INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team LogiCal

Logic and Calculus

Futurs

THEME 2A

Table of contents

1. Team **1**
2. Overall Objectives **1**
3. Scientific Foundations **2**
3.1. Formalisation of mathematics 2
3.2. The Calculus of Inductive Constructions 2
3.3. Environments for interactive proof development 3
3.4. Deduction Modulo 3
3.5. Proofs and programs 3
4. Application Domains **4**
5. Software **4**
5.1. Coq 4
5.2. Why 5
5.3. Krakatoa 5
5.4. Fatalis 6
6. New Results **6**
6.1. Development of theories and tactics 6
6.1.1. Extraction of modules 6
6.1.2. Extraction of arbitrary precision numbers 6
6.1.3. Timed automata 6
6.1.4. Proof languages 7
6.1.5. Tactic implementing a first order sequent calculus 7
6.1.6. Formalisation of euclidean geometry 7
6.1.7. Unification modulo AC 7
6.1.8. Proof of Java programs 7
6.1.9. formalisation of sets and maps datatypes 7
6.1.10. Efficient primality test with Pocklington's criterion 8
6.1.11. Four-color Theorem 8
6.1.12. Safety of protocols in aeronautic 8
6.2. Development of systems 8
6.2.1. Release of Coq V7.4 8
6.2.2. Release of Coq V8.0 8
6.2.3. Tactics dealing with existential variables 8
6.2.4. Compilation of proofs towards a virtual machine 9
6.2.5. Incorporating rewriting to Coq 9
6.2.6. Why 9
6.2.7. Release of Krakatoa 9
6.2.8. CiME 9
6.3. Studies of formalisms 9
6.3.1. Induction over real numbers 9
6.3.2. Decision methods 9
6.3.3. Semantics of Type Theory 10
6.3.4. Cut elimination in set theory 10
6.3.5. Cut elimination in sequent calculus modulo 10
6.3.6. Asymetric deduction modulo 10
6.3.7. Automated termination proof 10
6.3.8. Name-free λ-calculus with explicit substitutions 10

6.3.9. Equivalence of conversion in Church and Curry styles in the Calculus of Inductive Constructions 11
6.3.10. Computational content of “ex falso quolibet” 11
6.3.11. Higher-order unification and type inference 11
6.3.12. Epsilon-symbol and Foundations of Linking 11
6.3.13. Defunctorisation 11
6.3.14. ML with refinement 11
6.3.15. Extentionality in the Calculus of Inductive Constructions 11
7. Contracts and Grants with Industry 12
7.1. Averroes 12
7.2. France Télécom 12
7.3. Verificard 12
7.4. Geccoo 12
7.5. Modulogic 12
8. Other Grants and Activities 12
8.1. European actions 12
8.1.1. Working Group TYPES 12
8.1.2. Concortium MoWGLI 12
8.2. Other cooperations 13
9. Dissemination 13
9.1. Animation of the scientific community 13
9.1.1. Editorial charges 13
9.1.2. Committees 13
9.1.3. Visits 14
9.1.4. Conferences 14
9.1.5. Other charges 15
9.2. Teaching 15
10. Bibliography 15

1. Team

The LogiCal project is a common project gathering researchers from INRIA-Futurs at LIX and Laboratoire de Recherche en Informatique of University Paris XI.

Scientific leader
Gilles Dowek [DR INRIA]

Scientific co-leader
Christine Paulin [Professor at University of Paris XI]

Vice leader
Benjamin Werner [CR INRIA]

Project assistants
Gina Grisvard [TR INRIA]
Stéphanie Meunier [TR INRIA]

INRIA staff
Bruno Barras [CR]
Hugo Herbelin [CR]
Claude Marché [CR, detached from Orsay University]

External researchers
Jean Duprat [Assistant professor at ENS-Lyon]
Gérard Huet [DR INRIA]

Paris XI staff
Judicaël Courant [Assistant professor at University of Paris XI]
Jean-Pierre Jouannaud [Professor at University of Paris XI]

CNRS staff
Jean-Christophe Filliâtre [CR]
Évelyne Contejean [CR]

Post-doctorates
Frédéric Blanqui [École polytechnique]
Alexandre Miquel [ATER at University Paris XI]

Ph.D. students
Jacek Chrząszcz [University of Warsaw]
Pierre Corbineau [École Normale Supérieure]
Benjamin Grégoire [MENRT, also member of Cristal project]
Olivier Hermant [DGA subsidee]
Florent Kirchner [ENS Cachan]
Pierre Letouzey [MENRT]
Julien Narboux [ENS Cachan]
Nicolas Oury [ENS Lyon]
Clément Renard [“Allocataire-moniteur” at University of Paris XI]
Julien Signoles [MENRT]
François-Régis Sinot [École Polytechnique]
Daria Walukiewicz-Chrząszcz [University of Warsaw]

2. Overall Objectives

The goal of the research carried out in the project is to build *proof assistants*. These systems can check that a proof is correct, they can help users build demonstrations interactively or automatically, store them in libraries, or export them towards other systems, ...

Using a computer to process mathematical demonstrations allows to establish that these demonstrations are faultless with a high level of certainty. In particular, one can get convinced that the argumentation justifying the soundness of a piece of hardware or software is correct. This is especially important in application domains where a malfunction may jeopardize human life, health or the environment and where large amounts of money are at stakes: computer-aided healthcare, transportations, telecommunications, e-commerce, ... Using a proof processor also allows building large scale demonstrations, for instance proofs using polynomials of hundreds of monomials. Finally, this participates in the quest of a new idea of exactness and rigour in mathematics: a point where nothing is understated, and where the reader can therefore be replaced by a program.

The main effort of our work is the development of the Coq system which nowadays has an important user community among industrials and academics. Nonetheless, we believe that the development of a system cannot be accomplished without a reflection about applications and specific usages in various domains (real geometry, proof of imperative or object-oriented programs, cryptographic protocols, ...), and a more fundamental reflection about the formalisation of mathematics (representation of proofs, integration of a programming language within the mathematical formalism, notion of bound variables, ...). This research is organised around two key notions: logical reasoning and computation. The LogiCal (*Logique et Calcul*) project is named after these two notions.

3. Scientific Foundations

3.1. Formalisation of mathematics

Key words: *mathematical language*, *programming language*, *predicate logic*, *set theory*, *constructive proofs*, *algorithm*.

A traditional language used to formalise mathematics is set theory, expressed in first order predicate logic. Unfortunately, this framework does not address exactly our needs. It has been elaborated in the early twentieth century to study mathematically the properties of mathematical reasoning. For this purpose, being able to formalise mathematics « in principle » is enough. Nowadays, the problem is not to formalise « in principle » but to formalise « in practice ».

This leads to study variants of set theory that offer a rich and compact language to express mathematical objects (for instance functions), and in particular languages with binding symbols such as the λ-calculus. Among the tools to study the properties of such languages are the notion of de Bruijn indices and that of explicit substitution.

Writing mathematical demonstrations with all the details also leads us to study formalism that include both reasoning and computation in order to avoid to write demonstrations for propositions that can be asserted by mere computation.

Considering building mathematical theories about programs and their properties leads to consider the effective aspect of mathematics and especially the relation between the notion of constructive proof and that of algorithm, so that from the proof of the existence of a function, one can produce a algorithm that computes it. More generally, we claim that there is no clear frontier between the mathematical language and programming languages and that a modern mathematical language should contain a programming language as a sub-language.

3.2. The Calculus of Inductive Constructions

Key words: *Calculus of Inductive Constructions*.

The **Coq** system is an implementation of a formalism called Calculus of Inductive Constructions.

In this formalism, one can develop proofs in a higher-order predicate logic and in this respect it is similar to the logics implemented in the proof systems HOL and PVS. These logics are well fitted to abstract reasoning and allow a natural representation of structured datatypes and predicates defined by fixpoints.

However, the Calculus of Inductive Constructions differs from Church's higher-order logic on several aspects we develop in the **Coq** system:

- Proofs are represented by λ-terms which are first-class objects. Checking the validity of a proof is made by type-checking the corresponding λ-term.
- The language allows the definition of functions by an algorithm (using a subset of a functional programming language). In some cases, reasoning about these functions can be done by computation rather than equational reasoning.
- The logic is a constructive one (it does not use the excluded-middle principle, nor proof by contradiction). As a consequence, of proof of existence of an object can be interpreted as an algorithm computing it.

3.3. Environments for interactive proof development

Key words: *tactic*.

The **Coq** system helps in developping mathematical theories by introducing definitions, hypotheses, by stating theorems and finally by giving a proof of these theorems. The user can build these proofs semi-automatically, thanks to decision procedures and *tactics*.

A tactic is a program, included in the system or user defined, that transforms a proposition to be proved into a set of new sufficient propositions. The implementation of tactics and decision procedures involves unification algorithms, theorem databases handling and computation (normalisation, rewriting).

The certification of proofs relies on typing and evaluation algorithms. In the case of decision procedures, it relies on so-called « reflection » techniques consisting in internalising the correctness proof of the procedure.

3.4. Deduction Modulo

Key words: *logic*, *calculus*, *deduction modulo*.

The Calculus of Inductive Constructions uses a conversion rule that identifies two propositions equivalent modulo a set of computation rules, so that any proof of the former is automatically considered as a proof of the latter, so the equivalence needs not be proven. This articulation between reasoning and computing can be expressed in a broader framework than the Calculus of Inductive Constructions. This framework is another alternative to first order predicate logic called *deduction modulo* in which a theory is defined by a set of axioms and computation rules.

Deduction modulo allows, among other things, the design of automated deduction algorithms that use this opposition between reasoning and computation. It also paves the way to a unified theory of cut elimination. An open problem, about deduction modulo, is the characterisation of the theories that can be expressed by the only means of computation rules.

3.5. Proofs and programs

Key words: *Curry-Howard isomorphism*, *realisability*, *intuitionistic logic*, *Calculus of Inductive Constructions*.

The main original aspect of the Calculus of Constructions is that proofs are objects, just like numbers, functions or sets are. Thus, an even number is represented by a pair formed with a number and a proof that it is even. Another original aspect is that any term denoting an object of a datatype can be reduced to a value of this datatype.

These properties combined together allows to think of a program specification as a relation relating the input value and the output value of that program. If $Q(x, y)$ is such a relation, a proof in the Calculus of Inductive Constructions of the totality of that relation (namely the proposition $\forall x \exists y \quad Q(x, y)$) is a function that associates to any object x an object y and a proof of $Q(x, y)$. From this totality proof, it is possible to

express in the calculus both a functional program f and its correctness proof. The function f applied to an input x will reduce to yield an output y.

But proofs are generally not efficient programs, and it is necessary, in practice, to discard parts of the proofs not usefull for computation: this step is called program extraction. Then, extracted programs can be translated to an ordinary programming language such as ML, and then compiled to machine code. In this approach, the program specification is a mathematical formula and the certified program is obtained from the proof of this formula.

This interpretation is especially well fitted to functional program certification. Now, it extends to other programming styles such as imperative programming and object-oriented programming through monadic translations. This is the key idea of systems Why and Krakatoa developed in the team.

4. Application Domains

Key words: *health-care*, *transportation*, *telecommunications*, *electronic commerce*, *networking*.

Proof assistants, and more generally formal method tools, are useful in domains where safety (that is the lack of unvoluntary errors) and security (that is the protection against malicious attacks) of computer systems are crucial. Especially applications where a functional flaw may expose human life, health or the environment to a danger and those where large amounts of money are exchanged. Thus, our application domains are medical informatics, transportation, telecommunication, electronic shopping, networks, ...

The **Coq** system is used to model cryptographic protocols for electronic shopping, security policies for Java Card smart cards, studies on compliance checks in telecommunication networks, proof of compiler of reactive languages, certification of computer algebra systems, ...

These developments are for the most part made by our partners, especially industrial partners: Trusted Logic, France Telecom, Gemplus, Schlumberger-Sema, ...

5. Software

5.1. Coq

Participants: Bruno Barras, Jean-Christophe Filliâtre, Benjamin Grégoire, Hugo Herbelin, Pierre Letouzey, Christine Paulin.

Le système **Coq** développé dans le projet est un système de traitement de démonstrations mathématiques qui permet de développer interactivement des spécifications et des démonstrations. The main original aspect of the **Coq** system is its formalism that includes:

- a primitive notion of mutual inductive definitions allowing high level specification either in a functional style by declaring concrete datatypes and defining functions by equations representing computations, or in a declarative style by specifying relations thanks to clauses;
- an interpretation of proofs as certified programs, implemented by the compilation of proofs as ML programs but also tools to associate a program to a specification and automatically generate proof obligations to assert its correctness;
- a primitive notion of co-inductive definitions allowing a direct representation of infinite reational data structures and build proofs upon such objects without resorting to the classical notion of bisimulation.

At the architectural level, the main features are:

- an interactive loop that allows to define mathematical and computational objects and to state lemmas,

- the interactive development of proofs thanks to a large and extendable set of tactics that decompose into elementary tactics (giving a precise control over the proof structure and thus over the underlying program) and decision or semi-decision procedures.
- a modular standard library and retrieving tools,
- a mechanism to perform partial or total evaluation of programs written within the language of **Coq**,
- the possibility to develop evolved tactics written in the implementation language of **Coq** (namely Objective Caml), and that can be dynamically loaded and used from the toplevel,
- the isolation of the critical code preforming the proof checking in a kernel small enough to reach higher levels of reliability of the whole system (with the current goal of achieving the self-validation), and the production of an abstract interface of that kernel granting that theories can only be built using the features of the kernel.

Among the most significative achievements realised using **Coq**, it worths mentioning:

- the model of authentication protocol CSET used in electronic shopping and the proof of properties of this protocol,
- the correctness proof of a compiler of the reactive language Lustre, used in the industrial setting of Scade,
- a proof of the critical kernel of the **Coq** environment,
- several models of the properties of the π-calculus,
- the development of libraries about algebra, analysis and geometry,
- a certified version of Buchberger's algorithm used in computer algebra,
- the proof of FTA theorem,
- the proof of Taylor's approximation theorem.

The **Coq** system is available from URL http://coq.inria.fr/. Written in Objective Caml and Camlp4, it is ported to mosts Unix architectures, but also to Windows and MacOS.

Coq is used in hundreds of sites. We have demanding users in industry (France Telecom R & D, Dassault-Aviation, Trusted Logic, Gemplus, Schlumberger-Sema, ...) in the academic world in Europe (Scotland, Netherlands, Spain, Italy, Portugal) and in France (Bordeaux, Lyon, Marseille, Nancy, Nantes, Nice, Paris, Strasbourg).

An electronic mailing list (mailto:coq-club@pauillac.inria.fr) fosters exchange between persons interested by the system.

5.2. Why

Participant: Jean-Christophe Filliâtre.

The **Why** tool produces verification conditions from annotated programs given as input. It differs from other systems in that it outputs conditions for several existing provers (including Coq but also PVS, HOL-light, Mizar, Simplify and haRVey). Why is aimed at being used as a back-end for other tools dealing with real programming languages. It is already used by the Krakatoa tool for the verification of Java programs [41] and by a forthcoming tool for the verification of C programs.

It is available at URL http://why.lri.fr/.

5.3. Krakatoa

Participants: Claude Marché, Christine Paulin, Xavier Urbain.

Translation tool **Krakatoa** accepts JAVA/JAVACARD programs (hence imperative programs) annotated in JML and produces another program with the same semantics but to be used as input by the proof obligation generator **Why**.

Krakatoa is available at URL http://krakatoa.lri.fr

5.4. Fatalis

Participant: Jean-Pierre Jouannaud.

Coq can be used as a logical framework, to develop and verify proofs in a logical system described in **Coq**. Based on this paradigm, Fatalis is a system allowing the user to specify timed transition systems, as well as the safety properties that they should satisfy. The specification language is based on a fragment of linear logic, and every run of the system identifies itself with a linear logic proof. These proofs can therefore be verified by simply type checking them via the implementation of this linear logic fragment in **Coq**. The main advantage of this approach is the possibility of considering specifications of dynamic systems, in which agents —of some class— can be dynamically created (via a linear logic implication without premiss and a —possibly existentially quantified— conclusion) or destroyed (via a linear logic implication without conclusion and with a —possibly existentially quantified— premiss).

Another advantage is the flexibility of the formalism, which allows for a smooth transition from finite systems, for which safety properties are decidable, and infinite systems, for which proving safety properties requires (possibly non-terminating) proof search. This is not implemented yet. Future evolutions of the system will adress the questions of (i) representing "similar" runs of the systems by using formulae constraining local clocks with Presburger arithmetic constraints, and (ii) applying Fatalis to verifying security properties of cryptographic protocols.

6. New Results

6.1. Development of theories and tactics

6.1.1. Extraction of modules

Participants: Pierre Letouzey, Nicolas Oury.

Pierre Letouzey continued his studies of the extraction mechanism in the Coq proof assistant, aimed at building Ocaml programs from Coq proofs. On the practical level, the Coq extraction has been adapted to support the new Coq module system. Extracting a Coq module, functor or signature now gives an Ocaml module, functor or signature.

Nicolas Oury worked on the substitution of a simple data structure with an efficient one at program extraction time in the COQ assistant. This work resulted in a publication at TLCA 2003 [29].

6.1.2. Extraction of arbitrary precision numbers

Participant: Pierre Letouzey.

Pierre Letouzey has cooperated with computer scientists of Nijmegen and München, in order to transform constructive real analysis developed in Coq into a library of real numbers with arbitrary precision via extraction.

6.1.3. Timed automata

Participant: Christine Paulin.

C. Paulin participated to the AVERROES project for the development of an environment for the specification proof and testing of algorithms represented as parameterised times automata. She is working on a functional representation of probabilistic transformations of the variables during action transitions.

6.1.4. Proof languages

Participant: Florent Kirchner.

Florent Kirchner started a Ph.D this year on the semantics of proof languages, in collaboration with INRIA-Futurs and the National Institute of Aerospace. The semantics of a subset of the proof languages, called tacticals, was the initial approach that was chosen, and the results were exposed at the STRATA workshop in September 2003 [26]. Considering the fact that proof languages are similar to imperative languages, he now intends to formalise the semantics of those languages.

6.1.5. Tactic implementing a first order sequent calculus

Participant: Pierre Corbineau.

Pierre Corbineau proved the contraction- and cut-elimination properties for a contraction-free intuitionnistic first-order sequent calculus including inductive definitions similar to those in Coq. Those results are available as a technical report (see [34]). This report was also made into a deliverable for the Averroes RNTL project.

This result lead to the implementation of a Coq tactic for first-order reasoning.

6.1.6. Formalisation of euclidean geometry

Participants: Jean Duprat, Julien Narboux.

Julien Narboux has implemented in Coq a decision method for euclidean geometry using the Ltac language. This method, first introduced by Chou, Gao and Zhang, is based on the elimination of some geometric quantities and produces "readable" proofs.

A collaboration with Frédérique Guilhot (INRIA-Sophia, Lemme) has started. It consists in combining the decision method with Frédérique Guilhot's formalization of highschool geometry in order to provide a pedagogical tool.

Julien Narboux has initiated a discussion with Jean-Duprat (ENS-Lyon), Frédérique Guilhot (INRIA-Sophia, Lemme) and Loïc Pottier (INRIA-Sophia, Lemme) toward the definition of a common formal langage for stating geometry theorems.

Jean Duprat has continued his work about a well-fitted formalization of plane geometry for a proof assistant like Coq. The main idea is that Euclide's elements correspond more to a constructive vision than Hilbert's axiomatization. Following that idea, the ruler and the compass are the constructors of lines and circles, and their intersections give the constructive points of the figure.

6.1.7. Unification modulo AC

Participant: Evelyne Contejean.

E. Contejean is currently modelling and certifying in Coq the AC-matching algorithm of CiME.

6.1.8. Proof of Java programs

Participants: Claude Marché, Christine Paulin, Xavier Urbain.

Claude Marché, Xavier Urbain and Christine Paulin have investigated the theoretical background of Krakatoa, that is the Coq modeling of Java classes and memory heap. Christine Paulin has been working on the underlying memory model and the corresponding Coq library. This model uses a separate array for each object field, internalising in the model, that different fields correspond to different memory cells. This has been presented in a paper, which will appear soon [11].

In the context of the Verificard project, KRAKATOA has been experimented on a real applet case study proposed by Schlumberger, and the result of experiments have been published as a Verificard deliverable [33].

6.1.9. formalisation of sets and maps datatypes

Participants: Jean-Christophe Filliâtre, Pierre Letouzey.

Jean-Christophe Filliâtre and Pierre Letouzey formalized several applicative data structures for finite sets using the Coq proof assistant [23]. These implementations use functors from the ML module system and

the verification follows closely this scheme using the newly Coq module system. One of the verified implementation is the actual code for sets and maps from the Objective Caml standard library. The process of verification exhibited two small errors in the balancing scheme, which have been fixed and then verified. This work illustrates the use and benefits of modules and functors in a logical framework.

6.1.10. Efficient primality test with Pocklington's criterion

Participant: Benjamin Grégoire.

Benjamin Grégoire enhanced Martijn Oostdijk's library about Pocklington's criterion to prove the primality of large number by reflection. The outcome is a reduction of proof size and a higher efficiency to check primality proof. The size of number that is possible to prove has been extend by a factor of 3.

6.1.11. Four-color Theorem

Participants: Georges Gonthier [Project Moscova], Benjamin Werner.

G. Gonthier and B. Werner made further progress in the Coq formalization of the proof of the four-color theorem.

6.1.12. Safety of protocols in aeronautic

Participants: Gilles Dowek, César Muñoz [NIA], Victor Carreño [NASA].

Together with César Muñoz, and Victor Carreño, Gilles Dowek has carried out an exaustive state exploration of the *Small Aircraft Transportation System* concept of operations.

An earlier paper on the *Airborne Information for Lateral Spacing* concept has been published as a Journal paper.

6.2. Development of systems

6.2.1. Release of Coq V7.4

Participants: Bruno Barras, Jacek Chrząszcz, Jean-Christophe Filliâtre, Hugo Herbelin, Pierre Letouzey, Benjamin Monate, Christine Paulin, Clément Renard.

Bruno Barras, Jean-Christophe Filliâtre, Hugo Herbelin, Pierre Letouzey, Christine Paulin, Clément Renard participated in the release of a new version of Coq (V7.4) on february 2003. This version is still experimental and provides several novelties, including a new system of parametric modules and interfaces, and support for the Why (http://why.lri.fr/) and Krakatoa (http://krakatoa.lri.fr) systems.

Pierre Letouzey improved the extraction tool in order to support extraction of modules, and to ensure that extracted Ocaml code will always type-check correctly. Moreover the types of these extracted terms are predictable and are placed in a signature file.

Jacek Chrząszcz has developped a new version of Coq with modules and functors "à la OCAML". This version of Coq is now used by all developpers, and is the basis of all recent developments made in the project.

Benjamin Monate wrote an interface to Coq called CoqIde, based on GTK2 library. It is close in style from the Proof General Emacs interface. It is faster and its integration with Coq makes interactive developments more friendly.

6.2.2. Release of Coq V8.0

Participants: Bruno Barras, Hugo Herbelin.

Hugo Herbelin worked with Bruno Barras on the Coq version 8 project. He designed and implemented a new mechanism for easy-writing of mathematical symbolic notations. He restructured and made uniform the standard library of Coq as a better foundation for the next years developments in Coq.

6.2.3. Tactics dealing with existential variables

Participant: Clément Renard.

Clément Renard also took part to the development and maintenance of the Coq proof asistant. He added some new tactics dealing with existential variables and corrected some minor bugs in the current unification procedure.

6.2.4. Compilation of proofs towards a virtual machine

Participant: Benjamin Grégoire.

Benjamin Grégoire continued the development of his prototype of Coq where proofs and programs are compiled rather than interpreted. He extended it to cope with modules.

6.2.5. Incorporating rewriting to Coq

Participants: Frédéric Blanqui, Jean-Pierre Jouannaud, Daria Walukiewicz.

Jean-Pierre Jouannaud, Frédéric Blanqui and Daria Walukiewicz have continued their work on incorporating rewriting facilities to Coq via the conversion rule. Blanqui's most recent results cover the entire calculus of inductive constructions (without universes) by viewing all strong elimination rules of CIC as particular cases of an extended version of the higher-order schema of Jouannaud and Okada [4]. A prototype implementation of rewriting in Coq is on its way. In her thesis, Daria Walukiewicz has described a generalization to the calculus of constructions of the first version of the higher-order recursive path ordering of Jouannaud and Rubio. A further generalization along the lines of the new version of the higher-order recursive path ordering is on its way, and should provide with an extremely powerful tool for designing calculi like the calculus of constructions incorporating additionnal rewrite rules like CIC.

6.2.6. Why

Participant: Jean-Christophe Filliâtre.

J.-C. Filliâtre has worked on the development of the Why verification tool [35][40].

6.2.7. Release of Krakatoa

Participants: Claude Marché, Christine Paulin, Xavier Urbain.

A preliminary version of this tool is now publicly available on the web (http://krakatoa.lri.fr),

6.2.8. CiME

Participants: Evelyne Contejean, Claude Marché, Xavier Urbain.

E. Contejean, C. Marché and X. Urbain continued the development of the CiME rewrite tool (http://cime.lri.fr).

E. Contejean is in charge of the matching, the unification and the completion parts (standard but also with associative-commutative symbols).

6.3. Studies of formalisms

6.3.1. Induction over real numbers

Participants: Gilles Dowek, Assia Mahboubi.

Gilles Dowek has started to investigate a positive form of the completeness property of real numbers, called real induction : if a property is closed, holds for a real number c and and whenever it holds for a real number x then it holds on an interval $x, x + \varepsilon$, for some $\varepsilon > 0$, then its holds for all real numbers bigger than c. A preliminary work has been presented in [38]. This formulation has revealed to be classicaly equivalent to a dual principle called "open induction" due to Thierry Coquand and then studied by Wim Veldman.

In her Master thesis [43], Assia Mahboubi has studied the constructive content of the open induction principle. She has provided an algorithm to extract a witness from an existence proof using this principle. The existence of this algorithm highly depends on the type of definition of the notion of open set used in this principle.

6.3.2. Decision methods

Participants: Gilles Dowek, Ying Jiang [Chinese Academy of Sciences].

In cooperation with Ying Jiang, of the Chinese Academy of Sciences in Beijing, Gilles Dowek has provided a decision algorithm for the positive fragment of intuitionistic predicate logic. A corolary of this result is the decidability of the positive fragment of higher-order intuitionistic predicate logic and of inhabitation of positive types in system F. This work has been published in [22].

6.3.3. Semantics of Type Theory

Participants: Alexandre Miquel, Benjamin Werner.

A. Miquel and B. Werner clarified some aspects of the simple (proof-irrelevant) models of impredicative type theories [27].

6.3.4. Cut elimination in set theory

Participants: Jérémie Cabessa, Gilles Dowek, Alexandre Miquel.

Together with Alexandre Miquel, Gilles Dowek has started a work on an expression of set theory in deduction modulo.

In his masters thesis Jéremie Cabessa has proposed an implementation of this formalism.

6.3.5. Cut elimination in sequent calculus modulo

Participant: Olivier Hermant.

Olivier Hermant has dealt with cut elimination in sequent calculus modulo. He worked in classical logic over a wide range of rewrite systems, including quantifier free or HOL rewrite system. Now he switched to the intuitionnistic case and tries to understand what are the links between cut elimination in classical and intuitionnistic case. He is also trying to understand the links between cut elimination and normalization.

6.3.6. Asymetric deduction modulo

Participant: Gilles Dowek.

In [20], Gilles Dowek has proposed a variant of deduction modulo, called *Asymetric deduction modulo* and proved that the confluence of a rewrite system is equivalent to the cut elimination property for the theory formed by this rewrite system in assymetric deduction modulo.

Two earlier papers on deduction modulo have been published as Journal papers [7] and [8].

6.3.7. Automated termination proof

Participants: Evelyne Contejean, Claude Marché, Xavier Urbain.

E. Contejean, C. Marché and X. Urbain continued some work on the theme of automatic proof of termination. Some of this work have been presented at the Workshop on Termination [28][18], and a full paper with X. Urbain is currently submitted [45].

A full paper on modular termination proofs by X. Urbain is to appear [12]. A full paper on AC termination proofs by X. Urbain and C. Marché is currently submitted [45], as well as a full paper on polynomial interpretations by E. Contejean, C. Marché, A. P. Thomás and X. Urbain [37].

6.3.8. Name-free λ-calculus with explicit substitutions

Participants: François-Régis Sinot, Maribel Fernández [King's Collega, London], Ian Mackie [King's College, London].

François-Régis Sinot, Maribel Fernández and Ian Mackie developped a name free λ-calculus with explicit substitutions, based on a generalized notion of director strings. Terms are annotated with information that indicate how substitutions should be propagated. A calculus where arbitrary β-reduction steps can be simulated was designed, and then simplified to model the evaluation of functional programs (reduction to weak head normal form). This formalism was also shown to be adequate to define the closed reduction strategy, which is a weak strategy originally introduced by Fernández and Mackie, which, in contrast with standard weak strategies, allows certain reductions to take place inside λ-abstractions thus offering more sharing. François-Régis Sinot implemented a series of abstract machines based on this and other strategies, whose experimental

results confirmed that, for large combinator based terms, these weak evaluation strategies out-perform standard evaluators. Moreover, abstract machines for strong reduction were derived, which inherit the efficiency of the weak evaluators.

Most of this work was described in the paper "Efficient Reductions with Director Strings".

6.3.9. Equivalence of conversion in Church and Curry styles in the Calculus of Inductive Constructions

Participants: Bruno Barras, Benjamin Grégoire.

Benjamin Grégoire proved in collaboration with Bruno Barras that conversion on well-typed terms of CIC is not affected if some annotations (such as type annotation of functions, inductive type of constructors) are removed. This leads to an improved conversion test.

6.3.10. Computational content of "ex falso quolibet"

Participants: Hugo Herbelin, Zena Ariola [University of Oregon].

Hugo Herbelin worked with Zena Ariola on the computational content of the "ex falso quodlibet" (= $\bot \to A$) intuitionistic axiom. They showed that it should be interpreted as a constant denoting the "toplevel" continuation. Their work lead to a better-behaved presentation of Felleisen's C control operator. It has been published in [15].

6.3.11. Higher-order unification and type inference

Participants: Bruno Barras, Clément Renard.

Clément Renard worked on higher order unification in the Calculus of Constructions. He developed a prototype implementation of an unification algorithm wich give high priority to invertible rules. This implementation is based on a calculus with typed metavariables with a very limited kind of explicit substitutions. The unification rules have been proved correct and complete on a subclass of problems. He then developed a typechecker based on this unification algorithm and then obtained ML-style type inference (not full type inference since it is undecidable in the Calculus of Constructions).

6.3.12. Epsilon-symbol and Foundations of Linking

Participants: Benjamin Werner, Martin Abadi [UCSC], Georges Gonthier [Project Moscova].

Together with Martin Abadi (UCSC) and G. Gonthier (INRIA-Rocquencourt), B. Werner investigated a Curry-Howard interpretation for Hilbert ε-symbol. They showed that the resulting typed calculus yields a form of *dynamic linking*; this gives a new theoretical foundation for this feature. The first results are published in [14].

6.3.13. Defunctorisation

Participant: Julien Signoles.

Julien Signoles ended its study on a static calculus of application of parameterized modules (called *defunctorization*). This study led on an article [30] and an implementation (*ocamldefun* [44]).

6.3.14. ML with refinement

Participant: Julien Signoles.

Julien Signoles has begun to study an extension of ML with refinement. The aim of this PhD work is to add formal specifications in ML programs in order to prove them.

6.3.15. Extentionality in the Calculus of Inductive Constructions

Participant: Nicolas Oury.

Nicolas Oury has studied the conservativity of extentional Calculus of Inductive Constructions over intentional CIC.

7. Contracts and Grants with Industry

7.1. Averroes

We are part of project AVERROES which started in october 2002. Labelised by National Network of Software Technologies (Réseau National des Technologies Logicielles, RNTL), it follows project Calife and have the same partners: CRIL, France Telecom R & D, INRIA, LaBRI (Bordeaux), LORIA, LRI (Orsay) and LSV (ENS Cachan). The goal of the project is to develop formal methods able to reliably check properties raising in industrial problems. It extends project Calife in not limiting to functional properties. It also studies stochastic properties and resources consumption of protocols.

7.2. France Télécom

We have a collaboration with France Télécom about Rewriting and Modules for Coq. It is a compagnon contract to CALIFE, an RNRT project preceding AVERROES, with the same partners.

7.3. Verificard

European Project Verificard studies security and safety aspects for the new generation of smart cards.

It is composed of INRIA, University of Nijmegen, University of Munich, Unversity of Hagen, Swedish Institute of Computer Science, in the academic world, and the Gemplus and Schlumberger-Sema companies.

7.4. Geccoo

Geccoo is an *ACI* about generating certified code for object-oriented applications. It began in july 2003 and should end in july 2006. The partners are TFC team (LIFC), project CASSIS (LORIA) project Everest and VASCO team (LSR, Grenoble). It is described at URL http://geccoo.lri.fr.

7.5. Modulogic

ModuLogic is an ACI about security. Its goal is to build a laboratory for the construction of certified software. Our partners are: group FOC (LIP6, CEDRIC, INRIA-Rocquencourt), project PROTHEO (LORIA) and action MIRO. It is described at URL http://pauillac.inria.fr/modulogic/.

8. Other Grants and Activities

8.1. European actions

8.1.1. Working Group TYPES

Working Group « TYPES » is about computer aided development of proofs and programs.

It is composed of teams from Helsinki, Chambéry, Paris, Lyon, Rocquencourt, Sophia Antipolis, Orsay, Darmstadt, Freiburg, München, Birmingham, Cambridge, Durham, Edinburgh, Manchester, London, Sheffield, Padova, Torino, Udine, Nijmegen, Utrecht, Bialystok, Warsaw, Minho, Chalmers, and also from Prover Technology, France Telecom, Nokia, Dassault-Aviation, Trusted Logic and Xerox companies.

8.1.2. Concortium MoWGLI

Concortium « MoWGLI » (Mathematics on the Web, Get it by Logic and Interface) is about developping an hypertext library of mathematical theories, organised around a notation for document and mathematical formulas in XML format (OnDoc and MathML), the design of search analysis tools and the design of interfaces capable of handling theories.

It is composed of teams from Berlin, Bologne, Nijmegen, Saarbrücken, Sophia-Antipolis, and Trusted Logic company.

8.2. Other cooperations

Jean-Pierre Jouannaud has strong cooperations with

- Daria Walukiewicz et Jacek Chrzaszcz at University of Warsaw (both have made their PhD under his supervision), on the topic of proof assistants,
- Albert Rubio (Technological University of Cataluña, Barcelona) and Femke Van Raamsdonk (Free University, Amsterdam), on the topic of higher-order rewriting and orderings
- Mitsuhiro Okada (Keïo University, Tokyo), on the topic of specification and verification of real-time systems,
- José Meseguer and Mark-Olliver Stehr (University of Illinois at Urbana-Champaign), on the topic of Maude (fast proyotyping type-theoretic calculi), through a contract between CNRS and Urbana-Champaign.

9. Dissemination

9.1. Animation of the scientific community

9.1.1. Editorial charges

C. Paulin participated to the program committee of the conferences TPHOLs'03 (16th International Conference on Theorem Proving in Higher Order Logics TPHOLs'03) TLDI'03 (Workshop on Types in Language Design and Implementation TLDI'03) RTA'03 (14th International Conference on Rewriting Techniques and Applications RTA'03)

C. Paulin is a proposed member of the new IFIP working group (WG 2.11) on Program Generation http://www.cs.rice.edu/~taha/wg2.11/.

Jean-Christophe Filliâtre is a member of the TPHOLs program committee (2003 and 2004).

Hugo Herbelin was a member of the ICFP 2003 and JFLA 2004 program committees.

Jean-Christophe Filliâtre organized the 14th *Journées Francophones des Langages Applicatifs* (January 27-28 2003, Chamrousse, France).

Hugo Herbelin co-organised with Zena Ariola and John Mitchell the 2nd Eugene, Oregon INRIA-sponsored summerschool. This year topic was "Foundations of security".

9.1.2. Committees

Jean-Pierre Jouannaud has been member of the committee of the Special ETACS Award. This award id given every year to a computer scientist for his works.

Jean-Pierre Jouannaud est is member of the *commissions de spécialité* of ENS-Cachan and of University d'Aix-Marseille, and a meber of the human resources committee at École polytechnique.

Benjamin Werner is member of the evaluation board of INRIA. He is also member of the *comissions de spécialistes* of Ecole Normale Supérieure and of the University of Paris 12-Créteil.

Jean-Pierre Jouannaud was president of the PhD committees of Véronique Cortier (2002) and Mathieu Turuani (2003), both about security of cryptographic protcols.

Hugo Herbelin was a member of Sylvain Baro's Phd thesis committee.

Gilles Dowek has been a member of the thesis committee of Dimitri Hendriks (Utrecht) and of the habilitation committee of Loic Pottier (Nice).

C. Paulin was a referee for the thesis of : Simão Melo de Sousa and Nicolas Magaud. She also participated to the thesis committee of Guillaume Dufay.

Christine Paulin has been a referee of the PhD of Laurent Chicli (Nice, defended 26th november 2003). Benjamin Werner was also member of Chicli's PhD thesis committee.

Gilles Dowek is a CADE "trustee".

9.1.3. Visits

In 2003, François-Régis Sinot was mainly at King's College and worked with Maribel Fernández and Ian Mackie.

Florent Kirchner spent two months this summer visiting the *National Intitute of Aerospace*.

Xavier Urbain spent five months at Nijmegen University (The Netherlands) so as to study the approach chosen by the Loop team, and to compare the LOOP tool and KRAKATOA on a case study provided by Schlumberger.

9.1.4. Conferences

Gilles Dowek has given a course at the Summer School *Proof technology and computation*, Marktoberdorf, Germany (July 29-August 9). Olivier Hermant and Pierre Letouzey attended this summerschool.

Gilles Dowek, Jean-Christophe Filliâtre, Hugo Herbelin, Olivier Hermant, Pierre Letouzey, Julien Narboux and Benjamin Werner have participated to the TYPES meeting in Torino, Italy (April 30-May 4).

Gilles Dowek and François-Régis Sinot have participated to the *14th International Conference on Rewriting Techniques and Applications* in Valencia, Spain, (June 9-11). Gilles Dowek has given a presentation.

Gilles Dowek has participated to seminar *Verification and Constructive Algebra* in Dagstuhl, Germany (January, 5-10) where he has given a presentation.

Gilles Dowek has participated to the workshop *Mathematics, Logic and Computation, In honour of N.G. de Bruijn's 85th anniversary*, Eindhoven, Netherlands (July, 4th-5th) where he has given a presentation. C. Paulin gave a course on inductive definitions at the Summerschool on the "Foundations of Security" at the University of Oregon.

Hugo Herbelin attended ICALP 2003 Colloquium in Eindhoven, the Netherlands.

Claude Marché and Evelyne Contejean and Xavier Urbain participed to the first Federated Confernec on Rewriting, Deduction and Programming (RDP'03), in Valencia, Spain, June 8-14, 2003. Claude Marché presented both the CiME rewrite tool [18] and the TALP tool for termination of logic programs [28] at the satellite workshop on termination.

Evelyne Contejean was an entrant in the CASC'19 system competition, with the CIME tool.

C. Marché and X. Urbain participed to the Verificard annual meeting on 20-22 january 2003, in Munich, Germany, where he presented a demo of the Krakatoa tool.

C. Marché participated to the joint VeriSafe workshop, in Rennes, France, on september 10-12, 2003.

C. Marché participated to the final review of the Verificard project, in Bruxelles, Belgium, on november 26, 2003.

Hugo Herbelin attended the MoWGLI meetings in Bertinoro, Italy and Sophia-Antipolis, France.

Julien Narboux has attended the international summer school : "Foundations of Security 2003".

Olivier Hermant participated to the Second Days of Logic and Computability in St-Petersburg conference (Russia, Aug. 24 - 26).

Florent Kirchner attended TPHOL 2003 at Roma, Italy (September 9-12 2003).

Benjamin Werner gave a seminar talk at Kyoto Sangyo University.

Gilles Dowek has participated to the colloque *Logique, Informatique, Mathématique et Physique* in Paris, France (April 25-26) where he has given a presentation.

Gilles Dowek has participated to the workshop *Algorithmique et programmation* at Luminy, France (May 5-9) where he has given a presentation.

Pierre Letouzey attended the 14th *JournÃ©es Francophones des Langages Applicatifs* (January 27-28 2003, Chamrousse, France).

François-Régis Sinot attended to "ECOLE JEUNES CHERCHEURS EN PROGRAMMATION" 2003 (Aussois).

Yves Bertot and Jean Duprat gave a course about "Functional Programming and Proof" to the second year students of the Magistere Informatique et Modelisation of the ENS of Lyon.

Jean Duprat has organized a day about "geometry and proof assistant" in the ENS of Lyon on june 21th, 2003. Julien Narboux gave a talk.

9.1.5. Other charges

Jean-Pierre Jouannaud is the leader of the LIX laboratory. He is president of AFIT, and member of "council of ETACS".

Gilles Dowek is a member of the steering committee of TYPES group.

C. Paulin is the Orsay site leader for the europeean Working Group TYPES.

Hugo Herbelin is the correspondent of the LogiCal team within the european MoWGLI project.

Bruno Barras has been a consultant in formal methods at Trusted Logic.

Gilles Dowek has been a consultant for the *National Intitute of Aerospace* supporting the *NASA Langley's Formal Methods Group* (September 1-28).

C. Paulin is the coordinator of the GECCOO project (ACI Security) http://geccoo.lri.fr.

C. Paulin participate to the Comité de Pilotage of Reseau Thématique Pluridisciplinaire - SECC Systèmes Embarqués Complexes ou Contraints

9.2. Teaching

Jean-Pierre Jouannaud and Maribel Fernández (King's College London) supervised the PhD thesis of François-Régis Sinot.

Bruno Barras supervised the PhD thesis of Clément Renard.

Benjamin Werner is co-director of the PhD of Benjamin Grégoire (Paris 7 defended 19th december 2003)

Hugo Herbelin is the supervisor of Julien Narboux PhD thesis.

Hugo Herbelin supervised Aaron Bohannon master-degree two-monthes training period.

Gilles Dowek is the advisor of the doctoral work of Olivier Hermant and Florent Kirchner. He has been the advisor of the master thesis (stage de DEA) of Assia Mahboubi, Florent Kirchner and Jeremie Cabessa.

Jean-Christophe Filliâtre is supervising the Ph.D. of Julien Signoles (addition of refinement to the ML language).

Claude Marché is supervisor of the PhD thesis of Pierre Corbineau.

Christine Paulin is the thesis advisor of Pierre Letouzey and Nicolas Oury. She also is co-advisor of June Andronick's thesis (CIFRE) at Axalto.

Jean-Pierre Jouannaud teaches at University Paris Sud (*licence*), at École polytechnique (*majeure 2*), and at the master degree *programmation* (common cursus and option "proof").

Christine Paulin is responsible for the 3rd and 4th year of undergraduate studies in Computer Science at University Paris Sud. She was also the coordinator of the master project.

Bruno Barras, Hugo Herbelin and Christine Paulin taught a master degree course "Calculus of Inductive Constructions".

Claude Marché has been teaching at the *DEA Programmation, Sémantique, Preuves,et Langages*, optional course *Terminaison*.

Jean-Christophe Filliâtre taught computer science at École Polytechnique (January-June 2003) and Université Paris Sud (September-December 2003).

Clément Renard taught functional programming to second year students at Orsay university and compilation to fourth year students.

Benjamin Werner teaches Coq and theory of programming for undergraduates at ENSTA (Paris). He teaches a graduate course on logic and Type Theory at the DEA "Sémantique, Preuves et Programmes".

10. Bibliography

Books and Monographs

[1] G. CHARDIN, G. DOWEK, M. LACHIÈZE-REY, H. THIS. E. KLEIN, editor, *Quand la science a dit c'est bizarre !*. Le Pommier, 2003.

[2] *Extended Abstracts of the 6th International Workshop on Termination, WST'03.* A. RUBIO, editor, June, 2003, Technical Report DSIC II/15/03, Universidad Politécnica de Valencia, Spain.

Articles in referred journals and book chapters

[3] F. BLANQUI. *Definitions by rewriting in the Calculus of Constructions.* in « Mathematical Structures in Computer Science », 2003.

[4] F. BLANQUI, J.-P. JOUANNAUD, M. OKADA. *Inductive Data Type Systems.* in « Theoretical Computer Science », number 272(1-2), 2003, pages 41–68.

[5] G. DOWEK. *Au cœur d'une calculatrice.* D. WILGENBUS, B. SALVIAT, M. JULIA, editors, in « Graines de Sciences 5 », Le Pommier, 2003.

[6] G. DOWEK. *La Théorie des types et les systèmes informatiques de traitement de démonstrations mathématiques.* in « Mathématiques et Sciences Humaines », 2003.

[7] G. DOWEK, T. HARDIN, C. KIRCHNER. *Theorem proving modulo.* in « Journal of Automated Reasoning », volume 31, 2003, pages 33–72.

[8] G. DOWEK, B. WERNER. *Proof Normalization modulo.* in « The Journal of Symbolic Logic », 2003.

[9] J. DUPRAT. *Using ProofAssistant for Plane Geometry.* volume 12, 2003.

[10] O. HERMANT. *A Model-Based Cut Elimination Proof.* in « Annals of Pure And Applied Logic », 2003, special issue.

[11] C. MARCHÉ, C. PAULIN-MOHRING, X. URBAIN. *The* KRAKATOA *Tool for Certification of* JAVA/JAVACARD *Programs annotated in* JML. in « Journal of Logic and Algebraic Programming », 2003, http://krakatoa.lri.fr, To appear.

[12] X. URBAIN. *Modular and Incremental Automated Termination Proofs.* in « Journal of Automated Reasoning », 2003, To appear, 43 pages.

[13] D. WALUKIEWICZ-CHRZASZCZ. *Termination of rewriting in the Calculus of Constructions.* in « J. Functional Programming », 2003, to be published in 2004.

Publications in Conferences and Workshops

[14] M. ABADI, G. GONTHIER, B. WERNER. *Choice in Dynamic Linking.* in « FOSSACS 2004 », 2003.

[15] Z. ARIOLA, H. HERBELIN. *Minimal classical logic and control.* in « Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, Eindhoven, The Netherlands, June 30 - July 4, 2003. Proceedings », series LNCS, volume 2719, Springer, J. C. M. BAETEN, J. K. LENSTRA, J. PARROW, G. J. WOEGINGER, editors, pages 871–885, 2003.

[16] F. BLANQUI. *Inductive types in the Calculus of Algebraic Constructions.* in « Proceedings of the 6th International Conference on Typed Lambda Calculi and Applications », series LNCS, number 2701, 2003.

[17] F. BLANQUI. *Rewriting modulo in Deduction modulo.* in « Proceedings of the 14th International Conference on Rewriting Techniques and Applications », series LNCS, number 2706, 2003.

[18] E. CONTEJEAN, C. MARCHÉ, B. MONATE, X. URBAIN. *Proving Termination of Rewriting with CiME.* in « Extended Abstracts of the 6th International Workshop on Termination, WST'03 », A. RUBIO, editor, June, 2003, http://cime.lri.fr, Technical Report DSIC II/15/03, Universidad Politécnica de Valencia, Spain.

[19] V. CREMET, K. HASEBE, J.-P. JOUANNAUD, A. KREMER, M. OKADA. *FATALIS : Real time processes as linear logic specifications.* in « International Workshop on Automated Verification of Infinite-State Systems, Varsaw », 2003.

[20] G. DOWEK. *Confluence as a cut elimination property.* in « Rewriting Technique and Applications », series LNCS, volume 2706, Springer-Verlag, R. NIEUWENHUIS, editor, pages 2–13, 2003.

[21] G. DOWEK. *La notion de modèle suppose-t-elle une conception réaliste de la vérité mathématique ?.* in « Logique, Mathématiques, Informatique et Philosophie », 2003, to appear.

[22] G. DOWEK, Y. JIANG. *Eigenvariables, bracketing and the decidability of positive minimal intuitionistic logic.* in « Electronic Notes in Theoretical Computer Science », volume 85, Elsevier, H. GEUVERS, F. KAMAREDDINE, editors, 2003.

[23] J.-C. FILLIÂTRE, P. LETOUZEY. *Functors for Proofs and Programs.* in « Proceedings of The European Symposium on Programming », Barcelona, Spain, March 29-April 2, 2003, http://www.lri.fr/~filliatr/ftp/publis/fpp.ps.gz, To appear.

[24] K. HASEBE, J.-P. JOUANNAUD, A. KREMER, M. OKADA, R. ZUMKELLER. *Formal Verification of Dynamic Real-Time State-Transition Systems Using Linear Logic.* in « Proceedings of the Software Science Conference,Nagoya », 2003.

[25] O. HERMANT. *A Model-Based Cut Elimination Proof.* in « Second St. Petersburg Days Of Logic And Computability, Saint-Petersburg », 2003.

[26] F. KIRCHNER. *Coq Tacticals and PVS Strategies: A Small-Step Semantics.* in « Design and Application of Strategies/Tactics in Higher Order Logics », NASA publisher, M. A. ET AL., editor, pages 69–83, September, 2003.

[27] A. MIQUEL, B. WERNER. *On the not-so-simple proof-irrelevant model of CC.* in « Proceedings of TYPES'02 », series LNCS, Springer-Verlag, F. WIEDIJK, H. GEUVER, editors, 2003.

[28] E. OHLEBUSCH, C. CLAVES, C. MARCHÉ. *The TALP Tool for Termination Analysis of Logic Programs.* in « Extended Abstracts of the 6th International Workshop on Termination, WST'03 », A. RUBIO, editor, June, 2003, http://bibiserv.techfak.uni-bielefeld.de/talp/, Technical Report DSIC II/15/03, Universidad Politécnica de Valencia, Spain.

[29] N. OURY. *Observational equivalence and program extraction in the Coq proof assistant.* in « Typed Lambda Calculi and Applications 2003 », volume 2701, LNCS, M. HOFMANN, editor, 2003.

[30] J. SIGNOLES. *Calcul statique des applications de modules paramétrés.* in « Journées Francophones des Langages Applicatifs », 2003.

[31] FRANÇOIS-RÉGIS. SINOT, M. FERNÁNDEZ, I. MACKIE. *Efficient Reductions with Director Strings.* in « Proceedings of Rewriting Techniques and Applications (RTA'03) », series LNCS, volume 2706, Springer-Verlag, R. NIEUWENHUIS, editor, pages 46–60, 2003.

[32] C. MUÑOZ, R. BUTLER, V. CARREÑO, G. DOWEK. *Formal verification of conflict detection algorithms.* in « International Journal on Software Tools for Technology Transfer », series 4, number 3, pages 371–380, 2003.

Internal Reports

[33] N. CATAÑO, M. GAWKOWSKI, M. HUISMAN, B. JACOBS, C. MARCHÉ, C. PAULIN, E. POLL, N. RAUCH, X. URBAIN. *Logical Techniques for Applet Verification.* Deliverable, number 5.2, VerifiCard Project, 2003, Available from http://www.verificard.org.

[34] P. CORBINEAU. *First-order reasoning in the Calculus of Inductive Constructions.* Research report, number 1380, Laboratoire de Recherche en Informatique, Orsay, December, 2003.

[35] J.-C. FILLIÂTRE. *Why: a multi-language multi-prover verification tool.* Research Report, number 1366, LRI, Université Paris Sud, March, 2003, http://www.lri.fr/~filliatr/ftp/publis/why-tool.ps.gz.

Miscellaneous

[36] J. CABESSA. *Une implémentation en CAML d'un vérificateur de démonstrations pour la déduction modulo.* Mémoire de stage, DEA Programmation : Sémantique, Preuves et Langages, 2003.

[37] E. CONTEJEAN, C. MARCHÉ, A. P. TOMÁS, X. URBAIN. *Mechanically proving termination using polynomial interpretations.* 2003, submitted, 32 pages.

[38] G. DOWEK. *Preliminary investigations on induction over real numbers.* 2003, manuscript.

[39] G. DOWEK, C. MUÑOZ, V. CARREÑO. *An abstract Model of the SATS concept of operations.* 2003, manuscript.

[40] J.-C. FILLIÂTRE. *The Why verification tool.* 2003, http://why.lri.fr/.

[41] J.-C. FILLIÂTRE, C. MARCHÉ, C. PAULIN, X. URBAIN. *The* KRAKATOA *proof tool.* 2003, http://krakatoa.lri.fr/.

[42] F. KIRCHNER. *Towards a Common Tactical Language : The Case of Coq and PVS.* Technical report, DEA Programmation : Sémantique, Preuves et Langages, 2003.

[43] A. MAHBOUBI. *Induction over real numbers.* Technical report, DEA Programmation : Sémantique, Preuves et Langages, 2003.

[44] J. SIGNOLES. *ocamldefun.* 2003, http://www.lri.fr/~signoles/ocamldefun.

[45] X. URBAIN, C. MARCHÉ. *Modular & Incremental Proofs of AC-Termination.* 2003, submitted, 28 pages.