Lambda Calculus with Patterns

Vincent van Oostrom

Department of Mathematics and Computer Science,
Vrije Universiteit, de Boelelaan 1081 A, 1081 HV Amsterdam.
email: oostrom@cs.vu.nl

Abstract

The $\lambda\phi$-calculus is an extension of the λ-calculus with a pattern matching facility. The form of the argument of a function can be specified and hence $\lambda\phi$-calculus is more convenient than ordinary λ-calculus. We explore the basic theory of $\lambda\phi$-calculus, establishing results such as confluence. In doing so, we find some requirements for patterns that guarantee confluence. Our work can be seen as giving some foundations for implementations of functional programming languages.

1. Introduction

Functional programming languages can be considered as more or less sugared versions of the lambda calculus (λ). Therefore it is natural to specify the meaning of the constructs in functional programming languages via a translation of these constructs into λ, after which one can profit from the extensive amount of theory that is known about λ, such as CR (the Church-Rosser or confluence property), see e.g. [HS86] or [Bar84]. In [Pey87] a translation of a subset of Miranda[1] is given in two steps, first into an intermediate language, the extended lambda calculus, and then into λ. The extended lambda calculus is λ extended with a *pattern matching lambda abstraction* facility. (There are some other, less important, features in this extended lambda calculus that we omit in the present work.)

Pattern matching lambda abstraction is a generalisation of ordinary abstraction. In λ one transforms a term into a function by abstracting from a variable. Application of that function to an argument is then performed by substituting the argument for the free occurrences of the variable in the term (the function body). We can apply this function to any λ-term because a variable 'matches', i.e. can be instantiated to, any λ-term. In a pattern matching lambda abstraction the pattern abstracted from specifies which *form* the argument must have in order to match. Such a function is only applicable to instantiations of the pattern.

[1] Miranda is a trademark of Research Software Limited.

Application is then performed by substituting the terms bound to the free variables in the pattern into the function body.

We use this idea for *typed* lambda calculus to extend *untyped* lambda calculus with a similar construct. The undecidability of equality in untyped lambda calculus forces us to use 'syntactic' matching instead of the 'semantic' matching used in [Pey87].

As a simple example consider the natural numbers defined in the usual Peano style, that is, n is represented by $S^n(0)$. The predecessor and successor functions are easily expressed as

$$\mathrm{P}^- \equiv (\lambda S(x).x) \text{ and } \mathrm{S}^+ \equiv (\lambda x.S(x))$$

respectively. As another example, the projection functions for pairs can be expressed as

$$\pi_0 \equiv (\lambda[x,y].x) \text{ and } \pi_1 \equiv (\lambda[x,y].y)$$

In this paper a way in which $S(x)$, $[x,y]$ and other 'constructs' can be represented as syntactic patterns is given.

If we allow abstraction from arbitrary λ-terms we run, not surprisingly, into difficulties because different evaluation orders can lead to different results. As patterns are meant to be fixed constructs which can not be evaluated, we should allow as patterns only terms which can not be (partially) destroyed by evaluating parts of the term they are in. We give a restriction, the *rigid pattern condition* (RPC), for this to hold. We prove that the system $\lambda\phi$, with patterns in a set Φ which satisfies RPC, has the Church-Rosser property. A consequence is that $\lambda\phi$ is a conservative extension of λ in the sense that the convertibility relation for $\lambda\phi$ restricted to λ-terms coincides with the convertibility relation for λ.

A set Π is given satisfying some simple syntactic restrictions such that it is RPC. The way in which constructors can be modelled in $\Lambda\Pi$ is shown.

Next we prove the Finite Developments Theorem for RPC-systems. We obtain an exact upper bound on the maximal development of a term.

In this paper we will use the 'naive' notation for terms, that is, with variable names. However in Appendix C we give a simple extension of De Bruijn's notation to handle $\lambda\phi$-terms.

Many of the notions used and proofs given are taken from [Bar84] and adapted to our systems. (For readers familiar with [Bar84], inspection of how the notions are extended probably will do to verify our proofs.)

2. Preliminaries

For the standard notation used in this paper we refer the reader to Appendix A. Now we fix the somewhat less common notations regarding trees and occurrences. The reader already familiar with substitutions defined by means of occurrences can safely skip this section.

Terms can be considered as labelled trees. It is then natural to identify a node with the unique path leading from the root towards it, where paths are specified by sequences of numbers. We first define some operations on sequences.

DEFINITION 2.1. SEQUENCES

1. $\mathrm{Seq}(X)$ is the set of finite sequences over a set X.
2. Sequences are denoted by 'arrowed' letters, e.g. $\vec{x}$, $\vec{N}$.
3. The *length* of a sequence $\vec{x}$ is denoted by $|\vec{x}|$.
4. If we want to name the individual elements of a sequence $\vec{x}$ of length $n \in \mathbb{N}$, we denote $\vec{x}$ by $\langle x_1, \ldots, x_n \rangle$. $\langle\rangle$ denotes the empty sequence.

Where possible we extend our notions for elements of a set X to sequences over X.

DEFINITION 2.2. OPERATIONS ON SEQUENCES
Let $\vec{x}, \vec{y} \in \mathrm{Seq}(X)$ and $S, T \subseteq \mathrm{Seq}(X)$.

1. If $\vec{x} = \langle x_1, \ldots, x_n \rangle, \vec{y} = \langle y_1, \ldots, y_m \rangle$, then $\vec{x} * \vec{y} = \langle x_1, \ldots, x_n, y_1, \ldots, y_m \rangle$ is the *concatenation* of $\vec{x}$ and $\vec{y}$.
2. $\vec{z} = \vec{x} \backslash \vec{y}$ is another notation for $\vec{x} * \vec{z} = \vec{y}$. We say $\vec{z}$ is the *left division of $\vec{y}$ by $\vec{x}$*.
3. $S * T = \{\vec{x} * \vec{y} \mid \vec{x} \in S \ \& \ \vec{y} \in T\}$.
4. $S \backslash T = \{\vec{x} \backslash \vec{y} \mid \vec{x} \in S \ \& \ \vec{y} \in T \ \& \ \exists \vec{z} : \vec{x} \backslash \vec{y} = \vec{z}\}$

DEFINITION 2.3. ORDERINGS ON SEQUENCES
Let $\prec$ be a partial order on X, $\vec{x}, \vec{y} \in \mathrm{Seq}(X)$, then

1. $\vec{x}$ is a *prefix* of $\vec{y}$, notation $\vec{x} \trianglelefteq \vec{y}$, if $\exists \vec{z} : \vec{x} * \vec{z} = \vec{y}$,
2. $\vec{x}$ is a *proper* prefix of $\vec{y}$, notation $\vec{x} \triangleleft \vec{y}$, if $\vec{x} \trianglelefteq \vec{y}$, but $\vec{y} \not\trianglelefteq \vec{x}$.
3. $\vec{x}$ and $\vec{y}$ are *disjoint*, notation $\vec{x} \perp \vec{y}$, if $\vec{x}$ and $\vec{y}$ are incomparable that is, both $\vec{x} \not\trianglelefteq \vec{y}$ and $\vec{y} \not\trianglelefteq \vec{x}$. (This does not mean that disjoint $\vec{x}$,$\vec{y}$ may not have elements in common. The word 'disjoint' anticipates on future usage for disjoint subterms.)
4. $\vec{x}$ is lexicographically less than or equal to $\vec{y}$, notation $\vec{x} \precsim_{\mathrm{l}} \vec{y}$, if $\nexists \vec{z}, \vec{x'}, \vec{y'}, y \prec x$, such that $\vec{x} = \vec{z} * \langle x \rangle * \vec{x'}$ and $\vec{y} = \vec{z} * \langle y \rangle * \vec{y'}$,
5. $\vec{x}$ is lexicographically less than $\vec{y}$, if $\vec{x} \prec_{\mathrm{l}} \vec{y}$, where $\prec_{\mathrm{l}}$ is the partial order generated by $\precsim_{\mathrm{l}}$.

If S is the singleton set $\{s\}$ then we write $s * T, s \backslash T$ for $S * T, S \backslash T$ respectively.
Trees are just 'prefix-closed sets of sequences'.

DEFINITION 2.4. TREES
Let $\emptyset \neq T \subseteq \mathrm{Seq}(X)$.

1. A set T *is an X-tree* if

$$\vec{y} \in T \ \& \ \vec{x} \triangleleft \vec{y} \ \Rightarrow \ \vec{x} \in T$$

2. $\langle\rangle$ is the *root of* T.

3. The elements from T are T*-nodes*.

4. For $\vec{z} \in T$, $\delta(\vec{z}) = \#\{x \mid \vec{z} * \langle x \rangle \in T\}$, is its *out-degree*.

5. If a node has out-degree 0, then it is a T*-leaf*.

6. If a node has out-degree > 0, then it is an *internal* T-node.

7. T is *complete* if for all internal T-nodes $\vec{x}$ and all $y \in X : \vec{x} * \langle y \rangle \in T$.

8. u,v,w,... denote arbitrary nodes of trees, also called *occurrences*. Notice that occurrences are *not* arrowed.

If X is equipped with a partial order $\prec$, then the lexicographic ordering gives a total ordering on nodes, so a tree is then called an $\prec$*-ordered* tree. The set of all $\prec$-ordered X-trees is denoted by $\mathrm{Tree}(X, \prec)$. A complete tree in $\mathrm{Tree}(\{0,1\}, <)$-tree is a *binary tree*. The set of all binary trees is denoted by Bintree.

As terms correspond to trees, operations on terms, e.g. replacing an expression by its result, correspond to operations on terms. Note that if $u \in T$, then $u\backslash T$ is an (ordered) tree again.

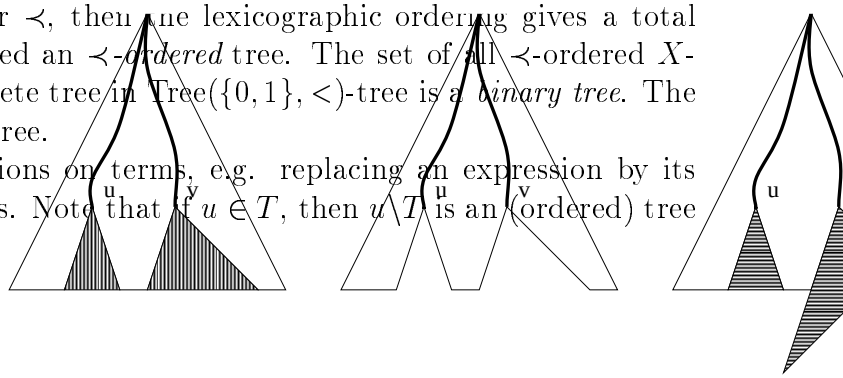

Figure 1: Tree operations

Definition 2.5. Operations on Trees
Let S be an (ordered) tree, let $\vec{u}$ be a sequence of pairwise disjoint S-nodes and let $\vec{T}$ be a sequence of (ordered) trees (with the same length n as $\vec{u}$).

1. The $\vec{u}$*-pruning of* S (notation $S[\vec{u}]$ or $S[u_1, \ldots, u_n]$) is the tree :

$$S - \bigcup_{i=1}^{n} u_i * (u_i \backslash S)$$

(So, $S[\vec{u}]$ originates from S by leaving away the subtrees located at $u_1, \ldots, u_n$, see Figure 1.)

2. The *replacement of* $\vec{u}$ *by* $\vec{T}$ *in* S (notation $S[\vec{T}/\!/\vec{u}]$ or $S[T_1,\ldots,T_n/\!/u_1,\ldots,u_n]$) is the tree :
$$S[\vec{u}] \cup \bigcup_{i=1}^{n} u_i * T_i$$
(see Figure 1)

3. A special case of replacement is called substitution, namely if the $\vec{u}$ in $S[\vec{T}/\!/\vec{u}]$ is a sequence of S-leaves. Then we write $S[\vec{T}/\vec{u}]$ and say that $S[\vec{T}/\vec{u}]$ is the *substitution of* $\vec{T}$ *for* $\vec{u}$ *in* S.

4. It will be implicit in the notations $S[\vec{T}/\!/\vec{x}]$ and $S[\vec{T}/\vec{x}]$, that $\vec{T}$ and $\vec{x}$ are sequences of the same length.

The definitions can easily be extended to labelled trees, i.e. trees where each node has a label attached to it.

Definition 2.6. Labelled Trees
Let A be a ranked set, i.e. a set with associated ranking function $\varrho : A \to \mathbb{N}$.

1. An *A-labelled* X-tree is a pair $M = (T, \ell)$, where
   (a) T is an X-tree.
   (b) $\ell : T \to A$ is a total function such that $\forall u \in T : \delta(u) = \varrho(\ell(u))$.

2. T is the *underlying tree* of M, and ℓ is the *labeling* of M. We often suppress mention of M and write T_M and ℓ_M instead of M,ℓ.

This is definition is easily extended to ordered trees. The set of all A-labelled binary trees is denoted by Lbintree(A).

Definition 2.7. Operations on Labelled Trees
Let $M, \vec{N}$ be (sequences) of A-labelled trees, and let $\vec{u}$ be a sequence of pairwise disjoint M-nodes (with the same length n as $\vec{N}$).

1. The *replacement of* $\vec{u}$ *by* $\vec{N}$ *in* M (notation $M[\vec{N}/\!/\vec{u}]$ or $M[N_1,\ldots,N_n/\!/u_1,\ldots,u_n]$) is the A-labelled tree defined by :
   (a) $T_{M[\vec{N}/\!/\vec{u}]} = T_M[T_{\vec{N}}/\!/\vec{u}]$,
   (b) $\ell_{M[\vec{N}/\!/\vec{u}]}(w) = \ell_M(w)$, $\forall w \in T_M[\vec{u}]$
   $\ell_{M[\vec{N}/\!/\vec{u}]}(w) = \ell_{N_i}(u_i \backslash w)$, $\forall i \in \mathbb{N}, \forall w \in u_i * N_i$

2. As for unlabelled trees, to denote that $\vec{u}$ is a set of leaves in the replacement $M[\vec{N}/\!/\vec{u}]$, we write $M[\vec{N}/\vec{u}]$ and say that $M[\vec{N}/\vec{u}]$ is the substitution of $\vec{N}$ for $\vec{u}$ in M.

3. For $a \in A$, $\mathcal{O}_a(M) = \{u \in T_M \mid \ell_M(u) = a\}$, is the set of a-occurrences of M. We extend this notation to subsets of A. This function yields all occurrences whose label is in the specified set.

Where symbols are understood from the context we sometimes omit them in our notations.

3. Generalised Lambda Terms

In this section we give the formal definition of the generalised lambda-calculus and extend the definitions of convertibility, α-congruence, substitution and contexts accordingly.

Throughout this paper we assume $\mathcal{V} = \{v_0, v_1, v_2, \ldots\}$ to be a countably infinite set of *variables*. Furthermore x,y,z,... denote arbitrary variables.

Definition 3.1. Generalised Lambda Terms

1. *Generalised λ-terms* are words over the alphabet $\mathcal{V} \cup \{\lambda, ., (,)\}$. We use L,M,N as typical variables over generalised λ-terms.

2. The set $\Lambda\Lambda$ of generalised λ-terms is defined inductively as follows :

   (a) $\mathcal{V} \subseteq \Lambda\Lambda$

   (b) $M, N \in \Lambda\Lambda \;\Rightarrow\; (MN) \in \Lambda\Lambda$ (*application*)

   (c) $M, N \in \Lambda\Lambda \;\Rightarrow\; (\lambda M.N) \in \Lambda\Lambda$ (*generalised abstraction*)

Alternatively we can view generalised λ-terms as labelled binary trees. The representations of the terms $(\lambda M.N)$, (MN) and $((\lambda x.x)y)$ are given in Figure 2.

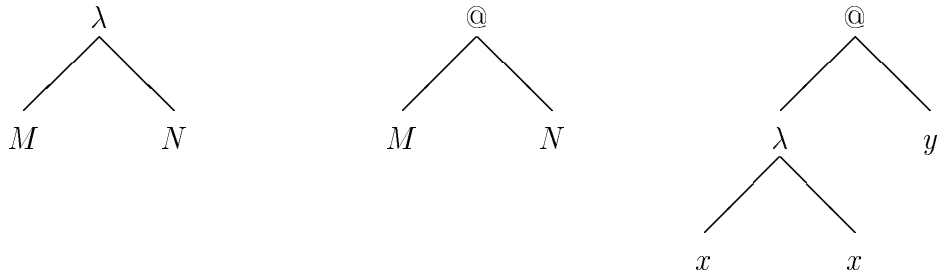

Figure 2: tree representation of terms

Alternative Definition 3.2. Generalised Lambda Terms
$\Lambda\Lambda = \mathrm{Lbintree}(\mathcal{L})$, where $\mathcal{L} = \mathcal{V} \cup \{\lambda, @\}$, with ranking function ϱ defined by

1. $\varrho(b) = 0$, for $b \in \mathcal{V}$,

2. $\varrho(b) = 2$, for $b \in \{\lambda, @\}$.

We will freely mix usage of the two, obviously equivalent, representations. To denote that a generalised λ-term M has the same representation, in either the first or second sense, as a generalised λ-term N, we write $M \equiv N$, so $\equiv$ is the diagonal of the set of pairs of terms. If M is a generalised λ-term in the first sense, we write (T_M, ℓ_M) to denote the corresponding representation of M in the second sense (as labelled tree). By abuse of language we sometimes

say that a term M has a certain property when we mean that the underlying tree of M has that property, e.g. $u \in M$ instead of $u \in T_M$.

These two definitions allow us to specify various operations on terms in both a structural and a denotational way. In most cases we give the specification in whichever is the most convenient way.

DEFINITION 3.3. PATTERN AND BODY

1. M is the *pattern* and N is the *body* of the term $(\lambda M.N)$.
2. $\mathcal{PO}(M) = \{u \in M \mid u = v * \langle 0 \rangle \wedge v \in \mathcal{O}_\lambda(M)\}$, the set of pattern occurrences of M.
   $\mathcal{BO}(M) = \{u \in M \mid u = v * \langle 1 \rangle \wedge v \in \mathcal{O}_\lambda(M)\}$, the set of body occurrences of M.

Sometimes we need to restrict the set of patterns.

DEFINITION 3.4. $\lambda\phi$-TERMS

1. For all $\Phi \subseteq \Lambda\Lambda$, the set $\Lambda\Phi$ of *$\lambda\phi$-terms* is defined inductively as follows:
   (a) $\mathcal{V} \subseteq \Lambda\Phi$
   (b) $M, N \in \Lambda\Phi \;\Rightarrow\; (MN) \in \Lambda\Phi$
   (c) $X \in \Phi,\ M \in \Lambda\Phi \;\Rightarrow\; (\lambda X.M) \in \Lambda\Phi$
2. With X,Y,Z,... we denote arbitrary patterns in Φ.

If we choose $\Phi = \Lambda\Lambda$, then we do not restrict $\Lambda\Lambda$, so $\Lambda\Lambda\Lambda = \Lambda\Lambda$. If we choose $\Phi = \mathcal{V}$ then we obtain obviously the ordinary set of λ-terms, so we identify $\Lambda\mathcal{V}$ with Λ.

NOTATION

1. If $\vec{X} \equiv \langle X_1, \ldots, X_n \rangle$ then $(\lambda\vec{X}.M) \equiv (\lambda X_1.(\lambda X_2.\cdots(\lambda X_n.M)...))$.
   If $n = 0$ then $(\lambda\vec{X}.M) \equiv M$.
2. If $\vec{N} \equiv \langle N_1, \ldots, N_n \rangle$ then $(M\vec{N}) \equiv (...((MN_1)N_2)\cdots N_n)$.
   If $n = 0$ then $(M\vec{N}) \equiv M$.
3. $|M|$ is the *length* of M, that is, the number of symbols in M.

Note that $(\lambda(xy).(xy))$ and $(\lambda\langle x, y\rangle.(xy))$ do not denote the same generalised λ-term. The former is an abstraction with an application as pattern while the latter is a repeated abstraction from a variable.

The basic equivalence relation on $\lambda\phi$-terms is that of *convertibility*. This relation will be generated, just like in ordinary λ-calculus, by axioms. In order to formulate these axioms, a substitution operator is needed. $M[\vec{N}/\vec{x}]$ denotes the result of simultaneously substituting elements of $\vec{N}$ for corresponding elements of $\vec{x}$ in M. As in the case of ordinary λ-calculus, some care is needed in defining this operation in order to avoid confusion between free and bound variables. This care will be postponed for a moment.

Definition 3.5. Convertibility
The theory $\boldsymbol{\lambda\phi}$ has as formulas

$$M = N$$

where $M, N \in \Lambda\Phi$ and is axiomatised by the following axioms and rules :

1. Reflexivity

$$\frac{}{M = M}$$

2. Symmetry

$$\frac{M = N}{N = M}$$

3. Transitivity

$$\frac{M = N, N = L}{M = L}$$

4. Left monotonicity

$$\frac{M = N}{(LM) = (LN)}$$

5. Right monotonicity

$$\frac{M = N}{(ML) = (NL)}$$

6. Generalised rule ξ

$$\frac{M = N}{(\lambda X.M) = (\lambda X.N)}$$

7. Generalised β-conversion
   Let $\vec{x}$ be the sequence of free variables of X. (to be defined shortly)

$$\frac{}{((\lambda X.M)X[\vec{N}/\vec{x}]) = M[\vec{N}/\vec{x}]}$$

Provability in $\boldsymbol{\lambda\phi}$ of an equation is denoted by $\boldsymbol{\lambda\phi} \vdash M = N$ or often just by $M = N$. If $M = N$, then M and N are called *convertible*. The following example shows the use of generalised β-conversion.

Example 3.6.

1. $((\lambda x.x)y) \equiv ((\lambda x.x)x[y/x]) = x[y/x] \equiv y$.
   Provability in $\boldsymbol{\lambda}$ is a special case of provability in $\boldsymbol{\lambda\phi}$.

2. $((\lambda(xy).(yx))(MN)) \equiv ((\lambda(xy).(yx))(xy)[\langle M,N\rangle/\langle x,y\rangle]) =$
$(yx)[\langle M,N\rangle/\langle x,y\rangle] \equiv (NM)$.
A function F which exchanges the elements of an application does not exist in ordinary lambda calculus as it would lead to the following inconsistency :
$\mathbf{K} = (\mathbf{IK}) = (F(\mathbf{KI})) = (F(\mathbf{I}(\mathbf{KI}))) = ((\mathbf{KI})\mathbf{I}) = \mathbf{I}$.
(So we should and will not allow this in our pattern-matching lambda calculus either.)

3. $((\lambda(xx).x)(MM)) \equiv ((\lambda(xx).x)(xx)[M/x]) = x[M/x] \equiv x$.
We can test whether two terms are identical; another unrealistic feature.

4. $((\lambda(\lambda z.((zx)y)).x)(\lambda z.((zM)N))) = x[\langle M,N\rangle/\langle x,y\rangle] \equiv M$.
Projection on the first element of a pair.

Definition 3.7. Subterm and Variable Occurrences

1. If $N \in \mathrm{Sub}(M) = \{u\backslash M \mid u \in M\}$ then N is a *subterm* of M (notation $N \subseteq M$). If $N \in \mathrm{Sub}(M)$, but $N \not\equiv M$ then N is a *proper* subterm of M (notation $N \subset M$).

   A subterm may occur several times; e.g. $M \equiv (\lambda(x\mathbf{I}).((x\mathbf{I})(x\mathbf{I})))$ has three occurrences of the subterm $\mathbf{I} \equiv (\lambda y.y)$.

2. $\mathcal{V}(M) = \{\ell_M(u) \mid u \in \mathcal{O}_{\mathcal{V}}(M)\}$, the set of variables which occur in M.

3. $\mathcal{FVO}(M)$, the set of *free* variable occurrences of M, and $\mathcal{FV}(M)$, the set of free variables of M, can be defined inductively as follows :

   (a) $\mathcal{FVO}(x) = \{\langle\rangle\}$
   $\mathcal{FV}(x) = \{x\}$,

   (b) $\mathcal{FVO}((MN)) = \langle 0\rangle * \mathcal{FVO}(M) \cup \langle 1\rangle * \mathcal{FVO}(N)$
   $\mathcal{FV}((MN)) = \mathcal{FV}(M) \cup \mathcal{FV}(N)$,

   (c) $\mathcal{FVO}((\lambda X.M)) = \langle 1\rangle * \{u \in \mathcal{FVO}(M) \mid \ell_M(u) \notin \mathcal{FV}(X)\}$
   $\mathcal{FV}((\lambda X.M)) = \mathcal{FV}(M) - \mathcal{FV}(X)$.

4. A variable occurrence that is not free is said to be *bound*.

5. M is *closed* if $\mathcal{FV}(M) = \emptyset$.

6. M is *linear* if $\#\mathcal{FV}(M) = \#\mathcal{FVO}(M)$, i.e. no variable has multiple occurrences in M.

7. If $u\backslash M \equiv (xN)$ then the occurrence $u' = u * \langle 0\rangle$ of x is *active*.

In the sequel, if M denotes a term, we take $\vec{m}$ to be the sequence of free variables of M, unless stated otherwise. (Note that the sequence is not unique but can be made so by assuming an ordering on $\mathcal{V}$.)

Next α-congruence for $\lambda\phi$ can be defined as for λ. See Appendix A for a formal definition or Appendix C for a formalism, based on the De Bruijn notation [dB72] for λ, which identifies α-congruent terms on a syntactic level. Because α-congruent terms exhibit identical functional

behaviour, we consider them to be syntactically equivalent (terms are considered modulo α-congruence). Obviously $\Lambda\Phi$ should be closed under α-congruence.

VARIABLE CONVENTION
If $M_1, \ldots, M_n$ occur in a certain mathematical context (e.g. definition, proof), then in these terms all bound variables are chosen to be different from the free variables.

DEFINITION 3.8. SUBSTITUTION
The result of substituting elements of $\vec{N}$ for the free occurrences of the corresponding variables in a sequence $\vec{x}$, in M (notation $M[\vec{N}/\vec{x}]$ or $M[N_1, \ldots, N_n/x_1, \ldots, x_n]$) is inductively defined as follows.

1. $x_i[\vec{N}/\vec{x}] \equiv N_i$,
   $y[\vec{N}/\vec{x}] \equiv y$, if $y \notin \vec{x}$,
2. $(LM)[\vec{N}/\vec{x}] \equiv (L[\vec{N}/\vec{x}]M[\vec{N}/\vec{x}])$,
3. $(\lambda Y.M)[\vec{N}/\vec{x}] \equiv (\lambda Y.M[\vec{N}/\vec{x}])$.

In the third clause it is not needed to say "provided that $\vec{y} \cap \vec{x} = \emptyset$ and $\vec{y} \cap \mathcal{FV}(\vec{L}) = \emptyset$". By the variable convention this is the case.

One easily verifies that this definition is equivalent to the following one, linking the definition above to the tree formalism described in Section 2.

ALTERNATIVE DEFINITION 3.9. SUBSTITUTION

$$M[\vec{N}/\vec{x}] \equiv M[\vec{N}_1 * \cdots * \vec{N}_n/\vec{u_1} * \cdots * \vec{u_n}],$$

where $\vec{u_i} = \langle v \in \mathcal{FVO}(M) \mid \ell_M(v) \equiv x_i \rangle$, the sequence of free variable occurrences which have label x_i, $\vec{N}_i = \langle N_i, \ldots, N_i \rangle$ and $|\vec{N}_i| = |\vec{u_i}|$.

The substitution lemma for λ is easily extended to $\lambda\phi$. The easy proof is given to get acquainted with the notation.

LEMMA 3.10. SUBSTITUTION LEMMA
If $\vec{x} \cap \vec{y} = \emptyset$ and $\vec{x} \cap \mathcal{FV}(\vec{L}) = \emptyset$, then

$$M[\vec{N}/\vec{x}][\vec{L}/\vec{y}] \equiv M[\vec{L}/\vec{y}][\vec{N}[\vec{L}/\vec{y}]/\vec{x}].$$

PROOF. By induction on the structure of M.

1. M is a variable.
   (a) $M \equiv x_i$, then both sides equal $N_i[\vec{L}/\vec{y}]$.
   (b) $M \equiv y_i$, then both sides equal L_i.
   (c) $M \equiv z \notin \vec{x} \cup \vec{y}$, then both sides equal z.

2. $M \equiv (M_1 M_2)$. Then the statement follows from the induction hypothesis.

3. $M \equiv (\lambda Z.M_1)$. By the variable convention we may assume that $\vec{z} \cap (\vec{x} \cup \vec{y}) = \emptyset$ and $\vec{z} \cap (\mathcal{FV}(\vec{N}) \cup \mathcal{FV}(\vec{L})) = \emptyset$. Then by the induction hypothesis

$$(\lambda Z.M_1)[\vec{N}/\vec{x}][\vec{L}/\vec{y}] \equiv (\lambda Z.M_1[\vec{N}/\vec{x}][\vec{L}/\vec{y}]) \equiv$$
$$(\lambda Z.M_1[\vec{L}/\vec{y}][\vec{N}[\vec{L}/\vec{y}]/\vec{x}]) \equiv (\lambda Z.M_1)[\vec{L}/\vec{y}][\vec{N}[\vec{L}/\vec{y}]/\vec{x}]. \quad \Box$$

DEFINITION 3.11. CONTEXT

1. A *context* $C[\ ]$ is an $(\mathcal{L} \cup \{\Box\})$-labelled binary tree M, where $\varrho(\Box) = 0$ and $\Box$ does not occur in the pattern of any subterm of M. More informally it is a term with some holes ("no-name" variables) in it.

2. If $C[\ ]$ is a context and $M \in \Lambda\Phi$, then $C[M]$ denotes the result of placing M in the holes of $C[\ ]$. In this act free variables of M may become bound in $C[M]$. Formally

$$C[M] \equiv C[\ ][\vec{M}/\vec{u}],$$

where $\langle u_1, \ldots, u_n \rangle = \mathcal{O}_{\Box}(M)$ and $\vec{M} = \langle M, \ldots, M \rangle$.

4. The Church-Rosser Property for $\lambda\phi$

To prove that a theory (in our case $\boldsymbol{\lambda\phi}$) is consistent it is sufficient to show that there are terms which are not convertible. The standard way, for λ related systems anyway, to prove consistency consists of three parts. First show that the convertibility relation is a subset of the convertibility relation $=_R$ induced by some binary relation $\boldsymbol{R}$. Then show that the reduction relation $\twoheadrightarrow_R$ induced by $\boldsymbol{R}$ has the Church-Rosser property, that is, every two R-convertible terms have a common R-reduct. Finally pick two distinct terms which are R-normal forms. Then we are done because these two terms have no common R-reduct and hence are not R-convertible and so not convertible either.

First we formalise the terminology. It is the terminology from [Bar84], extended to handle our case.

DEFINITION 4.1. RELATIONS ON TERMS
Let $\boldsymbol{R}$ be a binary relation on $\Lambda\Phi$.

1. $\boldsymbol{R}$ is *compatible* if $\forall M, N, L \in \Lambda\Phi$ and $X \in \Phi$, we have that $(M, N) \in \boldsymbol{R}$ implies

$$((LM), (LN)) \in \boldsymbol{R},$$
$$((ML), (NL)) \in \boldsymbol{R} \text{ and}$$
$$((\lambda X.M), (\lambda X.N)) \in \boldsymbol{R}.$$

2. $\boldsymbol{R}$ is a *reduction relation* if it is compatible, reflexive and transitive.

3. $\boldsymbol{R}$ is an *equality relation* if it is a symmetric reduction relation.

4. $\boldsymbol{R}$ is *substitutive* if $\forall M, N, \vec{L} \in \Lambda\Phi$ and all $\vec{x}$ one has

$$(M, N) \in \boldsymbol{R} \;\Rightarrow\; (M[\vec{L}/\vec{x}], N[\vec{L}/\vec{x}]) \in \boldsymbol{R}$$

The next binary relation will turn out to satisfy the above stated properties.

DEFINITION 4.2. The binary relation $\boldsymbol{\beta}_{\Lambda\Phi}$ is defined by the rule

$$\boldsymbol{\beta}_{\Lambda\Phi} : ((\lambda X.M)X[\vec{N}/\vec{x}]) \to M[\vec{N}/\vec{x}]$$

In most cases we leave out the subscript $\Lambda\Phi$ from $\boldsymbol{\beta}$.

DEFINITION 4.3. REDUCTION
Let $\boldsymbol{R}$ be a binary relation on $\Lambda\Phi$. $\boldsymbol{R}$ induces the binary relations

$\to_R$ *one step R-reduction*,

$\twoheadrightarrow_R$ *R-reduction*, and

$=_R$ *R-convertibility*,

inductively defined as follows.
First, $\to_R$ is the compatible closure of $\boldsymbol{R}$. That is

1. $(M, N) \in \boldsymbol{R} \;\Rightarrow\; M \to_R N$,

2. $M \to_R N \;\Rightarrow\; (LM) \to_R (LN)$,

3. $M \to_R N \;\Rightarrow\; (ML) \to_R (NL)$,

4. $M \to_R N \;\Rightarrow\; (\lambda X.M) \to_R (\lambda X.N)$.

Next, $\twoheadrightarrow_R$ is the reflexive, transitive closure of $\to_R$. That is

1. $M \to_R N \;\Rightarrow\; M \twoheadrightarrow_R N$,

2. $M \twoheadrightarrow_R M$,

3. $M \twoheadrightarrow_R N,\ N \twoheadrightarrow_R L \;\Rightarrow\; M \twoheadrightarrow_R L$.

Finally, $=_R$ is the equivalence relation generated by $\to_R$. That is

1. $M \twoheadrightarrow_R N \;\Rightarrow\; M =_R N$,

2. $M =_R N \;\Rightarrow\; N =_R M$,

3. $M =_R N,\ N =_R L \;\Rightarrow\; M =_R L$.

We leave out the subscript β from the relations induced by $\boldsymbol{\beta}$ when no confusion arises. Some of the proofs of the following propositions are routine and therefore omitted.

PROPOSITION 4.4. *The relation $\to_R$ is compatible, $\twoheadrightarrow_R$ is a reduction relation and $=_R$ is an equality relation.*

PROPOSITION 4.5. *If $\boldsymbol{R}$ is substitutive, then so are $\rightarrow_R$, $\twoheadrightarrow_R$ and $=_R$.*

PROPOSITION 4.6. *$\boldsymbol{\beta}$ is substitutive.*

PROOF. Let $(M, N) \in \boldsymbol{\beta}$. Then $M \equiv ((\lambda Y.P)Y[\vec{Q}/\vec{y}])$ and $N \equiv P[\vec{Q}/\vec{y}]$. Hence

$M[\vec{L}/\vec{x}] \equiv ((\lambda Y.P[\vec{L}/\vec{x}])Y[\vec{Q}[\vec{L}/\vec{x}]/\vec{y}])$,

$N[\vec{L}/\vec{x}] \equiv P[\vec{Q}/\vec{y}][\vec{L}/\vec{x}] \equiv P[\vec{L}/\vec{x}][\vec{Q}[\vec{L}/\vec{x}]/\vec{y}]$,

by the Substitution Lemma 3.10 and the Variable Convention, so $(M[\vec{L}/\vec{x}], N[\vec{L}/\vec{x}]) \in \boldsymbol{\beta}$. □

PROPOSITION 4.7. MONOTONICITY AND SUBSTITUTIVITY

1. $\vec{N} \twoheadrightarrow_R \vec{N'} \;\Rightarrow\; M[\vec{N}/\vec{x}] \twoheadrightarrow_R M[\vec{N'}/\vec{x}]$.
2. $M \twoheadrightarrow M' \;\Rightarrow\; M[\vec{N}/\vec{x}] \twoheadrightarrow M'[\vec{N}/\vec{x}]$.
3. $M \twoheadrightarrow M' \;\&\; \vec{N} \twoheadrightarrow \vec{N'} \;\Rightarrow\; M[\vec{N}/\vec{x}] \twoheadrightarrow M'[\vec{N'}/\vec{x}]$.

PROOF.

1. By induction on the structure of M.
   (a) If $M \in \mathcal{V}$ then we can distinguish two cases :
      i. $M \equiv x_i \in \vec{x} \;\Rightarrow\; x_i[\vec{N}/\vec{x}] \equiv N_i \twoheadrightarrow_R N_i' \equiv x_i[\vec{N'}/\vec{x}]$
      ii. $M \equiv y \notin \vec{x} \;\Rightarrow\; y[\vec{N}/\vec{x}] \equiv y \twoheadrightarrow_R y \equiv y[\vec{N'}/\vec{x}]$
   (b) If $M \equiv (M_1 M_2)$ then
   $(M_1 M_2)[\vec{N}/\vec{x}] \equiv (M_1[\vec{N}/\vec{x}]M_2[\vec{N}/\vec{x}]) \twoheadrightarrow_R (M_1[\vec{N'}/\vec{x}]M_2[\vec{N'}/\vec{x}]) \equiv (M_1 M_2)[\vec{N'}/\vec{x}]$
   (c) If $M \equiv (\lambda Y.M_1)$ then
   $(\lambda Y.M_1)[\vec{N}/\vec{x}] \equiv (\lambda Y.M_1[\vec{N}/\vec{x}]) \twoheadrightarrow_R (\lambda Y.M_1[\vec{N'}/\vec{x}]) \equiv (\lambda Y.M_1)[\vec{N'}/\vec{x}]$
2. By Proposition 4.5 and Proposition 4.6.
3. By 1 and 2. □

With the properties above it is not difficult to show that $=_{\beta_{\Lambda\Phi}}$ contains the convertibility relation of $\boldsymbol{\lambda\phi}$.

PROPOSITION 4.8. $\forall M, N \in \Lambda\Phi$

$$\boldsymbol{\lambda\phi} \;\vdash\; M = N \;\Rightarrow\; M =_{\beta_{\Lambda\Phi}} N$$

PROOF. Straightforward. (Note that we also have the other direction.) □

4.1. Confluence for $\beta_{\Lambda\Phi}$

In order to show that $\boldsymbol{\beta}_{\Lambda\Phi}$ has the Church-Rosser property—our second obligation for proving consistency—we have to do more work. In this subsection, we will try to adapt the proof strategy of P. Martin-Löf and W. Tait for proving the Church-Rosser property, to our system. It turns out that the Church-Rosser property does not hold for arbitrary Φ. Therefore we require Φ to satisfy the rigid pattern condition. We prove that under this condition $\lambda\phi$ indeed has the Church-Rosser property. An easy consequence is then that $\lambda\phi$ is a conservative extension of λ. We introduce a set Π (properly containing $\mathcal{V}$) which satisfies the rigid pattern condition. But first, again, the standard terminology.

DEFINITION 4.9. R-REDEXES
Let $\boldsymbol{R}$ be a binary relation on $\Lambda\Phi$. Let $M \in \Lambda\Phi$.

1. M is an *R-redex* if $(M, N) \in \boldsymbol{R}$ for some term N. In this case N is called an *R-contractum* of M.
2. $\mathcal{RO}_R(M) = \{u \in M \mid u\backslash M \text{ is an } R\text{-redex}\}$ is the set of all R-redex occurrences of M.
3. A term M is called an *R-normal form* (*R-nf*) if $\mathcal{RO}_R(M) = \emptyset$. The set of all R-nfs is denoted by $\mathrm{NF}(R)$.
4. A term N *is an R-nf of* M (or M *has the R-nf* N) if N is an R-nf and $M =_R N$.

We will need to be able to pinpoint 'where the action takes place' in a reduction. Therefore the following definition is useful.

DEFINITION 4.10. REDUCTION PATH
Let $\boldsymbol{R}$ be a binary relation on $\Lambda\Phi$.

1. For all $M, N \in \Lambda\Phi$, we write $M \xrightarrow{u}_R N$ if $(u\backslash M, u\backslash N) \in \boldsymbol{R}$ and $N \equiv M[(u\backslash N)/u]$.
2. An *R-reduction* (*path*) is a finite or infinite sequence

$$M_0 \xrightarrow{u_0}_R M_1 \xrightarrow{u_1}_R M_2 \xrightarrow{u_2}_R \cdots.$$

PROPOSITION 4.11.
The following are equivalent :

1. $M \to_R N$,
2. $M \xrightarrow{u}_R N$ *for some* $u \in \mathcal{RO}_R(M)$ *(not necessarily unique), and*
3. $M \equiv C[P]$, $N \equiv C[Q]$ *for some* $(P, Q) \in \boldsymbol{R}$ *and* $C[\ ] \in \Lambda\Phi$ *with* $\mathcal{O}_{\Box}(C[\ ]) = \{u\}$.

Note that if $M \in \mathrm{NF}(R)$, then for no N one has $M \to_R N$.

CONVENTIONS

1. σ, τ, ... range over reduction paths.

2. The reduction path σ in Definition 4.10 *starts* with M_0. If there is a last term M_n in σ, then σ *ends* with M_n. In that case one also says that σ is a reduction path *from* M_0 *to* M_n with length n. We also write $\sigma : M_0 \twoheadrightarrow_R M_n$ to specify σ.

3. Sometimes the u_0, u_1, ... are left out in denoting a reduction path.

4. We often write $\sigma : M_0 \to M_1 \to \cdots$ to indicate that σ *is* the path $M_0 \to M_1 \to \cdots$.

5. If $\sigma : M_0 \to \cdots \to M_n$ and $\tau : M_n \to \cdots \to M_m$, then

$$\sigma * \tau : M_0 \to \cdots \to M_n \to \cdots \to M_m \ .$$

6. The one step reduction $M \overset{u}{\to}_R N$ is denoted by (u).

7. If σ is an R-reduction path, then $\|\sigma\|$ is its *length*, i.e. the number of $\to_R$ steps in it. Note that $\|\sigma\| \in \mathbb{N} \cup \{\infty\}$.

DEFINITION 4.12. CR AND UN
Let $\boldsymbol{R}$ be a binary relation on $\Lambda\Phi$.

1. $\boldsymbol{R}$ *is CR* (or *has the Church-Rosser property*) if $=_R \subseteq \twoheadrightarrow_R \cdot \twoheadleftarrow_R$; whenever two terms are R-convertible they have a common R-reduct.

2. $\boldsymbol{R}$ *is UN* (or *has unique normal forms*) if $=_R \subseteq \equiv$ on $\mathrm{NF}(R)$; whenever two normal forms are R-convertible they are equivalent.

3. $\boldsymbol{R}$ *is confluent* if $\twoheadleftarrow_R \cdot \twoheadrightarrow_R \subseteq \twoheadrightarrow_R \cdot \twoheadleftarrow_R$; whenever we can do two diverging reductions we can find converging reductions. This can also be stated as $\twoheadrightarrow_R \models \diamond$ (see Appendix A).

4. $\boldsymbol{R}$ *is locally confluent* if $\leftarrow_R \cdot \to_R \subseteq \twoheadrightarrow_R \cdot \twoheadleftarrow_R$; whenever we can do two diverging reduction steps we can find converging reductions.

It is well known (and easy to show) that CR implies UN, confluence implies local confluence and that confluence is equivalent to CR. Because of this last fact and because local confluence does not imply confluence, local confluence is also called *weak Church-Rosser* or WCR for short.

By the observations just made it is sufficient to prove $\twoheadrightarrow \models \diamond$. The following lemma comes in handy. (Here '+' denotes the transitive closure; see Appendix A.)

LEMMA 4.13. *Let $\rightarrowtail$ be a binary relation. Then*

$$\rightarrowtail \models \diamond \Rightarrow \rightarrowtail^+ \models \diamond$$

Now if we can find a relation $\rightarrowtail$ such that $\overset{\equiv}{\to} \subseteq \rightarrowtail \subseteq \twoheadrightarrow$ and $\rightarrowtail \models \diamond$ then we are done because $\rightarrowtail^+ \models \diamond$ by the previous lemma, $\twoheadrightarrow = \overset{\equiv}{\to}^+ \subseteq \rightarrowtail^+ \subseteq \twoheadrightarrow^+ = \twoheadrightarrow$, and hence $\twoheadrightarrow \models \diamond$ as required for proving confluence.

DEFINITION 4.14. CONCURRENT REDUCTION
Define a binary relation $\underset{1}{\twoheadrightarrow}$ on $\Lambda\Phi$ inductively as follows:

1. $M \underset{1}{\twoheadrightarrow} M$;
2. $M \underset{1}{\twoheadrightarrow} M'$, $N \underset{1}{\twoheadrightarrow} N' \;\Rightarrow\; (MN) \underset{1}{\twoheadrightarrow} (M'N')$;
3. $M \underset{1}{\twoheadrightarrow} M' \;\Rightarrow\; (\lambda X.M) \underset{1}{\twoheadrightarrow} (\lambda X.M')$;
4. $M \underset{1}{\twoheadrightarrow} M'$, $\vec{N} \underset{1}{\twoheadrightarrow} \vec{N'} \;\Rightarrow\; ((\lambda X.M)X[\vec{N}/\vec{x}]) \underset{1}{\twoheadrightarrow} M'[\vec{N'}/\vec{x}]$

The idea of concurrent reduction is to reduce an arbitrary number of redexes present in a term concurrently. Two extreme case are reduction of no redex at all ('empty reduction'), and reduction of all redexes present ('full substitution').

We first show that $\underset{1}{\twoheadrightarrow}$ is a possible candidate and prove some simple properties. Some routine proofs are again omitted.

LEMMA 4.15. $\overset{\equiv}{\rightarrow} \subseteq \underset{1}{\twoheadrightarrow} \subseteq \twoheadrightarrow$

LEMMA 4.16. *If* $M \underset{1}{\twoheadrightarrow} M'$ *and* $\vec{N} \underset{1}{\twoheadrightarrow} \vec{N'}$, *then* $M[\vec{N}/\vec{x}] \underset{1}{\twoheadrightarrow} M'[\vec{N'}/\vec{x}]$

PROOF. Straightforward (or see Appendix B). □

PROPOSITION 4.17.

1. $(\lambda X.M) \underset{1}{\twoheadrightarrow} N$ *implies* $N \equiv (\lambda X.M')$ *with* $M \underset{1}{\twoheadrightarrow} M'$.
2. $(MN) \underset{1}{\twoheadrightarrow} L$ *implies either*
   $L \equiv (M'N')$ *with* $M \underset{1}{\twoheadrightarrow} M'$, $N \underset{1}{\twoheadrightarrow} N'$, *or*
   $L \equiv P'[\vec{N'}/\vec{x}]$, $M \equiv (\lambda X.P)$, $N \equiv X[\vec{N}/\vec{x}]$, *and* $P \underset{1}{\twoheadrightarrow} P'$, $\vec{N} \underset{1}{\twoheadrightarrow} \vec{N'}$.

Note that $\underset{1}{\twoheadrightarrow}$ does not satisfy the diamond property for arbitrary Φ, as the following example shows.

EXAMPLE 4.18.

1. Consider the term $M_1 \equiv ((\lambda(\mathbf{I}y).z)(\mathbf{I}y))$. Then both $M_1 \underset{1}{\twoheadrightarrow} z$ and $M_1 \underset{1}{\twoheadrightarrow} ((\lambda(\mathbf{I}y).z)y)$. Now z and $((\lambda(\mathbf{I}y).z)y)$ will $\underset{1}{\twoheadrightarrow}$-reduce to themselves only and therefore have no common reduct. The problem is that the pattern of M_1 is not in normal form, so a term which matches with it, needs not match any more after doing a reduction step.
2. Consider the term $M_2 \equiv ((\lambda(xy).z)(\mathbf{I}y))$. Clearly (xy), the pattern of M_2, is in normal form, but we have the reductions $M_2 \underset{1}{\twoheadrightarrow} z$ and $M_2 \underset{1}{\twoheadrightarrow} ((\lambda(xy).z)y)$. Again the resulting terms only reduce to themselves. The problem is that the variable occurrence of x in the pattern is free and active. By instantiating it, for example by $\mathbf{I}$, part of the pattern can become reducible.

3. Finally consider the term $M_3 \equiv ((\lambda(\lambda x.((xy)y)).z)(\lambda x.((x(\mathbf{II}))(\mathbf{II}))))$. The pattern of M_3 is in normal form and has no active free variable occurrences. Nevertheless we have the reductions $M_3 \underset{1}{\twoheadrightarrow} z \equiv M'$ and $M_3 \underset{1}{\twoheadrightarrow} ((\lambda(\lambda x.((xy)y)).z)(\lambda x.((x\mathbf{I})(\mathbf{II})))) \equiv M''$. The argument of M'' does not match with its pattern, so we need more than one step to reach a common reduct. The problem now lies in the repetition of the free variable y in the pattern.

The following condition on the set of patterns turns out to be sufficient for proving that $\underset{1}{\twoheadrightarrow}$ satisfies the diamond property.

DEFINITION 4.19. RIGID PATTERN CONDITION
Let $\Phi \subseteq \Lambda\Lambda$. Φ *is* RPC (or *satisfies the rigid pattern condition*) if
$\forall X \in \Phi$ and $\forall M, \vec{Q} \in \Lambda\Phi$, $\exists \vec{Q'} \in \Lambda\Phi$ such that

$$X[\vec{Q}/\vec{x}] \underset{1}{\twoheadrightarrow} M \;\Rightarrow\; \vec{Q} \underset{1}{\twoheadrightarrow} \vec{Q'} \;\&\; M \equiv X[\vec{Q'}/\vec{x}]$$

Obviously, $\mathcal{V}$ and $\emptyset$ are RPC, but $\Lambda\Lambda$ is not. RPC is only needed in the proof of the following (crucial) lemma.

LEMMA 4.20. *Let Φ be RPC, then*

$$\underset{1}{\twoheadrightarrow} \models \diamond$$

PROOF. By induction on the definition of $M \underset{1}{\twoheadrightarrow} M'$ it will be shown that for all $M \underset{1}{\twoheadrightarrow} M''$ there is an M''' such that $M' \underset{1}{\twoheadrightarrow} M'''$, $M'' \underset{1}{\twoheadrightarrow} M'''$.

1. $M \underset{1}{\twoheadrightarrow} M'$ is $M \underset{1}{\twoheadrightarrow} M$. Then we can take $M''' \equiv M''$.

2. $M \underset{1}{\twoheadrightarrow} M'$ is $(PQ) \underset{1}{\twoheadrightarrow} (P'Q')$ and is a direct consequence of $P \underset{1}{\twoheadrightarrow} P'$, $Q \underset{1}{\twoheadrightarrow} Q'$.
   One can distinguish two subcases.
   (a) $M'' \equiv (P''Q'')$ with $P \underset{1}{\twoheadrightarrow} P''$, $Q \underset{1}{\twoheadrightarrow} Q''$. Take $M''' \equiv (P'''Q''')$ (use the i.h.).
   (b) $M'' \equiv P_1''[\vec{Q''}/\vec{x}]$, $P \equiv (\lambda X.P_1)$, $Q \equiv X[\vec{Q}/\vec{x}]$, and $P_1 \underset{1}{\twoheadrightarrow} P_1''$, $\vec{Q} \underset{1}{\twoheadrightarrow} \vec{Q''}$. By Proposition 4.17 one has $P' \equiv (\lambda X.P_1)$ with $P_1 \underset{1}{\twoheadrightarrow} P_1'$. Because Φ is RPC, we can find a $\vec{Q'}$ such that $Q' \equiv X[\vec{Q'}/\vec{x}]$ and $\vec{Q} \underset{1}{\twoheadrightarrow} \vec{Q'}$. By the induction hypothesis there is a term-sequence $\vec{Q'''}$ with $\vec{Q'} \underset{1}{\twoheadrightarrow} \vec{Q'''}$, $\vec{Q''} \underset{1}{\twoheadrightarrow} \vec{Q'''}$. By Lemma 4.16 one can take $M''' \equiv P_1'''[\vec{Q'''}/\vec{x}]$.

3. $M \underset{1}{\twoheadrightarrow} M'$ is $((\lambda X.P)X[\vec{Q}/\vec{x}]) \underset{1}{\twoheadrightarrow} P'[\vec{Q'}/\vec{x}]$ and is a consequence of $P \underset{1}{\twoheadrightarrow} P'$, $\vec{Q} \underset{1}{\twoheadrightarrow} \vec{Q'}$. By Proposition 4.17 there are again two subcases.
   (a) $M'' \equiv ((\lambda X.P'')Q'')$ with $P \underset{1}{\twoheadrightarrow} P''$, $X[\vec{Q}/\vec{x}] \underset{1}{\twoheadrightarrow} Q''$. Analogous to case 2b.
   (b) $M'' \equiv P''[\vec{Q''}/\vec{x}]$ with $P \underset{1}{\twoheadrightarrow} P''$, $\vec{Q} \underset{1}{\twoheadrightarrow} \vec{Q''}$. Take $M''' \equiv P'''[\vec{Q'''}/\vec{x}]$.

4. $M \underset{1}{\twoheadrightarrow} M'$ is $(\lambda X.P) \underset{1}{\twoheadrightarrow} (\lambda X.P')$ and is a direct consequence of $P \underset{1}{\twoheadrightarrow} P'$. Then $M'' \equiv (\lambda X.P'')$. By the induction hypothesis one can take $M''' \equiv (\lambda X.P''')$. □

THEOREM 4.21. *$\boldsymbol{\beta}_{\Lambda\Phi}$ is CR.*

PROOF. By Lemmas 4.13, 4.15 and 4.20. □

As a consequence $\boldsymbol{\beta}_{\Lambda\Phi}$ is UN, so a term can have at most one normal form.

COROLLARY 4.22.

1. *$\boldsymbol{\lambda\phi}$ is a conservative extension of $\boldsymbol{\lambda}$.*
2. *$\boldsymbol{\lambda\phi}$ is consistent.*

PROOF.

1. We have to prove that for all λ-terms M and N

$$\boldsymbol{\lambda\phi} \vdash M = N \Rightarrow \boldsymbol{\lambda} \vdash M = N$$

$\boldsymbol{\lambda\phi} \vdash M = N \Rightarrow M =_{\beta_{\Lambda\phi}} N \Rightarrow M \twoheadrightarrow_{\beta_{\Lambda\phi}} \cdot \twoheadleftarrow_{\beta_{\Lambda\phi}} N \Rightarrow M \twoheadrightarrow_{\beta} \cdot \twoheadleftarrow_{\beta} N \Rightarrow \boldsymbol{\lambda} \vdash M = N$ by Lemma 4.8, the previous Theorem and the observation that $\rightarrow_{\beta_{\Lambda\Phi}}$ restricted to λ-terms coincides with the one step β-reduction relation for λ.

2. We have to prove that there exist $\lambda\phi$-terms M and N such that

$$\boldsymbol{\lambda\phi} \nvdash M = N$$

By 1 we can take any two λ-terms which are not convertible in $\boldsymbol{\lambda}$, e.g. $\mathbf{I}$ and $\mathbf{K}$. □

We wanted to extend λ with patterns but is there a set which is RPC and which properly contains $\mathcal{V}$? Because of Example 4.18 the following definition seems reasonable.

DEFINITION 4.23. $\Pi = \{M \in \mathrm{NF}(\beta) \mid M$ is linear and has no active free variables$\}$

One easily verifies that Π is RPC. That the examples in the introduction can be modelled in Π is shown in the next example.

EXAMPLE 4.24.

1. Let $[x,y]$ be an abbreviation of $(\lambda z.((zx)y))$, and in general let $[M,N]$ be an abbreviation of $[x,y][\langle M, N\rangle/\langle x, y\rangle]$, then the projection functions are defined by

$$\pi_0 = (\lambda[x,y].x) \text{ and } \pi_1 = (\lambda[x,y].y).$$

2. Let $\mathbf{T} \equiv (\lambda x.(\lambda y.x))$ and $\mathbf{F} \equiv (\lambda x.(\lambda y.y))$ (true and false). Let $S(x)$ and 0 be abbreviations of $[\mathbf{F},x]$ and $[\mathbf{T},\mathbf{I}]$ respectively, then the predecessor and successor functions are defined by

$\mathrm{P}^- \equiv (\lambda S(x).x)$ and $\mathrm{S}^+ \equiv (\lambda x.S(x))$.

The test for zero can be implemented by π_0.

However, Π is not the maximal set for which the corresponding β-reduction is CR.

EXAMPLE 4.25.
Take $\Pi_{\boldsymbol{\Omega}} = \{\boldsymbol{\Omega}\} \cup \Pi$ where $\boldsymbol{\Omega} \equiv ((\lambda x.(xx))(\lambda x.(xx)))$. It is easy to see that $\Pi_{\boldsymbol{\Omega}}$ is RPC and hence that $\boldsymbol{\beta}_{\Lambda\Pi_{\boldsymbol{\Omega}}}$ satisfies the Church-Rosser property.

RPC does not prevent reductions from taking place inside patterns. However, one easily verifies that $X \to Y$ implies $Y \equiv X$. So redexes in patterns reduce in one step to themselves. For λ there is, due to the limitation of patterns being variables, only one such redex, $\boldsymbol{\Omega}$. In $\Lambda\Lambda$ one can construct others as the following example shows.

EXAMPLE 4.26.

1. Take $\omega_3 \equiv (\lambda(xx).(x(xx)))$, then

$$\boldsymbol{\Omega}_3 \equiv (\omega_3(\omega_3\omega_3)) \to (x(xx))[\omega_3/x] \equiv (\omega_3(\omega_3\omega_3))$$

2. Take $\omega_y \equiv (\lambda(x_1x_2).(x_1(x_1y)))$, then

$$\boldsymbol{\Omega}_y \equiv (\omega_y(\omega_y y)) \to (x_1(x_1y))[\langle\omega_y, y\rangle/\langle x_1, x_2\rangle] \equiv (\omega_y(\omega_y y))$$

   Note that $\boldsymbol{\Omega}_y$ contains y as free variable, and that by replacing y by an arbitrary term there are infinitely many redexes which reduce in one step to themselves.

We consider patterns reducing to themselves to be pathological cases, so we assume henceforth that Φ does not contain such patterns. We leave it as an exercise to the reader to show the way in which our notions can be extended to patterns containing redexes.

4.2. Confluence by Marking

Besides the one presented in the previous section, there is another well-known method for proving confluence. The idea is that we can keep track of what happens to redexes in a reduction. For this purpose we mark some redexes in a term and then "trace the marked redexes through a reduction path", resulting in a set of residuals in the term the reduction path ends with. Then by investigating the interaction between marked reduction and ordinary reduction we are able to prove confluence again.

We first introduce an auxiliary extension of $\Lambda\Phi$ in which lambda's can be marked.

DEFINITION 4.27. MARKED TERMS

1. $\Lambda'\Phi$ is a set of words over the alphabet $\mathcal{V} \cup \{., (,), \lambda\} \cup \{\lambda_0, \lambda_1, \lambda_2, \ldots\}$.

2. Let $\Phi \subseteq \Lambda\Lambda$ The set $\Lambda'\Phi$ of $\lambda'\phi$-terms is inductively defined as follows.

(a) $\mathcal{V} \subseteq \Lambda'\Phi$,

(b) i. $M, N \in \Lambda'\Phi \Rightarrow (MN) \in \Lambda'\Phi$,

ii. $X \in \Phi$, M, $\vec{N} \in \Lambda'\Phi \Rightarrow ((\lambda_i X.M)X[\vec{N}/\vec{x}]) \in \Lambda'\Phi$, for all $i \in \mathbb{N}$

(c) $X \in \Phi$, $M \in \Lambda'\Phi \Rightarrow (\lambda X.M) \in \Lambda'\Phi$,

Alternatively, we can view marked terms as trees with some indexed application-nodes; $\Lambda'\Phi$ is a subset of Lbintree($\mathcal{L}'$), where $\mathcal{L}' = \mathcal{L} \cup \{@_0, @_1, @_2, \ldots\}$.

Notice that, as patterns are taken from Φ they can not contain indexed lambda's. The relation β on $\Lambda\Phi$ is extended to β' on $\Lambda'\Phi$ as follows.

Definition 4.28. Marked Reduction

1. Substitution on $\Lambda'\Phi$ is defined as in Definition 3.8 with the added clause

$$((\lambda_i P.M)P[\vec{Q}/\vec{y}])[\vec{N}/\vec{x}] \equiv ((\lambda_i P.M[\vec{N}/\vec{x}])P[\vec{Q}[\vec{N}/\vec{x}]/\vec{y}]).$$

2. The binary relation $\beta'_{\Lambda'\Phi}$ on $\Lambda'\Phi$ is defined by $\beta'_{\Lambda'\Phi} = \beta_{\Lambda'\Phi} \cup \bigcup_{i\in\mathbb{N}} \beta_i$ where

$\beta_{\Lambda'\Phi} : ((\lambda X.M)X[\vec{N}/\vec{x}]) \to M[\vec{N}/\vec{x}]$, and

$\beta_i : ((\lambda_i X.M)X[\vec{N}/\vec{x}]) \to M[\vec{N}/\vec{x}]$, for all $i \in \mathbb{N}$.

3. If $(M, N) \in \beta_i$ then M is an *i-redex*. We abbreviate $\beta'_{\Lambda'\Phi}$ to β', $\mathcal{RO}_{\beta_i}(M)$ to $\mathcal{RO}_i(M)$.

4. We extend these notations to subsets I of $\mathbb{N}$, e.g. $\beta_I = \bigcup_{i\in I} \beta_i$ and $\mathcal{RO}_I(M) = \bigcup_{i\in I} \mathcal{RO}_i(M)$.

Notice that in $\Lambda'\Phi$ only redexes can be marked, so it is crucial that $\Lambda'\Phi$ is closed under reduction. By reexamining Example 4.18 and Definition 4.19 it should be clear that we again must impose the rigid pattern condition on Φ. We now examine the connections between ordinary and marked reduction.

Definition 4.29. Projection of Reduction

If $M \in \Lambda'\Phi$, then $\lfloor M \rfloor \in \Lambda\Phi$ is obtained from M by leaving out all indices. For example, $\lfloor((\lambda_1 \mathbf{I}.((\lambda_2 x.x)x))\mathbf{I})\rfloor \equiv ((\lambda \mathbf{I}.(\mathbf{I}x))\mathbf{I})$.

Lemma 4.30. Lifting and Projecting of Reductions

1. $\forall M, N \in \Lambda\Phi$ *and* $\forall M' \in \Lambda'\Phi$, $\exists N' \in \Lambda'\Phi$

$$M \xrightarrow{u}_\beta N \wedge \lfloor M' \rfloor \equiv M \Rightarrow M' \xrightarrow{u}_{\beta'} N' \wedge \lfloor N' \rfloor \equiv N$$

2. $\forall M', N' \in \Lambda'\Phi$

$$M' \to_{\beta'} N' \Rightarrow \lfloor M' \rfloor \to_\beta \lfloor N' \rfloor$$

Proof.

1. By Proposition 4.11 we have $M \equiv C[P]$, $N \equiv C[Q]$ for some $(P, Q) \equiv C[\ ] \in \Lambda\Phi$ with $\mathcal{O}_{\Box}(C[\ ]) = \{u\}$ and $(((\lambda Y.P_1)Y[\vec{Q_1}/\vec{y}]), P_1[\vec{Q_1}/\vec{y}]) \in \boldsymbol{\beta}$. Therefore $M' \equiv C'[P']$ for some $C'[\ ]$ and $P' \in \Lambda'\Phi$ such that $\lfloor C'[\ ] \rfloor \equiv C[\ ]$ and $\lfloor P' \rfloor \equiv P$. So either

   $$P' \equiv ((\lambda Y.P'_1)Y[\vec{Q'_1}/\vec{y}]), \text{ or}$$
   $$P' \equiv ((\lambda_i Y.P'_1)Y[\vec{Q'_1}/\vec{y}])$$

   and in both cases we can take $N' \equiv C'[P'_1[\vec{Q'_1}/\vec{y}]]$ to yield the desired result.

2. Similar but easier; $\lfloor \boldsymbol{\beta}' \rfloor$ 'is' $\boldsymbol{\beta}$. $\Box$

By repeated application the result above also holds for the transitive closure of the reduction relations.

Apart from just leaving out all indices, there is another natural way to do away with the indices; just reduce all the indexed redexes.

DEFINITION 4.31. FULL-SUBSTITUTION
Let $M \in \Lambda'\Phi$. Define $\psi(M) \in \Lambda\Phi$ by induction on the structure of M as follows:

1. $M \in \mathcal{V} \ \Rightarrow \ \psi(M) \equiv M$,
2. $M \equiv (PQ)$ then we distinguish two cases
   (a) $M \notin \boldsymbol{\beta}_{\mathbb{N}}$[2] $\Rightarrow \ \psi(M) \equiv (\psi(P)\psi(Q))$,
   (b) $M \in \boldsymbol{\beta}_{\mathbb{N}}$, $M \equiv ((\lambda_i Y.P')Y[\vec{Q'}/\vec{y}]) \ \Rightarrow \ \psi(M) \equiv \psi(P')[\psi(\vec{Q'})/\vec{y}]$,
3. $M \equiv (\lambda Y.P) \ \Rightarrow \ \psi(M) \equiv (\lambda Y.\psi(P))$.

In other words, ψ contracts all the redexes with an index in a term from the inside to the outside. How this operation interacts with marked reduction is stated in the following lemmas which should be intuitively clear.

LEMMA 4.32. *1. Let M, $\vec{N} \in \Lambda'\Phi$, then $\psi(M[\vec{N}/\vec{x}]) \equiv \psi(M)[\psi(\vec{N})/\vec{x}]$,*

2. $\forall M, N \in \Lambda'\Phi$, then $M \twoheadrightarrow_{\beta'} N \ \Rightarrow \ \psi(M) \twoheadrightarrow_{\beta} \psi(N)$.

PROOF. See Appendix B. $\Box$

LEMMA 4.33. $\forall M \in \Lambda'\Phi \ \lfloor M \rfloor \twoheadrightarrow_{\beta} \psi(M)$.

PROOF. By a standard induction on the structure of M, or see Appendix B. $\Box$

LEMMA 4.34. STRIP LEMMA
$\forall M, M'$ *and* $N \in \Lambda\Phi$, $\exists N \in \Lambda\Phi$ *such that* $M \rightarrow_{\beta} M' \wedge M \twoheadrightarrow_{\beta} N \ \Rightarrow \ N \twoheadrightarrow_{\beta} N' \wedge M' \twoheadrightarrow_{\beta} N'$.

[2] By abuse of notation we write $M \in \boldsymbol{\beta}_{\mathbb{N}}$ for $\exists N[(M, N) \in \boldsymbol{\beta}_{\mathbb{N}}]$

PROOF. Let $M \xrightarrow{u}_{\beta} M'$ and let $\tilde{M} \in \Lambda'\Phi$ be obtained from M by indexing the lambda at occurrence u. Then obviously $\lfloor \tilde{M} \rfloor \equiv M$ and $\psi(\tilde{M}) \equiv M'$. By Lemma 4.30(1) $\exists \tilde{N}$ such that $\tilde{M} \twoheadrightarrow_{\beta'} \tilde{N}$ and $\lfloor \tilde{N} \rfloor \equiv N$. Now if we choose $N' \equiv \psi(\tilde{N})$ then we have $M' \equiv \psi(\tilde{M}) \twoheadrightarrow_{\beta} \psi(\tilde{N}) \equiv N'$ by Lemma 4.32(2), and $N \equiv \lfloor \tilde{N} \rfloor \twoheadrightarrow_{\beta} \psi(\tilde{N}) \equiv N'$ by Lemma 4.33 as required. □

PROPOSITION 4.35. $\twoheadrightarrow_{\beta} \models \diamond$

PROOF. By the Strip Lemma 4.34 and a simple diagram chase. □

So both of the methods presented in this and the previous section are easily extended from λ to handle $\lambda\pi$. Once CR is proved, the obvious question is to ask whether also the other standard result from λ, the Finite Developments theorem (FD), holds for $\lambda\phi$. This question is investigated in the next section.

5. The Finiteness of Developments

The main theorem in this section states that for each $M \in \Lambda\Phi$ the so called developments (a special kind of reduction starting with M) are always finite. A more succinct formulation is

$$\mathrm{SN}(\boldsymbol{\beta}_{\mathrm{IN}})$$

that is, reductions on $\Lambda'\Phi$ contracting only indexed redexes are always finite. As is the case for λ, confluence of β can be derived from this theorem. For the proof we extend the terminology for marked reduction given in the previous subsection following [Bar84].

DEFINITION 5.1. PROJECTION OF REDUCTION PATHS
Let σ' be a $\boldsymbol{\beta}'$-reduction path starting with $M' \in \Lambda'\Phi$, say

$$\sigma' : M'_0 \xrightarrow{u_0}_{\beta'} M'_1 \xrightarrow{u_1}_{\beta'} \cdots$$

then $\lfloor \sigma' \rfloor$ is defined by

$$\lfloor \sigma' \rfloor : \lfloor M'_0 \rfloor \xrightarrow{u_0}_{\beta} \lfloor M'_1 \rfloor \xrightarrow{u_1}_{\beta} \cdots$$

LEMMA 5.2. LIFTING AND PROJECTING OF REDUCTION PATHS

1. *Let σ be a β-reduction starting with $M \in \Lambda\Phi$. Then for each $M' \in \Lambda'\Phi$ with $\lfloor M' \rfloor \equiv M$ there is a (unique) β'-reduction path σ' starting with M' such that $\lfloor \sigma' \rfloor = \sigma$.*

2. *Let σ' be a $\boldsymbol{\beta}'$-reduction path, then $\lfloor \sigma' \rfloor$ is a $\boldsymbol{\beta}$-reduction path.*

Let U be a set of redex occurrences in a $\lambda\phi$-term M. If we are interested in what happens with the elements of U during a reduction, then we can lift M to $\Lambda'\Phi$ by indexing the elements of U.

DEFINITION 5.3. MARKING TERMS
Let $U \subseteq \mathcal{RO}_\beta(M)$, $M \in \Lambda\Phi$. Then $(M,U) \in \Lambda'\Phi$ is the indexed term obtained from M by indexing the redex occurrences of M that are in U by 0. Formally (M,U) is defined by

1. $T_{(M,U)} = T_M$
2. $\ell_{(M,U)}(w) = @_0$, $\forall w \in U$,
   $\ell_{(M,U)}(w) = \ell_M(w)$, otherwise.

In this notation $M \in \Lambda\Phi$ is identified with $(M,\emptyset) \in \Lambda'\Phi$.

DEFINITION 5.4. RESIDUALS

1. Let $M,N \in \Lambda\Phi$, $U \subseteq \mathcal{RO}_\beta(M)$ and let $\sigma : M \twoheadrightarrow_\beta N$. The set of *residuals of* U *in* N *relative to* σ (notation U/σ) is defined as follows. Lift σ to $\sigma' : (M,U) \twoheadrightarrow (N,U')$, then $U/\sigma = U'$.
2. Let $M',N' \in \Lambda'\Phi$, $U \subseteq \mathcal{RO}_{\beta'}(M')$ and let $\sigma' : M' \twoheadrightarrow_{\beta'} N'$, then $U/\sigma' = U/\lfloor\sigma'\rfloor$.
3. If U is a singleton set then we leave out the set-braces and write u/σ instead of $\{u\}/\sigma$.

The next lemma states some properties of residuals. The last one states that we can 'trace' several redexes simultaneously.

LEMMA 5.5.

1. *Let* $M,N \in \Lambda\Phi$, $\sigma : M \twoheadrightarrow N$ *and* $U = \{u_1,\ldots,u_n\} \subseteq \mathcal{RO}(M)$, *then* $U/\sigma = \bigcup_{i=1}^n u_i/\sigma$.
2. *Let* $\sigma : M \twoheadrightarrow N$, $\tau : N \twoheadrightarrow L$, *and* $U \subseteq \mathcal{RO}(M)$, *then* $U/\sigma * \tau = (U/\sigma)/\tau$.
3. *Let* $\sigma' : M' \twoheadrightarrow_{\beta'} N'$, M',$N' \in \Lambda'\Phi$, *then* $\mathcal{RO}_i(M')/\lfloor\sigma'\rfloor = \mathcal{RO}_i(N')$.

COROLLARY 5.6. *Let* $\sigma : M \twoheadrightarrow N$, M, $N \in \Lambda\Phi$ *and let* $u \neq v \in M$. *Then* $u/\sigma \cap v/\sigma = \emptyset$.

Now a special kind of reduction paths, the so called developments are defined. In a development only residuals of redexes of the term the reduction path starts with, may be contracted. Or equivalently, in a development no newly created redexes may be contracted.

DEFINITION 5.7. DEVELOPMENTS

1. Let $M \in \Lambda\Phi$ and $U \subseteq \mathcal{RO}(M)$. A *development of* (M,U) is a reduction path
$$\sigma : M \equiv M_0 \stackrel{u_0}{\to} M_1 \stackrel{u_1}{\to} \cdots$$
such that each redex occurrence $u_i \in \mathcal{RO}(M_i)$ is a residual of a redex in U relative to $(u_0) * \cdots * (u_{i-1})$.
2. $\sigma : M \twoheadrightarrow N$ is a *complete development of* (M,U), notation $\sigma : (M,U) \underset{\text{cpl}}{\twoheadrightarrow} N$, if σ is a development of (M,U) and moreover $U/\sigma = \emptyset$.

3. A *development of* M is a development of $(M, \mathcal{RO}(M))$.

4. $M \underset{\text{dev}}{\twoheadrightarrow} N$ iff N occurs in some development of M.

Developments and $\beta_{\mathbb{N}}$-reductions correspond to each other in the following way.

PROPOSITION 5.8. *σ is a development of (M, U) iff σ lifted to σ' starting with (M, U) is a $\beta_{\mathbb{N}}$-reduction.*

In the next section we deviate from the line followed in [Bar84].

5.1. SN($\beta_{\mathbb{N}}$)

The Finite Developments Theorem states that all developments are finite. According to the previous lemma this is equivalent to stating that all $\beta_{\mathbb{N}}$-reductions are finite. We prove this by a proof strategy which strongly resembles the one given in [dV87]. We establish an exact upper bound on the length of the maximal development (maximal with respect to its length) of a $\lambda\phi$-term. The idea of proof is to assign to each $M \in \Lambda'\Phi$ a special norm (integer) such that

$$M \to_{\beta_{\mathbb{N}}} N \;\Rightarrow\; \text{the norm of } M \text{ is strictly greater than the norm of } N.$$

DEFINITION 5.9. STRONG NORMALISATION
Let $\boldsymbol{R}$ be a binary relation on $\Lambda'\Phi$.

1. A term $M \in \Lambda'\Phi$, *R-strongly normalises* if there is no infinite R-reduction starting with M.

2. $\boldsymbol{R}$ is *strongly normalising* (notation SN($\boldsymbol{R}$)) if every $M \in \Lambda'\Phi$ R-strongly normalises.

DEFINITION 5.10. WEIGHT OF A TERM

1. For all *weighings*, $\rho : \mathcal{V} \to \mathbb{N}$, we define the *$\rho$-norm of* $M \in \Lambda'\Phi$ by induction on the structure of M as follows.

   (a) $M \equiv x \in \mathcal{V} \;\Rightarrow\; \|M\|_\rho = \rho(x)$

   (b) $M \equiv (M_1 M_2)$, then there are two cases to consider :

   $M \notin \boldsymbol{\beta}_{\mathbb{N}} \;\Rightarrow\; \|M\|_\rho = \|M_1\|_\rho + \|M_2\|_\rho$

   $M \in \boldsymbol{\beta}_{\mathbb{N}}$ so $M_1 \equiv (\lambda_i Y.P)$, $M_2 \equiv Y[\vec{Q}/\vec{y}] \;\Rightarrow$
   $\|M\|_\rho = 1 + \|P\|_{\rho[\vec{Q}/\vec{y}]} + \Sigma_{y_i \in \vec{y} - \vec{p}} \|Q_i\|_\rho$

   where $\rho[\vec{Q}/\vec{y}] = \rho[\|\vec{Q}\|_\rho/\vec{y}]$ is defined by

   $\rho[\vec{Q}/\vec{y}](x) = \|Q_i\|_\rho$ for all $x \equiv y_i \in \vec{y}$
   $\rho[\vec{Q}/\vec{y}](x) = \|x\|_\rho$ otherwise.

   (c) $M \equiv (\lambda X.M_1) \;\Rightarrow\; \|M\|_\rho = \|M_1\|_{\rho[\vec{0}/\vec{x}]}$, where $\vec{0} = \langle 0, \ldots, 0 \rangle$ has the same length as $\vec{x}$.

2. We assume a pointwise ordering on weighings.

3. $\|M\|$, the *norm of* M is the c_0-norm of M, where c_0 is the constant zero function.

The reader should verify that this norm represents the intuition of the maximal number of steps of a development. In the example below some simple situations are being investigated.

EXAMPLE 5.11.

1. The norm of variables is zero, as we expect.

$$\|x\| = \|x\|_{c_0} = c_0(x) = 0$$

2. The norm of an 'inactive application' is the sum of the norms of the parts.

$$\|(M_1 M_2)\| = \|(M_1 M_2)\|_{c_0} = \|M_1\|_{c_0} + \|M_2\|_{c_0} = \|M_1\| + \|M_2\|$$

3. The norm of a 'redex application' consists of three parts :
   (a) one for actually reducing the redex,
   (b) the norm of the body with pattern-variables weighed by the norms of the terms bound to them in the argument, and
   (c) the sum of the norms of the terms bound to the pattern-variables in the argument which have no free occurrence in the body.

$$\|((\lambda_0[y_0,y_1].(y_0 Q_2))[Q_0,Q_1])\| =$$
$$1 + \|(y_0 Q_2)\|_{c_0[\langle Q_0,Q_1\rangle/\langle y_0,y_1\rangle]} + \textstyle\sum_{y_i \in \langle y_0,y_1\rangle - \langle y_0\rangle * \vec{q_2}} \|Q_i\|_{c_0} =$$
$$1 + \|Q_0\|_{c_0} + \|Q_2\|_{c_0} + \|Q_1\|_{c_0}$$

4. The norm of an abstraction is the norm of its body with pattern-variables having weight zero.

$$\|(\lambda X.M)\| = \|M\|_{c_0[\vec{0}/\vec{x}]} = \|M\|_{c_0} = \|M\|$$

LEMMA 5.12. NORM PROPERTIES

1. Using the notation of page 7, we have :

$$\|P(\vec{y} - \vec{p})\|_{\rho[\vec{Q}/\vec{y}]} = \|P\|_{\rho[\vec{Q}/\vec{y}]} + \sum_{y_i \in \vec{y} - \vec{p}} \|Q_i\|_{\rho}$$

2. Finiteness and Monotonicity
For all weighings ρ_1, ρ_2, *for all terms* $M \in \Lambda'\Phi$ *and for every relation* $\square \in \{=, >\}$

$$\rho_1 \restriction \mathcal{FV}(M) \square \rho_2 \restriction \mathcal{FV}(M) \;\Rightarrow\; \|M\|_{\rho_1} \square \|M\|_{\rho_2}$$

3. Substitution Lemma

$$\|M\|_{\rho[\vec{N}/\vec{x}]} = \|M[\vec{N}/\vec{x}]\|_\rho$$

and therefore, if $\vec{x} \cap \vec{y} = \emptyset$ *and* $\vec{y} \cap \mathcal{FV}(\vec{N}) = \emptyset$

$$\rho[\vec{N}/\vec{x}][\vec{Q}/\vec{y}] = \rho[\vec{Q}[\vec{N}/\vec{x}]/\vec{y}][\vec{N}/\vec{x}]$$

4. Splitting
For all $\vec{x} \subseteq \mathcal{FV}(M)$

$$\|M\|_\rho \geq \|M\|_{\rho[\vec{0}/\vec{x}]} + \sum_{z \in \vec{x}} \|z\|_\rho$$

Proof.

1. Immediate from the definitions.

2. Easy induction on the structure of M.

3. Induction on the structure of M, the only difficult case being $M \equiv ((\lambda i.Y)P)Y[\vec{Q}/\vec{y}]$. By the variable convention we may assume $\vec{x} \cap \vec{y} = \vec{y} \cap \mathcal{FV}(\vec{N}) = \emptyset$. Now we have

$$\|M\|_{\rho[\vec{N}/\vec{x}]} = \|((\lambda_i Y.P)Y[\vec{Q}/\vec{y}])\|_{\rho[\vec{N}/\vec{x}]} = 1 + \|P(\vec{y} - \vec{p})\|_{\rho[\vec{N}/\vec{x}][\vec{Q}/\vec{y}]} =$$

$$1 + \|P(\vec{y} - \vec{p})\|_{\rho[\vec{Q}[\vec{N}/\vec{x}]/\vec{y}][\vec{N}/\vec{x}]} =$$

$$1 + \|P[\vec{N}/\vec{x}](\vec{y} - \vec{p})\|_{\rho[\vec{Q}[\vec{N}/\vec{x}]/\vec{y}]} = \|((\lambda_i Y.P[\vec{N}/\vec{x}])Y[\vec{Q}[\vec{N}/\vec{x}]/\vec{y}])\|_\rho = \|M[\vec{N}/\vec{x}]\|_\rho$$

by 1, the induction hypothesis and the Substitution Lemma 3.10. The second part is again a consequence of Lemma 3.10.

4. See Appendix B for the straightforward but somewhat technical proof by induction on the structure of M. □

The first norm property gives a more convenient notation for the weighing of an indexed redex as defined in Definition 5.10(1b), which we will use henceforth. The second property states that the ρ-norm of a term M only depends on the weighings of the free variables of M ($=$), and that if we increase the weighing of one of these, the ρ-norm of M also increases ($>$). The third property states that it makes no difference whether one first substitutes subterms for free variables in a term and then takes the ρ-norm or one takes the norm of the term with the variables weighed by the ρ-norms of the corresponding subterms. The last property roughly states that the weight of a term is greater than or equal to the sum of the weights of the parts. Now the technical work has been done we can prove the main lemma by a simple case analysis.

Lemma 5.13. *For all weighings* ρ

$$M \to_{\beta_{\mathbb{N}}} N \;\Rightarrow\; \|M\|_\rho > \|N\|_\rho$$

PROOF. By induction on the generation of $\rightarrow_{\beta_{\mathbb{N}}}$.

1. $M \rightarrow_{\beta_{\mathbb{N}}} N$ because $(M,N) \in \boldsymbol{\beta}_{\mathbb{N}}$, so $M \equiv ((\lambda_i Y.P)Y[\vec{Q}/\vec{y}])$, $N \equiv P[\vec{Q}/\vec{y}]$, then

   $$\|M\|_\rho = \|((\lambda_i Y.P)Y[\vec{Q}/\vec{y}])\|_\rho = 1 + \|P(\vec{y}-\vec{p})\|_{\rho[\vec{Q}/\vec{y}]} > \|P\|_{\rho[\vec{Q}/\vec{y}]} = \|P[\vec{Q}/\vec{y}]\|_\rho = \|N\|_\rho$$

   by the substitution lemma.

2. $M \rightarrow_{\beta_{\mathbb{N}}} N$ because $M \equiv (LM_2)$, $M_2 \rightarrow_{\beta_{\mathbb{N}}} N_2$ and $(LN_2) \equiv N$. We consider two cases

   (a) $M \notin \boldsymbol{\beta}_{\mathbb{N}}$. Then

   $$\|M\|_\rho = \|L\|_\rho + \|M_2\|_\rho > \|L\|_\rho + \|N_2\|_\rho = \|N\|_\rho$$

   by the induction hypothesis for $M_2 \rightarrow_{\beta_{\mathbb{N}}} N_2$.

   (b) $M \in \boldsymbol{\beta}_{\mathbb{N}}$ so $L \equiv (\lambda_i Y.P)$, $M_2 \equiv Y[\vec{Q}/\vec{y}]$, $N_2 \equiv Y[\vec{Q'}/\vec{y}]$ and $\vec{Q} \rightarrow_{\beta_{\mathbb{N}}} \vec{Q'}$, then

   $$\|M\|_\rho = \|((\lambda_i Y.P)Y[\vec{Q}/\vec{y}])\|_\rho = 1 + \|P(\vec{y}-\vec{p})\|_{\rho[\vec{Q}/\vec{y}]} >$$
   $$1 + \|P(\vec{y}-\vec{p})\|_{\rho[\vec{Q'}/\vec{y}]} = \|((\lambda_i Y.P)Y[\vec{Q'}/\vec{y}])\|_\rho = \|N\|_\rho$$

   by the induction hypothesis for $\vec{Q} \rightarrow_{\beta_{\mathbb{N}}} \vec{Q'}$ and monotonicity.

3. $M \rightarrow_{\beta_{\mathbb{N}}} N$ because $M \equiv (M_1 L)$, $M_1 \rightarrow_{\beta_{\mathbb{N}}} N_1$ and $(N_1 L) \equiv N$. Again two cases arise

   (a) $M \notin \boldsymbol{\beta}_{\mathbb{N}}$. Then

   $$\|M\|_\rho = \|M_1\|_\rho + \|L\|_\rho > \|N_1\|_\rho + \|L\|_\rho = \|N\|_\rho$$

   by the induction hypothesis for $M_1 \rightarrow_{\beta_{\mathbb{N}}} N_1$.

   (b) $M \in \boldsymbol{\beta}_{\mathbb{N}}$ so $M_1 \equiv (\lambda_i Y.P)$, $L \equiv P[\vec{Q}/\vec{y}]$, $N_1 \equiv (\lambda_i Y.P')$ and $P \rightarrow_{\beta_{\mathbb{N}}} P'$, then

   $$\|M\|_\rho = \|((\lambda_i Y.P)Y[\vec{Q}/\vec{y}])\|_\rho = 1 + \|P(\vec{y}-\vec{p})\|_{\rho[\vec{Q}/\vec{y}]} >$$
   $$1 + \|P'(\vec{y}-\vec{p'})\|_{\rho[\vec{Q}/\vec{y}]} = \|((\lambda_i Y.P')Y[\vec{Q}/\vec{y}])\|_\rho = \|N\|_\rho$$

   by splitting (for $\vec{x} = \vec{p} - \vec{p'}$), finiteness and the induction hypothesis for $P \rightarrow_{\beta_{\mathbb{N}}} P'$.

4. $M \rightarrow_{\beta_{\mathbb{N}}} N$ because $M \equiv (\lambda X.M_1)$, $M_1 \rightarrow_{\beta_{\mathbb{N}}} N_1$ and $(\lambda X.N_1) \equiv N$ then

   $$\|M\|_\rho = \|M_1\|_{\rho[\vec{0}/\vec{x}]} > \|N_1\|_{\rho[\vec{0}/\vec{x}]} = \|N\|_\rho$$

   by the induction hypothesis for $M_1 \rightarrow_{\beta_{\mathbb{N}}} N_1$. □

Observe that the norm is an exact upper bound on the length of the development of a term. If we had taken

$$\|M\|_\rho = 1 + \|P\|_{\rho[\vec{Q}/\vec{y}]} + \sum_{y_i \in \vec{y}} \|Q_i\|_\rho$$

in Definition 5.10(1b), for the ρ-norm of an indexed redex, there would have been no need for splitting in Lemma 5.13(3b), thus yielding a considerable simplification of the proof. The price to be paid is that we then no longer obtain an exact upper bound.

LEMMA 5.14. $\mathrm{SN}(\boldsymbol{\beta}_{\mathbb{N}})$ *(I.e. $\beta_{\mathbb{N}}$-reduction is strongly normalising).*

PROOF. Suppose

$$\sigma : M_0 \to_{\beta_{\mathbb{N}}} M_1 \to_{\beta_{\mathbb{N}}} M_2 \to_{\beta_{\mathbb{N}}} \cdots$$

is a $\beta_{\mathbb{N}}$-reduction starting with $M_0 \in \Lambda'\Phi$. Then by the previous lemma

$$\|M_0\| > \|M_1\| > \|M_2\| > \cdots$$

and because IN is well-founded this sequence and hence σ must be finite. □

THEOREM 5.15. FINITENESS OF DEVELOPMENTS (FD)
Let $M \in \Lambda\Phi$. Then all developments of M are finite.

PROOF. By Lemma 5.14 and Lemma 5.8. □

COROLLARY 5.16. *Let $M \in \Lambda\Phi$*

1. *For $U \subseteq \mathcal{RO}(M)$ each development of (M, U) can be extended to a complete one.*
2. *The set $\{N \mid M \underset{\text{dev}}{\twoheadrightarrow} N\}$ is finite.*

PROOF.

1. Immediate by FD.
2. By FD and Königs lemma. □

5.2. Confluence by Finite Developments

Now it will be proved that all complete developments of an (M, U) terminate with the same result. For this purpose it is enough to show that $\boldsymbol{\beta}_{\mathbb{N}}$ is CR, because CR implies UN. That, in the presence of $\mathrm{SN}(\boldsymbol{\beta}_{\mathbb{N}})$, it is sufficient to prove WCR is stated in the next lemma.

LEMMA 5.17. (SPECIALISED VERSION OF) NEWMAN'S LEMMA [New42]
$\mathrm{SN}(\boldsymbol{\beta}_{\mathbb{N}})$ & $\boldsymbol{\beta}_{\mathbb{N}}$ *is WCR* $\Rightarrow$ $\boldsymbol{\beta}_{\mathbb{N}}$ *is CR.*

In the next lemma, $\to$ will stand for $\to_{\beta_{\mathbb{N}}}$.

LEMMA 5.18. *The relation $\to$ satisfies the weak diamond property.*

PROOF. Suppose $M \to M_1$, $M \to M_2$ in order to construct an M_3 such that $M_1 \twoheadrightarrow M_3$ and $M_2 \twoheadrightarrow M_3$. Let $(u_i) : M \to M_i$, $i = 1, 2$, with $\Delta_i \equiv u_i \backslash M \equiv ((\lambda_{k_i} X_i.P_i)Q_i)$, $Q_i \equiv X_i[\vec{Q_i}/\vec{x_i}]$, and let $\Delta_i' \equiv P_i[\vec{Q_i}/\vec{x_i}]$. We give a case analysis by considering the relative positions of u_1 and u_2.

1. $u_1 \perp u_2$, so Δ_1 and Δ_2 are disjoint. Then

   $$M_1 \equiv \cdots \Delta_1' \cdots \Delta_2 \cdots$$

$M_2 \equiv \cdots \Delta_1 \cdots \Delta_2' \cdots$

and we can take

$M_3 \equiv \cdots \Delta_1' \cdots \Delta_2' \cdots$

2. $u_1 = u_2$, so Δ_1 and Δ_2 are identical. Then we can take $M_1 \equiv M_3 \equiv M_2$.

3. $u_2 \lhd u_1$, so Δ_1 is a proper subterm of Δ_2. Depending on whether Δ_1 is in the body or in the argument of Δ_2, we distinguish two cases.

   (a) $u_2 * \langle 0,1 \rangle \trianglelefteq u_1$, so $\Delta_1 \subseteq P_2$, Δ_1 is in the body of Δ_2.
   Then $M \equiv \cdots((\lambda_{k_2} X_2.\cdots \Delta_1 \cdots)Q_2)\cdots$, where $\cdots \Delta_1 \cdots \equiv P_2$,

   $M_1 \equiv \cdots((\lambda_{k_2} X_2.\cdots \Delta_1' \cdots)Q_2)\cdots$

   $M_2 \equiv \cdots(\cdots \Delta_1 \cdots)[\vec{Q_2}/\vec{x_2}]\cdots$

   Take

   $M_3 \equiv \cdots(\cdots \Delta_1' \cdots)[\vec{Q_2}/\vec{x_2}]\cdots$

   Then clearly $M_1 \to M_3$ and $M_2 \to M_3$ by substitutivity of $\boldsymbol{\beta}_{\mathbb{N}}$ (proved as in Proposition 4.7(2)).

   (b) $u_2 * \langle 1 \rangle \trianglelefteq u_1$, so $\Delta_1 \subseteq Q_2$, Δ_1 is in the argument of Δ_2.
   Then $M \equiv \cdots((\lambda_{k_2} X_2.P_2)\cdots \Delta_1 \cdots)\cdots$, where $\cdots \Delta_1 \cdots \equiv Q_2$,

   $M_1 \equiv \cdots((\lambda_{k_2} X_2.P_2)\cdots \Delta_1' \cdots)\cdots$

   $M_2 \equiv \cdots P_2[\vec{Q_2}/\vec{x_2}]\cdots$

   By RPC, we can find a term $Q' \equiv X_2[\vec{Q'}/\vec{x_2}]$ such that $Q_2 \twoheadrightarrow Q'$ and $\cdots \Delta_1' \cdots \equiv X_2[\vec{Q'}/\vec{x_2}]$. Then clearly $M_1 \equiv \cdots((\lambda_{k_2} X_2.P_2)X_2[\vec{Q'}/\vec{x_2}])\cdots \twoheadrightarrow M_3$ and $M_2 \twoheadrightarrow M_3$ by Proposition 4.7(1).

4. $u_1 \lhd u_2$. This case can be treated analogously to case 3. $\square$

By Newman's Lemma 5.17 we have that $\boldsymbol{\beta}_{\mathbb{N}}$ is CR. In the same way we can prove that $\boldsymbol{\beta}'$ is WCR. By leaving out all indices, we obtain a proof of WCR for $\boldsymbol{\beta}$.

REMARK. By examining the proof above (especially case 3b), it is clear that RPC can be weakened to the following condition to prove WCR for $\boldsymbol{\beta}$. For marked reduction this is not the case, because then it is required that the common reduct is reached by 'essentially the same reduction paths'.

DEFINITION 5.19. WCR CONDITION
Let $\Phi \subseteq \Lambda\Lambda$. Φ *is* WCR (or *satisfies the WCR condition*) if
$\forall X \in \Phi$ and $\forall M, \vec{Q} \in \Lambda\Phi$, $\exists \vec{Q'} \in \Lambda\Phi$ such that

$$X[\vec{Q}/\vec{x}] \to M \;\Rightarrow\; \vec{Q} \twoheadrightarrow \vec{Q'} \;\&\; M \twoheadrightarrow X[\vec{Q'}/\vec{x}]$$

In other words, we do not need linearity of the patterns to prove WCR for unmarked reduction. This condition is not sufficient for proving CR because one has the following counterexample based on [Klo80].

EXAMPLE 5.20. Consider $\Pi_D = \Pi \cup \{[x,x]\}$, and the following terms

$$D \equiv (\lambda[x,x].e)$$

$$C \equiv (\boldsymbol{\Theta}(\lambda c.(\lambda x.(D[x,(cx)]))))$$

$$A \equiv (\boldsymbol{\Theta} C)$$

where $\boldsymbol{\Theta} \equiv (\boldsymbol{\Theta}'\boldsymbol{\Theta}')$, (Turings fixed point combinator) and $\boldsymbol{\Theta}' \equiv (\lambda x.(\lambda y.(y((xx)y))))$. Now Π_D is WCR, but we have both

$$A \twoheadrightarrow (CA) \twoheadrightarrow (D[A,(CA)]) \twoheadrightarrow (D[(CA),(CA)]) \to e$$

and

$$A \twoheadrightarrow (CA) \twoheadrightarrow (Ce)$$

and these terms have no common reduct by arguments similar to the ones in [Klo80].

This example is a counterexample to CR but not to UN as is shown in [dV87]. It is conceivable that by a method similar to the one used there one can show UN, and therefore consistency of $\boldsymbol{\lambda\phi}$ for Φ which are WCR. We do not consider this here.

THEOREM 5.21. FD!
Let $M \in \Lambda\Phi$ and $U \subseteq \mathcal{RO}(M)$.

1. *All developments of M are finite.*
2. *All developments of (M,U) can be extended to a complete development of (M,U).*
3. *All complete developments of (M,U) end with the same term.*

PROOF.

1. By FD and its corollary.
2. By FD and its corollary.
3. By Lemma 5.18, Newman's Lemma 5.17 and Lemma 5.8. □

DEFINITION 5.22.

1. Let $M' \in \Lambda'\Phi$. Then $\mathrm{Cpl}(M')$ is the unique $\boldsymbol{\beta}_{\mathbb{N}}$-nf of M'.
2. For $M \in \Lambda\Phi$, $U \subseteq \mathcal{RO}(M)$ define $M \underset{1}{\twoheadrightarrow} \mathrm{Cpl}((M,U))$.
   (There is no ambiguity with the earlier definition of $\underset{1}{\twoheadrightarrow}$.)

It is easy to show that Lemma 4.15 holds. That $\underset{1}{\twoheadrightarrow}$ satisifies the diamond property can be shown in another way.

PROOF. (Lemma 4.20)
Define $M''' \equiv \mathrm{Cpl}((M, U_1 \cup U_2))$. Then $M' \underset{1}{\twoheadrightarrow} M'''$. $M \underset{1}{\twoheadrightarrow} M'$ results from a development of $(M, U_1 \cup U_2)$. Hence by FD! we can complete this development and this complete development ends with the term M'''. Similarly for M'' and one has the diamond property. □

6. Conclusions

Pattern matching lambda calculus, $\lambda\pi$, allows one to write functions in a way resemblant of functional programs without losing the fundamental theorems (CR and FD) from ordinary λ-calculus. But not everthing is hunky-dory. For systems which satisfy the Church-Rosser property no matter what reductions we perform, we know we can still reach the normal form (if it exists). For systems which are UN but not CR this no longer holds, as Example 5.20 shows. In this case the *reduction strategy*, i.e. the strategy determining which redex to reduce next, is obviously important. However also for Church-Rosser systems it is. If we reduce in a naive way we could get stuck in a loop, e.g. $((\mathbf{KI})\boldsymbol{\Omega}) \to ((\mathbf{KI})\boldsymbol{\Omega}) \to \ldots$, but $((\mathbf{KI})\boldsymbol{\Omega}) \to \mathbf{I}$. For λ (CR) we have the Standardization Theorem, which states that always reducing the leftmost outermost redex, i.e. the one with the least occurrence with respect to the lexicographic ordering, leads to the normal form (if one exists). For $\lambda\phi$ this is not the case due to the fact that patterns can be 'built', as the next example based on [HL79] shows.

EXAMPLE 6.1. Take the following term

$$F \equiv ((\lambda[x,\mathbf{I}].e)[\boldsymbol{\Omega},(\mathbf{II})])$$

If we use standard reduction, then we get nowhere

$$F \to F \to F \to \ldots$$

However we can reduce this term to normal form as follows

$$F \to ((\lambda[x,\mathbf{I}].e)[\boldsymbol{\Omega},\mathbf{I}]) \to e$$

This is similar to the situation in term rewriting systems investigated in [HL79]. For an overview of the theory of term rewriting systems see e.g. [Klo90].

As the examples show, $\lambda\pi$ has characteristics of both lambda calculus (evidently) and term rewriting systems (where rewriting is based on pattern matching). As shown in the examples term rewriting systems can be 'coded' in $\lambda\pi$, but at present the coding of a case analysis is a bit clumsy as the following example shows.

EXAMPLE 6.2. Consider the term rewriting system defined by the rules

$A(0, y) \to y$

$A(S(x), y) \to A(x, S(y))$

This is in fact a recursive defintion of addition :

$A((x, y)) =$ IF $x = 0$ THEN y ELSE $A((x, S(y)))$,

which can be code in $\lambda\pi$ as

$A \equiv (\boldsymbol{\Theta}(\lambda\langle a, [[c,x],y]\rangle.\textbf{IF } c = 0 \textbf{ THEN } y \textbf{ ELSE } (a[x,S(x)])))$,

where the definitions are taken from the previous examples and ordinary lambda calculus. Now

$$(A[S(0),0]) \twoheadrightarrow (A[0,S(0)]) \twoheadrightarrow S(0)$$

A. Notations

$\mathbb{N}$ is the set of natural numbers ($0 \in \mathbb{N}$).

Sets Let X, Y and Z be sets.

That X is a subset of Y will be denoted by $X \subseteq Y$. That X is a proper subset of Y will be denoted by $X \subset Y$. The union of X and Y will be denoted by $X \cup Y$. The intersection of X and Y will be denoted by $X \cap Y$. The difference of X and Y will be denoted by $X - Y$. (We use the same notations for operations on sequences.) The product of X and Y will be denoted by $X \times Y$. We inductively define X^n :

1. $X^0 = \{\emptyset\}$
2. $X^{n+1} = X^n \times X$

Functions Let A, B, C and D be sets.

$f : A \mapsto B$ denotes that f is a total function with domain A and codomain B.

$f \restriction C : A \cap C \mapsto B$, the *restriction of f to C*, is defined by $f \restriction C(c) = f(c)$.

$\mathrm{id}_A : A \mapsto A$ is the identity map on A.

Let $f : A \mapsto B$ and $g : B \mapsto C$ then $g \circ f : A \mapsto C$ is the *composition of f and g* and is defined by $g \circ f(a) = g(f(a))$.

If $f : A \mapsto B$ and $g : A \mapsto C$ then $\langle f, g\rangle : A \mapsto B \times C$ is the *pairing of f and g* and is defined by $\langle f, g\rangle(a) = (f(a), g(a))$.

If $f : A \mapsto C$ and $g : B \mapsto D$ then $f \times g : A \times C \mapsto B \times D$ is the *product of f and g* and is defined by $f \times g((a, b)) = (f(a), g(b))$.

Relations If $\rightarrowtail \subseteq X \times Y$, then $\rightarrowtail$ is a *relation on X and Y* . To indicate that $(x, y) \in \rightarrowtail$ we also write $x \rightarrowtail y$. If $X = Y$ then $\rightarrowtail$ is a *binary relation on X*. The diagonal of X will be denoted by $=_X$ or just X if it is clear from the context that a binary relation is meant. Let

P be a property of relations. $\rightarrowtail'$ is the P *closure* of $\rightarrowtail$ if it is the least relation (with respect to the subset ordering) extending $\rightarrowtail$ that has the property P, if such a relation exists.

Let $\rightarrowtail_1$, $\rightarrowtail_2$ be relations on X and Y and on Y and Z respectively.

The *composition of* $\rightarrowtail_1$ *and* $\rightarrowtail_2$ is the relation $\rightarrowtail_1 \cdot \rightarrowtail_2$ on X and Z defined by $x \rightarrowtail_1 \cdot \rightarrowtail_2 z \Leftrightarrow \exists y[x \rightarrowtail_1 y \rightarrowtail_2 z]$.

Let $\rightarrowtail$ be a binary relation on X. We define $\rightarrowtail^n$ by induction as follows.

1. $\rightarrowtail^0 = X$
2. $\rightarrowtail^{n+1} = \rightarrowtail^n \cdot \rightarrowtail$

Let $\rightarrowtail$ be a relation on X and Y. The *inverse of* $\rightarrowtail$ is the relation $\rightarrowtail^{-1}$ on Y and X defined by $y \rightarrowtail^{-1} x \Leftrightarrow x \rightarrowtail y$. If the symbol used to denote the relations can be 'mirrored' then also the mirrored symbol will be used to denote the inverse relation, e.g. $\leftarrowtail = \rightarrowtail^{-1}$.

Given a binary relation $\rightarrowtail$ on X, we say it is

1. *reflexive* if $X \subseteq \rightarrowtail$,
2. *irreflexive* if $X \cap \rightarrowtail = \emptyset$,
3. *symmetric* if $\rightarrowtail \subseteq \leftarrowtail$,
4. *transitive* if $\rightarrowtail^2 \subseteq \rightarrowtail$.

The reflexive closure of $\rightarrowtail$ will be denoted by $\overset{X}{\rightarrowtail}$. For the symmetric closure of $\rightarrowtail$ we have no special notation. The transitive closure of $\rightarrowtail$ will be denoted by $\rightarrowtail^+$.

A relation is a *quasi-order* if it is reflexive and transitive. A relation is a *partial order* if it is irreflexive and transitive. A relation $\rightarrowtail$ is a *total order* if it is a quasi-order and $\rightarrowtail \cup \leftarrowtail = X^2$. An equivalence relation is a symmetric quasi-order.

Let $\precsim$ be a quasi-order. The *equivalence relation generated by* $\precsim$ is $\precsim \cap \succsim$ and is denoted by $\sim$. The *partial order generated by* $\precsim$ is $\precsim - \sim$ and is denoted by $\prec$. (Of course only in cases where the quasi-order symbol admits these notations.)

A binary relation $\rightarrowtail$ *satisfies the diamond property* (notation $\rightarrowtail \models \diamond$) if $\leftarrowtail \cdot \rightarrowtail \subseteq \rightarrowtail \cdot \leftarrowtail$.

α-congruence

DEFINITION A.1. Let M and M' be generalised lambda terms. M is *α-congruent* with M', notation $M \equiv_\alpha M'$, if

1. $T_M = T_{M'}$,
2. $\mathcal{O}_\lambda(M) = \mathcal{O}_\lambda(M')$, $\mathcal{O}_@(M) = \mathcal{O}_@(M')$ and $\mathcal{O}_{\mathcal{FV}(M)}(M) = \mathcal{O}_{\mathcal{FV}(M')}(M')$, and
3. $\forall w \in \mathcal{O}_\lambda(M)$, For all free variable occurrences u, v, in either the pattern or the body of $w \backslash M$:

$$\ell_{w\backslash M}(u) \equiv \ell_{w\backslash M}(v) \in \mathcal{FV}(w * \langle 0 \rangle \backslash M) \Leftrightarrow \ell_{w\backslash M'}(u) \equiv \ell_{w\backslash M'}(v) \in \mathcal{FV}(w * \langle 0 \rangle \backslash M')$$

In words this reads : M and M' have the same underlying tree and variables that are bound by a variable occurrence in the pattern of an abstraction in M, are bound by that occurrence in M' and vice versa.

B. Proofs

PROOF. (Lemma 4.16)
By induction on the definition of $M \underset{1}{\twoheadrightarrow} M'$.

1. $M \underset{1}{\twoheadrightarrow} M'$ is $M \underset{1}{\twoheadrightarrow} M$. Then one has to show $M[\vec{N}/\vec{x}] \underset{1}{\twoheadrightarrow} M[\vec{N'}/\vec{x}]$. This follows by induction on the structure of M.
   (a) $M \equiv x_i \in \vec{x}$, then $M[\vec{N}/\vec{x}] \equiv N_i \underset{1}{\twoheadrightarrow} N_i' \equiv M[\vec{N'}/\vec{x}]$ by assumption.
   (b) $M \equiv y \notin \vec{x}$, then $M[\vec{N}/\vec{x}] \equiv y \equiv M[\vec{N'}/\vec{x}]$ by Definition 4.14(1).
   (c) $M \equiv (PQ)$, then $M[\vec{N}/\vec{x}] \equiv (P[\vec{N}/\vec{x}]Q[\vec{N}/\vec{x}]) \underset{1}{\twoheadrightarrow} (P[\vec{N'}/\vec{x}]Q[\vec{N'}/\vec{x}]) \equiv M[\vec{N'}/\vec{x}]$ by the induction hypothesis and Definition 4.14(2).
   (d) $M \equiv (\lambda Y.P)$, then $M[\vec{N}/\vec{x}] \equiv (\lambda Y.P[\vec{N}/\vec{x}]) \underset{1}{\twoheadrightarrow} (\lambda Y.P[\vec{N'}/\vec{x}]) \equiv M[\vec{N'}/\vec{x}]$ by the induction hypothesis and Definition 4.14(3).
2. $M \underset{1}{\twoheadrightarrow} M'$ is $(PQ) \underset{1}{\twoheadrightarrow} (P'Q')$ and is a direct consequence of $P \underset{1}{\twoheadrightarrow} P'$, $Q \underset{1}{\twoheadrightarrow} Q'$. Then $M[\vec{N}/\vec{x}] \equiv (P[\vec{N}/\vec{x}]Q[\vec{N}/\vec{x}]) \underset{1}{\twoheadrightarrow} (P'[\vec{N'}/\vec{x}]Q'[\vec{N'}/\vec{x'}]) \equiv M'[\vec{N'}/\vec{x}]$ by Definition 4.14(2) and the induction hypothesis.
3. $M \underset{1}{\twoheadrightarrow} M'$ is $((\lambda Y.P)Y[\vec{Q}/\vec{y}]) \underset{1}{\twoheadrightarrow} P'[\vec{Q'}/\vec{y}]$ and is a direct consequence of $P \underset{1}{\twoheadrightarrow} P'$, $\vec{Q} \underset{1}{\twoheadrightarrow} \vec{Q'}$. Then

$$\begin{aligned} M[\vec{N}/\vec{x}] &\equiv ((\lambda Y.P[\vec{N}/\vec{x}])Y[\vec{Q}/\vec{y}][\vec{N}/\vec{x}]) \\ &\equiv ((\lambda Y.P[\vec{N}/\vec{x}])Y[\vec{Q}[\vec{N}/\vec{x}]/\vec{y}]) \\ &\underset{1}{\twoheadrightarrow} P'[\vec{N'}/\vec{x}][\vec{Q'}[\vec{N'}/\vec{x}]/\vec{y}] \\ &\equiv P'[\vec{Q'}/\vec{y}][\vec{N'}/\vec{x}] \\ &\equiv M'[\vec{N'}/\vec{x}] \end{aligned}$$

   by the Variable Convention, Definition 4.14(4), the induction hypothesis and the Substitution Lemma 3.10.
4. $M \underset{1}{\twoheadrightarrow} M'$ is $(\lambda Y.P) \underset{1}{\twoheadrightarrow} (\lambda Y.P')$ and is a direct consequence of $P \underset{1}{\twoheadrightarrow} P'$. Then $M[\vec{N}/\vec{x}] \equiv (\lambda Y.P[\vec{N}/\vec{x}]) \underset{1}{\twoheadrightarrow} (\lambda Y.P[\vec{N}/\vec{x}]) \equiv M'[\vec{N'}/\vec{x}]$ by the Variable Convention, Definition 4.14(3) and the induction hypothesis. □

PROOF. (Lemma 4.32)

1. By induction on the structure of M :

   (a) $M \in \mathcal{V}$ then we can distinguish two cases :

      i. $M \equiv x_i \in \vec{x}$, then $\psi(x_i[\vec{N}/\vec{x}]) \equiv \psi(N_i) \equiv \psi(x_i)[\psi(\vec{N})/\vec{x}]$

      ii. $M \equiv y \notin \vec{x}$, then $\psi(y[\vec{N}/\vec{x}]) \equiv \psi(y) \equiv \psi(y)[\psi(\vec{N})/\vec{x}]$

   (b) $M \equiv (M_1 M_2)$ then again we distinguish two cases :

      i. $M \notin \boldsymbol{\beta}_{\mathbb{N}}$, then

$$\begin{aligned}\psi((M_1M_2)[\vec{N}/\vec{x}]) &\equiv \psi((M_1[\vec{N}/\vec{x}]M_2[\vec{N}/\vec{x}])) \\ &\equiv (\psi(M_1[\vec{N}/\vec{x}])\psi(M_2[\vec{N}/\vec{x}])) \\ &\equiv (\psi(M_1)[\psi(\vec{N})/\vec{x}]\psi(M_2)[\psi(\vec{N})/\vec{x}]) \\ &\equiv (\psi(M_1)\psi(M_2))[\psi(\vec{N})/\vec{x}] \\ &\equiv \psi((M_1M_2))[\psi(\vec{N})/\vec{x}]\end{aligned}$$

      by the induction hypothesis.

      ii. $M \in \boldsymbol{\beta}_{\mathbb{N}}$, so $M_1 \equiv (\lambda_i Y.P)$ and $M_2 \equiv Y[\vec{Q}/\vec{y}]$, then

$$\begin{aligned}\psi((M_1M_2)[\vec{N}/\vec{x}]) &\equiv \psi(((\lambda_i Y.P[\vec{N}/\vec{x}])Y[\vec{Q}[\vec{N}/\vec{x}]/\vec{y}])) \\ &\equiv \psi(P[\vec{N}/\vec{x}])[\psi(\vec{Q}[\vec{N}/\vec{x}])/\vec{y}] \\ &\equiv \psi(P)[\psi(\vec{N})/\vec{x}][\psi(\vec{Q})[\psi(\vec{N})/\vec{x}]/\vec{y}] \\ &\equiv \psi(P)[\psi(\vec{Q})/\vec{y}][\psi(\vec{N})/\vec{x}] \\ &\equiv \psi(((\lambda_i Y.P)Y[\vec{Q}/\vec{y}]))[\psi(\vec{N})/\vec{x}] \\ &\equiv \psi((M_1M_2))[\psi(\vec{N})/\vec{x}]\end{aligned}$$

      by the induction hypothesis and the Substitution Lemma 3.10.

   (c) $M \equiv (\lambda Y.M_1)$ then

$$\begin{aligned}\psi((\lambda Y.M_1)[\vec{N}/\vec{x}]) &\equiv \psi((\lambda Y.M_1[\vec{N}/\vec{x}])) \\ &\equiv (\lambda Y.\psi(M_1[\vec{N}/\vec{x}])) \\ &\equiv (\lambda Y.\psi(M_1)[\psi(\vec{N})/\vec{x}]) \\ &\equiv \psi((\lambda Y.M_1))[\psi(\vec{N})/\vec{x}]\end{aligned}$$

2. We prove, by induction on the generation of $\rightarrow_{\beta'}$, $\forall M,\ N \in \Lambda'\Phi$

$$M \rightarrow_{\beta'} N \ \Rightarrow\ \psi(M) \twoheadrightarrow_{\beta} \psi(N).$$

   The result then follows by transitivity.

(a) $M \to_{\beta'} N$ because $(M,N) \in \boldsymbol{\beta}'$, so $M \equiv ((\lambda_i Y.P)Y[\vec{Q}/\vec{y}])$ and $N \equiv P[\vec{Q}/\vec{y}]$, then $\psi(M) \equiv \psi(P)[\psi(\vec{Q})/\vec{y}] \equiv \psi(N)$ by Lemma 4.32(1).

(b) $M \to_{\beta'} N$ because $M_1 \to_{\beta'} N_1$, $M \equiv (M_1 L)$ and $N \equiv (N_1 L)$, then either

i. $M \notin \boldsymbol{\beta}_{\mathbb{N}}$, then $\psi((M_1 L)) \equiv (\psi(M_1)\psi(L)) \twoheadrightarrow_\beta (\psi(N_1)\psi(L)) \equiv \psi((N_1 L))$ by the induction hypothesis, compatibility of $\twoheadrightarrow_\beta$ and the observation that $N \notin \boldsymbol{\beta}_{\mathbb{N}}$, or

ii. $M \in \boldsymbol{\beta}_{\mathbb{N}}$ so $M_1 \equiv (\lambda_i Y.P)$, $L \equiv Y[\vec{Q}/\vec{y}]$ and $N_1 \equiv (\lambda_i Y.P')$ with $P \to_{\beta'} P'$, then $\psi((M_1 L)) \equiv \psi(P)[\psi(\vec{Q})/\vec{y}] \twoheadrightarrow_\beta \psi(P')[\psi(\vec{Q})/\vec{y}] \equiv \psi((N_1 L))$ by the induction hypothesis and Proposition 4.7(2).

(c) $M \to_{\beta'} N$ because $M_2 \to_{\beta'} N_2$, $M \equiv (L M_2)$ and $N \equiv (L N_2)$, then either

i. $M \notin \boldsymbol{\beta}_{\mathbb{N}}$ then $\psi((L M_2)) \equiv (\psi(L)\psi(M_2)) \twoheadrightarrow_\beta (\psi(L)\psi(N_2)) \equiv \psi((L N_2))$ by the induction hypothesis, compatibility of $\twoheadrightarrow_\beta$ and the observation that $N \notin \boldsymbol{\beta}_{\mathbb{N}}$, or

ii. $M \in \boldsymbol{\beta}_{\mathbb{N}}$ so $L \equiv (\lambda_i Y.P)$, $M_2 \equiv Y[\vec{Q}/\vec{y}]$ and $N_2 \equiv Y[\vec{Q'}/\vec{y}]$ with $\vec{Q} \to_{\beta'} \vec{Q'}$, then
$\psi((L M_2)) \equiv \psi(P)[\psi(\vec{Q})/\vec{y}] \twoheadrightarrow_\beta \psi(P)[\psi(\vec{Q'})/\vec{y}] \equiv \psi((L N_2))$ by the induction hypothesis and Proposition 4.7(1).

(d) $M \to_{\beta'} N$ because $M_1 \to_{\beta'} N_1$, $M \equiv (\lambda Y.M_1)$ and $N \equiv (\lambda Y.N_1)$, then $\psi((\lambda Y.M_1)) \equiv (\lambda Y.\psi(M_1)) \twoheadrightarrow_\beta (\lambda Y.\psi(N_1)) \equiv \psi((\lambda Y.N_1))$ by the induction hypothesis and compatibility of $\twoheadrightarrow_\beta$. □

PROOF. (Lemma 4.33)

1. $M \equiv x \in \mathcal{V}$, then
$\lfloor x \rfloor \equiv x \equiv \psi(x)$.

2. $M \equiv (M_1 M_2)$, then there are two cases :

(a) $M \notin \boldsymbol{\beta}_{\mathbb{N}}$, then $\lfloor (M_1 M_2) \rfloor \equiv (\lfloor M_1 \rfloor \lfloor M_2 \rfloor) \twoheadrightarrow_\beta (\psi(M_1)\psi(M_2)) \equiv \psi((M_1 M_2))$.

(b) $M \in \boldsymbol{\beta}_{\mathbb{N}}$, so $M_1 \equiv (\lambda_i Y.P)$ and $M_2 \equiv Y[\vec{Q}/\vec{y}]$, then

$$\begin{array}{rl} \lfloor (M_1 M_2) \rfloor \equiv & ((\lambda_i Y.\lfloor P \rfloor) Y[\lfloor \vec{Q} \rfloor / \vec{y}]) \\ \twoheadrightarrow_\beta & ((\lambda_i Y.\psi(P)) Y[\psi(\vec{Q})/\vec{y}]) \\ \to_\beta & \psi(P)[\psi(\vec{Q})/\vec{y}] \\ \equiv & \psi((M_1 M_2)) \end{array}$$

3. $M \equiv (\lambda X.M_1)$, then $\lfloor (\lambda X.M_1) \rfloor \equiv (\lambda X.\lfloor M_1 \rfloor) \twoheadrightarrow_\beta (\lambda X.\psi(M_1)) \equiv \psi((\lambda X.M_1))$. □

PROOF. (Lemma 5.12(4))
By induction on the structure of the term M.

1. $M \in \mathcal{V}$, two cases arise :
   (a) $M \equiv x$, $\langle x \rangle \equiv \vec{x}$, then $\|x\|_\rho = \rho[\vec{0}/\vec{x}](x) + \|x\|_\rho = \|x\|_{\rho[\vec{0}/\vec{x}]} + \sum_{z \in \vec{x}} \|z\|_\rho$
   (b) $M \equiv y$, $y \notin \vec{x} = \langle\rangle$, then $\|y\|_\rho = \|y\|_{\rho[\vec{0}/\vec{x}]} + \sum_{z \in \langle\rangle} \|z\|_\rho$
2. $M \equiv (M_1 M_2)$, again we consider two cases
   (a) $M \notin \beta_{\mathbb{N}}$, then

$$\begin{aligned}
\|M\|_\rho &= \|M_1\|_\rho + \|M_2\|_\rho \\
&\geq \|M_1\|_{\rho[\vec{0}/\vec{x} \cap \vec{m_1}]} + \sum_{z \in \vec{x} \cap \vec{m_1}} \|z\|_\rho + \|M_2\|_{\rho[\vec{0}/\vec{x} \cap \vec{m_2}]} + \sum_{z \in \vec{x} \cap \vec{m_2}} \|z\|_\rho \\
&= \|(M_1 M_2)\|_{\rho[\vec{0}/\vec{x}]} + \sum_{z \in \vec{x}} \|z\|_\rho
\end{aligned}$$

   (b) $M \in \beta_{\mathbb{N}}$, $M_1 \equiv (\lambda_i Y.P)$ and $M_2 \equiv Y[\vec{Q}/\vec{y}]$, then

$$\begin{aligned}
\|((\lambda_i Y.P)Y[\vec{Q}/\vec{y}])\|_\rho &= 1 + \|P\|_{\rho[\vec{Q}/\vec{y}]} + \sum_{y_i \in \vec{y} - \vec{p}} \|Q_i\|_\rho \\
&\geq 1 + \|P\|_{\rho[\vec{Q}/\vec{y}][\vec{0}/\vec{x} \cap \vec{p}]} + \sum_{z \in \vec{x} \cap \vec{p}} \|z\|_{\rho[\vec{Q}/\vec{y}]} + \sum_{y_i \in \vec{y} - \vec{p}} \left(\|Q_i\|_{\rho[\vec{0}/\vec{x} \cap \vec{q_i}]} + \sum_{z \in \vec{x} \cap \vec{q_i}} \|z\|_\rho \right) \\
&\geq 1 + \|P\|_{\rho[\vec{0}/\vec{x}][\|\vec{Q}\|_\rho / \vec{y} \cap \vec{p}]} + \sum_{z \in \vec{x} \cap \vec{p}} \|z\|_\rho + \sum_{y_i \in \vec{y} - \vec{p}} \|Q_i\|_{\rho[\vec{0}/\vec{x}]} + \sum_{z \in \vec{x} \cap \left(\bigcup_{y_i \in \vec{y} - \vec{p}} \vec{q_i} \right)} \|z\|_\rho \\
&\geq 1 + \|P\|_{\rho[\vec{0}/\vec{x}][\vec{Q}/\vec{y}]} + \sum_{z \in \vec{x} \cap \left(\bigcup_{y_i \in \vec{y} \cap \vec{p}} \vec{q_i} \right)} \|z\|_\rho + \\
&\quad \sum_{y_i \in \vec{y} - \vec{p}} \|Q_i\|_{\rho[\vec{0}/\vec{x}]} + \sum_{z \in \vec{x} \cap \left(\vec{p} \cup \bigcup_{y_i \in \vec{y} - \vec{p}} \vec{q_i} \right)} \|z\|_\rho \\
&= \|((\lambda_i Y.P)Y[\vec{Q}/\vec{y}])\|_{\rho[\vec{0}/\vec{x}]} + \sum_{z \in \vec{x}} \|z\|_\rho
\end{aligned}$$

3. $M \equiv (\lambda Y.M_1)$, then

$$\begin{aligned}
\|(\lambda Y.M_1)\|_\rho &= \|M_1\|_{\rho[\vec{0}/\vec{y}]} \\
&\geq \|M_1\|_{\rho[\vec{0}/\vec{y}][\vec{0}/\vec{x} \cap \vec{m_1}]} + \sum_{z \in \vec{x} \cap \vec{m_1}} \|z\|_{\rho[\vec{0}/\vec{y}]} \\
&= \|M_1\|_{\rho[\vec{0}/\vec{x}][\vec{0}/\vec{y}]} + \sum_{z \in \vec{x}} \|z\|_\rho \\
&= \|(\lambda Y.M_1)\|_{\rho[\vec{0}/\vec{x}]} + \sum_{z \in \vec{x}} \|z\|_\rho
\end{aligned}$$

C. Generalised Nameless Terms

Instead of working modulo α-congruence, de Bruijn has shown [dB72] how to give a denotation of terms which identifies α-congruent terms.

We will extend this for generalised λ-terms. To see how this can be done we will first repeat the (slightly modified) definition of the de Bruijn notation of a λ-term.

DEFINITION C.1. NAMELESS TERMS

1. *nameless terms* are words over the alphabet $\mathbb{N} \cup \{\lambda, ., (,)\}$.
2. The set Λ^* of *nameless* (or λ^*-) terms is defined inductively as follows :
   (a) $i \in \mathbb{N} \Rightarrow i \in \Lambda^*$
   (b) $A, B \in \Lambda^* \Rightarrow (AB) \in \Lambda^*$
   (c) $B \in \Lambda^* \Rightarrow (\lambda 0.B) \in \Lambda^*$ (*nameless abstraction*)
3. The function $\Lambda^*(M) = [\![M]\!]_{\mathrm{id}_{\mathbb{N}}}$ maps a λ-term to its corresponding nameless term. We define $[\![M]\!]_\rho$, for all $\rho : \mathbb{N} \to \mathbb{N}$ by induction on the structure of M.
   (a) $M \equiv v_i \Rightarrow [\![M]\!]_\rho = \rho(i)$
   (b) $M \equiv (M_1 M_2) \Rightarrow [\![M]\!]_\rho = ([\![M_1]\!]_\rho [\![M_2]\!]_\rho)$
   (c) $M \equiv (\lambda v_i.M_2) \Rightarrow [\![M]\!]_\rho = (\lambda [\![v_i]\!]_{\rho'}.[\![M_2]\!]_{\rho'}) = (\lambda 0.[\![M_2]\!]_{\rho'})$,
   where $\rho' = (\rho \circ \mathrm{succ})[0/i]$ and succ is the successor function.

Observe that if we take $\mathcal{V} = \mathbb{N}$ then we almost have $\Lambda^* = \Lambda$. In nameless terms the name of the variable in a pattern is not needed in the body (only its position), so we take 0 as the pattern of an abstraction.

LEMMA C.2. $\Lambda^*(M) \equiv \Lambda^*(N) \Leftrightarrow M \equiv_\alpha N$

PROOF. Tedious. □

What changes for generalised lambda terms? For each variable we must specify, in addition to the distance to its binding abstraction, by which occurrence in the pattern of that abstraction the variable is bound. If we choose an ordering of the free variables of a term which solely depends on the underlying tree (so on the occurrence of a variable rather than on its name) we can define a denotation which identifies α-congruent terms.

DEFINITION C.3. GENERALISED NAMELESS TERMS

1. *Generalised nameless terms* are words over the alphabet $\mathbb{N}^2 \cup \{\lambda, ., (,)\}$.
2. The set $\Lambda^*\Lambda^*$ of *generalised* nameless terms is defined inductively as follows :
   (a) $(i, j) \in \mathbb{N}^2 \Rightarrow (i, j) \in \Lambda^*\Lambda^*$
   (b) $A, B \in \Lambda^*\Lambda^* \Rightarrow (AB) \in \Lambda^*\Lambda^*$

(c) $A,\ B \in \Lambda^*\Lambda^* \ \Rightarrow\ (\lambda A.B) \in \Lambda^*\Lambda^*$ (generalised nameless abstraction)

3. The function $\Lambda^*\Lambda^*(M) = [\![M]\!]_{\langle c_0, \mathrm{id}_{\mathbb{N}}\rangle}$, where c_0 is the constant zero function, maps a term to its corresponding nameless term. We define $[\![M]\!]_\rho$, for all $\rho : \mathbb{N} \to \mathbb{N}^2$ by induction on the structure of M.

   (a) $M \equiv v_i \ \Rightarrow\ [\![M]\!]_\rho = \rho(i)$

   (b) $M \equiv (M_1 M_2) \ \Rightarrow\ [\![M]\!]_\rho = ([\![M_1]\!]_\rho [\![M_2]\!]_\rho)$

   (c) $M \equiv (\lambda M_1.M_2) \ \Rightarrow\ [\![M]\!]_\rho = (\lambda [\![M_1]\!]_{\rho'}.[\![M_2]\!]_{\rho'})$,
   where $\rho' = (\rho \circ \mathrm{succ}')[(0,0), \ldots, (0,n-1)/i_0, \ldots, i_{n-1}]$, $\mathrm{succ}' = \mathrm{succ} \times \mathrm{id}_{\mathbb{N}}$ and $\langle v_{i_0}, \ldots, v_{i_{n-1}}\rangle$ is the sequence of free variables in M_1, lexicographically ordered.

Note that some generalised nameless terms do not correspond to generalised lambda terms, e.g. $(1,0)$, $(\lambda(0,0).(0,1))$. We do have however the following correspondence.

LEMMA C.4. $\Lambda^*\Lambda^*(M) \equiv \Lambda^*\Lambda^*(N) \ \Leftrightarrow\ M \equiv_\alpha N$

PROOF. Just as tedious. □

As for nameless terms we can define substitution directly for generalised nameless terms. The only technicality is to update the numbering of the terms to substitute when we encounter an abstraction.

DEFINITION C.5. NAMELESS SUBSTITUTION
Let A, $\vec{B} = \langle B_1, \ldots, B_n\rangle \in \Lambda^*\Lambda^*$, then $A[\vec{B}/\vec{i}] \equiv A[\sigma[\vec{B}/c_0 \times \vec{i}]]$, where for all $\sigma : \mathbb{N}^2 \to \Lambda^*\Lambda^*$, $A[\sigma]$ is defined by induction on the structure of A as follows

1. $(i,j)[\sigma] = \sigma((i,j))$

2. $(A_1 A_2)[\sigma] = (A_1[\sigma] A_2[\sigma])$

3. $(\lambda A_1.A_2)[\sigma] = (\lambda A_1[\sigma'].A_2[\sigma'])$,
   where $\sigma'((0,j)) = (0,j)$ and $\sigma'(\mathrm{succ}'((i,j))) = \sigma(i,j)[\mathrm{succ}']$

LEMMA C.6. $\Lambda^*\Lambda^*(M[\vec{N}/\vec{x}]) = \Lambda^*\Lambda^*(M)[\Lambda^*\Lambda^*(\vec{N})/\Lambda^*\Lambda^*(\vec{x})]$

PROOF. Even more tedious. □

References

[Bar84] H.P. Barendregt. *The Lambda Calculus, its Syntax and Semantics*. North-Holland, 2nd edition, 1984.

[dB72] N.G. de Bruijn. Lambda-calculus notation with nameless dummies, a tool for automatic formula manipulation. *Indagationes Mathematicae*, 34:381–392, 1972.

[dV87] R.C. de Vrijer. *Surjective Pairing and Strong Normalization: Two Themes in Lambda Calculus*. PhD thesis, Universiteit van Amsterdam, January 1987.

[HL79] G. Huet and J.-J. Lévy. Call by need computations in non-ambiguous linear term rewriting systems. Report 359, INRIA, 1979.

[HS86] J.R. Hindley and J.P. Seldin. *Introduction to Combinators and λ-Calculus*, volume 1 of *London Mathematical Society Students Texts*. Cambridge University Press, 1986.

[Klo80] J.W. Klop. *Combinatory Reduction Systems*, volume 127 of *Mathematical Centre Tracts*. Centre for Mathematics and Computer Science, Amsterdam, 1980. PhD thesis.

[Klo90] J.W. Klop. Term rewriting systems. In S. Abramsky, D. Gabbay, and T. Maibaum, editors, *Handbook of Logic in Computer Science*, volume I. Oxford University Press, 1990. to appear.

[New42] M.H.A. Newman. On theories with a combinatorial definition of equivalence. *Annals of Mathematics*, 43(2):223–243, 1942.

[Pey87] S.L. Peyton Jones. *The Implementation of Functional Programming Languages*. Prentice-Hall International, 1987.