ISSN 2186-7437

NII Shonan Meeting Report

No. 2014-9

Summer School on Coq

Pierre Castéran
Jacques Garrigue
David Nowak

August 25–29, 2014

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Summer School on Coq

Organizers:
Pierre Castéran (LaBRI, France)
Jacques Garrigue (Nagoya University, Japan)
David Nowak (CNRS & Lille 1 University, France)

August 25–29, 2014

The objective of this school was to teach the use of the Coq proof assistant that has received the SIGPLAN Programming Languages Software 2013 Award and the 2013 ACM Software System Award. It consisted of lectures and practices in English by internationally renowned experts in Coq.

Prerequisites. No previous knowledge of Coq was necessary to follow the lectures. Students were however requested to attend the school with their own computer with the latest stable version of Coq and one of its user interfaces (CoqIDE distributed with Coq, or Proof General) installed. They are freely available online:

- Coq and CoqIDE: `http://coq.inria.fr`
- Proof General: `http://proofgeneral.inf.ed.ac.uk`

What is the Coq proof assistant? In principle, all mathematics can be formalized in axiomatic set theory, and then checked automatically by a computer. In the last century, more practical foundations for mathematics, based on type theory, have been designed and implemented as proof assistants. One of the most prominent is Coq, developed in France since 1984. They can check proofs and organize them in searchable libraries. They also provide convenient interfaces that help the user make proofs incrementally, and fill in automatically the trivial parts.

What can be done with Coq? In the beginning, proof assistants could only be used to formalize toy examples. It is however not anymore the case. Proof assistants have reached maturity and can deal with difficult mathematical results. For example, in December 2004, Gonthier has announced the full formalization of the four colors theorem in the Coq proof assistant (cf. Notices of the AMS 55(11), 2008). More recently, the proof of the Feit-Thompson theorem was completed in Coq in 2012.

Mathematicians are not the only ones to make mathematical proofs. Computer scientists, for instance, make their own proofs. In a software, a small error can indeed result in serious damages in terms of safety and security. It is thus important to provide formal proofs that a software is correct: extensive testing of software or hardware is not enough because it may still miss errors that

would be avoided with the use of a proof assistant. One prominent example is the formalization of a complete Java Card system in the Coq proof assistant and the certification that it meets the highest security requirements for industrial product, i.e., the Common Criteria EAL7 level. This was achieved in 2003 by Trusted Logic, a company that provides secure software for smart cards, terminals and consumer devices. Another example is the CompCert compiler: it is equipped with a proof in Coq that it will generate assembly code that behaves as prescribed by the semantics of the source code in C, which ensures that no bugs are introduced during compilation. From the mathematicians eye, some of those proofs may not appear difficult but they are nonetheless tricky and error-prone because they involve huge formal objects such as programs or automata, or require to consider hundreds of cases.

1 Sponsors

This summer school was sponsored by:

- Japanese-French Laboratory for Informatics, CNRS
- Inria

2 List of participants

Lecturers

- Dr. Yves Bertot, Inria, France
- Prof. Sandrine Blazy, University of Rennes 1, France
- Prof. Pierre Castéran, LaBRI, France
- Dr. Assia Mahboubi, Inria, France

Students

1. Mr. Mitsuru Arakawa, Nagoya University
2. Mr. Andrei Arusoaie, Inria
3. Mr. Soichiro Fujii, University of Tokyo
4. Dr. Ken-etsu Fujita, Gunma University
5. Mr. Makoto Fujiwara, Tohoku University
6. Dr. Makoto Hamana, Gunma University
7. Dr. Koji Hasebe, University of Tsukuba
8. Mr. Tomoaki Hashizaki, JAIST
9. Mr. Yoshihiro Imai, IT Planning
10. Mr. Kim Joonhee, Yonsei University

11. Ms. Sunyoung Kim, Yonsei University
12. Mr. Daisuke Kinoshita, University of Electro-communications
13. Mr. Yuki Manabe, Tohoku University
14. Dr. Yasuhiko Minamide, University of Tsukuba
15. Mr. Takashi Miyamoto, Demand Side Science, Inc.
16. Prof. Yoshihiro Mizoguchi, Kyushu University
17. Mr. Shun Mizukami, Nagoya University
18. Mr. Hiromitsu Morita, NT Engineering Corporation
19. Mr. Yu Nishiki, Nagoya University
20. Mr. Hiroshi Ogawa, University of Tokyo
21. Mr. Kazuhiko Sakaguchi, University of Tsukuba
22. Mr. Noriaki Sakamoto, Tokyo Institute of Technology
23. Mr. Masahiro Sato, Nagoya University
24. Mr. François Serman, Lille 1 University
25. Mr. Keishi Suda, NTT Data Mathematical Systems, Inc.
26. Mr. Hisaharu Tanaka, Saga University
27. Dr. Tadanori Teruya, AIST
28. Prof. Hideki Tsuiki, Kyoto University
29. Mr. Chiharu Usui, Tsukuba University
30. Mr. Kimitaka Watanabe, University of Tsukuba
31. Mr. Shohei Yasutake, Tokyo Institute of Technology

3 Program

The following program contains seven teaching sessions, each one being composed of a 1h30 lecture, then a 1h30 session of exercises on laptops on the same topic.

As a guiding thread for the lectures and exercises, we showed how to define in Coq various semantics of a toy programming language and prove their equivalence.

While this program is designed to ensure that people with little or no experience using Coq can get on, we adapted it to the audience so that even experimented people could deepen their understanding of the system.

In addition to the exercises associated with each lecture, three projects were proposed to people who already knew the basics of Coq, and wished to work on advanced topics in collaboration with the lecturers:

I. Sorts and Permutations

II. Semantics and Programming Languages

III. Transition Systems, Simulation and Bisimulation

Two more projects, one about compiling imperative languages and the other one about advanced tactic design, arose from the interaction between advanced students and the lecturers.

August 24th (Sunday)

- 15:00 – 19:00 Hotel check-in (early check-in from 12:00 is negotiable)
- 19:00 – 21:00 Welcome Banquet

August 25th (Monday)

- 09:00 Morning Lecture 1: *Coq as a functional programming language*, by Yves Bertot

  Introduction to Coq, Type checking and computations, Functional programming, Pattern matching on booleans and natural numbers.

- 13:30 Afternoon Lecture 2: *Structural recursive programming*, by Assia Mahboubi

  Recursive definitions on natural numbers, The list data type, Recursive programming on lists, Polymorphic types and functions in Coq.

August 26th (Tuesday)

- 08:30 Morning Lecture 3: *Propositions and Predicates*, by Assia Mahboubi

  Representation of logical information through Coqs type system, Propositional, first-order and higher order logic, Application to program specifications and mathematical statements.

- 13:30 Afternoon Lecture 4: *Interactive Proofs*, by Sandrine Blazy

  Basic components of interactive theorem proving : statements, goals and tactics, Introduction and elimination tactics, Proof by induction, Rewriting techniques.

August 27th (Wednesday)

- 08:30 Morning Lecture 5: *Inductive Predicates I*, by Pierre Castéran

  Inductive predicate definitions, Proofs by induction on such a predicate, Examples taken from Coq's standard library.

- 13:00 excursion, directly followed by the banquet in the evening

August 28th (Thursday)

- 08:30 Morning Lecture 6: *Representing Programs in Coq*, by Yves Bertot

  Representation of the abstract syntax and semantics of a toy imperative, programming language, This example uses all the concepts seen in previous lectures.

- 13:30 Afternoon Lecture 7: *Inductive Predicates II*, by Pierre Castéran

  Continuation of Lecture 5, Inversion techniques, Comparison between induction on data and induction on predicates.

August 29th (Friday): Advanced applications and research

- 08:30 – 09:30 *A Computer-Algebra Based Formal Proof of the Irrationality of* $\zeta(3)$, by Assia Mahboubi
- 10:00 – 11:00 *Building a certified optimizing C compiler*, by Sandrine Blazy
- 11:00 – 12:00 Short presentations by three students who previously developed in Coq:
  - *Formalization of Category Theory*, by Keishi Suda
  - *Proof of Normalization of Girard's System* $\boldsymbol{F}$, by Kazuhiko Sakaguchi
  - *Formalization of properties of industrial code for inter-process communication.*, by Yoshihiro Imai
  - *Contribution: RegExp*, by Takashi Miyamoto

4 Feedback

The following points were observed by the organizers and lecturers:

- Availability of extra rooms for advanced exercices and technical discussions allowed us to adapt the School to the varied level of the students.
- The presence of advanced students was an appreciated help to the beginners during the lab sessions. Conversely, the staff (organizers + lecturers) was composed of 6 experts who could answer to students of any level. Thus the variety of levels of the students was not at all an issue.
- A longer and previously announced session of short presentations by the students should be included in the schedule.