Coq à la Tarski: a predicative calculus of constructions with explicit subtyping

Ali Assaf\textsuperscript{1,2}

\textsuperscript{1} INRIA Paris-Rocquencourt, Paris, France
\textsuperscript{2} École Polytechnique, Paris, France

The predicative Calculus of Inductive Constructions (pCIC), the theory behind the Coq proof system, contains an infinite hierarchy of predicative universes

\[ \text{Type}_0 \in \text{Type}_1 \in \text{Type}_2 \in \ldots \]

and an impredicative universe \( \text{Prop} \) for propositions, together with an implicit cumulativity relation

\[ \text{Prop} \subseteq \text{Type}_0 \subseteq \text{Type}_1 \subseteq \text{Type}_2 \subseteq \ldots \]

This gives rise to a subtyping relation \( \leq \) which is used in the subsumption rule

\[
\frac{\Gamma \vdash M : A \quad A \leq B}{\Gamma \vdash M : B}.
\]

Subtyping in Coq is implicit, and is handled by the kernel. An attempt to simplify the theory would be to make subtyping explicit, by inserting explicit coercions such as

\[ c_{0,1} : \text{Type}_0 \rightarrow \text{Type}_1 \]

and rely on a kernel that only uses the classic conversion rule

\[
\frac{\Gamma \vdash M : A \quad A \equiv B}{\Gamma \vdash M : B}.
\]

However, because of dependent types, coercions change the shape of the types and therefore interfere with type checking.

We present a formulation of the predicative calculus of constructions using Tarski-style universes \cite{4} where subtyping is explicit. Other such systems have been proposed in the past \cite{5,2,3}. However, they do not preserve equality: a term in the original Coq system can have many non-equivalent representations in the new system, which breaks typing. As a result, these systems lose some of the expressivity of Russell-style universes with implicit subtyping, and are therefore incomplete.

Our system fully preserves equality. By adding additional equations between terms, we ensure that every well-typed term in the original system has a unique canonical representation in our system. To our knowledge, this is the first time such work has been done for the full predicative calculus of constructions. We will also show how to orient the equations into reduction rules. This work can be used as a basis for embedding Coq in a logical framework like the \( \lambda \Pi \)-calculus modulo \cite{1}, implemented in Dedukti \cite{6}. 
References


