Activity Report 2015

Project-Team MARELLE

Mathematical, Reasoning and Software

RESEARCH CENTER
Sophia Antipolis - Méditerranée

THEME
Proofs and Verification

Table of contents

Project-Team MARELLE

Creation of the Project-Team: 2006 November 01

Keywords:

Computer Science and Digital Science:

2.1.11. - Proof languages
2.4.3. - Proofs
4.5. - Formal methods for security
7.4. - Logic in Computer Science

Other Research Topics and Application Domains:

6.1. - Software industry
9.4.1. - Computer science
9.4.2. - Mathematics

1. Members

Research Scientists

Yves Bertot [Team leader, Inria, Senior Researcher, HdR]
Cyril Cohen [Inria, Researcher]
Benjamin Grégoire [Inria, Researcher]
José Grimm [Inria, Researcher]
Laurence Rideau [Inria, Researcher]
Enrico Tassi [Inria, Researcher]
Laurent Théry [Inria, Researcher]

Engineer

Maxime Dénès [Inria, granted by Caisse des Dépôts et Consignations]

PhD Student

Boris Djalal [Inria, from Mar 2015]

Visiting Scientists

Isabela Dramnesc [University of Timisoara, from Jun 2015 until Jul 2015]
Tsvetan Dunchev [University of Bologna, Jul 2015]

Administrative Assistant

Nathalie Bellesso [Inria]

Others

Cécile Baritel-Ruet [ENS Cachan, Student, from Mar 2015 until May 2015]
Jean-Yves Franceschi [ENS Lyon, Student, from Jun 2015 until Jul 2015]
Loic Pottier [Min. de l'Education Nationale, High School teacher, HdR]

2. Overall Objectives

2.1. Overall Objectives

We want to concentrate on the development of mathematical libraries for theorem proving tools. This objective contributes to two main areas of application: tools for mathematicians and correctness verification tools for software dealing with numerical computation.

In the short term, we aim for mathematical libraries that concern polynomials, algebra, group theory, floating point numbers, real numbers, big integers, probabilities and geometrical objects. In the long run, we think that this will involve any function that may be of use in embedded software for automatics or robotics (in what is called hybrid systems, systems that contain both software and physical components) and in cryptographical systems. We want to integrate these libraries in theorem proving tools because we believe they will become important tools for mathematical practice and for engineers who need to prove the correctness of their algorithms and software.

We believe that theorem proving tools are good tools to produce highly dependable software, because they provide a framework where algorithms and specifications can be studied uniformly and often provide means to mechanically derive programs that are correct by construction.

We also study the extensibility of interactive theorem proving tools based on decision procedures that free designers from the burden of verifying some of the required properties. We often rely on "satisfiability modulo theory" procedures, which can be connected to theorem proving tools in a way that preserves the trustability of the final results.

3. Research Program

3.1. Type theory and formalization of mathematics

The calculus of inductive constructions is a branch of type theory that serves as a foundation for theorem proving tools, especially the Coq proof assistant. It is powerful enough to formalize complex mathematics, based on algebraic structures and operations. This is especially important as we want to produce proofs of logical properties for these algebraic structures, a goal that is only marginally addressed in most scientific computation systems.

The calculus of inductive constructions also makes it possible to write algorithms as recursive functional programs which manipulate tree-like data structures. A third important characteristic of this calculus is that it is also a language for manipulating proofs. All this makes this calculus a tool of choice for our investigations. However, this language is still being improved and part of our work concerns these improvements.

3.2. Verification of scientific algorithms

To produce certified algorithms, we use the following approach: instead of attempting to prove properties of an existing program written in a conventional programming language such as C or Java, we produce new programs in the calculus of constructions whose correctness is an immediate consequence of their construction. This has several advantages. First, we work at a high level of abstraction, independently of the target implementation language. Secondly, we concentrate on specific characteristics of the algorithm, and abstract away from the rest (for instance, we abstract away from memory management or data implementation strategies). Therefore, we are able to address more high-level mathematics and to express more general properties without being overwhelmed by implementation details.

However, this approach also presents a few drawbacks. For instance, the calculus of constructions usually imposes that recursive programs should explicitly terminate for all inputs. For some algorithms, we need to use advanced concepts (for instance, well-founded relations) to make the property of termination explicit, and proofs of correctness become especially difficult in this setting.

3.3. Programming language semantics

To bridge the gap between our high-level descriptions of algorithms and conventional programming languages, we investigate the algorithms that are present in programming language implementations, for instance algorithms that are used in a compiler or a static analysis tool. For these algorithms, we generally base our work on the semantic description of a language. The properties that we attempt to prove for an algorithm are, for example, that an optimization respects the meaning of programs or that the programs produced are free of some unwanted behavior. In practice, we rely on this study of programming language semantics to propose extensions to theorem proving tools or to participate in the verification that compilers for conventional programming languages are exempt from bugs.

4. Application Domains

4.1. Reliability of embedded software

Software embedded in physical devices performs computations where the inputs are provided by measures and the outputs are transformed into actions performed by actuators. to improve the quality of these devices, we expect that all the computations performed in this kind of software will need to be made more and more reliable. We claim that formal methods can serve this purpose and we develop the libraries and techniques to support this claim. This implies that we take a serious look at how mathematics can be included in formal methods, especially concerning geometry and calculus.

4.2. Security and Cryptography

The modern economy relies on the possibility for every actor to trust the communications they perform with their colleagues, customers, or providers. We claim that this trust can only be built by a careful scrutiny of the claims made by all public protocols and software that are reproduced in all portable devices, computers, and internet infrastructure systems. We advocate the use of formal methods in these domains and we provide easy-to-use tools for cryptographers so that the formal verification of cryptographic algorithms can become routine and amenable to public scrutiny.

4.3. Mathematics and Education

As librairies for theorem provers evolve, they tend to cover an ever increasing proportion of the mathematical background expected from engineers and scientists of all domains. Because the content of a formally verified library is extremely precise and explicit, we claim that this will provide a new kind of material for teaching mathematics, especially useful in remote education.

5. New Software and Platforms

5.1. Coq

KEYWORDS: Proof - Certification - Formalisation
FUNCTIONAL DESCRIPTION

Coq provides both a dependently-typed functional programming language and a logical formalism, which, altogether, support the formalisation of mathematical theories and the specification and certification of properties of programs. Coq also provides a large and extensible set of automatic or semi-automatic proof methods. Coq's programs are extractible to OCaml, Haskell, Scheme, ...

- Participants: Benjamin Grégoire, Enrico Tassi, Bruno Barras, Yves Bertot, Pierre Boutillier, Xavier Clerc, Pierre Courtieu, Maxime Denes, Stéphane Glondu, Vincent Gross, Hugo Herbelin, Pierre Letouzey, Assia Mahboubi, Julien Narboux, Jean-Marc Notin, Christine Paulin-Mohring, Pierre-Marie Pédrot, Loïc Pottier, Matthias Puech, Yann Régis-Gianas, François Ripault, Matthieu Sozeau, Arnaud Spiwack, Pierre-Yves Strub, Benjamin Werner, Guillaume Melquiond and Jean-Christophe Filliâtre
- Partners: CNRS - Université Paris-Sud - ENS Lyon - Université Paris-Diderot
- Contact: Hugo Herbelin
- URL: http://coq.inria.fr/

Enrico Tassi and Maxime Dénès brought notable contributions to the Coq system in 2015. In particular, Enrico worked on the new user-interface that makes it possible to have several logical engines working on proofs simultaneously and Maxime Dénès supervised the release process for Coq 8.5, to be released in the early days of January.

In 2015, the Coq system is the object of intense activity within the Marelle project-team. Yves Bertot and Maxime Dénès are working at creating a consortium around this system, so that academic and industrial users find a suitable structure to voice there wishes for the evolution of the system, fund improvements, and coordinate developments for further improvement. This work is done in close collaboration with the $\pi.r^2$ project-team.

A first outcome of this animation work is the organization of regular events for developers to meet (coding sprints), the first of which happened in Sophia Antipolis in June 2015. Subsequently, Maxime Dénès was hired in Sophia Antipolis (in the Marelle project-team), and Matej Kosik was hired in Paris (in the $\pi.r^2$) team. A close collaboration was also set up with the Massachusetts Institute of Technology (MIT), with a software engineer to be hired at MIT to work on Coq in early 2016.

5.2. Easycrypt

FUNCTIONAL DESCRIPTION

EasyCrypt is a toolset for reasoning about relational properties of probabilistic computations with adversarial code. Its main application is the construction and verification of game-based cryptographic proofs. EasyCrypt can also be used for reasoning about differential privacy.

- Participants: Gilles Barthe, Benjamin Grégoire and Pierre-Yves Strub
- Contact: Gilles Barthe
- URL: https://www.easycrypt.info/trac/

5.3. Math-Components

Mathematical Components library
FUNCTIONAL DESCRIPTION

The Mathematical Components library is a set of Coq libraries that cover the mechanization of the proof of the Odd Order Theorem.

- Participants: Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Georges Gonthier, Stéphane Le Roux, Assia Mahboubi, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi and Russell O'connor
- Contact: Assia Mahboubi
- URL: http://www.msr-inria.fr/projects/mathematical-components-2/

5.4. Ssreflect

FUNCTIONAL DESCRIPTION

Ssreflect is a tactic language extension to the Coq system, developed by the Mathematical Components team.

- Participants: Cyril Cohen, Yves Bertot, Laurence Rideau, Enrico Tassi and Laurent Théry
- Contact: Yves Bertot
- URL: http://ssr.msr-inria.inria.fr/

5.5. Zoocrypt

FUNCTIONAL DESCRIPTION

ZooCrypt is an automated tool for analyzing the security of padding-based public-key encryption schemes (i.e. schemes built from trapdoor permutations and hash functions). This years we extended the tool to be able to deal with schemes based on cyclic groups and bilinear maps.

- Participants: Benjamin Grégoire, Gilles Barthe and Pierre-Yves Strub
- Contact: Gilles Barthe
- URL: https://www.easycrypt.info/zoocrypt/

6. New Results

6.1. IDE for Coq

Participants: Enrico Tassi, Alexander Faithfull [ITU Copenhagen], Jesper Bengtson [ITU Copenhagen], Carst Tankink.

User interfaces for interactive proof assistants should rely on the advanced software available in integrated development environments. we collaborated with researchers from Copenhagen to build an Eclipse-based environment for the Coq system. This exploits the quick compilation chain that was developed for Coq 8.5. This work has been published in [15].

6.2. ELPI, Fast, Embeddable, λ-Prolog Interpreter

Participants: Enrico Tassi, Cvetan Dunchev [University of Bologna], Ferruccio Guidi [University of Bologna], Claudio Sacerdoti Coen [University of Bologna].

We developed a new interpreter that runs consistently faster than the other available implementations of λ-prolog. The key insight is the identification of a fragment of the language, which we call reduction-free fragment, that occurs quite naturally and that admits constant time reduction and unification rules. In the long run, we hope that this will contribute to developing elaborators that support a more efficient and adaptable usage of interactive proof tools. This work is published in [14].

6.3. Verified Proofs of Higher-Order Masking

Participants: Gilles Barthe [IMDEA Software, Madrid], Sonia Belaïd [Thales Communication], François Dupressoir [IMDEA Software, Madrid], Pierre-Alain Fouque [Université de Rennes, IUF], Benjamin Grégoire, Pierre-Yves Strub [IMDEA Software, Madrid].

We study the problem of automatically verifying higher-order masking countermeasures. We propose a method based on program verification techniques, to check the independence of sets of intermediate variables from secrets. This new language-based technique makes it possible to implement several algorithms that reduce the number of sets of variables that need consideration. The tool also has the capability to to give useful information when proofs fail, for instance by discovering possible attacks. This is based on EasyCrypt. This work has been published in [8].

6.4. Relational Reasoning via Probabilistic Coupling

Participants: Gilles Barthe [IMDEA Software, Madrid], Thomas Espitau [ENS Cachan], Benjamin Grégoire, Justin Hsu [University of Pennsylvania], Léo Stefanesco [ENS Lyon], Pierre-Yves Strub [IMDEA Software, Madrid].

Probabilistic coupling is a powerful tool for analyzing pairs of probabilistic processes. While the mathematical definition of coupling looks rather complex and cumbersome to manipulate, we show that the relational program logic pRHL—the logic underlying the EasyCrypt cryptographic proof assistant—already internalizes a generalization of probabilistic coupling. With this insight, constructing couplings is no harder than constructing logical proofs. We demonstrate how to express and verify classic examples of couplings in pRHL, and we mechanically verify several couplings in EasyCrypt. This work is described in [9].

6.5. Automated Proofs of Pairing-Based Cryptography

Participants: Gilles Barthe [IMDEA Software, Madrid], Benjamin Grégoire, /benedikt Schmidt [IMDEA Software, Madrid].

We implement a new tool, called AutoG&P, which supports extremely compact, and often fully automated, proofs of cryptographic constructions based on (bilinear or multilinear) Diffie-Hellman assumptions. For instance, we provide a 100-line proof of Waters' Dual System Encryption (CRYPTO'09), and fully automatic proofs of Boneh-Boyen Identity-Based Encryption (CRYPTO'04). Finally, we provide an automated tool that generates independently verifiable EasyCrypt proofs from AutoG&P proofs. This work has been published in [10].

6.6. Improvements on CBC MAC formalized in EasyCrypt

Participants: Benjamin Grégoire, Cécile Baritel-Ruet, Pierre-Alain Fouque.

In a paper of 2003, J. Black and P. Rogaway propose variations of cipher block chaining message authentication codes for the efficient authentication of arbitrary length messages. We formalize their work in EasyCrypt, resulting in formal proofs for CBC-MAC, EMAC, ECBC, FCBC and the most efficient of these variations, XCBC.

This work required the development of new EasyCrypt theories. A small flaw in the original paper was found and a fix has been proposed. This work was partially funded by the Brutus ANR project.

6.7. Buchberger's algorithm and advanced formalization of multinomials

Participant: Laurent Théry.

We studied how the Mathematical Components library could improve the formalization of of algorithms based on multivariate polynomials. In particular, we re-used Pierre-Yves Strub library of multivariate polynomials and re-did the proofs of correctnes for Buchberger's algorithm. This new piece of formalized algorithm is now available at the following address https://github.com/thery/grobner.

6.8. Proofs that e and π and transcendental

Participants: Sophie Bernard, Laurence Rideau, Yves Bertot, Pierre-Yves Strub [IMDEA Software, Madrid].

In the previous year, we developed formally verified proofs that e and π are transcendental. This year we cleaned up these proofs to obtain a common lemma that applies in both cases with simple hypotheses. In parallel, P.-Y. Strub streamlined the library on multivariate polynomials which plays a significant role in the case of π. This work has been published in [11].

In the future, we will probably extend this work to more general proofs of transcendance.

6.9. Algorithms for Real Algebraic Geometry

Participant: Cyril Cohen.

We formalized an efficient algorithm to count roots of a polynomial satisfying polynomial inequalities. This work was presented at the Types workshop in May and the Workshop on Algebra, Geometry, and Proofs in Symbolic computation.

6.10. Nominal sets in Coq

Participants: Cyril Cohen, Nicolas Tabareau, Matthieu Sozeau, Gabriel Lewertoski.

Cyril Cohen collaborated with members of the team $\pi.r^2$ on the implementation of nominal sets in Coq.

6.11. Formal Description of Dynamic Logic

Participants: Yves Bertot, Cyril Cohen, Jean-Yves Franceschi.

We developed a formal description of the language of dynamic logic in the Coq system.

6.12. Cubical Type Theory

Participants: Cyril Cohen, Thierry Coquand, Simon Huber, Anders Mörtberg.

We participate to the development of a software prototype, `cubicaltt`, https://github.com/mortberg/cubicaltt, that is expected to support an extension of type theory suited for homotopy type theory.

6.13. Finite set and finite maps

Participant: Cyril Cohen.

We extend the Math-Components library with a module concerning finite sets (in potentially infinite types) and finite maps. This module will play a crucial role in other experiments, like the experiments on dynamic logic, nominal sets, and cubical sets.

6.14. Formalization of a Newton Series Representation of Polynomials

Participants: Boris Djalal, Cyril Cohen.

We formalize an algorithm to change the representation of a polynomial to a Newton power series. This provides a way to compute efficiently polynomials whose roots are the sums or products of roots of other polynomials, and hence provides a base component of efficient computation for algebraic numbers. In order to achieve this, we formalize a notion of truncated power series and develop an abstract theory of poles of fractions. This work is described in [13].

6.15. Formal description of catalan numbers

Participant: José Grimm.

Catalan number can be defined by a recurrence, or by explicit formulas using binomial numbers. An important property is $C_{n+1} = \sum_{k \le n} C_k C_{n-k}$. The easiest way to prove this formula is to use Dyck paths.

A Dyck path of size $2n$ is a sequence l of integers $+1$ and -1 so that the sum s_k of the k first terms is ≥ 0 for $k \leq 2n$ and $s_{2n} = 0$. The relation between Dyck paths and Catalan numbers is easy to prove and then properties of Dyck paths are quite simple to state and verify.

The proofs have been done with the Math-Components library.

6.16. Latex to XML translator

Participant: José Grimm.

This year, we released version 2.15.4 of Tralics, our LaTeX to XML translator. Array handling has been redesigned: for instance, an array preamble of the form `{>{$}c<{$}}` is now correctly interpreted; there is a possibility to add an attribute pair to any table, row or cell; for math environments like “align”, one label and one tag per row is allowed. Tralics is also able to read an XML file, and there are some primitives for inserting the result (or part of it) into the XML code under construction.

7. Bilateral Contracts and Grants with Industry

7.1. Bilateral Contracts with Industry

In 2015, we discussed a contract with a potential industrial partner, but these discussions are currently covered by a non-disclosure agreement. We expect this discussion to become visible in 2016.

8. Partnerships and Cooperations

8.1. National Initiatives

8.1.1. ANR

We are currently members of two projects funded by the French national agency for research funding.

- BRUTUS "Chiffrements authentifiés et résistants aux attaques par canaux auxiliaires", started on October 1st, 2014, for 60 months, with a grant of 41 kEuros for Marelle. Other partners are Université de Rennes 1, CNRS, secrétariat Général de la défense et de la sécurité nationale, and Université des Sciences et Technologies de Lille 1. The corresponding researcher for this contract is Benjamin Grégoire.
- FastRelax, "Fast and Reliable Approximations", started on October 1st, 2014, for 60 months, with a grant of 75 kEuros for Marelle. Other partners are Inria Grenoble (ARIC project-team), LAAS-CNRS (Toulouse), Inria Saclay (Toccata and Specfun project-teams), and LIP6-CNRS (Paris). The corresponding researcher for this contract is Laurence Rideau.

8.2. European Initiatives

8.2.1. Collaborations in European Programs, except FP7 & H2020

Program: COST

Project acronym: CA15123EUTYPES

Project title: The European research network on types for programming and verification

Duration: 30 October 2015– 29 October 2019

Coordinator: Herman Geuvers (Radboud University, Nijmegen)

Other partners: List too long to repeat here.

Abstract: Types are pervasive in programming and information technology. A type defines a formal interface between software components, allowing the automatic verification of their connections, and greatly enhancing the robustness and reliability of computations and communications. In rich dependent type theories, the full functional specification of a program can be expressed as a type. Type systems have rapidly evolved over the past years, becoming more sophisticated, capturing new aspects of the behaviour of programs and the dynamics of their execution.

This COST Action will give a strong impetus to research on type theory and its many applications in computer science, by promoting (1) the synergy between theoretical computer scientists, logicians and mathematicians to develop new foundations for type theory, for example as based on the recent development of "homotopy type theory", (2) the joint development of type theoretic tools as proof assistants and integrated programming environments, (3) the study of dependent types for programming and its deployment in software development, (4) the study of dependent types for verification and its deployment in software analysis and verification. The action will also tie together these different areas and promote cross-fertilisation.

Europe has a strong type theory community, ranging from foundational research to applications in programming languages, verification and theorem proving, which is in urgent need of better networking. A COST Action that crosses the borders will support the collaboration between groups and complementary expertise, and mobilise a critical mass of existing type theory research.

8.3. International Initiatives

8.3.1. Inria International Partners

8.3.1.1. Informal International Partners

We have important collaborations with the team of Thierry Coquand at Chalmers and University of Göteborg.

We are setting up a collaboration with the team of Adam Chlipala at the Massachusetts Institute of Technology.

8.4. International Research Visitors

8.4.1. Visits of International Scientists

Isabela Dramnesc, from the University of Timişoara in Romania, visited our team in June and July to study proving techniques in the Coq context.

Tsvetan Dunchev, from the University of Bologna, visited our team in July to work on ELPI, the λ-prolog interpreter.

8.4.2. Visits to International Teams

Yves Bertot organised a meeting with representants of University of Pennsylvania, Princeton University, Yale University, Harvard University, and the Massachusetts Institute of Technology in Boston in April. Janet Bertot, Philippe Nain, and Matthieu Sozeau from Inria also attended this meeting. The agenda of the meeting was preliminary discussions for the creation of a consortium around the Coq software system.

Enrico Tassi visited the team of Jesper Bengtson at the IT University in Copenhagen for a week at the end of September.

Cyril Cohen visited Chalmers university in Febrary and October to work on cubical type theory.

Cyril Cohen was invited by AIST in Japan for a one-week stay in Tsukuba in November to work on formalization problems for robotics.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific events organisation

9.1.1.1. Member of the organizing committees

Yves Bertot is member of the organizing committee for the UITP (User-Interfaces for Theorem Provers) series of workshops.

Enrico Tassi organized a Coq coding Sprint in Sophia Antipolis, June 22-25, in Sophia Antipolis. There were 39 attendants.

Yves Bertot organized a Coq workshop in Sophia Antipolis on June 26. There were around 40 attendants.

9.1.2. Scientific events selection

9.1.2.1. Member of the conference program committees

Yves Bertot was a member of the conference program committee for ITP'15 (Interactive Theorem Proving).

Cyril Cohen was a member of the conference program committee for JFLA'16 (Journées Francophones des Langages Applicatifs).

Laurent Théry was a member of the program committee for PxTP'15 (Proof exchange for Theorem Provers).

9.1.2.2. Reviewer

Members of the project reviewed papers for the conferences CPP'16 (Certified Programs and Proofs), ESOP'15 (European Symposium on Programming), Types, LICS'15 (Logic in Computer Science), Cade'15 (Conference on Automated Deduction), ITP'15 (Interactive Theorem Proving), JFLA'16 (Journées Francophones des langages applicatifs).

9.1.3. Journal

9.1.3.1. Reviewer - Reviewing activities

Members of the project-team reviewed papers for the journals *Journal of Functional Programming*, *Journal of Formal Reasoning*, *Journal of Automated Reasoning*.

9.1.4. Invited talks

Laurence Rideau gave an invited talk at the conference TFP'15 (Trends in Functional Programming).

Cyril Cohen and Yves Bertot gave invited talks at the workshop on Algebra, Geometry, and Proofs in Symbolic Computation in Toronto in December.

Cyril Cohen and Enrico Tassi gave lectures at the tutorial on Coq organized as satellite event to ITP'15 in Nanjing, China, in August.

9.1.5. Research administration

- Members of the project-team evaluated project for the French national agency for research funding (ANR).
- José Grimm is a member of the local committee for Hygiene and Work Safety.
- Cyril Cohen regularly serves as secretary for the local committee of project-team leaders.
- Yves Bertot is the chairman of the *Coq Steering committee*, Enrico Tassi is a member of this committee.

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

Licence : Laurence Rideau, “programming and algorithms", 50 hours, Lycée Masséna, Nice, France.

Licence : Cyril Cohen, “paradigms of programming languages”, 12 hours, University of Nice, France.

Master : Yves Bertot and Enrico Tassi, "software verification and computer proof", 21 hours, Université de Nice, France.

Master : Laurent Théry, "introduction to Coq", 3 hours, Ecole des Mines de Paris, France.

9.2.2. Supervision

PhD in progress : Boris Djalal, starting in October 2015, supervised by Yves Bertot and Cyril Cohen

9.2.3. Juries

Yves Bertot was examiner with written report duty for the thesis of Catherine Lelay (University of Paris-south) and examiner for the thesis of Robbert Krebbers (Radboud University Nijmegen).

10. Bibliography

Major publications by the team in recent years

[1] G. Barthe, B. Grégoire, S. Heraud, S. Z. Béguelin. *Computer-Aided Security Proofs for the Working Cryptographer*, in "Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings", Lecture Notes in Computer Science, Springer, 2011, vol. 6841, pp. 71-90, Best Paper Award

[2] Y. Bertot, P. Castéran. *Interactive Theorem Proving and Program Development, Coq'Art:the Calculus of Inductive Constructions*, Springer-Verlag, 2004

[3] Y. BERTOT, G. GONTHIER, S. O. BIHA, I. PAŞCA. *Canonical Big Operators*, in "Proceedings of the 21st International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2008)", Lecture Notes in Computer Science, Springer, August 2008, vol. 5170, pp. 12–16, http://hal.inria.fr/inria-00331193/

[4] G. GONTHIER, A. ASPERTI, J. AVIGAD, Y. BERTOT, C. COHEN, F. GARILLOT, S. LE ROUX, A. MAHBOUBI, R. O'CONNOR, S. OULD BIHA, I. PASCA, L. RIDEAU, A. SOLOVYEV, E. TASSI, L. THÉRY. *A Machine-Checked Proof of the Odd Order Theorem*, in "ITP 2013, 4th Conference on Interactive Theorem Proving", Rennes, France, S. BLAZY, C. PAULIN, D. PICHARDIE (editors), LNCS, Springer, 2013, vol. 7998, pp. 163-179 [*DOI :* 10.1007/978-3-642-39634-2_14], http://hal.inria.fr/hal-00816699

[5] G. GONTHIER, A. MAHBOUBI, L. RIDEAU, E. TASSI, L. THÉRY. *A Modular Formalisation of Finite Group Theory*, in "Proceedings of the 20th International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2007)", K. SCHNEIDER, J. BRANDT (editors), LNCS, Springer-Verlag, September 2007, vol. 4732, pp. 86-101, http://hal.inria.fr/inria-00139131

Publications of the year

Articles in International Peer-Reviewed Journals

[6] É. MARTIN-DOREL, G. HANROT, M. MAYERO, L. THÉRY. *Formally verified certificate checkers for hardest-to-round computation*, in "Journal of Automated Reasoning", 2015, vol. 54, n^{o} 1, pp. 1-29 [*DOI :* 10.1007/s10817-014-9312-2], https://hal.inria.fr/hal-00919498

International Conferences with Proceedings

[7] B. BARRAS, C. TANKINK, E. TASSI. *Asynchronous processing of Coq documents: from the kernel up to the user interface*, in "Proceedings of ITP", Nanjing, China, August 2015, https://hal.inria.fr/hal-01135919

[8] G. BARTHE, S. BELAID, F. DUPRESSOIR, P.-A. FOUQUE, B. GRÉGOIRE, P.-Y. STRUB. *Verified Proofs of Higher-Order Masking*, in "Advances in Cryptology – EUROCRYPT 2015", Sofia, Bulgaria, Lecture Notes in Computer Science, April 2015, vol. 9056 [*DOI :* 10.1007/978-3-662-46800-5_18], https://hal.inria.fr/hal-01216699

[9] G. BARTHE, T. ESPITAU, B. GRÉGOIRE, J. HSU, L. STEFANESCO, P.-Y. STRUB. *Relational Reasoning via Probabilistic Coupling*, in "Logic for Programming, Artificial Intelligence, and Reasoning - 20th International Conference, LPAR-20 2015", Suva, Fiji, 2015 [*DOI :* 10.1007/978-3-662-48899-7_27], https://hal.inria.fr/hal-01246719

[10] G. BARTHE, B. GRÉGOIRE, B. SCHMIDT. *Automated Proofs of Pairing-Based Cryptography*, in "Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security", Denver, United States, October 2015 [*DOI :* 10.1145/2810103.2813697], https://hal.inria.fr/hal-01246713

[11] S. BERNARD, Y. BERTOT, L. RIDEAU, P.-Y. STRUB. *Formal Proofs of Transcendence for e and π as an Application of Multivariate and Symmetric Polynomials*, in "Certified Programs and Proofs", St Petersburg, Florida, United States, J. AVIGAD, A. CHLIPALA (editors), ACM Press, January 2016, 12 p. , https://hal.inria.fr/hal-01240025

[12] Y. BERTOT. *Fixed Precision Patterns for the Formal Verification of Mathematical Constant Approximations*, in "Certified Programs and Proofs (CPP'15)", Mumbai, India, ACM, January 2015 [*DOI :* 10.1145/2676724.2693172], https://hal.inria.fr/hal-01074927

[13] C. COHEN, B. DJALAL. *Formalization of a Newton Series Representation of Polynomials*, in "Certified Programs and Proofs", St Petersburg, Florida, United States, J. AVIGAD, A. CHLIPALA (editors), January 2016, https://hal.inria.fr/hal-01240469

[14] C. DUNCHEV, F. GUIDI, C. SACERDOTI COEN, E. TASSI. *ELPI: fast, Embeddable, λProlog Interpreter*, in "Proceedings of LPAR", Suva, Fiji, November 2015, https://hal.inria.fr/hal-01176856

[15] A. FAITHFULL, J. BENGTSON, E. TASSI, C. TANKINK. *Coqoon An IDE for interactive proof development in Coq*, in "TACAS", Eindhoven, Netherlands, April 2016, https://hal.inria.fr/hal-01242295

Research Reports

[16] G. GONTHIER, A. MAHBOUBI, E. TASSI. *A Small Scale Reflection Extension for the Coq system*, Inria Saclay Ile de France, 2015, n^{o} RR-6455, https://hal.inria.fr/inria-00258384

[17] J. GRIMM. *Implementation of Bourbaki's Elements of Mathematics in Coq: Part Two; Ordered Sets, Cardinals, Integers*, Inria Sophia Antipolis ; Inria, 2015, n^{o} RR-7150, 685 p. , https://hal.inria.fr/inria-00440786

Other Publications

[18] Y. BERTOT. *Semantics for programming languages with Coq encodings*, March 2015, Lecture, https://hal.inria.fr/cel-01130272

[19] L. THÉRY. *Formally-Proven Kosaraju's algorithm*, February 2015, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01095533