Mission confiée

The Celtique team develops mechanised semantics for full-fledged programming languages focusing on the source C language and JavaScript. In addition, the Celtique team conducts mechanization efforts related to the implementation of source languages. This includes the formal study of intermediate representations used in optimizing compilation chains, or runtime environment components.

A particular effort is dedicated to the semantic modeling and analysis of intermediate representations such as those found in modern optimizing compilers, e.g. SSA and variants thereof. We build on the CompCert C compiler [1], which is mainly developed by X. Leroy from the EPI Gallium. CompCert is an industrial-strength C compiler which is both programmed and proved correct inside the Coq proof assistant. This is a flagship of formal verification which is the cornerstone of national (ANR Verasco) and international research projects (NSF DeepSpec).

The actual CompCert C Compiler does not use SSA-based optimization techniques. In the Celtique team, we study this particular techniques through the CompCertSSA middle-end [2], that we started to develop in the team several years ago now. CompCertSSA extends CompCert with a optimizing middle-end based on the SSA (Static Single Assignment) form. Currently, the compiler has satisfying performance in terms of compile-time and running-time of generated code, but there are a couple of remaining open problems we would like to explore.

In particular, we would like to improve it with techniques relying on the explicit representation of data and control-dependencies of programs. It improves on the initial SSA form, which was initially seen as a simple control-flow-graph. Examples of such techniques include the Program Dependence Graph, Sink-of-Nodes [3], of Monadic Gated SSA [4].

The challenge here is to shift from an operational setting where an abstract machine is running instructions one after another, to a more equational view of program behaviours, where data and control dependencies are what drive the program execution. Additionally, we will need to design adequate reasoning and proof techniques that scale to large mechanized developments such as realistic verified compilers.

We have started to work in this direction on SSA variants such as the Sea-of-Nodes [5], and we would like to start working on the monadic gated SSA form (4).

A short-to-middle term objective is the formal definition of an equational semantics for SSA, perhaps reusing ideas from the synchronous languages community where programs are seen as value streams generators. In the context of verified compilation, one of the main challenges is to be able to relate such an equational semantics back to its operational version, which is essentially what is actually implemented by the compiler backend. The long-term goal is to make this semantic approach a foundation of verified program optimizers, or verified translation validators, embedded in optimizing verified compilers.

References

Principales activités

- Design and implement efficient validation techniques for SSA-based optimizations
- Semantic characterization of the validation process, proof of correctness
- Contribute to the state-of-the-art of compilers formal verification
- Disseminate new contributions in international conferences or journals

Compétences

- PhD in Computer Science
- Background in optimizing compilers techniques
- Background in SSA-based optimization techniques
- Background in formal verification in interactive proof assistants
- Fluent in English
- Autonomy
- Ability to work in team

Avantages

- Subsidized meals
- Partial reimbursement of public transport costs

Rémunération

Monthly gross salary amounting to 2 653 euros