2-year Postdoc Position

Compilation Techniques for Improving Expressivity and Performance of E-ACSL, a Runtime Verification Tool for C Programs

Keywords: runtime verification, program transformation, compilation, static analysis

Programming Languages: OCaml, C

Tools: Frama-C, E-ACSL

Context: CEA LIST, Software Security Lab

The Software Reliability Laboratory (LSL) at CEA LIST has an ambitious goal: help designers, developers, and validation experts ship high-confidence systems and software. Objects in our surroundings are getting more and more complex, and we have built a reputation for efficiently using formal reasoning to demonstrate their trustworthiness. Within the CEA LIST Institute, LSL is dedicated to inventing the best possible means to conduct formal verification. We design methods and tools that leverage innovative approaches to ensure that real-world systems can comply with the highest safety and security standards. In doing so, we get to interact with the most creative people in academia and the industry.

Our organizational structure is simple: those who pioneer new concepts are the ones who get to implement them. We are a forty-person team, and your work will have a direct and visible impact on the state of formal verification. CEA LIST’s offices are located at the heart of Campus Paris Saclay, in the largest European cluster of public and private research.

Work Description

Our team develops Frama-C [5] (http://frama-c.com), a code analysis platform for C programs which provides several collaborative analyzers as plug-ins. Frama-C itself is developed in OCaml. Frama-C allows the user to annotate C programs with formal specifications written in the ACML specification language [1]. Frama-C can then ensure that a C program satisfies its formal specification by relying on several techniques including abstract interpretation, weakest preconditions calculus, and runtime verification.

E-ACSL is the Frama-C plug-in dedicated to runtime verification [7]. It converts a C program extended with formal annotations written in a subset of ACML into a new C program which checks the validity of annotations at runtime: the program execution stops whenever one annotation is violated, or behaves in the same way than the input program if all its annotations are valid [8]. One key feature of E-ACSL is the expressivity of its specification language [2] which allows the user to describe powerful safety and security properties. Another key feature is the efficiency of the generated code which relies on a custom memory library [9] and dedicated static analyses [3, 4].

However E-ACSL is still a research tool and lots of improvements can be done both to extend its expressivity and to improve its efficiency. The goal of the hired postdoc will be to transform this promising prototype into a powerful runtime verification tool usable to efficiently verify industrial-size C programs. To reach this goal, (s)he shall:

- support the missing features of the specification language (e.g. memory separation, frame condition, real numbers, see [6]);
• improve the compilation scheme to reduce the memory footprint of the generated code;
• adapt and implement compilation techniques to optimize the efficiency of the generated code;
• design novel dedicated fast static analyses to minimize the generated code;
• evaluate the benefits of these improvements on concrete use cases (provided by LSL’s academic and industrial partners).

References


Application

Knowledge in at least one of the following fields is required:

• OCaml programming (at least, functional programming)
• C programming
• runtime verification
• compilation
• static analysis
• semantics of programming languages
• formal specification

Salary: academic competitive (vary w.r.t. diploma and former experience)

Availability: From February 2019; a 3+-month procedure for administrative and security purposes is required

Contact: Julien Signoles (julien.signoles@cea.fr)

Please join a detailed CV, a motivation letter and possibly reference letters.