A functional system with transfinitely defined types

Mariko YASUGI and Susumu HAYASHI

Introduction

The first author formulated a system of generalized terms (called term-forms) in [9] in order to provide with the foundations to the "consistency proofs." Technically, it is a device to realize a functional interpretation of a formalized intuitionistic analysis (called **ASOD** in [9]), which has transfinite recursive definitions of predicates along some well-ordered structures and bar induction applied to "admissible formulas".

Here we present a revised version of the system of term-forms, named **TRM** and consider **ASOD** in a more general setting, renaming it as **TRDB**. **TRM** is interesting by itself (aside from proof-theoretical necessity), and **TRDB** has its place among formal systems of constructive analysis.

Our major objectives in this article are to prove the normalization theorem in **TRM** and to distil the computational mechanism of **TRDB** via a translation of **TRDB** into **TRM**. The paper is designed to be nearly self-contained.

TRM is based on a kind of parametric types, called type-forms. Among the constructors of type-forms are the transfinite recursor (along some well-ordered structure), parametric abstraction, projection to parametric objects and the conditional definition (Section 2). Term-forms belonging to such type-forms are defined in terms of abstraction, application, conditional definition, primitive recursion and bar recursion (Section 4).

Section 1 furnishes the reader with some preparatory background.

Normalizability of type-forms is proved by the transfinite induction on the "degree" of type-forms (Section 3), and the same property of term-forms is proved in terms of the reducibility set, following Tait method (Section 5). The Church-Rosser property can be established by well-known arguments (Section 6). This completes the first of our objective.

Arithmetization of the normalization proof can be executed in **TRDB** with standard technique (Section7). The opposite direction, realization of

TRDB in **TRM** has been demonstrated in [9] as a consequence of the modified realizability (mr-)interpretation. We thus obtain the computational equivalence of **TRM** and **TRDB**. We also present, however, a direct translation of **TRDB** to **TRM**, associating formulas with type-forms and proofs with term-forms (Section 8). We can then complete our second objecticve.

Zucker has worked on the modified realizability interpretation of an extension of HA (Heyting arithmetic) with "twice iterated inductive definitions" applied to a subfamily of formulas. Interested readers can refer to [11].Our idea of **TRM** is an expansion of the "construction principle", which was proposed in [8], in order to interpret the transfinite induction up to ε_0. A type system which requires reductions is not novel. We have quoted [2] and [6] as references, although their systems are impredicative while ours is not.

We have taken much of notations, technical terms and proof techniques from [1], [3] and [5]. In the separate paper [10], we have presented interpretations of **TRM** in sets and in **TRDB**. We also plan to define an extension of **TRDB** which is complete with respect to the mr-translation.

1. Preliminaries

We first define formal expressions of number-theoretic objects and relations (in the basic language $\mathcal{BL}$), and then give an account of the continuity of functionals.

Definition 1.1. Our entire theory depends on a pre-supplied, primitive recursive well-ordered structure $\mathcal{I} \equiv (I, <_I)$ on natural numbers. Namely, we identify I with the set of natural numbers, and $<_I$ is a primitive recursive well-ordering on it. For the simplicity of arguments, we assume $\mid I \mid < \epsilon_0$, where $\mid I \mid$ represents the order type of I.

We assume $\mathcal{I}$ is accessible. That is, there is a functional μ (modulus of finiteness functional for $\mathcal{I}$) such that, given any $\{i_n\}_n$, a decreasing sequence from I, for any $m \geq \mu(\{i_n\})$, i_m is in fact empty.

Note. There can be more than one order structure $\mathcal{I}$ (See Definition1.1.), but we assume just one for the sake of simplicity.

(1) The language $\mathcal{BL}$ consists of the following.

(1.1) Propositional connectives.

(1.2) There are variables, say ${x_0}^n, {x_1}^n, {x_2}^n, \cdots, {x_p}^n, \cdots$, where $n = 0, 1, 2, \cdots$. A variable with superscript n is called an n-ary variable.

(1.3) Constant symbols for natural numbers, say 0, 1, 2,$\cdots$.

(1.4) Constant symbols for all the number-theoretic functions which are primitive recursive in function parameters.

(1.5) Constants for primitive recursive relations such as $=$ and $<$ (of numbers). We assume the elementary relations concerning the structure $\mathcal{I}$ (such as $<_I$) have symbols in