INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team TypiCal

Types, Logic and Computation

Saclay - Île-de-France

THEME SYM

Table of contents

The TypiCal project is a common project gathering researchers from INRIA Saclay – Île-de-France sud, Ecole Polytechnique and CNRS at LIX.

1. Team

Research Scientist

Benjamin Werner [CR INRIA, HdR]
Bruno Barras [CR INRIA]
Germain Faure [CR INRIA, since september 2008]
Hugo Herbelin [CR INRIA, HdR]
Assia Mahboubi [CR INRIA]

Faculty Member

Gilles Dowek [Professor, Ecole Polytechnique, HdR]

Technical Staff

Jean-Marc Notin [IR CNRS]

PhD Student

Lisa Allali [Région Ile-de-France]
Bruno Bernardo [DGA]
Mathieu Boespflug [AMN]
Denis Cousineau [MENRT]
Danko Iliḱ [Polytechnique Monge grant]
Sylvain Lebresne [MENRT, until december 2008]
Élie Soubiran [MENRT, until december 2008]
Vincent Silés [AMN, until december 2008]
Arnaud Spiwack [ENS Cachan]
Pierre-Yves Strub [EADS, until september 2008]

Administrative Assistant

Lydie Fontaine [TR INRIA]

2. Overall Objectives

2.1. Presentation

Mathematics is among the many human activities that have been transformed by the invention of the computer and its broad diffusion in the second half of the XX$^{\text{th}}$ century. Mathematicians could, from then on, use a tool allowing to carry out operations that were too long or too 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 TypiCal project-team is to develop such *proof assistants*. The main effort of the project-team is to contribute to 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 creation of the TypiCal INRIA project-team, following the previous LogiCal project-team in january 1st 2008. Furthermore, Pierre-Louis Curien and Hugo Herbelin created a new INRIA team as part of the PPS laboratory at Paris 7; so Hugo Herbelin and his students leave TypiCal at the end of 2008.

Roland Zumkeller defended his PhD thesis which opens new connections between formal proofs and techniques from applied mathematics like real optimization.

The version 8.2 of the Coq system is about to be released. Also, the SSR proof tactics package has been released by the INRIA MSR joint centre with an active participation of Typical researchers.

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 TypiCal 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 TypiCal 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 TypiCal, as a back-end of their systems Why, Krakatoa and Caduceus.

5. Software

5.1. Coq

Participants: Bruno Barras, Hugo Herbelin, Jean-Marc Notin.

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. In his PhD, defended in october, he proposes ways to replace Taylor polynomials by better approximations for computing approximations of real functions. The tools suite he proposes can solve most inequalities encoutered in Hales's proof.

6.1.2. Modular formal libraries

Participant: Assia Mahboubi.

Jointly with with G. Gonthier (Microsoft Cambridge), Y. Bertot, S. Ould Biha, L. Rideau, L. Théry (INRIA Sophia Antipolis), F. Garillot (INRIA Saclay).

In the context of the Mathematical Components group inside the Microsoft INRIA Joint Centre, Assia Mahboubi has worked on the developement of Coq libraries on group theory. This work consists in a global reflexion on the appropriate methodology for the design of large and modular libraries of formalised mathematics. In particular, Assia Mahboubi has pursued her reflexion on quotients by studying the formalisation of composition series and by formalising the generalised Jordan-Hölder theorem for finite groups.

6.1.3. Treatment of binders

Participants: Gilles Dowek, Jean-Marc Notin, Benjamin Werner.

Gilles Dowek, M. James Gabbay, and Dominic Mulligan have proposed a new form of nominal terms where explicit freshness contraints have been internalized in the syntax of terms. They have proposed a new unification algorithm for such terms.

Gilles Dowek and M. James Gabbay have proposed a reduction of these nominal unification problems to pattern unification in the style of Levy and Villaret. They have shown that this reduction preserves not only unifiability but also the set of solutions.

Gilles Dowek, M. James Gabbay, and Dominic Mulligan have proposed a new form of nominal terms where explicit freshness contraints have been internalized in the syntax of terms. They have proposed a new unification algorithm for such terms.

Gilles Dowek and M. James Gabbay have proposed a reduction of these nominal unification problems to pattern unification in the style of Levy and Villaret. They have shown that this reduction preserves not only unifiability but also the set of solutions.

Benjamin Werner and Jean-Marc Notin have started a formalization aimed at a more generic representation of variable binding in Coq. It builds on last year's work by Werner and François Garillot.

6.1.4. Type Theory

Participants: Bruno Barras, Bruno Bernardo, Vincent Silès.

Bruno Bernardo and Bruno Barras on an Implicit Calculus of Inductive Constructions with decidable type inference. In this calculus all the static information (types and proof objects), though it appears explicitly, is transparent and does not affect the computational behavior. Bruno Bernardo has already defined and studied an Implicit Calculus of Constructionswith decidable typing and is now working on extending it with Inductive Types, so we can express the usual data structures that every common programming language has. A problem in the metatheory of this extension has been uncovered: as in first-order logic with implicit existential quantifier, the subject-reduction property do not hold. However, this negative result do not affect more explicit versions of ICC, where introduction and elimination rules of the implicit quantifier are used explicitely in the proof derivation, but are considered implicit in the conversion rule.

Vincent Silès is working on a formalization of PTS which includes η-reduction, subtyping and typed conversion which is notoriously difficult. He proved subject reduction property and still trying to deal with the injectivity of Π function types. He also recently formalized in Coq a proof that untyped lambda calculus (as in no types in lambdas) with $\beta\eta$-reduction is confluent.

6.1.5. Programming-driven formalization of category theory

Participant: Arnaud Spiwack.

As part of his PhD work, Arnaud Spiwack has modelised some linear algebra in Coq; in particular he has a certified algorithm in Coq which computes bases of kernels of rationnal-valued matrices (as part of a more abstract framework). Also he modelized chain complexes (main tool for homological algebra). Work in progress is to find a more concise mathematical core to make these developpement more robust (investigating in particular polycategorical and higher-categorical settings).

6.2. Development of systems

6.2.1. Coq 8.1

Participants: Hugo Herbelin, Bruno Barras.

Hugo Herbelin coordinated the development of Coq. A patch level release of Coq 8.1 went out in October while a new version Coq 8.2 came out for beta-test in June with a final release in December.

An INRIA technological development action (ADT) dedicated to Coq was set up. Hugo Herbelin coordinates this 2-year action which started in October.

Hugo Herbelin and Élie Soubiran improved the module system of Coq. Matthieu Sozeau (from ProVal) and Herbelin improved the support for unification and type inference in Coq. Bruno Barras, Matthieu Sozeau, Pierre Letouzey (from Paris 7) and Stéphane Glondu (Paris 7) improved the tactic language of Coq.

6.2.2. Release, maintenance and documentation of the ssreflect extension

Participant: Assia Mahboubi.

Together with with Georges Gonthier (Microsoft Cambridge), Assia Mahboubi maintains the code of the ssreflect extension for the Coq system. This extension consists both in an extension of Coq's language and in a subset of the libraries developed in the Mathematical Component group. She was in charge of the 1-1 release of this extension, which is the first joint INRIA - Microsoft Research release of code, under Cecil B license. Gonthier and Mahboubi have documented this language extension [15].

6.2.3. Tagging verified compiled libraries

Participant: Bruno Barras.

Bruno Barras has developped an experimental version of the standalone checker of compiled libraries which allows signing the verified libraries. The idea is to share the checking of libraries between different "checking sessions". This is achieved by appending certificates to compiled libraries. The current prototype uses very weak encryption of such "checking certificates" in order to guarantee their authenticity, but it could be extended by using algorithms of cryptographic strength.

6.2.4. Complete decision Procedures for arithmetic(s)

Participant: Assia Mahboubi.

Joint work with Pierre-Yves Strub (LIAMA, Beijing) and Pierre Letouzey (Université Paris 7).

Assia Mahboubi investigates the way to modernise and modularise the existing decision procedures of the Coq system for integer, rational, and real arithmetic. This projects covers several aspects of the problem of implementing proof producing decision procedures. The first aspect is to think again the implementation of these procedures in terms of Coq efficient programs, meant to be both executed and formally proved correct. The second aspect is to reorganise from scratch the implementation of these procedures, taking into account the relative dependencies of the decision problems (like the links between real and integer arithmetic). An other aspect is to integrate these procedure in a modular ways. Like for modern software, there is a need for shared data structure libraries (like polynomials) and modular code (like for the abstraction of the concrete representation of the numbers). Finally we want to understand how complete but inefficient decision procedure interact with efficient specialised heuristics in a proof assistant. In this context, Mahboubi and Strub has started the implementation of a simplex algorithm in Coq.

6.2.5. Goal directed proof search methods

Participants: Germain Faure, Stéphane Lengrand, Assia Mahboubi.

When building formal proof in a proof assistant, proof-search is central and we want it to be as automated as possible. Indeed, proof-search is a common aspect of formal mathematics and automated reasoning, as well as high-level programming paradigms such as Logic programming. But to enlarge the scope where a specific method applies, one can combine both generic proof search mechanisms with specific methods. This is similar to Satisfiability Modulo Theories (SMT) and Constraint Logic Programming: to have efficient solvers for dedicated theories within a general-purpose engine. In fact, if over the last fifteen years, new features (polarisation, focusing, etc) have emerged to enhance the proof-search mechanisms at the heart of logic programming, their design has so far been guided by the objective of completing a proof. We investigate how how these proof search strategies can be guided by the more general objective of reaching the scope where a domain-specific method can be applied. We also want these goal directed strategies to serve as a basis for proof search automation inside higher-order logic based proof assistants.

6.2.6. SAT-solving

Participant: Germain Faure.

This work is in close collaboration with the BARCELOGIC team headed by Robert Nieuwenhuis.

First, we contribute to the development of SAT-solvers. We propose some new techniques to analyze and simplify the implication graph used to represent binary clauses. This has some important consequences. In particular, it allows to speed up the unit propagation, which is at the heart of SAT-solvers. Moreover, some important benchmarkings has been done to show that the more clever exploitation of the implication graph allows to detect new units at decision level 0.

6.2.7. Operational research and SAT modulo the theory of linear arithmetic

Participant: Germain Faure.

This work is in close collaboration with the BARCELOGIC team headed by Robert Nieuwenhuis.

Here we analyze why it is difficult to use linear arithmetic optimization tools inside systems for SAT Modulo Theories (SMT) for linear arithmetic. We show what the difficulties are and how to solve them. We report on extensive experiments that ILOG CPLEX tend to be slower than Barcelogic.

6.2.8. Semantics of rewriting strategies

Participant: Germain Faure.

Germain Faure is currently working with Horatiu Cirstea (PAREO team INRIA Lorraine - Grand Est) to propose an encoding of strategic rewriting in pattern-based λ-calculi. The first aim is to give a semantics to rewriting strategies. Moreover, when the congruence of deduction modulo is defined by a strategic rewriting system, we want to propose a proof term language sufficiently expressive.

6.3. Study of Formalisms

6.3.1. 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, a former member of TypiCal. Bruno Bernardo has already defined and studied a decidable Implicit Calculus of Constructions and is now working on extending it with Inductive Types.

6.3.2. Calculus of Congruent Constructions

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

In his Ph.D., Pierre-Yves Strub gives the definition of and studies a new calculus, the Calculus of Congruent and Inductive Constructions, an extension of the Calculus of Inductive Constructions integrating in its computational part the entailment relation - via decision procedures - of a first order theory over equality. A major technical innovation of this work lies in the computational mechanism: goals are sent to the decision procedure together with a set of user hypotheses available from the current context.

Pierre-Yves Strub shows that this extension does not compromise the main properties of the Calculus of Induction Constructions: subject reduction, strong normalization of the reduction, logical consistency and decidability of proof checking are all preserved (as soon as the incorporated theory is itself decidable).

This work has been presented in [8] and a journal version of the paper is in preparation.

6.3.3. 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.

6.3.4. Normalization and deduction modulo

Participants: Mathieu Boespflug, Denis Cousineau.

Mathieu Boespflug has been working on generalizing normalization by evaluation to term reduction modulo arbitrary rewrite rules in addition to beta-reduction. This allows for a cheap yet efficient implementation of a normalizer, as required in many proof assistants and also finds applications in partial evaluation. The implementation is cheap because most of the work is offloaded to an existing evaluator. His work has focused in particular on minimizing any overhead on beta-reduction caused by normalization by evaluation, showing that normalization by evaluation can reduce terms at nearly the same speed as the underlying evaluator. A paper has been submitted to PLDI. Other work on minimizing computation in proof assistants is ongoing.

Denis Cousineau has worked on the property of strong normalization in logical frameworks where theories are expressed with rewrite rules. He defined, in particular, a semantic criterium not only correct but also complete for the property of strong normalization for theories expressed in Minimal Deduction modulo (a minimal version of Deduction modulo with the only two connectors $\Rightarrow$ and $\forall$). These results were presented et the french workshop *Modulo*, and are to be submitted to an international conference. Denis Cousineau is now working on extending these results to dependent types ($\lambda\Pi$-calculus modulo), and on adapting these new tools to weak normalization.

6.4. New Computation Paradigms

6.4.1. Normalization

Participants: Lisa Allali, Denis Cousineau, Gilles Dowek.

Lisa Allali worked with Baul Brauner (Protheo/LORIA) on new representations of inductive types. There are usually two ways of interpreting them in a Curry-Howard manner, either by encoding their inhabitants by lambda-terms, either by recursors. They propose to link the two approaches the following way: recursors of inductive types are naturally defined as the elimination rules generated by the transformation of equations into supernatural deduction rules. This simplifies and breaks the asymmetry of usual inductive types presentations. Moreover, the strong normalization property of the resulting system is obtained by using a simple semantic argument. This results are submitted for publication.

Lisa Allali is also currently working on several research considerations including:

- Models of Linear Logic modulo with Olivier Hermant
- Arithmetic in Linear modulo with David Baelde (Parsifal)
- Linear Calculus Of Construction and modulo with Nicolas Guenot (Parsifal)
- Logic of definitions and deduction modulo with Alexis Saurin (Torino)

6.4.2. Expressive Type Systems for Computation

Participants: Sylvain Lebresne, Vincent Silès.

Sylvain Lebresne has finished developing a type system for exceptions in call-by-name languages. This type system, where types mentions the uncaught exceptions a term can raise, uses the new notion of "type corruption" to handle the specificities of call-by-name evaluation. An adaptation of this type system to System F has led to a publication in the ICALP 08 conference [11]. He defended his Ph.D. [1] on this topic on December 5^{th}, 2008.

Vincent Silès has worked on using a type inference algorithm derived of the one of System F to make the computation of implicits arguments more efficient. This has lead to the PLMMS submission but the resultats is that this method will not scale to type dependant system, so this research is now stopped.

6.5. Quantum Computation

6.5.1. Quantum computing

Participants: Gilles Dowek, Benoit Valiron.

Pablo Arrighi and Gilles Dowek have published they work on the langage Lineal that is an extension of lambda-calculus allowing to handle linear functions.

Benoit Valiron recently joined the team and works with Gilles Dowek on the study of algebraic lambda-calculi: typed and untyped languages and their semantics. The work is done in partnership with the research group on quantum computation (QCG, member of CAPP) in the computer science lab of Grenoble.

6.5.2. Physics of computation

Participant: Gilles Dowek.

Gilles Dowek has presented a paper on non determinsitic computation over the real numbers first at the meeting *"La thèse de Church, hier aujourd'hui et demain"* then at the "NKS Midwest conference: What is Computation? How does Nature Compute?"

6.5.3. lambda-calculi

Participant: Hugo Herbelin.

Hugo Herbelin worked with Stéphane Zimmermann (PPS, Paris 7) on a fine analysis of the operational properties of call-by-value λ-calculus. They improved over Plotkin's and Moggi's approaches and, in presence of control operators, over Sabry and Felleisen's and Hofmann's approaches, by showing that the theory of call-by-value can be split into three distinct independent subsystems: the operational subsystem which is necessary and sufficient to evaluate closed terms, the structural subsystem which is necessary and sufficient to ensure normalisation of open terms, and finally the observational subsystem which is necessary for identifying observationally undiscriminable terms.

7. Contracts and Grants with Industry

7.1. INRIA Microsoft Research Joint Centre

TypiCal 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.

7.2. Digithéo PASO

The PASO project *(Preuves, Interprétation abstraite, Optimisation)* est commun avec l'équipe-projet Maxplus (INRIA, CMAP), l'équipe MEASI (CEA, LIX) et Supélec.

7.3. ANR Decert

Assia Mahboubi is part of the ANR Decert *Décision certifiée* coordinated by Thomas Jensen in Rennes.

7.4. European actions

7.4.1. Formal mathematics

The *Working Group* "TYPES" about computer aided development of proofs and programs ended this year.

It was 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.

We have applied for a new project on formal mathematics, with some of these partners.

8. Dissemination

8.1. Animation of the scientific community

8.1.1. Editorial charges

Gilles Dowek has been co-PC chair for IJCAR.

8.1.2. Committees

Gilles Dowek has been part of the evaluation committee of the Teaching Programe Review of the University of Edinburgh

Gilles Dowek has been a member of the Commission de Specialiste de Paris 7

Gilles Dowek has been in the Jury of the HDR of Etienne Moreau in the thesis committee of Cristobal Rojas and in that of David Baelde

Gilles Dowek has been PC member for CSL, LSFA, TCSB.

Hugo Herbelin has been PC member for PLMMS '08 (workshop of MKM), CSLSI '08 (workshop of ESSLLI), POPL '09, JFLA '09. He was PhD Jury member for Alexander Summers (referee, November 2008, London), Sylvain Lebresne (co-supervisor, December 2008, Paris), Boris Yakobowski (referee, December 2008, Paris).

Benjamin Werner has been PhD Jury member for Roland Zumkeller. He was PC member for FLOPS 09.

8.1.3. 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.

8.1.4. Visits

Gilles Dowek visited Ying Jiang in Beijing for one week.

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

8.1.5. Conferences

Gilles Dowek, Hugo Herbelin, Arnaud Spiwack, Lisa Allali, Assia Mahboubi attended TYPES'08 (Torino).

Hugo Herbelin presented a paper at POPL'08. He attended MKM'08 and ICFP'08.

Gilles Dowek gave presentations at RTA 08, TAMC 2008 (Xian, China), the NKS Midwest conference "How does Nature Compute", the meeting "Avenir de l'Enseignement et de la Recherche", the Jacques Morgenstern Colloquium (Sophia-Antipolis).

Arnaud Spiwack gave seminar talks in Nijmegen, Eindhoven and Saragossa. He attended the MA summer school in Trieste.

Vincent Silès attended the Summer School on Logic and Theorem Proving in Programming Languages in Eugene, Oregon.

Bruno Bernardo has attended the Dependently Typed Programming 2008 (DTP'08) workshop at University of Nottingham, UK, where he gave a talk (February 2008).

Bruno Bernardo and Bruno Barras have attended Tools and Techniques for Verification of System Infrastructure, A Festschrift in Honour of Prof. Michael J. C. Gordon FRS at Roy al Society, London, UK (March 2008).

Bruno Bernardo and Bruno Barras presented an article at FoSSaCS'08, which is part of ETAPS 2008. They attended the ETAPS 2008 conferences in Budapest, Hungary (April 2008).

Bruno Bernardo has attended the Réasibilité à Chambéry workshop at Université de Savoie, Chambéry, France (June 2008).

Mathieu Boespflug and Arnaud Spiwack attended the Dependently Typed Programming Workshop in Nottingham.

Pierre-Yves Strub presented a paper at IFIP TCS in Milano.

8.1.6. Popular science

Benjamin Werner gave an interview to the magazine *La Recherche* in december.

Denis Cousineau, Bruno Bernardo and Elie Soubiran conceived and presened a poster about Programming Languages and Compilation at the *Fête de la Science*, a national popular science event, at Université Paris 7 - Denis Diderot (November 2008).

8.1.7. Other charges

Gilles Dowek is vice-head of the department of computer science of Ecole Polytechnique.

Benjamin Werner is member of the scientific council of INRIA.

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 TypiCal websites.

8.2. 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 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.

Bruno BArras, 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.

Vincent Silès is teaching assistant at Ecole Polytechnique in April-July.

Gilles Dowek is professor at École Polytechnique.

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

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

9. Bibliography

Year Publications

Doctoral Dissertations and Habilitation Theses

[1] S. LEBRESNE. *Une approche de la détection statique d'exceptions non rattra pées en appel par nom*, Ph. D. Thesis, Université Paris Diderot – Paris 7, 2008.

[2] B. WERNER. *Faire simple pour pouvoir faire compliqué: contributions à une théorie des types pratique*, Habilitation à diriger des Recherches, Ph. D. Thesis, Université Paris-Sud, 2008.

Articles in International Peer-Reviewed Journal

[3] L. LIQUORI, A. SPIWACK. *Extending FeatherTrait Java with Interfaces*, in "Theoretical Computer Science Volume 398, Issues 1-3, , Calculi, Types and Applications: Essays in honour of M. Coppo, M. Dezani-Ciancaglini and S. Ronchi Della Rocca", 2008, p. 243-260.

[4] L. LIQUORI, A. SPIWACK. *FeatherTrait: A modest extension of Featherweight Java*, in "ACM Trans. Program. Lang. Syst.", vol. 30, n^{o} 2, 2008, p. 1–32, http://doi.acm.org/10.1145/1330017.1330022.

[5] B. WERNER. *On the strength of proof-irrelevant type theories*, in "Logical Methods in Computer Science", vol. Selected Papers of "International Joint Conference on Automated Reasoning 2006", 2008.

International Peer-Reviewed Conference/Proceedings

[6] P. ARRIGHI, G. DOWEK. *Linear-algebraic lambda-calculus: higher-order, encodings and confluence*, in "Rewriting techniques and applications, RTA 08", A. VORONKOV (editor), Lecture Notes in Computer Science, Springer-Verlag, 2008.

[7] B. BARRAS, B. BERNARDO. *The Implicit Calculus of Constructions as a Programming Language with Dependent Types*, in "FoSSaCS", R. M. AMADIO (editor), Lecture Notes in Computer Science, vol. 4962, Springer, 2008, p. 365-379, http://dx.doi.org/10.1007/978-3-540-78499-9_26.

[8] F. BLANQUI, J.-P. JOUANNAUD, P.-Y. STRUB. *From Formal Proofs to Mathematical Proofs: A safe, incremental way for building in first-order decision procedures*, in "5th IFIP International Conference on Theoretical Computer Science", 2008.

[9] G. FAURE, R. NIEUWENHUIS, A. OLIVERAS, E. RODRÍGUEZ-CARBONELL. *SAT Modulo the Theory of Linear Arithmetic: Exact, Inexact and Commercial Solvers*, in "SAT", H. K. BÜNING, X. ZHAO (editors), Lecture Notes in Computer Science, vol. 4996, Springer, 2008, p. 77-90, http://dx.doi.org/10.1007/978-3-540-79719-7_8.

[10] H. HERBELIN, S. GHILEZAN. *An Approach to Call-by-Name Delimited Continuations*, in "Proceedings of the 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2008, San Francisco, California, USA, January 7-12, 2008", G. C. NECULA, P. WADLER (editors), ACM, January 2008, p. 383-394, http://doi.acm.org/10.1145/1328438.1328484.

[11] S. LEBRESNE. *A System F with call-by-name exceptions*, in "ICALP 2008, part II", L. ACETO (editor), LNCS, vol. 5126, Springer Verlag, 2008, p. 323-335.

Scientific Books (or Scientific Book chapters)

[12] R. M. AMADIO (editor). *Foundations of Software Science and Computational Structures, 11th International Conference, FOSSACS 2008, Held as Part of ETAPS 2008, Budapest, Hungary, March 29 - April 6, 2008. Proceedings*, Lecture Notes in Computer Science, vol. 4962, Springer, 2008.

[13] H. K. BÜNING, X. ZHAO (editors). *Theory and Applications of Satisfiability Testing - SAT 2008, 11th International Conference, SAT 2008, Guangzhou, China, May 12-15, 2008. Proceedings*, Lecture Notes in Computer Science, vol. 4996, Springer, 2008.

[14] M. MICULAN, I. SCAGNETTO, F. HONSELL (editors). *Types for Proofs and Programs, International Conference, TYPES 2007, Cividale des Friuli, Italy, May 2-5, 2007, Revised Selected Papers*, Lecture Notes in Computer Science, vol. 4941, Springer, 2008.

Research Reports

[15] G. GONTHIER, A. MAHBOUBI. *A Small Scale Reflection Extension for the Coq system*, INRIA Technical Report, Inria Microsoft Research Joint Centre, 2008, http://hal.inria.fr/inria-00258384/fr/.