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 **1**
2. Overall Objectives **1**
2.1. Overall Objectives 1
3. Scientific Foundations **1**
3.1. Type theory and formalization of mathematics 1
3.2. Verification of scientific algorithms 2
3.3. Programming language semantics 2
3.4. Proof environments 2
4. Application Domains **3**
4.1. Certified scientific algorithms 3
5. New Results **3**
5.1. Type theory and formalization of mathematics 3
5.1.1. Relations between the $\mathcal{X}$ and $\lambda\mu$ Calculi 3
5.1.2. Typing with ambiguities 3
5.1.3. Formalization in Coq of high-school mathematics 3
5.1.4. Cylindric algebraic decomposition 3
5.1.5. Group theory 4
5.2. Verification of scientific algorithms 4
5.2.1. Co-recursion and real numbers 4
5.2.2. Number theory 4
5.2.3. Formalising Elliptic Curves 5
5.2.4. Reflecting Buchberger Algoritm 5
5.3. Programming language semantics 5
5.3.1. Formal verification of probabilistic programs 5
5.3.2. General recursion with Tarski's fixpoint theorem 5
5.3.3. An integrated study of semantic styles 6
5.4. Tools for proof environments 6
5.4.1. Coqweb 6
5.4.2. Proof script maintenance 6
6. Other Grants and Activities **6**
6.1. National initiatives 6
6.2. European initiatives 7
7. Dissemination **7**
7.1. Conference and workshop attendance, travel 7
7.2. Leadership within scientific community 7
7.3. Miscellaneous 7
7.4. Supervision of Ph.D. projects 7
7.5. Teaching 8
8. Bibliography **8**

1. Team

Head of project team
Yves Bertot [Research scientist INRIA, HdR]

Vice-head of project team
Laurence Rideau [Research scientist INRIA]

Administrative Assistant
Nathalie Bellesso [Administrative assistant]

Staff members INRIA
Loïc Pottier [Research scientist INRIA, HdR]
Laurent Théry [Research scientist INRIA]

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

Ph.D. students
Sidi Ould Biha [Phd. student, arrived on Sept. 1st, advised by L. Théry]
Nicolas Julien [Teaching Assistant, arrived on Oct. 1st, advised by Y. Bertot]
Assia Mahboubi [Teaching Assistant, left on Sept. 1st, advised by L. Pottier]

Master students
Sidi Ould Biha [Master in Mathematics at the University of Nice]
Nicolas Julien [Master in Computer Science at the University of Nice]
Etienne Miret [1st year student at Ecole Normale Supérieure de Lyon]

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 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 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 automatically derive programs that are correct by construction.

In 2006, we focussed on algorithms concerning polynomials, arithmetic with large numbers, and finite-group theory.

3. Scientific Foundations

3.1. Type theory and formalization of mathematics

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

The calculus of inductive constructions is a branch of type theory that serves as 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

Keywords: *Coq*, *algorithms*, *certification*.

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. Second, 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). Thus, 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 explicitely 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

Keywords: *Coq*, *programming languages*, *semantics*.

To bridge the gap between our high-level descriptions of algorithms and conventional programming languages, we also investigate the algorithms that occur when implementing programming languages, 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 of bugs.

3.4. Proof environments

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

We study how to improve mechanical tools for searching and verifying mathematical proofs so that they become practical for 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 and programs for scientific computing. This is related to algorithms used in industry in the following respects:

- Arithmetical 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. Type theory and formalization of mathematics

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

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

We investigated relations, both syntactic and computational between the $\mathcal{X}$-calculus and the $\lambda\mu$-calculus. This results in a new translation scheme for proof derivations for classical logic into the $\lambda\mu$-terms, which preserves types and reflects $\mathcal{X}$ propagation rules, as soon as so-called CBN and CBV strategies are fixed among propagation rules, hence avoiding critical pairs in the source calculus. This work is under publication process. Beyond this step, we have extended our understanding to more connectors in $\mathcal{X}$ and to negation. This led us to a calculus which comes very close to Wadler's dual calculus.

5.1.2. Typing with ambiguities

Participant: Loïc Pottier.

The main originality of coqweb is the language used to write mathematics, which is based on type theory with ambiguities. The algorithm for ambiguous typing which is integrated to coqweb has been extended. Overloading and coercions are now available for types themselves. Implementation techniques have been designed to improve the efficiency of this algorithm, which was very slow in some cases, and even non-terminating.

This new typing algorithm has been tested on several formal developments made by members of the Laboratory of Mathematics of the University of Nice: set theory, group theory, category theory, calculus. This algorithm is described in the research report [12].

5.1.3. Formalization in Coq of high-school mathematics

Participant: Frédérique Guilhot.

We completed our study of plane geometry with formal proofs of Heron's formula and Morley's theorem.

We also developed formal proofs for theorems concerned with the real numbers: the divergence of the harmonic series and the convergence of the series containing the inverses of triangular numbers.

5.1.4. Cylindric algebraic decomposition

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

Assia Mahboubi implemented the cylindric algebraic decomposition algorithm of Collins in Coq. The various elements that have been addressed this year are:

- Completing proofs of the ring properties for our efficient polynomial representation (sparse Horner representation).
- Proving the correctness of the procedure for isolating the roots of single variable polynomials (based on a simplification of Descartes' law of signs). A second part has been added; this part concerns the computation of Bernstein coefficients for split intervals from the coefficient on the initial interval.
- Proving the structure lemma for the sub-resultant algorithm, this proof relies heavily on a notion of symbolic determinants that we had to formalize first (using the method proposed in Basu, Pollack, and Roy's work).
- Proving the correctness of the optimized polynomial division algorithm. The main characteristic of this algorithm is that it is structurally recursive. This makes it possible to have an elementary proof of correctness, even though this proof is long and tedious. A few lemmas are still needed for this proof.

This work is described in Assia Mahboubi's Ph.D. thesis, which was defended in November [3] and in two articles [9] and [5].

5.1.5. Group theory

Participants: Georges Gonthier [Microsoft Research], Laurence Rideau, Laurent Théry.

We participate in the collaborative research agreement "Mathematical Components" with Microsoft Research. This project aims at evaluating the applicability of a new approach to mathematical proofs called "small-scale reflection", especially in the domain of finite group theory. As first steps, we defined the notions of centralizers, normal subgroups, and actions on groups, and we verified formally a collection of foundation theorems: the Lagrange theorem and the Sylow theorems. The interest of this work does not only lie in the fact that are able to verify these theorems but also in that we are establishing a new method for the formal description of abstract mathematics. It is described in [13].

5.2. Verification of scientific algorithms

5.2.1. Co-recursion and real numbers

Participants: Yves Bertot, Nicolas Julien.

The traditional understanding that real numbers are fractional numbers with an infinite sequence of digits after the decimal point can be modeled using infinite streams of digits, a special case of co-inductive datatypes. To be effective, the digits must depart from the traditional approach to have *redundant digit systems*. We develop a library of basic functions for exact real computation based on this idea. In particular, we concentrate on the implementation of series and we take as examples mathematical constants such as e and π. The main work of this year was to adapt this library to an arbitrary basis (instead of the basis 2 which was used before [4]). As a result, we are able to compute certified bounds for small intervals containing e and π of width 2^{-500} and 2^{-300}, respectively, in less than a minute inside the theorem prover. This work constitutes most of Nicolas Julien's Master thesis [14].

5.2.2. Number theory

Participants: Sidi Ould Biha, Benjamin Grégoire [project-team EVEREST], Laurent Théry, Benjamin Werner [project-team LOGICAL].

Prime certificates such as Pocklington certificate provide elegant and short proofs of primality. We have developed an efficient way of verifying such certificates inside the Coq prover. This work is presented in [8].

We developed a certified C library for arbitrary precision arithmetics using the verification condition generator Caduceus. On top of this library, we then certified a C implementation of the Lucas-Lehmer primality test. With this program we were able to verify the primality of Mersenne numbers up to $2^{44497}+1$. This work is presented in Sidi Ould Biha's master thesis.

We have improved the binary arithmetic of Coq by providing an implementation of modular arithmetic based on binary trees. Changing datastructures made it possible to use efficient recursive algorithms such as Karatsuba multiplication. This work is presented in [7].

5.2.3. Formalising Elliptic Curves

Participants: Laurent Théry, Benjamin Grégoire [project-team EVEREST].

Elliptic curves are widely used in cryptographic systems. For this reason, we developed a formal library for elliptic curves inside Coq. If defining the curves is quite direct, the difficult part is well-known: it is the associativity of the group law. Proofs in the literature are usually by means of a geometric argument. For the moment, such argument would not be easy to formalize in a prover. We have followed a more algebraic approach using intensive computation to formally get the associativity. With this library, it has then been possible to formally prove the Goldwasser-Kilian algorithm.

5.2.4. Reflecting Buchberger Algoritm

Participant: Laurent Théry.

With the progresses of both the evaluation mechanism and the arithmetic operations, it is now possible to perform non-trivial computation inside Coq. In an attempt to merge theorem proving and computer algebra techniques in a single system, we have turned an existing certified version of the Buchberger algorithm described in [2] into a reflective algorithm that can be directly evaluated inside Coq.

5.3. Programming language semantics

5.3.1. Formal verification of probabilistic programs

Participants: Philippe Audebaud, Christine Paulin [project-team PROVAL].

Randomized algorithms are widely used either for finding efficiently approximated solutions to complex problems, for instance in primality testing, or for obtaining good average behaviour, for instance in distributed computing. We propose a new method for proving properties of these algorithms in a proof assistant based on higher-order logic, which only uses functionals and algebraic properties of the unit interval. We follow Kozen by interpreting probabilistic programs as measure transformers; the originality of our approach is to view this interpretation as a monadic transformation on functional programs. We apply this to the formal verification of two programs: one implements a Bernouilli distribution from a coin flip and the other studies the termination of a random walk. All these results are formalized using Coq and described in the publication [6].

We are currently investigating rules for total correctness for λ_o, an enhanced functional calculus, designed by Park, Pfenning and Thrun. As new results, we obtain a denotational and an inference system for axiomatic semantics à la Hoare. More work is needed to define accurate rules for total correctness. As a paradigmatic example, we study a program implementing the geometric law (where the problem is to obtain a uniform distribution in a geometric figure A from a uniform distribution in a figure B such that A is included in B). This is an example of a non-terminating program that should be accepted because in terminates almost surely.

5.3.2. General recursion with Tarski's fixpoint theorem

Participant: Yves Bertot.

We transposed ideas from domain theory to the practical description of recursive functions. The main idea is that a fixpoint theorem, like Tarski's theorem in the realm of complete partial orders, can be used as a programming tool to define new recursive functions. In the case of recursive functions, this means we use non-constructive mathematics to justify the existence of functions that are not guaranteed to terminate (a significant departure from the type theoretic tradition). However, we show that the extraction mechanism can be adapted in a simple way to give a practical meaning to the functions obtained in this manner. This work is described in [11].

5.3.3. An integrated study of semantic styles

Participant: Yves Bertot.

We designed an experiment to show the relations between a variety of approaches to programming language semantics, using a very simple imperative programming language as example. Our example covers two variants of operational semantics: natural semantics (à la Kahn) and structural operational semantics (à la Plotkin); two variants of program verification semantics: axiomatic semantics (à la Hoare) and weakest pre-condition calculus (à la Dijkstra); denotational semantics (à la Scott); and abstract interpretation (à la Cousot & cousot). The main interest of this experiment is to show how the proofs of consistency between the various styles can be performed formally and to describe the relation between non-executable semantics and executable tools: interpreters, verification condition generators, and static analysis tools.

5.4. Tools for proof environments

5.4.1. Coqweb

Participant: Loïc Pottier.

Coqweb is a web interface to Coq, developed in collaboration with André Hirschowitz, Gang Xiao et Joachim Yameogo from the laboratory of mathematics of the University of Nice. It has been used since october 2004 by 10 teachers in mathematics at the University of Nice for 200 students in first year.

An experience has been made with A. Hirschowitz of a wiki which integrates coqweb. The goal is to allow several mathematicians (researchers and teachers) to develop mathematical theories both with unformal (usual) texts and formal texts, with formal proofs, in connection with the Coq system and using the ambiguous typing algorithm designed for the coqweb tool. The address for this tool is
http://pcmath165.unice.fr/wcw/spikini

5.4.2. Proof script maintenance

Participants: Yves Bertot, Etienne Miret.

In conventional usage, formally verified proofs are long sequences of commands for theorem proving tools. These sequences are called *proof scripts* and usually have very little structure. We developed a tool to add structure to a proof script by recording the dependencies between various commands. The structure information can then be used in maintenance activities to support systematic update of the proof script when adapting to new versions of libraries or proof tools. In the long run, we also hope such a tool could be used to help port formal proofs from one proof system to another.

6. Other Grants and Activities

6.1. National initiatives

- We participate in the national contract A3PAT, which started on Dec. 1st 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, which started on Jan. 1st 2006. 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 common laboratory between INRIA and Microsoft Research, in the Collaborative research action "Mathematical components". Other participants in this contract are the INRIA project-teams LOGICAL and PROVAL. The goals of this contract is to study the impact of small-scale reflective approaches to the formalization of mathematics, especially in finite group theory and to experiment with extension of theorem provers with native arithmetics.

6.2. European initiatives

Marelle participates in the network Types (type theory).

7. Dissemination

7.1. Conference and workshop attendance, travel

Yves Bertot attended the JFLA'06 conference in January, an NSF sponsored workshop in Palo Alto in April, a school for young researchers in programming in June, a Dagstuhl workshop on the challenge of software verification in July, and the PariSTIC workshop on reserch in software security and reliability funded by the French national research agency (ANR-Setin) in November.

Assia Mahboubi gave a talk at the MAP meeting (Mathematics, Algorithms, Proofs) in Castro Urdiales in Spain in January. She presented a paper at the IJCAR conference in Seattle in August.

Loïc Pottier participated to the MAP meeting (Mathematics, Algorithms, Proofs) in Castro Urdiales in Spain in January. He gave a talk at the "Hot topics workshop on the evolution of mathematical communication in the age of digital libraries" in December in Minneapolis.

Laurence Rideau attended the JFLA'06 conference in January.

Laurent Théry was a lecturer at the Special Semester on Gröbner Bases in Linz in March. He attended the ICMS conference in Castro Urdiales in Spain in September and gave a demonstration in mathematical proofs on the computer. He gave a talk at the Symposium on Grand Challenge in Informatics in Budapest in September.

7.2. Leadership within scientific community

- Yves Bertot was a member of the program committee for UITP, MSFP.
- Yves Bertot gave talks at the School for young researchers in programming on *introduction to type theory* and *Coq in a Hurry*.
- Laurent Théry organized the TYPES workshop on numbers and proofs in Orsay in June, in collaboration with Benjamin Grégoire (from the Everest project).
- Project members reviewed papers for the journals AMAI (Annals of mathematics and aritificial intelligence), TSI (Techniques et Sciences Informatiques), JLC (Journal of Logic and Computation), and for the conferences IJCAR, MSFP, MPC, FM, UITP.

7.3. Miscellaneous

- 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.
- Loïc Pottier was member of two Ph.D. defence committees: Assia Mahboubi (as Ph.D. advisor) and Julien Narboux (as report author).

7.4. Supervision of Ph.D. projects

- Loïc Pottier supervised the Ph.D. project of Assia Mahboubi, which was successfully defended in November.
- Laurent Théry supervises the Ph.D. project of Sidi Ould Biha, which started on Sept. 1st, with funding from the INRIA-Microsoft research common laboratory.
- Yves Bertot supervises the Ph.D. project of Nicolas Julien, which started on Oct. 1st, with funding from the French ministry of research and a teaching assistant grant.

7.5. Teaching

Philippe Audebaud *Programmation fonctionnelle* (functional programming) 2nd and 3rd year License (36 hours), University of Nice, *Statistiques avec Excel* (statistics using Excel) 1st year License (48 hours), University of Nice, *Algorithmique théorique* (theory of algorithmics) 2nd year License (42 hours), University of Nice.

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 mechanization*, 2nd year Master (5th year, 6 hours) University of Marseille.

Loïc Pottier *Sémantique des langages de programmation I* (Programming language semantics I), 1st year Master (50 hours), University of Nice, *Preuves formelles* (Formal proofs), 2nd year Master (20 hours), University of Nice, *Preuves formelles* (Formal proofs), 2nd year Master (3 hours), University of Aix-Marseille.

Laurent Théry *Formal methods and advanced programming languages*, University of L'Aquila, Italy (40 hours), *Proof Mechanization*, 2nd year Master, University of Marseilles (12 hours). *Introduction to Coq*, École des Mines, (3 hours).

8. Bibliography

Major publications by the team in recent years

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

[2] L. THÉRY. *A Machine-Checked Implementation of Buchberger's Algorithm*, in "Journal of Automated Reasoning", vol. 26, 2001, p. 107–137.

Year Publications

Doctoral dissertations and Habilitation theses

[3] A. MAHBOUBI. *Contribution à la certification des calculs dans R : théorie, preuves, programmation*, Ph. D. Thesis, Université de Nice-Sophia Antipolis, Nice, 2006.

Articles in refereed journals and book chapters

[4] Y. BERTOT. *Affine functions and series with co-inductive real numbers*, in "Mathematical Structure in Computer Sciences", to appear, vol. 17, n^{o} 1, 2007.

[5] A. MAHBOUBI. *Implementing the cylindrical algebraic decomposition within the Coq system*, in "Mathematical Structure in Computer Sciences", to appear, vol. 17, n^{o} 1, 2007.

Publications in Conferences and Workshops

[6] P. AUDEBAUD, C. PAULIN. *Proofs of randomized algorithms in Coq*, in "Mathematics of Program Construction", T. UUSTALU (editor). , n^o 4014, LNCS, 2006.

[7] B. GRÉGOIRE, L. THÉRY. *A purely functional library for modular arithmetic and its application for certifying large prime numbers*, in "3rd International Joint Conference on Automated Reasoning (IJCAR)", U. FURBACH, N. SHANKAR (editors). , Lecture Notes in Artificial Intelligence, vol. 4130, Springer-Verlag, 2006, p. 423-437.

[8] B. GRÉGOIRE, L. THÉRY, B. WERNER. *A Computational Approach to Pocklington Certificates in Type Theory.*, in "Functional and Logic Programming Symposium (FLOPS)", LNCS, vol. 3945, Springer, 2006, p. 97-113.

[9] A. MAHBOUBI. *Proving Formally the Implementation of an Efficient gcd Algorithm for Polynomials*, in "3rd International Joint Conference on Automated Reasoning (IJCAR)", U. FURBACH, N. SHANKAR (editors). , Lecture Notes in Artificial Intelligence, vol. 4130, Springer-Verlag, 2006, p. 438-452, https://hal.inria.fr/inria-00001270/en.

[10] L. THÉRY, P. LETOUZEY, G. GONTHIER. *Coq.*, in "The Seventeen Provers of the World", LNCS, vol. 3600, Springer, 2006, p. 28-35.

Internal Reports

[11] Y. BERTOT. *Extending the Calculus of Constructions with Tarski's fix-point theorem*, Technical report, INRIA, October 2006, http://hal.inria.fr/inria-00105529.

[12] L. POTTIER. *Ambiguous typing*, Research Report, n^o 6041, INRIA, December 2006, http://hal.inria.fr/inria-00117458.

[13] L. THÉRY, L. RIDEAU. *Formalising Sylow's theorems in Coq*, Technical Report, n^o 0327, INRIA, November 2006, https://hal.inria.fr/inria-00113750.

Miscellaneous

[14] N. JULIEN. *Vérification formelle d'arithmétique réelle exacte et fonctions analytiques*, Technical report, Ecole doctorale STIC, Université de Nice, 2006.

[15] S. OULD BIHA. *Vérification d'un algorithme de test de primalité avec Caduceus*, Technical report, Ecole doctorale SFA, Université de Nice, 2006.