Cumulative inductive types in Coq

Amin Timany[1], Matthieu Sozeau[2], and Bart Jacobs[1]

[1] imec-Distrinet, KU Leuven, Belgium, `firstname.lastname@cs.kuleuven.be`
[2] Inria Paris & IRIF, France, `matthieu.sozeau@inria.fr`

In order to avoid well-know paradoxes associated with self-referential definitions, higher-order dependent type theories stratify the theory using a countably infinite hierarchy of universes (also known as sorts), $\texttt{Set} = \texttt{Type}_0 : \texttt{Type}_1 : \cdots$. Such type systems are called cumulative if for any type T we have that $T : \texttt{Type}_i$ implies $T : \texttt{Type}_{i+1}$. The predicative calculus of inductive constructions (pCIC) [2, 3] at the basis of the Coq proof assistant, is one such system.

Earlier work [4] on universe-polymorphism in Coq allows constructions to be polymorphic in universe levels. The quintessential universe-polymorphic construction is the polymorphic definition of categories: $\texttt{Record Category}_{\texttt{i,j}} := \{\texttt{Obj} : \texttt{Type}_{\texttt{i}}; \texttt{Hom} : \texttt{Obj} \to \texttt{Obj} \to \texttt{Type}_{\texttt{j}}; \cdots\}$.[1]

However, pCIC does not extend the subtyping relation (induced by cumulativity) to inductive types. As a result there is no subtyping relation between instances of a universe polymorphic inductive type. That is, for a category $\texttt{C}$, having both $\texttt{C} : \texttt{Category}_{\texttt{i,j}}$ and $\texttt{C} : \texttt{Category}_{\texttt{i}',\texttt{j}'}$ is only possible if $\texttt{i} = \texttt{i}'$ and $\texttt{j} = \texttt{j}'$. In previous work Timany et al. [5] extend pCIC to pCuIC (predicative Calculus of Cumulative Inductive Constructions). This is essentially the system pCIC with a single subtyping rule added to it:[2]

C-Ind

$$\frac{\begin{array}{c} I \equiv (\mathsf{Ind}(X : \Pi\vec{x} : \vec{N}.\ s)\{\Pi\vec{x_1} : \vec{M_1}.\ X\ \vec{m_1}, \ldots, \Pi\vec{x_n} : \vec{M_n}.\ X\ \vec{m_n}\}) \\ I' \equiv (\mathsf{Ind}(X : \Pi\vec{x} : \vec{N'}.\ s')\{\Pi\vec{x_1} : \vec{M'_1}.\ X\ \vec{m'_1}, \ldots, \Pi\vec{x_n} : \vec{M'_n}.\ X\ \vec{m'_n}\}) \\ \forall i.\ N_i \preceq N'_i \qquad \forall i,j.\ (M_i)_j \preceq (M'_i)_j \qquad \mathit{length}(\vec{m}) = \mathit{length}(\vec{x}) \qquad \forall i.\ X\ \vec{m_i} \simeq X\ \vec{m'_i} \end{array}}{I\ \vec{m} \preceq I'\ \vec{m}}$$

The two terms I and I' are two inductive definitions (type constructors[3]) with indexes of types $\vec{N}$ and $\vec{N'}$ respectively. They are respectively in sorts (universes) s and s'. They each have n constructors, the i^{th} constructor being of type $\Pi\vec{x_i} : \vec{M_i}.\ X\ \vec{m_i}$ and $\Pi\vec{x_i} : \vec{M'_i}.\ X\ \vec{m'_i}$ for I and I' respectively. With this out of the way, the reading of the rule C-Ind is now straightforward. The type $I\ \vec{m}$ is a subtype of the type $I'\ \vec{m}$ if the corresponding parameters of corresponding constructors in I are sub types of those of I'. In other words, if the terms $\vec{v}$ can be applied to the i^{th} constructor of I to construct a term of type $I\ \vec{m}$ then the same terms $\vec{v}$ can be applied to the corresponding constructor of I' to construct a term of type $I'\ \vec{m}$. Using the rule C-Ind above (in the presence of universe polymorphism) we can derive $\texttt{Category}_{\texttt{i,j}} \preceq \texttt{Category}_{\texttt{i}',\texttt{j}'}$ whenever $\texttt{i} \leq \texttt{i}'$ and $\texttt{j} \leq \texttt{j}'$.

The category theory library by Timany et al. [6] represents (relative) smallness and largeness of categories through universe levels. Smallness and largeness side-conditions for constructions are inferred by the kernel of Coq. In loc. cit. the authors prove a well-known theorem stating that any small and complete category is a preorder category. Coq infers that this theorem can apply to a category $\texttt{C} : \texttt{Category}_{\texttt{i,j}}$ if $\texttt{j} \leq \texttt{i}$ and thus not to the category $\texttt{Types@\{i\}} : \texttt{Category}_{\texttt{i,i+1}}$ of types at level $\texttt{i}$ (and functions between them) which is complete but not small. In a system with the rule C-Ind we have $\texttt{Types@\{i\}} : \texttt{Category}_{\texttt{k,l}}$ for $\texttt{i} < \texttt{k}$, $\texttt{i} + 1 < \texttt{l}$ and $\texttt{l} \leq \texttt{k}$. However, subtyping would not allow for the proof of completeness of

[1] Records in Coq are syntactic sugar for an inductive type with a single constructor.
[2] The rule C-Ind is slightly changed here so that it applies to template polymorphism explained below.
[3] Not to be confused with constructors of inductive types

`Types@{i}` to be lifted as required. Intuitively, that would require the category to have limits of all functors from possibly larger categories.

Template Polymorphism Before the addition of full universe polymorphism to Coq, the system enjoyed a restricted form of polymorphism for inductive types, which was since coined template polymorphism. The idea was to give more precise types to applications of inductive types to their parameters, so that e.g. the inferred type of `list nat` is $\mathtt{Type}_0$ instead of $\mathtt{Type}_i$ for a global type level i.

Technically, consider an inductive type I of arity $\forall \vec{P}, \vec{A} \to s$ where $\vec{P}$ are the parameters and $\vec{A}$ the indices. When the type of the n-th parameter is $\mathtt{Type}_l$ for some level l and l occurs in the sort s (and nowhere else), the inductive is made parametric on l. When we infer the type of an application of I to parameters $\vec{p}$, we compute its type as $\forall \vec{A} \to s[l'/l]$ where $p_n : \mathtt{Type}_{l'}$, using the actual inferred types of the parameters.

This extension allows to naturally identify $\mathtt{list}(\mathtt{nat} : \mathtt{Set})$ and $\mathtt{list}(\mathtt{nat} : \mathtt{Type}_i)$ by convertibility, whereas with full universe polymorphism when comparing to $\mathtt{list@}\{\mathtt{Set}\}\ (\mathtt{nat} : \mathtt{Set})$ and $\mathtt{list@}\{i\}\ (\mathtt{nat} : \mathtt{Type}_i)$ with $\mathtt{Set} < i$ we would fail as equating i and $\mathtt{Set}$ is forbidden. With our new rule, this conversion will be validated as these two `list` instances become convertible. Indeed, convertibility on inductive applications will now be defined as cumulativity in both directions and in this case $\mathtt{list@}\{i\}$ cumulativity imposes no constraint on its universe variable. This change will allow a complete compatibility with template polymorphism.

Consistency and Strong Normalization The model constructed for pCIC by Lee et al. [3] is a set theoretic model that for inductive types considers the (fixpoints of the function generated by) constructors applied to all applicable terms. Therefore, the model readily includes all elements of the inductive types including those added by the rule C-Ind. Hence it is only natural to expect (and it is our conjecture that) the same model proves consistency of Coq when extended with the rule C-Ind. We are investigating using the abstract framework of B. Barras [1] to prove Strong Normalization with this extension.

Implementation The rule C-Ind above can be implemented in Coq very efficiently. The idea is that as soon as we define an inductive type, we compare two fresh instances of it (with two different sets of universe variables) to compute the set of constraints necessary for the subtyping relation to hold on different instances of that inductive type. Subsequent comparisons during type checking/inference will use these constraints. It is our plan to implement the rule C-Ind for the next release of Coq (Coq 8.7) and accordingly remove support for template polymorphism.

References

[1] Bruno Barras. *Semantical Investigation in Intuitionistic Set Theory and Type Theoris with Inductive Families.* PhD thesis, University Paris Diderot – Paris 7, 2012. Habilitation thesis.

[2] Coq Development Team. Coq reference manual, 2016. Available at https://coq.inria.fr/doc/.

[3] Gyesik Lee and Benjamin Werner. Proof-irrelevant model of CC with predicative induction and judgmental equality. *Logical Methods in Computer Science*, 7(4), 2011.

[4] Matthieu Sozeau and Nicolas Tabareau. Universe polymorphism in Coq. In *Interactive Theorem Proving - 5th International Conference, ITP 2014, Proceedings*, pages 499–514, 2014.

[5] Amin Timany and Bart Jacobs. First steps towards cumulative inductive types in CIC. In *Theoretical Aspects of Computing - ICTAC 2015, Proceedings*, pages 608–617, 2015.

[6] Amin Timany and Bart Jacobs. Category theory in coq 8.5. In *Conference on Formal Structures for Computation and Deduction, FSCD 2016, Proceedings*, pages 30:1–30:18, 2016.