Type system TS_0 and its models

Vladimir Voevodsky

Started June 16, 2011

Contents

1 Introduction

We consider the simplest type system TS where we can formulate the main homotopy notions and some of the resizing axioms. For that we need the standard dependent products, dependent sums, Martin-Lof identity types and a universe UU. In our specification we use pure contexts (without judgements attached) to represent types which normally would be "parked" in an auxiliary universe $UU1$ which can not be a part of any constructions. In our approach to specifications of type systems we do not need to introduce a name for this auxiliary universe.

We will build TS in several steps. The first type system we consider has only UU and dependent products and does not have any reduction rules (other than α-conversion i.e. the renaming of bound variables). It will be denoted TS_0.

2 Type system TS_0 and its models

1 System of expressions underlying TS_0

The system of expressions underlying TS_0 is the free system of expressions with the following "generators":

1. constant UU,
2. one-variable quantifier of valency 2 denoted by $\prod$,
3. one variable quantifier of valency 2 denoted by λ,
4. function symbol of valency 2 denoted by $eval$.

We use bold font to denote names of variables and the usual font to denote names of special symbols and expressions. We use the notation $\prod \mathbf{x} : R.S$ for $\prod$ which bounds variable x and has R and S as

its first and second branch respectively. Similarly we use $\lambda\,\mathbf{x} : R.f$ for λ which bounds $\mathbf{x}$ and has R and f as the first and second branch respectively. We use parenthesis to avoid ambiguity whenever necessary. For *eval* we use the standard convention when one writes $A\,B$ instead of $eval(A, B)$.

All expressions are considered modulo α-equivalence i.e. up to renaming of bound variables.

2 Contexts and judgements of TS_0

Below are the rules for the formation of contexts and judgements in TS_0. As usual Γ, Δ etc. denote an arbitrary context or a segment of a context and it is understood that any expression has as free variables only the variables which have been declared and assigned types strictly prior to the use of this expression. We use commas to separate variable declarations in a context. The notation $\mathbf{x} \notin \Gamma$ means that the variable name $\mathbf{x}$ has not been declared in Γ.

Empty context, diagonals

$$\frac{}{\vdash} \qquad \frac{\Gamma,\, \mathbf{x} : T,\, \Gamma' \vdash}{\Gamma,\, \mathbf{x} : T,\, \Gamma' \vdash \mathbf{x} : T}$$

Universe structure

$$\frac{\Gamma \vdash}{\Gamma,\, \mathbf{x} : UU \vdash} \qquad \frac{\Gamma \vdash T : UU}{\Gamma,\, \mathbf{x} : T \vdash} \qquad (\mathbf{x} \notin \Gamma)$$

Dependent Products

$$\frac{\Gamma,\, \mathbf{x} : R,\, \mathbf{y} : S \vdash}{\Gamma,\, \mathbf{z} : \prod\, \mathbf{x} : R.\, S \vdash} \qquad (\mathbf{z} \notin \Gamma)$$

$$\frac{\Gamma,\, \mathbf{x} : R \vdash s : S}{\Gamma \vdash \lambda\, \mathbf{x} : R.s : \prod\, \mathbf{x} : R.S} \qquad \frac{\Gamma \vdash f : \prod\, \mathbf{x} : R.S \quad \Gamma \vdash r : R}{\Gamma \vdash f\ r : S[\mathbf{x}/r]}$$

Dependent products and the universe

$$\frac{\Gamma \vdash R : UU \quad \Gamma,\, \mathbf{x} : R \vdash S : UU}{\Gamma \vdash \prod\, \mathbf{x} : R.\, S : UU}$$

Remark 2.1 One could re-write our specification using two universes UU and $UU1$ and only using judgements but not "pure" contexts. Then the system described so far will look very similar to the system $\lambda\underline{\omega}$ of Barndregt's λ-cube (see [1][p. 206]) which has the "signature" (loc.cit. p.214):

$$\mathcal{S} = (UU, UU1)$$

$$\mathcal{A} = ((UU, UU1))$$

$$\mathcal{R} = ((UU, UU, UU), (UU1, UU1, UU1))$$

and considered without β-reduction. However, we also have the inclusion $UU \subset UU1$ which takes us outside of the class of systems considered by Barendregt. We prefer the specification given above since it emphasizes the fact that the second universe $UU1$ can not be a part of any constructions.

Remark 2.2 As was pointed out by Dan Doel, in the two universe specification, it is possible to achieve the effect of having the inclusion $UU \subset UU1$ by introducing an additional rule to the $\mathcal{R}$ component of the signature of the form $(UU, UU, UU1)$. The idea is that since $\prod$ is the only type forming constructor any particular type expression which types to UU according to the repeated application of the (UU, UU, UU) rule can be re-typed to $UU1$ using the rule $(UU, UU, UU1)$ the last time the rule (UU, UU, UU) was used. This however requires a formal proof and leads to proliferation of rules in the case of more complex systems and we prefer to have a direct inclusion rule instead.

3 Proof that TS_0 is a type system

Let us recall the following definitions.

References

[1] H. P. Barendregt. Lambda calculi with types. In *Handbook of logic in computer science, Vol. 2*, volume 2 of *Handb. Log. Comput. Sci.*, pages 117–309. Oxford Univ. Press, New York, 1992.