Formal Verification of Code Generators for Modeling Languages

(Invited Talk)

Xavier Leroy
Inria, Paris, France
xavier.leroy@inria.fr

Automatic code generation from modeling or domain-specific languages, also known as model-driven code generation, is known to improve software quality and is widely used in industry. At the same time, automatic code generators, just like compilers in general, face miscompilation issues, where the produced code fails to implement the meaning of the source model or domain-specific program.

The formal verification of compilers aims at eradicating miscompilation issues by applying program proof techniques to the compilers themselves. The CompCert and CakeML projects demonstrate the efficiency of compiler verification in the case of compilers for the C and ML languages, respectively.

This talk will discuss the applicability and efficiency of formal compiler verification techniques to code generators for modeling languages and domain-specific languages. I will describe joint work with T. Bourke, L. Brun, P. É. Dagand, M. Pouzet and L. Rieg on the formal verification of a code generator for the Lustre reactive language to the CompCert subset of the C language. Lustre is the core language of the SCADE model-based environment for the development of critical embedded software and a nice example of a domain-specific language that is widely used for modeling, programming, andverification purposes.

Owing to the highly declarative nature of Lustre and to its elegant, mathematical semantics, the verification of a Lustre-to-C code generator is both a challenge and a pleasure, as I will try to illustrate. This verification also renews interest in earlier approaches that were set aside by lack of confidence in the code generator, such as aggressive optimizations and source-level static analysis.

Submitted to:
VPT 2018