Individual Project Proposal

Hans-Dieter A. Hiep

ABSTRACT

A project proposal for research and development of a foundational framework for verifying distributed algorithms in the proof assistant Coq.

1. INTRODUCTION

Based on the work of previous projects (1: DaViz, a simulation and visualization tool for distributed algorithms, and, 2: a visualization tool for Branching Bisimulation game semantics) the proposer wishes to formalize a foundational framework for distributed algorithms. The goal is to lay a solid foundation within the proof assistant Coq as a basis for developing and verifying distributed algorithms. The project aims for the following results: first, to formulate a hierarchical set of real-world assumptions (e.g. reliable/unreliable channels, process crashes, et cetera). Second, to define a model for expressing executions of distributed algorithms (cf. trace monoids). Third, to find and formalize related work in (modal) logic and specification languages, such as bisimulation and various process calculi, and define correctness properties. Last, to prove correctness of a selection of well-known distributed algorithms.

It is expected that the result of this project may be useful for the advancement of future projects of the proposer: the resulting framework may be used by the proposer to develop a blockchain-based protocol for a project commencing summer of 2017. The resulting framework may also be used as a first step in the proposer's master thesis.

A summary of the project is given in section 2, an overview of preliminary design choices given in section 3, and the proposal concludes with section 4, the projected deliverables.

2. OVERVIEW

Supervisor	Alban Ponse
Department	Computer Science
Section	Theory of Computer Science
Student	Hans-Dieter Hiep
Student no.	#10196951
ECTS	6
Hours	$6 \times 28 = 168$
Start date	June 6th, 2017
End date	June 30th, 2017

The proposed supervisor has worked on μCRL, "a toolset used for modelling, validation and verification of concurrent systems and protocols." Furthermore, the supervisor is the editor of the Handbook of Process Algebra, 2000. From the Preface: "In a process-algebraic approach to system verification, one typically writes two specifications. One, call it SYS, captures the design of the actual system and the other, call it SPEC, describes the system's desired 'high-level' behavior. One may then establish the correctness of SYS with respect to SPEC by showing that SYS behaves the 'same as' SPEC." Both are relevant to this project, where μCRL(2) serves as an inspiration and the handbook is used as an important source.

The proposed project is best summarized as a detailed study in the verification of distributed programming languages and process algebras and its semantics, applied to the setting of theorem proving and mechanical verification of proofs.

The total duration of the project is four weeks. The following schedule is only a sketch and may be deviated from:

1. Gathering literature (3 weeks)
2. Defining a hierarchy of assumptions (2 weeks)
3. Formulating a model for executions, corresponding to assumption classes (2,5 weeks)
4. Relating specification languages (2,5 weeks)
5. Formulating correctness properties of distributed algorithms (1,5 weeks)
6. Proving (correctness) properties of well-known distributed algorithms (2 week)

During the execution of the project, the student will (at least) once per week report status of the project either by e-mail or in person. The student is responsible for arranging meetings, if necessary. At the end of the project, a short (informal) talk is given to present the results of the research, where (at least) the supervisor is present.

3. DESIGN PARAMETERS

The following preliminary assumption classes are known from the previous project:

- Network topology
  - Undirected network, i.e. bidirectional channels
  - Directed networks, i.e. unidirectional channels
  - Static networks, i.e. fixed topology during whole execution
  - Dynamic networks, i.e. changing topology during execution
  - Self-modifying networks, i.e. topology is configurable by algorithm during execution
  - Connected networks, i.e. every process is reachable from every other process

  - Disconnected networks, i.e. networks may be split in disparate parts

- Channels
  - FIFO, i.e. channels are buffered and message order is preserved
  - Non-FIFO, i.e. unbuffered channels where messages may be arbitrarily reordered
  - Unreliable channels, i.e. channels that may drop messages
- Processes
  - Crashes, i.e. processes may halt unexpectedly
  - Failure detectors, i.e. eventually providing information to neighbors of a crashed process
  - Byzantine processes, i.e. a process may become Byzantine and send arbitrary messages and not follow any predefined algorithm
- Anonymity
  - Known networks, i.e. processes are uniquely identifiable and have global knowledge of topology
  - Locally-known networks, i.e. processes are uniquely identifiable up to neighbors
  - Anonymous networks, i.e. networks without process identification

Additionally, the proposer has already worked on (preliminary) definitions resulting from previous projects which will be combined and extended into a fully-fledged framework as is proposed. The specifics of these definitions are not relevant for this proposal, but will be used as a starting ground from which the exploration will commence.

4. DELIVERABLES

The project concludes with showing the following work products. The most important one is the development of a foundational framework in Coq. Secondary is a technical report that describes the results of the project.

1. A set of Coq files containing the proof development of a foundational framework, including proof-of-concept distributed algorithms, a set of correctness properties, and (some) proofs of these properties of the given algorithms.
2. A report, similar in structure as this proposal, between 4 and 10 pages. The report includes a description of (the most important of) the formalized definitions, and a rationalization for choosing these definitions. The report includes a bibliography of all useful literature. The report shows examples of proven properties in a human-readable fashion of aforementioned algorithms.

The (delivered) work is performed and authored by the proposer only. However, useful suggestions or hints may be provided by the supervisor or by other students.