INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team MARELLE

Mathematics, Reasoning, and Software

Sophia Antipolis

THEME SYM

Table of contents

1. Team

Head of project team
Yves Bertot [Research scientist INRIA]

Administrative Assistant
Nathalie Bellesso

Staff members INRIA
Loïc Pottier [Research scientist INRIA]
Laurence Rideau [Research scientist INRIA]
Laurent Théry [Research scientist INRIA]

Research scientists
Philippe Audebaud [Lecturer, ENS Lyon]
Frédérique Guilhot [Qualified teacher, *académie de Nice*]

Ph.D. students
Assia Mahboubi [Teaching Assistant]

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 complete functional correctness verification 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 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). 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.

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 automatically derive programs that are correct-by-construction.

3. Scientific Foundations

3.1. Type theory and formalization of mathematics

Keywords: *Coq*, *formalization*, *mathematics*, *type theory*.

The calculus of inductive constructions is powerful enough to formalize complex mathematics, based on algebraic structures and operations (as was also done in the Axiom computer algebra system). This is especially important as we want to produce proofs of logical properties for these 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, based on rich data structures. A third important characteristic of this calculus is that it is also a language for manipulating proofs, thanks to the Curry-Howard isomorphism. All this makes this calculus a tool of choice for our investigations. However, this language is difficult to learn, like an assembly language, although its conciseness and expressive power make it palatable for experts.

3.2. Verification of scientific algorithms

Keywords: *Coq, algorithms, certification.*

To produce certified algorithms, we use the following approach: instead of attempting to prove properties of an existing program (through a formalization of its semantics), we produce programs 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. Second, we concentrate on specific characteristics of the algorithm, and abstract away from the rest (*e.g.* memory management or data implementation strategies).

However, this approach also presents a few drawbacks. For instance, it is still difficult to prove properties of recursive algorithms where termination relies on advanced concepts. It is also difficult to work in an imperative style or with operations that have side-effects.

3.3. Programming language semantics

Keywords: *Coq, programming languages, semantics.*

We also investigate the algorithms that occur when implementing programming languages. 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 it preserves the semantics of programs (when the algorithm is a transformation or optimization algorithm) or that the programs produced are free of some unwanted behavior (when the algorithm is a compiler or a program verifier). For these algorithms, the difficulty sometimes lies in the size of the language description, and sometimes in the intrinsic complexity of the algorithm, which may use other algorithms such as graph traversal or unification algorithms. We usually talk about "proofs in programming language semantics" to refer to this class of algorithms and their correctness proofs. In addition to its intrinsic interest, this work is also useful for our work on scientific algorithms. Using the semantics of the programming language used to implement the algorithms, we can guarantee the correctness of these implementations.

3.4. Proof environments

Keywords: *Coq, environments, man-machine interface, proofs.*

We study how to improve mechanical tools for searching and verifying mathematical proofs used by engineers and mathematicians to develop software and formal mathematical theories. There are two complementary objectives. The first is to improve the means of interaction between users and computers, so that the tools become usable by engineers, who have otherwise little interest in proof theory, and by mathematicians, who have little interest in programming or other kinds of formal constraints. The second objective is to make it easier to maintain large formal mathematical developments, so they can be re-used in a wide variety of contexts. Thus, we hope to increase the use of formal methods in software development, both by making it easier for beginners and by making it more efficient for expert users.

4. Application Domains

4.1. Certified scientific algorithms

For some applications, it is mandatory to build zero-default software. One way to reach this high level of reliability is to develop not only the program, but also a formal proof of its correctness. In the Marelle team, we are interested in certifying algorithms for scientific computing. This is related to algorithms used in industry in the following respects:

- Arithmetics hardware in micro-processors,
- Arithmetical libraries in embedded software where precision is critical (global positioning, transportation, aeronautics),
- Verification of geometrical properties for robots (medical robotics),
- Fault-tolerant and dependable systems.

5. New Results

5.1. Tools for proof environments

5.1.1. Coqweb

Participant: Loïc Pottier.

Coqweb is a web interface to Coq, developped in collaboration with André Hirschowitz, Gang, Xiao et Joachim Yameogo. It is used by 10 teachers in mathematics at Nice and 200 students at the first level of University in mathematics since october 2004. For us the goal is experiment new techniques for solving ambiguities in mathematical expressions, and show that type theory is now able to efficiently manage elementary mathematics. For the teachers, the goal is to show to students thaht the notion of mathematical proof is not so magic, as it can be understood by a machine. Concretly, the coqweb interface allows teachers to develop exercices, and students to prove them following (or not) the indications given by the teacher. All that is done in the context of the WIMS server.

5.2. Type theory and formalization of mathematics

5.2.1. Relations between the $\mathcal{X}$ and $\lambda\mu$ Calculi

Participants: Philippe Audebaud, Steffen van Bakel [project-team MIMOSA].

The $\mathcal{X}$-calculus is strongly related to the sequent calculus, and consists in a quite simple encoding for the subset consisting in the implication rule, no structural rules, and a modified axiom. As such, it offers an extremely natural presentation of the classical propositional calculus with implication, and is a variant of system LK. While the $\lambda\mu$-Calculus handles multi conclusions in a natural deduction fashion, by allowing to focus on a single (active) conclusion at once, the $\mathcal{X}$-calculus is genuinely multi conclusions oriented. However, only the former calculus is confluent. Still, restricting ourselves to Call by Name and Call by Value reductions in $\mathcal{X}$ gives confluent subcalculi.

We set out to study inhabitation in $\lambda\mu$ via our understanding of $\mathcal{X}$. To this purpose, we have provided translations from $\mathcal{X}$ into (pure and typed versions of) $\lambda\mu$-calculus, based on the Call by Name reductions. For the Call by Value reduction, the result seems to be symmetric; however, we expect to give a more canonical approach to Call by Value reduction in $\lambda\mu$.

5.2.2. Typing with ambiguities

Participant: Loïc Pottier.

The main originality of coqweb is the langage used to write mathematics: type theory with ambiguities. This means that we allow a symbol to have more than one meaning, i.e. more that one type. Combined with the existence of coercions (i.e. operators without lexical incarnation) in functional position or in argument position, this lead to a lazy typing algorithm which has been difficult to implement, specially when new functionnalities have been added and when complexity had to be reduced. However, the implementation is now reasonably good to treat non trivial examples, for example those developed in category theory by a M2 student this summer.

5.2.3. Toward Geometric Views on the λ-Calculus

Participant: Philippe Audebaud.

We showed that type interpretations benefit a more uniform and consistent presentation as closed sets of particular polynomials built on an extension of the λ-calculus. The purpose of our work is thus to provide means for using well understood techniques and formalisms from commutative algebra and algebraic geometry for the study of local properties of terms.

Compared to the former presentation, we have progressed by ensuring the λ calculus can be extended consistently with ring operations in such a way that these operations can be handled *transparently* in the quotient of our previous E-calculus with respect to the ring equations. Also, the notion of polynomial is

now formally the same as the notion of free term. Finally, we have applied our model to Berardi's pruning techniques used for dead code analysis. This work is published in [9].

5.2.4. Co-recursion and real numbers

Participant: Yves Bertot.

Recursive functions can be designed to produce conceptually infinite data structure and consume these data-structures on a call-by-need basis. Previous work by T. Coquand, L. Paulson, and E. Gimenez showed that these functions belonged to a class of recursive functions that is different from the class that is usually considered in theorem provers. This new class of functions is called the class of co-recursive functions and the infinite data-structures are known as co-inductive structures. However, none of the previous studies had considered partial functions. We showed that partial function could be modeled if one relied on a co-inductive description of their domain of definition and applied this technique to a description of Eratosthene's sieve [4].

The traditional understanding that real numbers are fractional numbers with an infinite sequence of digits after the decimal point can be modeled by representing real numbers as infinite sequences of digits using co-inductive types. To be effective, the digits must depart from the traditional approach to have *redundant digit systems* as first advocated by Cauchy. These co-inductive types have already been studied by Ciaffaglione and di Gianantonio and we have extended their approach to show that one could easily compute power series.

5.2.5. Formalization in Coq of high-school mathematics

Participants: Frédérique Guilhot, Loïc Pottier.

An article describing the work of Frédérique Guilhot on high-school geometry for Coq has been published in a French-speaking journal [3].

Based on the library dedicated to high-school geometry developped by Frédérique Guilhot, we collaborate with Marc Daumas, Jean Duprat and Guillaume Melchion at ENS Lyon, who have the project to develop a web environment where teachers in mathematics (in high school and "classes préparatoires") could develop courses in mathematics with interactive exercises. This environment is also based on coqweb.

5.2.6. A sharp efficiency increase for ring equalities

Participants: Benjamin Grégoire, Assia Mahboubi, Loïc Pottier.

We have explored a new approach to proving equalities between polynomial expressions in a ring using reflection. This new approach is based on the Hörner encoding of polynomial expressions. Based on a variety of test cases, this new approach proves to be much more efficient than the traditional approach based on full development and ordering of monomials. This work has been published in [6].

5.2.7. Verifying probabilistic algorithms

Participants: Philippe Audebaud, Tahina Ramananandro [ENS Ulm internship], Laurent Théry.

We have developed a deep embedding in Coq of the probabilistic functional language λO proposed in work by Park, Pfenning and Thrun. This lets us prove the correctness of some basic λO programs that simulate standard distribution laws.

We have also developed a plateform for generating proof obligations from annotated λO programs which aims at the integration into `why`.

5.2.8. Number theory

Participant: Laurent Théry.

Some well-known properties of numbers have been formalised to enhance Coq's libraries for elementary number theory. A first property that has been formalised is that a number n can be written as the sum of two square numbers if and only if each prime factor$p$ of n that is equal to 3 modulo 4 has an even exponent in the decomposition of n. The proof of this theorem relies on Wilson's theorem.

A second effort is the formalization of some well-known primality tests: Pocklington certificates, Pepin's test and Lucas' test. These tests rely on theorems like Lagrange's theorem and Fermat's little theorem. These

tests have been used to generate relatively large (upto 2000 digits) certified prime numbers in Coq using the reflection mechanism. This is joint work with Benjamin Grégoire (Everest) and Benjamin Werner (Logical).

5.2.9. Computer arithmetics and relevant proof tools

Participants: Yves Bertot, Laurent Théry.

An initial formalisation for an efficient summation algorithm has been presented in work by Fousse and Zimmermann (SPACES) . The formalisation has been completed to cover all the theorems presented in the paper including the worst-case examples. This is joint work with Laurent Fousse and Paul Zimmermann (SPACES).

Simplifying equalities and inequalities over polynomial expressions is one of the most frequent task when doing formal proofs on computer arithmetics. We have developed ad-hoc tactics to provide a more user friendly interaction for dealing with such equalities and inequalities within the Coq prover. The idea behing the tactics is presented in [11]

Our work on the formalization of square, cubic, and nth root algorithms for integers has been submitted and accepted for publication [2].

5.2.10. Cylindric algebraic decomposition

Participants: Yves Bertot, Assia Mahboubi, Loïc Pottier, Laurence Rideau, Laurent Théry.

Assia Mahboubi implemented the cylindric algebraic decomposition of Collins in Coq [7]. The main work of this year has been to test this implementation on a variety of examples, organize the collaborative work for the proof of correctness, and prove the basic blocs. The various elements that have been addressed this year are:

- Proving the ring properties for our efficient polynomial representation (sparse Hörner representation). The difficulty lies in working modulo an equivalence relation on the coefficients and proving the correctness of the efficient operations. This work is complete and represents 2000 lines of Coq.
- Proving the correctness of the procedure for isolating the roots of single variable polynomials (based on a simplification of Descartes' law of signs). This work is not complete yet.
- Proving the correctness of the optimized polynomial division algorithm,
- Proving the correctness of the subresultant computation (based on Basu, Pollack, and Roy's method of symbolic determinants).

The novelty of this work is that we address optimized data structures and algorithms, to make sure that the complete algorithm will have a reasonable complexity. Ultimately, this implementation should be part of a usable decision procedure for use in our other proof developments in the Coq system.

5.3. Programming language semantics

5.3.1. Optimizations at the RTL-CFG level

Participants: Yves Bertot, Benjamin Grégoire [project-team EVEREST], Xavier Leroy [project-team CRISTAL].

Our work on the formal verification of optimizations in a compiler for a subset of C has been published in [5].

5.3.2. Parallel Move

Participants: Laurence Rideau, Bernard Serpette [project-team OASIS].

During the previous year (2004) we studied a well-known algorithm for the parallel assignment of registers, where the values of the target registers are taken from a collection of source registers, among which the target registers may also occur. The difficulty was to find a suitable order so that the value of a source register were not overwritten before it had been moved to a target register, while using only one extra temporary register.

This algorithm had been formally proved. During this year, work has been done to adapt our proof to integrate it to in the whole compiler proof: the first version of the algorithm was an abstract one using abstract registers .We have adapted the algorithm to work on the reals registers of the compiler.

This work has been published in [8].

6. Other Grants and Activities

6.1. National initiatives

- We participate in the national contract A3PAT, scheduled to start at the end of 2005. Other participants in this contract are CEDRIC-CNAM (Evry) LABRI (Bordeaux), and LRI (Orsay). The objective of this contract is to study the possible combination of the rewriting engine Cime and the Coq system, especially in the verification that recursive algorithms do terminate.
- We participate in the national contract CompCert, scheduled to start at the end of 2005. Other participants in this contract are the project-team CRISTAL (INRIA Rocquencourt), CEDRIC-CNAM (Evry), and PPS (Paris). The objective of this contract is to study the development of a formally verified compiler for a significant subset of C.
- We participate in the collaborative laboratory between INRIA and Microsoft Research, in the Collaborative research action “Mathematical components”. Other participants in this contract are the project-tream LOGICAL and PROVAL at INRIA-Futures.

6.2. European initiatives

Marelle participates in the networks Types (type theory).

7. Dissemination

7.1. Conference and workshop attendance, travel

Yves Bertot Yves Bertot attended the conference TLCA’05 in Nara, Japan in April and the IFIP working conference on Verified Software, Theories, Tools, and Experiments in Zürich, Switzerland in October. He was a lecturer at the Types summer school in Göteborg, Sweden, in August, and attended a workshop on Constructive analysis, types and exact real numbers in Nijmegen in October. In all these events, he presented some work.

Assia Mahboubi Assia Mahboubi presented a paper at the Dagstuhl Seminar on “Mathematics, Algorithms, and Proofs” in Dagstuhl (Germany) in January. She gave an invited talk at the Colloquium of the Laboratory E. Picard in Toulouse in January. She visited Radboud University in Nijmegen for one month in March. She presented a paper at the TPHOLs conference in Oxford in August, and attended the Types summer school in Göteborg, Sweden. She attended a workshop on Constructive analysis, types and exact real numbers, in Nijmegen in October. She gave an in invited lecture at the workshop on Real algebra, quadratic forms and model theory, part of the special semester on Real Geometry at Institut Henri Poincaré in Paris in November.

Loïc Pottier Loïc Pottier participated to the Dagstuhl Seminar on “Mathematics, Algorithms, and Proofs” in Dagstuhl (Germany) in January. He attended the last Mowgli meeting in Bruxelles in march. He gave a talk on coqweb at the Journée de géométrie organized by Jean Duprat àt ENS Lyon in june.

Laurence Rideau Laurence Rideau presented a paper at *the conference JFLA’05* in March and attended the Types summer school in Göteborg, Sweden in August.

Laurent Théry Laurent Théry was a lecturer at the Types summer school in Göteborg, Sweden, in August. He gave a lecture at the workshop “Qualité et Sûreté du Logiciel” in Nancy in February.

7.2. Leadership within scientific community

- Yves Bertot was a member of the program committee for TPHOLS'05, UITP'05,
- Laurent Théry organized a workshop on formal methods at the 17th IMACS World Congress in Paris (July 05), with Marc Daumas (project-team Arénaire).
- Project members reviewed papers for the journals Information and Computation, ITA and TSI, and for the conferences RNC, FOSSACS, RTA'05, TPHOLs'05, and UITP'05.

7.3. Miscellaneous

- Yves Bertot was an external examiner for the application of Dr. A. Bove to a position of assistant professor at University of Chalmers (Sweden).
- Yves Bertot was a reviewer for the thesis of O. Boite (CNAM) and a member of the Jury for the thesis O. Ponsini (Université de Nice).
- Yves Bertot was an external examiner for the evaluation of the CNRS laboratory IRIT at Toulouse in November.
- Yves Bertot is a member of the jury for the SPECIF thesis award.
- Yves Bertot is a member of the *Conseil National des Universités* (National University Council), 27th section. This position brings more than a month of work in reviewing nationwide applications for university professor positions.
- Laurent Théry was a member of the defence Jury for the thesis of Kristofer Johannisson at Chalmers University (Sweden).

7.4. Supervision of Ph.D. projects

- Loïc Pottier supervises the Ph.D. projects of Assia Mahboubi and Frédérique Guilhot.

7.5. Teaching

Philippe Audebaud *Programmation fonctionnelle* TPs 12h L3 Informatique et 24h L2 Informatique. *Statistiques avec Excel* 48h L1 MASS. *Algorithmique théorique* 42h L2 MASS. Plus divers dont encadrement TE (Travaux d'étude, L3 Informatique), pour un total de 160h.

Yves Bertot *Sémantique des langages de programmation I* (Programming language semantics I), 1st year Master (18 hours), University of Nice. *Programmation fonctionnelle et preuves* (Proofs and Functional Programming), 1st year Master, (32 hours), ENS Lyon, *Sémantique des langages de programmation II* (Programming language semantics II), 2nd year Master (5th year, 24 hours), University of Nice, *Proof mechanisation*, 2nd year Master (5th year, 6 hours).

Assia Mahboubi *Informatique théorique, Programmation en C, Systèmes d'exploitation, Logique, Programmation en Java* (theoretical computer science, programming in C and Java, Operating systems, and Logic, for a total of 130 hours between the second semester of 2004-2005 and the first semester of 2005-2006),

Loïc Pottier *Sémantique des langages de programmation I* (Programming language semantics I), 1st year Master (18 hours), University of Nice. *Preuves formelles* (Formal proofs), 2nd year Master (6 hours), University of Aix-Marseille

Laurent Théry *Formal methods and advanced programming languages*, University of L'Aquila, Italy (50 hours), *Proof Mechanisation*, 2nd year Master, University of Marseilles (6 hours).

8. Bibliography

Major publications by the team in recent years

[1] Y. BERTOT, P. CASTÉRAN. *Interactive Theorem Proving and Program development*, Springer, 2004.

Articles in refereed journals and book chapters

[2] Y. BERTOT. *Vérification formelle d'extractions de racines entiéres*, in "RSTI– Techniques et Sciences de l'informatique", vol. 24, 2005, p. 1161–1185.

[3] F. GUILHOT. *Formalisation en Coq d'un cours de géométrie pour le lycée*, in "RSTI – Techniques et Scienses de l'informatique", vol. 24, 2005.

Publications in Conferences and Workshops

[4] Y. BERTOT. *Filters on CoInductive Streams, an Application to Eratosthenes' sieve*, in "Typed Lambda Calculi and Applications, TLCA 2005", P. URZYCZYN (editor). , Springer-Verlag, 2005, p. 102–115.

[5] B. GRÉGOIRE, Y. BERTOT, X. LEROY. *A structured approach to proving compiler optimizations based on dataflow analysis*, in "Proceedings of TYPES'04", J.-C. FILLIÂTRE, C. MARCHÉ, C. PAULIN (editors). , Lecture Notes in Computer Science, to appear, Springer-Verlag, 2005, http://www-sop.inria.fr/everest/personnel/Benjamin.Gregoire/Publi/concert.ps.gz.

[6] B. GRÉGOIRE, A. MAHBOUBI. *Proving equalities in a commutative ring done right in Coq*, in "Theorem Proving in Higher Order Logics (TPHOLS'05)", LNCS, nº 3603, Springer Verlag, 2005.

[7] A. MAHBOUBI. *Programming and certifying the CAD in the Coq system*, in "Mathematics, Algorithms, Proofs", to appear, nº Seminar No. 05021, Dagstuhl International Conference and Research Center for Computer Science, January 2005.

[8] L. RIDEAU, B. P. SERPETTE. *Coq à la conquête des moulins*, in "Journées Francophones des Langages Applicatifs", March 2005.

Internal Reports

[9] P. AUDEBAUD. *Toward a geometric view on computations*, Technical report, nº RR-5492, Inria, july 2005, http://www.inria.fr/rrrt/rr-5492.html.

[10] Y. BERTOT. *Calcul de formules affines et de séries entières en arithmétique exacte avec types co-inductifs*, Technical report, 2005, http://hal.inria.fr/inria-00000480/en/.

[11] L. THÉRY. *Simplifying Polynomial Expressions in a Proof Assistant*, Research report, nº RR-5614, INRIA, June 2005, ftp://ftp.inria.fr/INRIA/publication/publi-pdf/RR/RR-5614.pdf.