J. Austral. Math. Soc. (*Series A*) **28** (1979), 269–282

A LAMBDA CALCULUS WITH NAIVE SUBSTITUTION

JOHN STAPLES

(Received 14 August 1978; revised 9 April 1979)

Communicated by J. N. Crossley

Abstract

An alternative approach is proposed to the basic definitions of the classical lambda calculus. A proof is sketched of the equivalence of the approach with the classical case. The new formulation simplifies some aspects of the syntactic theory of the lambda calculus. In particular it provides a justification for omitting in syntactic theory discussion of changes of bound variable.

1980 Mathematics subject classification (*Amer. Math. Soc.*): 03 B 40.

Introduction

This paper proposes an alternative approach to the basic definitions of the lambda calculus, and describes a natural sense in which it is equivalent to the classical calculus.

There are two ways in which our alternative approach can be used. It could be regarded as the basic definition of the lambda calculus, the classical notation being regarded simply as a convenient informal notation. Alternatively, it could be regarded as a justification for the growing tendency to ignore changes of bound variables, in the syntactic theory of the classical lambda calculus. The author favours the former rather than the latter view, on the grounds of simplicity, but the work has value from both points of view.

The well-known difficulty with the classical definition of the lambda calculus, see, for example, Hindley *et al.* (1972), is that the use of bound variables requires a non-naive definition of substitution, which in turn requires a careful consideration of changes of bound variable throughout the syntactic theory of the lambda calculus. The situation is similar to that which arises in linear algebra when co-ordinate-dependent arguments introduce spurious difficulties which do not occur in a coordinate-free approach. In the case of the lambda calculus a particular choice of bound variables for a term is like a particular choice of coordinate system in linear algebra. In each case the particular choice may introduce irrelevant complications into further discussion. In each case it is desirable to have an alternative mode of discussion which can focus on essentials. In the case of linear algebra the alternative is the axiomatic or geometric theory of vector spaces. In the case of the lambda calculus alternatives have been rather neglected; it is the purpose of this paper to give one of them.

In view of the recent growth of the model theory of the lambda calculus it is natural to question the purpose of different, equivalent formulations of lambda calculus syntax. Why should not freer model-thoeretic arguments replace the syntactic approach? The answer is that for systems such as the lambda calculus there are intrinsically syntactic questions which are of prime importance. For example, the efficiency problem, to show how to evaluate a term with normal form in a minimum number of steps, is a purely syntactic question because the model theory of the lambda calculus does not distinguish between a term with normal form and its normal form. The efficiency problem is important as a model of a significant problem in the compilation of computer programs. One of the motivations for the development of the current work was to simplify the arguments about efficient evaluation which are given in Staples (1979).

There have been other attacks on the irrelevant complications caused by bound variables. O'Donnell (1976) modified the conventional definition of the lambda calculus in a minimal way, by introducing separate sets of free and bound variables and by insisting that the choice of bound variable for a particular binding in a term should be determined by the position in that term of the binding. Although O'Donnell's approach is adequate for his purposes, it complicates the definitions of substitution and β-contraction.

On the other hand, de Bruijn (1972) has eliminated both free and bound variables in his approach to the lambda calculus. The substitution properties of a position in a term are determined by storing in that position the relative address of what is essentially a binding operator. The approach of de Bruijn is more radical than is necessary to deal with the problem of bound variables.

In the present approach free variables remain; only bound variables are removed. Binding is accomplished by operators which incorporate a list of relative addresses of the positions which they bind. This approach is shown to be practicable by a sketch of a proof of its equivalence with the classical approach.

Of course, combinatory logic is a formalism which does not have variables at all, and within which the lambda calculus can, in a weak sense, be defined. However, that does not help the study of efficient evaluation (for example) since the correspondence between the two syntaxes is not close enough. In particular, a contraction in the lambda calculus is not generally modelled by a contraction in combinatory logic.

The modified calculus discussed here is not intended to be a system of graph-like expressions; it can be regarded as a subtree replacement system in the sense of Rosen (1973). Neither is it intended primarily for computer implementation of lambda calculus calculations. Its primary purpose is to expose the underlying simplicity of substitution, β-contraction and η-contraction, free of the irrelevant complications of α-conversion, in order to facilitate the syntactical theory of the lambda calculus. In the author's experience it is very suitable for that purpose.

1. Definition of the calculus N

1.1. Throughout this paper we shall need to compare our new lambda calculus with the classical one. For definiteness we take the set L of terms of the classical lambda calculus to be as defined in Hindley *et al.* (1972); the definition of L is also sketched in 2.2 below.

1.2. In order to define the set N of terms of the new lambda calculus we first state the symbols with which the terms will be defined. They are left and right parentheses (and), comma , , lower case Greek lambda λ, numerals 0 and 1 and an infinite sequence $c_0, c_1, c_2, \ldots$ of other symbols which we shall call variables. In what follows the symbol c_0 will play the role of a blank symbol, and $c_0, c_1, c_2, \ldots$ will play roles similar to those played by free variables in the classical lambda calculus.

The key to the definition of the terms of N is that we shall specify the positions to be associated with a given occurrence of λ, that is we specify the scope of λ, by a list of names for these positions, rather than by a bound variable.

1.3. We simultaneously recursively define the terms of N, the positions of subterms of a term and the free positions of variables in a term as follows. For the sake of readability the notion of the subterm is not defined as precisely as the other concepts, but the usual notion of subterm is intended. Positions will be (named by) strings of binary digits; we write e for the empty string, *not* for the blank symbol. Thus 11 and $1e1$ denote the same two-symbol string.

Case 1. Each variable c_i is a term; it is its own unique subterm, which is defined to be in position e, and e is a free position.

Case 2. If P, Q are terms then so is (PQ). A subterm of P in position b is a subterm of (PQ) in position $0b$, which is a free position if and only if b is a free position of P. A subterm of Q in position b is a subterm of (PQ) in position $1b$, which is a free position if and only if b is a free position of Q. Finally, (PQ) is a subterm of (PQ) in position e.

Case 3. If T is a term and if α is a list $(p_1, \dots, p_n)$, $n \geqslant 0$, of some free positions of distinct occurrences of c_0 in T, in increasing lexicographic order, then $\lambda\alpha T$ is a term. A subterm of T in position b is a subterm of $\lambda\alpha T$ in position $1b$, which is a free position if and only if b is a free position in T which is not one of the entries in the list. Finally, $\lambda\alpha T$ is a subterm of $\lambda\alpha T$ in position e.

REMARK. In the classical lambda calculus L one can lambda-abstract with respect to any variable. Here is an example to show how that is modelled in N.

In the classical calculus $T \equiv x_2(x_1 x_2)$ is a term, and its lambda-abstraction with respect to x_1 is $\lambda x_1(x_2(x_1 x_0))$. The term corresponding to T in N is $c_2(c_1 c_0)$, and its lambda-abstraction with respect to c_1 is obtained as follows.

(i) Write α for the list of free positions of c_1; in this case α contains just one entry, 10.

(ii) The required lambda-abstraction is then

$$\lambda\alpha([c_0/\alpha](c_2(c_1 c_0))) \equiv \lambda\alpha(c_2(c_0 c_0)) \equiv \lambda(10)(c_2(c_0 c_0));$$

in which c_0 has both a free and a bound occurrence.

Notice also that the lambda abstraction of the resulting term with respect to (free occurrences of) c_0 is

$$\lambda(111)(\lambda(10)(c_2(c_0 c_0))),$$

in which the two occurrences of c_0 are within the scopes of different lambdas.

1.4. Positions of variables in a term which are not free positions are called bound positions. Notice that from the above definition only the variable c_0 can occur in bound positions, and that c_0 may also occur in free positions. If an occurrence of c_0 in a term T is in a bound position p, and if T' is the largest subterm of T in which this occurrence is in a free position, then from the above definition T has a subterm $\lambda\alpha T'$ in position q such that some position r in the list α is such that $p = q1r$.

We may call $\lambda\alpha T'$ the binding subterm of this occurrence of c_0. The positions in the list α may be called the scope of the occurrence of λ which immediately precedes α; we may also say that this occurrence of c_0 is within the scope of λ. We may call such an occurrence of a list α in a term a λ-list.

We adopt throughout this paper the convention that when lists of positions are referred to, they are always implied to be listed in increasing lexicographic order. The empty position is the first in that order.

1.5. We adopt the notation conventions that outermost parentheses may be omitted, and that parentheses may be omitted by association to the left. Thus for example, $\lambda\alpha XY$ denotes $((\lambda\alpha X)Y)$, not $\lambda\alpha(XY)$.

It will also be convenient to introduce an informal composition notation, as in the following example:

$$A \circ B \circ C \circ DX \quad \text{abbreviates} \quad A(B(C(DX))).$$

1.6. As is usual, we write $X \equiv Y$ to denote that the two strings X and Y of symbols are identical.

1.7. In any lambda calculus a fundamental role is played by the notion of substitution for free occurrences of a variable. For N substitution is defined recursively as follows, where in all cases α is a list of some, not necessarily all, positions of occurrences of a variable in a term Y. We do not insist that all the positions in α be free, though it is only the free positions which are acted upon; it would make no essential difference however, to insist that all positions in α be free.

Case 1. For all terms X and Y, if α includes no free positions of Y then $([X/\alpha]Y)$ is defined to be Y.

Case 2. Otherwise; if Y is a variable then $\alpha = (e)$ and $([X/(e)]Y)$ is defined to be X.

Case 3. Otherwise; if $Y \equiv (Y_1 Y_2)$ then each position p in α has one of the forms $0q$ or $1r$; write $P_0\alpha$ for the list of such q's, write $P_1\alpha$ for the list of such r's, and define $([X/\alpha]Y)$ to be

$$(([X/P_0\alpha]Y_1)([X/P_1\alpha]Y_2)).$$

Case 4. Otherwise, $Y \equiv \lambda\gamma Z$ and every position p in α has the form $1r$; we write $P_1\alpha$ for the list of such r's and define $([X/\alpha]Y)$ to be $\lambda\gamma([X/P_1\alpha - \gamma]Z)$, where $P_1\alpha - \gamma$ denotes the list of positions which are in $P_1\alpha$ but not in γ.

Now the key result is:

1.8. Substitution is naive. That is, if α is a list of some of the positions of a variable c_i in a term Y, then $[X/\alpha]Y$ is obtained from Y by substituting X for each of the free occurrences of c_i in the positions listed in α.

The proof is by induction following the definition of substitution and is evident in each of the four cases.

1.9. Notations for lists of positions. It is already evident from the definition of substitution that some notations for manipulating lists of positions would be helpful; we use these:

(i) The empty list may be denoted ( ).

(ii) As already indicated, for every list α of positions the list of positions p such that $0p$ (respectively $1p$) is in α may be denoted $P_0\alpha$ (respectively $P_1\alpha$).

(iii) When a list α is transformed by left concatenation of all its positions p with a fixed position q to give positions qp, we may denote the resulting list $q\alpha$. For example, if $\alpha = (01, 100)$ and $q = 1$ then $q\alpha = (101, 1100)$.

(iv) When a list γ is formed from a list α and a fixed position q by including in γ a position p whenever qp is in α, then γ may be denoted $\bar{q}\alpha$. For example, $\bar{0}\alpha = P_0\alpha$; more particularly if $\alpha = (01, 100)$ then $\bar{0}\alpha = (1)$.

(v) For a list α of positions in a term T such that the subterm of T in position p has the form $\lambda\gamma XY$, we denote by α_p the list which is obtained from α by replacing each entry of the form $p01x$ by px, and by replacing each entry of the form $p1y$ by entries pqy, one for each q in γ. For example, if T is $(c_0(\lambda(1)(c_0 c_0)c_0))$ and p is 1 and α is $(0, 1010, 11)$, then α_p is $(0, 10, 11)$ and will be the list of free positions corresponding to α in the β-contractum of T, $(c_0(c_0 c_0))$.

(vi) When a list ε is obtained from a list α by deleting all entries which are in a list γ, we may write $\alpha - \gamma$ for ε.

(vii) When a list ε is obtained from a list α by replacing, for some fixed position p, each entry of α of the form $p10x$ by px, then ε may be denoted α_p^η. For example, a term T of the form $\lambda(1)(Mc_0)$ will have an η-contractum M; if α is a list of free positions in the subterm M of T, then α_p^η is the corresponding list of free positions in M.

1.10. In the following recursive definition, β-contraction of a term, and the position of a β-contraction, are defined.

Case 1. For every term of the form $\lambda\alpha XY$, $\lambda\alpha XY \to [Y/\alpha]X$ is a β-contraction with position e.

Case 2. If $X \to X'$ is a β-contraction with position p then $XY \to X'Y$ and $YX \to YX'$ are β-contractions with positions $0p$, $1p$ respectively.

Case 3. If $X \to X'$ is a β-contraction with position p then $\lambda\alpha X \to \lambda\alpha_p X'$ is a β-contraction with position $1p$.

Beta-contraction of terms will be denoted in future by $\to$. Beta-reduction of terms, to be denoted $\geqslant$, is then defined in the usual way as follows. For all terms X, Y, Z:

Case 1. $X \geqslant X$.

Case 2. If $X \to Y$ and $Y \geqslant Z$ then $X \geqslant Z$.

We may also write $X \geqslant Z$ to denote loosely a particular sequence of β-contractions of X to Z, and may also call such a sequence a β-reduction.

1.11. Eta-construction of terms is denoted $\rightarrow_\eta$ and is defined as follows.

Case 1. $\lambda(1)(Xc_0) \rightarrow_\eta X$, and has position e.

Case 2. If $X \rightarrow_\eta X'$ with position p, then $\lambda\alpha X \rightarrow_\eta \lambda\alpha_p^\eta X'$, and has position $1p$.

Case 3. If $X \rightarrow_\eta X'$ has position p then $XY \rightarrow_\eta X'Y$ has position $0p$, and $YX \rightarrow_\eta YX'$ has position $1p$.

Eta-reduction $\geqslant_\eta$ is then defined in the obvious way.

1.12. Strong reduction if terms of N is then defined to be a sequence of contractions, each of which is either a β-contraction or an η-contraction.

2. The correspondence between N and L; definitions and basic properties

2.1. In order to make precise the obviously close correspondence between N and L we define maps $V: N \rightarrow L$ and $U: L \rightarrow N$ as follows, and then derive some of their basic properties.

2.2. First, however, we recall briefly for the classical calculus L that its symbols are λ, (,), and an infinite sequence $x_0, x_1, x_2, \ldots$ of variables. Each variable is a term, if T_1 and T_2 are terms then so is $(T_1\, T_2)$, and if T is a term and x is a variable then $\lambda x T$ is a term.

2.3. The map $V: N \rightarrow L$ is defined recursively as follows.

Case 1. For every variable c_i, $Vc_i \equiv x_i$.

Case 2. If $T \equiv T_1\, T_2$ then $VT \equiv (VT_1)(VT_2)$.

Case 3. If $T \equiv \lambda\alpha X$ then $VT \equiv \lambda x_j[x_j/\alpha](VX)$, where x_j is the first variable of L other than x_0 which is neither free nor bound in $V([c_0/\alpha]\, X)$, and where $[x_j/\alpha]$ denotes the naive substitution of x_j for the occurrences of x_0 which are in the positions listed in α.

Remark. Whenever in this paper substitution for a list of free positions in a term of N is indicated, it is always naive substitution which is intended. For L

however, substitution for a list of free positions must as usual change bound variables to avoid clashes of bound variables with substituted free variables.

2.4. The map $U\colon L \to N$ is defined recursively as follows.

Case 1. For every variable x_i, $Ux_i \equiv c_i$.

Case 2. If $T \equiv T_1\,T_2$ then $UT \equiv (UT_1)(UT_2)$.

Case 3. If $T \equiv \lambda x_i\,X$ then $UT \equiv \lambda\alpha[c_0/\alpha](UX)$, where α is the list of positions of all free occurrences of x_i in X.

EXAMPLE. Consider the classical term $T \equiv \lambda x_1(\lambda x_2(x_1\,x_2))$. Then

$$UT \equiv \lambda(10)(\lambda(1)(c_0\,c_0)),$$

from the definition of U, and

$$V(UT) \equiv \lambda x_2(\lambda x_1(x_2\,x_1)),$$

from the definition of V. Thus T and $V(UT)$ are not identical, but are equal in the sense that they differ only by changes of bound variable.

Now we begin the study of the basic properties of U and V. At first some parts of proofs are given in detail so as to illustrate the method, but later proofs will be brief.

2.5. *For all $T \in L$, all x_k, and all lists α of some of the free positions of x_k in T; in the notation of* 1.5,

$$U \circ [x_j/\alpha]\,T \equiv [c_j/\alpha] \circ UT \quad \textit{for all } j.$$

PROOF. By induction following the construction of T. The nontrivial case is when $T \equiv \lambda x_i\,X$, $i \neq k$. Write γ for the list of free positions of x_i in X. Provided $i \neq j$,

$U \circ [x_j/\alpha]\,T \equiv \lambda\gamma([c_0/\gamma] \circ U \circ [x_j/P_1\,\alpha]\,X)$
$\equiv \lambda\gamma([c_0/\gamma] \circ [c_j/P_1\,\alpha] \circ UX)$, by inductive hypothesis,
$\equiv [c_j/\alpha]\,\lambda\gamma([c_0/\gamma] \circ UX)$, since $P_1\,\alpha, \gamma$ have no positions in common.

From 2.4 the latter term is just $[c_j/\alpha](UT)$.

The case $i = j$ is then a corollary:

$U \circ [x_j/\alpha]\,\lambda x_j\,X \equiv U \circ (\lambda x_m[x_j/P_1\,\alpha]\,[x_m/x_j]\,X)$ by definition of substitution in L; note $m \neq j$;
$\equiv U \circ [x_j/\alpha]\,\lambda x_m[x_m/x_j]\,X$ similarly,
$\equiv [c_j/\alpha] \circ U(\lambda x_m[x_m/x_j]\,X)$ by inductive hypothesis,
$\equiv [c_j/\alpha] \circ U(\lambda x_j\,X)$ from the definition of U, as required.

2.6. *For all $T \in N$, all variables c_k and all lists α of free positions of c_k in T,*

$$V \circ [c_0/\alpha]\, T \equiv [x_0/\alpha] \circ VT.$$

Proof. By induction following the construction of T. The only interesting case is when $T \equiv \lambda\gamma X$, when by definition of α, $P_1\,\alpha$ and γ have no positions in common. Thus

$$\begin{aligned} V \circ [c_0/\alpha]\, T &\equiv \lambda x_j([x_j/\gamma] \circ V \circ [c_0/P_1\,\alpha]\, X) \\ &\equiv \lambda x_j([x_j/\gamma] \circ [x_0/P_1\,\alpha] \circ VX) \quad \text{by inductive hypothesis,} \\ &\equiv [x_0/\alpha]\, \lambda x_j([x_j/\gamma] \circ VX). \end{aligned}$$

Now by definition of V, x_j is the least variable other than x_0 which is not free or bound in $V([c_0/\gamma]\, X)$. Thus $\lambda x_j([x_j/\gamma] \circ VX) \equiv VT$ and we have the stated result.

2.7. *For all $T \in N$, $U \circ VT \equiv T$.*

Proof. By induction following the construction of T. The interesting case is when $T \equiv \lambda\alpha X$. Then $VT \equiv \lambda x_j([x_j/\alpha] \circ VX)$, where x_j is as described in 2.3. Thus, writing γ for the list of free positions of x_j in $[x_j/\alpha] \circ VX$, we have by definition of x_j that $\alpha \equiv \gamma$, so

$$\begin{aligned} U \circ VT &\equiv \lambda\alpha([c_0/\alpha] \circ U \circ [x_j/\alpha] \circ VX) \\ &\equiv \lambda\alpha([c_0/\alpha] \circ [x_j/\alpha] \circ U \circ VX) \quad \text{by 2.5,} \\ &\equiv \lambda\alpha([c_0/\alpha] \circ U \circ VX) \\ &\equiv \lambda\alpha([c_0/\alpha]\, X) \quad \text{by inductive hypothesis,} \\ &\equiv T. \end{aligned}$$

2.8. To derive for $V \circ U$ the result corresponding to 2.7 is not so easy, because of problems arising from changes of bound variables in L. We sketch the argument as follows.

2.9. *If $T \in N$ has just k distinct variables occurring in free positions and l distinct occurrences of λ, if c_a is a variable such that $a > k+l+1$ which occurs in T in just the free positions comprising the list α, and if R (respectively S) is the set of variables x_j such that $0 < j < a$ and x_j is not free or bound in VT (respectively $V \circ [c_0/\alpha]\, T$) then R, S are nonempty and $R = S$.*

Proof. That R and S are nonempty is clear; $R = S$ is proved by induction following the construction of T.

2.10. Corollary. *In the notation of 2.9, the first variable other than x_0 which is not free or bound in VT is the same as the first variable other than x_0 which is not free or bound in $V \circ [c_0/\alpha]\, T$.*

It is well known that

2.11. *If X, $Y \in L$, if no variables free in X are bound in Y, if y is a variable of L which is not bound in Y and if α is the list of all free positions of y in Y, then $[X/y]\,Y$ results from Y by naive substitution: that is, $[X/y]\,Y \equiv [X/\alpha]\,Y$.*

2.12. In the next result and thereafter we write $X = Y$ to denote that terms X, Y differ only by changes of bound variable, as defined by Hindley *et al.* (1972), p. 6.

If S, $T \in L$ and $S = T$ then $US \equiv UT$.

PROOF. It is enough to assume that T results from S by a single change of bound variable. The proof is then by induction following the definition of change of bound variable.

2.13. COROLLARY. *If $S = T$ then $V \circ US \equiv V \circ UT$.*

2.14. *For all $T \in L$, $V \circ UT = T$.*

PROOF. By induction following the construction of T. The only interesting case is when $T \equiv \lambda x_a X$, say. Suppose that T has k distinct free variables and l distinct occurrences of λ; we assume, as we can in view of 2.13, that T has all its bound variables, including x_a chosen from the list

$$x_{k+l+2}, x_{k+l+3}, \ldots .$$

We also assume that no bound variable of T occurs free in T or within the scope of two distinct occurrences of λ. Then, writing α for the list of free occurrences of x_a in X, α is the list of all occurrences of x_a in X and, from 2.6, 2.9 and 2.11,

$$V \circ UT \equiv \lambda x_j [x_j/x_a]\,(V \circ UX).$$

Thus

$$V \circ UT = \lambda x_a (V \circ UX),$$

and by inductive hypothesis $V \circ UX = X$, so $V \circ UT = T$ as asserted.

3. The main result

3.1. (a) *For all X, $Y \in L$,*

(i) *$X = Y$ if and only if $UK \equiv UY$,*

(ii) *$X \rightarrow Z$ for some $Z = Y$ if and only if $UX \rightarrow UY$,*

(iii) *$X \rightarrow_\eta Z$ for some $Z = Y$ if and only if $UX \rightarrow_\eta UY$.*

(b) *For all* $X, Y \in N$,

(i) $X \equiv Y$ *if and only if* $VX = VY$,

(ii) $X \to Y$ *if and only if* $VX \to Z$ *for some* $Z = VY$,

(iii) $X \to_\eta Y$ *if and only if* $VX \to_\eta Z$ *for some* $Z = VY$.

In order to prove this result we show first that it is enough to prove the following lemma.

3.2. (a) *For all* $X, Y \in L$,

(i) *if* $X \to Y$ *then* $UX \to UY$,

(ii) *if* $X \to_\eta Y$ *then* $UX \to_\eta UY$.

(b) *For all* $X, Y \in N$,

(i) *if* $X \to Y$ *then* $VX \to Z$ *for some* $Z = VY$,

(ii) *if* $X \to_\eta Y$ *then* $VX \to_\eta VY$.

3.3. Proof of 3.1 from 3.2.

Proof of (a)(i). One part is 2.12; the converse is immediate from 2.14.

Proof of (a)(ii). If $X \to Z$ then $UX \to UZ$ from 3.2(a)(i), and $UZ \equiv UY$ from 2.12.

If conversely $UX \to UY$ then from 3.2(b)(i), $V \circ UX \to W$, for some $W = V \circ UY$. Now from 2.14, $V \circ UX = X$, so from a well-known lemma of the classical lambda calculus (see, for example, Hindley *et al.*), $X \to Z$ for some $Z = W$, as required.

Proof of (a)(iii). One part is immediate from 3.2(a)(ii) and 2.14. If conversely $UX \to_\eta UY$ then from 3.2(b)(ii),

$$V \circ UX \to_\eta V \circ UY.$$

Now $V \circ UX = X$ from 2.14, so it follows from a well-known lemma of the classical lambda calculus that $X \to_\eta Z$ for some $Z = V \circ UY$, as required.

Proof of (b)(i). One part is trivial; the converse is immediate from 2.12.

Proof of (b)(ii). If $X \to Y$ then from 2.7 $U \circ VX \to U \circ VY$, so from 3.1(a)(i) already proved, $VX \to Z$ for some $Z = VY$. If conversely $VX \to Z$ for some $Z = VY$ then from 2.7, 3.1(a)(ii) and 2.14,

$$X \equiv U \circ VX \to UZ \equiv U \circ VY \equiv Y.$$

Proof of (b)(iii). One part is trivial from 3.2(b)(ii). If conversely $VX \to_\eta Z$ for some $Z = VY$, then from 2.7, 3.1(a)(iii) and 2.14,

$$X \equiv U \circ VX \to_\eta UZ \equiv U \circ VY \equiv Y.$$

In order to complete the proof of 3.1 by proving 3.2 we need the following lemmas.

3.4. *If $X, Z \in L$ and α is the list of all free positions of x_i in Z then*

$$U \circ [X/x_i]Z \equiv [UX/\alpha]\, UZ.$$

3.5. *If X, Y, S, $T \in L$ are such that no variable occurring bound in X or S occurs free in Y or T, if $X = S$ and $Y = T$, and if α is a list of positions of some of the free occurrences of x_i in X, then*

$$[Y/\alpha]\, X = [T/\alpha]\, S.$$

Proof. It is enough to assume that S is obtained from X by a single change of bound variable. The argument is then by induction following the construction of X.

3.6. *If P, $Q \in N$, if α is a list of some of the positions of free occurrences of a variable c_i in P, and if $s_0, \ldots, s_k$ is a list of variables of L, then there is $W \in L$ such that $W = VP$, no bound variables of W are $s_0, \ldots, s_k$, and*

$$V \circ [Q/\alpha]P \equiv [VQ/\alpha]\, W.$$

Proof. By induction following the construction of P.

3.7. Proof of 3.2.

Proof of (a)(i). By induction following the definition of $X \to Y$. The only interesting case is when $X \equiv \lambda x_j PQ$ and $Y \equiv [Q/x_j]P$. Then, writing α for the list of positions of free occurrences of x_j in P,

$$\begin{aligned} UX &\equiv \lambda\alpha([c_0/\alpha] \circ UP)\, UQ \\ &\to [UQ/\alpha] \circ [c_0/\alpha] \circ UP \\ &\equiv [UQ/\alpha] \circ UP \\ &\equiv UY \quad \text{from 3.4.} \end{aligned}$$

Proof of (a)(ii). By induction following the definition of $X \to_\eta Y$. All cases are evident so details are omitted.

Proof of (b)(i). By induction following the definition of $X \to Y$. The only interesting case is when $X \equiv \lambda\alpha PQ$ and $Y \equiv [Q/\alpha]P$. Then from 3.6,

$$\begin{aligned} VY &\equiv V \circ [Q/\alpha]P \\ &\equiv [VQ/\alpha]\, W \quad \text{for some } W = VP \end{aligned}$$

such that x_j (the first variable other than x_0 which is not free or bound in

$V([c_0/\alpha]P))$ and all variables free in VQ are not bound in W. Thus

$$\begin{aligned} VX &\equiv \lambda x_j([x_j/\alpha] \circ VP)\, VQ \\ &= \lambda x_j([x_j/\alpha]\, W)\, VQ \quad \text{from 3.5,} \\ &\to [VQ/\alpha]\, W \\ &= VY. \end{aligned}$$

Thus by the lemma of Hindley *et al.* (1972), p. 141, $VX \to W$ for some $W = VY$.

Proof of (b)(ii). By induction following the definition of $X \to_\eta Y$. All cases are evident.

4. Discussion

4.1. It is clear from the main result that it would usually be possible to prove a lambda calculus result in N, without the burden of changes of bound variable, and then transfer the result to L. However, the transfer may be unnecessary, since there seems to be no reason why N should not replace L entirely.

4.2. In the remainder of this section we discuss a technicality which, though not a continuing difficulty comparable to α-conversion, does need to be dealt with before the benefits of N can be enjoyed.

Consider, for example, a list α of some free positions of a variable c_k in a term T, and consider two reductions R and S of T to say T'. We need to know, for example when considering the corresponding reductions of $\lambda\alpha[c_0/c_k]\,T$, that the list of positions of T' corresponding by the reduction R to the positions of T in α, is the same as the list of positions in T' corresponding by the reduction S to the positions of T in α. We prove that now.

4.3. First, we recursively define as follows the list $D_R\,\alpha$ of descendants, after a reduction R of a term T, of a list α of free positions of occurrences of c_k in T.

Case 1. If R is the trivial reduction $T \geqslant T$ then $D_R\,\alpha = \alpha$.

Case 2. If R is the reduction $T \geqslant T'' \to T'$, where $T'' \to T'$ is a β-contraction with position p, and where the reduction $T \geqslant T''$ is denoted F, then $D_R\,\alpha = (D_F\,\alpha)_p$, in the notation of 1.9(v). Similarly if $T'' \to_\eta T'$ is an η-contraction then $D_R\,\alpha = (D_F\,\alpha)_p^\eta$.

4.4. *Given terms X, Y and a reduction R of X to Y, and writing $(c_k)_X$ (respectively $(c_k)_Y$) for the list of all positions of free occurrences of c_k in X (respectively Y),*

$$D_R(c_k)_X \equiv (c_k)_Y.$$

Proof. Clearly it is enough for an inductive proof to prove the result for contractions. However, in each of the cases of β-contraction and η-contraction the result is evident from the definitions.

4.5. *If R, S are two reductions of a term X to a term Y and if α is a list of some free positions of a variable c_i in X, then*

$$D_R\alpha \equiv D_S\alpha.$$

PROOF. Consider that R applies also to $X' \equiv [c_k/\alpha]X$, where c_k is a variable which does not occur free or bound in X. Also, writing γ for the list of all free occurrences of c_k in X', evidently $\gamma \equiv \alpha$ and $D_R\alpha \equiv D_R\gamma$. The result then follows from 4.4, since

$$D_R\alpha \equiv D_R\gamma \equiv (c_k)_Y \equiv D_S\gamma \equiv D_S\alpha.$$

4.6. Hence the following notation can be used. When α is a list of free positions of a variable in a term X, and when $X \geqslant Y$, denote the descendant of α in Y by $D\alpha$. Notice that the notation $D\alpha$ can denote different lists, depending on which reductum Y of X is being referred to. Usually one need not distinguish between these different meanings, since the intended meaning can be inferred from the context.

Acknowledgements

I thank Roger Hindley, Martin Bunder, Mike O'Donnell, N. G. de Bruijn and the referee for helpful criticisms of earlier versions of this paper.

References

N. G. de Bruijn (1972), 'Lambda calculus notation with nameless dummies', *Indag. Math.* **34,** 381–392.

J. R. Hindley, B. Lercher and J. P. Seldin (1972), *Introduction to combinatory logic* (London Math. Soc. Lecture Notes no. 7, C.U.P., London).

M. J. O'Donnell (1976), *Reduction strategies in subtree replacement systems* (Ph.D. Thesis, Cornell University).

Barry K. Rosen (1973), 'Tree-manipulating systems and Church–Rosser theorems', *J.A.C.M* **20,** 160–187.

John Staples (1979), 'A graph-like lambda calculus for which leftmost-outermost reduction is optimal', *Proceedings of* 1978 *International Graph Grammars Workshop, Bad Honnef, Germany* (Lecture Notes in Computer Science 73, Springer-Verlag, Berlin).

Department of Mathematics and Computer Science
Queensland Institute of Technology
G.P.O. Box 2434
Brisbane 4001
Australia