IN PARTNERSHIP WITH:
CNRS

Université Denis Diderot (Paris 7)

Activity Report 2013

Project-Team PI.R2

Design, study and implementation of languages for proofs and programs

IN COLLABORATION WITH: Laboratoire Preuves, Programmes et Systèmes

RESEARCH CENTER
Paris - Rocquencourt

THEME
Proofs and Verification

Table of contents

Project-Team PI.R2

Keywords: Programming Languages, Interactive Theorem Proving, Type Systems, Proofs Of Programs, Proof Theory

Creation of the Team: 2009 January 01*, updated into Project-Team:* 2011 January 01.

1. Members

Research Scientists
Pierre-Louis Curien [Team leader, CNRS, Senior Researcher, HdR]
Yves Guiraud [Inria, Researcher]
Hugo Herbelin [Inria, Senior Researcher, HdR]
Alexis Saurin [CNRS, Researcher, until Dec 2013]
Matthieu Sozeau [Inria, Researcher]

Faculty Members
Pierre Letouzey [Univ. Paris VII, Associate Professor]
Philippe Malbos [Univ. Lyon I, Associate Professor, en délégation chez Inria]
Yann Régis-Gianas [Univ. Paris VII, Associate Professor]

External Collaborator
Paul-Andre Mellies [CNRS]

PhD Students
Pierre Boutillier [Univ. Paris VII]
Cyrille Chenavier [Univ. Paris VII, FOCAL project, from Sep 2013]
Guillaume Claret [ENS Paris]
Ludovic Patey [ENS Paris]
Lourdes Del Carmen González Huesca [Inria, ANR PARAL-ITP project]
Maxime Lucas [Univ. Paris VII, ENS Ulm, from Sep 2013]
Guillaume Munch [Inria, ANR PIR.2- RECRE project, until Dec 2013]
Pierre-Marie Pédrot [Univ. Paris VII]

Post-Doctoral Fellows
Jaime Gaspar [Inria, Fondation Sciences Mathématiques de Paris, until Sep 2013]
Eric Finster [Inria, ANR PIR.2- RECRE project, from Dec 2013]
Marc Lasson [Inria, from Nov 2013]

Visiting Scientist
Zena Ariola [Inria, Univ. Paris VII, until Jul 2013]

Administrative Assistants
Marine Meyer [Inria]
Lindsay Polienor [Inria]

Other
Luke Maurer [Inria Internship, from April to July 2013]

2. Overall Objectives

2.1. Overall Objectives

The research conducted in πr^2 is devoted both to the study of foundational aspects of formal proofs and programs and to the development of the Coq proof assistant software, with a focus on the dependently typed programming language aspects of Coq. The team acts as one of the strongest teams involved in the development of Coq as it hosts in particular the current coordinator of the Coq development team.

Since 2012, the team has also extended its scope to the study of the homotopy of rewriting systems, which shares foundational tools with recent advanced works on the semantics of type theories.

2.2. Highlights of the Year

The Coq team received the 2013 ACM SIGPLAN Programming Language Software Award, which was presented at POPL'14 in San Diego. To quote: "The Programming Languages Software Award is given by ACM SIGPLAN to an individual or an institution to recognize the development a software system that has had a signicant impact on programming language research, implementations, and tools. The impact may be reflected in the wide-spread adoption of the system or its underlying concepts by the wider programming language community either in research projects, in the open-source community, or commercially."

3. Research Program

3.1. Proof theory and the Curry-Howard correspondence

3.1.1. Proofs as programs

Proof theory is the branch of logic devoted to the study of the structure of proofs. An essential contributor to this field is Gentzen [40] who developed in 1935 two logical formalisms that are now central to the study of proofs. These are the so-called "natural deduction", a syntax that is particularly well-suited to simulate the intuitive notion of reasoning, and the so-called "sequent calculus", a syntax with deep geometric properties that is particularly well-suited for proof automation.

Proof theory gained a remarkable importance in computer science when it became clear, after genuine observations first by Curry in 1958 [34], then by Howard and de Bruijn at the end of the 60's [45], [61], that proofs had the very same structure as programs: for instance, natural deduction proofs can be identified as typed programs of the ideal programming language known as λ-calculus.

This proofs-as-programs correspondence has been the starting point to a large spectrum of researches and results contributing to deeply connect logic and computer science. In particular, it is from this line of work that Coquand's Calculus of Constructions [30] stemmed out – a formalism that is both a logic and a programming language and that is at the source of the Coq system [58].

3.1.2. Towards the calculus of constructions

The λ-calculus, defined by Church [29], is a remarkably succinct model of computation that is defined via only three constructions (abstraction of a program with respect to one of its parameters, reference to such a parameter, application of a program to an argument) and one reduction rule (substitution of the formal parameter of a program by its effective argument). The λ-calculus, which is Turing-complete, i.e. which has the same expressiveness as a Turing machine (there is for instance an encoding of numbers as functions in λ-calculus), comes with two possible semantics referred to as call-by-name and call-by-value evaluations. Of these two semantics, the first one, which is the simplest to characterise, has been deeply studied in the last decades [26].

For explaining the Curry-Howard correspondence, it is important to distinguish between intuitionistic and classical logic: following Brouwer at the beginning of the 20^{th} century, classical logic is a logic that accepts the use of reasoning by contradiction while intuitionistic logic proscribes it. Then, Howard's observation is that the proofs of the intuitionistic natural deduction formalism exactly coincide with programs in the (simply typed) λ-calculus.

A major achievement has been accomplished by Martin-Löf who designed in 1971 a formalism, referred to as modern type theory, that was both a logical system and a (typed) programming language [51].

In 1985, Coquand and Huet [30], [31] in the Formel team of Inria-Rocquencourt explored an alternative approach based on Girard-Reynolds' system F [41], [56]. This formalism, called the Calculus of Constructions, served as logical foundation of the first implementation of Coq in 1984. Coq was called CoC at this time.

3.1.3. The Calculus of Inductive Constructions

The first public release of CoC dates back to 1989. The same project-team developed the programming language Caml (nowadays coordinated by the Gallium team) that provided the expressive and powerful concept of algebraic data types (a paragon of it being the type of list). In CoC, it was possible to simulate algebraic data types, but only through a not-so-natural not-so-convenient encoding.

In 1989, Coquand and Paulin [32] designed an extension of the Calculus of Constructions with a generalisation of algebraic types called inductive types, leading to the Calculus of Inductive Constructions (CIC) that started to serve as a new foundation for the Coq system. This new system, which got its current definitive name Coq, was released in 1991.

In practice, the Calculus of Inductive Constructions derives its strength from being both a logic powerful enough to formalise all common mathematics (as set theory is) and an expressive richly-typed functional programming language (like ML but with a richer type system, no effects and no non-terminating functions).

3.2. The development of Coq

Since 1984, about 40 persons have contributed to the development of Coq, out of which 7 persons have contributed to bring the system to the place it is now. First Thierry Coquand through his foundational theoretical ideas, then Gérard Huet who developed the first prototypes with Thierry Coquand and who headed the Coq group until 1998, then Christine Paulin who was the main actor of the system based on the CIC and who headed the development group from 1998 to 2006. On the programming side, important steps were made by Chet Murthy who raised Coq from the prototypical state to a reasonably scalable system, Jean-Christophe Filliâtre who turned to concrete the concept of a small trustful certification kernel on which an arbitrary large system can be set up, Bruno Barras and Hugo Herbelin who, among other extensions, reorganised Coq on a new smoother and more uniform basis able to support a new round of extensions for the next decade.

The development started from the Formel team at Rocquencourt but, after Christine Paulin got a position in Lyon, it spread to École Normale Supérieure de Lyon. Then, the task force there globally moved to the University of Orsay when Christine Paulin got a new position there. On the Rocquencourt side, the part of Formel involved in ML moved to the Cristal team (now Gallium) and Formel got renamed into Coq. Gérard Huet left the team and Christine Paulin started to head a Coq team bilocalised at Rocquencourt and Orsay. Gilles Dowek became the head of the team which was renamed into LogiCal. Following Gilles Dowek who got a position at École Polytechnique, LogiCal moved to the new Inria Saclay research center. It then split again, giving birth to ProVal. At the same time, the Marelle team (formerly Lemme, formerly Croap) which has been a long partner of the Formel team, invested more and more energy in both the formalisation of mathematics in Coq and in user interfaces for Coq.

After various other spreadings resulting from where the wind pushed former PhD students, the development of Coq got multi-site with the development now realised by employees of Inria, the CNAM and Paris 7.

We next briefly describe the main components of Coq.

3.2.1. The underlying logic and the verification kernel

The architecture adopts the so-called de Bruijn principle: the well-delimited *kernel* of Coq ensures the correctness of the proofs validated by the system. The kernel is rather stable with modifications tied to the evolution of the underlying Calculus of Inductive Constructions formalism. The kernel includes an interpreter of the programs expressible in the CIC and this interpreter exists in two flavours: a customisable lazy evaluation machine written in OCaml and a call-by-value bytecode interpreter written in C dedicated to efficient computations. The kernel also provides a module system.

3.2.2. Programming and specification languages

The concrete user language of Coq, called *Gallina*, is a high-level language built on top of the CIC. It includes a type inference algorithm, definitions by complex pattern-matching, implicit arguments, mathematical notations and various other high-level language features. This high-level language serves both for the development of programs and for the formalisation of mathematical theories. Coq also provides a large set of commands. Gallina and the commands together forms the *Vernacular* language of Coq.

3.2.3. Libraries

Libraries are written in the vernacular language of Coq. There are libraries for various arithmetical structures and various implementations of numbers (Peano numbers, implementation of $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ with binary digits, implementation of $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ using machine words, axiomatisation of $\mathbb{R}$). There are libraries for lists, list of a specified length, sorts, and for various implementations of finite maps and finite sets. There are libraries on relations, sets, orders.

3.2.4. Tactics

The tactics are the methods available to conduct proofs. This includes the basic inference rules of the CIC, various advanced higher level inference rules and all the automation tactics. Regarding automation, there are tactics for solving systems of equations, for simplifying ring or field expressions, for arbitrary proof search, for semi-decidability of first-order logic and so on. There is also a powerful and popular untyped scripting language for combining tactics into more complex tactics.

Note that all tactics of Coq produce proof certificates that are checked by the kernel of Coq. As a consequence, possible bugs in proof methods do not hinder the confidence in the correctness of the Coq checker. Note also that the CIC being a programming language, tactics can be written (and certified) in the own language of Coq if needed.

3.2.5. Extraction

Extraction is a component of Coq that maps programs (or even computational proofs) of the CIC to functional programs (in OCaml, Scheme or Haskell). Especially, a program certified by Coq can further be extracted to a program of a full-fledged programming language then benefiting of the efficient compilation, linking tools, profiling tools, ... of the target software.

3.3. Dependently typed programming languages

Dependently typed programming (shortly DTP) is an emerging concept referring to the diffuse and broadening tendency to develop programming languages with type systems able to express program properties finer than the usual information of simply belonging to specific data-types. The type systems of dependently-typed programming languages allow to express properties *dependent* of the input and the output of the program (for instance that a sorting program returns a list of same size as its argument). Typical examples of such languages were the Cayenne language, developed in the late 90's at Chalmers University in Sweden and the DML language developed at Boston. Since then, various new tools have been proposed, either as typed programming languages whose types embed equalities (Ωmega at Portland, ATS at Boston, ...) or as hybrid logic/programming frameworks (Agda at Chalmers University, Twelf at Carnegie, Delphin at Yale, OpTT at U. Iowa, Epigram at Nottingham, ...).

DTP contributes to a general movement leading to the fusion between logic and programming. Coq, whose language is both a logic and a programming language which moreover can be extracted to pure ML code plays a role in this movement and some frameworks for DTP have been proposed on top of Coq (Concoqtion at Rice and Colorado, Ynot at Harvard, Why in the ProVal team at Inria). It also connects to Hoare logic, providing frameworks where pre- and post-conditions of programs are tied with the programs.

DTP approached from the programming language side generally benefits of a full-fledged language (e.g. supporting effects) with efficient compilation. DTP approached from the logic side generally benefits of an expressive specification logic and of proof methods so as to certify the specifications. The weakness of the approach from logic however is generally the weak support for effects or partial functions.

3.3.1. Type-checking and proof automation

In between the decidable type systems of conventional data-types based programming languages and the full expressiveness of logically undecidable formulae an active field of research explores a spectrum of decidable or semi-decidable type systems for possible use in dependently programming languages. At the beginning of the spectrum, this includes for instance the system F's extension ML_F of the ML type system or the generalisation of abstract data types with type constraints (G.A.D.T.) such as found in the Haskell programming language. At the other side of the spectrum, one finds arbitrary complex type specification languages (e.g. that a sorting function returns a list of type "sorted list") for which more or less powerful proof automation tools (generally first-order ones) exist.

3.3.2. Libraries

Developing libraries for programming languages takes time and generally benefits of a critical mass effect. An advantage is given to languages that start from well-established existing frameworks for which a large panel of libraries exist. Coq is such a framework.

3.4. Around and beyond the Curry-Howard correspondence

For two decades, the Curry-Howard correspondence was limited to the intuitionistic case but in 1990, an important stimulus spurred on the community following the discovery by Griffin that the correspondence was extensible to classical logic. The community then started to investigate unexplored potential fields of connection between computer science and logic. One of these fields was the computational understanding of Gentzen's sequent calculus while another one was the computational content of the axiom of choice.

3.4.1. Control operators and classical logic

Indeed, a significant extension of the Curry-Howard correspondence has been obtained at the beginning of the 90's thanks to the seminal observation by Griffin [42] that some operators known as control operators were typable by the principle of double negation elimination ($\neg\neg A \Rightarrow A$), a principle which provides classical logic.

Control operators are operators used to jump from one place of a program to another place. They were first considered in the 60's by Landin [50] and Reynolds [55] and started to be studied in an abstract way in the 80's by Felleisen *et al* [36], culminating in Parigot's $\lambda\mu$-calculus [54], a reference calculus that is in fine Curry-Howard correspondence with classical natural deduction. In this respect, control operators are fundamental pieces of the full connection between proofs and programs.

3.4.2. Sequent calculus

The Curry-Howard interpretation of sequent calculus started to be investigated at the beginning of the 90's. The main technicality of sequent calculus is the presence of *left introduction* inference rules, for which two kinds of interpretations of these rules are applicable. The first approach interprets left introduction rules as construction rules for a language of patterns but it does not really address the problem of the interpretation of the implication connective. The second approach, started in 1994, interprets left introduction rules as evaluation context formation rule. This line of work culminated in 2000 with the design by Hugo Herbelin and Pierre-Louis Curien of a symmetric calculus exhibiting deep dualities between the notion of programs and evaluation contexts and between the standard notions of call-by-name and call-by-value evaluation semantics.

3.4.3. Abstract machines

Abstract machines came as an intermediate evaluation device, between high-level programming languages and the computer microprocessor. The typical reference for call-by-value evaluation of λ-calculus is Landin's SECD machine [49] and Krivine's abstract machine for call-by-name evaluation [47], [46]. A typical abstract machine manipulates a state that consists of a program in some environment of bindings and some evaluation context traditionally encoded into a "stack".

3.4.4. Delimited control

Delimited control extends the expressiveness of control operators with effects: the fundamental result here is a completeness result by Filinski [37]: any side-effect expressible in monadic style (and this covers references, exceptions, states, dynamic bindings, ...) can be simulated in λ-calculus equipped with delimited control.

4. Software and Platforms

4.1. COQ (http://coq.inria.fr)

Participants: Bruno Barras [Inria Saclay], Yves Bertot [Marelle team, Sophia], Pierre Boutillier, Xavier Clerc [SED team], Pierre Courtieu [CNAM], Maxime Dénès [Marelle team, Sophia], Julien Forest [CNAM], Stéphane Glondu [CARAMEL team, Nancy Grand Est], Benjamin Grégoire [Marelle team, Sophia], Vincent Gross [Consultant at NBS Systems], Hugo Herbelin [correspondant], Pierre Letouzey, Assia Mahboubi [SpecFun team, Saclay], Julien Narboux [University of Strasbourg], Jean-Marc Notin [Ecole Polytechnique], Christine Paulin [Proval team, Saclay], Pierre-Marie Pédrot, Loïc Pottier [Marelle team, Sophia], Matthias Puech, Yann Régis-Gianas, François Ripault, Matthieu Sozeau, Arnaud Spiwack [Abstraction team, ENS], Pierre-Yves Strub [IMDEA, Madrid], Enrico Tassi [SpecFun team, Saclay], Benjamin Werner [Ecole Polytechnique].

4.1.1. Version 8.5

Version 8.5 is expected to be released after the summer of 2014. It will be a major release of the Coq proof assistant, including 6 major new features:

- Parallel development and compilation, inside files and across files, by Enrico Tassi (Inria SpecFun), a result of the Paral-ITP ANR project.
- New proof engine of Arnaud Spiwack (formerly pi.r2 postdoc), more expressive and with clearer semantics.
- Native compilation by Maxime Dénès and Benjamin Grégoire (Inria Marelle, M. Dénès is now at the University of Pennsylvania). A compilation scheme from Coq to OCaml to native code, considerably improving on the previous virtual machine implementation by B. Grégoire.
- A Universe Polymorphic extension by Matthieu Sozeau that allows universe-generic developments, as required by the Homotopy Type Theory library for example.
- Primitive projections for records by Matthieu Sozeau.
- A new document generation system by F. Ripault and Yann Régis-Gianas.

A more detailed description of all the new features will be made in next year's report, but some elements can already be found below.

4.1.2. Evaluation algorithms

Pierre Boutillier has worked on the practical implementation of the unfolding algorithm for global constants that he proposed last year, so that it could become the default process to simplify terms by tactics. A formal presentation has been written in his PhD, to be defended in February 2014.

4.1.3. Internal representation of projections

The change of representation for record projections implemented by Matthieu Sozeau will be part of the 8.5 release. It provides not only exponential gains in performance (type-checking and comparison time and space usage) but also a better basis to work with canonical structures in the unification algorithm, allowing to improve for example the ssreflect inference mechanism significantly. Benchmarks on the HoTT library and a groupoid model construction confirm the exponential gain in performance.

4.1.4. Universes

Matthieu Sozeau followed up his work on universe polymorphism and uncovered important theoretical problems regarding conversion and unification of universe polymorphic constants in the presence of cumulativity and the Prop $\leq$ Type rule. After a careful study of the alternative solutions, he designed a practical correction for the issue and developed a paper proof of conservativity of the complete new system over the original theory of Coq. A paper describing this work has been submitted. The universe polymorphic system, already in use by the HoTT community, will be part of the upcoming 8.5 release.

4.1.5. The Equations plugin

Matthieu Sozeau continued work on the Equations plugin and fixed the remaining bugs preventing full automation of a middle-size example of formalization of the normalization proof of a simply-typed lambda calculus. Wojciech Jedynak, a student of Dariusz Biernacki and Małgorzata Biernacka at the University of Wroclaw, is working under his supervision to do the remaining work before an official release of the plugin can be done. Wojciech Jedynak applied for a 4-month internship supervised by Matthieu Sozeau on an extension of Equations, to start in March 2014.

4.1.6. Internal architecture of the Coq software

With the help of many others, Pierre Letouzey organized in November 2013 the migration of the official Coq source repository from subversion to git. The native use of this decentralized version control system eases the exchange of code amongst Coq developpers (either from the Coq dev team or from external contributors).

Pierre Letouzey, Pierre-Marie Pédrot and Xavier Clerc have continued to work at improving the quality of the OCaml code which composes Coq :

- Many modules have been revised, in particular with cleaner naming convention.
- Almost all uses of the generic OCaml comparison has been chased and transformed into specific code. This avoided many potential bugs with advanced structures, while improving performances at the same time.
- The codes handling OCaml exceptions have been reworked to avoid undue interceptions of critical exceptions.
- Issues involving exceptions are now quite simpler to debug, thanks to easy-to-obtain backtraces.

4.1.7. Efficiency

Pierre-Marie Pédrot has been working on the overall optimization of Coq, by tracking hotspots in the code. Coq trunk is currently much more efficient than its v8.4 counterpart, and is about as quick as v8.3, while having been expanded with a lot of additional features.

Pierre Letouzey has improved the representation of Coq binary files : these files are now smaller (thanks to more sharing), and are reloaded quickly by Coq (thanks to deferred loading of opaque proof terms, which are large and almost never accessed by the user).

4.1.8. Documentation generation

François Ripault and Yann Régis-Gianas developed a new version of coqdoc, the documentation generator of Coq. This new implementation is based on the interaction protocol with the Coq system and should be more robust with respect to the evolution of Coq.

4.1.9. General maintenance

Pierre Letouzey has been the main maintainer of Coq with extra contributions from Hugo Herbelin, Pierre Boutillier, Matthieu Sozeau, Pierre-Marie Pédrot, ...

4.1.10. Modules in Coq

In 2013, Pierre Letouzey has proposed an important rework of the code implementing the module system of Coq. This code was inherited from the successive works of several PhD students, and was in pretty poor shape. While being equivalent in terms of features for the user, the new code should be quite more readable and robust, as well as more efficient: the memory sharing of modular structures should be better, leading to reduced memory footprint for Coq as well as smaller Coq compiled files.

4.1.11. The Coq extraction

Pierre Letouzey has collaborated with colleagues with the aim of extending the extraction tool to additional target languages:

- `C++` with Gabriel Dos Reis and his student Robert Schumacher
- `F#` with David Monniaux

These experiments have been quite promising. In the case of `C++`, an article has been written, it should be re-submitted soon for publication.

4.1.12. Formalisation in Coq

Hugo Herbelin's type-theoretic construction of semi-simplicial sets [22] has been formalised in Coq.

Matthieu Sozeau and Nicolas Tabareau formalised a groupoid model in Coq http://github.com/mattam82/groupoid.

Jaime Gaspar has verified in Coq the correctness of Jean-Louis Krivine's proof that Zermelo-Fraenkel set theory without choice ZF is contained in a variant ZF_ε (useful for Krivine's classical realisability).

4.2. Other software developments

In collaboration with François Pottier (Inria Gallium), Yann Régis-Gianas maintained Menhir, an LR parser generator for OCaml.

Yann Régis-Gianas started the development of the "Hacking Dojo", a web platform to automatically grade programming exercises.

In a different vein, Jaime Gaspar showed that is not always possible to get cross-references right in LaTeX by presenting a LaTeX file where a cross-reference is always wrong (no matter how many times we compile the file).

5. New Results

5.1. Proof-theoretical and effectful investigations

Participants: Pierre Boutillier, Guillaume Claret, Pierre-Louis Curien, Yann-Régis Gianas, Hugo Herbelin, Guillaume Munch-Maccagnoni, Ludovic Patey, Pierre-Marie Pédrot, Alexis Saurin.

5.1.1. Sequent calculus and computational duality

Categorical semantics.

During his collaboration with Marcelo Fiore and Pierre-Louis Curien, Guillaume Munch-Maccagnoni characterised the polarised evaluation order through a categorical structure where the hypothesis that composition is associative is relaxed. Duploid is the name of the structure, as a reference to Jean-Louis Loday's duplicial algebras. The main result, in the lineage of Führmann's [38] direct-style characterisation of monadic models, is a reflection Adj→Dupl where Dupl is a category of duploids and duploid functors, and Adj is the category of adjunctions and pseudo maps of adjunctions. The result suggests that the various biases in denotational semantics: indirect, call-by-value, call-by-name... are a way of hiding the fact that composition is not always associative. This work was accepted for publication in FoSSaCs 2014 [53].

Pierre-Louis Curien, in connection with his increasing interests in operads and algebraic structures of various kinds, found out that the core syntax of system L (underlying the duality of computation) could be used with profit to describe the wiring structures underlying operads, dioperads, cyclic operads, and more generally Lamarche's structads [48]. He also showed a syntactic equivalence between Munch-Maccagnoni's (pre)duploids and system L syntax. These results were presented in his invited talks at the Loday's Mathematical Legacy workshop in Strasbourg and at the workshop Algebra and Computation in Lyon, in January 2014.

Duality of construction.

Paul Downen and Zena Ariola developed a generalized theory of the sequent calculus for understanding the concepts of evaluation strategy and of data (for example, pairs in ML) and co-data (for example, functions) in programming languages. This theory provides a single framework for user-defined data and co-data types as well as a generalized treatment of evaluation strategies, including call-by-value, call-by-name, and call-by-need, that are given as parameters to the theory. In the end, the framework encompasses the previously known duality of call-by-name and call-by-value in the sequent calculus, both by Curien and Herbelin [3] and Wadler [59] , while also including call-by-need and its dual. Additionally, the framework reveals connections with approaches by Zeilberger [60], Munch-Maccagnoni [6], and Curien and Munch-Maccagnoni [33], for using polarization and focalization to provide deterministic strategies for classical computation with structures and pattern matching. This work will be presented at ESOP 2014 [15].

Luke Maurer and Zena Ariola in collaboration with Daniele Varacca studied the connections between π-calculus encodings of the λ-calculus and similar continuation-passing style (CPS) transformations, extending the connections for call-by-value and call-by-name encodings to include the call-by-need π-calculus encoding as well. This development revealed a better understanding of the computational effect needed in the λ-calculus to model call-by-need evaluation, which better reflects the way that memoization for call-by-need is implemented. The work is going to be submitted to RTA-TLCA.

Constructive interpretation of an involutive negation.

Guillaume Munch-Maccagnoni developped a syntax of delimited control operators that exposes a formulae-as-types correspondence between an involutive negation in classical natural deduction, and the idea that captured contexts, unlike continuations, can be inspected. This decomposes technical artefacts found in call-by-name classical realisability, and simplifies witness extraction from proofs of Σ formulae. This work has been submitted and appears in his PhD thesis [11].

5.1.2. Dependent monads

Guillaume Claret and Yann Régis-Gianas are developping a monadic translation from functional programs with effects to Coq that uses a dependent monad. The aim of this work is to allow to reason about effectful programs directly in Coq.

5.1.3. Linear dependent types

Pierre-Marie Pédrot developped a dependent version of the Dialectica translation, that gives interesting insights into the possibility to design linear dependent types. Indeed, Dialectica can be decomposed as a translation acting on linear types instead of intuitionistic ones.

5.1.4. Delimited continuations, polarity and computational effects

Guillaume Munch-Maccagnoni's polarised decomposition of delimited control calculi appeared in his PhD thesis [11].

5.1.5. Reverse mathematics

Ludovic Patey studied with Laurent Bienvenu and Paul Shafer the deep connections between algorithmic randomness and reverse mathematics by defining formally the ability of computing a solution to a problem by probabilistic means within the framework of reverse mathematics, the No Randomized Algorithm property (NRA). They provided a classification of the whole revese mathematics zoo created by Damir Dzhafarov in terms of having the NRA property or not, answering to some open separation questions.

Ludovic Patey stated two dichotomy theorems about satisfiability problems within reverse mathematics and proved them using clone theory. The corresponding paper is submitted to Computability in Europe 2014. He studied also ramseyan theorems related to the Rainbow Ramsey theorem and provided characterizations in terms of diagonally non-computable functions, algorithmic randomness, and related it to the Erdös Moser theorem and Thin set theorem.

5.1.6. Gödel's functional interpretation

Pierre-Marie Pédrot showed that the Dialectica translation could be explained in terms of the Krivine abstract machine, in a way similar to the usual presentation of classical realizability. This opens the door to a better understanding of related translations, as well as adding semi-classical effect into PTS.

5.1.7. Logical foundations of call-by-need evaluation

Alexis Saurin and Pierre-Marie Pédrot developed a structured reconstruction of call-by-need based on linear head reduction which arose in the context of linear logic. This opens new directions both to extend call-by-need to control and to apply linear logic proof-theory (and particularly proof-nets) to call-by-need evaluation.

5.1.8. Streams and classical logic

Alexis Saurin and Fanny He have been working on transfinite term rewriting in order to model stream calculi and their connections with lambda-calculi for classical logic.

Jaime Gaspar identified the eight simplest variants (some already known) of the Kuroda negative translation that translate classical logic into minimal logic.

5.2. Type theory and the foundations of Coq

Participants: Pierre Boutillier, Pierre-Louis Curien, Hugo Herbelin, Pierre-Marie Pédrot, Yann Régis-Gianas, Alexis Saurin, Matthieu Sozeau.

5.2.1. Substitutions and isomorphisms

Pierre-Louis Curien completed his joint work with Richard Garner and Martin Hofmann on relating syntax unstrictification through coercions with model strictification (cf. πr^2 report 2012), adding a careful treatment of identity types. The corresponding paper was accepted for publication in the TCS special issue for Glynn Winskel's anniversary.

5.2.2. Homotopy type theory

Hugo Herbelin, Matthieu Sozeau and Pierre-Louis Curien participated to the univalent foundations program. A collaborative book [18] on the results of this program has been published.

5.2.3. Models of type theory

Simplicial sets and their extensions as Kan complexes can serve as models of homotopy type theory. Hugo Herbelin developed a concrete type-theoretic formalisation of semi-simplicial sets following ideas from Steve Awodey, Peter LeFanu Lumsdaine and other researchers both at Carnegie-Mellon University and at the Institute of Advanced Study. This has been accepted for publication in a special issue of MSCS on homotopy type theory [22].

The technique he used generalises to provide type-theoretic constructions for arbitrary presheaves on Reedy categories, thus including simplicial sets. In particular, this provides with a formulation of simplicial sets where degeneracies are decidable, which is not the case with the definition as a presheaf.

Hugo Herbelin also investigated hybrid constructive definitions of simplicial sets where face maps are axiomatised but degeneracies are built. Again, this provides with a formulation of simplicial sets where it is decidable whether a given simplex is degenerate or not.

5.2.4. Internalizing the setoid model of type theory

As an example use of the new polymorphic universe extension of Coq, Matthieu Sozeau developed together with Nicolas Tabareau (Inria Ascola team, École des mines Nantes) a complete groupoid model of type theory, following the seminal work of Hofmann and Streicher. A preliminary paper presenting a partial generalization of this model to 2-goupoids was written and will be resubmitted [23].

A completed version of this model has since been formalized and will be submitted shortly. This model showcases the use of the polymorphic universes: in the course of its formalization we uncovered hidden assumptions in the interpretation of substitution and sigma types in the original presentation thanks to the universe system.

5.2.5. Proof irrelevance, eta-rules

Matthieu Sozeau finished his implementation of a proof-irrelevant system but did not publish it. Indeed, the homotopy type theory interpretation suggests new ways to introduce proof-irrelevance using bracket types that seem to significantly depart from the syntactic treatment developped by Werner and himself. An investigation of the relationship between the presentation of the calculus of inductive constructions given by Hugo Herbelin and Arnaud Spiwack in [44] which includes the bracket construction and the aforementioned syntactic version will be part of a master's internship supervised by Matthieu Sozeau in 2014.

5.3. Homotopy of rewriting systems

Participants: Cyrille Chenavier, Pierre-Louis Curien, Yves Guiraud, Maxime Lucas, Philippe Malbos.

5.3.1. The homotopical completion-reduction procedure

In [39], Stéphane Gaussent (Institut Camille Jordan), Yves Guiraud and Philippe Malbos have introduced the homotopical completion-reduction procedure as a higher-dimensional rewriting method to compute coherent presentations of monoids. The results of this procedure on Artin monoids of spherical type have been implemented by Yves Guiraud in a Python library, available on his webpage. The procedure is currently improved towards the explicit computation of full polygraphic resolutions of Artin monoids to provide a purely algebraic and constructive account of well-known geometric objects, such as Caylay graphs and Salvetti complexes.

In [16], Yves Guiraud, Philippe Malbos and Samuel Mimram (CEA Saclay) have further investigated the homotopical completion-reduction procedure, extended with the adjunction/elimination of redundant generators, with successful application to two new classes of monoids: the plactic and the Chinese monoids. This work has been implemented by Samuel Mimram and Yves Guiraud into a prototype, that can be tested at http://www.pps.univ-paris-diderot.fr/~smimram/rewr, and has been presented to RTA 2013 by Philippe Malbos, where it has received the best paper award.

5.3.2. New methods for the computation of coherent presentations

During his M2 internship, Maxime Lucas, supervised by Yves Guiraud, has improved the rewriting method used in [43] for the computation of homotopy bases of monoids and categories. This allows a more effective computation in several cases, based on the notion of Anick chain [25] instead of the broader notion of critical branching. Maxime Lucas has now started a PhD thesis, supervised by Yves Guiraud and Pierre-Louis Curien, and currently investigates the use of Garside-like structures [35] to further improve the computation of coherent presentations for higher-dimensional categories.

5.3.3. Higher-dimensional linear rewriting

Cyrille Chenavier, Pierre-Louis Curien, Yves Guiraud and Philippe Malbos investigate with Eric Hoffbeck (LAGA, Université Paris 13) and Samuel Mimram (CEA Saclay) the links between set-theoretic rewriting theory and the computational methods known in symbolic algebra, such as Gröbner bases [28]. This interaction is supported by the Focal project of the IDEX Sorbonne Paris Cité. Yves Guiraud, Eric Hoffbeck and Philippe Malbos are currently working on an improvement, based on the homotopical completion-reduction procedure, of the methods known in algebra to compute homological invariants of algebras and operads. Cyrille Chenavier has started a PhD thesis, supervised by Yves Guiraud and Philippe Malbos, to use Berger's theory of reduction operators [27] to design new methods for the study of rewriting systems.

5.3.4. Homotopical and homological finiteness conditions

Yves Guiraud and Philippe Malbos have written a comprehensive introduction [21] on the links between higher-dimensional rewriting, the homotopical finiteness condition "finite derivation type" and the homological finiteness condition "FP_3", from the point of view of higher categories and polygraphs. The purpose of this work is to provide an introduction to the field, formulated in a contemporary language, and with new, more formal proofs of classical results.

In [19], Yves Guiraud and Philippe Malbos have introduced a notion of identities among relations for higher categories presented by polygraphs. This notion is well-known in combinatorial group theory, where it is linked to the explicit computation of homological invariants and of formal representations of groups as crossed complexes. The main result of [19] is a procedure based on higher rewriting to compute generators of the identities among relations. They have related the facts that the natural system of identities among relations is finitely generated and that the higher category has finite derivation type (a homotopical finiteness condition introduced in [43] for higher categories after Squier's work for monoids [57]).

5.4. Coq as a functional programming language

Participants: Pierre Boutillier, Guillaume Claret, Lourdes Del Carmen González Huesca, Hugo Herbelin, Pierre Letouzey, Matthias Puech, Yann Régis-Gianas, Matthieu Sozeau.

5.4.1. Type classes and libraries

Type Classes are heavily used in the HoTT/Coq library (http://github.com/HoTT/coq) developed by the Univalent Foundations program at the IAS, to which Matthieu Sozeau participated.

5.4.2. Dependent pattern-matching

How to encode structurally dependent pattern matching into case analysis by hand has been described by Jean François Monin in [52]. Pierre Boutillier, with the help of Thomas Braibant (GALLIUM team), has mechanized this process and exhibited a missing part to make it scale. These are the main results presented in Pierre Boutillier's forthcoming thesis.

5.4.3. Incrementality in proof languages

Lourdes González and Yann Régis-Gianas studied incremental computing and self-adjusting computation [24] as a starting point to develop an applicative notion of change over data structures, to be applied to lambda-terms. They formalized in Coq a notion of derivative of an inductive function to define how to compute a new result from an input that has changed, this is done by using the derivative of the function and the difference on inputs and old outputs. They are working out a technique that allows a specification of functions using derivatives and old inputs and outputs including a cost analysis of the benefits of reusing previous computations.

5.4.4. Lightweight proof-by-reflection

In collaboration with Beta Ziliani (MPI), In the context of the ANR project Paral-ITP, Lourdes del Carmen González Huesca, Guillaume Claret and Yann Régis-Gianas developed a new technique for proof-by-reflection based on a notion of *a posteriori* simulation of effectful computations in Coq. This work has been presented at ITP 2013 ([14]).

6. Partnerships and Cooperations

6.1. National Initiatives

Pierre-Louis Curien (coordinator), Yves Guiraud and Philippe Malbos are members of the three-years Focal project of the IDEX Sorbonne-Paris-Cité, started in June 2013. This project, giving the support for the PhD grant of Cyrille Chenavier, concerns the interactions between higher-dimensional rewriting and combinatorial algebra with researchers from LAGA (Univ. Paris 13)

Pierre-Louis Curien (coordinator), Yves Guiraud and Philippe Malbos are members of the four-years Cathre ANR project, accepted in 2013, to begin in January 2014. This project will investigate the general theory of higher-dimensional rewriting, the development of a general-purpose library for higher-dimensional rewriting, and applications in the fields of combinatorial algebra, combinatorial group theory and theoretical computer science.

Matthieu Sozeau, Hugo Herbelin, Lourdes del Carmen González Huesca and Yann Régis-Gianas are members of the ANR Paral-ITP started November 2011. Paral-ITP is about preparing the Coq and Isabelle interactive theorem provers to a new generation of user interfaces thanks to massive parallelism and incremental type-checking.

Hugo Herbelin is the coordinator of the PPS site for the ANR Récré accepted in 2011, which started in January 2012. Récré is about realisability and rewriting, with applications to proving with side-effects and concurrency.

Matthieu Sozeau is member of the ANR Typex project (Types and certification for XML) and is coordinator of one of the tasks of the project on formalisation and certification of XML tools. The project kicked-off on January 8th, 2012 and is a joint project with LRI, PPS and Inria Grenoble.

6.2. European Initiatives

6.2.1. FP7 Projects

Yann Régis-Gianas is a participant of the EU-FP7 Certified Complexity project (CerCo). This European project started in February 2010 as a collaboration between Bologna university (Asperti, Sacerdoti Coen), Edinburgh university (Pollack) and Paris Diderot university (Amadio, Régis-Gianas). The CerCo project aims at the construction of a formally verified complexity preserving compiler from a large subset of the C programming language to some typical micro-controller assembly language, of the kind traditionally used in embedded systems.

6.2.2. Collaborations in European Programs, except FP7

Pierre-Louis Curien, Yves Guiraud and Philippe Malbos are collaborators of the Applied and Computational Algebraic Topology (ACAT) networking programme of the European Science Foundation.

6.3. International Initiatives

6.3.1. Inria Associate Teams

Title: Proof theory and functional programming languages (SEMACODE)

Inria principal investigator: Alexis SAURIN

International Partner:

Institution: University of Oregon (United States)

Laboratory: Computer and Information Science Department

Researcher: Zena ARIOLA

International Partner:

Institution: University of Novi Sad

Laboratory: Faculty of Engineering

Researcher: Silvia GHILEZAN

Duration: 2011 - 2013

See: http://www.pps.univ-paris-diderot.fr/~saurin/EA-SEMACODE

6.3.2. Inria International Partners

We are setting up a partnershiop with the University of Wroclaw (our interlocutors are D. Biernacki and M. Biernacka).

6.3.3. Participation In other International Programs

Pierre-Louis Curien participates to the ANR International French-Chinese project LOCALI (coordinated by Gilles Dowek), and to a MathAmSud project in algebraic operads with the university of Talca (Chile).

6.3.4. Other international cooperations

MIT (Adam Chlipala, Jason Gross).

6.4. International Research Visitors

6.4.1. Visits of International Scientists

Beta Ziliani (MPI, Saarbrücken) visited πr^2 and PPS for a week in January to collaborate with Yann Régis-Gianas and Matthieu Sozeau.

Zena Ariola visited πr^2 and PPS for the whole academic year 2012-2013 with SEMACODE associate team to collaborate with Pierre-Louis Curien, Hugo Herbelin and Alexis Saurin. Her two PhD students joined for shorter terms (Paul Downen from November 2012 to July 2013 and Luke Maurer, being funded by the INTERNSHIP program – see below – from March 2013 to July 2013).

Marco Gaboardi visited πr^2 and PPS in for 10 days in may and again in december 2013 to collaborate with Alexis Saurin.

Olivier Danvy visited πr^2 and PPS in the fall 2013.

Fernando Ferreira (Univ. of Lisbon) and Ulrich Kohlenbach visited πr^2, hosted by Jaime Gaspar.

6.4.2. Internships

Participant: Luke Maurer.

Subject: Foundation for lazy languages

Date: from Mar 2013 until Jul 2013

Institution: University of Oregon (United States)

6.4.3. Visits to International Teams

Pierre Boutillier visited MSP group at the university of Strathclyde for a month in March 2013.

Hugo Herbelin visited Thomas Streicher at the University of Darmstadt in May 2013.

Hugo Herbelin visited the Institute of Cybernetics in Tallinn, Estonia in September and October 2013.

Pierre-Louis Curien visited IAS for 3 weeks in March, towards the end of the Special Year on Univalent Foundations. Matthieu Sozeau visited Vladimir Voevodsky at the IAS in Princeton for 15 days in May 2013.

7. Dissemination

7.1. Scientific Animation

7.1.1. Collective responsibilities

Pierre-Louis Curien is member of the Conseil Scientifique of the INSII (CNRS). He is also a member of the Conseil Scientifique of CIRM (since June 2013).

7.1.2. Editorial activities

Pierre-Louis Curien is co-editor in chief of Mathematical Structures in Computer Science, and is an editor of Higher-Order and Symbolic Computation.

7.1.3. Program committees and organising committees

Alexis Saurin has been co-chair of the program committee and editor of the proceedings of the workshop *Control Operators and their Semantics* (COS'13, organized at RDP 2013, http://cos2013.di.unito.it and http://rvg.web.cse.unsw.edu.au/eptcs/content.cgi?COS2013). The proceedings were published by EPTCS.

Alexis Saurin has been a member of the scientific committee of the Summer School *Linear Logic and Geometry of Interaction* (http://www.logoi.fr/events/school) which took place in Torino as a satellite event of CSL 2013.

Alexis Saurin is a PC member of GaLoP 2014 (International Workshop on Games and Logic for Programming) which will take place during ETAPS 2014 in Grenoble.

Hugo Herbelin, Pierre Letouzey and Matthieu Sozeau are co-organizing and co-chairing the TYPES'14 conference in Paris in May.

Matthieu Sozeau was in the PC of DTP'13, JFLA'13 and JFLA'14. He is in the newly formed steering committee of the DTP workshop https://www.pps.univ-paris-diderot.fr/dtp/.

Matthieu Sozeau co-organized a (private) meeting at POPL'14 in San Diego, there were 30 participants. He gave a talk on the upcoming 8.5 version of Coq and a tutorial.

Hugo Herbelin organized with Gyesik Lee a workshop on constructive reverse mathematics in Seoul, March 2013.

Yves Guiraud and Philippe Malbos are organisers of the 5-weeks programme Mathematical Structures of Computation, held in Lyon in January-February 2014, and supported by the Labex MILYON. They are also organisers of the second week, Algebra and Computation, while Pierre-Louis Curien and Hugo Herbelin organised the first week, Recent Trends in Type theory.

Pierre-Louis Curien, Hugo Herbelin and Paul-André Melliès are the organisers of the IHP trimester Semantics of proofs and certified mathematics, to be held next spring. They also organise a spring school at CIRM preceding the trimester.

Pierre-Louis Curien is member of the steering committee of the conferences Typed Lambda Calculi (TLCA) and Applications and of the international workshops Games for Logic and Programming Languages (GaLop).

7.1.4. Jury participation

Alexis Saurin was an examinor for the computer science oral at the entrance exam for Ecoles normales supérieures and a member of the jury for Ecole normale supérieure de la rue d'Ulm.

Alexis Saurin has been member of the Jury for LMFI Master.

7.1.5. Invited talks

Hugo Herbelin gave an invited talk on proving Gödel's completeness theorem with side-effects at the conference TLCA'13 in Eindhoven.

Philippe Malbos was invited speaker at the Sao Paulo - Lyon Algebra meeting, in October 2013.

Pierre-Louis Curien was an invited speaker at Glynn Winskel's anniversary workshop in Cambridge (where he presented his joint work with Garner and Hofmann on coherence issues in type theory), at the workshop Higher structures in China (Lanzhou, August), where he presented homotopical completions and reductions, at the International Symposium on Domain Theory, where he presented Thomas Ehrhard's result "Scott is the extensional collapse of Rel", and at the Loday's Mathematical Legacy conference in Strasbourg as well as at the Algebra and Computation workshop in Lyon where he presented his work on languages for operads and related structures.

7.1.6. Presentation of papers

Philippe Malbos has presented [16] at RTA 2013.

Lourdes González presented [14] at TYPES 2013.

Yann Régis-Gianas presented [14] at ITP 2013.

7.1.7. Other presentations

Yves Guiraud gave a talk on "Coherent presentations of Artin monoids" during the Journées PPS in September 2013.

Pierre Boutillier gave a talk on "An abstract machine to conceal strong reduction and fixpoints" during the journées PPS.

Yann Régis-Gianas presented the CerCo european project at two satellite workshops of HIPEAC (Berlin) in March 2013 and ETAPS (Roma) in April 2013.

Hugo Herbelin gave two lectures on reducibility candidates and their relation with completeness proofs to the Réalisabilité à Chambéry #6 workshop in June.

Hugo Herbelin gave a talk on the development of Coq at the Coq workshop '13 in Rennes.

Hugo Herbelin gave a talk at FOMCAF 2013 in Padova on the status of equality in type theory.

Hugo Herbelin gave a talk at TYPES 2013 in Toulouse on the status of equality in type theory.

Hugo Herbelin gave a talk at PCC 2013 in Toulouse on the Computational interpretation for the big five systems of reverse mathematics (joint work with Gyesik Lee and Keiko Nakata).

Matthieu Sozeau gave a talk at TYPES 2013 in Toulouse on universe polymorphism.

Matthieu Sozeau gave a talk at the TYPEX meeting in Paris on universe polymorphism.

Ludovic Patey gave a talk on Probabilistic Algorithms and Ramsey-Type Principles in Reverse Mathematics at the Workshop in Type Theory, in Séoul, on Rainbow Ramsey theorem for pairs at Computability in Europe, Milano, and on Classifying principles by the no randomized algorithm property at Logic Colloquium, Évora.

Jaime Gaspar gave a talk on Krivine's classical realisability and the unprovability of the axiom of choice and the continuum hypothesis at the 6th Young Set Theory Workshop 2013, Oropa, Italy. He gave a talk on "Refuting" Cantor at the Logic Colloquium 2013, Évora, Portugal.

7.1.8. Talks in seminars

Yves Guiraud gave three talks on "Coherent presentations of Artin monoids" at the Groupe de Travail Catégories supérieures, polygraphes et homotopie, PPS, Paris 7, in May-June 2013.

Matthieu Sozeau gave a talk on universe polymorphism at the Coq working group in Paris.

Alexis Saurin gave a talk at "Journées PPS" in september 2013 on linear head reduction and call-by-need, entitled "Éloge de la paresse".

Guillaume Claret gave a talk at Gallium on the Cybele plugin to do proofs by reflection in Coq using the extraction mechanism.

Guillaume Munch-Maccagnoni gave a talk on duploids at the PPS days in September 2013.

Zena Ariola presented a talk on Call-by-need: reduction, continuation passing style and abstract machines at Paris 7 University, 20 December 2012, and at Inria Saclay, 27 February 2013.

Jaime Gaspar gave a talk on Formalising ZF $\subseteq$ ZF$_\varepsilon$ in Paris, and at the Mathematical Logic Seminar, Lisbon. He gave a talk on Proof interpretations: what they are and what they are good for, at Junior Seminar, Rocquencourt, and at Parsifal team seminar, Palaiseau. He gave a talk on Krivine's classical realizability (see also above) at Technical University of Darmstadt, Darmstadt, and at University of Applied Sciences, Wiesbaden.

7.1.9. Attendance to conferences, workshops, schools,...

Yves Guiraud and Philippe Malbos attended RTA 2013, Eindhoven, in June.

Pierre Boutillier, Alexis Saurin and Pierre-Marie Pédrot attended the JFLA 2013 conference in Aussois.

Pierre Boutillier and Lourdes González attended the Coq workshop 2013 in Rennes.

Pierre Boutillier attended ICFP and DTP 2013 in Boston.

Lourdes González attended the Workshop on Formal Meta-Theory organized by the Parsifal team (Lix and Inria) the 5-6th March 2013.

Lourdes González, Guillaume Claret and Pierre-Marie Pédrot attended TYPES 2013 in Toulouse.

Yann Régis-Gianas, Guillaume Claret and Lourdes González attended ITP 2013 in Rennes.

Lourdes González and Pierre-Marie Pédrot attended the Summer School on Linear Logic and Geometry of Interaction in Turin.

Lourdes González attended CSL 2013 and its satellite events in Turin.

Pierre Letouzey attended the Coq workshop 2013 in Rennes.

Matthieu Sozeau attended the JFLA'13 in Aussois.

Matthieu Sozeau attended the Coq workshop 2013 and ITP 2013 in Rennes.

Matthieu Sozeau attended the TYPES 2013 conference in Toulouse.

Matthieu Sozeau attended the POPL'14 conference in San Diego.

Pierre Letouzey, Matthieu Sozeau, Lourdes González, Gullaume Munch and Pierre-Marie Pédrot attended Pierre-Louis Curien's 60th birthday meeting in Venice.

Matthieu Sozeau attended the Conference on Type Theory, Homotopy Theory and Univalent Foundations at the Centre de Recerca Matemàtica in Barcelona.

Guillaume Claret attended the EJCP 2013 school at Saint-Malo and Rennes.

Alexis Saurin attended BLL, a workshop on Bounded Linear Logic, in November 2013.

Zena Ariola attended POPL 2013, Rome.

Jaime Gaspar attended Proof Theory in Lisbon (a one-day workshop), and a workshop on Reverse Mathematics in Paris.

7.1.10. Groupe de travail Théorie des types et réalisabilité

Pierre-Marie Pédrot gave a talk on a classical realizability account of the Dialectica transformation, as well as a dependent version of it, in November.

Matthieu Sozeau gave a talk on a groupoid interpretation in March.

This year's other internal speakers were Hugo Herbelin, Philippe Malbos, and the external speakers were Maribel Fernández (King's College, London), Gregory Malecha (MIT), Marc Lasson (then postdoc at Cambridge Univ.), Keiko Nakata (Institute of Cybernetics, Tallinn), Cyril Cohen (Univ. of Göteborg), Eduardo Bonelli (Universidad Nacional de Quilmes), Barbara Petit (Sardes team, Inria Grenoble), Lionel Rieg (ENS Lyon), Fernando Ferreira (Univ. of Lisbonne), Chantal Keller (Univ. of Aarhus), Matthias Puech (Univ. of Aarhus), Dominic Hughes (Stanford University), Thomas Braibant (Gallium team), and Alexander Kreuzer (ENS Lyon).

7.2. Teaching - Supervision - Juries

7.2.1. Teaching

Licence: Pierre Boutillier has a temporary research and teaching position (A.T.E.R) at University Paris 7 for the academic year 2013–2014. He teaches this year Programmation fonctionnelle (L3) and Analyse lexicale et syntaxique (L3).

Master: Compilation, (Boutillier) 1/4, M1, Université Paris Diderot.

Master: Assistants de Preuve (Sozeau), 9, M2, Université Paris Diderot.

Master: Functional Programming and Type Systems (Regis-Gianas), 12, M2, Université Paris Diderot.

Master: Lambda-calculus (Saurin), 24H, M2 of Mathematics "Logique Mathématique et Fondements de l'Informatique", Université Paris Diderot.

Master Pierre-Louis Curien teaches in the course Models of programming languages: domains, categories, games of the MPRI (together with Thomas Ehrhard and Paul-André Melliès).

7.2.2. Supervision

Internship: Yves Guiraud has supervised the M2 internship of Maxime Lucas.

Internship: Alexis Saurin has supervised the internship of Fanny He.

Internship: Alexis Saurin has supervised (with Claudia Faggian) the M2 internship of Amina Doumane.

PhD in progress: Pierre Boutillier, Représentation des effets et inférence de type dans le cadre du développement d'un langage de programmation à types riches, September 2010, Hugo Herbelin.

PhD in progress: Lourdes del Carmen González Huesca, Un langage de tactiques typées pour Coq, December 2011, Hugo Herbelin and Yann Régis-Gianas.

PhD in progress: Guillaume Claret, Programmation avec effets en Coq, September 2012, Hugo Herbelin and Yann Régis-Gianas.

PhD in progress: Pierre-Marie Pédrot, Logique linéaire et types dépendants, september 2012, supervised by Alexis Saurin and Hugo Herbelin.

PhD in progress : Cyrille Chenavier, supervised by Yves Guiraud and Philippe Malbos.

PhD in progress : Maxime Lucas, supervised by Yves Guiraud and Pierre-Louis Curien.

PhD defended: Guillaume Munch-Maccagnoni, Syntaxe et modèles d'une composition non-associative des programmes et des preuves, Université Paris Diderot - Paris 7, December 10 2013, Pierre-Louis Curien and Thomas Ehrhard. [11]. Hugo Herbelin was a member of the jury.

7.2.3. Juries

Pierre-Louis Curien was referee and Yves Guiraud was jury member for the PhD defence of Pierre Rannou ("Réécriture de diagrammes et de Sigma-diagrammes", Univ. Aix-Marseille).

Pierre-Louis Curien was jury member of the thesis of Florian Hatat ("Jeux graphiques et théorie de la démonstration", Univ. Chambéry).

Pierre-Louis Curien was (as thesis director) in the jury of the theses of Stéphane Zimmermann (“Vers une ludique différentielle”, Univ. Paris Diderot) and of Alexis Goyet (“Le lambda lambda-bar calcul, un calcul dual pour les stratégies non contraintes”, Univ. Paris Diderot).

Pierre Letouzey was a referee for the PhD thesis of Pierre-Nicolas Tollitte in December 2013 (“Extraction de code fonctionnel certifié à partir de spécifications inductives”, CNAM), while Hugo Herbelin was a member of this PhD committee.

7.3. Popularization

Yann Régis-Gianas organised the "Fête de la Science" event for the computer science department of the University Paris 7.

Yann Régis-Gianas co-organised the “Journée Francilienne de Programmation”, a programming contest between undergraduate students of three universities of Paris (UPD, UPMC, UPS).

Yann Régis-Gianas gave several conferences about computer science in several primary schools of Paris.

8. Bibliography

Major publications by the team in recent years

[1] Z. ARIOLA, H. HERBELIN, A. SABRY. *A Type-Theoretic Foundation of Delimited Continuations*, in "Higher Order and Symbolic Computation", 2007, http://dx.doi.org/10.1007/s10990-007-9006-0

[2] P.-L. CURIEN. *Substitution up to isomorphism*, in "Fundamenta Informaticae", 1993, vol. 19, pp. 51-85

[3] P.-L. CURIEN, H. HERBELIN. *The duality of computation*, in "Proceedings of the Fifth ACM SIGPLAN International Conference on Functional Programming (ICFP ’00)", Montreal, Canada, SIGPLAN Notices 35(9), ACM, September 18-21 2000, pp. 233–243 [*DOI :* 10.1145/351240.351262], http://hal.archives-ouvertes.fr/inria-00156377/en/

[4] 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, G. C. NECULA, P. WADLER (editors), ACM, January 7-12 2008, pp. 383-394

[5] H. HERBELIN. *An intuitionistic logic that proves Markov’s principle*, in "Logic In Computer Science", Edinburgh, Royaume-Uni, IEEE Computer Society, 2010, http://hal.inria.fr/inria-00481815/en/

[6] G. MUNCH-MACCAGNONI. *Focalisation and Classical Realisability*, in "Computer Science Logic ’09", E. GRÄDEL, R. KAHLE (editors), Lecture Notes in Computer Science, Springer-Verlag, 2009, vol. 5771, pp. 409–423

[7] Y. RÉGIS-GIANAS, F. POTTIER. *A Hoare Logic for Call-by-Value Functional Programs*, in "Proceedings of the Ninth International Conference on Mathematics of Program Construction (MPC’08)", Lecture Notes in Computer Science, Springer, July 2008, vol. 5133, pp. 305–335, http://gallium.inria.fr/~fpottier/publis/regis-gianas-pottier-hoarefp.ps.gz

[8] A. SAURIN. *Separation with Streams in the $\Lambda\mu$-calculus*, in "Symposium on Logic in Computer Science (LICS 2005)", Chicago, IL, USA, Proceedings, IEEE Computer Society, 26-29 June 2005, pp. 356-365

[9] A. SAURIN. *On the Relations between the Syntactic Theories of λμ-Calculi*, in "17th Annual Conference of the EACSL 17th EACSL Annual Conference on Computer Science Logic - CSL 2008", Bertinoro Italie, Lecture notes in computer science, Springer, 2008, vol. 5213, pp. 154-168 [*DOI :* 10.1007/978-3-540-87531-4_13], http://hal.archives-ouvertes.fr/hal-00527930/en/

[10] M. SOZEAU, N. OURY. *First-Class Type Classes*, in "Theorem Proving in Higher Order Logics, 21st International Conference, TPHOLs 2008, Montreal, Canada, August 18-21, 2008. Proceedings", O. A. MOHAMED, C. MUÑOZ, S. TAHAR (editors), Lecture Notes in Computer Science, Springer, 2008, vol. 5170, pp. 278-293

Publications of the year

Doctoral Dissertations and Habilitation Theses

[11] G. MUNCH-MACCAGNONI. , *Syntaxe et modèles d'une composition non-associative des programmes et des preuves*, Université Paris-Diderot - Paris VII, December 2013, http://hal.inria.fr/tel-00918642

Articles in International Peer-Reviewed Journals

[12] R. M. AMADIO, Y. REGIS-GIANAS. *Certifying and reasoning about cost annotations of functional programs*, in "Higher-Order and Symbolic Computation", January 2013, http://hal.inria.fr/inria-00629473

International Conferences with Proceedings

[13] B. BARRAS, L. D. C. GONZALEZ HUESCA, H. HERBELIN, Y. RÉGIS-GIANAS, E. TASSI, M. WENZEL, B. WOLFF. *Pervasive Parallelism in Highly-Trustable Interactive Theorem Proving Systems*, in "MKM/Calculemus/DML", Bath, United Kingdom, 2013, pp. 359-363, http://hal.inria.fr/hal-00908980

[14] G. CLARET, L. D. C. GONZALEZ HUESCA, Y. RÉGIS-GIANAS, B. ZILIANI. *Lightweight proof by reflection using a posteriori simulation of effectful computation*, in "Interactive Theorem Proving", Rennes, France, July 2013, http://hal.inria.fr/hal-00870110

[15] P. DOWNEN, Z. ARIOLA. *The duality of construction*, in "ESOP 2014 : European Symposium on Programming", Grenoble, France, 2014, 15 p. , http://hal.inria.fr/hal-00938317

[16] Y. GUIRAUD, P. MALBOS, S. MIMRAM. *A Homotopical Completion Procedure with Applications to Coherence of Monoids*, in "RTA - 24th International Conference on Rewriting Techniques and Applications - 2013", Eindhoven, Netherlands, F. VAN RAAMSDONK (editor), Leibniz International Proceedings in Informatics (LIPIcs), Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, June 2013, vol. 21, pp. 223-238 [*DOI :* 10.4230/LIPICS.RTA.2013.223], http://hal.inria.fr/hal-00818253

[17] P.-M. PÉDROT. *Un régime au concentré d'automate*, in "JFLA - Journées francophones des langages applicatifs", Aussois, France, D. POUS, C. TASSON (editors), Damien Pous and Christine Tasson, February 2013, http://hal.inria.fr/hal-00779752

Scientific Books (or Scientific Book chapters)

[18] P. ACZEL, B. AHRENS, T. ALTENKIRCH, S. AWODEY, B. BARRAS, A. BAUER, Y. BERTOT, M. BEZEM, T. COQUAND, E. FINSTER, D. GRAYSON, H. HERBELIN, A. JOYAL, D. LICATA, P. LUMSDAINE, A. MAHBOUBI, P. MARTIN-LÖF, S. MELIKHOV, A. PELAYO, A. POLONSKY, M. SHULMAN, M. SOZEAU, B. SPITTERS, B. VAN DEN BERG, V. VOEVODSKY, M. WARREN, C. ANGIULI, A. BORDG, G. BRUNERIE,

C. KAPULKIN, E. RIJKE, K. SOJAKOVA, J. AVIGAD, C. COHEN, R. CONSTABLE, P.-L. CURIEN, P. DYBJER, M. ESCARDÓ, K.-B. HOU, N. GAMBINO, R. GARNER, G. GONTHIER, T. HALES, R. HARPER, M. HOFMANN, P. HOFSTRA, J. KOCH, N. KRAUS, N. LI, Z. LUO, M. NAHAS, E. PALMGREN, E. RIEHL, D. SCOTT, P. SCOTT, S. SOLOVIEV. , *Homotopy Type Theory: Univalent Foundations of Mathematics*, Aucun, 2013, 448 p. , http://hal.inria.fr/hal-00935057

[19] Y. GUIRAUD, P. MALBOS. *Identities among relations for higher-dimensional rewriting systems*, in "Operads 2009", Séminaires et Congrès, Société Mathématique de France, 2013, vol. 26, pp. 145-161, http://hal.inria.fr/hal-00426228

Other Publications

[20] P. BOUTILLIER. , *Simple simpl*, 2013, http://hal.inria.fr/hal-00816918

[21] Y. GUIRAUD, P. MALBOS. , *Polygraphs of finite derivation type*, January 2014, 46 p. , http://hal.inria.fr/hal-00932845

[22] H. HERBELIN. , *A dependently-typed construction of semi-simplicial types*, 2013, http://hal.inria.fr/hal-00935446

[23] M. SOZEAU, N. TABAREAU. , *Univalence for free*, 2013, http://hal.inria.fr/hal-00786589

References in notes

[24] U. A. ACAR. , *Self-adjusting Computation*, Carnegie Mellon UniversityPittsburgh, PA, USA, 2005, AAI3166271

[25] D. J. ANICK. *On the Homology of Associative Algebras*, in "Trans. Amer. Math. Soc.", 1986, vol. 296, n^{o} 2, pp. 641–659

[26] H. P. BARENDREGT. , *The Lambda Calculus: Its Syntax and Semantics*, North HollandAmsterdam, 1984

[27] R. BERGER. *Confluence and Koszulity*, in "J. Algebra", 1998, vol. 201, n^{o} 1, pp. 243–283

[28] B. BUCHBERGER. *An algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal*, in "J. Symbolic Comput.", 2006, vol. 41, n^{o} 3-4, pp. 475–511, Translated from the 1965 German original by Michael P. Abramson

[29] A. CHURCH. *A set of Postulates for the foundation of Logic*, in "Annals of Mathematics", 1932, vol. 2, pp. 33, 346-366

[30] T. COQUAND. , *Une théorie des Constructions*, University Paris 7, January 1985

[31] T. COQUAND, G. HUET. *Constructions : A Higher Order Proof System for Mechanizing Mathematics*, in "EUROCAL'85", Linz, Lecture Notes in Computer Science, Springer Verlag, 1985, vol. 203

[32] T. COQUAND, C. PAULIN-MOHRING. *Inductively defined types*, in "Proceedings of Colog'88", P. MARTIN-LÖF, G. MINTS (editors), Lecture Notes in Computer Science, Springer Verlag, 1990, vol. 417

[33] P.-L. CURIEN, G. MUNCH-MACCAGNONI. *The duality of computation under focus*, in "Theoretical Computer Science", 2010, pp. 165–181

[34] H. B. CURRY, R. FEYS, W. CRAIG. , *Combinatory Logic*, North-Holland, 1958, vol. 1, §9E

[35] P. DEHORNOY, L. PARIS. *Gaussian groups and Garside groups, two generalisations of Artin groups*, in "Proc. London Math. Soc. (3)", 1999, vol. 79, n^{o} 3, pp. 569–604

[36] M. FELLEISEN, D. P. FRIEDMAN, E. KOHLBECKER, B. F. DUBA. *Reasoning with continuations*, in "First Symposium on Logic and Computer Science", 1986, pp. 131-141

[37] A. FILINSKI. *Representing Monads*, in "Conf. Record 21st ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages, POPL'94", Portland, OR, USA, ACM Press, 17-21 Jan 1994, pp. 446-457

[38] C. FÜHRMANN. *Direct Models for the Computational Lambda Calculus*, in "Electr. Notes Theor. Comput. Sci.", 1999, vol. 20, pp. 245-292

[39] S. GAUSSENT, Y. GUIRAUD, P. MALBOS. , *Coherent presentations of Artin monoids*, 2013, 45 p.

[40] G. GENTZEN. *Untersuchungen über das logische Schließen*, in "Mathematische Zeitschrift", 1935, vol. 39, pp. 176–210,405–431

[41] J.-Y. GIRARD. *Une extension de l'interpretation de Gödel à l'analyse, et son application à l'élimination des coupures dans l'analyse et la théorie des types*, in "Second Scandinavian Logic Symposium", J. FENSTAD (editor), Studies in Logic and the Foundations of Mathematics, North Holland, 1971, n^{o} 63, pp. 63-92

[42] T. G. GRIFFIN. *The Formulae-as-Types Notion of Control*, in "Conf. Record 17th Annual ACM Symp. on Principles of Programming Languages, POPL '90", San Francisco, CA, USA, 17-19 Jan 1990, ACM Press, 1990, pp. 47–57

[43] Y. GUIRAUD, P. MALBOS. *Higher-dimensional categories with finite derivation type*, in "Theory Appl. Categ.", 2009, vol. 22, n^{o} 18, pp. 420-478

[44] H. HERBELIN, A. SPIWACK. , *The Rooster and the Syntactic Bracket*, 2013

[45] W. A. HOWARD. *The formulae-as-types notion of constructions*, in "to H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism", Academic Press, 1980, Unpublished manuscript of 1969

[46] J.-L. KRIVINE. *A call-by-name lambda-calculus machine*, in "Higher Order and Symbolic Computation", 2005

[47] J.-L. KRIVINE. , *Un interpréteur du lambda-calcul*, 1986, Unpublished

[48] F. LAMARCHE. *On the Algebra of Structural Contexts*, in "Math. Struct. in Comp. Sci.", 2014, to appear

[49] P. LANDIN. *The mechanical evaluation of expressions*, in "The Computer Journal", January 1964, vol. 6, n^{o} 4, pp. 308–320

[50] P. LANDIN. , *A generalisation of jumps and labels*, UNIVAC Systems Programming Research, August 1965, n^{o} ECS-LFCS-88-66, Reprinted in Higher Order and Symbolic Computation, 11(2), 1998

[51] P. MARTIN-LÖF. , *A theory of types*, University of Stockholm, 1971, n^{o} 71-3

[52] J.-F. MONIN. *Proof Trick: Small Inversions*, in "Second Coq Workshop", Edinburgh Royaume-Uni, Y. BERTOT (editor), July 2010, http://hal.inria.fr/inria-00489412/en/

[53] G. MUNCH-MACCAGNONI. *Models of a Non-Associative Composition*, in "17th International Conference on Foundations of Software Science and Computation Structures (FoSSaCs)", 2014

[54] M. PARIGOT. *Free Deduction: An Analysis of "Computations" in Classical Logic*, in "Logic Programming, Second Russian Conference on Logic Programming", St. Petersburg, Russia, A. VORONKOV (editor), Lecture Notes in Computer Science, Springer, September 11-16 1991, vol. 592, pp. 361-380, http://www.informatik.uni-trier.de/~ley/pers/hd/p/Parigot:Michel.html

[55] J. C. REYNOLDS. *Definitional interpreters for higher-order programming languages*, in "ACM '72: Proceedings of the ACM annual conference", New York, NY, USA, ACM Press, 1972, pp. 717–740

[56] J. C. REYNOLDS. *Towards a theory of type structure*, in "Symposium on Programming", B. ROBINET (editor), Lecture Notes in Computer Science, Springer, 1974, vol. 19, pp. 408-423

[57] C. SQUIER, F. OTTO, Y. KOBAYASHI. *A finiteness condition for rewriting systems*, in "Theoret. Comput. Sci.", 1994, vol. 131, n^{o} 2, pp. 271–294

[58] THE COQ DEVELOPMENT TEAM. , *The Coq Reference Manual, version 8.2*, September 2008, http://coq.inria.fr/doc

[59] P. WADLER. *Call-by-value is dual to call-by-name*, in "Proceedings of ICFP", ACM, 2003, pp. 189–201

[60] N. ZEILBERGER. , *The Logical Basis of Evaluation Order and Pattern-Matching*, Carnegie Mellon University, 2009

[61] N. DE BRUIJN. , *AUTOMATH, a language for mathematics*, Technological University Eindhoven, November 1968, n^{o} 66-WSK-05