The Metatheory of UTT

Healfdene Goguen*

INRIA Sophia-Antipolis, 2004, route des Lucioles, B. P. 93
06902 Sophia-Antipolis Cedex, France

Abstract. This paper outlines the development of the metatheory for Luo's type theory UTT, the type theory implemented in the proof assistant LEao and containing as subsystems Martin-Löf's type theory and the Calculus of Constructions. The approach used is to define a typed operational semantics for the system and to establish the important metatheoretic properties, such as Church–Rosser, strong normalization and subject reduction, for this operational presentation of the theory. These properties are then transferred to the usual presentation by soundness and completeness results. This technique gives a new and simpler development of the metatheory for systems with dependent types and η-equality.

1 Introduction

Luo [16] proposes the type theory UTT as a unified system in which to specify, implement and verify programs. However, although he studies the properties of the subsystem ECC, he does not give a metatheoretic justification for the full theory UTT. This paper outlines our approach to giving such a justification, as developed fully in the author's thesis [9].

Because Martin-Löf's type theory and the Calculus of Constructions are subsystems of UTT, we have established the important metatheoretic results for these systems as well. UTT is also closely related to the type theory implemented in LEGO, so we have also given a theoretical basis for confidence in this proof assistant. The significant difference between the two is that UTT is formulated in Martin-Löf's Logical Framework, where the function space used to encode inductive types is outside the type theory; LEGO instead bases its inductive type mechanism upon the internal function space of the type theory. As far as we are aware, our work and Werner's development for the Calculus of Inductive Constructions [27] are the only justifications of complex type theories with general schemas for inductive types.

The central new idea in our approach to the study of metatheoretic properties is the use of the formal systems we call typed operational semantics, already introduced in the context of the simply typed lambda calculus [10]. A typed operational semantics is essentially a reduction to normal form for terms which are

* This material is based upon work supported by the North Atlantic Treaty Organization under a Grant awarded in 1993.
well-typed in the type theory. We can have different typed operational semantics encoding different notions of reduction to normal form.

Studying a type theory with typed operational semantics can be divided into two parts. First, we establish metatheoretic results about the semantics itself. In this paper we consider a variant of standard reduction, universal with respect to reduction strategies. Because of the formulation of this system, the rules of inference for the typed operational semantics provide a general induction principle for showing results relating reduction and well-typedness, such as Church–Rosser, strong normalization and subject reduction.

The second part of studying a type theory through typed operational semantics involves showing the equivalence of the usual presentation and the operational one. It is through the equivalence of the two presentations that we are able to transfer the metatheoretic results about the typed operational semantics to the usual presentation of the type theory. Of particular interest in the context of dependent types is that we use the new system to show strengthening and subject reduction. Although completeness of the semantics is straightforward, the soundness proof, that well-typed terms in the usual presentation are well-typed in the semantics, is a complex proof similar to traditional proofs of normalization.

Because of the complexity of the soundness proof for the typed operational semantics of \( \text{UTT} \) and its similarity to existing proofs of normalization, we shall give a very brief outline of this and instead concentrate on the properties of the typed operational semantics. The differences between our approach and the usual one are emphasized here. Specifically, although the progression of results about the type theory is similar to the usual approach [2, 13, 26], the system for which we show these results has not been studied before. Furthermore, this technique gives an elegant solution to two related problems encountered in normalizing type theories with dependent types, associated with \( \eta \)-equality, also studied by Coquand [4], Geuvers [8] and Salvesen [23], and labels on constants in type theories expressed in the Logical Framework, as encountered for example in Löfwall and Sjödin’s proof of normalization for Martin-Löf type theory [12]. We also show our results in Martin-Löf’s Logical Framework with abstractions with type labels, omitted in the original presentation because of problems with the metatheory.

2 \( \text{UTT} \)

The system \( \text{UTT} \) has been discussed extensively by Luo [14, 16] and is also presented in the author’s thesis [9]. We shall therefore limit ourselves to giving a brief overview of the system and its uses and presenting the rules of inference in Appendix A. We note, however, that the definition of schemas in the universe, Definition 16, is slightly different from Luo’s.

The calculus \( \text{UTT} \) is formulated in Martin-Löf’s Logical Framework [21], a formal system that Martin-Löf intended to be used as a framework for defining type theory. The Logical Framework is a weak theory that has functional types