A NEW TYPE ASSIGNMENT FOR $\lambda$-TERMS*

M. Coppo and M. Dezani-Ciancaglini

Abstract

In the present paper we propose a new type assignment for $\lambda$-terms whose motivation is to introduce a system with simple inferential rules to study termination (i.e. the property of having a normal form) of $\lambda$-terms. The main results that will be proved in this paper are:

a) all $\lambda$-terms in normal form possess a type,

b) all $\lambda$-terms which possess a type reduce to normal form.

1. Introduction

In recent years many type assignments to terms of the $\lambda$-calculus have been studied. An exhaustive review of these theories, in the context of the logical type theory [7], is given in [12, pp. 1—14]. In all these theories the functional character of $\lambda$-terms is generalized by representing types with more powerful objects than that ones introduced in the basic theory of functionality [7, Chapter 9].

In our theory no such object is required but an equivalence and a partial order relation between types are introduced with the assumption that each $\lambda$-term which possesses a type $\tau$ possesses also all types equivalent or lower than $\tau$.

The aim of the present type assignment is to introduce a powerful method (suitable also for mechanical implementation) for studying the property for a $\lambda$-term to possess a normal form (n.f.). In fact the basic result of this paper is that the property of having a type for a $\lambda$-term implies that this term and all its subterms possess n.f. Moreover the set of types that can be assigned to a $\lambda$-term in n.f. is proved to be always decidable.

The set of types is built recursively from two atomic elements, 0 and 1. Type 0 represents the property for a $\lambda$-term to have a n.f., while type 1 represents the property for a $\lambda$-term to be such that its applications to an arbitrary number of $\lambda$-terms possessing n.f., possesses a n.f. too. The other types are built from 0 and 1 by the usual operation of composition (represented by object F in [7]). We observe that the functional character represented by the fundamental types 0 and 1 (and, consequently, by all other types) can be interpreted only in terms of

* Eingegangen am 27.7.1977.
termination properties. So, if a $\lambda$-term $X$ possesses type $(0)0$\(^1\) this means that, if $Y$ is a $\lambda$-term which possesses type 0 (i.e. $Y$ possesses a n.f.) then $XY$ too possesses type 0, i.e. $XY$ reduces to a n.f.

The informal description of the properties represented by types 0 and 1 suggests some considerations about their relations.

First we notice that if $X$ is a $\lambda$-term whose applications to arbitrary $\lambda$-terms possessing n.f. possesses a n.f. too and $Y$ is a $\lambda$-term which possesses a n.f., then $XY$ has the same property as $X$. It turns out that if $X$ is a $\lambda$-term with type 1 and $Y$ is a $\lambda$-term with type 0, $XY$ too must be a $\lambda$-term with type 1. This fact can be formalized by postulating that types 1 and $(0)1$ are, in effect, equivalent, i.e. they represent the same property. In a similar way, as it will be shown in the paper, we can justify the assumption that $(1)0$ is equivalent to 0.

Secondly, it is evident that the property represented by type 1 is much more restrictive than that one represented by type 0. Then we can assume that each $\lambda$-term which possesses type 1 possesses also type 0. This property, which will be formalized by the relation $0 = t$, can be generalized in a natural way and leads to the introduction of a partial order relation between types. We can so postulate that, if a given $\lambda$-term possesses a type $\tau$, it possesses also all types which are lower than $\tau$. In the paper it will be proved that all these assumptions lead to a consistent system.

An advantage of “despoiling” types of any meaning not related to termination properties is that we can obtain, in this way, a system in which, with a very simple structure of types and type assignment rules, we can assign types also to $\lambda$-terms for which this is impossible (or quite troublesome) in most other theories.

For example in the basic theory of functionality [7] it is impossible to assign a type to the combinator $W_\ast \equiv \lambda x(xx)$ because of the difficulties produced by the application of a variable $x$ to itself. But, with respect to termination properties, it is enough, for example, that $x$ in $\lambda x(xx)$ is replaced by a $\lambda$-term which has type $(0)0$ (i.e. it has a n.f. and when applied to an arbitrary $\lambda$-term which has a n.f. produces a $\lambda$-term which has a n.f.). So we can deduce immediately that $W_\ast$ has type $((0)0)0$.

Moreover since the termination properties of an application $XY$ depend not only on $X$ itself but also on $Y$, we can expect that a given $\lambda$-term $X$ possesses, in general, more than one type. So, for example, it will be shown that $W_\ast$ possesses (among infinitely many others) also type $(1)1$. The intuitive meaning of this type is straightforward. We notice here that if this functional characterization requires a very strong limitation on the possible arguments of $W_\ast$ (the subset of $\lambda$-terms which have type 1 is properly included in subsets of $\lambda$-terms which have each other type) it gives also the strongest characterization to the result of the application. Actually termination properties represented by types $((0)0)0$ and $(1)1$ (which are incomparable with respect to the equivalence and partial order rela-

\(^1\) For example $\lambda xx$ has type $(0)0$. 