1}$ are types. Speaking in terms of trees, this means that there are n abstractors $\lambda x_{\alpha_1},\ldots,\lambda x_{\alpha_n}$ lying on the main branch of the tree representation $\hat{t}$ of t (and in that order) that cannot be removed by some β-reduction in t; i.e. for each abstractor λx_{α_i} there is no matching δ (or rather: δA_i) such that this $\delta\lambda$-pair can be removed by means of a suitable sequence of β-reductions.

Before giving the next definition we introduce some notation concerning sequences. For an elaborate treatment of sequences we refer to Section 2.1. At this stage it is only important to know that a sequence is seen as a function with some interval [1..n] of $\mathbb{N}$ ($n \geq 0$) as its domain, where n will be the length of the sequence in question.

Notation. Let C be some non-empty set (called an *alphabet*).

- C^* denotes the set of sequences over C (including the empty sequence denoted by $\emptyset$ (the empty set)).
- if $c \in C$ then <c> denotes the sequence of length 1 consisting of the "symbol" c.
- if $F,G \in C^*$ then F & G denotes the concatenation of the sequences F and G, in particular if F is a sequence of length n ($n \geq 0$) then F = <F(1)> & <F(2)> & ... & <F(n)> .
- if $F \in C^*$ then $\bar{F}$ denotes the reversed sequence of F, i.e. if F = <F(1)> & <F(2)> & ... & <F(n)> then $\bar{F}$ = <F(n)> & ... & <F(2)> & <F(1)> .

In the following definition we offer an alternative version of $\Lambda_T V$ in which operations on types are made more explicit.

Definition 1.2.2 ($\Lambda_{T\gamma} V$) .

(1) Types ($T\gamma$)

The set of types $T\gamma$ is the smallest set X such that

(i) $\otimes \in X$;

(ii) $F \in (X\backslash\{\otimes\})^* \Rightarrow \gamma(F) \in X$.

(2) Primitive symbols

The set of primitive symbols consists of

(i) variables: $x_f, y_f, z_f, \ldots \qquad f \in T\gamma\backslash\{\otimes\}$;

(ii) the symbols λ (abstractor)
and δ (applicator) .

(3) Terms ($\lambda_{T\gamma} V$)

The set $\lambda_{T\gamma} V$ is the smallest set X such that

(i) $x_f \in X$, for every variable x_f ;

(ii) $t \in X \Rightarrow \lambda x_f t \in X$, for every variable x_f ;

(iii) $u,v \in X \Rightarrow \delta uv \in X$.

(4) Types of terms

The function γ-typ on $\lambda_{T\gamma} V$ is defined inductively for terms t by

(i) $\gamma\text{-typ}(x_f) = f$;

(ii) $$\gamma\text{-typ}(\lambda x_f u) = \begin{cases} \gamma(\langle f\rangle \,\&\, G), & \text{if } \gamma\text{-typ}(u) = \gamma(G), \\ & \text{for some } G \in (T\gamma\backslash\{\otimes\})^* ; \\ \otimes, & \text{otherwise} \end{cases}$$

(iii) $$\gamma\text{-typ}(\delta uv) = \begin{cases} \gamma(G), & \text{if } \gamma\text{-typ}(u) = f \text{ and} \\ & \gamma\text{-typ}(v) = \gamma(\langle f\rangle \,\&\, G), \\ & \text{for some } f \in T\gamma\backslash\{\otimes\} \text{ and} \\ & G \in (T\gamma\backslash\{\otimes\})^* . \\ \otimes, & \text{otherwise} \end{cases}$$

(5) The set of correct terms ($\Lambda_{T\gamma} V$)

$\Lambda_{T\gamma} V = \{t \in \lambda_{T\gamma} V \mid \gamma\text{-typ}(t) \neq \otimes\}$. □

Remarks.

(1) We note that the symbol γ is of no particular interest in itself, and the reason for introducing it is basically historical in nature. In de Bruijn [78,b] types of $\Lambda_{T\gamma}$-terms (i.e. *non-segments*) were called "green" types, whereas types of segments were called "red" types. The symbol γ has been chosen for the construction of the type of a $\Lambda_{T\gamma}$V-term purely for mnemonic reasons. In Definition 1.1.5 ($\lambda_T\sigma$) we shall construct types of segments, and these types will be of the form $\rho(F,G,H)$. Here the symbol ρ is used in the construction of types of segments, again, purely for mnemonic reasons.

(2) $\gamma(\emptyset)$ is the analogue of the ground type e in Definition 1.2.1.

(3) $\gamma(<f> \& G)$ is the type of those terms which map terms of type f to terms of type $\gamma(G)$ (cf. clause (4) (ii) above).

(4) In terms of trees, if $\gamma\text{-typ}(t) = \gamma(<f_1> \& \ldots \& <f_n>)$, then this means that there are n abstractors $\lambda x_{f_1},\ldots,\lambda x_{f_n}$ lying on the main branch of the tree representation $\hat{t}$ of t that cannot be removed by means of a suitable sequence of β-reductions in t (cf. comment (3) in the remarks on Definition 1.2.1).

In the following definition we go one step further and introduce a new type-constructor π which takes two arguments, both sequences of types. We recall that $\gamma(F)$ denotes the type of those terms with n abstractors lying on the main branch of their corresponding trees (we assume that F is a sequence $<f_1> \& \ldots \& <f_n>$ of length n) that cannot be removed by suitable β-reductions. In the case of segments, however, we can also have terms with *applicators* lying on the main branch of their tree representations which cannot be removed by means of suitable β-reductions. When we write $\pi(F,G)$, where F and G are sequences of types $<f_1> \& \ldots \& <f_n>$ and $<g_1> \& \ldots \& <g_m>$, respectively, then F denotes the sequence of n non-removable abstractors and G denotes the sequence of m non-removable applicators. We also introduce a product operation "$\star$" between π-types and γ-types with which we can calculate types of terms. We note that terms in the system $\Lambda_{T\pi\gamma}$V, defined below, are never typed as π-types; π-types in $\Lambda_{T\pi\gamma}$V are only used as intermediate constructs for calculating the eventual type (a

γ-type) of a term. When we calculate the type of a $\Lambda_{T\pi\gamma}$-term t we first calculate the type of a beginning part of that term (such a beginning part is a segment and will thus have a π-type as its type), say that this results in the π-type π(F,G). Then we calculate the type of the remaining part of t (which is not a segment and thus has a γ-type as its result type), say that this remaining part of t has type γ(H). The product π(F,G) * γ(H) will result in the eventual type of t. With the interpretation of π(F,G) as the type of a beginning part of a term with F as the sequence of non-removable λ's and G as the sequence of non-removable δ's, Definition 1.2.3 should not be too hard to understand. After this definition we shall give an example of calculating the type of a $\Lambda_{T\pi\gamma}$ V-term.

Definition 1.2.3 ($\Lambda_{T\pi\gamma}$ V).

(1) Quasi-types (T_π)
The set of quasi-types T_π is defined as

$$\{\pi(F,G) \mid F,G \in (T\gamma\setminus\{\otimes\})^*\} .$$

(2) Products of quasi-types and types (*)
Let F, G and H be elements of $(T\gamma\setminus\{\otimes\})^*$. The product of a quasi-type and a type is defined as follows

$$\pi(F,G) * \otimes = \otimes$$

$$\pi(F,G) * \gamma(H) = \begin{cases} \gamma(F \,\&\, I), & \text{if } H = \bar{G} \,\&\, I \text{ for some } I \in (T\gamma\setminus\{\otimes\})^* \\ \otimes, & \text{otherwise} \end{cases} .$$

(3) Terms ($\lambda_{T\pi\gamma}$ V)

$$\lambda_{T\pi\gamma} V = \lambda_{T\gamma} V .$$

(4) Types of terms
The function πγ-typ on $\lambda_{T\pi\gamma}$ V is defined inductively for terms t by

(i) $\pi\gamma\text{-typ}(x_f) = f$;

(ii) $\pi\gamma\text{-typ}(\lambda x_f u) = \pi(<f>,\emptyset) * \pi\gamma\text{-typ}(u)$;

(iii) $\pi\gamma\text{-typ}(\delta uv) = \pi(\emptyset,<\pi\gamma\text{-typ}(u)>) * \pi\gamma\text{-typ}(v)$.

(5) The set of correct terms ($\Lambda_{T\pi\gamma}V$)

$$\Lambda_{T\pi\gamma}V = \{t \in \lambda_{T\pi\gamma}V \mid \pi\gamma\text{-typ}(t) \neq \otimes\} \ . \qquad \square$$

A simple example of calculating the $\pi\gamma$-type of a $\Lambda T\pi\gamma$-V term

Consider the following term t

$$\lambda\, x_f\, \delta\, x_g\, \lambda\, x_g\, x_h$$

and assume that $h = \gamma(H)$, where H is some element of $(T\gamma\backslash\{\otimes\})^*$. According to the rules given in Definition 1.2.3, the type of t is calculated as follows

$$\begin{aligned}
&\pi\gamma\text{-typ}(\lambda\, x_f\, \delta\, x_g\, \lambda\, x_g\, x_h) = \\
&= \pi(\langle f\rangle,\emptyset) * \pi\gamma\text{-typ}(\delta\, x_g\, \lambda\, x_g\, x_h) = \\
&= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle g\rangle) * \pi\gamma\text{-typ}(\lambda\, x_g\, x_h) = \\
&= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle g\rangle) * \pi(\langle g\rangle,\emptyset) * \pi\gamma\text{-typ}(x_h) = \\
&= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle g\rangle) * \pi(\langle g\rangle,\emptyset) * \gamma(H) = \\
&= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle g\rangle) * \gamma(\langle g\rangle \,\&\, H) = \\
&= \pi(\langle f\rangle,\emptyset) * \gamma(H) = \\
&= \gamma(\langle f\rangle \,\&\, H)
\end{aligned}$$

and this result is indeed as expected: as mentioned earlier in comment (3) by Definition 1.2.2, $\gamma(\langle f\rangle \,\&\, H)$ is to be interpreted as the type of those terms which map terms of type f to terms of type $\gamma(H)$, and clearly t is a term of that type. Also note that t β-reduces to the term $\lambda\, x_f\, x_{\gamma(H)}$ which, as expected, also has type $\gamma(\langle f\rangle \,\&\, H)$.

The systems $\Lambda_T V$, $\Lambda_{T\gamma} V$ and $\Lambda_{T\pi\gamma} V$ are, though different in their respective descriptions, essentially equivalent in the sense that the expressive power of each of these systems is exactly the same. The reason for deviating from the notations and constructs employed in the original system $\Lambda_T V$ is that we eventually want to give a description of a typing mechanism for $\lambda_T\sigma$, a simple-typed version of the name-free system $\lambda\sigma$. In $\lambda_T\sigma$ we shall construct a completely new kind of types, called ρ-*types*, for segments. What will be shown is that

the employment of π-types, γ-types and the $*$-operation provides for not only an exact but also a concise description of a typing mechanism for segments and segment variables written in name-free notation.

We now proceed by defining a typed version $\lambda_T\xi$ of the name-free system $\lambda\xi$. Types in $\lambda_T\xi$ are elements of $T\gamma$. In order to calculate a type of a name-free term in $\lambda_T\xi$ we introduce the concept of a ξ-*context*, denoted by τ, which is a sequence of elements of $T\gamma\backslash\{\otimes\}$.

Definition 1.2.4 ($\lambda_T\xi$)

(1) Terms ($\lambda_T\xi$)
The set of terms $\lambda_T\xi$ is the smallest set X satisfying

(i) $\xi(n) \in X$, for every $n \in \mathbb{N}$;

(ii) $t \in X \Rightarrow \lambda_f t \in X$, for every $f \in T\gamma\backslash\{\otimes\}$;

(iii) $u,v \in X \Rightarrow \delta uv \in X$.

(2) ξ-Type contexts (τ)
A ξ-context τ is an element of $(T\gamma\backslash\{\otimes\})^*$.
(Note that a type context τ is a function of the form $\tau : [1..\text{length}(\tau)] \to T\backslash\{\otimes\}$.)

(3) The typing function ξ-typ
Let τ be a ξ-type context. The function ξ-typ is defined inductively for $\lambda_T\xi$-terms t by

(i) $\xi\text{-typ}(\xi(n),\tau) = \begin{cases} \tau(n), & \text{if } n \in \text{dom}(\tau) \\ \otimes, & \text{otherwise} \end{cases}$;

(ii) $\xi\text{-typ}(\lambda_f u,\tau) = \pi(<f>,\emptyset) * \xi\text{-typ}(u,<f> \,\&\, \tau)$;

(iii) $\xi\text{-typ}(\delta uv,\tau) = \pi(\emptyset,<\xi\text{-typ}(u,\tau)>) * \xi\text{-typ}(v,\tau)$.

(4) The set of correct terms
Let τ be a ξ-type context. The set of correct $\lambda_T\xi$-terms with respect to τ is

$$\{t \in \lambda_T\xi \mid \xi\text{-typ}(t,\tau) \neq \otimes\}.$$ □

Remarks.

(1) In $\lambda_T\xi$ we just write $\lambda_f,\lambda_g,\lambda_h,\ldots$ instead of $\lambda x_f,\lambda x_g,\lambda x_h,\ldots$ (the names of variables are dropped).

(2) The type of an occurrence of a variable $\xi(n)$ in a $\lambda_T\xi$-term t is found as follows. First we form the tree representation $\hat{t}$ of t, then we descend from that occurrence of $\xi(n)$ in $\hat{t}$ towards the root of the tree and the n-th lambda, say λ_f, is the lambda that binds this occurrence of $\xi(n)$ and the type f attached to this lambda is the type of $\xi(n)$. (If the total number of λ's encountered on the root path of this occurrence of $\xi(n)$ is less than n (implying that this occurrence of $\xi(n)$ is free) then the type context will see to it that this occurrence of $\xi(n)$ is suitably typed.)

(3) The correspondence between name-carrying terms in $\Lambda_{T\pi\gamma}V$ and name-free terms in $\lambda_T\xi$ is as follows. If t is a $\Lambda_{T\pi\gamma}V$-term not containing free occurrences of variables then we have the following correspondence

$$\pi\gamma\text{-typ}(t) = \xi\text{-typ}(\tilde{t},\emptyset) ,$$

where $\tilde{t}$ denotes the name-free equivalent of t. If t contains free occurrences of variables then we have the correspondence

$$\pi\gamma\text{-typ}(t) = \xi\text{-typ}(\tilde{t},\tau) ,$$

where the ξ-context τ is such that it is of sufficient length and sees to it that all free occurrences of variables in $\tilde{t}$ are typed in the same way as they were typed in t.

We now move on to the definition of the full $\lambda_T\sigma$-system by constructing types for segments. In order to do so we introduce a new kind of types, called ρ-types, for segments. A ρ-type has three parameters and is written as $\rho(F,G,H)$, where F, G and H are sequences of γ- and, possibly, ρ-types. The extra parameter H has a purely administrative function; intuitively H is the sequence of *all* types attached to the λ's, including those hidden inside segment variables, lying on the main branch of the tree representation of the segment in question. The sequences F and G have the same meaning as before in the case of the quasi-type $\pi(F,G)$, namely the sequence of non-removable λ's and

the sequence of non-removable δ's, respectively. We need such an extra parameter H in $\rho(F,G,H)$ in order to determine the type of those variables which refer to a λ hidden in a segment variable, a situation which we now explain. Suppose that we have a $\lambda_T\sigma$-term t in which we have a segment $s\omega(\psi)$ and an occurrence of a segment variable $\sigma(n,m)$ which abbreviates $s\omega(\psi)$ in t. From $\sigma(n,m)$ we see that $s\omega(\psi)$ has m ($m \geq 0$) λ's lying on the main branch of its tree representation: these m λ's are hidden inside $\sigma(n,m)$ and they can be referred to by variables in t occurring to the right of $\sigma(n,m)$. In order to be able to type those variables which refer to one of the λ's hidden inside $\sigma(n,m)$ we inspect the third parameter H of the type, say $\rho(F,G,H)$, of $s\omega(\psi)$. Suppose that the m λ's lying on the main branch of the tree representation of $s\omega(\psi)$ occur in the order $\lambda_{h_1},\ldots,\lambda_{h_{m-1}},\lambda_{h_m}$, then H shall be the sequence $\langle h_m\rangle$ & $\langle h_{m-1}\rangle$ & ... & $\langle h_1\rangle$. If a variable in t refers to the i-th ($0 \leq i \leq m$) λ (from the right) hidden inside $\sigma(n,m)$ then it will be typed by the i-th member h_i of H. Our definition of $\lambda_T\sigma$ will also take into account the reallocational effects that segmaps ψ have on references to λ's lying on the main branch of the segments in question.

We now give our definition, which at first sight might be a bit hard to understand. We shall give an example of calculating the type of a $\lambda_T\sigma$-term which should help clarify the rules stated in Definition 1.2.5. We note that the construct $\pi(F,G)$, given below, is the same construct $\pi(F,G)$ as in Definition 1.2.3: it is an intermediate construct used for evaluating the product of a number of types in order to evaluate the eventual type of a term (including segments), which is either a γ-type or a ρ-type (but never a π-type).

Definition 1.2.5 ($\lambda_T\sigma$).

(1) Types (T)

The set of types T is the smallest set X satisfying

(i) $\otimes \in X$;

(ii) $\forall F \in (X\setminus\{\otimes\})^* : \gamma(F) \in X$;

(iii) $\forall F,G,H \in (X\setminus\{\otimes\})^* : \rho(F,G,H) \in X$.

(Note that $\gamma(\emptyset) \in X$ and $\rho(\emptyset,\emptyset,\emptyset) \in X$.)

(2) Quasi-types (T_π)

The set of quasi-types T_π is defined as

$$\{\pi(F,G) \mid F,G \in (T\backslash\{\otimes\})^*\} .$$

(3) Products of quasi-types and types (*)

Let F, G, H, I and J be elements of $(T\backslash\{\otimes\})^*$. The product of a quasi-type and a type is defined as follows

$$\pi(F,G) * \otimes = \otimes$$

$$\pi(F,G) * \gamma(H) = \begin{cases} \gamma(F \,\&\, I) , & \text{if } H = \bar{G} \,\&\, I \text{ for some } I \in (T\backslash\{\otimes\})^* \\ \otimes , & \text{otherwise} \end{cases}$$

$$\pi(F,G) * \rho(H,I,J) = \begin{cases} \rho(F \,\&\, K,I,J) , & \text{if } H = \bar{G} \,\&\, K \text{ for some } K \in (T\backslash\{\otimes\})^* \\ \rho(F,K \,\&\, I,J) , & \text{if } G = K \,\&\, \bar{H} \text{ for some } K \in (T\backslash\{\otimes\})^* \\ \otimes , & \text{otherwise} \end{cases} .$$

(4) Terms ($\lambda_T\sigma$)

The set of $\lambda_T\sigma$-terms is the smallest set X satisfying

(i) $\xi(n) \in X$, for every $n \in \mathbb{N}$;

(ii) if ψ is a segmap then $\omega(\psi) \in X$;

(iii) if $u \in X$ and $f \in T\backslash\{\otimes\}$ then $\lambda_f u \in X$;

(iv) if $u \in X$ then $\sigma(n,m)u \in X$, for every $n \in \mathbb{N}$ and $m \in \mathbb{N} \cup \{0\}$;

(v) if $u,v \in X$ then $\delta uv \in X$.

(5) Type contexts

A type context is an element of $(T\backslash\{\otimes\})^*$.

(6) The typing function (typ)

Let τ be a type context. The function typ is defined inductively for $\lambda_T\sigma$-terms t by

(i) $$\mathrm{typ}(\xi(n),\tau) = \begin{cases} \tau(n) , & \text{if } n \in \mathrm{dom}(\tau) \text{ and } \tau(n) \text{ is a } \gamma\text{-type} \\ \otimes , & \text{otherwise} \end{cases} ;$$

(ii) $$\mathrm{typ}(\omega(\psi),\tau) = \begin{cases} \rho(\emptyset,\emptyset,\tau \circ \psi) , & \text{if } \mathrm{rge}(\psi) \subseteq \mathrm{dom}(\tau) \\ \otimes , & \text{otherwise} \end{cases} ;$$

(iii) $typ(\lambda_f u,\tau) = \pi(<f>,\emptyset) * typ(u,<f> \& \tau)$;

(iv) $typ(\sigma(n,m)u,\tau) =$

$$\begin{cases} \pi(F,G) * typ(u,H \& \tau) \text{ , if } n \in dom(\tau) \text{ and } \tau(n) \text{ is a } \rho\text{-type of} \\ \qquad \text{the form } \rho(F,G,H), \text{ where } H \text{ is a sequence of length } m \text{ ;} \\ \otimes \qquad\qquad\qquad\qquad \text{, otherwise} \end{cases}$$

(v) $typ(\delta uv,\tau) = \pi(\emptyset,<typ(u,\tau)>) * typ(v,\tau)$;

(vi) <u>The set of correct terms</u>

Let τ be a type context. The set of correct $\lambda_T\sigma$-terms with respect to τ is

$$\{t \in \lambda_T\sigma \mid typ(t,\tau) \neq \otimes\} .$$

We now give a further explanation of the rules stated in Definition 1.2.5, and we shall do so by means of a non-trivial example in which all of the features for calculating γ- and ρ-types are incorporated. In this example we shall employ the following notational conventions

$$f_1 * \ldots * f_{n-1} * f_n = (f_1 * (f_2 * \ldots * (f_{n-2} * (f_{n-1} * f_n))\ldots)$$

(association to the right)

$$<f_1,f_2,\ldots,f_n> = <f_1> \& <f_2> \& \ldots \& <f_n> .$$

Consider the following term t written in tree form

$$\begin{array}{l} \qquad\qquad\qquad\qquad\quad /\,\xi(2) \qquad\qquad /\,\xi(1) \\ \qquad\qquad\quad /\,\lambda_g - \delta - \lambda_h - \lambda_i - \delta - \omega(\psi) \\ \lambda_f - \delta - \lambda_j - \sigma(1,3) - \xi(1) \end{array} \qquad (t\hat{})$$

where f, g, h, i and j are certain elements of $T\backslash\{\otimes\}$ and ψ is a permutation of the interval [1..3] defined by $\psi(1) = 2$, $\psi(2) = 3$ and $\psi(3) = 1$. According to the rules given in Definition 1.2.5, the type of t with respect to the empty context $\emptyset$ is calculated, step by step, as follows

$$typ(\lambda_f \delta \lambda_g \delta \xi(2) \lambda_h \lambda_i \delta \xi(1) \omega(\psi) \lambda_j \sigma(1,3) \xi(1),\emptyset) =$$

$$= \pi(<f>,\emptyset) * typ(\delta \lambda_g \delta \xi(2) \lambda_h \lambda_i \delta \xi(1) \omega(\psi) \lambda_j \sigma(1,3) \xi(1),<f>) =$$

$$= \pi(<f>,\emptyset) * \pi(\emptyset,<typ(u,<f>)>) * typ(\lambda_j \sigma(1,3) \xi(1),<f>)$$

where u is the segment $\lambda_g \delta \xi(2) \lambda_h \lambda_i \delta \xi(1) \omega(\psi)$, or in tree form

$$\lambda_g - \delta \overset{\xi(2)}{-} \lambda_h - \lambda_i - \delta \overset{\xi(1)}{-} \omega(\psi) \ . \qquad (u\hat{})$$

First we calculate typ(u,<f>):

$\mathrm{typ}(\lambda_g \delta \xi(2) \lambda_h \lambda_i \delta \xi(1) \omega(\psi),\langle f\rangle) =$

$= \pi(\langle g\rangle,\emptyset) * \mathrm{typ}(\delta \xi(2) \lambda_h \lambda_i \delta \xi(1) \omega(\psi),\langle g,f\rangle) =$

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle \mathrm{typ}(\xi(2),\langle g,f\rangle)\rangle) * \mathrm{typ}(\lambda_h \lambda_i \delta \xi(1) \omega(\psi),\langle g,f\rangle) =$

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle f\rangle) * \mathrm{typ}(\lambda_h \lambda_i \delta \xi(1) \omega(\psi),\langle g,f\rangle) =$

(if f is a γ-type, otherwise the product is equal to $\otimes$)

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle f\rangle) * \pi(\langle h\rangle,\emptyset) * \pi(\langle i\rangle,\emptyset) *$

$* \mathrm{typ}(\delta \xi(1) \omega(\psi),\langle i,h,g,f\rangle) =$

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle f\rangle) * \pi(\langle h\rangle,\emptyset) * \pi(\langle i\rangle,\emptyset) *$

$* \pi(\emptyset,\langle \mathrm{typ}(\xi(1),\langle i,h,g,f\rangle)\rangle) * \mathrm{typ}(\omega(\psi),\langle i,h,g,f\rangle) =$

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle f\rangle) * \pi(\langle h\rangle,\emptyset) * \pi(\langle i\rangle,\emptyset) * \pi(\emptyset,\langle i\rangle) *$

$* \mathrm{typ}(\omega(\psi),\langle i,h,g,f\rangle) =$

(if i is a γ-type, otherwise the product is equal to $\otimes$)

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle f\rangle) * \pi(\langle h\rangle,\emptyset) * \pi(\langle i\rangle,\emptyset) * \pi(\emptyset,\langle i\rangle) *$

$* \rho(\emptyset,\emptyset,\langle h,g,i\rangle) =$

(note that the composition of the sequence <i,h,g,f> with the segmap ψ yields not only a permuted but also reduced sequence <h,g,i> of <i,h,g,f>)

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle f\rangle) * \pi(\langle h\rangle,\emptyset) * \pi(\langle i\rangle,\emptyset) * \rho(\emptyset,\langle i\rangle,\langle h,g,i\rangle) =$

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle f\rangle) * \pi(\langle h\rangle,\emptyset) * \rho(\langle i\rangle,\langle i\rangle,\langle h,g,i\rangle) =$

$= \pi(\langle g\rangle,\emptyset) * \pi(\emptyset,\langle f\rangle) * \rho(\langle h,i\rangle,\langle i\rangle,\langle h,g,i\rangle) =$

$= \pi(\langle g\rangle,\emptyset) * \rho(\langle i\rangle,\langle i\rangle,\langle h,g,i\rangle) =$

(if f = h, otherwise the product is equal to $\otimes$)

$= \rho(\langle g,i\rangle,\langle i\rangle,\langle h,g,i\rangle)$

and this is indeed as expected: the segment u has two non-removable abstractors (λ_g and λ_i) lying on the main branch of its tree; it has one non-removable applicator with i as the type of its argument; it has a total number of three abstractors lying on the main branch of its tree, which, due to the reallocational effect of the segmap ψ, are referred to in the order λ_h, λ_g and λ_i (from the right).

Now that we have evaluated typ(u,<f>) we can proceed with calculating typ(t,$\emptyset$):

typ(t,$\emptyset$) =

$= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle\rho(\langle g,i\rangle,\langle i\rangle,\langle h,g,i\rangle)\rangle) * typ(\lambda_j\, \sigma(1,3)\, \xi(1),\langle f\rangle) =$

$= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle\rho(\langle g,i\rangle,\langle i\rangle,\langle h,g,i\rangle)\rangle) * \pi(\langle j\rangle,\emptyset) *$

$* typ(\sigma(1,3)\, \xi(1),\langle j,f\rangle) =$ (1)

$= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle\rho(\langle g,i\rangle,\langle i\rangle,\langle h,g,i\rangle)\rangle) * \pi(\langle j\rangle,\emptyset) * \pi(F,G) *$

$* typ(\xi(1),\langle h_1,h_2,h_3,j,f\rangle) =$

(where $j = \rho(F,G,\langle h_1,h_2,h_3\rangle)$ for some $F,G \in (T\backslash\{\otimes\})^*$ and $h_1,h_2,h_3 \in T\backslash\{\otimes\}$ (cf. clause (6)(iv)), otherwise the product if equal to $\otimes$)

$= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle\rho(\langle g,i\rangle,\langle i\rangle,\langle h,g,i\rangle)\rangle) * \pi(\langle j\rangle,\emptyset) * \pi(F,G) * h_1 =$

(if h_1 is a γ-type, otherwise the product is equal to $\otimes$)

$= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle\rho(\langle g,i\rangle,\langle i\rangle,\langle h,g,i\rangle)\rangle) * \pi(\langle j\rangle,\emptyset) * \gamma(F \,\&\, H_1) =$

(where $h_1 = \gamma(\bar{G} \,\&\, H_1)$ for some $H_1 \in (T\backslash\{\otimes\})^*$ (cf. clause (3)(ii)), otherwise the product is equal to $\otimes$)

$= \pi(\langle f\rangle,\emptyset) * \pi(\emptyset,\langle\rho(\langle g,i\rangle,\langle i\rangle,\langle h,g,i\rangle)\rangle) * \gamma(\langle j\rangle \,\&\, F \,\&\, H_1) =$

$= \pi(\langle f\rangle,\emptyset) * \gamma(F \,\&\, H_1) =$

(if $j = \rho(F,G,\langle h_1,h_2,h_3\rangle) = \rho(\langle g,i\rangle,\langle i\rangle,\langle h,g,i\rangle)$, i.e. if $F = \langle g,i\rangle$, $G = \langle i\rangle$, $h_1 = h$, $h_2 = g$, $h_3 = i$, otherwise the product is equal to $\otimes$)

$= \gamma(\langle f\rangle \,\&\, F \,\&\, H_1) =$

$= \gamma(\langle f,g,i\rangle \,\&\, H_1)$

(by definition of j)

and this is indeed the expected result: t is a non-segment and therefore its type is a γ-type; if we assume that $H = \langle i\rangle \,\&\, H_1 = \langle i\rangle \,\&\, \&\, \langle h_1,\ldots,h_n\rangle$ for certain $h_1,\ldots,h_n \in T\backslash\{\otimes\}$, then the non-removable abstractors lying on the main branch of the tree representation of t occur in the order $\lambda_f,\lambda_g,\lambda_i,\lambda_{h_1},\ldots,\lambda_{h_n}$, since the non-removable abstractors hidden in $\sigma(1,3)$ are λ_g and λ_i, and the first type i in the sequence $\langle i,h_1,\ldots,h_n\rangle$ is removed because the type of the argument $\xi(1)$ of the last applicator occurring in the segment

$$\lambda_g - \delta \overset{\xi(2)}{/} - \lambda_h - \lambda_i - \delta \overset{\xi(1)}{/} - \omega(\psi) \qquad (u\hat{})$$

is equal to i (remember that the last variable $\xi(1)$ occurring in t has type $\gamma(\langle i,h_1,\ldots,h_n\rangle)$ which means that the first non-removable abstractor of the term that this occurrence of $\xi(1)$ intends to abbreviate would be λ_i, and that this λ_i matches the $\delta\,\xi(1)$-part in the segment u).

Note also that t β-reduces to the following term written in tree form

$$\lambda_f - \lambda_g - \delta \overset{\xi(2)}{/} - \lambda_h - \lambda_i - \delta \overset{\xi(1)}{/} - \xi(2)\ ,$$

where we have substituted the segment u for $\sigma(1,3)$ (the reference number 1 in the last variable $\xi(1)$ in t has been changed to 2 because of the reallocational effect of the segmap ψ). This new term can be β-reduced once more, resulting in

$$\lambda_f - \lambda_g - \lambda_i - \delta - \xi(3) \ , \qquad \xi(1)$$

wherewe have substituted an updated version of the first occurrence of the variable $\xi(2)$ for the second occurrence of $\xi(2)$ (which was bound by the abstractor λ_h of the redex). The variable $\xi(1)$ in this term has type i, and the variable $\xi(3)$ has type $f = h = \gamma(<i,h_1,\ldots,h_n>)$; therefore the type of the whole term is equal to $\gamma(<f,g,i> \ \& \ <h_1,\ldots,h_n>)$, which is the same type as we have calculated for t: an expected result. In general, one would expect the type of a term and its β-reduct to be the same. This property of equality of types for terms and their β-reducts with respect to a certain context is called *the closure property*. A proof of the closure property for $\lambda_T\sigma$ is given in Chapter 4. We note that in Chapter 4 we shall also define the product of two quasi-types and furthermore show that this extended version of the $*$-operation is associative, i.e. $(f * g) * h = f * (g * h)$ for all quasi-types f,g and quasi-types and types h. Products of quasi-types and the associativity of the $*$-operation will prove to be useful for facilitating the calculations of types of $\lambda_T\sigma$-terms.

1.3. Reduction and related properties

The language theory of λ-calculus is concerned with the syntactical structure of terms and properties of reduction relations. The study of the β-reduction rule is of particular interest in this respect. This rule tells us how to compute the value that a function takes when applied to a certain argument. In this section we shall define basic relations on some abstract set X by starting from an abstract reduction relation on X denoted by $\rightarrow_R$. Such a structure $<X,\rightarrow_R>$ provides an abstract framework in which reduction relations can be discussed for term-rewriting systems in general.

Notions of reduction

The following definitions are taken from Barendregt [81,pp. 50 - 58].

Definition 1.3.1.

A *notion of reduction* on a set X is a binary relation on X. □

Definition 1.3.2.

Let $\to_R$ be a notion of reduction on X and $t,u,v \in X$.

(i) $\overset{*}{\to}_R$ is the transitive reflexive closure of $\to_R$ defined by

(1) $t \to_R u \Rightarrow t \overset{*}{\to}_R u$;

(2) $t \overset{*}{\to}_R t$;

(3) $t \overset{*}{\to}_R u, \quad u \overset{*}{\to}_R v \Rightarrow t \overset{*}{\to}_R v$.

(ii) $=_R$ is the equivalence relation generated by $\overset{*}{\to}_R$ defined by

(1) $t \overset{*}{\to}_R u \Rightarrow t =_R u$;

(2) $t =_R u \Rightarrow u =_R t$;

(3) $t =_R u, \quad u =_R v \Rightarrow t =_R v$.

(iii) $t \downarrow^*_R u$ iff $\exists v : t \overset{*}{\to}_R v \wedge u \overset{*}{\to}_R v$.

(iv) The basic relations derived from $\to_R$ are pronounced as follows:

$t \overset{*}{\to}_R u$: t R-*reduces to* u or u *is an* R-*reduct of* t ;

$t \to_R u$: t R-*reduces to* u *in one step* ;

$t =_R u$: t *is* R-*convertible to* u ;

$t \downarrow^*_R u$: t *and* u *have a common* R-*reduct* . □

The relations $\overset{*}{\to}_R$ and $=_R$ have been introduced inductively. Therefore properties about these relations can be proved inductively. Such inductive proofs are called proofs by induction on the generation of these relations.

For the remainder of this section let X denote some set and let $\to_R$ be a notion of reduction on X. We shall use the meta-symbols t,u,v,w,... to range over elements of X (called *terms*).

Definition 1.3.3.

(i) A term t is called an R-*normal form* (R-*nf*) if

$$\neg \exists u : t \to_R u .$$

(ii) A term u *is* an R-*nf of* t (or t *has the* R-*nf* u) if u is an R-nf and $t \overset{*}{\to}_R u$. □

Definition 1.3.4.

(i) Let $\succ$ be a binary relation on X. Then $\succ$ satisfies the *diamond property* if

$$\forall t,u,v\ (t \succ u \wedge t \succ v \Rightarrow \exists w : u \succ w \wedge v \succ w)$$

see Figure 1.3.1.

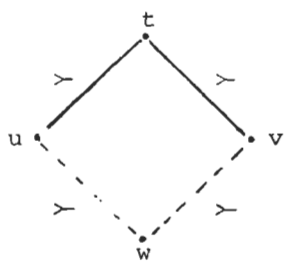


FIG. 1.3.1.

(ii) A notion of reduction $\to_R$ is said to be *Church-Rosser* (CR) if $\overset{*}{\to}_R$ satisfies the diamond property. □

Theorem 1.3.1.

If $\to_R$ is CR then a term t can have at most one R-nf.

Proof: Suppose that u_1, u_2 are both R-nf's of t. From $\to_R$ being CR it follows that there exists a term v such that $u_1 \overset{*}{\to}_R v$ and $u_2 \overset{*}{\to}_R v$. But since $\overset{*}{\to}_R$ is the transitive reflexive closure of $\to_R$ it holds for all R-nf's w that if $w \overset{*}{\to}_R w'$ then $w = w'$, and therefore $u_1 = v = u_2$. □

Theorem 1.3.2.

If $\to_R$ is CR then

$$t =_R u \Rightarrow \exists v : t \overset{*}{\to}_R v \wedge u \overset{*}{\to}_R v\ .$$

Proof: By induction on the generation of $=_R$. □

Definition 1.3.5.

An R-*reduction path* is a finite or infinite sequence $t_0, t_1, t_2, \ldots$ such that $t_0 \to_R t_1 \to_R t_2 \to_R \ldots$. □

Conventions.

(i) The meta-symbol ν ranges over reduction paths.

(ii) The reduction path in Definition 1.3.5 *starts* with t_0. If there is a last term t_n in ν, then ν *ends* with t_n. In that case we say that ν is a reduction path *from* t_0 *to* t_n. □

Definition 1.3.6.

Let $t \in X$.

(i) t R-*normalizes* (R-N(t)) if t has an R-nf.

(ii) t R-*strongly normalizes* (R-SN(t)) if there is no infinite R-reduction path starting with t.

(iii) t is R-*infinite* (R-∞(t)) if not R-SN(t).

(iv) $\to_R$ is *normalizing* (N) if $\forall t \in X : \text{R-N}(t)$.

(v) $\to_R$ is *strongly normalizing* (SN) if $\forall t \in X : \text{R-SN}(t)$. □

Definition 1.3.7.

A notion of reduction $\to_R$ on X is said to be *weakly Church-Rosser* (WCR) if

$$\forall t,u,v\ (t \to_R u \wedge t \to_R v \Rightarrow \exists w : u \overset{*}{\to}_R w \wedge v \overset{*}{\to}_R w) .$$ □

Theorem 1.3.3 (Newman [42]).

For notions of reduction $\to_R$ one has

$$\text{SN} \wedge \text{WCR} \Rightarrow \text{CR} .$$

Proof: The following elegant proof is taken from Barendregt [81], p. 58.

By SN each term R-reduces to an R-nf. It suffices to show that this R-nf is unique. Call a term t *ambiguous* if t R-reduces to two distinct R-nf's, say t_1 and t_2. If t is ambiguous then there exists a term u such that $t \to_R u$ and u ambiguous, which we now show. The following

two figures suggest how t can reduce to t_1 and t_2.

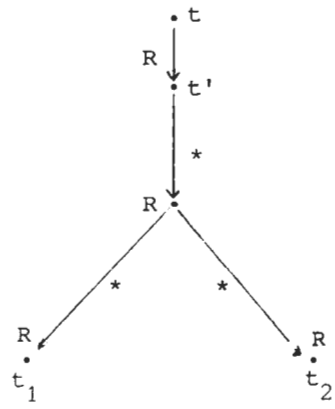


FIG. 1.3.2.

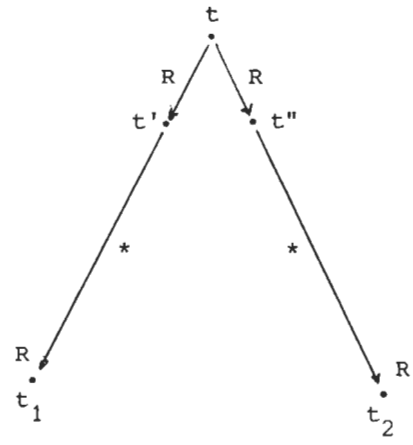


FIG. 1.3.3.

In the case of Figure 1.3.2 it is immediately clear that the ambiguous term u exists by taking t' for u. In the case of Figure 1.3.3 it follows from WCR that t' and t" have a common reduct t"' and, by SN, t"' has an R-nf t_3 as indicated in the figure below.

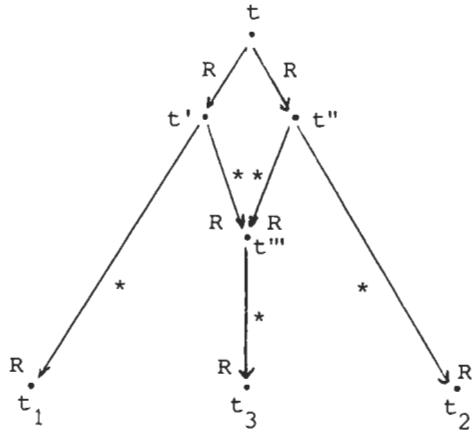


FIG 1.3.4.

From $t_1 \neq t_2$ it follows that either $t_3 \neq t_1$ or $t_3 \neq t_2$. If $t_3 \neq t_1$ then we can take t' for u, and if $t_3 \neq t_2$ then we can take t" for u. Since all ambiguous terms R-reduce in one step to another ambiguous term, we have obtained a contradiction with SN, hence ambiguous terms do not exist. □

Theorem 1.3.4.

Let $\rightarrow_R$ be a notion of reduction that is both CR and N and let $t,u \in X$. Then

$$t =_R u \Leftrightarrow R\text{-nf}(t) = R\text{-nf}(u)$$

where R-nf(t) and R-nf(u) denote the R-nf's of t and u.

Proof: First note that t and u have unique R-nf's by CR, N and Theorem 1.3.1.

(i) $\Rightarrow$: if $t =_R u$ then by Theorem 1.3.2 there exists a term v such that $t \overset{*}{\rightarrow}_R v$ and $u \overset{*}{\rightarrow}_R v$. This term v also has a unique R-nf, say v_0, and since $t \overset{*}{\rightarrow}_R v_0$ and $u \overset{*}{\rightarrow}_R v_0$ it follows immediately that

$$R\text{-nf}(t) = v_0 = R\text{-nf}(u) \ .$$

(ii) $\Leftarrow$: trivial.

Remark. A consequence of Theorem 1.3.4 is the following decidability result: if one has an effective procedure for computing R-nf's of terms then one also has an effective procedure for determining whether $t =_R u$ holds or not, for all terms t,u.

2. BASIC NOTIONS AND RESULTS

The usual set-theoretic notation will be used in the metalanguage, including the abbreviations $\forall$ (for all), $\exists$ (there exists), $\Rightarrow$ (if... then ...), $\Leftrightarrow$ (if and only if), $\wedge$ (and), $\vee$ (or) and $\neg$ (not). We shall adopt the following conventions concerning the natural numbers

$$\mathbb{N} = \{1,2,3,\ldots\}$$

$$\mathbb{M} = \mathbb{N} \cup \{0\}$$

$\mathbb{N}_k = \{n \in \mathbb{N} \mid n \leq k\}$, for every $k \in \mathbb{M}$. Hence, $\mathbb{N}_0 = \emptyset$.

The set of mappings from the set A to the set B is denoted by $[A \to B]$. Domain and range of a function f are denoted by dom(f) and rge(f), respectively. If f and g are functions then the composition $f \circ g$ of f and g is the function with domain $\{x \in dom(g) \mid g(x) \in dom(f)\}$ and, for every $x \in dom(f \circ g)$, $f \circ g(x) = f(g(x))$.
The set of permutations of $\mathbb{N}_k$ is denoted by Perm(k) and id(k) denotes the identity map on $\mathbb{N}_k$. The set $\cup\{perm(k) \mid k \in \mathbb{M}\}$ is denoted by Perm. Furthermore, for every function f with $rge(f) \subseteq \mathbb{N}$, we have the function f - 1 defined by

$$dom(f - 1) = dom(f)$$

and

$$\forall x \in dom(f) : f - 1(x) = f(x) - 1 .$$

2.1. Sequences

Definition 2.1.
An *alphabet* is a non-empty set C.

$$C^* = \cup\{[\mathbb{N}_k \to C] \mid k \in \mathbb{M}\} ;$$

$$C^+ = C^* \setminus \{\emptyset\} .$$

C^* is called the set of C-*sequences* and C^+ is called the set of *non-empty* C-sequences. If f is a C-sequence and $dom(f) = \mathbb{N}_k$ then k is called the *length* of f and is denoted by L(f). Elements of $[\mathbb{N}_1 \to C]$ are called C-symbols. If f is a C-sequence of length $k \geq 1$, then the *first symbol* of f, denoted by FS(f), is defined as f(1) and the *last symbol* of f, denoted by LS(f), is defined as f(k). □

Definition 2.2.

If f is a C-sequence of length k then $\bar{f}$ is the C-sequence of length k defined by

$$\forall i \in \mathbb{N}_k : \bar{f}(i) = f(k - i + 1) .$$

$\bar{f}$ is called the *reversed* sequence of f. □

Definition 2.3.

If f and g are C-sequences and L(f) = k and L(g) = m, then the C-sequence f & g of length k + m is defined by

$$f \,\&\, g(i) = \begin{cases} f(i) , & \text{if } i \in \mathbb{N}_k \\ g(i - k) , & \text{if } i \in \mathbb{N}_{k+m} \setminus \mathbb{N}_k \end{cases} .$$

f & g is called the *concatenation* of the C-sequences f and g. □

Note that concatenation is an associative operation on C-sequences; i.e. (f & g) & h = f & (g & h), for every f, g and h. We shall make frequent use of this property of concatenation since we can omit parentheses and write f & g & h without fearing ambiguities.

Remark. Whenever it is clear which C-sequences f and g are being concatenated, we shall write fg instead of explicitly writing f & g. Furthermore, if it is clear which alphabet C is being used to form C-sequences, we shall often drop the C and tacitly speak about sequences instead of C-sequences. □

2.2. Language definition of the formal system λσ

Definition 2.4.

Let C be a countable set and let C_1 denote the set of C-symbols. We introduce a set of mappings which are considered fixed from now on.

(i) ξ is an injection from $\mathbb{N}$ into C_1 ;

(ii) ω is an injection from Perm into C_1 ;

(iii) λ is an element of C_1 ;

(iv) σ is an injection from $\mathbb{N} \times \mathbb{M}$ into C_1 ;

(v) δ is an element of C_1 ;

(vi) $rge(\xi)$, $rge(\omega)$, $\{\lambda\}$, $rge(\sigma)$ and $\{\delta\}$ are mutually disjoint subsets of C. □

Definition 2.5.

Λ is the smallest set X satisfying

(i) $\xi(n) \in X$, for every $n \in \mathbb{N}$;

(ii) $\omega(\psi) \in X$, for every $\psi \in Perm$;

(iii) if $t \in X$ then $\lambda \,\&\, t \in X$;

(iv) if $t \in X$ then $\sigma(p) \,\&\, t \in X$, for every $p \in \mathbb{N} \times \mathbb{M}$;

(v) if $t,u \in X$ then $\delta \,\&\, t \,\&\, u \in X$. □

Elements of Λ are called $\lambda\sigma$-*terms* (or *terms*, for short). Elements of $rge(\xi) \cup rge(\sigma)$ are called *variables*; elements of $rge(\xi)$ are called ξ-variables and elements of $rge(\sigma)$ are called σ-variables. We shall use the meta-symbols $\eta,\eta',\eta'',\ldots$ to range over variables. For every ξ-variable η there is exactly one $n \in \mathbb{N}$ such that $\eta = \xi(n)$, and for every σ-variable η there is exactly one pair $(n,m) \in \mathbb{N} \times \mathbb{M}$ such that $\eta = \sigma(n,m)$. In both cases the number n is called the *reference number* of η. These reference numbers determine the λ, if any, that binds an occurrence of a variable in a term. Terms with last symbol $\omega(\psi)$, for some $\psi \in Perm$, are called *segments*. Elements of Perm are called *segment mappings* (or *segmaps*, for short). Note that terms are written in prefix-notation: each $\xi(n)$ and $\omega(\psi)$ has arity 0; λ and each $\sigma(p)$ has arity 1; δ has arity 2.

For an informal description of $\lambda\sigma$-terms we refer to the introduction. Furthermore, we note that frequent use will be made of the 1-1 correspondence between $\lambda\sigma$-terms and their tree representations. The reason for this is that tree representations of $\lambda\sigma$-terms, as described in the introduction, facilitate the reading (parsing) of these terms. We now proceed by introducing some important concepts concerning variables.

Definition 2.6.

Var(t) is defined inductively for terms t by

(i) $Var(\xi(n)) = \{\xi(n)\}$;

(ii) $Var(\omega(\psi)) = \emptyset$;

(iii) $Var(\lambda u) = Var(u)$;

(iv) $Var(\sigma(p)u) = \{\sigma(p)\} \cup Var(u)$;

(v) $Var(\delta uv) = Var(u) \cup Var(v)$. □

Var(t) is called the set of variables of t. If $\eta \in Var(t)$ then we say that the variable η *occurs in* t.

Remark. If a variable η occurs in a term t then it can occur in t at different places. Sometimes we would like to speak only of some specific occurrence of η in t; i.e. we would like to speak of the variable η occurring in t at a specific place. We shall reserve the informal term "*an occurrence of* η *in* t" when we wish to refer to a variable η occurring in t at a specific place. Following this terminology we can say that a variable η can occur in a term t, but, at the same time, there may also be different occurrences of η in t.

Definition 2.7.

Let η be a variable.

The set $D(\eta,t)$ is defined inductively for terms t by

(i) $D(\eta,\xi(n)) = \begin{cases} \{n\}, & \text{if } \eta = \xi(n) \\ \emptyset, & \text{if } \eta \neq \xi(n) \end{cases}$;

(ii) $D(\eta,\omega(\psi)) = \emptyset$;

(iii) $D(\eta,\lambda u) = \{k-1 \mid k \in D(\eta,u)\}$;

(iv) $D(\eta,\sigma(n,m)u) = \begin{cases} \{n\} \cup \{k-m \mid k \in D(\eta,u)\}, & \text{if } \eta = \sigma(n,m) \\ \{k-m \mid k \in D(\eta,u)\}, & \text{if } \eta \neq \sigma(n,m) \end{cases}$;

(v) $D(\eta,\delta uv) = D(\eta,u) \cup D(\eta,v)$. □

Note that $D(\eta,t) \subseteq \mathbb{Z}$.

If $k \in D(\eta,t)$ then we say that η *occurs at reference depth* k *in* t. If η occurs at some reference depth in t then it occurs in t in the sense defined above. Formally:

If $\eta \notin Var(t)$ then $D(\eta,t) = \emptyset$.

This statement can easily be proved by induction on L(t).

Definition 2.8.
Let t be a term and let η be a variable. We say that η has an *external reference occurrence* in t if there exists a $k > 0$ such that $k \in D(\eta,t)$, and we say that η has an *internal reference occurrence* in t if there exists a $k \leq 0$ such that $k \in D(\eta,t)$. □

Example. Consider the following term t

$$\lambda - \lambda - \delta \overset{\lambda - \xi(6)}{/} - \lambda - \sigma(3,1) - \xi(2) \ .$$

The variables $\xi(6)$, $\sigma(3,1)$ and $\xi(2)$ occur at reference depths 3, 0 and -2 respectively. Furthermore, $\xi(6)$ has an external reference (i.e. $\xi(6)$ is bound by a λ outside of t) and the variables $\sigma(3,1)$ and $\xi(2)$ have internal references in t (i.e. both variables are bound by a λ inside of t). Note that if a variable $\xi(n)$ or $\sigma(n,m)$ occurs at reference depth k in a term t, then the reference number n will usually not be equal to k.

Remark. If a variable η occurs at different places in some term t then it can well be the case that η occurs at two different reference depths k, k' in t. It can even be the case that η has both an external as well as an internal reference in t. Each specific occurrence of η in t, though, has exactly one corresponding reference depth k in t. In informal discussions we will often speak of "*the reference depth* k *of an occurrence of* η *in* t", such to focus our attention on a specific occurrence of η in t instead of taking all occurrences of η in t into account. In the same way we shall speak of "*an external (internal) reference occurrence of* η *in* t".

2.3. Reference mappings

Reference mappings were introduced by N.G. de Bruijn in his paper de Bruijn [78,a] in order to describe the possible effects that a β-

reduction of a name-free term t can have on the variables occurring in t; more specifically: reference mappings see to it that a suitable updating of reference numbers of variables takes place after having performed some β-reduction on t. Reference mappings (or *refmaps* for short) are elements of $[\mathbb{N} \to \mathbb{N}]$, and for each refmap μ we also have a mapping $\underline{\mu}$ which works on terms. The effect that a mapping $\underline{\mu}$ has on a term t can be described as follows. Let η be a variable occurring in t. If a specific occurrence of η in t has reference depth $k > 0$ in t then that occurrence of η will be replaced by a variable η' which has reference depth $\mu(k)$ in $\underline{\mu}t$. Internal reference occurrences of η in t are not effected by $\underline{\mu}$.

For example, if we have the tree

$$\lambda - \lambda - \lambda - \underline{\mu}\delta \overset{\lambda - \xi(4)}{\diagup} - \sigma(2,1) - \xi(2)$$

and if $\mu(1) = 2$, $\mu(2) = 1$, $\mu(3) = 3$ and $\mu(n) = n$, for $n \geq 3$, then $\xi(2)$ is bound by the second λ (from the left), $\sigma(2,1)$ is bound by the third λ (from the left) and $\xi(4)$ is bound by the first λ (from the left). This means that

$$\underline{\mu}\delta \overset{\lambda - \xi(4)}{\diagup} - \sigma(2,1) - \xi(2)$$

is equal to

$$\delta \overset{\lambda - \xi(4)}{\diagup} - \sigma(1,1) - \xi(3) \ .$$

We now introduce four important classes of refmaps (Definition 9) and give a formal definition of applying a mapping $\underline{\mu}$, for each refmap μ, to a term t (Definition 10). An informal explanation of Definition 9 and 10 is given in Section 2.4.

Definition 2.9.

Let m be an element of $\mathbb{M}$ and n be an element of $\mathbb{N}$. We define the following mappings, all elements of $[\mathbb{N} \to \mathbb{N}]$.

(i) $\varphi_m(n) = n + m$;

(ii) $\vartheta_m(n) = \begin{cases} n+1 , & \text{if } n \leq m \\ 1 , & \text{if } n = m+1 \\ n , & \text{if } n > m+1 \end{cases}$;

(iii) if $\psi \in \mathrm{Perm}$ then

$$\psi\hat{}(n) = \begin{cases} \psi(n) , & \text{if } n \in \mathrm{dom}(\psi) \\ n , & \text{if } n \notin \mathrm{dom}(\psi) \end{cases} ;$$

(iv) if $\mu \in [\mathbb{N} \to \mathbb{N}]$ then

$$\mu^{<m>}(n) = \begin{cases} n , & \text{if } n \leq m \\ m + \mu(n-m) , & \text{if } n > m \end{cases} .$$

□

Lemma 2.1.

(1) $\varphi_k \circ \varphi_m = \varphi_{k+m}$;

(2) $\vartheta_m \circ \vartheta_k^{<m>} = \vartheta_{k+m}$;

(3) $(\mu^{<k>})^{<m>} = \mu^{<k+m>}$;

(4) $(\mu \circ \nu)^{<m>} = \mu^{<m>} \circ \nu^{<m>}$.

Proof. A straightforward check of Definition 2.9. □

Definition 2.10.

If μ is a refmap then $\underline{\mu}(t)$ is defined inductively for terms t by

(i) $\underline{\mu}(\xi(n)) = \xi(\mu(n))$;

(ii) $\underline{\mu}(\omega(\psi)) = \omega(\mu \circ \psi)$;

(iii) $\underline{\mu}(\lambda u) = \lambda \underline{\mu^{<1>}}(u)$;

(iv) $\underline{\mu}(\sigma(n,m)u) = \sigma(\mu(n),m) \underline{\mu^{<m>}}(u)$;

(v) $\underline{\mu}\, \delta uv = \delta\, \underline{\mu}(u)\, \underline{\mu}(v)$. □

Remark. $\underline{\mu}(t)$ is a $\lambda\sigma$-term (easily proved by induction on L(t)). Furthermore, from now on we shall write $\underline{\mu}t$ instead of $\underline{\mu}(t)$ in order to economize on the use of parentheses.

2.4. Informal discussion of Definitions 2.9 and 2.10

In this section we will give an informal description of the effect that the mappings $\underline{\varphi_m}$, $\underline{\vartheta_m}$, $\underline{\psi\hat{}}$ and $\underline{\mu^{<m>}}$ have on an arbitrary term t.

(i) $\underline{\varphi_m}\, t$: Let η be a variable occurring in t. The effect of $\underline{\varphi_m}$ on t will be that the reference number, say n, inside an external reference occurrence of η in t will be raised by m and thus change to $n + m$.

(ii) $\underline{\vartheta_m}\, t$: In order to understand the effect of $\underline{\vartheta_m}$ on a term t consider the following two specific examples of terms written in tree fashion

$$\delta - \lambda - \underbrace{\lambda - \ldots - \lambda}_{m\ \lambda's} - \overset{A}{\delta} - \overset{\xi(j)}{\xi(m+1)} \quad ; \tag{1}$$

$$\underbrace{\lambda - \ldots - \lambda}_{m\ \lambda's} - \delta - \lambda - \underline{\vartheta_m}\, \delta - \overset{\underline{\varphi_m} A}{\overset{\xi(j)}{\xi(m+1)}} \quad . \tag{1'}$$

What we shall try to show is that the terms (1) and (1') are, in a sense, equivalent. The claim is that all references to λ's in (1) and (1') are the same; i.e. each specific occurrence of a variable in (1) will refer to exactly the same λ after it has (possibly) been reallocated in (1'). The mapping $\underline{\varphi_m}$ placed in front of A in (1') has the effect that all external reference occurrences in A skip the block of m preceding λ's by raising their respective reference numbers by m and thus ensuring that they remain bound by their original λ's (i.e. by the same λ's as in (1)). Furthermore, the mapping $\underline{\vartheta_m}$, placed in front of $\delta\, \xi(j)\, \xi(m+1)$, ensures that both $\xi(j)$ and $\xi(m+1)$ remain bound by their original λ. By doing so we achieve that $\xi(m+1)$ is bound by its original λ ($\vartheta_m(m+1) = 1$) and by applying $\underline{\vartheta_m}$ to j we see that if $\xi(j)$ was originally bound by a λ occurring in the preceding block of m λ's in (1), then by raising j by 1 ($\vartheta_m(j) = j+1$, if $1 \leq j \leq m$) we achieve that $\xi(j)$ remains bound by that same λ in (1'). If $j > m+1$ then $\underline{\vartheta_m}$ has no effect on $\xi(j)$, so also in this case we see that $\xi(j)$ is

bound by the same λ in both (1) and (1'). The purpose of this example is to show that one can reallocate a δAλ-part in a term t to some other place in t and, by introducing suitable refmaps φ_m and ϑ_m, can still preserve references to original λ's (i.e. references to λ's prior to this reallocation). As pointed out earlier in the introduction (Section 1.1.5). such reallocations of δAλ-parts will occur often after performing certain β-reductions inside segments. By β-reducing certain redices inside a segment we sometimes get the situation that the "δAλ-part" of the redex reappears and is placed directly in front of the end-point, say ω(ψ), of the segment. In order to ensure that all of the original references to λ's (i.e. references to λ's as they appeared in the segment before the β-reduction was performed) remain intact we introduce suitable mappings φ_m and ϑ_m; the mapping φ_m is placed in front of the term A and ϑ_m is placed in front of ω(ψ), thus obtaining $\delta\,\varphi_m\,A\,\lambda\,\vartheta_m\,\omega(\psi)$ at the end of the reduced segment. This way variables which are bound by a λ hidden inside a σ-variable abbreviating the segment just discussed, remain bound by the same λ after performing a β-reduction inside that segment (see also clause (iii) below). A formal description of how mappings ϑ_m are introduced in terms is given in Definition 2.11 (substitution).

(iii) $\psi\hat{}\,t$: If ψ is a segmap (i.e., an element of Perm) then $\psi\hat{}$ extends the domain of ψ to $\mathbb{N}$ by defining $\psi\hat{}(n)$ as n for each $n \in \mathbb{N}\backslash dom(\psi)$. Since $\psi\hat{}$ is an element of $[\mathbb{N} \to \mathbb{N}]$ we have extended the segmap ψ to a refmap $\psi\hat{}$. Such refmaps $\psi\hat{}$ are called upon when segments are substituted for segment variables. As described earlier in the introduction, the segmaps ψ occurring at end-points of segments reallocate references to λ's lying on the main branch of such segments. When we substitute a segment for some segment variable η we will introduce a refmap $\psi\hat{}$ which will have the intended reallocational effect on those variables which refer to a λ hidden inside η. As an example consider the following term

$$\lambda - \lambda - \delta - \lambda - \sigma(1,2) \;\overset{\delta - \lambda - \delta - \lambda - \omega(\psi)}{\overset{A \quad B}{\longrightarrow}}\; \delta - \xi(1) \qquad \xi(5) \qquad (2)$$

where ψ is the mapping with domain $\{1,2\}$ and $\psi(1) = 2$ and $\psi(2) = 1$. In (2) we see that $\sigma(1,2)$ abbreviates the segment $\delta\, A\, \lambda\, \delta\, B\, \lambda\, \omega(\psi)$ and that $\xi(1)$ - apparently - is bound by the first λ (from the right) hidden inside $\sigma(1,2)$. Although the reference number 1 in $\xi(1)$ apparently indicates that $\xi(1)$ is bound by the first λ hidden inside $\sigma(1,2)$, the segmap ψ reallocates this reference to the *second* λ hidden inside $\sigma(1,2)$ since $\psi(1) = 2$. By performing a β-reduction in (2) we get

$$\lambda - \lambda - \delta - \lambda - \delta - \lambda \overset{A}{\diagup}\!\!\overset{B}{\diagup}\!\!\overset{\xi(4)}{\diagup} \underline{\psi}\hat{}\,\delta - \xi(1)\ . \qquad (2')$$

Substitution of the segment $\delta\, A\, \lambda\, \delta\, B\, \lambda\, \omega(\psi)$ for $\sigma(1,2)$ has lead not only to the introduction of a mapping $\underline{\psi}\hat{}$ in (2'), but also to the replacement of the variable $\xi(5)$ by $\xi(4)$. The reason for replacing $\xi(5)$ by $\xi(4)$ is that by β-reducing (2) the λ of the redex that has been β-reduced has disappeared, and since this λ lies on the root path of $\xi(5)$ in (2) we have lowered the reference number 5 with 1 in order to keep this variable bound by its original λ (i.e., the same λ it was bound by in (2)). The mapping $\underline{\psi}\hat{}$ introduced in (2') has the intended effect on the variables $\xi(4)$ and $\xi(1)$: $\psi\hat{}(4) = 4$ and $\psi\hat{}(1) = 2$, so $\xi(4)$ and $\xi(1)$ refer to the same λ's in (2') as $\xi(5)$ and $\xi(1)$ refer to in (2). A formal description of how refmaps $\psi\hat{}$ are introduced is given in Definition 2.11 (substitution).

(iv) $\underline{\mu}^{\langle m\rangle}\, t$: When we evaluate the effect that a refmap μ has on a term t then we shift the mapping $\underline{\mu}$ through the tree representation of t, as seen from Definition 2.10, until we reach an end-point, where the (possibly altered) refmap μ is either applied to a reference number occurring in a ξ-variable or is composed with a segmap ψ occurring in some $\omega(\psi)$. In this shifting process we may encounter a λ and have to evaluate an expression like $\underline{\mu}\,\lambda u$. We define $\underline{\mu}\,\lambda u$ as $\lambda\, \underline{\mu}^{\langle 1\rangle} u$, for in this way the specific reference depths of all variables occurring in $\underline{\mu}\,\lambda u$ and $\lambda\, \underline{\mu}^{\langle 1\rangle} u$ are exactly the same. If an occurrence of a variable η has reference depth 0 in λu then this occurrence also has reference depth 0 in $\underline{\mu}\,\lambda u$; i.e. the mapping $\underline{\mu}$ has no influence on this occurrence of η. This same occurrence of η

has reference depth 1 in u as it should also have in $\underline{\mu^{<1>}}u$, this is why $\mu^{<1>}(1)$ is defined as 1. If an occurrence of a variable η has reference depth $j > 0$ in λu then it has reference depth $\mu(j)$ in $\underline{\mu}\,\lambda u$. This occurrence of η has reference depth $j+1$ in u, so in order to ensure that this occurrence has the same reference depth in both $\underline{\mu}\,\lambda u$ and $\lambda\,\underline{\mu^{<1>}}u$ we define $\mu^{<1>}(j+1)$ as $\mu(j) + 1$ or, in general, $\mu^{<1>}(k)$ is defined as $\mu(k-1) + 1$, for all $k > 1$. When we proceed in evaluating $\underline{\mu^{<1>}}u$ other mappings $\underline{\mu^{<m>}}$ $(m > 1)$ will often arise. Say that u is of the form λv then according to the definition of $\underline{\mu^{<1>}}\lambda v$ we get $\lambda\,\underline{(\mu^{<1>})^{<1>}}v$. Lemma 2.1 says that $(\mu^{<1>})^{<1>} = \mu^{<1+1>} = \mu^{<2>}$; i.e. $\underline{\mu^{<1>}}\lambda v = \lambda\,\underline{\mu^{<2>}}v$. In general, if a mapping $\underline{\mu}$ is applied to a term $t = \lambda \ldots \lambda v$ beginning with a block of m λ's $(m \geq 0)$ then this results in $\underline{\mu}\,t = \lambda \ldots \lambda\,\underline{\mu^{<m>}}v$. This is also the case when $\underline{\mu}$ is applied to a term t of the form $\sigma(j,m)v$. The variable $\sigma(j,m)$ in $\sigma(j,m)u$ has reference depth j in t and the number m in $\sigma(j,m)$ indicates that there is a block of m λ's hidden inside $\sigma(j,m)$. Therefore, μ is applied to the reference number j and once past $\sigma(j,m)$, the mapping $\underline{\mu}$ is changed to $\underline{\mu^{<m>}}$; i.e. $\underline{\mu}t = \sigma(\mu(j),m)\,\underline{\mu^{<m>}}v$.

We shall now give an example of evaluating the application of a mapping $\underline{\mu}$ to a term t. Let μ be the refmap φ_6, which raises the reference numbers of all external references in t by 6, and let t be the term

$$\lambda\ -\ \lambda\ -\ \sigma(3,5)\ -\ \delta\ -\ \xi(4)\ \diagup\ \xi(9)\ .$$

Now application of $\underline{\varphi_6}$ to t results in

$$\underline{\varphi_6}\,\lambda\,\lambda\,\sigma(3,5)\ \delta\ \xi(9)\ \xi(4) =$$

$$= \lambda\,\underline{\varphi_6^{<1>}}\,\lambda\,\sigma(3,5)\ \delta\ \xi(9)\ \xi(4) =$$

$$= \lambda\,\lambda\,\underline{\varphi_6^{<1><1>}}\,\sigma(3,5)\ \delta\ \xi(9)\ \xi(4) =$$

$$= \lambda\,\lambda\,\underline{\varphi_6^{<2>}}\,\sigma(3,5)\ \delta\ \xi(9)\ \xi(4) =$$

$$= \lambda\,\lambda\,\sigma(\varphi_6^{<2>}(3),5)\ \underline{\varphi_6^{<2><5>}}\ \delta\ \xi(9)\ \xi(4) =$$

$$= \lambda\lambda\sigma(2 + \varphi_6(3-2),5)\ \underline{\varphi_6^{<7>}}\ \delta\,\xi(9)\ \xi(4) =$$

$$= \lambda\lambda\sigma(2+7,5)\ \delta\ \underline{\varphi_6^{<7>}}\ \xi(9)\ \underline{\varphi_6^{<7>}}\ \xi(4) =$$

$$= \lambda\lambda\sigma(9,5)\ \delta\ \xi(\varphi_6^{<7>}(9))\ \xi(\varphi_6^{<7>}(4)) =$$

$$= \lambda\lambda\sigma(9,5)\ \delta\ \xi(7 + \varphi_6(9-7))\ \xi(4) =$$

$$= \lambda\lambda\sigma(9,5)\ \delta\ \xi(15)\ \xi(4)\ .$$

This is indeed the expected result; the reference numbers in the external reference occurrences $\sigma(3,5)$ and $\xi(9)$ have both been raised by 6 and the internal reference occurrence $\xi(4)$ remains unaltered. As mentioned earlier, refmaps μ and mappings $\underline{\mu}$ have been introduced in order to describe the effect that β-reduction of a term can have on the variables occurring in t. In the next section we will give a formal definition of substitution and describe how β-reduction of a term invokes the introduction of refmaps.

2.5. Beta-reduction and substitution

If a term is of the form $\delta\,A\,\lambda\,B$ then we call it a *redex*. We can read such a redex as "the function λB applied to the argument A". Should we evaluate the application of the "function part" λB to the "argument part" A then we say that this redex is β-*reduced* (or *contracted*), thus resulting in the substitution of A for all occurrences of variables in B with reference depth 1 in B. This substitution is denoted by $\Sigma(A,B,1)$. In general, the meta-symbol Σ (denoting substitution) takes three arguments and is of the form $\Sigma(A,B,k)$. The expression $\Sigma(A,B,k)$ is to be read as the substitution of the term A for all occurrences of variables in B with reference depth k in B. In this section we will start by giving a formal definition of $\Sigma(A,B,k)$. This definition is then followed by a discussion of each of the clauses involved.

Definition 2.11 (substitution).
Let u be a term and k be an element of $\mathbb{N}$. $\Sigma(u,v,k)$ is defined inductively for terms v by

$$\text{(i)}\quad v = \xi(n) : \Sigma(u,\xi(n),k) = \begin{cases} \underline{\varphi_{k-1}}\, u & \text{if } n = k \text{ and } LS(u) \in \mathrm{rge}(\xi) \\ \xi(n) & \text{if } n < k \\ \xi(n-1) & \text{if } n > k \end{cases} ;$$

$$\text{(ii)}\quad v = \omega(\psi) : \Sigma(u,\omega(\psi),k) = \begin{cases} \delta\, \underline{\varphi_{k-1}}\, u\, \lambda\, \underline{\vartheta_{k-1}}\, \omega(\psi) & \text{if } k \in \mathrm{rge}(\psi) \\ \omega(\vartheta_{k-1} \circ \psi - 1) & \text{if } k \notin \mathrm{rge}(\psi) \end{cases} ;$$

$$\text{(iii)}\quad v = \lambda w : \Sigma(u,\lambda w,k) = \lambda\Sigma(u,w,k+1) ;$$

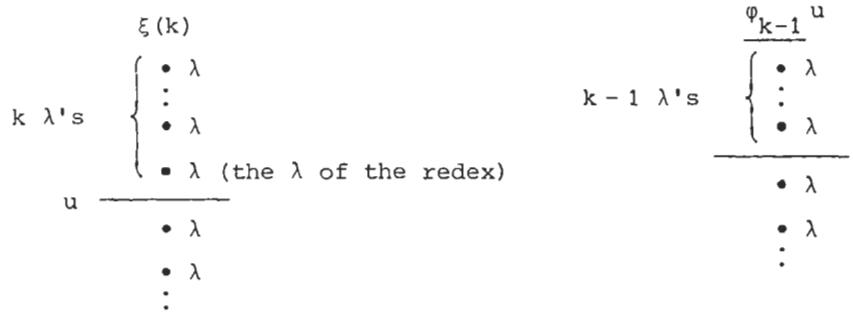
(iv) $v = \sigma(n,m)w$:

$$\Sigma(u,\sigma(n,m)w,k) = \begin{cases} s\, \underline{\psi}\hat{}\, \Sigma(u,w,k+m) & \text{if } n = k,\ W(u) = m^{*)}, \\ & \quad LS(u) \in \mathrm{rge}(\omega),\ \underline{\varphi_{k-1}}\, u = s\omega(\psi) \\ & \quad \text{and } \mathrm{rge}(\psi) \subseteq \mathbb{N}_m \\ \sigma(n,m)\, \Sigma(u,w,k+m) & \text{if } n < k \\ \sigma(n-1,m)\, \Sigma(u,w,k+m) & \text{if } n > k \end{cases} ;$$

$$\text{(v)}\quad v = \delta w_1 w_2 : \Sigma(u,\delta w_1 w_2,k) = \delta\Sigma(u,w_1,k)\, \Sigma(u,w_2,k) . \qquad \square$$

We now proceed with a discussion of each of the five clauses given in Definition 2.11.

(i) $v = \xi(n)$: If $n = k$ then we know that this occurrence of $\xi(k)$ has reference depth k in v and, thus, that u can be substituted for $\xi(k)$. We cannot, however, simply replace $\xi(k)$ by the term u, since external reference occurrences in u will then possibly get bound by the wrong λ's. This situation is clarified by the following diagrams



*) W(u), the *weight* of u, is formally defined in Definition 2.12.

The variable $\xi(k)$ is to be interpreted on an underlying "context" of k λ's, while the term u is to be interpreted on a context of λ's just below the context of $\xi(k)$. In order to ensure that the external reference occurrences in u remain bound by their original λ's after substitution of u for $\xi(k)$, the first k - 1 λ's have to be skipped when determining the λ's that bind these reference occurrences. The reason for this is that the end-stage of a substitution, resulting in an expression like $\Sigma(u,\xi(k),k)$, was the result of a β-reduction of some redex. This redex will be of the form $\delta\, u\, \lambda\, B$, where $\xi(k)$ occurs somewhere in the term B and $\xi(k)$ is bound by the λ of that redex. In the transition from $\delta\, u\, \lambda\, B$ to $\Sigma(u,B,1)$, the λ of the redex is dropped and all original references to that λ - like $\xi(k)$ - are replaced by u. After replacing $\xi(k)$ by u the term u is interpreted on a context of k - 1 extra λ's. By substituting $\varphi_{\underline{k-1}}\, u$ - instead of u - for $\xi(k)$ we ensure that the k - 1 extra λ's are skipped, with the effect that external reference occurrences in u remain bound by their original λ's. If $n > k$ then after reduction of the redex which give rise to $\Sigma(u,\xi(n),k)$ we have the situation that the λ of that redex has disappeared and since this λ occurred on the root path of $\xi(n)$ we lower the reference number n in $\xi(n)$ by 1 in order to maintain that this variable remains bound by its original λ. If $n < k$ then removal of this λ has no effect on $\xi(n)$ and we can let $\xi(n)$ remain unaltered. Note that we have only allowed substitution of the term u for $\xi(k)$ if the last symbol of u is a ξ-variable. This way we exclude substitution of segments for $\xi(k)$.

(ii) $v = \omega(\psi)$: As indicated earlier in the introduction (Section 1.1.5) we have to be careful when we evaluate expressions like $\Sigma(u,\omega(\psi),k)$. The reason for this is that $\Sigma(u,\omega(\psi),k)$ may have been the result of some internal β-reduction of a segment ending in $\omega(\psi)$. In that case we have to ensure that possible references to the λ of the redex which has been contracted remain intact. Should we, for example, simply define $\Sigma(u,\omega(\psi),k)$ as $\omega(\psi)$ then references to the λ of this redex are no longer possible, and this can lead to inconsistencies. If, for example, the segment in question is substituted for some segment variable

$\sigma(j,m)$ and there is a reference to the λ of this redex (one of the λ's of the block of m λ's hidden inside $\sigma(j,m)$) then after β-reduction this λ will have disappeared. As a consequence the argument u of this redex can no longer be substituted for those variables which originally, i.e. prior to β-reduction, referred to this now vanished λ. As an example consider the following term

$$\begin{array}{l} \qquad\qquad\quad / u \\ \quad / \lambda - \delta - \lambda - \lambda - \lambda - \omega(id_4) \\ \delta - \lambda - \sigma(1,4) - \xi(3) \end{array} \qquad . \qquad (3)$$

As we can see the variable $\sigma(1,4)$ abbreviates the segment $\lambda\,\delta\,u\,\lambda\,\lambda\,\lambda\,\omega(id_4)$ and $\xi(3)$ refers to the third λ (from the right) lying on the main branch of this segment; i.e. $\xi(3)$ refers to the λ of the redex $\delta\,u\,\lambda\,\lambda\,\lambda\,\omega(id_4)$ occurring in the segment $\lambda\,\delta\,u\,\lambda\,\lambda\,\lambda\,\omega(id_4)$. Should we β-reduce this redex and apply the rule that $\Sigma(u,\omega(id_4),3)$ results in $\omega(id_4)$, then the segment reduces to $\lambda\,\lambda\,\lambda\,\omega(id_4)$; i.e. the term (3) β-reduces to the term

$$\begin{array}{l} \quad / \lambda - \lambda - \lambda - \omega(id_4) \\ \delta - \lambda - \sigma(1,4) - \xi(3) \end{array} \qquad . \qquad (3')$$

In (3') two things are to be noticed: first, the variable $\sigma(1,4)$ now apparently abbreviates a segment with three λ's lying on the main branch of its tree, while four λ's are expected; second, the variable $\xi(3)$ now refers to a λ different from the λ it originally referred to in (3). From this we can conclude that not only is (3') ill-formed, it also contains references to λ's different from those in the original term. In order to remedy this situation we will have to come up with a different definition of $\Sigma(u,\omega(\psi),k)$. In the case of our example we shall define $\Sigma(u,\omega(id_4),3)$ as

$$\begin{array}{l} \quad / \underline{\varphi_2}\; u \\ \delta - \lambda - \underline{\vartheta_2}\; \omega(id_4) \end{array} \quad .$$

Following this definition, (3) β-reduces to

$$/\underline{\varphi_2}\ u$$
$$/\lambda - \lambda - \lambda - \delta - \lambda - \underline{\vartheta_2}\ \omega(id_4)$$
$$\delta - \lambda - \sigma(1,4) - \xi(3) \qquad . \qquad (3'')$$

The mapping $\underline{\varphi_2}$ ensures that all external reference occurrences in u remain bound by the same λ's as in (3), and the mapping $\underline{\vartheta_2}$ ensures that ξ(3) remains bound by its original λ as well ($\vartheta_2(3) = 1$), with the effect that u (or rather: $\underline{\varphi_2}\,u$) can still be substituted for ξ(3) (as originally intended in (3)) after performing suitable β-reductions (see also clause (iv)). In general, if $k \in rge(\psi)$ then $\Sigma(u,\omega(\psi),k)$ is defined as

$$/\underline{\varphi_{k-1}}\ u$$
$$\delta - \lambda - \underline{\vartheta_{k-1}}\ \omega(\psi)$$

with the effect that references to the λ of the redex that has been contracted inside some segment remain possible for all those variables indirectly bound by that λ from the outside — variables originally bound by this λ by means of an indirect reference to a corresponding λ hidden inside some segment variable σ(n,m) prior to reduction of the redex in question. If $k \notin rge(\psi)$ then the reference number k is not a customer for reallocation of references to λ's among the λ's lying on the main branch of the segment involved and in that case we could also define $\Sigma(u,\omega(\psi),k)$ as $\delta\,\underline{\varphi_{k-1}}\,u\,\lambda\,\underline{\vartheta_{k-1}}\,\omega(\psi)$. But if $k \notin rge(\psi)$ then $1 \notin rge(\vartheta_{k-1} \circ \psi)$. In other words $\vartheta_{k-1} \circ \psi$ will never reallocate a reference to the λ in $\delta\,\underline{\varphi_{k-1}}\,u\,\lambda\,\omega(\vartheta_{k-1} \circ \psi)$. In that case we can just as well discard the whole $\delta\,\varphi_{k-1}\,u\,\lambda$-part and simply write $\omega(\vartheta_{k-1} \circ \psi - 1)$ (we have subtracted 1 because the λ of the redex has disappeared).

(iii) $\Sigma(u,\lambda w,k)$: If an occurrence of a variable η has reference depth k in λw then η has reference depth k + 1 in w; therefore $\Sigma(u,\lambda w,k) = \lambda\Sigma(u,w,k+1)$.

(iv) $\Sigma(u,\sigma(n,m)w,k)$: If n = k then σ(n,m) has reference depth k in the term σ(n,m)w and u can be subsituted for σ(n,m). Certain conditions have to be met, though, if this substitution is to make sense. First of all, u has to be a segment; i.e. u is a

term ending in $\omega(\psi)$, for some $\psi \in \mathrm{Perm}$. Terms ending in a ξ-variable cannot be substituted for σ-variables; this would make no sense at all. Second, the weight of the term u (= the number of λ's lying on the main branch of u) has to be equal to the number m in $\sigma(n,m)$ (= the number of "hidden" λ's in $\sigma(n,m)$). When these two conditions have been fulfilled we can substitute u for $\sigma(n,m)$. Again, in order to maintain that external reference occurrences in u remain bound by their original λ's, we apply the mapping φ_{k-1} to u and replace $\sigma(n,m)$ by $\varphi_{k-1}u$; or more precisely: $\sigma(n,m)w$ is replaced by $s\,\underline{\psi}\hat{}\,\Sigma(u,w,k+m)$, where $\varphi_{k-1}u = s\omega(\psi)$. If an occurrence of a variable η in w has reference depth k in $\sigma(n,m)w$ then it has reference depth $k+m$ in w; this explains the part $\Sigma(u,w,k+m)$ in $s\,\underline{\psi}\hat{}\,\Sigma(u,w,k+m)$. The mapping $\underline{\psi}\hat{}$ is placed in front of $\Sigma(u,w,k+m)$ in order to ensure that the references of variables in $\Sigma(u,w,k+m)$ to a λ occurring in s get reallocated to their proper λ's as indicated by the segmap ψ. If $n \neq k$ then we have a situation analogous to the case $n \neq k$ in clause (i). The variable $\sigma(n,m)$ either remains $\sigma(n,m)$ or is changed to $\sigma(n-1,m)$, depending on whether $n < k$ or $n > k$.

(v) $\Sigma(u,\delta w_1 w_2,k)$: If an occurrence of a variable η has reference depth k in $\delta w_1 w_2$ then it either has reference depth k in w_1 or in w_2, depending on whether η occurs in w_1 or w_2. This obviously leads to the definition $\Sigma(u,\delta w_1 w_2,k) = \delta\Sigma(u,w_1,k)\,\Sigma(u,w_2,k)$.

We now proceed by stating some technical lemmas and theorems concerning refmaps and relations between refmaps and substitution.

Lemma 2.2.

If $k,\ell,m \in \mathbf{M}$ and μ is a refmap then

(i) $\mu^{<k>} \circ \varphi_k = \varphi_k \circ \mu$;

(ii) $\mu^{<k>} \circ \vartheta_{k-1} = \vartheta_{k-1} \circ \mu^{<k>}$;

(iii) $\varphi_m^{<k>} \circ \varphi_\ell = \begin{cases} \varphi_{m+\ell} & , \text{ if } k \leq \ell \\ \varphi_\ell \circ \varphi_m^{<k-\ell>} & , \text{ if } k > \ell \end{cases}$;

(iv) if $m < \ell$ then $\varphi_m^{<1>} \circ \vartheta_{\ell-m-1} = \vartheta_{\ell-1} \circ \varphi_m$;

(v) if $m \geq \ell$ then $\vartheta_m \circ \vartheta_{\ell-1} = \vartheta_{\ell-1}^{<1>} \circ \vartheta_m$.

Proof. Simple computation. □

Lemma 2.3.

For all refmaps μ and terms u, $L(\underline{\mu}u) = L(u)$.

Proof. By induction on the length of u and Lemma 2.1. □

Lemma 2.4.

For all refmaps μ, φ and term u, $(\underline{\mu \circ \varphi})u = \underline{\mu}(\underline{\varphi}u)$.

Proof. By induction on the length of u and Lemma 2.1. □

Definition 2.12.

W(u) is defined inductively for terms t by

(i) $W(\xi(n)) = 0$;

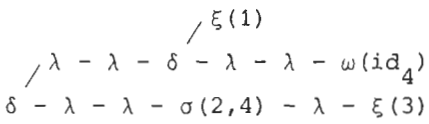
(ii) $W(\omega(\psi)) = 0$;

(iii) $W(\lambda u) = 1 + W(u)$;

(iv) $W(\sigma(n,m)u) = m + W(u)$;

(v) $W(\delta uv) = W(v)$. □

W(t) is called the *weight* of the term t. Informally, W(t) represents the number of all λ's (also those λ's hidden inside segment variables) lying on the main branch of the tree representation of t. For example, if t is the term $\delta\,\lambda\,\lambda\,\delta\,\xi(1)\,\lambda\,\lambda\,\omega(id_4)\,\lambda\,\lambda\,\sigma(2,4)\,\lambda\,\xi(3)$ then the tree equivalent of t is



and its weight is 1 + 1 + 4 + 1 = 7.

Lemma 2.5.

If $s\omega(\psi)$ is a segment and $W(s\omega(\psi)) = m$, then for all terms t and refmaps μ

$$\underline{\mu}(s \,\&\, t) = s' \,\&\, \underline{\mu}^{<m>} t \ ,$$

where $\underline{\mu}\, s\omega(\psi) = s'\omega(\psi')$.

Proof. By induction on L(s) and Lemma 2.1. □

Corollary.
If $LS(u) = \omega(\psi)$, $W(u) = m$ and $rge(\psi) \subseteq \mathbb{N}_m$ then $LS(\underline{\mu}u) = \omega(\psi)$.

Informally, the following theorem shows how to interpret substitutions for variables occurring at reference depth k (k > 1) as substitutions for variables occurring at reference depth 1.

Theorem 2.1.
Let u and v be terms and let k be an element of $\mathbb{N}$. If $(u,v,k) \in$ $\in dom(\Sigma)$ or $(\underline{\varphi_{k-1}}\, u, \underline{\vartheta_{k-1}}\, v, 1) \in dom(\Sigma)$, then

$$\Sigma(u,v,k) = \Sigma(\underline{\varphi_{k-1}}\, u, \underline{\vartheta_{k-1}}\, v, 1) .$$

Proof. By induction on L(v):

(i) $v = \xi(n)$:

$$A = \Sigma(u,\xi(n),k) = \begin{cases} \underline{\varphi_{k-1}}\, u & , \text{ if } n = k \\ \xi(n) & , \text{ if } n < k \\ \xi(n-1) & , \text{ if } n > k \end{cases} ;$$

$$B = \Sigma(\underline{\varphi_{k-1}}\, u, \underline{\vartheta_{k-1}}\, \xi(n), 1) =$$

$$= \Sigma(\underline{\varphi_{k-1}}\, u, \xi(\vartheta_{k-1}(n)), 1) = \begin{cases} \underline{\varphi_{k-1}}\, u & , \text{ if } \vartheta_{k-1}(n) = 1 \\ \xi(\vartheta_{k-1}(n) - 1) & , \text{ if } \vartheta_{k-1}(n) > 1 \end{cases} =$$

$$= \begin{cases} \underline{\varphi_{k-1}}\, u & , \text{ if } n = k \\ \xi(n) & , \text{ if } n < k \\ \xi(n-1) & , \text{ if } n > k \end{cases} .$$

Conclusion: A = B.

(ii) $v = \omega(\psi)$:

$$A = \Sigma(u,\omega(\psi),k) = \begin{cases} \delta\, \underline{\varphi_{k-1}}\, u\, \lambda\, \omega(\vartheta_{k-1} \circ \psi) & , \text{ if } k \in rge(\psi) \\ \omega(\vartheta_k \circ \psi - 1) & , \text{ if } k \notin rge(\psi) \end{cases} ;$$

$$B = \Sigma(\underline{\varphi_{k-1}}\, u, \underline{\vartheta_{k-1}}\, \omega(\psi), 1) = \Sigma(\underline{\varphi_{k-1}}\, u, \omega(\vartheta_{k-1} \circ \psi), 1) =$$

$$= \begin{cases} \delta\, \underline{\varphi_{k-1}}\, u\, \lambda\, \omega(\vartheta_{k-1} \circ \psi) , & \text{if } 1 \in \mathrm{rge}(\vartheta_{k-1} \circ \psi) \\ \omega(\vartheta_{k-1} \circ \psi - 1) , & \text{if } 1 \notin \mathrm{rge}(\vartheta_{k-1} \circ \psi) \end{cases} .$$

Since 1 is an element of the range of $\vartheta_{k-1} \circ \psi$ if and only if k is an element of the range of ψ we can conclude that A = B .

(iii) $v = \lambda w$:

$A = \Sigma(u, \lambda w, k) = \lambda\Sigma(u, w, k+1)$ and from the induction hypothesis it follows that $A = \lambda\Sigma(\underline{\varphi_k}\, u, \underline{\vartheta_k}\, w, 1)$.

$$B = \Sigma(\underline{\varphi_{k-1}}\, u, \underline{\vartheta_{k-1}}\, \lambda w, 1) =$$

$$= \Sigma(\underline{\varphi_{k-1}}\, u, \lambda\, \underline{\vartheta_{k-1}}^{<1>}\, w, 1) =$$

$$= \lambda\Sigma(\underline{\varphi_{k-1}}\, u, \underline{\vartheta_{k-1}}^{<1>}\, w, 2) .$$

From the induction hypothesis and Lemmas 2.1 and 2.4 we conclude

$$B = \lambda\Sigma(\underline{\varphi_1}\, (\underline{\varphi_{k-1}}\, u), \underline{\vartheta_1}\, (\underline{\vartheta_{k-1}}^{<1>}\, w), 1) =$$

$$= \lambda\Sigma(\underline{\varphi_1 \circ \varphi_{k-1}}\, u, \underline{\vartheta_1 \circ \vartheta_{k-1}}^{<1>}\, w, 1) =$$

$$= \lambda\Sigma(\underline{\varphi_k}\, u, \underline{\vartheta_k}\, w, 1)$$

and thus A = B .

(iv) $v = \sigma(n,m)w$: Let $A = \Sigma(u, \sigma(n,m)w, k)$ and $B = \Sigma(\underline{\varphi_{k-1}}\, u, \underline{\vartheta_{k-1}}\, \sigma(n,m)w, 1)$. By the induction hypothesis one of the following three cases holds for A or B

A: $n = k$ and $A = s\, \underline{\psi}\hat{}\, \Sigma(u, w, k+m) = s\, \underline{\psi}\hat{}\, \Sigma(\underline{\varphi_{k+m-1}}\, u, \underline{\vartheta_{k+m-1}}\, w, 1)$

$n < k$ and $A = \sigma(n,m)\, \Sigma(u, w, k+m) = \sigma(n,m)\, \Sigma(\underline{\varphi_{k+m-1}}\, u, \underline{\vartheta_{k+m-1}}\, w, 1)$

$n > k$ and $A = \sigma(n-1, m)\, \Sigma(u, w, k+m) =$

$$= \sigma(n-1, m)\, \Sigma(\underline{\varphi_{k+m-1}}\, u, \underline{\vartheta_{k+m-1}}\, w, 1) .$$

B: $B = \Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}_{k-1}\, \sigma(n,m)w, 1) = \Sigma(\underline{\varphi}_{k-1}\, u, \sigma(\vartheta_{k-1}(n), m)\, \underline{\vartheta}^{<m>}_{k-1}\, w, 1)$

(1) $n = k$ and $B = s\,\underline{\psi}\hat{}\,\Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}^{<m>}_{k-1}\, w, m+1) =$

$= s\,\underline{\psi}\hat{}\,\Sigma(\underline{\varphi}_{m}(\underline{\varphi}_{k-1}\, u), \underline{\vartheta}_{m}(\underline{\vartheta}^{<m>}_{k-1}\, w), 1) =$

$= s\,\psi\hat{}\,\Sigma(\underline{\varphi}_{m} \circ \underline{\varphi}_{k-1}\, u, \underline{\vartheta}_{m} \circ \underline{\vartheta}^{<m>}_{k-1}\, w, 1) =$ (Lemma 2.4)

$= s\,\psi\hat{}\,\Sigma(\underline{\varphi}_{m+k-1}\, u, \underline{\vartheta}_{m+k-1}\, w, 1)$ (Lemma 2.1)

(2) $n < k$ and $B = \Sigma(\underline{\varphi}_{k-1}\, u, \sigma(n+1,m)\, \underline{\vartheta}^{<m>}_{k-1}\, w, 1) =$

$= \sigma(n,m)\,\Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}^{<m>}_{k-1}\, w, m+1) =$

$= \sigma(n,m)\,\Sigma(\underline{\varphi}_{m+k-1}\, u, \underline{\vartheta}_{m+k-1}\, w, 1)$

(3) $n > k$ and $B = \Sigma(\underline{\varphi}_{k-1}\, u, \sigma(n,m)\, \underline{\vartheta}^{<m>}_{k-1}\, w, 1) =$

$= \sigma(n-1,m)\,\Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}^{<m>}_{k-1}\, w, m+1) =$

$= \sigma(n-1,m)\,\Sigma(\underline{\varphi}_{m+k-1}\, u, \underline{\vartheta}_{m+k-1}\, w, 1)$.

Conclusion: $A = B$.

(v) $v = \delta w_1 w_2$:

$A = \Sigma(u, \delta w_1 w_2, k) = \delta\Sigma(u, w_1, k)\,\Sigma(u, w_2, k)$ and from the induction hypothesis it follows that

$$A = \delta\Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}_{k-1}\, w_1, 1)\,\Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}_{k-1}\, w_2, 1) .$$

Furthermore, $B = \Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}_{k-1}\, \delta w_1 w_2, 1) =$

$= \Sigma(\underline{\varphi}_{k-1}\, u, \delta\, \underline{\vartheta}_{k-1}\, w_1\, \underline{\vartheta}_{k-1}\, w_2, 1) =$

$= \delta\Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}_{k-1}\, w_1, 1)\,\Sigma(\underline{\varphi}_{k-1}\, u, \underline{\vartheta}_{k-1}\, w_2, 1)$

and again we see that A = B, which completes the proof. □

Informally, the following theorem shows how refmaps interact with substitutions for variables occurring at reference depth 1 (Theorem 2.1 sees to it that all substitutions can be brought back to substitutions for variables occurring at this reference depth).

Theorem 2.2.
Let u and v be terms and μ be a refmap.
If $(u,v,1) \in \mathrm{dom}(\Sigma)$ or $(\underline{\mu}u, \underline{\mu^{<1>}}v,1) \in \mathrm{dom}(\Sigma)$, then

$$\underline{\mu}\,\Sigma(u,v,1) = \Sigma(\underline{\mu}u, \underline{\mu^{<1>}}v,1) \ .$$

Proof. By induction on L(v):

(i) $v = \xi(n)$:

(1) $n = 1$: $\underline{\mu}\,\Sigma(u,v,1) = \underline{\mu}u$, and furthermore $\mu^{<1>}(1) = 1$ from which it follows that $\Sigma(\underline{\mu}u, \underline{\mu^{<1>}}v,1) = \underline{\mu}u$.

(2) $n > 1$: $\underline{\mu}\,\Sigma(u,v,1) = \underline{\mu}\,\xi(n-1) = \xi(\mu(n-1))$, and furthermore $\mu^{<1>}(n) = 1 + \mu(n-1) > 1$; hence $\Sigma(\underline{\mu}u, \underline{\mu^{<1>}}v,1) =$
$= \Sigma(\underline{\mu}u, \xi(1 + \mu(n-1)),1) = \xi(\mu(n-1))$.

(ii) $v = \omega(\psi)$:

(1) $1 \in \mathrm{rge}(\psi)$: $\underline{\mu}\,\Sigma(u,v,1) = \underline{\mu}\,\delta\,\underline{\varphi_0}\,u\,\lambda\,\underline{\vartheta_0}\,\omega(\psi) = \underline{\mu}\,\delta\,u\,\lambda\,\omega(\psi) =$
$= \delta\,\underline{\mu}\,u\,\lambda\,\underline{\mu^{<1>}}\,\omega(\psi) = \delta\,\underline{\mu}\,u\,\lambda\,\omega(\mu^{<1>} \circ \psi)$, and furthermore $1 \in \mathrm{rge}(\mu^{<1>} \circ \psi)$; hence $\Sigma(\underline{\mu}u, \underline{\mu^{<1>}}v,1) = \Sigma(\underline{\mu}u, \omega(\mu^{<1>} \circ \psi),1) =$
$= \delta\,\underline{\mu}\,u\,\lambda\,\omega(\mu^{<1>} \circ \psi)$.

(2) $1 \notin \mathrm{rge}(\psi)$: $\underline{\mu}\,\Sigma(u,v,1) = \underline{\mu}\,\omega(\psi - 1) = \omega(\mu \circ (\psi - 1))$, and furthermore $1 \notin \mathrm{rge}(\mu^{<1>} \circ \psi)$; hence $\Sigma(\underline{\mu}u, \underline{\mu^{<1>}}v,1) =$
$= \Sigma(\underline{\mu}u, \omega(\mu^{<1>} \circ \psi),1) = \Sigma(\underline{\mu}u, \omega(1 + \mu \circ (\psi - 1)),1) =$
$= \omega((\vartheta_0 \circ (1 + \mu \circ (\psi - 1)) - 1) = \omega((1 + \mu \circ (\psi - 1)) - 1) =$
$= \omega(\mu \circ (\psi - 1))$.

(iii) $v = \lambda w$:

$$\underline{\mu}\, \Sigma(u,v,1) =$$

$$= \underline{\mu}\, \lambda\Sigma(u,w,2) =$$

$$= \lambda\, \underline{\mu^{<1>}}\, \Sigma(u,w,2) =$$

$$= \lambda\, \underline{\mu^{<1>}}\, \Sigma(\underline{\varphi_1}\, u, \underline{\vartheta_1}\, w, 1) = \qquad \text{(Theorem 2.1)}$$

$$= \lambda\, \Sigma(\underline{\mu^{<1>}}\, (\underline{\varphi_1}\, u), (\underline{\mu^{<1>}})^{<1>}\, (\underline{\vartheta_1}\, w), 1) = \qquad \text{(induction hypothesis)}$$

$$= \lambda\, \Sigma(\underline{\mu^{<1>} \circ \varphi_1}\, u, \underline{\mu^{<2>} \circ \vartheta_1}\, w, 1) = \qquad \text{(Lemmas 2.1 and 2.4)}$$

$$= \lambda\, \Sigma(\underline{\varphi_1 \circ \mu}\, u, \underline{\vartheta_1 \circ \mu^{<2>}}\, w, 1) = \qquad \text{(Lemma 2.2)}$$

$$= \lambda\, \Sigma(\underline{\varphi_1}\, (\underline{\mu} u), \underline{\vartheta_1}\, (\underline{\mu^{<2>}}\, w), 1) = \qquad \text{(Lemma 2.4)}$$

$$= \lambda\, \Sigma(\underline{\mu} u, \underline{\mu^{<2>}}\, w, 2) = \qquad \text{(Theorem 2.1)}$$

$$= \Sigma(\underline{\mu} u, \lambda\, \underline{\mu^{<2>}}\, w, 1) =$$

$$= \Sigma(\underline{\mu} u, \underline{\mu^{<1>}}\, \lambda\, w, 1) \ .$$

(iv) $v = \sigma(n,m)w$:

(1) $n = 1$:

$\underline{\mu}\, \Sigma(u,\sigma(1,m)w,1) = \underline{\mu}\, s\, \underline{\psi\hat{}}\, \Sigma(u,w,m+1)$, where $u = s\omega(\psi)$, $W(u) = m$ and $\mathrm{rge}(\psi) \subseteq \mathbb{N}_m$.

Furthermore

$$\underline{\mu}\, s\, \underline{\psi\hat{}}\, \Sigma(u,w,m+1) =$$

$$= s'\, \underline{\mu^{<m>}}\, (\underline{\psi\hat{}}\, \Sigma(u,w,m+1)) = \qquad \text{(Lemma 2.5)}$$

$$= s'\, \underline{\mu^{<m>} \circ \psi\hat{}}\, \Sigma(u,w,m+1) = \qquad \text{(Lemma 2.4)}$$

$$= s'\, \underline{\psi\hat{} \circ \mu^{<m>}}\, \Sigma(u,w,m+1) = \qquad (\mathrm{rge}(\psi) \subseteq \mathbb{N}_m)$$

$= s' \underline{\psi}\hat{} (\underline{\mu}^{<m>} \Sigma(u,w,m+1) =$ (Lemma 2.4)

$= s' \underline{\psi}\hat{} (\underline{\mu}^{<m>} \Sigma(\underline{\varphi}_m u, \underline{\vartheta}_m w,1) =$ (Theorem 2.1)

$= s' \underline{\psi}\hat{} \Sigma(\underline{\mu}^{<m>} (\underline{\varphi}_m u), \underline{\mu}^{<m+1>} (\underline{\vartheta}_m w),1) =$ (induction hypothesis)

$= s' \underline{\psi}\hat{} \Sigma(\underline{\varphi}_m (\underline{\mu} u), \underline{\vartheta}_m (\underline{\mu}^{<m+1>} w),1) =$ (Lemmas 2.2 and 2.4)

$= s' \underline{\psi}\hat{} \Sigma(\underline{\mu} u, \underline{\mu}^{<m+1>} w, m+1) =$ (Theorem 2.1)

$= \Sigma(\underline{\mu} u, \sigma(1,m)\, \underline{\mu}^{<m+1>} w,1) =$

$= \Sigma(\underline{\mu} u, \underline{\mu}^{<1>} \sigma(1,m) w,1)$.

(2) $n > 1$:

$\underline{\mu}\, \Sigma(u, \sigma(n,m) w,1) =$

$= \underline{\mu}\, \sigma(n-1,m)\, \Sigma(u,w,m+1) =$

$= \sigma(\mu(n-1),m)\, \underline{\mu}^{<m>} \Sigma(u,w,m+1) =$

$= \sigma(\mu(n-1),m)\, \underline{\mu}^{<m>} \Sigma(\underline{\varphi}_m u, \underline{\vartheta}_m w,1) =$ (Theorem 2.1)

$= \sigma(\mu(n-1),m)\, \Sigma(\underline{\mu}^{<m>} (\underline{\varphi}_m u), \underline{\mu}^{<m+1>} (\underline{\vartheta}_m w),1) =$ (induction hypothesis)

$= \sigma(\mu(n-1),m)\, \Sigma(\underline{\varphi}_m (\underline{\mu} u), \underline{\vartheta}_m (\underline{\mu}^{<m+1>} w),1) =$ (Lemmas 2.2 and 2.4)

$= \sigma(\mu(n-1),m)\, \Sigma(\underline{\mu} u, \underline{\mu}^{<m+1>} w, m+1)$. (Theorem 2.1)

If $n > 1$ then $\mu^{<1>}(n) = 1 + \mu(n-1)$, from which it follows that

$\Sigma(\underline{\mu} u, \underline{\mu}^{<1>} \sigma(n,m) w,1) =$

$= \Sigma(\underline{\mu} u, \sigma(1 + \mu(n-1),m)\, \underline{\mu}^{<m+1>} w,1) =$

$= \sigma(\mu(n-1),m)\, \Sigma(\underline{\mu} u, \underline{\mu}^{<m+1>} w, m+1)$.

(v) $v = \delta w_1 w_2$:

$$\underline{\mu}\,\Sigma(u,\delta w_1 w_2,1) =$$

$$= \underline{\mu}\,\delta\Sigma(u,w_1,1)\;\Sigma(u,w_2,1) =$$

$$= \delta\,\underline{\mu}\,\Sigma(u,w_1,1)\;\underline{\mu}\,\Sigma(u,w_2,1) =$$

$$= \delta\Sigma(\underline{\mu}u,\underline{\mu^{<1>}}\,w_1,1)\;\Sigma(\underline{\mu}u,\underline{\mu^{<1>}}\,w_2,1) = \qquad \text{(induction hypothesis)}$$

$$= \Sigma(\underline{\mu}u,\delta\,\underline{\mu^{<1>}}\,w_1\,\underline{\mu^{<1>}}\,w_2,1) =$$

$$= \Sigma(\underline{\mu}u,\underline{\mu^{<1>}}\,\delta w_1 w_2,1)\ . \qquad \square$$

Corollary.
If $m < \ell$ then $\underline{\varphi_m}\,\Sigma(u,v,\ell-m) = \Sigma(u,\underline{\varphi_m}\,v,\ell)$.

Proof.

$$\underline{\varphi_m}\,\Sigma(u,v,\ell-m) =$$

$$= \underline{\varphi_m}\,\Sigma(\underline{\varphi_{\ell-m-1}}\,u,\underline{\vartheta_{\ell-m-1}}\,v,1) = \qquad \text{(Theorem 2.1)}$$

$$= \Sigma(\underline{\varphi_m}\,(\underline{\varphi_{\ell-m-1}}\,u),\underline{\varphi_m^{<1>}}\,(\underline{\vartheta_{\ell-m-1}}\,v),1) = \qquad \text{(Theorem 2.2)}$$

$$= \Sigma(\underline{\varphi_{\ell-1}}\,u,\underline{\vartheta_{\ell-1}}\,(\underline{\varphi_m}\,v),1) = \qquad \text{(Lemmas 2.2 and 2.4)}$$

$$= \Sigma(u,\underline{\varphi_m}\,v,\ell)\ . \qquad \square$$

Informally, the following theorem shows that in some cases it is possible to short-cut the evaluation of a substitution.

Theorem 2.3.
Let k be an element of $\mathbb{N}$ and m an element of $\mathbb{M}$. If $k+m \geq \ell > m$ and $(u,\underline{\varphi_k^{<m>}}\,v,\ell) \in \mathrm{dom}(\Sigma)$ then $\Sigma(u,\underline{\varphi_k^{<m>}}\,v,\ell) = \underline{\varphi_{k-1}^{<m>}}\,v$.

Proof. By induction on L(v):

(i) $v = \xi(n)$: $\varphi_k^{<m>}(n) = \begin{cases} n\ , & \text{if } 1 \leq n \leq m \\ n+k\ , & \text{if } n > m \end{cases}$.

If $k+m \geq \ell > m$ then $\varphi_k^{<m>}(n) \neq \ell$.

$$\Sigma(u, \underline{\varphi_k^{<m>}}\, \xi(n), \ell) = \begin{cases} \xi(n) & , \text{ if } 1 \leq n \leq m \\ \xi(n+k-1) & , \text{ if } n > m \end{cases} =$$

$$= \underline{\varphi_{k-1}^{<m>}}\, \xi(n) .$$

(ii) $v = \omega(\psi)$: $\varphi_k^{<m>} \circ \psi(n) = \begin{cases} \psi(n) & , \text{ if } 1 \leq \psi(n) \leq m \\ \psi(n) + k & , \text{ if } \psi(n) > m \end{cases}$.

If $k+m \geq \ell > m$ then $\ell \notin \mathrm{rge}(\varphi_k^{<m>} \circ \psi)$.

$$\Sigma(u, \underline{\varphi_k^{<m>}}\, \omega(\psi), \ell) = \omega(\vartheta_{\ell-1} \circ \varphi_k^{<m>} \circ \psi - 1) .$$

Furthermore $\vartheta_{\ell-1} \circ \varphi_k^{<m>} \circ \psi - 1(n) = \begin{cases} \psi(n) & , \text{ if } 1 \leq \psi(n) \leq m \\ \psi(n) + k - 1 & , \text{ if } \psi(n) > m \end{cases} =$

$$= \underline{\varphi_{k-1}^{<m>}} \circ \psi(n) .$$

Conclusion: $\Sigma(u, \underline{\varphi_k^{<m>}}\, \omega(\psi), \ell) = \underline{\varphi_{k-1}^{<m>}}\, \omega(\psi)$.

(iii) $v = \lambda w$:

$$\Sigma(u, \underline{\varphi_k^{<m>}}\, w, \ell) =$$

$$= \Sigma(u, \lambda\, \underline{\varphi_k^{<m+1>}}\, v, \ell) =$$

$$= \lambda \Sigma(u, \underline{\varphi_k^{<m+1>}}\, v, \ell + 1) =$$

$$= \lambda\, \underline{\varphi_{k-1}^{<m+1>}}\, v = \qquad \text{(induction hypothesis)}$$

$$= \underline{\varphi_{k-1}^{<m>}}\, \lambda v .$$

(iv) $v = \sigma(n,p)w$:

$$S = \Sigma(u, \underline{\varphi_k^{<m>}}\, \sigma(n,p)w, \ell) = \Sigma(u, \sigma(\varphi_k^{<m>}(n), p)\, \varphi_k^{<m+p>}\, w, \ell) .$$

As seen earlier in i) $\ell \notin \mathrm{rge}(\varphi_k^{<m>})$, therefore $S =$

a) $1 \leq n \leq m$: $\sigma(n,p)\, \Sigma(u, \underline{\varphi_k^{<m+p>}}\, w, \ell + p) =$

$$= \sigma(n,p)\ \underline{\varphi_{k-1}^{<m+p>}}\ w = \qquad \text{(induction hypothesis)}$$

$$= \varphi_{k-1}^{<m>}\ \sigma(n,p)w\ ;$$

b) $n > m$: $\sigma(n+k-1,p)\ \Sigma(u,\underline{\varphi_k^{<m+p>}}\ w,\ell+p) =$

$$= \sigma(n+k-1,p)\ \underline{\varphi_{k-1}^{<m+p>}}\ w = \qquad \text{(induction hypothesis)}$$

$$= \underline{\varphi_{k-1}^{<m>}}\ \sigma(n,p)w\ .$$

(v) $v = \delta tw$:

$$\Sigma(u,\underline{\varphi_k^{<m>}}\ v,\ell) =$$

$$= \delta\Sigma(u,\underline{\varphi_k^{<m>}}\ t,\ell)\ \Sigma(u,\underline{\varphi_k^{<m>}}\ w,\ell) =$$

$$= \delta\ \underline{\varphi_{k-1}^{<m>}}\ t\ \underline{\varphi_{k-1}^{<m>}}\ w \qquad \text{(induction hypothesis)}$$

$$= \underline{\varphi_{k-1}^{<m>}}\ \delta tw\ .$$ □

Corollary.

$$\Sigma(u,\underline{\varphi_1}\ v,1) = v\ .$$

2.6. The permutation condition (PC)

With the definition of λσ-terms as it stands it is possible to construct terms that have undesirable properties. Consider as an example the following term represented in tree form by

$$\begin{array}{l} \qquad\qquad\qquad\qquad\quad /\,A \\ \qquad\qquad\qquad\quad /\,\lambda - \lambda - \delta - \lambda - \omega(\psi) \\ \lambda - \lambda - \delta - \lambda - \sigma(1,3) - B \end{array} \quad ,$$

where ψ is a segmap containing a number larger than 3 in its range. Such a segmap ψ also reallocates references to λ's that do not lie on the main branch of its segment. Should we substitute the segment

$$\lambda - \lambda - \overset{A}{\delta} - \lambda - \omega(\psi)$$

for $\sigma(1,3)$ then this results in

$$\lambda - \lambda - \lambda - \lambda - \overset{A}{\delta} - \lambda - \underline{\psi}\hat{} - B$$

and we see that variables in B which are bound by a λ in front of the segment are now influenced by ψ. This is not the intention, though. Such external reference occurrences in B should not be influenced by ψ; the sole role of a segmap is the reallocation of references to λ's which occur inside its corresponding segment (variables in B which are bound by λ's in front of the segment can refer directly to these λ's anyway, instead of indirectly by means of a segmap ψ). We therefore require that a segment t with $LS(t) = \omega(\psi)$ satisfies the condition $\psi \in Perm(m)$, where $m = W(t)$. That way the segmap ψ will only influence references to λ's occurring in the segment t. This condition is taken care of by the so-called *permutation condition* (PC) described below.

Definition 2.13 (PC).
Let k be an element of $\mathbb{M}$.
PC(t,k) is defined inductively for terms t by

(i) $PC(\xi(n),k)$,

(ii) $PC(\omega(\psi),k) \Leftrightarrow \psi \in Perm(k)$;

(iii) $PC(\lambda u,k) \Leftrightarrow PC(u,k+1)$;

(iv) $PC(\sigma(n,m)u,k) \Leftrightarrow PC(u,k+m)$;

(v) $PC(\delta uv,k) \Leftrightarrow PC(u,0) \wedge PC(v,k)$. □

Informally, the number k indicates the number of λ's encountered in the process of "recursively shifting" PC through the tree of t. Given a term t we will require PC(t,0), and if PC(t,0) holds we say that t satisfies the permutation condition.

We now give an example of a term which satisfies the permutation condition. Consider the following term t

$$\lambda\,\delta\,\lambda\,\delta\,\lambda\,\xi(2)\;\lambda\,\lambda\,\omega(\psi)\;\lambda\,\lambda\,\sigma(2,3)\;\lambda\,\omega(\varphi) \qquad (4)$$

where $\psi \in \mathrm{Perm}(3)$ and $\varphi \in \mathrm{Perm}(7)$. The tree representation of (4) is

$$\begin{array}{l} \qquad\qquad\qquad\quad /\,\lambda - \xi(2) \\ \qquad\qquad\quad /\,\lambda - \delta - \lambda - \lambda - \omega(\psi) \\ \lambda - \delta - \lambda - \lambda - \sigma(2,3) - \lambda - \omega(\varphi)\ . \end{array} \qquad (4')$$

From (4') we see that PC(t,0) holds if PC($\xi(2)$,1), PC($\omega(\psi)$,3) and PC($\omega(\varphi)$,7) hold. The first condition is trivial, and since $\psi \in \mathrm{Perm}(3)$ and $\varphi \in \mathrm{Perm}(7)$ we also have PC($\omega(\psi)$,3) and PC($\omega(\varphi)$,7). We now proceed by stating some properties concerning the permutation condition which will be used later on.

The following four lemmas are easily proved by induction on L(t).

Lemma 2.6.
Let m be an element of $\mathbb{M}$.
If $t = s\omega(id_m)$ and u are terms and W(t) = m then

$$PC(s \,\&\, u,k) \Leftrightarrow PC(t,0) \wedge PC(u,k+m)\ .$$

Lemma 2.7.
Let t be a term and k,ℓ be elements of $\mathbb{M}$.
If $\ell \geq k$ then, for all refmaps μ

$$PC(t,k) \Leftrightarrow PC(\underline{\mu}^{<\ell>}t,k)\ .$$

Corollary.

$$PC(t,0) \Leftrightarrow PC(\underline{\mu}t,0)\ .$$

Lemma 2.8.
Let t be a term and k be an element of $\mathbb{M}$.
If $\psi \in \mathrm{Perm}(m)$ and $m \leq k$ then

$$PC(\underline{\psi}\hat{}\,t,k) \Leftrightarrow PC(t,k)\ .$$

Lemma 2.9.
Let t be a term and k,m be elements of $\mathbb{M}$.

If $LS(t) \in rge(\xi)$ then

$$PC(t,k) \Leftrightarrow PC(t,m) \ .$$

Theorem 2.4.

Let t and u be terms and k,ℓ be elements of $\mathbb{M}$.

If $PC(t,k)$, $PC(u,0)$, $\ell > k$ and $(u,t,\ell) \in dom(\Sigma)$ then $PC(\Sigma(u,t,\ell),k)$.

Proof. By induction on $L(t)$:

(i) $t = \xi(n)$: $\Sigma(u,t,\ell) = \begin{cases} \underline{\varphi_{\ell-1}}\, u & , \text{ if } n = \ell \\ \xi(n) & , \text{ if } n < \ell \\ \xi(n-1) & , \text{ if } n > \ell \end{cases}$.

If $n \neq \ell$ then, by Lemma 2.9, it follows that $PC(\Sigma(u,\xi(n),\ell),k)$. If $n = \ell$ then $\Sigma(u,\xi(n),\ell) = \underline{\varphi_{\ell-1}}\, u$ and $LS(u) \in rge(\xi)$. From the corollary to Lemma 2.7 and $PC(u,0)$ it follows that $PC(\underline{\varphi_{\ell-1}}\, u,0)$. Furthermore, if $LS(u) \in rge(\xi)$ then $LS(\underline{\varphi_{\ell-1}}\, u) \in rge(\xi)$. From Lemma 2.9 and $PC(\underline{\varphi_{\ell-1}}\, u,0)$ it follows that $PC(\underline{\varphi_{\ell-1}}\, u,k)$.

(ii) $t = \omega(\psi)$: From $PC(\omega(\psi),k)$ it follows that $\psi \in Perm(k)$, and therefore that $\ell \notin rge(\psi)$. If $\ell \notin rge(\psi)$ then $\Sigma(u,\omega(\psi),\ell) = \omega(\vartheta_{\ell-1} \circ \psi - 1)$. Furthermore, if $\ell > k$, $\vartheta_{\ell-1} \circ \psi - 1 = \psi$. Therefore $\Sigma(u,\omega(\psi),\ell) = \omega(\psi)$ and, thus, $PC(\Sigma(u,\omega(\psi),\ell),k)$.

(iii) $t = \lambda v$: If $PC(t,k)$ then $PC(v,k+1)$. The induction hypothesis gives $PC(\Sigma(u,v,\ell+1),k+1)$ and therefore we have $PC(\Sigma(u,\lambda v,\ell),k)$.

(iv) $t = \sigma(n,m)v$:

$$\Sigma(u,\sigma(n,m)v,\ell) = \begin{cases} s\,\underline{\psi}\hat{}\ \Sigma(u,v,\ell+m) & , \text{ if } n = \ell; \text{ where} \\ & \underline{\varphi_{\ell-1}}\, u = s\omega(\psi) \text{ and } W(u) = m \\ \sigma(n,m)\ \Sigma(u,v,\ell+m) & , \text{ if } n < \ell \\ \sigma(n-1,m)\ \Sigma(u,v,\ell+m) & , \text{ if } n > \ell \end{cases} .$$

a) $n \neq \ell$: From $PC(\sigma(n,m)v,k)$ it follows that $PC(v,k+m)$. From the induction hypothesis it follows that $PC(\Sigma(u,v,\ell+m),k+m)$ and therefore both $PC(\sigma(n,m)\ \Sigma(u,v,\ell+m),k)$ and $PC(\sigma(n-1,m)\ \Sigma(u,v,\ell+m),k)$.

b) $n = \ell$: From $PC(u,0)$ and the corollary to Lemma 2.7 it follows that $PC(\varphi_{\underline{\ell-1}}\, u,0)$. From the corollary to Lemma 2.5 we see that if $\varphi_{\underline{\ell-1}}\, u = s\omega(\psi)$ then $LS(u) = \omega(\psi)$. Furthermore

$PC(u,0) \wedge PC(\sigma(\ell,m)v,k) \Rightarrow$

$\Rightarrow PC(\varphi_{\underline{\ell-1}}\, u,0) \wedge PC(v,k+m) \Rightarrow$

$\Rightarrow PC(s\omega(\psi),0) \wedge PC(\Sigma(u,v,\ell+m),k+m) \Rightarrow$ (induction hypothesis)

$\Rightarrow PC(s\omega(id_m),0) \wedge PC(\Sigma(u,v,\ell+m),k+m) \Rightarrow$

$\Rightarrow PC(s\omega(id_m),0) \wedge PC(\underline{\psi}\hat{}\,\Sigma(u,v,\ell+m),k+m) \Rightarrow$ (Lemma 2.8)

$\Rightarrow PC(s\,\underline{\psi}\hat{}\,\Sigma(u,v,\ell+m),k)$. (Lemma 2.6)

(v) $t = \delta vw$:

$PC(u,0) \wedge PC(t,k) \Rightarrow$

$\Rightarrow PC(u,0) \wedge PC(v,0) \wedge PC(w,k) \Rightarrow$

$\Rightarrow PC(\Sigma(u,v,\ell),0) \wedge PC(\Sigma(u,w,\ell),k) \Rightarrow$ (induction hypothesis)

$\Rightarrow PC(\delta\Sigma(u,v,\ell)\,\Sigma(u,w,\ell),k) \Rightarrow$

$\Rightarrow PC(\Sigma(u,\delta vw,\ell),k)$. □

Corollary.

$PC(u,0) \wedge PC(t,0) \Rightarrow PC(\Sigma(u,t,1),0)$.

The following theorem shows that the permutation condition is invariant with respect to β-reduction.

Theorem 2.5.
Let $\delta\, u\, \lambda\, t$ be a term and k be an element of $\mathbb{M}$.
If $(u,t,1) \in dom(\Sigma)$ then

$PC(\delta\, u\, \lambda\, t,k) \Rightarrow PC(\Sigma(u,t,1),k)$.

Proof. By induction on L(t).
From $PC(\delta\, u\, \lambda\, t,k)$ it follows that

a) $PC(u,0)$;

b) $PC(t,k+1)$.

(i) $t = \xi(n)$: $\Sigma(u,t,1) = \begin{cases} u & , \text{ if } n = 1 \\ \xi(n-1) & , \text{ if } n > 1 \end{cases}$.

If $n > 1$ then $PC(\Sigma(u,\xi(n),1),k)$ (trivial). If $n = 1$ then $\Sigma(u,t,1) = u$ and $LS(u) \in rge(\xi)$. From Lemma 2.9 and $PC(u,0)$ it follows that $PC(u,k)$.

(ii) $t = \omega(\psi)$: From $PC(t,k+1)$ it follows that $\psi \in Perm(k+1)$ and therefore $1 \in rge(\psi)$. If $1 \in rge(\psi)$ then $\Sigma(u,\omega(\psi),1) = \delta\, u\, \lambda\, \omega(\psi)$ and therefore $PC(\Sigma(u,t,1),k)$.

(iii) $t = \lambda v$: From the corollary to Lemma 2.7 and $PC(u,0)$ it follows that $PC(\underline{\varphi_1}\, u,0)$. Furthermore

$PC(\lambda v,k+1) \Leftrightarrow$

$\Leftrightarrow PC(v,k+2) \Leftrightarrow$

$\Leftrightarrow PC(\underline{\vartheta_1}\, v,k+2) \Leftrightarrow$ (Lemma 2.8)

$\Leftrightarrow PC(\lambda\, \underline{\vartheta_1}\, v,k+1)$.

From $PC(\underline{\varphi_1}\, u,0)$ and $PC(\lambda\, \underline{\vartheta_1}\, v,k+1)$ we have $PC(\delta\, \underline{\varphi_1}\, u\, \lambda\, \underline{\vartheta_1}\, v,k+1)$. From the induction hypothesis it follows that $PC(\Sigma(\underline{\varphi_1}\, u,\underline{\vartheta_1}\, v,1),k+1)$ and therefore $PC(\Sigma(u,v,2),k+1)$ (Theorem 2.1). If $PC(\Sigma(u,v,2),k+1)$ then $PC(\Sigma(u,\lambda v,1),k)$.

(iv) $t = \sigma(n,m)v$. If $n = 1$ then $\Sigma(u,\sigma(n,m)v,1) = s\, \underline{\psi}\hat{}\, \Sigma(u,v,m+1)$, where $u = s\omega(\psi)$ and $W(u) = m$. Furthermore

$PC(\delta\, u\, \lambda\, \sigma(1,m)v,k) \Leftrightarrow$

$\Leftrightarrow PC(u,0) \wedge PC(v,k+m+1) \Leftrightarrow$

$\Leftrightarrow PC(u,0) \wedge PC(\underline{\vartheta_m}\, v,k+m+1) \Leftrightarrow$ (Lemma 2.8)

$\Leftrightarrow PC(u,0) \wedge PC(\underline{\varphi_m}\, u,0) \wedge PC(\lambda\, \underline{\vartheta_m}\, v,k+m) \Leftrightarrow$ (cor. to Lemma 2.7)

$\Leftrightarrow PC(u,0) \wedge PC(\delta\, \underline{\varphi_m}\, u\, \lambda\, \underline{\vartheta_m}\, v,k+m) \Rightarrow$

$\Rightarrow$ PC(u,0) $\wedge$ PC($\Sigma(\underline{\varphi}_m u, \underline{\vartheta}_m v, 1), k+m$) $\Rightarrow$ (induction hypothesis)

$\Rightarrow$ PC(s$\omega(\psi)$,0) $\wedge$ PC($\Sigma(u,v,m+1), k+m$) $\Rightarrow$ (Theorem 2.1)

$\Rightarrow$ PC(s$\omega(\mathrm{id}_m)$,0) $\wedge$ PC($\underline{\psi}\hat{}\,\Sigma(u,v,m+1), k+m$) $\Rightarrow$ (Lemma 2.8)

$\Rightarrow$ PC(s $\underline{\psi}\hat{}\,\Sigma(u,v,m+1), k$) . (Lemma 2.6)

If $n > 1$ then $\Sigma(u,t,1) = \sigma(n-1,m)\,\Sigma(u,v,m+1) =$
$= \sigma(n-1,m)\,\Sigma(\underline{\varphi}_m u, \underline{\vartheta}_m v, 1)$ and therefore PC($\Sigma(u,t,1),k$) $\Leftrightarrow$
$\Leftrightarrow$ PC($\Sigma(\underline{\varphi}_m u, \underline{\vartheta}_m v, 1), m+1$). PC($\Sigma(\underline{\varphi}_m u, \underline{\vartheta}_m v, 1), m+1$) is proved as above for the case $n = 1$.

(v) $t = \delta vw$:

PC($\delta\, u\, \lambda\, t, k$) $\Leftrightarrow$

$\Leftrightarrow$ PC(u,0) $\wedge$ PC(t,k+1) $\Leftrightarrow$

$\Leftrightarrow$ PC(u,0) $\wedge$ PC(v,0) $\wedge$ PC(w,k+1) $\Rightarrow$

$\Rightarrow$ PC($\delta\, u\, \lambda\, w, k$) $\wedge$ PC($\Sigma(u,v,1),0$) $\Rightarrow$ (cor. to Theorem 2.4)

$\Rightarrow$ PC($\Sigma(u,w,1),k$) $\wedge$ PC($\Sigma(u,v,1),0$) $\Rightarrow$ (induction hypothesis)

$\Rightarrow$ PC($\delta\Sigma(u,v,1)\,\Sigma(u,w,1),k$) $\Rightarrow$

$\Rightarrow$ PC($\Sigma(u,\delta vw,1),k$) . □

Informally, the following lemma shows that with PC it is sometimes possible to short-cut the evaluation of a substitution.

Lemma 2.10.
Let t and u be terms and k,ℓ be elements of $\mathbb{M}$. If $\ell \leq k$, PC(u,ℓ) and

$\forall \eta \in \mathrm{Var}(u) : k+1 \notin D(\eta,u)$, then $\underline{\varphi}_1^{<k>}\,\Sigma(t,u,k+1) = u$.

Proof. By induction on L(u). □

Corollary.

PC(u,0) $\wedge$ $\forall \eta \in \mathrm{Var}(u) : 1 \notin D(\eta,u) \Rightarrow \underline{\varphi}_1\,\Sigma(t,u,1) = u$.

3. THE CHURCH-ROSSER THEOREM FOR THE TYPE FREE λσ-CALCULUS

In this section we offer a proof of the Church-Rosser property for β-reduction in type free λσ-calculus. The proof is basically along the lines of the proof given in Barendregt [81], pp. 279 - 289, employing so-called "finiteness of developments". The main theorem in this section states that the strong normalization property holds for a special kind of reduction (called β') in λσ. From this theorem together with the weak Church-Rosser property for β' (proved in Section 3.1) it will be shown that the Church-Rosser property holds for β-reduction in general.

3.1. Restricted reduction and the weak Church-Rosser property

In this section we introduce an extension of the set of λσ-terms by marking certain redices.

Definition 3.1.1 (Λ').
Let Λ denote the set of λσ-terms. The set Λ' is the smallest set X satisfying

(i) $\Lambda \subset X$;
(ii) $t \in X \Rightarrow \lambda t \in X$;
(iii) $t \in X \rightarrow \sigma(p)t \in X$, for every $p \in \mathbf{N} \times \mathbf{M}$;
(iv) $u,v \in X \Rightarrow \delta uv \in X$;
(v) $u,v \in X \Rightarrow \delta u \lambda' v \in X$. □

The elements of Λ' are called λ'σ-*terms*. The main difference between the definitions of Λ and Λ' lies in clause (v) in which redices are marked. This difference is essential, since only marked redices are contracted in Λ' (cf. Definition 3.1.2 below). But apart from differences regarding contractions of redices it is easily seen that the basic operations on λσ-terms introduced in Section 2 can be extended to λ'σ-terms in an obvious way, and moreover that the results obtained in Section 2 regarding these operations also hold for λ'σ-terms. In particular:

(1) If μ is a refmap then the application of $\underline{\mu}$ to a λ'σ-term t is defined as in Definition 2.10 with the additional clause that $\underline{\mu}\, \lambda' t$

is defined as $\lambda' \underline{\mu^{<1>}}\, t$. It is easily seen that Lemmas 2.3 and 2.4 also hold for $\lambda'\sigma$-terms.

(2) The weight $W(t)$ of a $\lambda'\sigma$-term t is defined as in Definition 2.12 with the additional clause that $W(\lambda' t)$ is defined as $1 + W(t)$. It is easily seen that Lemma 2.5 also holds for $\lambda'\sigma$-terms.

(3) Substitution of $\lambda'\sigma$-terms is defined as in Definition 2.11, though with two exceptions. The first exception is, of course, the addition of an extra clause telling us how to recursively shift the substitution operator, denoted by Σ' in Λ', past an abstractor λ'. This is done as follows. If u and w are $\lambda'\sigma$-terms and k is an element of $\mathbb{N}$ then $\Sigma'(u,\lambda' w,k)$ is simply defined as $\lambda' \Sigma'(u,w,k+1)$. The other exception is the adaption of clause (ii) in Definition 2.11 which tells how to evaluate a substitution at an end-point $\omega(\psi)$ of a segment. We recall that $\Sigma(u,\omega(\psi),k)$ was defined in Λ as $\delta\, \underline{\varphi_{k-1}}\, u\, \lambda\, \underline{\vartheta_{k-1}}\, \omega(\psi)$, if k is an element of $\mathbb{N}$ such that $k \in rge(\psi)$. In Λ' we have a different situation to take account of. In Λ, substitutions are the result of contracting some β-redex, whereas in Λ' substitutions can only be the result of contracting some β'-redex. It is for this reason that we define $\Sigma(u,\omega(\psi),k)$ in Λ' as $\delta\, \underline{\varphi_{k-1}}\, u\, \lambda'\, \underline{\vartheta_{k-1}}\, \omega(\psi)$, where λ' corresponds to the λ' of the β'-redex that gave rise to this substitution (see also the discussion of clause (ii) offered in Section 2 pp. 53 - 55). By checking the proofs of Theorems 2.1, 2.2 and 2.3 it is again easily seen that the results stated in these theorems also hold in Λ'.

Definition 3.1.2 ($\rightarrow_{\beta'}$).
The binary relation β' on Λ' is defined as follows.

(1) If $t,u \in \Lambda'$ then

$$t\, \beta'\, u \Leftrightarrow \exists v,w : t = \delta v \lambda' w \wedge u = \Sigma'(v,w,1) \wedge (v,w,1) \in dom(\Sigma') .$$

If $t\, \beta'\, u$ then t is called a β'-*redex*.

(2) The notion of reduction $\rightarrow_{\beta'}$ on Λ' is inductively defined by

(i) $u\, \beta'\, v \Rightarrow u \rightarrow_{\beta'} v$;

(ii) $u \rightarrow_{\beta'} v \Rightarrow \lambda u \rightarrow_{\beta'} \lambda v$;

(iii) $u \rightarrow_{\beta'} v \Rightarrow \lambda' u \rightarrow_{\beta'} \lambda' v$;

(iv) $u \to_\beta v \Rightarrow \sigma(n,m)u \to_\beta \sigma(n,m)v$;

(v) $u \to_\beta v \Rightarrow \delta uw \to_\beta \delta vw$;

(vi) $u \to_\beta v \Rightarrow \delta wu \to_\beta \delta wv$. □

Theorem 3.1.1.

Let $u,v \in \Lambda'$ and μ be a refmap.

If $u \to_\beta v$ then $\underline{\mu} u \to_\beta \underline{\mu} v$.

Proof. By induction on the generation of $\to_\beta$, and Theorem 2.2. □

Important.

We note that in the following lemmas and theorems concerning substitution it is tacitly assumed that the substitutions involved are indeed defined. We do this for the sole reason of economy of expression. Furthermore these lemmas and theorems are only secondary, in the sense that they are used as auxiliary results in proofs concerning contractions of redices, and since contractions of redices presuppose the well-definedness of their corresponding substitutions there is one reason less for fearing the omission of the explicit mentioning of well-definedness of the substitutions involved (- but, none the less, the following lemmas and theorems can only hold if the substitutions are indeed defined).

Lemma 3.1.1.

Let $w,s \,\&\, w,u \in \Lambda'$ and $m \in \mathbb{M}$ and $k \in \mathbb{N}$.

If $k + W(s\omega(id_m)) > m \geq W(s\omega(id_m))$ then

$$\Sigma'(u,s \,\&\, w,k) = A \,\&\, \Sigma'(u,w,k + W(s\omega(id_m)) ,$$

where $\Sigma'(u,s\omega(id_m)) = A \,\&\, \omega(id_m)$.

Proof. By induction on $L(s)$ and the definition of substitution. □

Lemma 3.1.2.

Let $u,v \in \Lambda'$ and $m \in \mathbb{M}$ and $k \in \mathbb{N}$.

If $\psi \in Perm(p)$ and $p \leq m$ then

$$\Sigma'(u,\underline{\psi}\hat{}\, v,k+m) = \underline{\psi}\hat{}\, \Sigma'(u,v,k+m) .$$

Proof. From Theorem 2.1 and Lemma 2.4 it follows that

$$\Sigma'(u,\underline{\psi}\hat{}\, v,k+m) =$$

$$= \Sigma'(\underline{\varphi_{k+m-1}}\, u,\underline{\vartheta_{k+m-1}\circ\psi\hat{}}\, v,1) \ .$$

Furthermore

(1) $\varphi_{k+m-1} = \psi\hat{}\circ\varphi_{k+m-1}$ (simple computation), and

(2) $\vartheta_{k+m-1}\circ\psi\hat{} = (\psi\hat{})^{<1>}\circ\vartheta_{k+m-1}$.

Result (2) is established by the following calculations. If n is an element of $\mathbb{N}$ then

$$\vartheta_{k+m-1}(\psi\hat{}(n)) = \begin{cases} \psi\hat{}(n)+1, & \text{if } 1 \le n \le p \\ n+1, & \text{if } p < n \le k+m-1 \\ 1, & \text{if } n = k+m \\ n, & \text{if } n > k+m \end{cases}$$

and

$$(\psi\hat{})^{<1>}(\vartheta_{k+m-1}(n)) = \begin{cases} (\psi\hat{})^{<1>}(n+1), & \text{if } 1 \le n \le k+m-1 \\ (\psi\hat{})^{<1>}(1), & \text{if } n = k+m \\ (\psi\hat{})^{<1>}(n), & \text{if } n > k+m \end{cases} =$$

$$= \begin{cases} 1+\psi\hat{}(n), & \text{if } 1 \le n \le p \\ 1+n, & \text{if } p < n \le k+m-1 \\ 1, & \text{if } n = k+m \\ n, & \text{if } n > k+m \end{cases} .$$

By (1) and (2) we have

$$\Sigma'(\underline{\varphi_{k+m-1}}\, u,\underline{\vartheta_{k+m-1}\circ\psi\hat{}}\, v,1) =$$

$$= \Sigma'(\underline{\psi\hat{}\circ\varphi_{k+m-1}}\, u,\underline{(\psi\hat{})^{<1>}\circ\vartheta_{k+m-1}}\, v,1) =$$

$$= \underline{\psi}\hat{}\,\Sigma'(\underline{\varphi_{k+m-1}}\, u,\underline{\vartheta_{k+m-1}}\, v,1) = \qquad \text{(Theorem 2.2)}$$

$$= \underline{\psi}\hat{}\,\Sigma'(u,v,k+m) \ . \qquad \text{(Theorem 2.1)} \qquad \square$$

Theorem 3.1.2.

Let $u,w,s \,\&\, w \in \Lambda'$ and $k \in \mathbb{M}$ and $k \in \mathbb{N}$. If $\psi \in \mathrm{Perm}(p)$ and $W(s\omega(\psi)) = m \geq p$ then

$$\Sigma'(u,s\,\underline{\psi}\hat{}\,w,k) = A \,\&\, \underline{\psi}\hat{}\,\Sigma'(u,w,k+m) \ ,$$

where $\Sigma'(u,s\omega(\psi),k) = A \,\&\, \omega(\psi)$.

Proof.

$$\Sigma'(u,s\,\underline{\psi}\hat{}\,w,k) =$$

$$= A \,\&\, \Sigma'(u,\underline{\psi}\hat{}\,w,k+m) \ ,$$

where $\Sigma'(u,s\omega(id_m),k) = A \,\&\, \omega(id_m)$ (cf. Lemma 3.1.1).

$$A \,\&\, \Sigma'(u,\underline{\psi}\hat{}\,w,k+m) =$$

$$= A \,\&\, \Sigma'(u,\underline{\psi}\hat{}\,w,(k+m-p)+p) =$$

$$= A \,\&\, \underline{\psi}\hat{}\,\Sigma'(u,w,(k+m-p)+p) = \qquad \text{(Lemma 3.1.2)}$$

$$= A \,\&\, \underline{\psi}\hat{}\,\Sigma'(u,w,k+m) \ . \qquad \square$$

Theorem 3.1.3.

Let $u,v,w \in \Lambda'$ and $k,\ell \in \mathbb{N}$. If $k \geq \ell$ then

$$\Sigma'(u,\Sigma'(v,w,\ell),k) \ \overset{*}{\underset{\beta'}{\rightleftarrows}} \ \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,w,k+1),\ell) \ .$$

Proof. By induction on $L(w)$. Let S_1 and S_2 denote $\Sigma'(u,\Sigma'(v,w,\ell),k)$ and $\Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,w,k+1),\ell)$, respectively. In this proof we shall write $\overset{*}{\leftrightarrow}_{\beta'}$ for $\overset{*}{\underset{\beta'}{\rightleftarrows}}$, i.e.

$$A \overset{*}{\leftrightarrow}_{\beta'} B \Leftrightarrow A \overset{*}{\rightarrow}_{\beta'} B \wedge B \overset{*}{\rightarrow}_{\beta'} A \ ,$$

for $\lambda'\sigma$-terms A and B.

(i) $w = \xi(n)$: $S_1 = \Sigma'(u,\Sigma'(v,\xi(n),\ell),k)$

(1) $n = \ell$: $S_1 = \Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)$

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\xi(n),k+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k+\ell-1),\xi(n),\ell) =$ ($k \geq \ell$ and $\ell = n$, therefore $k+1 > n$)

$= \underline{\varphi_{\ell-1}}\, \Sigma'(u,v,k+\ell-1) =$

$= \Sigma'(u,\underline{\varphi_{\ell-1}}\, v,k)$. (cor. to Th. 2.2)

Conclusion: $S_1 = S_2$.

(2) $n < \ell$:

$S_1 =$

$= \Sigma'(u,\xi(n),k) =$

$= \xi(n)$ $(n < \ell \leq k)$.

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\xi(n),k+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\xi(n),\ell) =$

$= \xi(n)$.

Conclusion: $S_1 = S_2$.

(3) $n > \ell$:

$S_1 =$

$= \Sigma'(u,\Sigma'(v,\xi(n),\ell),k) =$

$= \Sigma'(u,\xi(n-1),k)$.

(3.1) $n > \ell$ and $n-1 = k$: $S_1 = \underline{\varphi_{k-1}}\, u$

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\xi(n),k+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\underline{\varphi_k}\,u,\ell) =$

$= \underline{\varphi_{k-1}}\,u$. (Theorem 2.3)

Conclusion: $S_1 = S_2$.

(3.2) $n > \ell$ and $n-1 < k$: $S_1 = \xi(n-1)$

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\xi(n),\ell) =$

$= \xi(n-1)$ $(n > \ell)$.

Conclusion: $S_1 = S_2$.

(3.3) $n > \ell$ and $n-1 > k$: $S_1 = \xi(n-2)$

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\xi(n-1),\ell) =$

$= \xi(n-2)$ $(n-1 > k \geq \ell)$.

Conclusion: $S_1 = S_2$.

(ii) $w = \omega(\psi)$:

(1) $k+1 \in \mathrm{rge}(\psi)$: If $k+1 \in \mathrm{rge}(\psi)$ then $\psi \in \mathrm{Perm}(p)$, for some $p \geq k+1$; therefore $\ell \in \mathrm{rge}(\psi)$.

$S_1 =$

$= \Sigma'(u,\Sigma'(v,\omega(\psi),\ell),k) =$

$= \Sigma'(u,\delta\,\underline{\varphi_{\ell-1}}\,v\,\lambda'\,\underline{\vartheta_{\ell-1}}\,\omega(\psi),k) =$

$= \delta\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\,\lambda'\,\Sigma'(u,\underline{\vartheta_{\ell-1}}\,\omega(\psi),k+1)$.

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\omega(\psi),k+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\delta\,\underline{\varphi_k}\,u\,\lambda'\,\underline{\vartheta_k}\,\omega(\psi),\ell) =$

$= \delta\,\Sigma'(\Sigma'(u,v,k-\ell+1),\underline{\varphi_k}\,u,\ell)\;\lambda'\;\Sigma'(\Sigma'(u,v,k-\ell+1),$

$\underline{\vartheta_k}\,\omega(\psi),\ell+1) =$

$= \delta\,\underline{\varphi_{k-1}}\,u\,\lambda'\;\Sigma'(\Sigma'(u,v,k-\ell+1),\underline{\vartheta_k}\,\omega(\psi),\ell+1) =$ (Theorem 2.3)

$= \delta\,\underline{\varphi_{k-1}}\,u\,\lambda'\;\Sigma'(\underline{\varphi_\ell}\,\Sigma'(u,v,k-\ell+1),\underline{\vartheta_\ell \circ \vartheta_k}\,\omega(\psi),1) =$ (Theorem 2.1)

$= \delta\,\underline{\varphi_{k-1}}\,u\,\lambda'\;\Sigma'(\underline{\varphi_1}\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k),\underline{\vartheta_\ell \circ \vartheta_k}\,\omega(\psi),1) =$

(cor. to Theorem 2.2 and $\varphi_\ell = \varphi_1 \circ \varphi_{\ell-1}$)

$= \delta\,\underline{\varphi_{k-1}}\,u\,\lambda'\;\Sigma'(\underline{\varphi_1}\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k),\omega(\vartheta_\ell \circ \vartheta_k \circ \psi),1)$.

Furthermore

$1 \in rge(\vartheta_\ell \circ \vartheta_k \circ \psi) \Leftrightarrow \ell+1 \in rge(\vartheta_k \circ \psi) \Leftrightarrow \ell \in rge(\psi)$.

Hence

$S_2 =$

$= \delta\,\underline{\varphi_{k-1}}\,u\,\lambda'\;\delta\,\underline{\varphi_1}\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\;\lambda'\;\omega(\vartheta_\ell \circ \vartheta_k \circ \psi) =$

$\rightarrow_{\beta'}\;\Sigma'(\underline{\varphi_{k-1}}\,u,\delta\,\underline{\varphi_1}\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\;\lambda'\;\omega(\vartheta_\ell \circ \vartheta_k \circ \psi),1) =$

$= \delta\,\Sigma'(\underline{\varphi_{k-1}}\,u,\underline{\varphi_1}\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k),1)\;\lambda'\;\Sigma'(\underline{\varphi_{k-1}}\,u,$

$\omega(\vartheta_\ell \circ \vartheta_k \circ \psi),2) =$

$= \delta\,\underline{\varphi_0}\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\;\lambda'\;\Sigma'(\underline{\varphi_1 \circ \varphi_{k-1}}\,u,\underline{\vartheta_1}\,\omega(\vartheta_\ell \circ \vartheta_k \circ \psi),1) =$

(Theorems 2.1 and 2.3)

$= \delta\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\;\lambda'\;\Sigma'(\underline{\varphi_k}\,u,\omega(\vartheta_1 \circ \vartheta_\ell \circ \vartheta_k \circ \psi),1)$.

Furthermore

$\vartheta_1 \circ \vartheta_\ell = \vartheta_{\ell-1}^{<1>}$, hence $\vartheta_1 \circ \vartheta_\ell \circ \vartheta_k = \vartheta_{\ell-1}^{<1>} \circ \vartheta_k = \vartheta_k \circ \vartheta_{\ell-1}$

(Lemma 2.2), and therefore

$$\delta\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\;\lambda'\,\Sigma'(\underline{\varphi_k}\,u,\omega(\vartheta_1\circ\vartheta_\ell\circ\vartheta_k\circ\psi),1) =$$

$$= \delta\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\;\lambda'\,\Sigma'(\underline{\varphi_k}\,u,\underline{\vartheta_k\circ\vartheta_{\ell-1}}\,\omega(\psi),1) =$$

$$= \delta\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\;\lambda'\,\Sigma'(u,\underline{\vartheta_{\ell-1}}\,\omega(\psi),k+1)\ . \qquad \text{(Theorem 2.1)}$$

Conclusion: $S_2 \rightarrow_{\beta'} S_1$. (*)

And also

$$S_1 =$$

$$= \delta\,\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k)\;\lambda'\,\Sigma'(u,\underline{\vartheta_{\ell-1}}\,\omega(\psi),k+1) =$$

$$\rightarrow_{\beta'} \Sigma'(\Sigma'(u,\underline{\varphi_{\ell-1}}\,v,k),\Sigma'(u,\underline{\vartheta_{\ell-1}}\,\omega(\psi),k+1),1) =$$

$$= \Sigma'(\underline{\varphi_{\ell-1}}\,\Sigma'(u,v,k-\ell+1),\Sigma'(\underline{\varphi_k}\,u,\omega(\vartheta_k\circ\vartheta_{\ell-1}\circ\psi),1),1)\ .$$

(Theorem 2.1 and cor. to Theorem 2.2)

Furthermore, if $k \geq \ell$ and $p \geq k+1$ then

(1) $\vartheta_{\ell-1}\circ\varphi_k = \varphi_k$ (computation) ;

(2) $\vartheta_k\circ\vartheta_{\ell-1}\circ\psi = \vartheta_{\ell-1}^{<1>}\circ\vartheta_k\circ\psi$ (computation) .

Therefore

$$\Sigma'(\underline{\varphi_{\ell-1}}\,\Sigma'(u,v,k-\ell+1),\Sigma'(\underline{\varphi_k}\,u,\omega(\vartheta_k\circ\vartheta_{\ell-1}\circ\psi),1),1) =$$

$$= \Sigma'(\underline{\varphi_{\ell-1}}\,\Sigma'(u,v,k-\ell+1),\Sigma'(\underline{\vartheta_{\ell-1}\circ\varphi_k}\,u,\underline{\vartheta_{\ell-1}^{<1>}}\,\omega(\vartheta_k\circ\psi),1),1) =$$

$$= \Sigma'(\underline{\varphi_{\ell-1}}\,\Sigma'(u,v,k-\ell+1),\underline{\vartheta_{\ell-1}}\,\Sigma'(\underline{\varphi_k}\,u,\underline{\vartheta_k}\,\omega(\psi),1),1) =$$

(Theorem 2.2)

$$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\omega(\psi),k+1),\ell) =$$

(Theorem 2.1 and cor. to Theorem 2.2)

$$= S_2\ .$$

Conclusion: $S_1 \rightarrow_{\beta'} S_2$. (**)

From (*) and (**) it follows that $S_1 \overset{*}{\underset{\beta'}{\leftrightarrow}} S_2$.

(2) $k+1 \notin \mathrm{rge}(\psi)$ and $\ell \in \mathrm{rge}(\psi)$:

$S_1 =$

$= \Sigma'(u,\Sigma'(v,\omega(\psi),\ell),k) =$

$= \Sigma'(u,\delta\, \underline{\varphi_{\ell-1}}\, v\, \lambda'\, \underline{\vartheta_{\ell-1}}\, \omega(\psi),k) =$

$= \delta\, \Sigma'(u,\underline{\varphi_{\ell-1}}\, v,k)\, \lambda'\, \Sigma'(u,\underline{\vartheta_{\ell-1}}\, \omega(\psi),k+1)$.

Furthermore

$k+1 \in \mathrm{rge}(\vartheta_{\ell-1} \circ \psi) \Leftrightarrow k+1 \in \mathrm{rge}(\psi) \qquad (k+1 > k \geq \ell)$,

hence $S_1 = \delta\, \Sigma'(u,\underline{\varphi_{\ell-1}}\, v,k)\, \lambda'\, \omega(\vartheta_k \circ \vartheta_{\ell-1} \circ \psi - 1)$.

From $\mathrm{rge}(\psi) \subseteq \mathbb{N}_k$ and $\ell \leq k$ it follows that $\mathrm{rge}(\vartheta_{\ell-1} \circ \psi) \subseteq$ $\subseteq \mathbb{N}_k$, and therefore $\vartheta_k \circ \vartheta_{\ell-1} \circ \psi - 1 = \vartheta_{\ell-1} \circ \psi$. Hence

$S_1 = \delta\, \Sigma'(u,\underline{\varphi_{\ell-1}}\, v,k)\, \lambda'\, \omega(\vartheta_{\ell-1} \circ \psi)$.

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\omega(\psi),k+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\omega(\vartheta_k \circ \psi - 1),\ell) \qquad (k+1 \notin \mathrm{rge}(\psi))$.

Furthermore

$\ell \in \mathrm{rge}(\vartheta_k \circ \psi - 1) \Leftrightarrow \ell+1 \in \mathrm{rge}(\vartheta_k \circ \psi) \Leftrightarrow \ell \in \mathrm{rge}(\psi) \quad (\ell \leq k)$.

Hence

$S_2 =$

$= \delta\, \underline{\varphi_{\ell-1}}\, \Sigma'(u,v,k-\ell+1)\, \lambda'\, \underline{\vartheta_{\ell-1}}\, \omega(\vartheta_k \circ \psi - 1) =$

$= \delta\, \Sigma'(u,\underline{\varphi_{\ell-1}}\, v,k)\, \lambda'\, \omega(\vartheta_{\ell-1} \circ (\vartheta_k \circ \psi - 1))$. (Theorem 2.3)

From $\mathrm{rge}(\psi) \subseteq \mathbb{N}_k$ it follows that $\vartheta_k \circ \psi - 1 = \psi$, and therefore

$S_2 =$

$\delta\, \Sigma'(u, \underline{\varphi_{\ell-1}}\, v, k)\ \lambda'\, \omega(\vartheta_{\ell-1} \circ \psi) =$

$= S_1 .$

(3) $\ell \notin \mathrm{rge}(\psi)$: If $\ell \in \mathrm{rge}(\psi)$ then $k, k+1 \notin \mathrm{rge}(\psi)$

$S_1 =$

$= \Sigma'(u, \Sigma'(v, \omega(\psi), \ell), k) =$

$= \Sigma'(u, \omega(\vartheta_{\ell-1} \circ \psi - 1), k) =$

$= \Sigma'(u, \omega(\psi), k) =$

$= \omega(\vartheta_{k-1} \circ \psi - 1) =$

$= \omega(\psi) .$

$S_2 =$

$= \Sigma'(\Sigma'(u, v, k - \ell + 1), \Sigma'(u, \omega(\psi), k+1), \ell) =$

$= \Sigma'(\Sigma'(u, v, k - \ell + 1), \omega(\vartheta_k \circ \psi - 1), \ell) =$

$= \Sigma'(\Sigma'(u, v, k - \ell + 1), \omega(\psi), \ell) =$

$= \omega(\vartheta_{\ell-1} \circ \psi - 1) =$

$= \omega(\psi) .$

Conclusion: $S_1 = S_2$.

(iii) $w = \lambda w_1$ or $w = \lambda' w_1$: This case follows simply from the induction hypothesis.

(iv) $w = \sigma(n,m) w_1$:

(1) $n < \ell$:

$S_1 =$

$= \Sigma'(u, \Sigma'(v, \sigma(n,m) w_1, \ell), k) =$

$= \Sigma'(u,\sigma(n,m)\,\Sigma'(v,w_1,\ell+m),k) =$

$= \sigma(n,m)\,\Sigma'(u,\Sigma'(v,w_1,\ell+m),k+m) \overset{*}{\underset{\beta'}{\leftrightarrow}}$

$\overset{*}{\underset{\beta'}{\leftrightarrow}} \sigma(n,m)\,\Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,w_1,k+m+1),\ell+m) =$

(induction hypothesis)

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\sigma(n,m)\,\Sigma'(u,w_1,k+m+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\sigma(n,m)w_1,k+1),\ell) =$

$= S_2$.

(2) $n = \ell$: If $\varphi_{\underline{\ell-1}}\, v = s\omega(\psi)$ then $S_1 = \Sigma'(u,s\,\underline{\psi}\hat{}\,\Sigma'(v,w_1,\ell+m),k)$.
Since $W(v) = W(\varphi_{\underline{\ell-1}}\, v) = W(s\omega(\psi)) = m$ and $\psi \in \mathrm{Perm}(p)$ for some $p \leq m$, it follows from Theorem 3.1.1 that

$S_1 = A \;\&\; \underline{\psi}\hat{}\,\Sigma'(u,\Sigma'(v,w_1,\ell+m),k+m)$,

where $\Sigma'(u,s\omega(\psi),k) = A \;\&\; \omega(\psi)$.
By the induction hypothesis we have

$\Sigma'(u,\Sigma'(v,w_1,\ell+m),k+m) \overset{*}{\underset{\beta'}{\leftrightarrow}} \Sigma'(\Sigma'(u,v,k-\ell+1),$

$\Sigma'(u,w_1,k+m+1),\ell+m)$

and from Theorem 3.1.1 it follows that

$\underline{\psi}\hat{}\,\Sigma'(u,\Sigma'(v,w_1,\ell+m),k+m) \overset{*}{\underset{\beta'}{\leftrightarrow}} \underline{\psi}\hat{}\,\Sigma'(\Sigma'(u,v,k-\ell+1),$

$\Sigma'(u,w_1,k+m+1),\ell+m)$

and therefore

$S_1 \overset{*}{\underset{\beta'}{\leftrightarrow}} A \;\&\; \underline{\psi}\hat{}\,\Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,w_1,k+m+1),\ell+m)$.

Furthermore

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\sigma(n,m)w_1,k+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\sigma(n,m)\,\Sigma'(u,w_1,k+m+1),\ell)$

$(n = \ell < k+1)$.

Since

$\underline{\varphi_{\ell-1}}\ \Sigma'(u,v,k-\ell+1) =$

$= \Sigma'(u,\underline{\varphi_{\ell-1}}\ v,k) =$ (cor. to Theorem 2.2)

$= \Sigma'(u,s\omega(\psi),k) =$

$= A \ \& \ \omega(\psi)$

it follows that $S_2 = A \ \& \ \underline{\psi}\hat{}\ \Sigma'(\Sigma'(u,v,k-\ell+1), \Sigma'(u,w_1,k+m+1),\ell+m)$.

Conclusion: $S_1 \overset{*}{\underset{\beta'}{\leftrightarrow}} S_2$.

(3) $n > \ell$:

$S_1 =$

$= \Sigma'(u,\Sigma'(v,\sigma(n,m)w_1,\ell),k) =$

$= \Sigma'(u,\sigma(n-1,m)\ \Sigma'(v,w_1,\ell+m),k)$.

(3.1) $n < k+1$:

$S_1 =$

$= \sigma(n-1,m)\ \Sigma'(u,\Sigma'(v,w_1,\ell+m),k+m) \overset{*}{\underset{\beta'}{\leftrightarrow}}$

$\overset{*}{\underset{\beta'}{\leftrightarrow}}\ \sigma(n-1,m)\ \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,w_1,k+m+1),\ell+m) =$

(induction hypothesis)

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\sigma(n,m)\ \Sigma'(u,w_1,k+m+1),\ell)$

$(n > \ell)$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\sigma(n,m)w_1,k+1),\ell)$

$(n < k+1)$

$= S_2$.

Conclusion: $S_1 \overset{*}{\underset{\beta'}{\leftrightarrow}} S_2$.

(3.2) $n = k+1$:

$S_1 =$

$= s_1 \underline{\psi}\hat{}\, \Sigma'(u,\Sigma'(v,w_1,\ell+m),k+m)$,

where $\underline{\varphi_{k-1}}\, u = s_1\omega(\psi)$.

By the induction hypothesis and Theorem 3.1.1 we have

$S_1 \overset{*}{\underset{\beta'}{\leftrightarrow}} s_1 \underline{\psi}\hat{}\, \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,w_1,k+m+1),\ell+m)$.

Furthermore

$S_2 =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\sigma(n,m)w_1,k+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),s_2 \underline{\psi}\hat{}\, \Sigma'(u,w_1,k+m+1),\ell)$,

where $\underline{\varphi_k}\, u = s_2\omega(\psi)$.

From Theorem 2.3 it follows that

$\Sigma'(\Sigma'(u,v,k-\ell+1),\underline{\varphi_k}\, u,\ell) = \underline{\varphi_{k-1}}\, u$, since $k \geq \ell$.

From Theorem 3.1.2 and $\underline{\varphi_{k-1}}\, u = s_1\omega(\psi)$ it follows that

$S_2 = s_1 \underline{\psi}\hat{}\, \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,w_1,k+m+1),\ell+m)$.

Conclusion: $S_1 \overset{*}{\underset{\beta'}{\leftrightarrow}} S_2$.

(3.3) $n > k+1$:

$S_1 =$

$= \Sigma'(u,\Sigma'(v,\sigma(n,m)w_1,\ell),k) =$

$= \Sigma'(u,\sigma(n-1,m)\ \Sigma'(v,w_1,\ell+m),k) =$

$= \sigma(n-2,m)\ \Sigma'(u,\Sigma'(v,w_1,\ell+m),k+m) \overset{*}{\underset{\beta'}{\leftrightarrow}}$

$\overset{*}{\underset{\beta'}{\leftrightarrow}} \sigma(n-2,m)\ \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,w_1,k+m+1),\ell+m) =$

(induction hypothesis)

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\sigma(n-1,m)\ \Sigma'(u,w_1,k+m+1),\ell) =$

$= \Sigma'(\Sigma'(u,v,k-\ell+1),\Sigma'(u,\sigma(n,m)w_1,k+1),\ell) =$

$= S_2$.

Conclusion: $S_1 \overset{*}{\underset{\beta'}{\leftrightarrow}} S_2$.

(v) $w = \delta w_1 w_2$: This case follows simply from the induction hypothesis. □

Theorem 3.1.4.

Let $u,v,w \in \Lambda'$ and $k \in \mathbf{N}$.

If $v \rightarrow_{\beta'} w$ then $\Sigma'(u,v,k) \overset{=}{\downarrow_{\beta'}} \Sigma'(u,w,k)$.

Proof. By induction on the generation of $\rightarrow_{\beta'}$.

(i) $v = \delta P \lambda' Q$ and $w = \Sigma'(P,Q,1)$:

$\Sigma'(u,v,k) =$

$= \delta\, \Sigma'(u,P,k)\ \lambda'\, \Sigma'(u,Q,k+1) \rightarrow_{\beta'}$

$\rightarrow_{\beta'} \Sigma'(\Sigma'(u,P,k),\Sigma'(u,Q,k+1),1)$,

$\overset{*}{\underset{\beta'}{\leftrightarrow}} \Sigma'(u,\Sigma'(P,Q,1),k) =$ (Theorem 3.1.3)

$= \Sigma'(u,w,k)$.

Conclusion: $\Sigma'(u,v,k) \overset{=}{\downarrow_{\beta'}} \Sigma'(u,w,k)$.

(ii) $v = \lambda v_1 (\lambda' v_1)$, $v_1 \rightarrow_{\beta'} w_1$ and $w = \lambda w_1 (\lambda' w_1)$:

$\Sigma'(u,v,k) =$

$= \lambda\, \Sigma'(u,v_1,k+1) \overset{=}{\downarrow_{\beta'}}$

$\overset{=}{\downarrow_{\beta'}} \lambda\, \Sigma'(u,w_1,k+1) =$ (induction hypothesis)

$= \Sigma'(u,w,k+1)$.

(iii) $v = \sigma(n,m) v_1$, $v_1 \rightarrow_{\beta'} w_1$ and $w = \sigma(n,m) w_1$:

(1) $n < k$:

$\Sigma'(u,v,k) =$

$= \sigma(n,m)\, \Sigma'(u,v_1,k+m) \overset{=}{\downarrow_{\beta'}}$

$\overset{=}{\downarrow}_{\beta'}$ $\sigma(n,m)\ \Sigma'(u,w_1,k+m)$ = (induction hypothesis)

= $\Sigma'(u,w,k)$.

(2) $n = k$: If $\underline{\varphi_{k-1}}\, u = s\omega(\psi)$ then

$\Sigma'(u,v,k)$ =

= $s\,\underline{\psi}\hat{}\,\Sigma'(u,v_1,k+m)$ $\overset{=}{\downarrow}_{\beta'}$

$\overset{=}{\downarrow}_{\beta'}$ $s\,\underline{\psi}\hat{}\,\Sigma'(u,w_1,k+m)$ = (induction hypothesis and Theorem 3.1.1)

= $\Sigma'(u,\sigma(n,m)w_1,k)$ =

= $\Sigma'(u,w,k)$.

(3) $n > k$:

$\Sigma'(u,v,k)$ =

= $\sigma(n-1,m)\ \Sigma'(u,v_1,k+m)$ $\overset{=}{\downarrow}_{\beta'}$

$\overset{=}{\downarrow}_{\beta'}$ $\sigma(n-1,m)\ \Sigma'(u,w_1,k+m)$ =

= $\Sigma'(u,\sigma(n,m)w_1,k)$ =

= $\Sigma'(u,w,k)$.

(iv) $v = \delta v_1 v_2$, $v_1 \twoheadrightarrow_{\beta'} w_1$ and $w = \delta w_1 w_2$:

$\Sigma'(u,v,t)$ =

= $\delta\,\Sigma'(u,v_1,k)\ \Sigma'(u,v_2,k)$ $\overset{=}{\downarrow}_{\beta'}$

$\overset{=}{\downarrow}_{\beta'}$ $\delta\,\Sigma'(u,w_1,k)\ \Sigma'(u,v_2,k)$ = (induction hypothesis)

= $\Sigma'(u,w,k)$.

(v) $v = \delta v_1 v_2$, $v_2 \twoheadrightarrow_{\beta'} w_2$ and $w = \delta v_1 w_2$: as in (5). □

Theorem 3.1.5.

Let $u,v,w \in \Lambda'$ and $k \in \mathbb{N}$.

If $u \twoheadrightarrow_{\beta'} v$ then $\Sigma'(u,w,k)$ $\overset{=}{\downarrow}_{\beta'}$ $\Sigma'(v,w,k)$.

Proof. By induction on $L(w)$.

(i) $w = \xi(n)$:

(1) $n < k$: $\Sigma'(u,\xi(n),k) = \xi(n) = \Sigma'(v,\xi(n),k)$.

(2) $n = k$:

$\Sigma'(u,\xi(n),k) =$

$= \underline{\varphi_{k-1}}\, u \rightarrow_{\beta'}$

$\rightarrow_{\beta'} \underline{\varphi_{k-1}}\, v =$ (Theorem 3.1.1)

$= \Sigma'(v,\xi(n),k)$.

(3) $n > k$: $\Sigma'(u,\xi(n),k) = \xi(n-1) = \Sigma'(v,\xi(n),k)$.

(ii) $w = \omega(\psi)$:

(1) $k \in rge(\psi)$:

$\Sigma'(u,\omega(\psi),k) =$

$= \delta\, \underline{\varphi_{k-1}}\, u \lambda'\, \underline{\vartheta_{k-1}}\, \omega(\psi) \rightarrow_{\beta'}$

$\rightarrow_{\beta'} \delta\, \underline{\varphi_{k-1}}\, v \lambda'\, \underline{\vartheta_{k-1}}\, \omega(\psi) =$ (Theorem 3.1.1)

$= \Sigma'(v,\omega(\psi),k)$.

(2) $k \notin rge(\psi)$: $\Sigma'(u,\omega(\psi),k) = \omega(\vartheta_{k-1} \circ \psi - 1) = \Sigma'(v,\omega(\psi),k)$.

(iii) $w = \lambda w_1$ or $w = \lambda' w_1$: this case follows simply from the induction hypothesis.

(iv) $w = \sigma(n,m) w_1$:

(1) $n < k$:

$\Sigma'(u,\sigma(n,m)w_1,k)$

$= \sigma(n,m)\, \Sigma'(u,w_1,k+m) \overset{=}{\downarrow}_{\beta'}$

$\overset{=}{\downarrow}_{\beta'} \sigma(n,m)\, \Sigma'(v,w_1,k+m) =$ (induction hypothesis)

$= \Sigma'(v,\sigma(n,m)w_1,k) =$

$= \Sigma'(v,w,k)$.

(2) $n = k$: We shall prove the following statement by induction on $L(s)$.

Claim: Let $X,Y,s\omega(\psi) \in \Lambda'$. If $X \downarrow_\beta^= Y$ and $s\omega(\psi) \overset{=}{\rightarrow}_\beta s'\omega(\varphi)$ then $s\,\underline{\psi}\hat{}\,X \downarrow_\beta^= s'\underline{\varphi}\hat{}\,Y$.

(2.1) $L(s) = 0$: In this case $s = s' = \emptyset$ and $\psi = \varphi$. Furthermore, by Theorem 3.1.1 $\underline{\psi}\hat{}\,X \downarrow_\beta^= \underline{\psi}\hat{}\,Y$.

(2.2) $s = \lambda s_1$, $s_1\omega(\psi) \overset{=}{\rightarrow}_\beta s_1'\omega(\varphi)$ and $s' = \lambda s_1'$: From the induction hypothesis it follows that $s_1\,\underline{\psi}\hat{}\,X \downarrow_\beta^= s_1'\,\underline{\varphi}\hat{}\,Y$ and therefore $\lambda s_1\,\underline{\psi}\hat{}\,X \downarrow_\beta^= \lambda s_1'\,\underline{\varphi}\hat{}\,Y$.

(2.3) $s = \lambda' s_1$, $s_1\omega(\psi) \overset{=}{\rightarrow}_\beta s_1'\omega(\varphi)$ and $s' = \lambda' s_1'$: as in (2.2).

(2.4) $s = \sigma(p,q)s_1$, $s_1\omega(\psi) \overset{=}{\rightarrow}_\beta s_1'\omega(\varphi)$ and $s' = \sigma(p,q)s_1'$: From the induction hypothesis it follows that $s_1\,\underline{\psi}\hat{}\,X \downarrow_\beta^= s_1'\,\underline{\varphi}\hat{}\,Y$ and therefore $\sigma(p,q)s_1\,\underline{\psi}\hat{}\,X \downarrow_\beta^= \sigma(p,q)s_1'\,\underline{\varphi}\hat{}\,Y$.

(2.5) $s = \delta Z s_1$, $Z \overset{=}{\rightarrow}_\beta Z'$ and $s' = \delta Z' s_1$: From $\underline{\psi}\hat{}\,X \downarrow_\beta^= \underline{\psi}\hat{}\,Y$ it follows that $s_1\,\underline{\psi}\hat{}\,X \downarrow_\beta^= s_1\,\underline{\psi}\hat{}\,Y$ and therefore $\delta Z s_1\,\underline{\psi}\hat{}\,X \downarrow_\beta^= \delta Z' s_1\,\underline{\psi}\hat{}\,Y$.

(2.6) $s = \delta Z s_1$, $s_1\omega(\psi) \overset{=}{\rightarrow}_\beta s_1'\omega(\varphi)$ and $s' = \delta Z s_1'$: From the induction hypothesis it follows that $s_1\,\underline{\psi}\hat{}\,X \downarrow_\beta^= s_1'\,\underline{\varphi}\hat{}\,Y$, and therefore $\delta Z s_1\,\underline{\psi}\hat{}\,X \downarrow_\beta^= \delta Z s_1'\,\underline{\varphi}\hat{}\,Y$.

(2.7) $s = \delta Z\lambda' s_1$, $s\omega(\psi) \overset{=}{\rightarrow}_\beta s_1'\,\delta\,Z^*\lambda'\,\omega(\varphi) = s'\omega(\varphi)$: If $W(s_1\omega(\psi)) \simeq r$ then $Z^* = \underline{\varphi_r Z}$ and $\varphi = \vartheta_r \circ \psi$. Furthermore let R denote the common reduct of X and Y, then we have

the following reduction diagram

$$
\begin{array}{ccc}
\delta\, Z\, \lambda'\, s_1\, \underline{\psi}\hat{}\, X & & s_1'\, \delta\, Z^*\, \lambda'\, \underline{\varphi}\hat{}\, Y \\
\quad \downarrow_{\beta'}^{=} & & \quad \downarrow_{\beta'}^{=} \\
\delta\, Z\, \lambda'\, s_1\, \underline{\psi}\hat{}\, R & \text{- (Theorem 3.1.1) -} & s_1'\, \delta\, Z^*\, \lambda'\, \underline{\varphi}\hat{}\, R \\
\quad \downarrow_{\beta'}^{=} & & \\
\Sigma'(Z, s_1\, \underline{\psi}\hat{}\, R, 1) & & \| \\
\| & & \\
s_1'\, \Sigma'(Z, \underline{\psi}\hat{}\, R, 1 + r) & & \\
\| & & \\
s_1'\, \Sigma'(\underline{\varphi_r}\, Z, \underline{\vartheta_r} \circ \underline{\psi}\hat{}\, R, 1) & \text{(Theorem 2.1)} & \| \\
\| & & \\
s_1'\, \Sigma'(Z^*, \underline{\varphi}\hat{}\, R, 1) & & s_1'\, \Sigma'(Z^*, \underline{\varphi}\hat{}\, R, 1)
\end{array}
$$

Therefore $\delta\, Z\, \lambda'\, s_1\, \underline{\psi}\hat{}\, X \downarrow_{\beta'}^{=} s_1'\, \delta\, Z^*\, \lambda'\, \underline{\varphi}\hat{}\, Y$.

Now that we have established the correctness of the above mentioned claim we proceed by completing the proof of case (iv)(2):

$\Sigma'(u,w,k) =$

$= \Sigma'(u,\sigma(k,m),w_1,k) =$

$= s\, \underline{\psi}\hat{}\, \Sigma'(u,w_1,k+m)$, where $\underline{\varphi_{k-1}}\, u = s\omega(\psi)$.

Furthermore

$\Sigma'(v,w,k) =$

$= \Sigma'(v,\sigma(k,m)w_1,k) =$

$= s'\, \underline{\varphi}\hat{}\, \Sigma'(v,w_1,k+m)$, where $\underline{\varphi_{k-1}}\, v = s'\omega(\varphi)$.

By respectively taking $\Sigma'(u,w_1,k+m)$, $\Sigma'(v,w_1,k+m)$ and $\underline{\varphi_{k-1}}\, u$ for X, Y and $s\omega(\psi)$ in the above mentioned claim, we immediately see that $\Sigma'(u,w,k) \downarrow_{\beta'}^{=} \Sigma'(v,w,k)$.

(3) $n > k$:

$\Sigma'(u,\sigma(n,m)w_1,k) =$

$= \sigma(n-1,m)\,\Sigma'(u,w_1,k+m) \downarrow^{=}_{\beta}$

$\downarrow^{=}_{\beta}\ \sigma(n-1,m)\,\Sigma'(v,w_1,k+m) =$ (induction hypothesis)

$= \Sigma'(v,\sigma(n,m)w_1,k) =$

$= \Sigma'(v,w,k)$.

(v) $w = \delta w_1 w_2$: $\Sigma'(u,w,k) = \delta\,\Sigma'(u,w_1,k)\,\Sigma'(u,w_2,k)$ and from the induction hypothesis it follows that $\Sigma'(u,w_1,k) \downarrow^{=}_{\beta} \Sigma'(v,w_1,k)$ and $\Sigma'(u,w_2,k) \downarrow^{=}_{\beta} \Sigma(v,w_2,k)$. Therefore

$$\delta\,\Sigma'(u,w_1,k)\,\Sigma'(u,w_2,k) \downarrow^{=}_{\beta} \delta\,\Sigma'(v,w_1,k)\,\Sigma'(v,w_2,k) ,$$

hence

$$\Sigma'(u,w,k) \downarrow^{=}_{\beta} \Sigma'(v,w,k) .$$ □

Theorem 3.1.6.

$$WCR(\overset{=}{\rightarrow}_{\beta},) ,$$

i.e. if $u,v,w \in \Lambda'$ and $u \overset{=}{\rightarrow}_{\beta} v$ and $u \overset{=}{\rightarrow}_{\beta} w$ then
$\exists z \in \Lambda' : v \overset{*}{\rightarrow}_{\beta} z \wedge w \overset{*}{\rightarrow}_{\beta} z$.

Proof. By induction on $L(u)$.

(i) $u = \xi(n)$: trivial .

(ii) $u = \omega(\psi)$: trivial .

(iii) $u = \lambda u_1,\ u_1 \overset{=}{\rightarrow}_{\beta} v_1,\ u_1 \overset{=}{\rightarrow}_{\beta} w_1,\ v = \lambda v_1,\ w = \lambda w_1$: this case follows simply from the induction hypothesis.

(iv) $u = \lambda' u_1,\ u_1 \overset{=}{\rightarrow}_{\beta} v_1,\ u_1 \overset{=}{\rightarrow}_{\beta} w_1,\ v = \lambda' v_1,\ w = \lambda' w_1$: as in case (iii).

(v) $u = \sigma(n,m)u_1,\ u_1 \overset{=}{\rightarrow}_{\beta} v_1,\ u_1 \overset{=}{\rightarrow}_{\beta} w_1,\ v = \sigma(n,m)v_1,\ w = \sigma(n,m)w_1$: this case follows simply from the induction hypothesis.

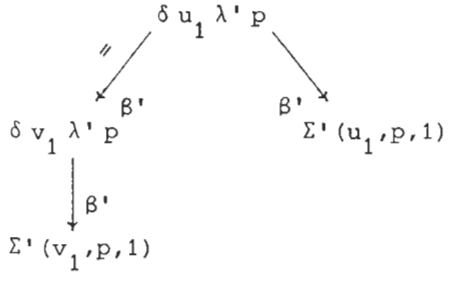
(vi) $u = \delta u_1 u_2$:

(1) $u_1 \overset{=}{\rightarrow}_\beta v_1$, $u_1 \overset{=}{\rightarrow}_\beta v_1'$, $v = \delta v_1 u_2$, $w = \delta v_1' u_2$: this case follows simply from the induction hypothesis.

(2) $u_1 \overset{=}{\rightarrow}_\beta v_1$, $u_2 \overset{=}{\rightarrow}_\beta v_2$, $v = \delta v_1 u_2$, $w = \delta u_1 v_2$: simple, take $\delta v_1 v_2$ for the common reduct.

(3) $u_2 \overset{=}{\rightarrow}_\beta v_2$, $u_2 \overset{=}{\rightarrow}_\beta v_2'$, $v = \delta u_1 v_2$, $w = \delta u_1 v_2'$: this case follows simply from the induction hypothesis.

(4) $u_1 \overset{=}{\rightarrow}_\beta v_1$, $u_2 = \lambda' p$, $v = \delta v_1 u_2$, $w = \Sigma'(u_1,p,1)$: consider the following reduction diagram



,

furthermore, by Theorem 3.1.5, $\Sigma'(u_1,p,1) \overset{=}{\downarrow}_\beta \Sigma'(v_1,p,1)$, and therefore $\delta v_1 \lambda' p \overset{=}{\downarrow}_\beta \Sigma'(u_1,p,1)$.

3.2. The strong normalization property for $\rightarrow_{\beta'}$

In this section we shall offer a proof of the strong normalization property for $\rightarrow_{\beta'}$, ($SN(\rightarrow_{\beta'})$). As mentioned earlier this proof is basically along the lines of the proof given in Barendregt [81] (pp. 283 - 286). $SN(\rightarrow_{\beta'})$ is an important result, since $SN(\rightarrow_{\beta'})$ and $WCR(\rightarrow_{\beta'})$ imply the Church-Rosser property for β-reduction (this is proved in Section 3.3).

The idea of the proof offered in this section is to assign special norms (positive integers) to λ'σ-terms. These norms satisfy the following property: if $u,v \in \Lambda'$ and $u \rightarrow_{\beta'} v$ then for each special norm for u there is a strictly smaller one for v. These special norms are introduced via an auxiliary system λ_0' defined below.

Definition 3.2.1 (λ_0').
The set of *numbered* λ'σ-*terms* λ_0' is the smallest set X satisfying

(i) $\xi^m(n) \in X$, for every $n,m \in \mathbb{N}$;

(ii) $\omega^m(\psi) \in X$, for every segmap ψ and $m \in \mathbb{N}$;

(iii) $u \in X \Rightarrow \lambda u \in X$;

(iv) $u \in X \Rightarrow \sigma^m(p)u \in X$, for every $p \in \mathbb{N} \times \mathbb{M}$ and $m \in \mathbb{N}$;

(v) $u,v \in X \Rightarrow \delta uv \in X$;

(vi) $u,v \in X \Rightarrow \delta\, u\, \lambda_0' v \in X$ and $\delta\, u\, \lambda_1' v \in X$. □

Remarks.

(1) Every numbered λ'σ-term (or *numbered term* for short) u_0 can be seen as a pair (u,I) where u is a λ'σ-term and I is a numbering function which assigns a positive integer to all occurrences of variables and ω's in u_0.

(2) Application of a mapping $\underline{\mu}$, where μ is some refmap, to a numbered term u_0 is defined in the obvious way (numbering of λ'σ-terms has no effect on the application of $\underline{\mu}$ to u_0).

Definition 3.2.2.

The function $|\ |_1 : \lambda_0' \to \lambda_0'$ is inductively defined for numbered terms t by

(i) $|\xi^m(n)|_1 = \xi^m(n)$;

(ii) $|\omega^m(\psi)|_1 = \omega^m(\psi)$;

(iii) $|\lambda u|_1 = \lambda |u|_1$;

(iv) $|\sigma^m(n,r)u|_1 = \sigma^m(n,r)\ \&\ |u|_1$;

(v) $|\delta uv|_1 = \delta |u|_1\, |v|_1$;

(vi) $|\delta\, u\, \lambda_i' v| = \delta\, |u|_1\, \lambda_1'\, |v|_1 \qquad (i = 0,1)$. □

Definition 3.2.3.

The substitution operator in λ_0', denoted by Σ_0', is defined as follows. If $u \in \lambda_0'$ and $k \in \mathbb{N}$ then $\Sigma_0'(u,t,k)$ is inductively defined for numbered terms t by

(i) $\Sigma_0'(u,\xi^m(n),k) = \begin{cases} \underline{\varphi_{k-1}}\, u & , \text{ if } n = k \text{ and } LS(u) = \xi^p(j) \text{ , for some } j,p \in \mathbb{N} \\ \xi^m(n) & , \text{ if } n < k \\ \xi^m(n-1) & , \text{ if } n > k \end{cases}$;

(ii) $\Sigma_0'(u,\omega^m(\psi),k) = \begin{cases} \delta\, \underline{\varphi_{k-1}}\, u\, \lambda_0'\, \omega^m(\vartheta_{k-1} \circ \psi) & , \text{ if } k \in rge(\psi) \\ \omega^m(\vartheta_{k-1} \circ \psi - 1) & , \text{ if } k \notin rge(\psi) \end{cases}$;

(iii) $\Sigma_0'(u,\lambda v,k) = \lambda\, \Sigma_0'(u,v,k+1)$;

(iv) $\Sigma_0'(u,\sigma^m(n,r)v,k) = \begin{cases} s\, \underline{\psi}\hat{}\, \Sigma_0'(u,v,k+r) & , \text{ if } n = k \text{ , } \\ \quad |\underline{\varphi_{k-1}}\, u|_1 = s\omega^{\ell}(\psi) \text{ , (for some } \ell \in \mathbb{N}) \text{ , } \\ \quad W(u) = r \text{ , } rge(\psi) \subseteq \mathbb{N}_r & \\ \sigma^m(n,r)\, \Sigma_0'(u,v,k+r) & , \text{ if } n < k \\ \sigma^m(n-1,r)\, \Sigma_0'(u,v,k+r) & , \text{ if } n > k \end{cases}$;

(v) $\Sigma_0'(u,\delta vw,k) = \delta\, \Sigma_0'(u,v,k)\, \Sigma_0'(u,w,k)$;

(vi) $\Sigma_0'(u,\delta\, v\, \lambda_i'\, w,k) = \delta\, \Sigma_0'(u,v,k)\, \lambda_i'\, \Sigma_0'(u,w,k+1) \qquad (i = 0,1)$. □

The following definition offers a norm for numbered terms.

Definition 3.2.4.

The function $\| \ \| : \lambda_0' \to \mathbb{N}$ is inductively defined for numbered terms t by

(i) $\|\xi^m(n)\| = m$;

(ii) $\|\omega^m(\psi)\| = m$;

(iii) $\|\lambda u\| = \|u\|$;

(iv) $\|\sigma^m(n,r)u\| = m + \|u\|$;

(v) $\|\delta uv\| = \|u\| + \|v\|$;

(vi) $\|\delta\, u\, \lambda_i'\, v\| = \|u\| + \|v\| + i \qquad (i = 0,1)$. □

Definition 3.2.5 ($\rightarrow_{\beta'_0}$).

(1) The binary relation β'_0 on λ'_0 is defined as follows. If $t,u \in \lambda'_0$ then

$$t\,\beta'_0\,u \Leftrightarrow \exists v,w \in \lambda'_0 :$$

$$: t = \delta\, v\, \lambda'_i\, w \wedge u = \Sigma'_0(v,w,1) \wedge (v,w,1) \in \mathrm{dom}(\Sigma'_0) .$$

If $t\,\beta'_0\,u$ then t is called a β'_0-*redex*.

(2) The notion of reduction $\rightarrow_{\beta'_0}$ on λ'_0 is inductively defined by

(i) $u\,\beta'_0\,v \Rightarrow u \rightarrow_{\beta'_0} v$;

(ii) $u \rightarrow_{\beta'_0} v \Rightarrow \lambda u \rightarrow_{\beta'_0} \lambda v$;

(iii) $u \rightarrow_{\beta'_0} v \Rightarrow \lambda'_i u \rightarrow_{\beta'_0} \lambda'_i v$;

(iv) $u \rightarrow_{\beta'_0} v \Rightarrow \sigma^m(n,r)u \rightarrow_{\beta'_0} \sigma^m(n,r)v$;

(v) $u \rightarrow_{\beta'_0} v \Rightarrow \delta uw \rightarrow_{\beta'_0} \delta vw$;

(vi) $u \rightarrow_{\beta'_0} v \Rightarrow \delta wu \rightarrow_{\beta'_0} \delta wv$. □

The permutation condition (PC) is defined for numbered terms in the obvious way (cf. Section 2.6, Definition 2.13). The numbered terms that we take into consideration in this section all satisfy the permutation condition.

Definition 3.2.6 (Λ'_0).

$$\Lambda'_0 = \{t \in \lambda'_0 \mid \mathrm{PC}(t,0)\} .$$ □

Remark.

From Theorem 2.5 (invariance of the permutation condition with respect to β-reduction) it follows immediately that the notion of reduction $\rightarrow_{\beta'_0}$ is a binary relation on Λ'_0; i.e. if $u \in \Lambda'_0$ and $u \rightarrow_{\beta'_0} v$ then $v \in \Lambda'_0$.

In Section 2, Definition 2.7 concerned reference depth values of variables in λσ-terms. This definition is extended to λ'_0 in the obvious way (marking of λ's and numbering of variables and ω's in λσ-terms has no effect on the definition of $D(\eta,t)$, where η is some numbered vari-

able and t is a numbered term). Analogous to Λ we shall often speak in λ'_0 of "an occurrence of a (numbered) variable η with reference depth k ($k \in \mathbb{Z}$) in a (numbered) term t", meaning that $k \in D(\eta,t)$ and - by abuse of language - that η is a specific occurrence of the variable η in t (cf. the remark made on page 44, concerning variables and variable occurrences, and their corresponding reference depth values).

Definition 3.2.7.

Let $u_0 = (u,I) \in \lambda'_0$, where I is a numbering function for u. The property $\downarrow(u,I)$ is defined inductively for terms $u \in \Lambda'$ by

(i) $\downarrow(\xi(n),I)$, if $u_0 = \xi^m(n)$, for some $m \in \mathbb{N}$;

(ii) $\downarrow(\omega(\psi),I)$, if $u_0 = \omega^m(\psi)$, for some $m \in \mathbb{N}$;

(iii) $\downarrow(\lambda v,I)$, if $\downarrow(v,I)$;

(iv) $\downarrow(\sigma(n,r)v,I)$, if

(1) $\downarrow(v,I)$

and

(2.1) $r = 0$ and $(\sigma(n,r)v,I) = \sigma^m(n,r)\ \&\ (v,I)$

or

(2.2) $r > 0$, $(\sigma(n,r)v,I) = \sigma^m(n,r)\ \&\ (v,I)$ and all occurrences of variables η in v with reference depth value p in v ($1 \leq p \leq r$) satisfy $\|\eta\| > m$;

(v) $\downarrow(\delta vw,I)$, if $w \neq \lambda' w_1$ (for any $w_1 \in \Lambda'$) and $\downarrow(v,I)$, $\downarrow(w,I)$;

(vi) $\downarrow(\delta\, v\, \lambda'_i w,I)$, if

(1) $\downarrow(v,I)$ and $\downarrow(w,I)$

and

(2) all occurrences of variables η in w with reference depth value 1 in w satisfy $\|\eta\| > \|v_1\|$, where $v_1 = |v|_1$. □

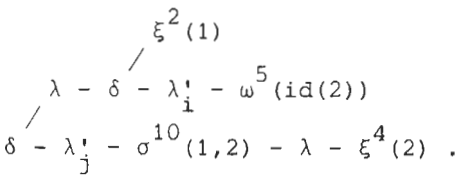
If $\downarrow(u,I)$ then we say that the numbering function I is *decreasing in* u, or: (u,I) *has a decreasing numbering*.

Informally, $\downarrow(u,I)$ implies that all occurrences of variables in (u,I) which are candidates for substitution of some sub-term in (u,I) by contracting a β'_0-redex in (u,I) have a norm larger than the norm of

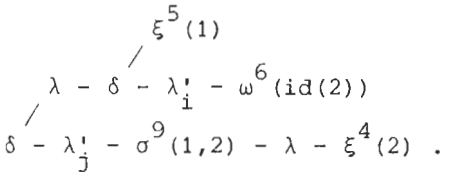
the terms by which they can be replaced.

Example.

Consider the following term written in tree form



This term has a decreasing numbering; this in contrast with the numbered term



Lemma 3.2.1.

For every $u \in \Lambda'$ there is a numbering function I such that $\vdash(u,I)$.

Proof. Number the occurrences of variables or ω's in u from the left to the right, and assign to the n-th occurrence ($n > 0$) the (high) index 2^{m+n-1}, where m is equal to the number of marked λ's to the left of that occurrence.

Example: If u is the $\lambda'\sigma$-term

$$\delta\,\lambda\,\delta\,\xi(1)\,\lambda'_0\,\omega(id(2))\,\lambda'_1\,\sigma(1,2)\,\lambda\,\xi(2)$$

then the result is

$$\delta\,\lambda\,\delta\,\xi^{1}(1)\,\lambda'_0\,\omega^{4}(id(2))\,\lambda'_1\,\sigma^{16}(1,2)\,\lambda\,\xi^{32}(2)\ .$$

Since $2^n > 2^{n-1} + \ldots + 2 + 1$, (u,I) has a decreasing numbering (where I is the numbering function for u as defined above). □

Remark.

The specific numbering function I defined in Lemma 3.2.1 also satisfies:

$$\vdash(u_1,I)\ ,\quad \text{where } u_1 = |u|_1\ .$$

On β_0'-normal forms

We recall that a numbered term t has a β_0'-normal form u if $t \overset{*}{\rightarrow}_{\beta_0'} u$ and $\neg\exists v \in \Lambda_0' : u \rightarrow_{\beta_0'} v$. In Λ_0' we have terms that, strictly speaking, do not have a normal form, but (in some sense) can be considered as terms already in normal form. We shall give some examples to illustrate this situation. First consider the term

$$\delta - \lambda_0' - \omega^{\ell}(\mathrm{id}(1)) \; . \qquad (1)$$

This term β_0'-reduces to itself and to no other term, hence it has no β_0'-normal form. However, β_0'-reduction of (1) involves no actual substitution of the argument of the β_0'-redex contracted. Another example of a term in Λ_0' that β_0'-reduces to itself without involving actual substitutions of arguments of contracted redices is

$$\delta - \lambda_0' - \delta - \lambda_0' - \omega^{\ell}(\mathrm{id}(2)) \; . \qquad (2)$$

By contracting the left-most β_0'-redex in (2) we obtain

$$\delta - \lambda_0' - \delta - \lambda_0' - \omega^{\ell}(\vartheta_1 \circ \mathrm{id}(2)) \; . \qquad (2')$$

By once more contracting the left-most redex in (2') we get

$$\delta - \lambda_0' - \delta - \lambda_0' - \omega^{\ell}(\vartheta_1 \circ \vartheta_1 \circ \mathrm{id}(2)) \qquad (2'')$$

and since $\vartheta_1 \circ \vartheta_1$ is equal to the identity mapping on $\mathbf{N}$ we see that (2") is the same term as (2), i.e. the term (2) β_0'-reduces to itself without having performed actual substitutions of the arguments of the contracted redices. In ordinary type-free λ-calculus we also have the situation that certain terms β-reduce to themselves, e.g. Church's well-known counter example - written in name-free notation - for normalization of this calculus

$$\delta - \lambda \;\frac{\lambda - \delta - \xi(1)}{\xi(1)}\;\frac{\xi(1)}{\delta - \xi(1)} \quad . \qquad (3)$$

There is a large difference, though, between (2) and (3), namely that (3) β-reduces to itself as the result of actual substitution of the argument $\lambda\,\delta\,\xi(1)\,\xi(1)$ for each of the two right-most occurrences of the variable $\xi(1)$ in (3), whereas β_0'-reduction of (2) involves no substitutions at all, the redices just change places via (2') to (2"). So,in a sense, the term (3) is a much more serious counter example for normalization in ordinary type-free λ-calculus than the terms (1) and (2) are in Λ_0', since β_0'-reduction of (1) and (2) just involve a shifting around of redices and no actual substitutions of arguments of contracted redices takes place. In Λ_0' we shall consider contractions of β_0'-redices inside a segment which do not give rise to actual substitutions of their corresponding arguments in that segment as non-essential, since these contractions have the sole effect (apart from updating of reference numbers in variables) that the contracted redices are just re-entered at the back of the segment in question without any substitutions of their respective arguments having taken place. As a consequence we shall consider terms that only give rise to non-essential β_0'-reductions as already being in β_0'-normal form.

We proceed by giving a formal definition of a class of terms in Λ_0' (called Δ_0') which only give rise to non-essential β_0'-reductions as defined above.

Definition 3.2.8 (δ_0').
The sub-set δ_0' of λ_0' is inductively defined as follows. The set δ_0' is

the smallest set X such that

(i) $\omega^m(\psi) \in X$, for every $m \in \mathbb{N}$ and every segmap ψ;

(ii) if $t \in \lambda'_0$, $u = s\omega^m(\psi) \in X$, $W(u) + 1 \in rge(\psi)$ and $\forall \eta \in Var(u) : 1 \notin D(\eta,u)$, then $\delta\, t\, \lambda'_0\, u \in X$.

Remark.
In clause (ii) of Definition 3.2.8 we see that β'_0-contraction of the redex $\delta\, t\, \lambda'_0\, u$ does not result in substitution of the argument t for any variable in u, since there are no occurrences of variables in u with reference depth 1 in u. Furthermore, from $W(u) + 1 \in rge(\psi)$ it follows that the redex-part with argument t reappears at the back of the segment u after contraction of $\delta\, t\, \lambda'_0\, u$.

Definition 3.2.9 (Δ'_0).

$$\Delta'_0 = \{t \in \delta'_0 \mid \exists k \in \mathbb{M} : PC(t,k)\} .$$ □

Lemma 3.2.2.
If $u \in \Delta'_0$ and $u \to_{\beta'_0} v$ then $v \in \Delta'_0$.
Proof. By induction on the generation of $\to_{\beta'_0}$. □

Lemma 3.2.2 motivates the following definition which induces an equivalence relation on Δ'_0.

Definition 3.2.10 ($\sim'_0$).
We inductively define the following binary relation $\sim'_0$ on Δ'_0 as follows

(i) $u \sim'_0 u$;

(ii) $\delta\, u\, \lambda'_0\, v \sim'_0 \Sigma'_0(u,v,1)$;

(iii) $u \sim'_0 v \Rightarrow \delta\, w\, \lambda_f\, u \sim'_0 \delta\, w\, \lambda_f\, v$;

(iv) $u \sim'_0 v$, $v \sim'_0 w \Rightarrow u \sim'_0 w$. □

Theorem 3.2.1.
$\sim'_0$ is an equivalence relation on Δ'_0.
Proof. By induction on the generation of $\sim'_0$ and the corollaries to Theorem 2.3 and Lemma 2.10. □

The following definition extends $\sim_0'$ to Λ_0'.

Definition 3.2.11.

We extend the relation $\sim_0'$ on Δ_0' to Λ_0' as follows. If $t,u,v,w \in \Lambda_0$ then

(i) $t \sim_0' t$;

(ii) $t,u \in \Delta_0'$, $t \sim_0' u$ (in Δ_0') $\Rightarrow t \sim_0' u$ (in Λ_0') ;

(iii) $t \sim_0' u \Rightarrow \lambda t \sim_0' \lambda u$, $\lambda' t \sim_0' \lambda' u$ and $\sigma^{\ell}(p)t \sim_0' \sigma^{\ell}(p)u$;

(iv) $t \sim_0' u$, $v \sim_0' w \Rightarrow \delta tv \sim_0' \delta uw$. □

Remark.

$\sim_0'$ is an equivalence relation on Λ_0'. (This follows easily from the fact that $\sim_0'$ is an equivalence-relation on Δ_0'.)

Definition 3.2.12.

Let $t,u \in \Lambda_0'$.

t *essentially* β_0'*-reduces to* $u \Leftrightarrow t \twoheadrightarrow_{\beta_0'} u \wedge \neg t \sim_{\beta_0'} u$. □

Definition 3.2.13.

Let $t \in \Lambda_0'$. We say that t is *in essential* β_0'*-normal form* if

$$t \twoheadrightarrow_{\beta_0'} u \Rightarrow t \sim_0' u .$$ □

If $t \in \Lambda_0'$ is in essential β_0'-normal form then this means that the only β_0'-reductions that we can perform in t are non-essential β_0'-reductions. We could also say that t is in essential β_0'-normal form if there is no term u such that t essentially β_0'-reduces to u.

Definition 3.2.14.

Let $t \in \Lambda_0'$.

(i) t *essentially* β_0'*-normalizes* (ess β_0'-N(t)) if t has an essential β_0'-normal form ;

(ii) t *essentially* β_0'*-strongly normalizes* (ess β_0'-SN(t)) if there is no β_0'-reduction path starting with t and containing an infinite number of essential β_0'-reductions ;

(iii) $\twoheadrightarrow_{\beta_0'}$ is *essentially normalizing* (ess N($\twoheadrightarrow_{\beta_0'}$)) if

$$\forall t \in \Lambda'_0 : \text{ess } \beta'_0\text{-N}(t) ;$$

(iv) $\to_{\beta'_0}$ is *essentially strongly normalizing* (ess $SN(\to_{\beta'_0})$) if

$$\forall t \in \Lambda'_0 : \text{ess } \beta'_0\text{-SN}(t) . \qquad \square$$

We now proceed by stating some technical lemmas which lead up to the most crucial result of this section, Lemma 3.2.11, which says that norms of numbered terms decrease after essential $\widetilde{\beta}'_0$-reduction.

Lemma 3.2.3.
Let t be an element of λ'_0 and μ some refmap.
If $1 \leq k \in D(\eta,\underline{\mu}t)$ then

$$\exists \eta' \in \text{Var}(t)\ \exists m \in \mathbb{N} : \mu(m) = k \wedge m \in D(\eta',t) \wedge \|\eta\| = \|\eta'\| .$$

Proof. By induction on L(t). $\square$

Lemma 3.2.4.
Let t be an element of λ'_0 and ψ be an element of perm(m), for some $m \in \mathbb{M}$. If $k \in D(\eta,t)$ and $m < k$ then $k \in D(\eta,\underline{\psi}\hat{}t)$.

Proof. By induction on L(t). $\square$

Lemma 3.2.5.
Let $t,u \in \lambda'_0$ and $k,\ell \in \mathbb{N}$. If $k \in D(\eta,\Sigma'_0(t,u,1+\ell))$ and $1 \leq k \leq \ell$ then $\eta \in \text{Var}(u)$ and $k \in D(\eta,u)$.

Proof. By induction on L(u) and Lemmas 3.2.3 and 3.2.4. $\square$

Lemma 3.2.6.
Let $t,u \in \lambda'_0$ and $k \in \mathbb{N}$ and $\ell \in \mathbb{M}$. If $\ell < k$ and $PC(t,\ell)$ holds and all occurrences of variables η in t with reference depth k in t satisfy $\|\eta\| > \|u_1\|$, where $u_1 = |u|_1$, then

(1) $\|\Sigma'_0(u,t,k)\| < \|t\|$, if there is an occurrence of a variable η in t with reference depth k in t ;

(2) $\|\Sigma'_0(u,t,k)\| = \|t\|$, if there are no occurrences of variables η in t with reference depth k in t.

Proof. By induction on L(t). $\square$

Remark.

The results (1) and (2) stated in Lemma 3.2.6 also hold for $S_1 = |\Sigma_0'(u,t,k)|_1$ and $t_1 = |t|_1$; i.e. $\|S_1\| < \|t_1\|$ or $\|S_1\| = \|t_1\|$, depending on whether or not there are occurrences of variables η in t_1 with reference depth k in t_1.

Lemma 3.2.7.

Let $t,u \in \lambda_0'$ and $k \in \mathbb{N}$ and $\ell \in \mathbb{M}$. Furthermore assume that t and u have decreasing numberings and $\ell \leq k$. If $PC(t,\ell)$ holds and all occurrences of variables η in t with reference depth k in t satisfy $\|\eta\| > \|u_1\|$, where $u_1 = |u|_1$, then $\Sigma_0'(u,t,k)$ has a decreasing numbering.

Proof. By induction on L(t).

(i) $t = \xi^m(n)$: If $n \neq k$ then $\downarrow\Sigma_0'(u,t,k)$ holds trivially. If $n = k$ then $\Sigma_0'(u,t,k) = \underline{\varphi_{k-1}}\, u$, and $\underline{\varphi_{k-1}}\, u$ has a decreasing numbering iff u has a decreasing numbering.

(ii) $t = \omega^m(\psi)$:

$$\Sigma_0'(u,t,k) = \begin{cases} \omega^m(\vartheta_{k-1} \circ \psi - 1) & , \text{ if } \ell < k \\ \delta\, \underline{\varphi_{k-1}}\, u\, \lambda_0'\, \omega^m(\vartheta_{k-1} \circ \psi) & , \text{ if } \ell = k \end{cases}.$$

In both cases it is easily seen that $\Sigma_0'(u,t,k)$ has a decreasing numbering.

(iii) $t = \lambda t_1$: This case follows easily from the induction hypothesis.

(iv) $t = \sigma^m(n,p)t_1$:

(1) $n > k$: Then $\Sigma_0'(u,t,k) = \sigma^m(n,p)\, \Sigma_0'(u,t_1,k+p)$ and it follows easily from the induction hypothesis that $\Sigma_0'(u,t_1,k+p)$ has a decreasing numbering. Suppose that $p > 0$ and η is an occurrence of a variable in $\Sigma_0'(u,t_1,k+p)$ with reference depth j $(1 \leq j \leq p)$ in $\Sigma_0'(u,t_1,k+p)$. From Lemma 3.2.5 it follows that η occurs in t_1 at reference depth j in t_1. From $\downarrow\sigma^m(n,p)t_1$ it follows that $\|\eta\| > m$, and hence $\sigma^m(n,p)\, \Sigma_0'(u,t_1,k+p)$ has a decreasing numbering.

(2) $n = k$: Then $\Sigma_0'(u,t,k) = s\ \&\ \underline{\psi}\hat{}\ \Sigma_0'(u,t_1,k+p)$, where $\underline{\varphi_{k-1}}\, u_1 = s\omega(\psi)$. In (1) we have already seen that

$\Sigma_0'(u,t_1,k+p)$ has a decreasing numbering.
There remain the following cases which we have to investigate in order to establish $\downarrow\Sigma_0'(u,t,k)$.

(2.1) $p > 0$ and $\underline{\varphi_{k-1}}\, u_1$ is a term of the form

$$s_1 - \delta \overset{/}{-} {}^{v} - \lambda_1' - s_2 - \omega^q(\psi) \ ,$$

and furthermore there is an occurrence of a variable η in $\underline{\psi}\hat{}\,\Sigma_0'(u,t_1,k+p)$ with reference depth $j+1$ in $\underline{\psi}\hat{}\,\Sigma_0'(u,t_1,k+p)$, where $j = W(s_2\,\omega^q(\psi))$:
In this case we have to show that $\|\eta\| > \|v_1\|$, where $v_1 = |v|_1$. From Lemma 3.2.3 it follows that there is an occurrence of a variable η' in $\Sigma_0'(u,t_1,k+p)$ and an $r \in \mathbb{N}$ such that this occurrence of η' has reference depth r in $\Sigma_0'(u,t_1,k+p)$ and $\psi(r) = 1+j$ and $\|\eta\| = \|\eta'\|$. Since $rge(\psi) \subseteq \mathbb{N}_p$, $1 \le r \le p$ and from Lemma 3.2.5 it follows that $\eta' \in Var(t_1)$ and $r \in D(\eta',t_1)$. From $\downarrow\sigma^m(n,p)t_1$ it follows that this occurrence of η' in t_1 satisfies $\|\eta'\| > m$. Furthermore, $n = k$ and $k \in D(\sigma^m(k,p),\sigma^m(k,p)t_1)$, hence $\|\sigma^m(k,p)\| = m > \|u_1\| = \|\underline{\varphi_{k-1}}\, u_1\|$.
Conclusion: $\|\eta\| = \|\eta'\| > m > \|\underline{\varphi_{k-1}}\, u_1\| > \|v_1\|$.

(2.2) $p > 0$ and $\underline{\varphi_{k-1}}\, u_1$ is a term of the form

$$s_1 - \sigma^i(h,r) - s_2 - \omega^q(\psi) \ ,$$

and furthermore there is an occurrence of a variable η in $\underline{\psi}\hat{}\,\Sigma_0'(u,t_1,k+p)$ with reference depth $j+a$ in $\underline{\psi}\hat{}\,\Sigma_0'(u,t_1,k+p)$, where $j = W(s_2\,\omega^q(\psi))$ and $1 \le a \le r$:
In this case we have to prove that $\|\eta\| > i$. The proof of this case is an exact analogue of the proof given in case (2.1) above.

(v) $t = \delta t_1 t_2$ and $FS(t_2) \neq \lambda_i'$: This case is simply proven by applying the induction hypothesis.

(vi) $t = \delta\, t_1\, \lambda_i'\, t_2$: Then $\Sigma_0'(u,t,k) = \delta\, \Sigma_0'(u,t_1,k)\, \lambda_i'\, \Sigma_0'(u,t_2,k+1)$,

and $\Sigma_0'(u,t,k)$ has a decreasing numbering iff

(1) $\downarrow\Sigma_0'(u,t_1,k)$ and $\downarrow\Sigma_0'(u,t_2,k+1)$;

(2) If η is an occurrence of a variable in $\Sigma_0'(u,t_2,k+1)$ with reference depth 1 in $\Sigma_0'(u,t_2,k+1)$ then $\|\eta\| > \|S_1\|$, where $S_1 = |\Sigma_0'(u,t_1,k)|_1$.

From $\downarrow t$ it follows that $\downarrow t_1$ and $\downarrow t_2$. Furthermore, $PC(t,\ell)$ holds iff $PC(t_1,0)$ and $PC(t_2,\ell+1)$ hold. By applying the induction hypothesis it is easily seen that $\downarrow\Sigma_0'(u,t_1,k)$ and $\downarrow\Sigma_0'(u,t_2,k+1)$ hold.

If $1 \in D(\eta,\Sigma_0'(u,t_2,k+1))$ then by Lemma 3.2.5 $\eta \in Var(t_2)$ and $1 \in D(\eta,t_2)$. From $\downarrow t$ it follows that such an occurrence of η in t_2 satisfies $\|\eta\| > \|t_1^*\|$, where $t_1^* = |t_1|_1$. Furthermore from Lemma 3.2.6 it follows that $\|t_1^*\| \geq \|S_1\|$, hence $\|\eta\| > \|S_1\|$. □

Corollary.

(1) If $\delta\, u\, \lambda_i'\, t \in \Lambda_0'$ and $\downarrow\delta\, u\, \lambda_i'\, t$ then $\downarrow\Sigma_0'(u,t,1)$;

(2) If $u \in \Lambda_0'$ and $u \to_{\beta_0'} v$ and $\downarrow u$ then $\downarrow v$.

Lemma 3.2.8.

Let $k \in \mathbb{N}$. If $v = \delta\, \underline{\varphi_{k-1}}\, u\, \lambda_0'\, \underline{\vartheta_{k-1}}\, t \in \delta_0'$ and $PC(v,\ell)$ holds for some $\ell \in \mathbb{N}$ then $\|\Sigma_0'(u,t,k)\| = \|v\|$.

Proof. By induction on $L(t)$.

(i) $t = \omega^m(\psi)$: Since $v \in \delta_0'$ it follows that $1 \in rge(\vartheta_{k-1} \circ \psi)$, hence $k \in rge(\psi)$ and $\Sigma_0'(u,t,k) = \delta\, \underline{\varphi_{k-1}}\, u\, \lambda_0'\, \underline{\vartheta_{k-1}}\, \omega^m(\psi) = v$.

(ii) $t = \delta\, w\, \lambda_0'\, s\, \omega^m(\psi)$:

$$\Sigma_0'(u,t,k) =$$

$$= \delta\, \Sigma_0'(u,w,k)\, \lambda_0'\, \Sigma_0'(u,s\omega^m(\psi),k+1) .$$

Furthermore $v \in \delta_0'$, hence

$$\forall \eta \in Var(t) : k \notin D(\eta,t) .$$

Therefore there are no occurrences of variables in w with reference depth k in w, and from $PC(v,\ell)$ it follows that $PC(\underline{\vartheta_{k-1}}\, w,0)$.

From $PC(\underline{\vartheta_{k-1}}\, w,0)$ and Lemma 2.7 it follows that PC(w,0) and therefore

$$\| \Sigma_0'(u,w,k) \| = \| w \| \quad , \quad \text{by Lemma 3.2.6 .} \qquad (*)$$

If $\delta\, \underline{\varphi_{k-1}}\, u\, \lambda_0'\, \underline{\vartheta_{k-1}}\, t \in \delta_0'$ then also $\delta\, \underline{\varphi_k}\, \lambda_0'\, \underline{\vartheta_k}\, s\omega^m(\psi) \in \delta_0'$. By the induction hypothesis and (*) it follows that

$$\| \delta\, \Sigma_0'(u,w,k)\, \lambda_0'\, \Sigma_0'(u,s\omega^m(\psi),k+1) \| =$$

$$= \| \Sigma_0'(u,w,k) \| + \| \Sigma_0'(u,s\omega^m(\psi),k+1) \| =$$

$$= \| w \| + \| \delta\, \underline{\varphi_k}\, u\, \lambda_0'\, \underline{\vartheta_k}\, s\omega^m(\psi) \| =$$

$$= \| w \| + \| \underline{\varphi_k}\, u \| + \| \underline{\vartheta_k}\, s\omega^m(\psi) \| =$$

$$= \| \underline{\vartheta_{k-1}}\, w \| + \| \underline{\varphi_{k-1}}\, u \| + \| \underline{\vartheta_{k-1}^{<1>}}\, s\omega^m(\psi) \| =$$

$$= \| \underline{\varphi_{k-1}}\, u \| + \| \underline{\vartheta_{k-1}}\, t \| =$$

$$= \| v \| \quad . \qquad \square$$

Corollary.

(1) If $v = \delta\, u\, \lambda_0'\, t \in \Delta_0'$ then $\| \Sigma_0'(u,t,1) \| = \| v \|$;

(2) if $u,v \in \Lambda_0'$ and $u \sim_0' v$ then $\| u \| = \| v \|$.

Definition 3.2.15.

Let $u \in \lambda_0'$ and let SUB(u) denote the set of sub-terms of u. We define the sub-set A of λ_0' as follows

$$A = \{ u \in \lambda_0' \mid \forall t \in SUB(u) : t = \delta\, v\, \lambda_0'\, w \Rightarrow t \in \delta_0' \} \quad . \qquad \square$$

Lemma 3.2.9.

If $u = \delta\, w\, \lambda_i'\, z \in A$ then $v = \Sigma_0'(w,z,1) \in A \quad (i = 0,1)$.

Proof. By induction on L(z).

(i) $z = \xi^m(n)$:

(1) $n = 1$: Then $v = w$, and since $w \in SUB(u)$ and $u \in A$ it follows that $v \in A$;

(2) $n > 1$: Then $v = \xi^m(n-1) \in A$.

(ii) $z = \omega^m(\psi)$: Then

$$v = \begin{cases} \delta\, w\, \lambda_0'\, \omega^m(\psi) , & \text{if } 1 \in rge(\psi) \\ \omega^m(\psi) , & \text{if } 1 \notin rge(\psi) \end{cases}$$

and in both cases $v \in A$.

(iii) $z = \lambda z_1$: This case follows simply from the induction hypothesis and Theorem 2.1.

(iv) $z = \sigma^m(n,r) z_1$:

(1) $n = 1$: Then $v = s\,\underline{\psi}\hat{}\,\Sigma_0'(w,z_1,1+r) = s\,\underline{\psi}\hat{}\,\Sigma_0'(\varphi_{\underline{r}}\, w, \vartheta_{\underline{r}}\, z_1, 1)$, where $|w|_1 = s\omega^{\ell}(\psi)$. If $t \in SUB(v)$ and $t = \delta\, t_1\, \lambda_0'\, t_2$, for some $t_1, t_2 \in \lambda_0'$, then $t \in SUB(w)$ or $t \in SUB(\Sigma_0'(\varphi_{\underline{r}}\, w, \vartheta_{\underline{r}}\, z_1, 1))$. If $t \in SUB(w)$ then $t \in SUB(u)$ and since $u \in A$ we have $t \in \delta_0'$. By applying the induction hypothesis to $\delta\, \varphi_{\underline{r}}\, w\, \lambda_i'\, \vartheta_{\underline{r}}\, z$ ($\in A$, for $i = 0,1$) we see that if $t \in SUB(\Sigma_0'(\varphi_{\underline{r}}\, w, \vartheta_{\underline{r}}\, z_1, 1))$ then $t \in \delta_0'$. Hence, $v \in A$.

(2) $n > 1$: Then $v = \sigma^m(n-1,r)\, \Sigma_0'(\varphi_{\underline{r}}\, w, \vartheta_{\underline{r}}\, z_1, 1)$ and the result follows immediately from the induction hypothesis.

(v) $z = \delta z_1 z_2$: Then $v = \delta\, \Sigma_0'(w,z_1,1)\, \Sigma_0'(w,z_2,1)$ and the result follows immediately from the induction hypothesis.

(vi) $z = \delta\, z_1\, \lambda_0'\, z_2$: Then $v = \delta\, \Sigma_0'(w,z_1,1)\, \lambda_0'\, \Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1)$. By applying the induction hypothesis we see that $\Sigma_0'(w,z_1,1)$, $\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1) \in A$. Furthermore from $u \in A$ it follows that $z \in \delta_0'$ and, hence, $z_2 \in \delta_0'$. If $z_2 \in \delta_0'$ then $\vartheta_{\underline{1}}\, z_2 \in \delta_0'$ and also $\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1) \in \delta_0'$ (it is easily proved that $p \in \delta_0'$ implies $\Sigma_0'(q,p,1) \in \delta_0'$, for all $q \in \lambda_0'$). In order to prove that $v \in A$ we have to show that $v \in \delta_0'$, and since $\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1) \in \delta_0'$ it suffices to show that

(1) there are no occurrences of variables in $\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1)$ with reference depth 1 in $\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1)$;

(2) if $LS(\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1) = \omega^{\ell}(\psi)$ then $1 + W(\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1)) \in rge(\psi)$.

From Lemma 3.2.5 it follows that $1 \in D(\eta, \Sigma_0'(w,z_2,2))$ implies $1 \in D(\eta,z_2)$, hence yielding a contradiction with $z \in \delta_0'$. Furthermore from $z \in \delta_0'$ it follows that $LS(z_2) = \omega^{\ell}(\varphi)$ and

$1 + r \in rge(\varphi)$, where $W(z_2) = r$, for some segmap φ. If $2 + r \in rge(\varphi)$ then $\psi = \vartheta_{r+1} \circ \varphi$ and $1 + W(\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1)) = 2 + r \in rge(\psi)$. If $2 + r \notin rge(\varphi)$ then $\psi = \varphi$ and $1 + W(\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1)) = 1 + r \in rge(\psi)$. Hence, $v \in A$.

(vii) $z = \delta\, z_1\, \lambda_1'\, z_2$: Then $v = \delta\, \Sigma_0'(w, z_1, 1)\, \lambda_1'\, \Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1)$. By applying the induction hypothesis we see that $\Sigma_0'(w, z_1, 1)$, $\Sigma_0'(\varphi_{\underline{1}}\, w, \vartheta_{\underline{1}}\, z_2, 1) \in A$. Hence, $v \in A$. □

Theorem 3.2.2.

If $u \in A$ and $u \to_{\beta_0'} v$ then $v \in A$.

Proof. By induction on the generation of $\to_{\beta_0'}$ and Lemma 3.2.9. □

Corollary.

If $u \in \lambda_0'$ and all marked λ's in u are indexed with the number 1 then $v \in A$ for all terms v such that $u \overset{*}{\to}_{\beta_0'} v$.

Lemma 3.2.10.

$$v = \delta\, u\, \lambda_0'\, t \in A \cap \Lambda_0' \Rightarrow \|v\| = \|\Sigma_0'(u, t, 1)\| .$$

Proof. The result follows from Lemma 3.2.8. □

Lemma 3.2.11.

Let $v = \delta\, \varphi_{\underline{k-1}}\, u\, \lambda_1'\, \vartheta_{\underline{k-1}}\, t \in A$. If $PC(t,k)$ and $\nmid v$ then $\|v\| > \|\Sigma_0'(u, t, k)\|$.

Proof. By induction on $L(t)$.

(i) $t = \xi^m(n)$:

(1) $n < k$: Then $\|\Sigma_0'(u,t,k)\| = \|\xi^m(n)\| = \|t\| = \|\vartheta_{\underline{k-1}}\, t\| < \|v\|$;

(2) $n = k$: Then $\|\Sigma_0'(u,t,k)\| = \|\varphi_{\underline{k-1}}\, u\| < \|v\|$;

(3) $n > k$: Then $\|\Sigma_0'(u,t,k)\| = \|\xi^m(n-1)\| = \|t\| =$

$$= \|\vartheta_{\underline{k-1}}\, t\| < \|v\| .$$

(ii) $t = \omega^m(\psi)$: From $PC(t,k)$ it follows that $\|\Sigma_0'(u,t,k)\| = \|\delta\, \varphi_{\underline{k-1}}\, u\, \lambda_0'\, \vartheta_{\underline{k-1}}\, \omega^m(\psi)\| < \|v\|$.

(iii) $t = \lambda z$: This case follows simply from the induction hypothesis.

(iv) $t = \sigma^m(n,r)z$:

(1) $n < k$: Then $\Sigma_0'(u,t,k) = \sigma^m(n,r)\,\Sigma_0'(u,z,k+r)$ and the result follows by applying the induction hypothesis.

(2) $n = k$: Then $\Sigma_0'(u,t,k) = s\underline{\psi}\hat{}\,\Sigma_0'(u,z,k+r)$, where $s\omega^{\ell}(\psi) = |\underline{\varphi_{k-1}}\,u|_1$.

$\|\Sigma_0'(u,t,k)\| =$

$= \|s\omega^{\ell}(\psi)\| + \|\Sigma_0'(u,z,k+r)\| \leq$

$\leq m + \|\Sigma_0'(u,z,k+r)\| <$ ($\downarrow v$)

$< m + \|\delta\,\underline{\varphi_{k+r-1}}\,u\,\lambda_1'\,\underline{\vartheta_{k+r-1}}\,z\| =$ (induction hypothesis)

$= m + \|\underline{\varphi_{k+r-1}}\,u\| + \|\underline{\vartheta_{k+r-1}}\,z\| + 1 =$

$= m + \|u\| + \|z\| + 1 =$

$= \|v\|$.

(3) $n > k$: Then $\Sigma_0'(u,t,k) = \sigma^m(n-1,r)\,\Sigma_0'(u,z,k+r)$ and the result follows by applying the induction hypothesis.

(v) $t = \delta z_1 z_2$: Then $\Sigma_0'(u,t,k) = \delta\,\Sigma_0'(u,z_1,k)\,\Sigma_0'(u,z_2,k)$. Furthermore

$\|\delta\,\Sigma_0'(u,z_1,k)\,\Sigma_0'(u,z_2,k)\| =$

$= \|\Sigma_0'(u,z_1,k)\| + \|\Sigma_0'(u,z_2,k)\| \leq$

$\leq \|\underline{\vartheta_{k-1}}\,z_1\| + \|\Sigma_0'(u,z_2,k)\| <$ (PC($\underline{\vartheta_{k-1}}\,z_1$,0) and Lemma 3.2.6)

$< \|\underline{\vartheta_{k-1}}\,z_1\| + \|\underline{\varphi_{k-1}}\,u\| + \|\underline{\vartheta_{k-1}}\,z_2\| + 1 =$ (induction hypothesis)

$= \|v\|$.

(vi) $t = \delta z_1 \lambda_i' z_2$: See case (v). □

Corollary.

(1) If $\delta u \lambda_1' t \in A \cap \Lambda_0'$ and $\downarrow\delta u \lambda_1' t$ then $\|\delta u \lambda_1' t\| > \|\Sigma_0'(u,t,1)\|$;

(2) If $u,v \in A \cap \Lambda_0'$, $\downarrow u$ and u essentially β_0'-reduces to v then $\|u\| > \|v\|$.

Theorem 3.2.3.

$\rightarrow_{\beta'_0}$ is essentially strongly normalizing on $\{t \in A \cap \Lambda'_0 \mid \downarrow t\}$.

Proof. The result follows from the corollaries to Lemmas 3.2.7, 3.2.8, 3.2.11 and Lemma 3.2.10. □

3.3. The Church-Rosser property for $\rightarrow_\beta$

In this section we offer a proof of the Church-Rosser property for $\rightarrow_\beta$ on $\{t \in \Lambda \mid PC(t,0)\}$ by using the results given in Sections 3.1 and 3.2. This is done as follows. By dropping the numberings from the definitions of δ'_0, Δ'_0, $\sim'_0$ and Λ'_0 we obtain the definitions of δ', Δ', $\sim'$ and Λ'_*, where $\Lambda'_* = \{t \in \Lambda' \mid PC(t,0)\}$. Furthermore, the notion of essential β'-reduction on Λ'_* is defined as follows.

Definition 3.3.1.

Let $t,u \in \Lambda'_*$.

t *essentially* β'-*reduces* to $u \Leftrightarrow t \rightarrow_{\beta'} u \wedge \neg(t \sim' u)$. □

From Definition 3.3.1 we get the obvious definition of $\rightarrow_{\beta'}$ being essentially strongly normalizing (ess $SN(\rightarrow_{\beta'})$). Since adequate numbering of terms is always possible the following result follows immediately from Theorem 3.2.3.

Theorem 3.3.1.

$\rightarrow_{\beta'}$ is essentially strongly normalizing on Λ'_*. □

Since the permutation condition is invariant with respect to β'-reduction (cf. Theorem 2.5) the following result follows immediately from Theorem 3.1.6.

Theorem 3.3.2.

$\rightarrow_{\beta'}$ is weakly Church-Rosser on Λ'_*. □

Theorem 3.3.3.

For every $t \in \Lambda'_*$ there exists an $u \in \Lambda'_*$ such that

(1) u is an essential β'-normal form of t;

(2) if v is an essential β'-normal form of t, then $u \sim' v$.

Proof. The following proof is along the lines of the proof given for Theorem 1.1.3. By Theorem 3.3.1 each term $t \in \Lambda'_*$ has an essential β'-normal form u. Furthermore, call a term $t \in \Lambda'_*$ *ambiguous* if t β'-reduces to two essential β'-normal forms u_1, u_2 such that $\neg(u_1 \sim' u_2)$. If t is ambiguous then there exists a term u such that $t \overset{*}{\to}_{\beta'} u$ and u is ambiguous and $\neg(t \sim' u)$, which we now show. The following two figures suggest how t can reduce to u_1 and u_2.

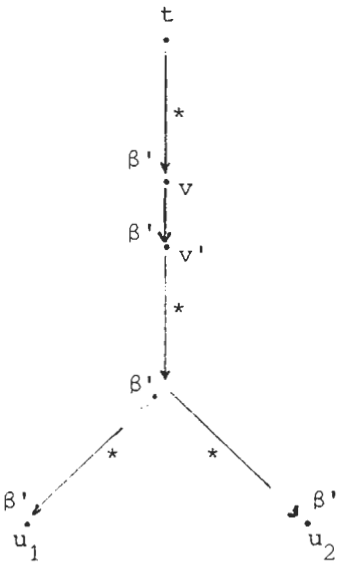


FIG. 3.3.1.

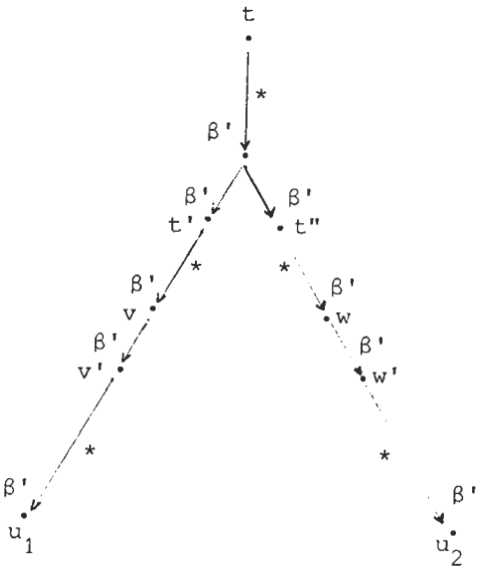


FIG. 3.3.2.

where $v \to_{\beta'} v'$ and $w \to_{\beta'} w'$ denote the first *essential* β'-reductions occurring on the reduction paths starting from t and ending in u_1, u_2. In the case of Figure 3.3.1 it is immediately clear that v' is ambiguous. In the case of Figure 3.3.2 it follows from $WCR(\overset{=}{\to}_{\beta'})$ that t' and t" have a common β'-reduct t'" and, by ess $SN(\to_{\beta'})$, t'" has an essential β'-normal form u_3 as indicated in the figure below.

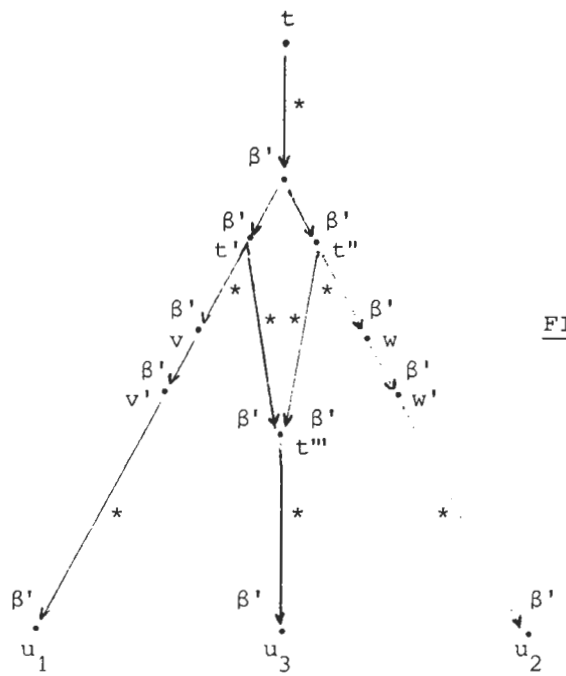


FIG. 3.3.3.

Since $v \overset{*}{\to}_{\beta'} v'$ was the first essential β'-reduction on the reduction path from t to u_1, it follows that $t' \sim' v$ and, by symmetry of $\sim'$, $v \sim' t'$. Hence, $v \overset{*}{\to}_{\beta'} t'$. From $v \overset{*}{\to}_{\beta'} t'$ and $t' \overset{*}{\to}_{\beta'} t_3$ it follows that $v \overset{*}{\to}_{\beta'} t_3$. Analogously, $w \overset{*}{\to}_{\beta'} u_3$. Furthermore, from $\neg(u_1 \sim' u_2)$ it follows that either $\neg(u_3 \sim' u_1)$ or $\neg(u_3 \sim' u_2)$. If $\neg(u_3 \sim' u_1)$ then we can take v' for the ambiguous term u, and if $\neg(u_3 \sim' u_2)$ then we can take w' for u.

Now that we have established that all ambiguous terms essentially β'-reduce to another ambiguous term we have obtained a contradiction with ess $SN(\to_{\beta'})$, hence ambiguous terms do not exist and the result follows. □

Definition 3.3.2.
Let $t,u \in \Lambda'_*$. The relation $>'$ on Λ'_* is defined as follows

$t >' u$ iff u is an essential β'-normal form of t and
if v is an essential β'-normal form of t then $v \sim' u$. □

Definition 3.3.3.
The function $|\ |\ : \Lambda' \to \Lambda$ is inductively defined for marked terms t as follows

(i) $|\xi(n)| = \xi(n)$;

(ii) $|\omega(\psi)| = \omega(\psi)$;

(iii) $|\lambda u| = \lambda|u|$;

(iv) $|\sigma(n,m)u| = \sigma(n,m)|u|$;

(v) $|\delta uv| = \delta|u||v|$;

(vi) $|\delta\, u\, \lambda' v| = \delta|u|\,\lambda|v|$ □

Definition 3.3.4.

$$\Lambda_* = \{t \in \Lambda \mid PC(t,0)\} \ .$$ □

Definition 3.3.5.

Let $t,u \in \Lambda_*$. The relation $>$ on Λ_* is defined as follows.

$$t > u \Leftrightarrow \exists t' \in \Lambda'_* : t = |t'| \wedge t' >' u' \wedge u = |u'| \ .$$ □

Definition 3.3.6 ($\rightarrow_\beta$).

(1) The binary relation β on Λ is defined as follows.

If $t,u \in \Lambda$ then

$$t \beta u \Leftrightarrow \exists v,w \in \Lambda : t = \delta v \lambda w \wedge u = \Sigma(v,w,1) \wedge (v,w,1) \in dom(\Sigma) \ .$$

If $t \beta u$ then t is called a β-*redex*.

(2) The notion of reduction $\rightarrow_\beta$ on Λ is inductively defined by

(i) $u \beta v \Rightarrow u \rightarrow_\beta v$;

(ii) $u \rightarrow_\beta v \Rightarrow \lambda u \rightarrow_\beta \lambda v$;

(iii) $u \rightarrow_\beta v \Rightarrow \sigma(n,m)u \rightarrow_\beta \sigma(n,m)v$;

(iv) $u \rightarrow_\beta v \Rightarrow \delta uw \rightarrow_\beta \delta vw$;

(v) $u \rightarrow_\beta v \Rightarrow \delta wu \rightarrow_\beta \delta wu$.

Theorem 3.3.4.

$\overset{*}{\rightarrow}_\beta$ is the transitive and reflexive closure $\overset{*}{>}$ of $>$ on Λ_*.

Proof. From Theorem 2.5 it follows that the permutation condition is invariant with respect to β-reduction; hence, if $u \in \Lambda_*$ and $u \rightarrow_\beta v$ then $v \in \Lambda_*$. Furthermore, from

$$\rightarrow_\beta \subseteq > \subseteq \overset{*}{\rightarrow}_\beta$$

it follows that

$$\overset{*}{\rightarrow}_\beta \subseteq \overset{*}{>} \subseteq \overset{*}{\rightarrow}_\beta$$

hence $\overset{*}{\rightarrow}_\beta = \overset{*}{>}$. □

Theorem 3.3.5.

> satisfies the diamond property.

Proof. Assume that t > u and t > v. From the definition of > it follows that there exist terms t',u',v' $\in \Lambda'_*$ such that t = |t'|, u = |u'|, v = |v'| and

(1) u', v' are essential β'-normal forms of t' ;

(2) u' ~ v' .

From (1) and (2) it follows that u' >' v' and v' > v'. Therefore u > v and v > v, and the result follows. □

Theorem 3.3.6.

$\overset{=}{\rightarrow}_\beta$ is Church-Rosser on Λ_*.

Proof. The result follows from Theorems 3.3.4 and 3.3.5. □

4. THE CLOSURE PROPERTY FOR THE TYPED SYSTEM $\lambda_T\sigma$

In Section 1.2 we introduced the typed system $\lambda_T\sigma$. The definition of $\lambda_T\sigma$ in Section 1.2 was completely formal and therefore when we speak of $\lambda_T\sigma$ we refer to the system $\lambda_T\sigma$ as defined in Section 1.2, Definition 1.2.5. We recall that we have the following relevant sets regarding types.

(1) The set of types T : we have γ-types, ρ-types and the incorrect type $\otimes$;

(2) The set of quasi-types T_π : elements of T_π are not types of terms in $\lambda_T\sigma$, but serve as intermediate constructs for evaluating the product $(*)$ of a number of types in order to calculate the eventual type of a $\lambda_T\sigma$-term, which is either the incorrect type $\otimes$, a γ-type or a ρ-type (but never a π-type).

The objective of this section is to show that the type of a correct $\lambda_T\sigma$-term t, with respect to a certain type context τ, and the type of its β-reduct (with respect to the same context) are the same, provided that t satisfies the permutation condition (the *closure property* for $\lambda_T\sigma$).

We note that basic operations on terms introduced in Section 2 for the type-free system Λ are extended to $\lambda_T\sigma$ in the obvious way; in particular $\underline{\mu}t$, W(t) and PC(t,k) are defined for typed terms t as in Section 2 (a typed lambda is treated in the same manner as a non-typed lambda). Furthermore, a substitution operator in $\lambda_T\sigma$ is denoted by Σ_f, for some type $f \in T\backslash\{\otimes\}$. The type f attached to a substitution operator Σ_f is the same type as attached to the lambda of the redex, say $\delta\, u\, \lambda_f\, v$, that - after contraction of $\delta\, u\, \lambda_f\, v$ - gave rise to the invokement of the substitution $\Sigma_f(u,v,1)$. We proceed by defining substitution in $\lambda_T\sigma$.

Definition 4.1 (substitution).
Let $f \in T\backslash\{\otimes\}$. If $u \in \lambda_T\sigma$ and $k \in \mathbb{N}$ then $\Sigma_f(u,t,k)$ is inductively defined for typed terms t by

$$\text{(i)}\quad \Sigma_f(u,\xi(n),k) = \begin{cases} \underline{\varphi_{k-1}}\, u & , \text{ if } n = k \text{ and } LS(u) \in rge(\xi) \\ \xi(n) & , \text{ if } n < k \\ \xi(n-1) & , \text{ if } n > k \end{cases} ;$$

(ii) $\Sigma_f(u,\omega(\psi),k) = \begin{cases} \delta\, \underline{\varphi_{k-1}}\, u\, \lambda_f\, \omega(\vartheta_{k-1} \circ \psi) , & \text{if } k \in \mathrm{rge}(\psi) \\ \omega(\vartheta_{k-1} \circ \psi - 1) , & \text{if } k \notin \mathrm{rge}(\psi) \end{cases}$;

(iii) $\Sigma_f(u,\lambda_g v,k) = \lambda_g\, \Sigma_f(u,v,k+1)$;

(iv) $\Sigma_f(u,\sigma(n,m)v,k) = \begin{cases} s\, \underline{\psi}\hat{}\, \Sigma_f(u,v,k+m) , & \text{if } n = k , \\ \quad \underline{\varphi_{k-1}}\, u = s\omega(\psi) , \ W(u) = m \text{ and } \mathrm{rge}(\psi) \subseteq \mathbb{N}_m \\ \sigma(n,m)\, \Sigma_f(u,v,k+m) , & \text{if } n < k \\ \sigma(n-1,m)\, \Sigma_f(u,v,k+m) , & \text{if } n > k \end{cases}$;

(v) $\Sigma_f(u,\delta vw,k) = \delta\, \Sigma_f(u,v,k)\, \Sigma_f(u,w,k)$. □

We note that the results concerning applications of refmaps to terms and the results concerning substitution in Section 2 hold equally in the typed system $\lambda_T\sigma$, since the typing of terms in $\lambda_T\sigma$ is completely irrelevant as far as establishing these results is concerned. The same holds for the results concerning the permutation condition in Section 2.3. We shall therefore make frequent use of these results simply by referring to the corresponding type-free results in Section 2.

Definition 4.2 ($\to_\beta$).

(1) The binary relation β on $\lambda_T\sigma$ is defined as follows.
If $t,u \in \lambda_T\sigma$ then

$$t\,\beta\,u \Leftrightarrow \exists v,w \in \lambda_T\sigma :$$

$$: t = \delta\, v\, \lambda_f\, w \wedge u = \Sigma_f(v,w,1) \wedge (v,w,1) \in \mathrm{dom}(\Sigma_f) .$$

If $t\,\beta\,u$ then t is called a β-*redex*.

(2) The notion of reduction $\to_\beta$ on $\lambda_T\sigma$ is inductively defined by

(i) $u\,\beta\,v \Rightarrow u \to_\beta v$;
(ii) $u \to_\beta v \Rightarrow \lambda_f u \to_\beta \lambda_f v$, for every $f \in T\backslash\{\otimes\}$;
(iii) $u \to_\beta v \Rightarrow \sigma(p)u \to_\beta \sigma(p)v$, for every $p \in \mathbb{N} \times \mathbb{M}$;
(iv) $u \to_\beta v \Rightarrow \delta uw \to_\beta \delta vw$;
(v) $u \to_\beta v \Rightarrow \delta wu \to_\beta \delta wv$. □

In order to facilitate the evaluation of the product of quasi-types and types in $\lambda_T\sigma$ we give the following definition of the product of two quasi-types ensuring that this extended version of the $*$-operation is associative; i.e. $f * (g * h) = (f * g) * h$, for all quasi-types f,g and types h.

Definition 4.3.
Let F, G, H, I and J be elements of $(T\backslash\{\otimes\})^*$. The product of two quasi-types is defined as follows

$$\pi(F,G) * \pi(H,I) = \begin{cases} \pi(F \,\&\, J, I) , & \text{if } H = \bar{G} \,\&\, J \\ \pi(F, J \,\&\, I) , & \text{if } G = J \,\&\, \bar{H} \\ \otimes , & \text{otherwise} \end{cases} .$$

□

Lemma 4.1.
The $*$-operation is associative; i.e. $f * (g * h) = (f * g) * h$, for all quasi-types f,g and types h.

Proof. The result follows from Definitions 1.2.5 and 4.3 by simple computation. □

Definition 4.4.
We define the following sub-sets of $T\backslash\{\otimes\}$

$$\Gamma = \{f \in T\backslash\{\otimes\} \mid FS(f) = \gamma\} ;$$

$$P = \{f \in T\backslash\{\otimes\} \mid FS(f) = \rho\} .$$

□

Lemma 4.2.
Let $t \in \lambda_T\sigma$ and τ be a type context. If $typ(t,\tau) \neq \otimes$ then

(1) $typ(t,\tau) \in \Gamma \Leftrightarrow LS(t) \in rge(\xi)$

(2) $typ(t,\tau) \in P \Leftrightarrow LS(t) \in rge(\omega)$.

Proof. By induction on L(t). □

Lemma 4.3.
Let C be a non-empty set and let μ be a refmap.
If $f,g \in C^*$ and $L(f) = m$ then $(f \,\&\, g) \circ \mu^{<m>} = f \,\&\, (g \circ \mu)$.

Proof. Simple computation. □

Lemma 4.4.

Let $t \in \lambda_T\sigma$, τ be a type context and μ be a refmap.
If $t \circ \mu \in (T\backslash\{\otimes\})^*$ then $typ(\underline{\mu}t,\tau) = typ(t,\tau \circ \mu)$.

Proof. By induction on $L(t)$ and Lemma 4.3. □

Lemma 4.5.

Let $t = s\omega(\psi)$ and $s \,\&\, u$ be elements of $\lambda_T\sigma$ and let τ be a type context. If $typ(t,\tau) \neq \otimes$ and $W(t) = m$ then

$$\exists F_1,F_2,F_3 \in (T\backslash\{\otimes\})^* :$$

$$: L(F_3) = m \wedge typ(s \,\&\, u,\tau) = \pi(F_1,F_2) * typ(u,F_3 \,\&\, \tau) .$$

Proof. By induction on $L(t)$. □

Lemma 4.6.

Let $t = s\omega(\psi) \in \lambda_T\sigma$ and τ be a type context. If $typ(t,\tau) \neq \otimes$, $W(t) = m$ and $PC(t,0)$ then $typ(t,\tau) = \rho(G_1,G_2,G_3)$ and $L(G_3) = m$, for certain $G_1,G_2,G_3 \in (T\backslash\{\otimes\})^*$.

Proof. From Lemma 4.5 it follows that $typ(t,\tau) = \pi(F_1,F_2) *$
$* typ(\omega(\psi),F_3 \,\&\, \tau)$, for certain $F_1,F_2,F_3 \in (T\backslash\{\otimes\})^*$ and $L(F_3) = m$. Furthermore, by Lemma 2.6 $PC(\omega(\psi),m)$ and, hence, $\psi \in perm(m)$. If $\psi \in perm(m)$ and $L(F_3) = m$ then $typ(\omega(\psi),F_3 \,\&\, \tau) = \rho(\emptyset,\emptyset,F_3 \circ \psi)$ and $L(F_3 \circ \psi) = m$. By taking F_1 for G_1, F_2 for G_2 and $F_3 \circ \psi$ for G_3 we see that $typ(t,\tau) = \rho(G_1,G_2,G_3)$ and $L(G_3) = m$. □

Theorem 4.1.

Let $\delta\, t\, \lambda_f\, u \in \lambda_T\sigma$, $k \in \mathbb{N}$ and τ be a type context. If $typ(\delta\, t\, \lambda_f\, u,\tau) \neq \otimes$, $PC(t,0)$ and $PC(u,k)$ then $(t,u,1) \in dom(\Sigma_f)$ and $typ(\delta\, t\, \lambda_f\, u,\tau) = typ(\Sigma_f(t,u,1),\tau)$.

Proof. To begin with we have the following data

1a. $PC(t,0)$;

1b. $PC(u,k)$;

2a. $typ(t,\tau) = f$;

2b. $typ(\delta\, t\, \lambda_f\, u,\tau) = typ(u,\langle f\rangle \,\&\, \tau) \neq \otimes$.

The proof is given by induction on $L(u)$.

(i) $u = \xi(n)$:

(1) $n = 1$: From 2b and Lemma 4.2 it follows that $f \in \Gamma$, hence, by 2a, $LS(t) \in rge(\xi)$. Therefore $\Sigma_f(t,u,1) = t$ and

$$typ(\Sigma_f(t,u,1),\tau) =$$

$$= typ(t,\tau) =$$

$$= f = \qquad (2a)$$

$$= typ(u,<f> \,\&\, \tau) =$$

$$= typ(\delta\, t\, \lambda_f\, u,\tau) \; . \qquad (2b)$$

(2) $n > 1$: Then $\Sigma_f(t,u,1) = \xi(n-1)$. From 2b and Lemma 4.2 it follows that $\tau(n-1) \in \Gamma$. Therefore

$$typ(\Sigma_f(t,u,1),\tau) =$$

$$= typ(\xi(n-1),\tau) =$$

$$= \tau(n-1) =$$

$$= typ(u,<f> \,\&\, \tau) =$$

$$= typ(\delta\, t\, \lambda_f\, u,\tau) \; . \qquad (2b)$$

(ii) $u = \omega(\psi)$: If $1 \in rge(\psi)$ then $\Sigma_f(t,u,1) = \delta\, t\, \lambda_f\, u$, and if $1 \notin rge(\psi)$ then $\psi = \emptyset$ and $\Sigma_f(t,u,1) = \omega(\emptyset)$. If $\Sigma_f(t,u,1) = \omega(\emptyset)$ then

$$typ(\Sigma_f(t,u,1),\tau) =$$

$$= \rho(\emptyset,\emptyset,\emptyset) =$$

$$= typ(u,<f> \,\&\, \tau) =$$

$$= typ(\delta\, t\, \lambda_f\, u,\tau) \; . \qquad (2b)$$

(iii) $u = \lambda_g v$: Then $\Sigma_f(t,u,1) = \lambda_g\, \Sigma_f(t,v,2) = \lambda_g\, \Sigma_f(\underline{\varphi_1}\, t, \underline{\vartheta_1}\, v, 1)$, if $(\underline{\varphi_1}\, t, \underline{\vartheta_1}\, v, 1) \in dom(\Sigma_f)$. We now apply the induction hypothesis to $\underline{\vartheta_1}\, v$. From 1a and the corollary to Lemma 2.7 it follows that $PC(\underline{\varphi_1}\, t, 0)$. From 1b it follows that $PC(v, k+1)$ and by Lemma 2.8 we have $PC(\underline{\vartheta_1}\, v, k+1)$ ($\vartheta_1 = \psi\hat{}$, for a $\psi \in perm(2)$). Furthermore

$$typ(\underline{\varphi_1}\, t, <g> \,\&\, \tau) =$$

$= \mathrm{typ}(t,(\langle g\rangle \,\&\, \tau) \circ \varphi_1) =$ (Lemma 4.4)

$= \mathrm{typ}(t,\tau) =$

$= f$ (2a)

and

$\mathrm{typ}(\underline{\vartheta_1}\, v,\langle f\rangle \,\&\, \langle g\rangle \,\&\, \tau) =$

$= \mathrm{typ}(v,(\langle f\rangle \,\&\, \langle g\rangle \,\&\, \tau) \circ \vartheta_1) =$ (Lemma 4.4)

$= \mathrm{typ}(v,\langle g\rangle \,\&\, \langle f\rangle \,\&\, \tau) \neq$

$\neq \otimes$. (2b)

Therefore $\mathrm{typ}(\delta\, \underline{\varphi_1}\, t\, \lambda_f\, \underline{\vartheta_1}\, v,\langle g\rangle \,\&\, \tau) = \mathrm{typ}(\underline{\vartheta_1}\, v,\langle f\rangle \,\&\, \langle g\rangle \,\&\, \tau) \neq \otimes$. By the induction hypothesis $(\underline{\varphi_1}\, t,\underline{\vartheta_1}\, v,1) \in \mathrm{dom}(\Sigma_f)$ and, hence, $\Sigma_f(t,u,1) = \lambda_g\, \Sigma_f(\underline{\varphi_1}\, t,\underline{\vartheta_1}\, v,1)$. Furthermore

$\mathrm{typ}(\Sigma_f(t,u,1),\tau) =$

$= \pi(\langle g\rangle,\emptyset) * \mathrm{typ}(\Sigma_f(\underline{\varphi_1}\, t,\underline{\vartheta_1}\, v,1),\langle g\rangle \,\&\, \tau) =$

$= \pi(\langle g\rangle,\emptyset) * \mathrm{typ}(\delta\, \underline{\varphi_1}\, t\, \lambda_f\, \underline{\vartheta_1}\, v,\langle g\rangle \,\&\, \tau) =$ (induction hypothesis)

$= \pi(\langle g\rangle,\emptyset) * \mathrm{typ}(v,\langle g\rangle \,\&\, \langle f\rangle \,\&\, \tau) =$

$= \mathrm{typ}(u,\langle f\rangle \,\&\, \tau) =$

$= \mathrm{typ}(\delta\, t\, \lambda_f\, u,\tau)$. (2b)

(iv) $u = \sigma(n,m)v$: From 2b it follows that $f \in P$ and therefore $f = \rho(F_1,F_2,F_3)$, for certain $F_1,F_2,F_3 \in (T\backslash\{\otimes\})^*$. Furthermore from 2a, 1a and Lemmas 4.2 and 4.6 it follows that $LS(t) = \omega(\psi)$, for some segmap ψ, and $\psi \in \mathrm{perm}(m)$ and $L(F_3) = m$, where $m = W(t)$.

(1) $n = 1$: Let $t = s\omega(\psi)$ then $\Sigma_f(t,u,1) = s\, \underline{\psi}\hat{}\, \Sigma_f(t,v,1+m) =$ $= s\, \underline{\psi}\hat{}\, \Sigma_f(\underline{\varphi_m}\, t,\underline{\vartheta_m}\, v,1)$, if $(\underline{\varphi_m}\, t,\underline{\vartheta_m}\, v,1) \in \mathrm{dom}(\Sigma_f)$. We now apply the induction hypothesis to $\underline{\vartheta_m}\, v$. From 1a and the corollary to Lemma 2.7 it follows that $PC(\underline{\varphi_m}\, t,0)$. From 1b it follows that $PC(v,k+m)$ and by Lemma 2.8 we have $PC(\underline{\vartheta_m}\, v,k+m)$. Furthermore

$$\mathrm{typ}(\underline{\varphi}_m\, t, F_3 \,\&\, \tau) =$$

$$= \mathrm{typ}(t, (F_3 \,\&\, \tau) \circ \varphi_m) = \qquad \text{(Lemma 4.4)}$$

$$= \mathrm{typ}(t, \tau) =$$

$$= f \ .$$

and

$$\mathrm{typ}(\underline{\vartheta}_m\, v, \langle f \rangle \,\&\, F_3 \,\&\, \tau) =$$

$$\mathrm{typ}(v, (\langle f \rangle \,\&\, F_3 \,\&\, \tau) \circ \vartheta_m) = \qquad \text{(Lemma 4.4)}$$

$$= \mathrm{typ}(v, F_3 \,\&\, \langle f \rangle \,\&\, \tau) \neq$$

$$\neq \otimes \ .$$

Therefore $\mathrm{typ}(\delta\, \underline{\varphi}_m\, t\, \lambda_f\, \underline{\vartheta}_m\, v, F_3 \,\&\, \tau) = \mathrm{typ}(\underline{\vartheta}_m\, v, \langle f \rangle \,\&\, F_3 \,\&\, \tau) \neq \otimes$. By the induction hypothesis $(\underline{\varphi}_m\, t, \underline{\vartheta}_m\, v, 1) \in \mathrm{dom}(\Sigma_f)$ and, hence, $\Sigma_f(t,u,1) = s\, \underline{\psi}\, \Sigma_f(\underline{\varphi}_m\, t, \underline{\vartheta}_m\, v, 1)$. Furthermore let $G_1, G_2, G_3 \in (T \setminus \{\otimes\})^*$ be such that $L(G_3) = m$ and $\mathrm{typ}(s \,\&\, w, \tau) = \pi(G_1, G_2) * \mathrm{typ}(w, G_3 \,\&\, \tau)$, for arbitrary $w \in \lambda_T\sigma$ such that $s \,\&\, w \in \lambda_T\sigma$ (cf. Lemma 4.5). By taking $w = \omega(\psi)$ we see that

$$f =$$

$$= \mathrm{typ}(t, \tau) =$$

$$= \pi(G_1, G_2) * \mathrm{typ}(\omega(\psi), G_3 \,\&\, \tau) =$$

$$= \pi(G_1, G_2) * \rho(\emptyset, \emptyset, (G_3 \,\&\, \tau) \circ \psi) =$$

$$= \pi(G_1, G_2) * \rho(\emptyset, \emptyset, G_3 \circ \psi) =$$

$$= \rho(G_1, G_2, G_3 \circ \psi) =$$

$$= \rho(F_1, F_2, F_3) \ .$$

Hence $G_1 = F_1$, $G_2 = F_2$ and $G_3 \circ \psi = F_3$. Furthermore

$$\mathrm{typ}(\Sigma_f(t,u,1), \tau) =$$

$= typ(s\,\underline{\psi}\hat{}\,\Sigma_f(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1),\tau) =$

$= \pi(G_1,G_2) * typ(\underline{\psi}\hat{}\,\Sigma_f(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1),G_3\,\&\,\tau) =$

$= \pi(F_1,F_2) * typ(\Sigma_f(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1),(G_3\,\&\,\tau)\circ\psi\hat{}) =$ (Lemma 4.4)

$= \pi(F_1,F_2) * typ(\Sigma_f(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1),F_3\,\&\,\tau) =$

$= \pi(F_1,F_2) * typ(\delta\,\underline{\varphi}_m\,t\,\lambda_f\,\underline{\vartheta}_m\,v,F_3\,\&\,\tau) =$ (induction hypothesis)

$= \pi(F_1,F_2) * typ(\underline{\vartheta}_m\,v,<f>\,\&\,F_3\,\&\,\tau) =$

$= \pi(F_1,F_2) * typ(v,F_3\,\&\,<f>\,\&\,\tau) =$

$= typ(\sigma(1,m)v,<f>\,\&\,\tau) =$

$= typ(\delta\,t\,\lambda_f\,u,\tau)$.

(2) $n > 1$: Then $\Sigma_f(t,u,1) = \sigma(n-1,m)\,\Sigma_f(t,v,m+1) =$
$= \sigma(n-1,m)\,\Sigma_f(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1)$, if $(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1) \in dom(\Sigma_f)$. From 2b it follows that $n-1 \in dom(\tau)$ and $\tau(n-1) = \rho(F_1,F_2,F_3)$. Analogous to case (iv)(1) we have $PC(\underline{\varphi}_m\,t,0)$, $PC(\underline{\vartheta}_m\,v,k+m)$ and $typ(\delta\,\underline{\varphi}_m\,t\,\lambda_f\,\underline{\vartheta}_m\,v,F_3\,\&\,\tau) \neq \otimes$. By the induction hypothesis $(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1) \in dom(\Sigma_f)$ and, hence, $\Sigma_f(t,u,1) =$
$= \sigma(n-1,m)\,\Sigma_f(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1)$. Furthermore

$typ(\Sigma_f(t,u,1),\tau) =$

$= \pi(F_1,F_2) * typ(\Sigma_f(\underline{\varphi}_m\,t,\underline{\vartheta}_m\,v,1),F_3\,\&\,\tau) =$

$= \pi(F_1,F_2) * typ(\delta\,\underline{\varphi}_m\,t\,\lambda_f\,\underline{\vartheta}_m\,v,F_3\,\&\,\tau) =$ (induction hypothesis)

$= \pi(F_1,F_2) * typ(\underline{\vartheta}_m\,v,<f>\,\&\,F_3\,\&\,\tau) =$

$= \pi(F_1,F_2) * typ(v,F_3\,\&\,<f>\,\&\,\tau) =$

$= typ(\sigma(n,m)v,<f>\,\&\,\tau) =$

$= typ(\delta\,t\,\lambda_f\,u,\tau)$. (2b)

(v) $u = \delta vw$: Then $\Sigma_f(t,u,1) = \delta\,\Sigma_f(t,v,1)\,\Sigma_f(t,w,1)$. From 1b and 2b it follows that $PC(v,0)$ and $typ(v,<f>\,\&\,\tau) \neq \otimes$. By the induction

hypothesis $(t,v,1) \in dom(\Sigma_f)$ and $typ(\delta\, t\, \lambda_f\, v,\tau) =$
$= typ(\Sigma_f(t,v,1),\tau) = typ(v,<f> \& \tau)$. Furthermore, from 1b and 2b it follows that $PC(w,k)$ and $typ(w,<f> \& \tau) \neq \otimes$. By the induction hypothesis $(t,w,1) \in dom(\Sigma_f)$ and $typ(\delta\, t\, \lambda_f\, w,\tau) =$
$= typ(\Sigma_f(t,w,1),\tau) = typ(w,<f> \& \tau)$. Therefore

$$typ(\Sigma_f(t,u,1),\tau) =$$

$$= typ(\delta\, \Sigma_f(t,v,1)\, \Sigma_f(t,w,1),\tau) =$$

$$= \pi(\emptyset,<typ(\Sigma_f(t,v,1),\tau)>) * typ(\Sigma_f(t,w,1),\tau) =$$

$$= \pi(\emptyset,<typ(\delta\, t\, \lambda_f\, v,\tau)>) * typ(\delta\, t\, \lambda_f\, w,\tau) =$$ (induction hypothesis)

$$= \pi(\emptyset,<typ(v,<f> \& \tau)>) * typ(w,<f> \& \tau) =$$

$$= typ(\delta vw,<f> \& \tau) =$$

$$= typ(\delta\, t\, \lambda_f\, u,\tau)\ .$$ (2b) □

Corollary (Closure).

(1) If $PC(\delta\, t\, \lambda_f\, u,\ell)$ $(\ell \geq 0)$ and $typ(\delta\, t\, \lambda_f\, u,\tau) \neq \otimes$ then $(t,u,1) \in dom(\Sigma_f)$ and $typ(\Sigma_f(t,u,1),\tau) = typ(\delta\, t\, \lambda_f\, u,\tau)$;

(2) Let $t,u \in \lambda_T\sigma$ and $t \twoheadrightarrow_\beta u$. If $PC(t,0)$ and $typ(t,\tau) \neq \otimes$ then $typ(t,\tau) = typ(u,\tau)$.

REFERENCES

Barendregt [81]: Barendregt, H.P. The Lambda calculus: Its Syntax and Semantics. North Holland, 1981.

de Bruijn [72]: de Bruijn, N.G. Lambda calculus notation with nameless dummies, a tool for automatic formula notation, with application to the Church-Rosser theorem. Indag. Math. 34, 1972, pp. 381-392.

de Bruijn [78a]: de Bruijn, N.G. Lambda calculus with namefree formulas involving symbols that represent reference transforming mappings. Indag. Math. 40, 1978, pp. 348-356.

de Bruijn [78b]: de Bruijn, N.G. A Namefree Lambda Calculus with Facilities for Internal Definition of Expressions and Segments. Dept. of Math. and Comp. Sci., Eindhoven University of Technology, 1978, TH-Report 78-WSK-03.

de Bruijn [80]: de Bruijn, N.G. A survey of the project AUTOMATH. Seldin & Hindley [80], pp. 579-607.

Church [40]: Church, A. A formulation of the simple theory of types. J. Symbolic Logic 5, 1940, pp. 56-68.

van Daalen [80]: van Daalen, D.T. The language theory of AUTOMATH. Dissertation. Eindhoven University of Technology, 1980.

van Dalen [78]: van Dalen, D., Doets, H.C., de Swart, H. Sets: Naive, Axiomatic and Applied. Pergamon Press, 1978.

Jutting [81]: van Benthem Jutting, L.S. Description of AUT 68. Memorandum 86-01. Dept. of Math. and Comp. Sci., Eindhoven University of Technology, 1986.

Nederpelt [73]: Nederpelt, R.P. Strong normalization for a typed lambda calculus with lambda structured types. Dissertation. Eindhoven University of Technology, 1973.

Newman [42]: Newman, M.H.A. On theories with a combinatorial definition of "equivalence". Ann. of Math. (2) 43, 1942, pp. 223-243.

Seldin & Hindley [80]: Seldin, J.P. and Hindley, J.R. To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism, edited by J.P. Seldin and J.R. Hindley. Academic Press, 1980.

Shoenfield [67]: Shoenfield, J.R. Mathematical logic. Addison Wesley, Reading (Mass.), 1967.

INDEX OF DEFINITIONS

SAMENVATTING

Het onderzoek waarvan in dit proefschrift verslag wordt gedaan heeft betrekking op een gegeneraliseerd systeem van λ-calculus, geheten $\lambda\sigma$. Het systeem wijkt af van bestaande λ-calculi doordat een geheel nieuwe klasse van termen is opgenomen, geheten *segmenten*. Segmenten waren oorspronkelijk ontworpen door N.G. de Bruijn om te zorgen voor bepaalde afkortingsfaciliteiten in de wiskundige taal AUTOMATH. Het onderwerp van dit proefschrift is een taaltheoretische studie van de $\lambda\sigma$-calculus.

In Hoofdstuk 1 wordt een uitgebreide informele beschrijving gegeven van het $\lambda\sigma$-systeem en worden de voornaamste verschillen aangegeven t.o.v. klassieke ongetypeerde λ-calculus. Tevens wordt er in Sectie 1.2 van dit hoofdstuk een beschrijving gegeven van een getypeerde versie van $\lambda\sigma$, geheten $\lambda_T\sigma$. De types in $\lambda_T\sigma$ zijn een extensie van zogenaamde "simple types" in de klassieke getypeerde λ-calculus, waarbij de uitbreiding hieruit bestaat dat er tevens types worden geconstrueerd voor segmenten en segmentvariabelen.

In Hoofdstuk 2 worden de belangrijkste definities en basisresultaten gegeven.

In Hoofdstuk 3 wordt een bewijs gegeven van de Church-Rosser eigenschap voor $\lambda\sigma$ volgens de methode van de eindige ontwikkelingen en in Hoofdstuk 4 wordt een bewijs gegeven van de geslotenheidseigenschap voor $\lambda_T\sigma$.

CURRICULUM VITAE

De schrijver van dit proefschrift werd op 2 februari 1952 geboren te Groningen. In 1971 behaalde hij aan de Rijks H.B.S. te Groningen het diploma H.B.S.-B. Na de vervulling van zijn diensttijd is hij Wis- en Natuurkunde gaan studeren aan de Rijks Universiteit Groningen. In maart 1982 slaagde hij voor het doctoraalexamen Wiskunde met als hoofdvak mathematische logica. Sinds april 1982 is hij werkzaam bij de Technische Hogeschool Eindhoven, meer in het bijzonder in de Onderafdeling der Wiskunde en Informatica bij Prof.dr. N.G. de Bruijn.