INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team LogiCal

Logic and Computation

Futurs

THEME SYM

Table of contents

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.

Head of the project-team
Benjamin Werner [CR INRIA]

Administrative assistant
Lydie Fontaine [TR INRIA]

Research scientists, INRIA staff
Bruno Barras [CR INRIA]
Gilles Dowek [Professor, Ecole Polytechnique, HdR]
Hugo Herbelin [CR INRIA, HdR]
Assia Mahboubi [CR INRIA, since september 2007]

Research scientists, Paris XI staff
Jean-Pierre Jouannaud [Professor, Université Paris-Sud, on leave from september 2006, HdR]

CNRS staff
Jean-Marc Notin [IR CNRS]

Post-doctoral fellows
Florent Kirchner [since september 2007, on leave at SRI, California]
Gyesik Lee [until May 2007]
Evgeny Makarov [until November 2007]

PhD students
Lisa Allali [Région Ile-de-France]
Bruno Bernardo [DGA]
Mathieu Boespflug [AMN, since september 2007]
Denis Cousineau [MENRT]
Danko Iliḱ [Polytechnique Monge grant, since november 2007]
Florent Kirchner [École polytechnique, until september 2007]
Sylvain Lebresne [MENRT, ATER Paris VII since september 2007]
Élie Soubiran [MENRT]
Vincent Silés [AMN, since september 2007]
Arnaud Spiwack [ENS Cachan]
Pierre-Yves Strub [EADS]
Roland Zumkeller [MENRT, granted by the INRIA MSR Joint Centre since september 2007]

2. Overall Objectives

2.1. Presentation

Many human activities have been transformed by the invention of the computer and its broad diffusion in the second half of the XX$^{\text{th}}$ century. In particular, mathematicians could, from then on, use a tool allowing to carry out operations that were too long or to tedious to be executed by hand. Like the use of the telescope in astronomy, the use of the computer opened many new prospects in mathematics. One of these prospects is the use of *proof assistants*, *i.e.* computer programs which perform some operations on mathematical proofs. The goal of the research developed in the LogiCal project-team is to develop such *proof assistants*. The main effort of the project-team is the development of the Coq system, which has an important community of users in industry and in academia. However, we believe that the development of a proof assistant cannot be accomplished without a joint reflection about the structure of mathematical proofs and about the use of proof assistants in various applicative domains. Thus, the questions addressed in the team range from questions

related to the Coq system, such as “What will be the features of the next version of Coq?”, to more theoretical questions of logic, such as “What is a proof?” and more applied ones, such as “How can we use a proof assistant to check whether a protocol is free of deadlocks?”.

2.2. Highlights of the year

The version 8.1 of the Coq system has been released in February 2007, under the coordination of Bruno Barras and Hugo Herbelin. This new evolution provides a number of new features, for libraries, tactics and meta-theory.

Gilles Dowek has received the "Grand Prix de Philosophie" of the Académie française for his book "Les métamorphoses du calcul" [11].

3. Scientific Foundations

3.1. Proof assistants

Keywords: *correctness*, *proof assistant*, *tactic language*.

The first operation that a proof assistant can perform on a proof is to check its correctness. This participates in the quest of a new step in mathematical rigor: the point where nothing is understated, and where the reader can therefore be replaced by a program. This quest for rigor is specially important for the large proofs, either hand written or computer aided, that mathematicians have built since the middle of the XX^{th} century. For instance, without using a proof assistant, it is quite difficult to establish the correctness of a proofs using symbolic computations on polynomials formed with hundreds of monomials, or a case analysis requiring the inspection of several hundreds of cases, or establishing that a complex object such as a long program or a complex digital circuit has some property. This quest for correctness is especially important in application domains where a malfunction may jeopardize human life, health or environment, such as transportations or computer aided surgery.

Besides this correctness check, proof assistants can help the users to build proofs interactively. The “tactic language” allowing the user to control the system in this proof construction process has always been the object of intensive studies. The ML language, for instance, was originally the tactic language of the LCF proof assistant. More recent questions about this language are focused on the formal expression of its operational semantic, in particular the handling of exceptions.

Proof assistants may also prove some easy lemmas automatically, transform mathematical proofs into other formal objects such as programs.

A more recent kind of applications is the construction of large libraries of mathematical results on the net.

3.2. Formalization of mathematics

Keywords: *Calculus of Constructions*, *constructive proofs*, *deduction modulo*, *mathematical language*, *predicate logic*, *programming language*, *set theory*, *type theory*.

A proof assistant implements a particular formalism allowing to express mathematics. A traditional formalism allowing to express mathematics is set theory, built on top of first-order predicate logic. Unfortunately, this formalism does not address exactly the needs of a proof assistant. Set theory has been elaborated at the beginning of the XX^{th} century to study mathematically the properties of mathematical reasoning. For this purpose, being able to formalize mathematics “in principle” was enough. Nowadays, the problem is not to formalize mathematics “in principle” but to formalize them “in facts”. Thus, the design of proof assistants has led to ask new questions in logic and, in particular, in proof theory.

Several variants or alternative to set theory have been designed to express mathematics in practice. The system Coq is based on a formalism called *The Calculus of Inductive Constructions*.

An important feature for such a formalism is the language allowing to express mathematical objects such as functions and sets. It is not possible to use a formalization of mathematics that has only existence axioms, or even one having the combinator's language obtained by skolemizing these axioms in predicate logic. It is important to have a rich and compact language, in particular a language with binders such as the λ-calculus.

Another important feature is the ability to integrate deduction and computation. It is not possible, when we use a proof assistant to consider that the proposition $2 + 2 = 4$ requires a proof, even a proof simple enough to be found by a automated theorem proving system. Several formalisms such as Martin-Löf's type theory, Boyer-Moore logic, the Calculus of Constructions and the Calculus of Inductive Constructions, include such a possibility to compute inside a proof. Thus, these formalisms designed to express mathematics contain a programming language as a sub-language.

More recently the research in this area has taken several different directions: first the study of *deduction modulo* that is the simplest extension of predicate logic allowing to mix deduction and computation. Deduction modulo has applications both in automated theorem proving and in proof theory, where it paves the way to a unified theory of cut elimination. Another direction is the design of extensions of the Calculus of Constructions with arbitrary computation rules, while the original calculus had a fixed set of rules. This extension called the *Calculus of Algebraic Constructions* may be the future formalism used in the Coq system. Finally, the need to improve the efficiency of computations in the system Coq, has led to the use of compilation techniques issued from the theory of programming language. This has brought logical languages and programming languages closer, allowing for instance to use the language of Coq as a general purpose programming language. This perspective of unifying languages and programming languages is a real challenge for future proof assistants.

Another property of the Calculus of Inductive Constructions is important for its use as the language of a proof assistant. The first is the possibility to write both constructive and classical proofs. When a proof of existence is constructive, the user can request the computation of a witness, but, of course, not when it is classical.

By insisting on this idea that constructive proofs must be distinguished from classical proofs, the project-team LogiCal participates to rise of a new form a constructivism, not trying to restrict mathematics to constructive mathematics, but trying to identify the part of mathematics that can be done constructively and the part that cannot.

A last property of the Calculus of Inductive Constructions is that proofs are objects of the formalism, exactly as numbers, functions and sets are. This property, based on the celebrated Curry-De Bruijn-Howard correspondence, allows to reduce the safety critical base of the Coq system to a quite small kernel.

4. Application Domains

4.1. Application Domains

Keywords: *algorithms*, *mathematics*, *programs*.

The applications of the research of the LogiCal project-team take several directions.

The first is the applications to pure mathematics. The use of proof assistants for proving genuine mathematical theorems has been considered as utopic for long. But several recent developments have changed the situation. First of all, the development of libraries of both constructive and classical analysis has led the possibility to use Coq, not only in remote areas of discrete mathematics, but also to prove mainstream mathematical theorem as taught in an undergrad textbook for instance. This direction culminated with the proof in Coq of the Fundamental Theorem of Algebra, a few years ago, by a group of researchers in Nijmegen. More recent work include a proof of the Four color theorem in Coq, proofs of lemma's on polynomials used in the proof of Hale's Sphere packing theorem (Kepler's conjecture), proofs in algebraic geometry by a group of mathematicians in Nice. The Mathematical Components group of the INRIA - MSR Joint Centre is working on the formalisation of the Feit Thompson theorem (1962) for groups of odd order, which is a milestone in the classification of finite groups.

Another direction is the proof of algorithms. In proofs of algorithms (as opposed to proofs of programs) a property is proved on an algorithms formalized in the language of Coq. An example is the recent proof of algorithms used in floating point arithmetic or the older proof carried out by the company *Trusted Logic* of the correctness that has reached, for the first time, the EAL7 level in common criteria.

The most applied use of Coq is the proof of programs where an actual program written in the syntax of a general purpose programming language (such as Caml, Java or C). The system Coq is used by the ProVal project-team, that has strong historical connections to LogiCal, as a back-end of their systems Why, Krakatoa and Caduceus.

5. Software

5.1. Coq

Participants: Bruno Barras, Hugo Herbelin, Christine Paulin.

The *Coq* system, developed in the project, is a processor of mathematical proofs allowing an interactive development of specifications and proofs. 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 rational 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*,
- a module system to manage name spaces, and featuring functors to develop parameterized development and making easier the instantiation of such functors,
- 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. A standalone checker of compiled libraries can be used to validate libraries with an even higher level of confidence.

Among the most significant achievements realized using *Coq*, it worths mentioning:

- the model of authentication protocol CSET used in electronic shopping and the proof of properties of this protocol,
- Gemalto's implementation of JavaCardTM,
- 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 ssr extention of the Coq system, developped by G. Gonthier while working on (see next item),
- the proof of the Four color theorem.

5.1.1. The Coq product

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

Coq is used in hundreds of sites. We have demanding users in industry (France Télécom 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.

6. New Results

6.1. Development of theories and tactics

6.1.1. Hales' Theorem

Participants: Roland Zumkeller, Benjamin Werner.

Roland Zumkeller has continued his work on global optimization, as part of an effort to formalize Thomas Hales' proof (1998) of the Kepler conjecture.

He has added several refinements to his Coq implementation of a Taylor-model based optimization algorithm. There is now a reasonable good working method to bound multi-variate polynomials. This is achieved by a branch and bound algorithm in the Bernstein basis. Furthermore, he has improved the error bounds of truncated Taylor series for a certain class of functions.

He is currently investigating ways to replace the Taylor polynomials by approximations of higher quality. A promising perspective is to import these into Coq from an external tool, such as the one developed in the team Arenaire at ENS Lyon.

6.1.2. Finite Group Theory

Participant: Assia Mahboubi.

Assia Mahboubi has contributed to the formalization of finite group theory [31], using the ssreflect extension of Coq developed by Georges Gonthier. This aim of this formalisation is to provide a formal proof of the Feit-Thompson theorem (1962), and beyond to craft a large corpus of modular and scalable libraries of formalized mathematics.

Assia Mahboubi has also written, together with Georges Gonthier, the documentation of the ssreflect extension, which will be available as an INRIA technical report.

This work is part of the Mathematical Component project in the INRIA-Microsoft Research Joint Centre.

6.1.3. Treatment of binders

Participants: François Garillot, Benjamin Werner.

Benjamin Werner and FranÃ§ois Garillot have formalized a new version of Normalization by Evaluation (NbE) in Coq. This techniques relates the shallow and deep embeddings of simply typed lambda-calculus in type theory [30]. Benjamin Werner now investigates how this technique can be applied for providing a generic treatent of languages with binders in type theory and in Coq in particular.

6.1.4. Formal libraries for numbers

Participants: Hugo Herbelin, Evgeny Makarov.

The arithmetic library of Coq has started to be almost completely rewritten, or at least restructured. The idea of the new library is to provide an axiomatic characterization of number classes (natural numbers, integers, rationals, etc.) and to allow multiple implementations of the axioms (Peano numbers, binary numbers, etc.) Several organizational models based on modules have been considered and tried. Currently the new library provides properties of addition, subtraction and multiplication for binary and Peano numbers, and the number of theorems is larger than in the old library.

This work, done by Evgeny Makarov has been presented by Hugo Herbelin at the TYPES MathWiki Workshop in Edinburgh (November 2007).

In the same effort to abstract representations of number structures, Evgeny Makarov has generalized the Coq part (as opposed to the ML part) of Frederic Besson 's tactic Micromega to work with arbitrary ordered ring instead of just integers. Micromega solves linear and polynomial equations and inequations and uses an external tool (CSDP – A Library for Semidefinite Programming).

6.1.5. Programming-driven formalization of category theory

Participant: Arnaud Spiwack.

Arnaud Spiwack has written a programming-driven developpement in Coq of a fragment of category theory. This formalisation has been used to develop a formalisation of effective homologies based on the notion of setoids. This aims at laying the bases for formalising a version of Kenzo fully internalised in Coq, which can then be used as a tool in reflexive proofs. This has helped understanding better the notion of effectivity, and its relation with formal proofs. It underlines in particular how and why the notion of setoids is central in Coq as a programming language.

Formal mathematics and programmation in effective homologies, seems to require a lot of dependent type programming. To make this smoother, Arnaud Spiwack is currently developping a new interactive proof engine for Coq, based on a finer refining procedure.

6.2. Development of systems

6.2.1. Coq 8.1

Participants: Hugo Herbelin, Bruno Barras.

Hugo Herbelin has provided support for the version 8 of Coq. See the coq web site for further details. Bruno Barras and Hugo Herbelin have coordinated the release of Coq 8.1 in February 2007, and two patch-level releases in July and October 2007.

6.2.2. Computing with machine integers

Participant: Arnaud Spiwack.

Arnaud Spiwack has extended the kernel of Coq to compute with machine integers, for both the interpreted and compiled evaluation schemes. This implementation has been interfaced with Benjamin Grégoire and Laurent Théry's library that provides algorithms on arbitrary precision numbers on top of a bounded integer library.

6.2.3. Unification

Participants: Bruno Barras, Hugo Herbelin, Vincent Silès.

Programming languages with dependant types (and also with inductive types, or subtypes) are a new target to computer language design. But to be used efficiently, they need some algorithms such as type inference (for programming languages) or automatic proof search (for logical languages). Those algorithms are based on higher order unification which is well known to be undecidable. However some subcases are still decidable.

Vincent Silès is starting a PhD under the direction of Bruno Barras and Hugo Herbelin, investigating how to improve the unification algorithms so that Coq can be used as a real programming language with dependant types.

6.2.4. Automatic scheme declaration over inductive types

Participants: Hugo Herbelin, Vincent Silès.

Inductive types are a powerful feature for a programming language, and so it is in the Coq system. But they are very few ways to handle them. Equality is a mandatory scheme to handle those types, but this equality is not always decidable. During his master degree internship, Vincent Silès has worked on a sub family where such a decidable equality can be defined. He has automatized the declaration of a boolean equality along with a proof of the decidability of Leibniz equality for inductive types in this family. He thus managed to improve the Injection tactic when dealings with dependant pairs whose first argument has a decidable equality.

6.2.5. Module system

Participants: Hugo Herbelin, Élie Soubiran.

Élie Soubiran is working with Hugo Herbelin on the module system of Coq. He has implemented new features and improvements which should be avaible in the next version of Coq. The aim of these improvements is to provide a more efficient module system by adding sharing between structures and to provide more facilities to parametrize modules and signatures.

6.2.6. A standalone checker of compiled libraries

Participant: Bruno Barras.

Bruno Barras has developed a tool to validate compiled libraries (.vo files). It can be viewed as yet another attempt to reduce the size of the kernel. Among the features that do not make it, we name the bytecode compiler, the inference of universe constraints and the incremental construction of modules and functors.

It is packaged as a standalone executable program. Since it is non extendable and works "a posteriori" on compiled files, it cannot be tainted by a malicious extension of the Coq proof editor (either ill-typed or using the ocaml Obj library). Hence, it still increases the confidence in the produced libraries.

*6.2.7. Coq**

Participant: Bruno Barras.

A variant of the Coq system has been developed. It implements a different formalism called "Implicit Calcul of Constructions", introduced by Alexandre Miquel [8].

The homepage of the system is located at http://www.lix.polytechnique.fr/Labo/Bruno.Barras/coq-implicit/.

6.3. Studies of formalisms

6.3.1. Deduction modulo

Participants: Olivier Hermant, Gilles Dowek, Benjamin Wack.

Gilles Dowek has introduced a new notion of algebras called *truth values algebras* that generalize both Heyting algebras and the algebra of the reducibility candidates. This notion of algebra allows to give a new semantic to intuitionistic deduction modulo and to introduce a model-based notion of "super-consistent" theories that implies strong normalization of proof reduction in these theories. This paper has been published in the post-proceedings of TYPES 2006.

Together with Olivier Hermant, Gilles Dowek has shown that the admissibility of the cut rule in super-consistent theories had a much simpler proof than the fact that super-consistent theories enjoyed strong normalization. They have built a new truth value algebra "the algebra of sequents" that can be seen as a collapse of the algebra of reducibility candidates. This paper has been presented at RTA 2007.

Gilles Dowek has given an invited talk that the *Second Workshop on Logical and Semantic Frameworks, with Application* that explain the steps that have lead to this convergence of reduction-based and model-based methods in proof theory.

Together with Paul Brauner and Benjamin Wack, Gilles Dowek has proved that the notion of cut elimination introduced by Benjamin Wack in Supernatural deduction coincides exactly with that of deduction modulo.

6.3.2. Towards an implementation of the Implicit Calculus of Constructions

Participants: Bruno Barras, Bruno Bernardo.

Bruno Bernardo has worked with Bruno Barras on an Implicit version of the Calculus of Inductive Constructions which is decidable. In this implicit version all the static information (types and proof objects) is transparent and does not affect the computational behavior, so we can have a practical programming language with dependent types. This worked is based on the PhD work of Alexandre Miquel [8], a former member of LogiCal. Bruno Bernardo has already defined and studied a decidable Implicit Calculus of Constructions and is now working on extending it with Inductive Types.

6.3.3. Calculus of Congruent Constructions

Participants: Jean-Pierre Jouannaud, Pierre-Yves Strub.

Pierre-Yves Strub has extended his Calculus of Congruent Construction (an extended version of the Calculus of Constructions which includes, in the rule of conversion, a decision procedure for the equality in the Presburger arithmetic). The calculus is now based on the Calculus of Inductive Constructions (i.e. it includes inductive types and, weak and strong recursors) and it allows the embedding of an arbitrary first order theory over equality in the rule of conversion.

Pierre-Yves Strub has studied the decidability for the type checking of this new calculus. He gave a decision algorithm for the case where the embedded theory is a combination of Shostak theories and ring theories.

6.3.4. Logical completeness and computations

Participants: Hugo Herbelin, Gyesik Lee, Danko Iliḱ.

Hugo Herbelin worked with Gyesik Lee and Danko Iliḱ on the very computational content of the notion of logical completeness.

Danko Iliḱ is starting a PhD on the formalization of meta-mathematical results, starting with a classical completeness theorem. The practical motivation of this work is to turn it into a tool for reflection in Coq, and the theoretical motivation is to understand better the computational content of the completeness theorem and of the axiom of choice.

6.3.5. Decidability

Participants: Gilles Dowek, Ying Jiang.

Together with Ying Jiang, Gilles Dowek has shown that the proofs of a proposition in the positive fragment of predicate logic did not form a context free langage but a quasi-context free langage, i.e. a set that is described by a context free langage and an algorithm transforming every word in this langage into a finite number of elements of the set [35].

6.3.6. Type theory

Participants: Gilles Dowek, Denis Cousineau.

Together with Denis Cousineau, Gilles Dowek has proved that all functionnal type systems could be embedded in the lambda Pi calculus modulo. This paper has been presented at TLCA 2007.

6.4. New Computation Paradigms

6.4.1. Exceptions for System F

Participant: Sylvain Lebresne.

Sylvain Lebresne has developed an extension of System F, adding a mechanism of typed exceptions to it. This mechanism differs from traditional exceptions ones in the sense that computation follows a call-by-name discipline. To type the exceptions of this system, he has introduced a new notion, the "type corruption", who have good properties and in particular allows modularity for the type system. This work has been submitted to the FLOPS conference [36]. He has started to design an adaptation of its exception mechanism for dependent types.

6.4.2. Control in λ-calculi

Participant: Hugo Herbelin.

Hugo Herbelin investigated with Zena Ariola (University of Oregon, USA) the relations between call-by-value theories of control. They show that Parigot's $\lambda\mu$-calculus satisfied much more interesting properties than the more standard λ_{e}-calculus of Felleisen et al's does [13].

Hugo Herbelin showed that two apparently disconnected calculi, namely de Groote and Saurin's variant of Parigot $\lambda\mu$-calculus and Danvy and Filinski λ-calculus of delimited control where actually the canonical call-by-name and call-by-value variants of a more general framework obtained by adding a control delimiter to Parigot $\lambda\mu$-calculus. These two calculi are interesting for their completeness properties. Especially the call-by-value variant is complete for the representation of effects. The work was done jointly with Silvia Ghilezan (University of Novi Sad, Serbia) [32].

6.4.3. Concurrency and planning

Participants: Gilles Dowek ,, Cesar Muñoz, Corina Pasareanu.

Together with Cesar Muñoz and Corina Pasareanu, Gilles Dowek, has given a formal semantics for the plan execution langage PLEXIL. This paper [29] has been presented at the *Third Workshop on Planning and Plan Execution for Real-World Systems*.

7. Contracts and Grants with Industry

7.1. EADS

The project has a a three year contract with EADS.

7.2. INRIA Microsoft Research Joint Centre

LogiCal has a strong link with the INRIA-Microsoft Research joint centre, of which Roland Zumkeller, Benjamin Werner, Assia Mahboubi and Bruno Barras are also members.

8. Other Grants and Activities

8.1. Collaboration with other teams

LogiCal and Makoto Tatsuta's team at NII (Tokyo, Japan), applied for an *équipe associées* funding.

Personnel from other INRIA teams and other academic sites have also contributed to the development of the Coq system: Christine Paulin (ProVal), Jean-Christophe Filliâtre (ProVal), Pierre Letouzey (ProVal and Paris 7), Claude Marché (ProVal), Pierre Corbineau (ProVal), Pierre Courtieu (ProVal, CNAM), Nicolas Ayache (ProVal), Matthieu Sozeau (ProVal), Benjamin Monate (ProVal), Yves Bertot (Marelle), Benjamin Grégoire, Laurent Théry (Marelle), Julien Forest (ProVal, Everest), Milad Niqui and Russell O'Connor (University of Nijmegen, Netherlands).

8.2. European actions

8.2.1. 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 Télécom, Nokia, Dassault-Aviation, Trusted Logic and Xerox companies.

For LogiCal, Benjamin Werner acts as a site leader for the whole of INRIA, and the subsites of Bologna and Minho.

9. Dissemination

9.1. Animation of the scientific community

9.1.1. Editorial charges

Gilles Dowek has been part of the organizing team of TPR and the Colloquium in honnor of Gérard Huet.

9.1.2. Committees

Gilles Dowek has been part of the thesis committee of Germain Faure and of the HDR committee of Luigi Luquori

Hugo Herbelin has been a reviewer of the Doctoral Dissernation of Dragiža Žunić (Lyon, December 2007).

Hugo Herbelin served on Lionel Vaux' Doctoral Dissernation Comitee (Marseille, November 2007).

Benjamin Werner has been member of the program committee of the FLOPS 2008 (Ise, Japan) conference.

9.1.3. Referees

Hugo Herbelin served as referee for the CSL '07, ICFP '07, TLCA '07, TYPES '07, POPL '08 and FLOPS '08 conferences.

Hugo Herbelin served as referer for the TOCL and MSCS journals.

Benjamin Werner served as referee for the TYPES '07, POPL '08 and ICALP '07 conferences.

9.1.4. Visits

Gilles Dowek has spent five weeks as a consultant at the Instutute of Aerospace.

Hugo Herbelin has visited Silvia Ghilezan at University of Novi Sad (Serbia) in November.

Benjamin Werner was invited by Makoto Tatsuta at NII, Tokyo, Japan, from December 15^{th} to 28^{th}.

Bruno Bernardo has given a talk at the seminar of the team Gallium at INRIA Rocquencourt, May 2007.

Arnaud Spiwack gave a talk at the Protheo seminar (INRIA Nancy) in July.

Roland Zumkeller visited the team Arenaire at ENS Lyon, where he gave a seminar talk.

9.1.5. Conferences

Benjamin Werner has presented an article [30] at TPHOLs 2007 (Kaiserslautern Germany).

Lisa Allali, Arnaud Spiwack, Bruno Bernardo, Denis Cousineau and Benjamin Werner have given a talk at the TYPES 2007 meeting (Cividale del Friuli, Italy) in May. Bruno Barras, Gilles Dowek, Hugo Herbelin, Sylvain Lebresne, Evgeny Makarov, Jean-Marc Notin, Élie Soubiran, and Pierre-Yves Strub have attended this conference as well.

Gilles Dowek has participated to the QICS workshop, to the *Second Workshop on Logical and Semantic Frameworks, with Application* (LSFA '07) and to the CADE '07 conference. He has given an invited talk at LSFA '07.

Hugo Herbelin has given talks at the TYPES MathWiki Worshop (November, Edinburgh) and at the TYPES Effects and Type Theory Workshop (December, Tallinn).

Denis Cousineau, Gilles Dowek, Hugo Herbelin and Evgeny Makarov and have attended the TLCA '07 (June, Paris) conference. Denis Cousineau presented there an article [25].

Bruno Barras, Gilles Dowek and Hugo Herbelin have attended the RTA '07 conference.

Bruno Barras and Bruno Bernardo have attended the TPR '07 (June, Paris) conference. Bruno Barras has given a talk there.

Lisa Allali, Bruno Bernardo, Denis Cousineau, Evgeny Makarov, Élie Soubiran and Arnaud Spiwack have attended the TYPES Summer School in August (Bertinoro, Italy).

Lisa Allali attended the *Journées Modulo* in Paris in January.

Lisa Allali attended the workshop of the *Logique Algèbre et Calcul* group in Chambéry February 2007.

Mathieu Boespflug attended the ICFP '07 conference (Freiburg, Germany) along with the associated Haskell Workshop'07 in October 2007.

Bruno Barras, Bruno Bernardo, Gilles Dowek, Evgeny Makarov and Benjamin Werner attended the Colloquium in Honor of Gérard Huet (June 22-23, 2007) in Paris.

Arnaud Spiwack gave a talk at the first french-spanish Congress of Mathematics (Saragossa, Spain).

Gilles Dowek, Arnaud Spiwack and Roland Zumkeller attended the MAP'07 Meeting (Leiden, Netherlands). Roland Zumkeller has given a talk at this meeting.

Pierre-Yves Strub has participated to the *Dagsthul Seminar* on Deduction and Decision Procedures, where he has given a talk.

Pierre-Yves Strub has attended CSL '07, where he has given a talk.

Roland Zumkeller attended Rencontres Arithmétiques de l'Informatique Mathématique in Montpellier, where he gave a talk.

9.1.6. Popular science

Bruno Bernardo, Denis Cousineau and Élie Soubiran presented a poster on *small worlds* for the *fête de la Science* on October the 12th and the 13th, at the University Paris VII.

Gilles Dowek has received the "Grand Prix de Philosophie" of the Académie française for his book "Les métamorphoses du calcul" [11].

9.1.7. Other charges

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

Bruno Barras is consultant in formal methods at Trusted Labs, located in Versailles.

Bruno Bernardo, Denis Cousineau and Jean-Marc Notin are the webmasters of the Coq and LogiCal websites.

9.2. Teaching

Gilles Dowek is the thesis advisor of Mathieu Boespflug, Denis Cousineau and Lisa Allali. Hugo Herbelin is the thesis advisor of Élie Soubiran, Danko Iliḱ and co-advisor of Sylvain Lebresne. Benjamin Werner is thesis advisor of Roland Zumkeller and co-adviser of Arnaud Spiwack. Bruno Barras is thesis advisor of Bruno Bernardo. Bruno Barras and Hugo Herbelin coadvise the thesis of Vincent Silès. Jean-Pierre Jouannaud is thesis advisor of Pierre-Yves Strub.

Gilles Dowek, Hugo Herbelin and Benjamin Werner teach at the *Master Parisien de Recherche en Informatique*.

Bruno Bernardo and Arnaud Spiwack are teaching assistants at the École Polytechnique.

Lisa Allali is teaching assistant at the *Museum National d'Histoire Naturelle*.

Denis Cousineau and Sylvain Lebresne are teaching assistants at the University Paris VII.

Élie Soubiran is teaching assistant at University Paris XII.

Pierre-Yves Strub has been a teaching assistant at Ecole Polytechnique in April-July.

Gilles Dowek is professor at École Polytechnique.

Benjamin Werner is part-time professor (proofesseur chargé de cours) at École Polytechnique since september.

Gilles Dowek has given a course at ISR in Nancy.

Roland Zumkeller has given a course on "Proofs of programs" at ENSTA.

Benjamin Werner served as a corrector for the entrance examination of Ecole Polytechnique.

10. Bibliography

Major publications by the team in recent years

[1] G. DOWEK. *Les Métamorphoses du Calcul*, Le Pommier, 2007.

[2] G. DOWEK, O. HERMANT. *A Simple Proof That Super-Consistency Implies Cut Elimination*, in "Term Rewriting and Applications, 18th International Conference, RTA 2007, Paris, France, June 26-28, 2007, Proceedings", F. BAADER (editor), Lecture Notes in Computer Science, vol. 4533, Springer, 2007, p. 93-106.

[3] G. DOWEK, B. WERNER. *Proof Normalization Modulo*, in "Journal of Symbolic Logic", vol. 68-4, 2003, p. 1289-1316.

[4] G. GONTHIER, A. MAHBOUBI, L. RIDEAU, E. TASSI, L. THÉRY. *A Modular Formalisation of Finite Group Theory*, in "Theorem Proving in Higher Order Logics, 20th International Conference, TPHOLs 2007, Kaiserslautern, Germany, September 10-13, 2007, Proceedings", K. SCHNEIDER, J. BRANDT (editors), Lecture Notes in Computer Science, vol. 4732, Springer, 2007, p. 86-101.

[5] B. GRÉGOIRE, L. THÉRY, B. WERNER. *A computational approach to Pocklington certificates in type theory*, in "FLOPS 2006", M. HAGIYA, P. WADLER (editors), LNCS, vol. 3945, Springer, 2006.

[6] H. HERBELIN, S. GHILEZAN. *An Approach to Call-by-Name Delimited Continuations*, in "Proceedings of the 35th SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2008", to appear, ACM, 2007.

[7] H. HERBELIN. *C'est maintenant qu'on calcule, au cœur de la dualité*, Habilitation à diriger des recherches, Ph. D. Thesis, Université Paris-Sud, 2005.

[8] A. MIQUEL. *Le Calcul des Constructions Implicites : syntaxe et sémantique.*, Ph. D. Thesis, Université Paris VII, 2001.

[9] J. NARBOUX. *Mechanical Theorem Proving in Tarski's geometry*, in "Proceedings of Automatical Deduction in Geometry", 266.

[10] R. ZUMKELLER. *Formal Global Optimisation with Taylor Models*, in "Int. Joint Conf. Automated Reasoning — IJCAR 2006", U. FURBACH, N. SHANKAR (editors), LNAI, vol. 4130, Springer, 2006.

Year Publications

Books and Monographs

[11] G. DOWEK. *Les Métamorphoses du Calcul*, Le Pommier, 2007.

Doctoral dissertations and Habilitation theses

[12] F. KIRCHNER. *Interoperable proof systems*, Ph. D. Thesis, École Polytechnique, 2007.

Articles in refereed journals and book chapters

[13] Z. M. ARIOLA, H. HERBELIN. *Control Reduction Theories: the Benefit of Structural Substitution*, in "Journal of Functional Programming", to appear, 2007.

[14] Z. M. ARIOLA, H. HERBELIN, A. SABRY. *A Type-Theoretic Foundation of Delimited Continuations*, in "Higher Order and Symbolic Computation", to appear, 2007.

[15] H. CIRSTEA, G. FAURE, M. FERNÁNDEZ, I. MACKIE, F.-R. SINOT. *From Functional Programs to Interaction Nets via the Rewriting Calculus*, in "Electronic Notes in Theoretical Computer Science", vol. 174, n^{o} 10, 2007, p. 39–56.

[16] G. GONTHIER, B. WERNER. *Le théorème des quatre couleurs: ingénierie d'une preuve formelle*, in "La lettre de l'Académie des sciences", vol. 21, 2007.

[17] M.-D. HERNEST. *Synthesis of moduli of uniform continuity by the Monotone Dialectica Interpretation in the proof-system* MINLOG, in "Electronic Notes in Theoretical Computer Science", Elsevier, vol. 174, n^{o} 5, 2007, p. 141-149.

[18] J.-P. JOUANNAUD, I. MACKIE. *Preface*, in "Electr. Notes Theor. Comput. Sci.", vol. 171, n^{o} 3, 2007, p. 1-2, http://dx.doi.org/10.1016/j.entcs.2006.12.036.

[19] J.-P. JOUANNAUD, A. RUBIO. *Polymorphic higher-order recursive path orderings*, in "J. ACM", vol. 54, n^{o} 1, 2007, http://doi.acm.org/10.1145/1206035.1206037.

[20] L. LIQUORI, A. SPIWACK. *FeatherTrait: A Modest Extension of Featherweight Java*, in "ACM Transaction on Programming Languages and Systems", to appear, 2007.

[21] J. NARBOUX. *A Graphical User Interface for Formal Proofs in Geometry*, in "the Journal of Automated Reasoning special issue on User Interface for Theorem Proving", vol. 39, n^{o} 2, 2007, p. 161–180, http://www.lix.polytechnique.fr/Labo/Julien.Narboux/papers/JARuitp-narboux.ps.gz.

Publications in Conferences and Workshops

[22] B. BARRAS, B. BERNARDO. *The Implicit Calculus of Constructions as a Programming Language with Dependent Types*, in "Workshop on Type theory, Proof theory, and Rewriting", 2007.

[23] F. BLANQUI, J.-P. JOUANNAUD, P.-Y. STRUB. *Building Decision Procedures in the Calculus of Inductive Constructions*, in "Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings", J. DUPARC, T. A. HENZINGER (editors), Lecture Notes in Computer Science, vol. 4646, Springer, 2007, p. 328-342.

[24] T. COQUAND, A. SPIWACK. *Towards Constructive Homological Algebra in Type Theory*, in "Proceedings of 14th Symposium, Calculemus 2007, 6th International Conference, MKM 2007", Lecture Notes in Artifical Intelligence, vol. 4573, Springer, 2007.

[25] D. COUSINEAU, G. DOWEK. *Embedding Pure Type Systems in the Lambda-Pi-Calculus Modulo*, in "Typed Lambda Calculi and Applications, 8th International Conference, TLCA 2007, Paris, France, June 26-28, 2007, Proceedings", S. RONCHI DELLA ROCCA (editor), Lecture Notes in Computer Science, vol. 4583, Springer, 2007, p. 102-117.

[26] G. DOWEK. *On the convergence of reduction-based and model-based methods in proof theory*, in "Proceedings of the econd Workshop on Logical and Semantic Frameworks, with Applications", 2007.

[27] G. DOWEK. *Truth values algebras and proof normalization*, in "Types for Proofs and Programs, International Workshop, TYPES 2006, Nottingham, UK, April 18-21, 2006, Revised Selected Papers", Lecture Notes in Computer Science, vol. 4502, Springer, 2007.

[28] G. DOWEK, O. HERMANT. *A Simple Proof That Super-Consistency Implies Cut Elimination*, in "Term Rewriting and Applications, 18th International Conference, RTA 2007, Paris, France, June 26-28, 2007, Proceedings", F. BAADER (editor), Lecture Notes in Computer Science, vol. 4533, Springer, 2007, p. 93-106.

[29] G. DOWEK, C. MUÑOZ, C. PASAREANU. *A Formal Analysis Framework for PLEXIL*, in "Proceedings of the Third Workshop on Planning and Plan Execution for Real-World Systems", 2007.

[30] F. GARILLOT, B. WERNER. *Simple Types in Type Theory: Deep and Shallow Encodings*, in "Theorem Proving in Higher Order Logics, 20th International Conference, TPHOLs 2007, Kaiserslautern, Germany, September 10-13, 2007, Proceedings", Lecture Notes in Computer Science, vol. 4732, Springer, 2007, p. 368-382.

[31] G. GONTHIER, A. MAHBOUBI, L. RIDEAU, E. TASSI, L. THÉRY. *A Modular Formalisation of Finite Group Theory*, in "Theorem Proving in Higher Order Logics, 20th International Conference, TPHOLs 2007, Kaiserslautern, Germany, September 10-13, 2007, Proceedings", K. SCHNEIDER, J. BRANDT (editors), Lecture Notes in Computer Science, vol. 4732, Springer, 2007, p. 86-101.

[32] H. HERBELIN, S. GHILEZAN. *An Approach to Call-by-Name Delimited Continuations*, in "Proceedings of POPL '08", to appear, ACM, 2007.

[33] F. KIRCHNER. *A Finite First-order Theory of Classes*, in "Types for Proofs and Programs, International Workshop, TYPES 2006, Nottingham, UK, April 18-21, 2006, Revised Selected Papers", Lecture Notes in Computer Science, vol. 4502, Springer, 2007.

Miscellaneous

[34] P. BRAUNER, G. DOWEK, B. WACK. *Normalization in Supernatural deduction and in Deduction modulo*, Manuscript, 2007.

[35] G. DOWEK, Y. JIANG. *Enumerating proofs of positive formulae*, Manuscript, 2007, http://www.lix.polytechnique.fr/~dowek/Publi/enumerationbracket.pdf.

[36] S. LEBRESNE. *A system F with exceptions*, Submitted for publication, 2007.