M2 internship: Semantics of a low-level language for verified computer algebra

Guillaume Melquiond
Inria, Université Paris-Saclay

1 Context

The Fresco project aims at turning the Coq proof assistant into a competitive tool for doing verified computer algebra. To do so, a two-language architecture is envisioned. First, there is an interpreted high-level language, with a functional mindset, which is the main programming language used by the users to input their queries. Second, there is a compiled low-level imperative language, so that experts can implement high-performance basic blocks. The added value of Coq is that algorithms written using these languages can eventually be formally verified, and their results can thus come with a high level of confidence, to the point where they could even be used inside formal proofs without endangering the consistency of the formal system.

2 Objectives

This 6-month internship at the Master 2 level aims at designing a first draft of the low-level language, which should satisfy a few constraints:

- It should provide the primitives and data structures needed to implement state-of-the-art libraries used in computer algebra systems (e.g., GMP, BLAS): arrays, matrices, pointers, and so on.
- It should be safe by construction. Even if the user did not verify the functional correctness of a low-level routine, executing it in Coq should not break the system by corrupting memory.
- It should be translatable to one of the intermediate languages of the CompCert C compiler, so as to leverage the huge work that went into this formally verified compiler [5].
- It should provide constructs that would ease the functional verification of algorithms, e.g., function contracts, array separation, ghost code.
- It should eventually be interfaced with the high-level language.

During the internship, the work will focus on the abstract syntax of the language as well as its semantics, on properties such as subject reduction, on writing first an interpreter for it and then a CompCert-based compiler, and on exercising it with some basic algorithms such as block matrix multiplication or a segmented prime sieve. A large amount of bibliographic work is expected, in order to understand what semantics and compilation of a low-level language entails, as done in CompCert, but also to understand what it means to formally verify low-level code, as done in Fiat-Crypto [3], Iris [4], VST [2], Why3 [1], and so on.

3 Location

Assuming the pandemic recedes, the internship will take place at the Formal Methods Laboratory (LMF) of Université Paris-Saclay, in the Inria Team Toccata, which is dedicated to writing tools for deductive program verification.
4 Prerequisites

Knowledge of the Coq proof assistant or of a closely-related formal system (e.g., Lean) is mandatory. Knowledge of programming language theory, semantics, compilation, and program verification, is also needed. No knowledge about computer algebra systems is needed.

5 Continuing into a PhD thesis

Funding for a PhD thesis has been secured, in case the internship is going well and the student wishes to stay in the academic world and keep working on this topic. The thesis would be in direct continuation. It would aim at making the low-level language more mature, at proving the preservation of semantics when compiling to CompCert, at adding tools (or interfacing with existing ones, e.g., VST, Iris) to enable deductive program verification, and so on.

References


