2018-00228 - Post-doc ``A Formally Verified Symbolic Interpreter for the CoLiS Language''

About Inria

Inria, the French National Institute for computer science and applied mathematics, promotes "scientific excellence for technology transfer and society". Graduates from the world's top universities, Inria's 2,700 employees rise to the challenges of digital sciences. With its open, agile model, Inria is able to explore original approaches with its partners in industry and academia and provide an efficient response to the multidisciplinary and application challenges of the digital transformation. Inria is the source of many innovations that add value and create jobs.

About the research centre or Inria department

Located at the heart of the main national research and higher education cluster, member of the Université Paris Saclay, a major actor in the French Investments for the Future Programme (Idex, LabEx, IRT, Equipes) and partner of the main establishments present on the plateau, the centre is particularly active in three major areas: data and knowledge, safety, security and reliability; modelling, simulation and optimisation (with priority given to energy).

The 450 researchers and engineers from Inria and its partners who work in the research centre's 31 teams, the 100 research support staff members, the high-level equipment at their disposal (image walls, high-performance computing clusters, sensor networks), and the privileged relationships with prestigious industrial partners, all make Inria Saclay Île-de-France a key research centre in the local landscape and one that is oriented towards Europe and the world.

Context

This is an offer for a 12 months post-doctoral position working in the context of the CoLiS project (http://colis.irif.univ-paris-diderot.fr/) joint with IRIF lab at University Paris-Diderot, Inria team Toccata at Inria-Saclay and Inria team Links at Inria-Lille. This post-doc grant is taking place in the Toccata team, whose offices are located in the campus of Université Paris-Sud at Orsay.

The CoLiS project aims at applying formal verification techniques to the analysis of Shell scripts, in particular those that are used for installation of software packages. These scripts play a central role in the installation of computers under UNIX and Linux. They interact with the state of the operating system, in particular at the level of the file system. Mistakes in such scripts may have damaging impacts since they are executed at critical moments, in administrator mode, in multiple situations probably not all taken into account by the author of the script. The CoLiS project aims in particular at analyzing scripts of the Debian GNU-Linux distribution.

A first step of CoLiS was the definition of shell-like dedicated language (itself named `\`CoLiS\`) to describe the actions operated by installation scripts. Unlike the POSIX shell, the syntax and the semantics of the CoLiS language were defined using a formal approach [1], within the Why3 environment (http://why3.lri.fr, [2,3]). Why3 is an environment for the formal specification and the proof of programs, developed in the Toccata team where the post-doc position is offered. A reference interpreter of the CoLiS language was developed in Why3 and formally proved correct with respect to the semantics [1].

Another initial step of the CoLiS project is the implementation of a parser of the POSIX shell language, in order to analyze the real installation scripts. This syntactic analyzer already allowed us to compute statistical data on the full Debian distribution to figure out which constructs were widely used and which were not. A translation tool of these scripts into the CoLiS language is in progress.

Assignment

The initial analysis of the needs lead us to decide that a useful method for analyzing scripts semantics in a fully automated way is symbolic execution. The first goal of this post-doc position is to participate to the design of this symbolic interpreter, in a formal way using Why3, building up on the already formalized concrete interpreter. A formal proof of the correctness of the symbolic execution with respect to the concrete semantics is expected. Also, a symbolic interpreter is tightly connected with a constraint language to describe symbolic states, in that case it will...
be a constraint language of the file system tree. The constraint language we need is still to be
determined, in cooperation with the partners of the CoLiS project. The symbolic interpreter to design
in this post-doc position is excepted to be somehow generic, i.e. modular, with respect to the
constraint language, in the same vein that formally verified abstract interpreters were recently
approach will allow us to automatically infer for each script what are
the preconditions on the file system (such as assumed existence of some files or directories) that are
needed to guarantee execution without errors. It should also automatically compute post-conditions
that are provided by the script execution of a package, that thus may be assumed by the scripts of
packages that depend on the former package. The second goal of the job is to contribute to the set-
up of an infrastructure for automatic execution of the symbolic interpreter on all Debian installation
scripts, in an adequate order with respect to packages dependencies. This approach should be also
used for proving properties about composition of scripts, such as
showing that installation followed by de-installation is more or less the identity.

[1] François Bobot, Jean-Christophe Filliâtre, Claude Marché, and Andrei Paskevich. Let's verify this
Felleisen and Philippa Gardner, editors, Proceedings of the 22nd European Symposium on
2013.
programming language. In Andrei Paskevich and Thomas Wies, editors, 9th Working Conference on
Verified Software: Theories, Tools and Experiments (VSTTE), Lecture Notes in Computer Science,
Heidelberg, Germany, July 2017. Springer.
formally-verified C static analyzer. In 42nd ACM SIGPLAN-SIGACT Symposium on Principles of
Programming Languages, pages 247–259, Mumbai, India, January 2015. ACM.

Main activities

Main activities (5 maximum):

- Design a symbolic execution engine for CoLiS programs
- Formalize this engine in Why3 and prove it correct
- Design an experimental platform for analysis of installation scripts
- Write reports and submit research articles

All these activities will be conducted in collaboration with other CoLiS project members. Regular
working groups are organized at IRIF laboratory in Paris

Skills

The candidate should have a PhD thesis in the domain of formal methods for software engineering. A
basic knowledge of an interactive (such as Coq, Isabelle, PVS) or autoactive verification system (such
as Why3, Dafny, KeY) is expected. Some taste in implementing using functional programming (OCaml,
Haskell, etc.) would be a plus.

Benefits package

- Subsidised catering service
- Partially-reimbursed public transport
- Social security
- Paid leave
- Flexible working hours
- Sports facilities

Remuneration

Monthly gross salary : 2.653 euros