Verification of Grid and P2P Algorithms

Domains	distributed computing, logic
Institute	LORIA: Laboratoire lorrain de recherche en informatique et ses applications
Address	Campus Scientifique – BP 239, 54506 Vandœuvre-lès-Nancy
Teams	Algorille and Mosel
Supervisors	Stephan Merz (`Stephan.Merz@loria.fr`, 03.54.95.84.78)
	Martin Quinson (`Martin.Quinson@loria.fr`, 03.83.59.20.98)
Head of lab.	Karl Tombre (`Karl.Tombre@loria.fr`, 03.83.59.30.01)

Context

GRAS (Grid Reality and Simulation [1]) is an API for the implementation of distributed algorithms, in particular for the Grid or P2P networks, on heterogeneous platforms. It can be in two modes: for the simulation of programs (via SimGrid) as well as for native execution on real platforms.

The objective of this internship is to extend GRAS so that certain properties of distributed applications can be verified. Although formal verification techniques for concurrent and distributed algorithms are well known, they are usually applied to an abstract model of the algorithm, which is described in a specific modeling language that can be analysed with the help of tools known as *model checkers* that explore all possible executions of the program. This approach requires the users to learn the new formalism to specify their models. It also complicates the interpretation of the results: even if the model is correct, its implementation in a standard programming language could still be flawed.

Subject

Two basic approaches are possible for simplifying and automating the verification of programs using GRAS. The first one consists in modeling the skeleton of a program in a language such as TLA$^+$ [3]. The idea is to define a language in which distributed algorithms can be described in a way as similar as possible to the existing GRAS interface. Verification is made possible by implementing a compiler that translates this new language into TLA$^+$ and then making use of existing verification tools such as the TLC model checker. Some inspiration can be found in the language +CAL [2] that supports this idea for parallel programs based on shared variables.

The second approach consists in extending the GRAS execution platform by support for model checking. One could make use of the virtualisation that is currently used by GRAS so that a program can be executed in either simulation or native execution modes. The objective of this internship is to first explore the advantages and disadvantages of the two approaches for simple distributed algorithms. One of the two techniques will then be implemented and evaluated over concrete algorithms in the area of Grid and P2P programming such as Chord.

The construction of libraries and models of distributed programming, and the verification of such algorithms are important research problems. The internship could therefore naturally give rise to a future thesis.

Prerequisites

The subject is at the intersection of two different research fields. It is desirable that the student working on this subject has basic knowledge of either distributed algorithms or formal modeling and verification techniques. An experience in both domains would of course be an advantage, but is not required.

The technical prerequisites depend on the chosen approach. For the first approach, experience in compilation techniques and good Java programming skills will be necessary (the existing tools for TLA$^+$ are all written in Java). In order to extend the GRAS platform by a model checking mode, knowledge of multi-threaded programming will be appreciated, and good programming skills in C are required.

References

[1] M. Quinson. *GRAS: A Research and Development Framework for Grid and P2P Infrastructures.* Proc. 18th IASTED Intl. Conf. on Parallel and Distributed Computing and Systems (PDCS 2006). See also `http://simgrid.gforge.inria.fr/doc/gras.html`.

[2] L. Lamport. *Checking a Multithreaded Algorithm with +CAL.* 20th Intl. Symp. Distributed Computing (DISC 2006). Lecture Notes in Computer Science 4167, pp. 151-163. See also `http://research.microsoft.com/users/lamport/tla/pluscal.html`.

[3] L. Lamport. *Specifying Systems.* Addison-Wesley, Boston, Mass., 2002. See also `http://research.microsoft.com/users/lamport/tla/tla.html`.