A Parametric CPS to Sprinkle CIC with Classical Reasoning

Pierre-Marie Pédrot

Dependent type theory was introduced by Martin-Löf as an alternative foundation for mathematics, stemming from the then-nascent proof-as-program correspondance [8]. Since its inception, it has been designed as an intrinsically intuitionistic system.

Meanwhile, in a parallel research path, the proof-as-program correspondance was extended to classical logic by means of control operators [5]. Such operators can be compiled away through a program translation known as continuation-passing style.

A natural question that arises is the compatibility of dependent type theory with classical logic, which would typically be implemented using control operators. Twenty years ago, Barthe et al. described a pure type system equipped with such operators [1]. This system requires a strong stratification of the logic, with types living in a pure world. Furthermore, it has also been noticed that dependent elimination does not mix well with computational classical logic [2, 6], a phenomenon that casts a doubt on the feasibility of a dependent CPS.

We propose an alternative approach inspired by our recent work on mixing effects with dependent type theory [9]. Our work fits in the paradigm of syntactical models that we have been advocating for [4], *i.e.* the justification of a given type theory by compilation into a more standard type theory. Here, our target theory is CIC, the calculus of inductive constructions, and our source theory is CIC extended with a modality $\|\cdot\| : \square_i \to \square_i$ on which classical reasoning is allowed.

The syntactical translation is intuitively an enrichment of the Lafont-Streicher-Reus CPS [7] with an additional layer of parametricity in the style of Lasson and Bernardy [3]. It is a close variant of our parametric effectful translation described in our latest paper. Essentially, all the intuitionistic content is pushed back into the parametricity layer. For instance, the translation of a type is somehow a doubly-negated type, which is irrelevant, together with a parametricity proof which embeds the actual type hidden inside this classical type. Similarly, a boolean is translated as a doubly-negated boolean together with a parametricity proof which represents the actual boolean. That is, $[\![\mathbb{B}]\!] \equiv \Sigma b : \neg\neg\mathbb{B}.\, \mathbb{B}_\varepsilon\ b$ where $\mathbb{B}_\varepsilon$ is inductively defined as follows.

$$\begin{array}{l}\texttt{Inductive}\ \mathbb{B}_\varepsilon : \neg\neg\mathbb{B} \to \square \\ \mid \texttt{true}_\varepsilon : \mathbb{B}_\varepsilon\ (\lambda k.\, k\ \texttt{true}) \\ \mid \texttt{false}_\varepsilon : \mathbb{B}_\varepsilon\ (\lambda k.\, k\ \texttt{false})\end{array}$$

Dependent elimination is recovered by a simple pattern-matching over the second component. Likewise, one can retrieve a type out of a parametric type by extracting it from the second component. Finding a proper fixpoint in order to type the rule $\square_i : \square_{i+1}$ through the translation is a bit subtle as the Lafont-Streicher-Reus CPS handles counter-types (that is, stacks) as first-class citizens, which makes the interpretation of universes slightly technical.

Given a type A, the classical modality $\|\cdot\|$ is defined as the type whose elements are the same as A, but whose parametricity relation is the full relation. That is, every classical proof of A is an inhabitant of $\|A\|$. The propositional subset of CIC can be readily embedded in modal types by following the standard CPS translation. Dependent elimination is not interpreted through the modality. Furthermore, one can define in the source theory the following operators that allow to reason classically on modal terms.

- A return function $\texttt{return} : \Pi A : \square.\, A \to \|A\|$.
- The control operator $\texttt{callcc} : \Pi A\ B : \square.\, ((A \to \|B\|) \to A) \to \|A\|$.

- The modal excluded middle $\texttt{em} : \Pi A : \Box.\, \|A + \neg A\|$.

Note how $\|\cdot\|$ is inserted in places requiring backtrack. Interestingly, $\|\cdot\|$ does not form a monad because there is no `bind` operator, and it is thus more complex than a double negation. For instance, there exists a very weak form of choice as one can write a term $\texttt{choice} : \Pi A\, B.\, (\Pi x : A.\, \|B\|) \to \|\Pi x : A.\, B\|$ which amounts computationally to a projection. Such a term would not exist in vanilla CIC if $\|\cdot\|$ was to be interpreted as a double negation, as double-negation shift is not provable.

This translation allows therefore to justify an extension of CIC with a weak form of classical reasoning, namely, excluded middle and Peirce's law on modal types, but without dependent elimination on such types. As a form of CPS translation, it can help understanding type-preserving compilation of CIC into a lower-level representation, expliciting the rôle of stacks. It remains to understand the interaction of this translation with other extensions of CIC such as *e.g.* the univalence axiom.

A very rough proof-of-concept implementation of this translation can be found in the following file: `https://github.com/CoqHott/coq-effects/blob/master/theories/misc/CPS.v`.

References

[1] G. Barthe, J. Hatcliff, and M. H. Sørensen. A notion of classical pure type system. *Electr. Notes Theor. Comput. Sci.*, 6:4–59, 1997.

[2] G. Barthe and T. Uustalu. CPS translating inductive and coinductive types. In *Proceedings of Partial Evaluation and Semantics-based Program Manipulation*, pages 131–142. ACM, 2002.

[3] J.-P. Bernardy and M. Lasson. *Realizability and Parametricity in Pure Type Systems*, pages 108–122. 2011.

[4] S. Boulier, P.-M. Pédrot, and N. Tabareau. The next 700 syntactical models of type theory. In *Proceedings of Certified Programs and Proofs*, pages 182–194. ACM, 2017.

[5] T. G. Griffin. A formulae-as-types notion of control. In *In Conference Record of the Seventeenth Annual ACM Symposium on Principles of Programming Languages*, pages 47–58. ACM Press, 1990.

[6] H. Herbelin. On the degeneracy of sigma-types in presence of computational classical logic. In P. Urzyczyn, editor, *Seventh International Conference, TLCA '05, Nara, Japan. April 2005, Proceedings*, volume 3461 of *Lecture Notes in Computer Science*, pages 209–220. Springer, 2005.

[7] Y. Lafont, B. Reus, and T. Streicher. Continuation semantics or expressing implication by negation, 1993.

[8] P. Martin-Löf. An intuitionistic theory of types, 1972.

[9] P.-M. Pédrot and N. Tabareau. An effectful way to eliminate addiction to dependence. In *LICS '17*, 2017.