Upper bound for the length of the norm of an expression in lambda-typed lambda calculus

M22

July 25, 1985

Upper bound for the length of the norm of an expression in lambda-typed lambda calculus

by N.G. de Bruijn

Department of Mathematics and Computing Science
Eindhoven University of Technology
PO Box 513, 5600MB Eindhoven, The Netherlands

1. For all notations used in this note we refer to [1]. Nevertheless it should be noted that for the material of the present note there is no substantial difference between the system $\Delta\Lambda$ of [1] and the system Λ of [2].

2. The length of a lambda tree or of a subtree of a lambda tree is just the number of end-points. So in particular, expressed in terms of character strings,

length(x) = 1, length(τ) = 1.
length(<U>V) = length(U) + length(V),
length([x:U]V) = length(U) + length(V).

3. If V is a lambda tree, its norm (as defined in [1], sections 5.9 and 5.10) is denoted by norm(V). The length of that norm, length(norm(V)), will be abbreviated to ln(V).

4. Theorem. If V is a lambda tree then

$$\ln(V) \leq 2^{\mathrm{length}(V) - 1} .$$

5. Sketch of a proof. We introduce the subdivided lambda trees (R,W,B) and (R,W,Y,B) as in [1],section 5.9. For these we have the norms norm(R,W,B) and norm(R,W,Y,B), which can be considered as the norms of of the subtrees WB and WYB, taken with typing of free variables by means of the abstractors in R.

As above, we shall abbreviate length(norm(R,W,B)) to ln(R,W,B) and length(norm(R,W,Y,B)) to ln(R,W,Y,B).

We consider all the dummies x for which $(R, \varepsilon, x) \in$ Slam3. In other words, these x's are the dummies attached to the abstractors in the main line of R. (We recall that R can be written as a sequence of applicator-abstractor pairs <P>[y:Q] and loose abstractors [z:U]). For each one of these we consider $\ln(R, \varepsilon, x)$, and the largest one of these numbers will be denoted by mxln(R). In case R is empty, we agree that mxln(R) = 1.

We are now in a position to announce

$$\ln(R,W,B) \leq f(R,B), \qquad \ln(R,W,Y,B) \leq f(R,B), \qquad (1)$$

where

$$f(R,B) = 2^{\mathrm{length}(B) - 1} \mathrm{mxln}(R).$$

The inequalities (1) can be proved by recursion, if we just follow the list of clauses of [1],section 5.9. In the cases (i) and (vi) we have ln(R, ε , τ) =1, ln(R, ε , ε , τ) = 1, and ln(R, τ) = mxln(R) $\geq$ 1. We shall explain what has to be done in the other cases: as a typical case we take case (iv).

In case (iv) it is stated what the norm is of (R, ε ,[x:U]B), under the assumption that we know the norms of (R, ε ,U) and (R[x:U], ε ,B). Accordingly we have to prove (1) for (R, ε ,[x:U]B), under the assumption that we know it for (R, ε ,U) as well as for (R[x:U], ε ,B).

We shall not carry out all this here. The work is of a simple nature. During the course of this work, however, we need a few auxiliary results which have to be proved separately by recursion. As one of these we mention ln(R,W,B) $\leq$ ln(R, ε ,B).

Once we have (1) for all cases, we have the theorem of section 4, just since the norm of the full lambda tree V was defined as norm(ε , ε ,V), and mxln(ε) was defined as 1.

6. The estimate in the theorem is best possible. The examples for which the upper bound are reached are given by

E1 =   ,

E2 = [x1:E1]x1,

E3 = [x2:E2]x2,

E4 = [x3:E3]x3,

......

We have length(En) = n, ln(En) = 2^{n-1} .

REFERENCES

1. N.G. de Bruijn. Generalizing Automath by means of a lambda-typed lambda calculus. To be published. (P153).

2. R.P. Nederpelt. Strong normalization in a typed lambda calculus with lambda structured types. Doctoral dissertation, Eindhoven University of Technology, 1973.