2018-00432 - Post-doctoral position: Compiling recursive functions to inductive definition in Coq

Contract type : Public service fixed-term contract
Renewable contract : Oui
Level of qualifications required : PhD or equivalent
Function : Post-Doctoral Research Visit

About the research centre or Inria department
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.

Context
The CISC “Certification of IoT Secure Compilation” project, which is funded by ANR, the French National Research Agency, aims to investigate multitier languages and compilers to build secure IoT applications with privacy guarantees. The goal of the project is to define language, semantics, attacker models, and policies for the IoT and investigate the automatic implementation of privacy and security policies. The consortium includes 10 researchers and professors from all over France. This job description concerns one of the open positions within this project.

Assignment
Many formal definitions can be expressed in Coq either as a recursive function or as an inductive definition. The former is more concise and supports evaluation, while the latter is generally better suited for conducting proofs, especially when case analysis is involved. This choice recursive-vs-inductive definition appears in particular when defining the semantics of a programming language.

Isabelle/HOL provides a tool to extract a recursive function from an inductive definition. However, it appears often more useful in practice to go in the other direction, because the recursive definition is more concise. The “Function” feature of Coq aims at automatically translating a recursive definition into its inductive counterpart; however, this translation involves a significant amount of duplication in the inductive definition, making the inductive version impractical to work with in practice. Recent work on the pretty-big-step semantics suggests a systematic way to translate recursive definitions into equivalent, linear-sized, practical inductive definitions.

The goal of this project is to investigate this approach and develop a tool that takes as input a recursive function, written either in Coq or in purely-functional OCaml, and produces as output an inductive definition well-suited for interactive proofs. This tool would furthermore provide specific support for recursive definitions written in monadic style (involving an effect and/or exception monad), so as to streamline the inductive definition produced.

Main activities
The concrete outcome of the project consists of:

1. a Coq plugin to turn Coq recursive definitions into inductive ones, this plugin should be useful to all Coq users,
2. a tool for generating pretty-big-step inductive semantics from recursive semantics, this tool should be useful to many practitioners of “formal metatheory”,
3. a translator from pure-OCaml to Coq, this tool could provide a practical input syntax for formally-verified purely-functional programs,
4. an application of these techniques to a motivating, real-world, large-scale example, namely the JSExplain semantics (https://github.com/jscert/jsexplain).

Skills
We seek candidates with a solid background in Computer Science, in particular in the Coq proof assistant, in formal semantics, and in program analysis.
Fluent English is required, both oral and written. Knowledge of French is not required.

**Benefits package**
- Subsidised catering service
- Partially-reimbursed public transport
- Sports facilities

**Remuneration**
Monthly gross salary amounting to 2,653 euros