Modern user interface for interactive theorem proving

- Lab: LSV, Cachan, France
- Team: Deducteam
- Advisor: Frédéric Blanqui (INRIA) and Emilio Gallego (Mines ParisTech)

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 (e.g. simplicial sets of infinite dimensions).

However, for developing large proofs, it is essential to have a good interface. Interactive theorem proving is built on the mutual interaction between a human and a prover. The human will submit a candidate proof, and the proof assistant will confirm or reject the user proposal. Building large proofs is a very difficult task, and users do require large amount of help from the tools. Searching, completion, project management, are all essential to the successful development of large proofs.

Goal. The goal of this internship is to develop a Lambdapi plugin for the VSCode or Emacs editors, based on the Language Server Protocol (LSP).

Adding features like auto complete, go to definition, or documentation on hover for a programming language takes significant effort. Traditionally this work had to be repeated for each development tool, as each tool provides different APIs for implementing the same feature.

A Language Server is meant to provide the language-specific smarts and communicate with development tools over a protocol that enables inter-process communication.

The idea behind the Language Server Protocol (LSP) is to standardize the protocol for how such servers and development tools communicate. This way, a single Language Server can be re-used in multiple development tools, which in turn can support multiple languages with minimal effort.

The Lambdapi server is already implemented (subdirectory `lp-lsp`) [2, 3], and a Lambdapi plugin for Atom called atom-dedukti has already been developed last summer [4], which could be useful.

Workplan.

- syntax coloring
- enable Unicode symbols
- display error messages and their locations
- display unsolved goals
- for an unsolved goal, display the assumptions
- provide buttons and short cuts for going forward or backward in a proof
- display informations about symbols or text selections (e.g. type, definition, rewrite rules)
- propose to LSP developers an extension of LSP for interactive proof development

Requirements. Knowledge of JavaScript and JSON. Knowledge of LISP required only if one wants to develop the Emacs plugin.

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] E. J. Gallego-Arias. SerAPI: Machine-Friendly, Data-Centric Serialization for Coq, 2016.

[3] E. J. Gallego-Arias, B. Pin, and P. Jouvelot. jsCoq: towards hybrid theorem proving interfaces. In *Proceedings of the 12th Workshop on User Interfaces for Theorem Provers*, 2016.

[4] I. Lachheb. Une interface pour Dedukti. `https://hal.inria.fr/hal-01898401`, 2018. Internship report.