Development of Correct-by-Construction Functional Parallel Programs

Frédéric Loulergue
Université d'Orléans – Laboratoire d'Informatique Fondamentale d'Orléans
45067 Orléans Cedex 2, France
Frederic.Loulergue@univ-orleans.fr

SAC 2016 Tutorial Proposal

Summary

With the current generalisation of parallel architectures and increasing requirement of parallel computation arises the concern of applying formal methods, which allow specifications of parallel and distributed programs to be precisely stated and the conformance of an implementation to be verified using mathematical techniques. However, the complexity of parallel programs, compared to sequential ones, makes them more error-prone and difficult to verify.

This calls for a strongly structured form of parallelism, which should not only be equipped with an abstraction or model that conceals much of the complexity of parallel computation, but also provides a systematic way of developing such parallelism from specifications for practically nontrivial examples.

Program calculation is a kind of program transformation based on the theory of constructive algorithms. An efficient program is derived step-by-step through a sequence of transformations that preserve the meaning and hence the correctness. With suitable data-structures, program calculation can be used for writing parallel programs.

The SYDPACC system is a set of libraries for the proof assistant Coq that allows to write naive (i.e. inefficient) functional programs then to transform them into efficient versions that could be automatically parallelised within the framework before being extracted from Coq to code in the functional language OCaml plus calls to the parallel functional programming library Bulk Synchronous Parallel ML.

The tutorial is an introduction both to Coq[1] and the SYDPACC system for the systematic development of *correct and verified* parallel programs.

Duration: half-day

Outline

1. Introduction & an overview of Coq
2. Functional programming in Coq
3. Program verification in Coq
4. Implicit parallel programming in Coq

[1] ACM Software System Award in 2013

Goals and Objectives

The goals of the tutorial is to provide an introduction to the development of correct-by-contruction parallel programs, and to able the attendees to develop functional parallel programs using the SYDPACC system and the Coq proof assistant.

In order to achieve these goals, the tutorial is organised in three main parts :

- writing functional (sequential) programs in Coq,
- proving program properties in Coq,
- using the previous steps to explain the design and use of SYDPACC to actually build correct-by-construction parallel programs.

Each part consists of a presentation using slides and live demonstration, and a session of exercises.

To work on the exercises, the attendees will need to install the Coq proof assistant and the SYDPACC system on their laptops. In addition to the source code of the tools, to ease the installation, we will provide :

- a virtual machine image for VirtualBox,
- Docker containers,

with all necessary tools pre-installed.

We will also provide additional exercises and we will be available during the conference (and after) to help the attendees who may require help.

Audience Background

The audience is expected to be curious about functional programming and formal methods. There are no prerequisites other than basic logic and programming.

Presenter

Frédéric Loulergue obtained his PhD in Computer Science from the University of Orléans in 2000 and his Habilitation in Computer Science from Université Paris Val-de-Marne in 2004. He is currently a full professor at Université d'Orléans. His research interest is high-level parallel programming: semantics and implementation of parallel languages, certification of parallel programs and compilers as well as parallel (scientific) applications. Software associated to his research work include Bulk Synchronous Parallel ML (BSML) and the SyDPaCC system for the systematic development of programs for parallel and cloud computing.

He co-organised the series of international workshop on High-Level Parallel Programming and Applications (HLPP) between 2003 and 2010 and is now a member of its steering committee. He created and (co)-organised the series of international workshop on Practical Aspects of High-Level Parallel Programming (PAPP) from 2004 to 2012. He is a member of the editorial board of Scalable Computing: Practice and Experience, and Technique et Science Informatiques. He was deputy director of the Laboratory of Algorithms, Complexity and Logic (LACL), and deputy director of the Laboratoire d'Informatique Fondamentale d'Orléans (LIFO). He is currently the head of the Logic Modelling and Verification (LMV) research team at LIFO.

Web site: `http://frederic.loulergue.eu`