Checking the type safety of rewrite rules

- Lab: LSV, Cachan, France
- Team: Deducteam
- Advisor: Frédéric Blanqui (INRIA)

Context. Lambdapi is a new proof assistant based on a logical framework called the $\lambda\Pi$-calculus modulo rewriting, which is an extension of the simply-typed λ-calculus (the basis of functional programming languages like OCaml or Haskell) with dependent types (e.g. vectors and matrices of some given dimension) and an equivalence relation on types generated by user-defined rewrite rules [1]. Thanks to rewriting, Lambdapi allows the formalization of proofs that cannot be done in other proof assistants. However, before adding a new rewrite rule $l \to r$ in the system, it is necessary to check that it preserves typing, that is, if l is of type T, then r is of type T too. It is a property easy to check with simple types but undecidable with dependent types [2, 3].

Goal. The goal of this internship is to work on a new algorithm based on the use of a Knuth-Bendix completion procedure for transforming equality constraints generated by typing into rewrite rules. This algorithm can use the fact that some functions are injective but we currently have no criterion for deciding whether a function defined by rewrite rules is injective. So, a subgoal of this internship could be to find out some sufficient condition for injectivity and implement it in Lambdapi.

Workplan.

- study a new algorithm for checking that a rewrite rule preserves typing
- prove its correctness
- implement it in Lambdapi
- define a sufficient condition for injectivity
- implement it in Lambdapi

Requirements. Some basic knowledge of functional programming (OCaml or Haskell), rewriting theory and dependent type theory is welcome but not necessary.

References

[1] A. Assaf, G. Burel, R. Cauderlier, D. Delahaye, G. Dowek, C. Dubois, F. Gilbert, P. Halmagrand, O. Hermant, and R. Saillard. Dedukti: a logical framework based on the $\lambda\Pi$-calculus modulo theory, 2016. Draft.

[2] F. Blanqui. Definitions by rewriting in the calculus of constructions. *Mathematical Structures in Computer Science*, 15(1):37–92, 2005.

[3] R. Saillard. *Type checking in the Lambda-Pi-calculus modulo: theory and practice*. PhD thesis, Mines ParisTech, France, 2015.