IN PARTNERSHIP WITH:
CNRS

Université Denis Diderot (Paris 7)

Activity Report 2014

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]
Jean-Jacques Lévy [Emeritus Senior Researcher, HdR]
Alexis Saurin [CNRS, Researcher]
Matthieu Sozeau [Inria, Researcher]

Faculty Members

Thierry Coquand [Inria International Chair, from Dec 2014]
Pierre Letouzey [Univ. Paris VII, Associate Professor]
Frédéric Loulergue [Univ. Orléans, Professor (in delegation), from Sep 2014]
Philippe Malbos [Univ. Lyon I, Associate Professor (in delegation), until Aug 2014]
Yann Régis-Gianas [Univ. Paris VII, Associate Professor]

PhD Students

Maxime Lucas [Univ. Paris VII, funded by ANR Cathre]
Akira Yoshimisu [Tokyo University, Inria internship, from Nov 2014]
Pierre Boutillier [Univ. Paris VII, until Feb 2014]
Cyrille Chenavier [Univ. Paris VII, funded by the IDEX FOCAL project]
Guillaume Claret [ENS Paris, Univ. Paris VII]
Amina Doumane [Univ. Paris VII, funded by FSMP from Oct 2014]
Thibaut Girka [CIFRE contract with Mitsubishi Rennes, from Oct 2014]
Lourdes Del Carmen González Huesca [ATER Univ. Paris VII from Sep 2014, funded by ANR PARAL-ITP until Aug 2014]
Étienne Miquey [Univ. Paris VII, from Oct 2014]
Jovana Obradovic [Univ. Paris VII, from Oct 2014]
Ludovic Patey [Univ. Paris VII]
Pierre-Marie Pédrot [Univ. Paris VII]

Post-Doctoral Fellows

Eric Finster [Inria, granted by ANR PI.R2 - RECRE project, until Dec 2014]
Marc Lasson [Inria]
Arnaud Spiwack [ADT Coq, from Sep 2014 to Oct 2014]

Visiting Scientists

Steven Awodey [Carnegie Mellon University, Professor, from May 2014 until Jun 2014]
Peter Azcel [University of Manchester, Emeritus Professor, from Apr 2014 until Jul 2014]
Bas Spitters [Univ. of Nijmegen, from Apr 2014 until Nov 2014]
Samuel Van Gool [PhD student, granted by ANR PI.R2 - RECRE project, until Mar 2014]
Vladimir Voevodsky [Institute for Advanced Study, Princeton, from Jun 2014 until Jul 2014]
Wojciech Jedynak [Wroclaw Univ., PhD student, May 2014]

Administrative Assistants

Kadidiatou Barry [Inria, from Sep 2014]

Lindsay Polienor [Inria, on maternity leave from Sep 2014]

Other

Paul-André Mellies [External collaborator, CNRS]

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.

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 [51] 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 [44], then by Howard and de Bruijn at the end of the 60's [54], [66], 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 [41] stemmed out – a formalism that is both a logic and a programming language and that is at the source of the Coq system [64].

3.1.2. Towards the calculus of constructions

The λ-calculus, defined by Church [40], 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 [37].

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 [60].

In 1985, Coquand and Huet [41], [42] in the Formel team of Inria-Rocquencourt explored an alternative approach based on Girard-Reynolds' system F [52], [63]. 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 called OCaml and 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 [43] 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 typed 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 exist – generally first-order ones.

3.4. Around and beyond the Curry-Howard correspondence

For two decades, the Curry-Howard correspondence has been limited to the intuitionistic case but since 1990, an important stimulus spurred on the community following Griffin's discovery that this correspondence was extensible to classical logic. The community then started to investigate unexplored potential connections between computer science and logic. One of these fields is the computational understanding of Gentzen's sequent calculus while another one is 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 [53] that some operators known as control operators were typable by the principle of double negation elimination ($\neg\neg A \Rightarrow A$), a principle that enables classical reasoning.

Control operators are used to jump from one location of a program to another. They were first considered in the 60's by Landin [58] and Reynolds [62] and started to be studied in an abstract way in the 80's by Felleisen *et al* [49], leading to Parigot's $\lambda\mu$-calculus [61], a reference calculus that is in close Curry-Howard correspondence with classical natural deduction. In this respect, control operators are fundamental pieces to establish a 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 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 rules. This line of work led in 2000 to 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 [57] and Krivine's abstract machine for call-by-name evaluation [56], [55]. 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 [50]: 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. New 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 [Gallium team, Rocquencourt], 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 [Toccata team, Saclay], Pierre-Marie Pédrot, Loïc Pottier [Marelle team, Sophia], Matthias Puech, Yann Régis-Gianas, François Ripault, Matthieu Sozeau, Arnaud Spiwack [Mines Paritech], Pierre-Yves Strub [IMDEA, Madrid], Enrico Tassi [Marelle team, Sophia], Benjamin Werner [Ecole Polytechnique].

4.1.1. Version 8.5

Version 8.5 was expected to be released after the summer of 2014, but this got delayed until the Coq Programming Language workshop mid-January 2015.

Coq 8.5 is a major release of the Coq proof assistant, including 5 major new features:

- Parallel development and compilation, inside files and across files, by Enrico Tassi (Inria SpecFun, then Marelle), a result of the Paral-ITP ANR project.
- Availability of all the features of Arnaud Spiwack's new proof engine, with more expressive, clearer semantics, multigoal tactics, deep backtracking,
- A compilation scheme from Coq to OCaml to native code by Maxime Dénès and Benjamin Grégoire (Inria Marelle, then University of Pennsylvania, then Inria Gallium), 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, with significant efficiency improvements.

Coq 8.5 also includes many improvements at different levels: the primitive tactics, the tactic language, the specification language, the tools associated to Coq, etc. For a full list of changes, the reader is invited to look at http://coq.inria.fr or at the files CHANGES of the Coq archive.

4.1.2. Evaluation algorithms

The new unfolding algorithm for global constants that was proposed by Pierre Boutillier is ready for use in Coq 8.5.

4.1.3. Internal representation of projections

A new internal representation of record projections has been implemented in the 8.5 release by Matthieu Sozeau. During the stabilisation of this feature, we added a backwards compatibility layer that allows users to switch seamlessly to the new representation, keeping the same user-level interface for primitive and non-primitive projections (the record types and values being unchanged). This new representation adds eta-conversion of records defined with primitive projections to the definitional equality of Coq, enlarging the set of conversion problems that can be automatically handled by the system.

4.1.4. Universes

The new universe polymorphism system by Matthieu Sozeau is part of the 8.5 release. The implementation has been stabilised, benchmarked and tested heavily in the last year, with much input from the Homotopy Type Theory development team. In [27], Matthieu Sozeau and Nicolas Tabareau presented the system formally. It has since been extended with user-friendly features like named universes and commands to display the status of universe constraints. With the help from Maxime Dénès (Gallium Team), the native compilation system has also been extended to fully support universe polymorphism.

4.1.5. Internal architecture of the Coq software

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 conventions.
- Almost all uses of the generic OCaml comparison have been chased and transformed into specific code, thereby avoiding 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.

4.1.6. Efficiency

Pierre-Marie Pédrot has been working on the overall optimisation 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.

4.1.7. Documentation generation

Yann Régis-Gianas continued the development of 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.

4.1.8. Maintenance and coordination

The maintenance and coordination of Coq has been jointly done by Hugo Herbelin, Pierre Boutillier, Pierre Letouzey, Matthieu Sozeau, Pierre-Marie Pédrot, in relation with the other participants to the development.

A Coq working group is organised every two months (5 times a year). From the end of October, a Coq lunch holds weekly welcoming any person interested in the development of Coq in general. Discussions about the development happen, in particular, on `coq-dev@inria.fr` and http://coq.inria.fr/bugs.

4.1.9. The Coq extraction

In 2014, Pierre Letouzey built an extension of the Coq extraction that targets directly one of the internal layers of the OCaml compiler. This way, it is possible to avoid the generation of OCaml concrete syntax by the extraction, followed by a parsing phase when the OCaml compiler is launched on the extracted code. Our extension is able to shortcut these two phases. The interest is twofold. First, it seriously reduces the amount of code that should be considered as critical during a program development via extraction. Secondly, with this approach we are able to directly compile and run certain extracted examples, and internalise the result back into Coq, leading to a new promising command Extraction Compute. This extension is currently quite experimental.

4.1.10. Parametricity for the Coq proof assistant

During his stay in the πr^2 team, Marc Lasson developed a plugin for parametricity theory in the Coq proof assistant.

Parametricity theory was originally introduced by John Reynolds in his seminal paper about polymorphic λ-calculus (also known as System F). It is used to formalise the opacity of abstract datatypes in programming languages that provide idioms to handle types generically. Polymorphic functions cannot inspect their arguments with an abstract type, and have to use them uniformly. The main tool of parametricity theory is that of logical relations, which are relations between programs of the same type that are defined by induction on the structure of types.

Marc Lasson's work consisted in developing a parametricity theory for the terms of Coq. The result of this work is a new plugin for the proof assistant that computes logical relations as well as the proof witnesses that programs satisfy these logical relations. It is available on github http://github.com/mlasson/paramcoq.

The purpose of this plugin is to allow to use parametric arguments in Coq proofs, the main direct application is the certification of parametric programs. Thanks to powerful expressiveness of the proof assistant, this plugin will allow future users to use parametric arguments to a larger scale. Although parametricity theory was originally developed for studying programs, the fact that we can use it in a proof assistant enables new uses in other contexts, such as the formalisation of mathematics and the meta-theory of proof assistants).

In [24], Marc Lasson showed that parametricity may also be useful to derive properties about the groupoidal interpretation of Type Theory. It was known that the equality types (also known as identity types) of type theory carry the algebraic structure of ω-groupoids (which is a higher-dimensional version of groups). Parametricity theory allows us to prove that the terms witnessing these algebraic laws are canonical, in the sense that there is only one way to implement them (up to higher-order equalities).

4.1.11. Formalisation in Coq

Hugo Herbelin's type-theoretic construction of semi-simplicial sets [9] has been formalised in Coq.

Matthieu Sozeau and Nicolas Tabareau formalised a setoid model of type theory in Coq http://github.com/mattam82/groupoid. They are working on extending this work to the groupoid model using the latest tools available in Coq 8.5.

Frédéric Loulergue collaborates with Frédéric Dabrowski and Thomas Pinsard (Univ. Orléans) to verify in Coq the compilation pass [21] for a language with nested atomic sections and thread escape to a language with only threads and locks, building on [45].

4.1.12. Systematic development of programs for parallel and cloud computing

During his stay in the πr^2 team, Frédéric Loulergue continues to collaborate with Kento Emoto (Kyushu Institute of Technology), Zhenjiang Hu (National Institute for Informatics, Japan), Julien Tesson (Univ. Paris-Est Créteil), Wadoud Bousdira (Univ. Orléans), Kiminori Matsuzaki (Kochi University of Technology) and Vitor Rodrigues (Rochester Institute of Technology) to develop the SyDPaCC framework (http://traclifo.univ-orleans.fr/SyDPaCC).

The goal of this framework is to ease the systematic development of correct parallel programs, in particular large scale data-intensive applications. In Coq, users write inefficient (sequential) functional programs and through (partly automated) program transformations based on the theory of list homomorphisms [32], bulk synchronous parallel homomorphisms [59] and semi-ring homomorphisms [48], an efficient sequential version is obtained. This version can then be automatically parallelised thanks to type class instance resolution and instances relating specific functions to their parallel counterparts. The parallel versions of the programs are written with a Coq axiomatisation of Bulk Synchronous Parallel ML (BSML) primitives. To obtain the final code, these Coq programs are extracted towards OCaml with calls to a parallel implementation of the BSML library.

As the SyDPaCC framework currently mixes certified code extracted from Coq and unverified code, Frédéric Loulergue and Pierre Letouzey are working on an extended extraction that generates, when possible, OCaml asserts for preconditions on function arguments. The next version of the generate-test-aggregate library of SyDPaCC will use Marc Lasson's plugin for parametricity to prove a "theorem for free": currently only instantiations of this theorem for each provided generator are proved.

4.1.13. Proofs of algorithms on graphs

Jean-Jacques Lévy's current research is to review basic algorithms and make their formal proofs of correctness in Why3 + Coq. Filliâtre and Pottier already started this research, but we plan to focus on graph algorithms, with concerns on the feasability of these formal proofs and on the design of good libraries on top of Coq or Ssreflect. The goal is not to disprove these algorithms which are most probably correct, but to develop a theory of tools for proving algorithms with proof assistants and provers. Standard techniques use assertions in Hoare logic or TLA or any other logic, which are written on paper. With the recent development of good computer proof-assistants and the fantastic progress of SMT provers, the goal of providing algorithms with their correctness proofs checked by computer seems possible. The plan of this research is to use Why3, Coq, Ssreflect on standard computing systems, and also to motivate a few students to work on this project. The challenge would be to compete with Filliâtre, Pottier and Monate's group at CEA (France), or Fournet, Swamy and Pierce at Microsoft Research or Univ. of Pennsylvania. We want to demonstrate that the use of SMT provers can be well coupled with the one of interactive provers as already done in Why3 and in F* with refined types in probable future. The expected outcome would be to extend to larger programs and real software. But this seems quite ambitious at present time, since large scale needs more technology as showed by Gonthier for his long proofs of mathematical theorems, and since the world of programming is much less structured than the world of mathematics.

We completed proofs of the following major algorithms as exposed in Sedgewick's book: sorting, searching, depth-first search in graphs. This work is performed in collaboration with Chen Ran at Iscas (Institute of Software, Chinese Academy of Sciences). Proofs can be found at http://jeanjacqueslevy.net/why3 (see also [10]).

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 has been developing the "Hacking Dojo", with the help of Alexandre Ly (master student of Paris Diderot). a web platform to automatically grade programming exercises. The platform is now used in several courses of the University Paris Diderot.

In collaboration with Grégoire Duchêne (master student at Paris Diderot), Yann Régis-Gianas developed Tamasheq, a fully-customisable interpreter for the OCaml programming language. Users of this interpreter can write plugins to instrument the interpretation of an OCaml program with visualisation, interactive debugging or logging. A paper is in preparation.

Yves Guiraud has updated the Catex tool for Latex, whose purpose is to automate the production of string diagrams from algebraic expressions (http://www.pps.univ-paris-diderot.fr/~guiraud/catex/catex.zip).

Yves Guiraud has developed the Python library Cox for the computation of coherent presentations of Artin monoids, after [18] (http://www.pps.univ-paris-diderot.fr/~guiraud/cox/cox.zip).

Yves Guiraud collaborates with Samuel Mimram (LIX) to develop the prototype Rewr that implements the homotopical completion-reduction procedure of [6] (http://www.pps.univ-paris-diderot.fr/~smimram/rewr).

Eric Finster has developed a new proof assistant, called Orchard, which aims to pursue the emerging connections between type theory and higher category theory by providing an environment in which to explicitly manipulate higher categorical diagrams using a notation based on a collection of shapes called opetopes. Opetopes have strong connections to concepts from computer science: they have a natural interpretation as a series of canonical indexed inductive types, and thus can be implemented and reasoned about using standard techniques from functional programming. The goal of the Orchard project is to forge links between the homotopical ideas of homotopy type theory, and the higher categorical ideas coming from higher-dimensional rewriting theory by providing a common language in which to reason about both. A preliminary implementation is available at https://github.com/ericfinster/orchard.

5. New Results

5.1. Highlights of the Year

We successfully organised the thematic trimester Semantics of Proofs and Certified Mathematics (IHP, April-July 2014). The trimester attracted over two hundred participants altogether (with about 60 "resident" participants staying a month or more), hosted 5 special workshops, as well as other related regevents such as Types, MAP (Mathematics, Algorithms, and Proofs). It was the first thematic trimester in the history of IHP to feature computer science prominently. There was a kick-off day on April 22, with talks of Georges Gonthier, Thomas Hales, Xavier Leroy, and Vladimir Voevodsky, with the presence of some science journalists. During the trimester, the Bourbaki Seminar devoted an afternoon (June 21) to these themes, with talks of Thomas Hales and Thierry Coquand.

Shortly before, Coq has received the Software System Award 2013 from the Association for Computing Machinery (ACM). Hugo Herbelin is one of the recipients of this prize.

5.2. Proof-theoretical and effectful investigations

Participants: Pierre Boutillier, Guillaume Claret, Pierre-Louis Curien, Amina Doumane, Hugo Herbelin, Etienne Miquey, Ludovic Patey, Pierre-Marie Pédrot, Yann Régis-Gianas, Alexis Saurin.

5.2.1. Proving with side-effects

In 2012, Hugo Herbelin showed that classical arithmetic in finite types extended with strong elimination of existential quantification proves the axiom of dependent choice. To get classical logic and choice together without being inconsistent is made possible first by constraining strong elimination of existential quantification to proofs that are essentially intuitionistic and secondly by turning countable universal quantification into an infinite conjunction of classical proofs evaluated along a call-by-need evaluation strategy so as to extract from them intuitionistic contents that complies to the intuitionistic constraint put on strong elimination of existential quantification. Étienne Miquey is currently working to get a presentation of this work in Curien-Herbelin's μ-$\widetilde{\mu}$-calculus, with the aim of getting in the end a CPS-translation. Such a translation would provide a strong argument of normalisation for the calculus, as well as a better undertanding of the mechanisms of the calculus, especially the side-effect part and the meaning of the existential quantifier restriction.

Hugo Herbelin and Danko Ilik carried on their work on the computational content of completeness proofs and in particular of the computational content of Gödel's completeness theorem. Hugo Herbelin presented their work at the workshop PSC 2014.

5.2.2. Reverse mathematics

Ludovic Patey studied with Laurent Bienvenu and Paul Shafer the provability strength of Ramsey-type versions of theorems like König's lemma. The corresponding paper is submitted to the Journal of Mathematical Logic. Ludovic Patey studied with Laurent Bienvenu the constructions of diagonal non-computable functions by probabilistic means. They submitted a paper to Information and Computation. Ludovic Patey worked on the existence of universal instances in reverse mathematics, and submitted a paper to Annals of Pure and Applied Logic. He worked on the relations between diagonal non-computability and Ramsey-type theorems and submitted a paper to the Archive for Mathematical Logic. He studied the links between the iterative forcing framework developed by Lerman, Solomon & Towsner and the notion of preservation of hyperimmunity and submitted a paper to Computability in Europe 2015.

5.2.3. Gödel's functional interpretation

Pierre-Marie Pédrot kept developing the proof-as-program interpretation of Gödel's Dialectica translation, as seen through the prism of classical realisability. This work was presented at TYPES 2014 and later published at LICS 2014 [26].

5.2.4. 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. This work was presented at JFLA 2014 [30] early 2014 and later expanded to the classical case, encompassing $\lambda\mu$-calculus.

5.2.5. 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. Their work gave rise to a presentation at the Workshop on Infinitary Rewriting that took place in Vienna last July as part of FLOC 2014.

5.2.6. Alternative syntaxes for proofs

Amina Doumane and Alexis Saurin, in a joint work with Marc Bagnol, studied the structure of several correctness criteria for linear logic proof-nets and could relate them through a new primitive notion of dependency. This work was first presented at JFLA 2014 [29] early 2014 and later at Structure and Deduction in Vienna as part of FLOC 2014. An expanded version has recently been accepted at FOSSACS 2015 [19].

5.3. Type theory and the foundations of Coq

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

5.3.1. Description of type theory

Hugo Herbelin and Arnaud Spiwack completed and published their characterisation of the type constructions of Coq in terms of atomic constructions rather than their usual description as a monolithic scheme [23]. This work permitted both a more pedagogical presentation of Coq's type system, and a more tractable and composable mathematical model of Coq on which meta-properties can be stated and proved.

5.3.2. 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 is in the process of being published in a special issue of MSCS on homotopy type theory [9].

The technique scales to provide type-theoretic constructions for arbitrary presheaves on Reedy categories, thus including simplicial sets.

5.3.3. Proof irrelevance, eta-rules

During his master's internship supervised by Matthieu Sozeau, Philipp Haselwarter studied a formulation of proof-irrelevance based on the rooster and the syntactic bracket presentation by Spiwack and Herbelin [23]. This resulted in a decomposition of the calculus cleanly showing the use of smashing and a better understanding of the restricted elimination rules of propositions. It also clearly shows that the inductive type for accessibility, used to justify general wellfounded definitions, can not be interpreted as a proof-irrelevant proposition in this calculus.

5.3.4. Unification

Matthieu Sozeau is continuing work in collaboration with Beta Ziliani (PhD at MPI-Saarbrücken) on formalising the unification algorithm used in Coq, which is central for working with advanced type inference features like Canonical Structures. This is the first precise formalisation of all the rules of unification including the ones used for canonical structure resolution. The presentation currently excludes some heuristics that were added on top of the core algorithm in Coq, until they can be studied more carefully. This work, part of B. Ziliani's thesis, was presented at the UNIF'14 workshop [28] and the Coq workshop in Vienna. A submission is in preparation.

5.3.5. Foundations and paradoxes

Arnaud Spiwack generalised previous works by Herman Geuvers and Hugo Herbelin to implement Hurkens's paradox of the impredicative system U^-. The resulting Coq implementation, which is completely independent from the impredicative features of Coq, generalises the two special cases which were previously used to prove negative results about impredicativity in Coq.

5.4. Homotopy of rewriting systems

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

5.4.1. Coherent presentations of Artin monoids

With Stéphane Gaussent (ICJ, Univ. de Saint-Étienne), Yves Guiraud and Philippe Malbos have used higher-dimensional rewriting methods for the study of Artin monoids, a class of monoids that is fundamental in algebra and geometry. This work uses the formal setting of coherent presentations (a truncation of polygraphic resolutions at the level above relations) to formulate, in a common language, several known results in combinatorial group theory: one by Tits about the fundamental group of a graph associated to an Artin monoid [65], and one by Deligne about the actions of Artin monoids on categories [47], both proved by geometrical methods. In this work, an improvement of Knuth-Bendix's completion procedure is introduced, called the homotopical completion-reduction procedure, and it is used to give a constructive proof and to extend both theorems. This work will appear in Compositio Mathematica [18] and has been implemented in a Python library.

The next objective of this collaboration is to extend those results in every dimension, first to Artin monoids, then to Artin groups, with a view towards two well-known open problems in the field: the word problem of Artin groups and the so-called $K(\pi, 1)$ conjecture.

5.4.2. New methods for the computation of polygraphic resolutions

Maxime Lucas, supervised by Pierre-Louis Curien and Yves Guiraud, develops Squier's theory in the setting of cubical ω-categories. This will allow easier and more explicit computations of polygraphic resolutions than in the globular setting of [5], and the use of new effective methods such as the reversing algorithm from Garside theory [46].

Yves Guiraud currently collaborates with Patrick Dehornoy (Univ. de Caen) and Matthieu Picantin (LIAFA, Univ. Paris 7) to extend the constructions of [18] to other important families of monoids, such as the plactic monoid, the Chinese monoid and the dual braid monoids.

5.4.3. Higher-dimensional linear rewriting

Cyrille Chenavier, Pierre-Louis Curien, Yves Guiraud and Philippe Malbos investigate with Eric Hoffbeck (LAGA, Univ. Paris 13) and Samuel Mimram (LIX, École Polytechnique) the links between set-theoretic rewriting theory and the computational methods known in symbolic algebra, such as Gröbner bases [39]. This interaction is supported by the Focal project of the IDEX Sorbonne Paris Cité.

With Eric Hoffbeck (LAGA, Univ. Paris 13), Yves Guiraud and Philippe Malbos have introduced the setting of linear polygraphs to formalise a theory of linear rewriting (in the sense of linear algebra), generalising Gröbner bases. They have adapted to algebras the procedure of [5] that computes polygraphic resolutions from convergent presentations of monoids, with applications to the decision of an important homological property called Koszulness. This work is contained in [35] and it has been presented at IWC 2014 [31].

Cyrille Chenavier, supervised by Yves Guiraud and Philippe Malbos, explores the use of Berger's theory of reduction operators [38] to design new methods for the study of linear rewriting systems, and to promote the use of rewriting techniques in combinatorial algebra.

5.4.4. Homotopical and homological finiteness conditions

Yves Guiraud and Philippe Malbos have written a comprehensive introduction [36] 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.

5.4.5. Wiring structure of operads and operad-like structures

Building on recent ideas of Marcelo Fiore on the one hand, and of François Lamarche on the other hand, Pierre-Louis Curien and Jovana Obradović develop a syntactic approach, using some of the kit of Curien-Herbelin's duality of computation and its polarised versions of Munch and Curien, to the definition of various structures that have appeared in algebra under the names of operads, cyclic operads, dioperads, properads, modular and wheeled operads, permutads, etc.... These structures are defined in the literature in different flavours. We seek to formalise the proofs of equivalence between these different styles of definition, and to make these proofs modular, so as not to repeat them for each variation of the notion of operad. Preliminary results are being presented in January 2015 at the Mathematical Institute of the Academy of Sciences (Belgrade).

5.5. Coq as a functional programming language

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

5.5.1. Type classes and libraries

Type Classes are heavily used in the HoTT/Coq library (http://github.com/HoTT/coq) started by the Univalent Foundations program at the IAS, to which Matthieu Sozeau participated. To ease the development of this sophisticated library, Matthieu Sozeau implemented a number of extensions to type class resolution to make it more predictable and efficient. These are now part of the Coq 8.5 release.

5.5.2. Dependent pattern-matching

The dissertation of Pierre Boutillier presents and formalises a new algorithm to compile dependent pattern-matching into a chain of Coq case analyses. It avoids the use of the "uniqueness of identity proofs" axiom in more cases than the former proposal by McBride and McKinna.

5.5.3. Incrementality in proof languages

Lourdes del Carmen González Huesca and Yann Régis-Gianas developed a new variant of the differential lambda calculus that has two main features: (i) it is deterministic ; (ii) it is based on a notion of a first-class changes. A paper is in preparation.

5.5.4. Proofs of programs in Coq

In collaboration with David Mentre (Mitsubishi Rennes), Thibaut Girka and Yann Régis-Gianas worked on a certified generator for correlating programs. A correlating program is a program that represents the semantic difference between two (close) versions of a program by performing a static scheduling of their instructions. Performing an abstract interpretation on the correlating program provides a representation of the semantic differences between the two versions of a program. A paper is written and should be submitted soon.

5.5.5. Typed tactic language

In collaboration with Beta Ziliani (MPI) and Thomas Refis (master 2 student at University Paris Diderot), Yann Régis-Gianas starts the development of the version 2 of Mtac, a tactic language for Coq. Mtac is a DSL embedded in the Coq proof assistant. Roughly speaking, it allows Coq to be used as a tactic language for itself. With this work, Mtac 2 now includes first class goals. A paper is in preparation.

5.5.6. Tactic engine

Arnaud Spiwack joined the team for two months (Sept—Oct 2014) to finalise the integration and documentation of his re-engineering of Coq's interactive proof engine for the v8.5 version. The new perspective taken by this new engine is to shift the primary focus from how tactics (proof instructions) can modify goals (proof obligations) to focus on the way tactics compose. By making sure that composition of tactics has good mathematical properties, the new engine makes it possible to combine tactics in a more predictable and more powerful way. This new engine is also notable for the introduction of an abstract interface for tactics and tactic composition which makes it easy to augment tactics with new capabilities. The most notable such features are so-called dependent subgoals, which makes more fine-grained proofs possible and significantly improves the support for dependent types; and backtracking which gives the possibility to deploy very modular proof-search components. During his two months in the team, Arnaud Spiwack also added support for tracing tactic execution (Info), again taking advantage of his modular design.

5.5.7. Effectful programming

Guillaume Claret and Yann Régis-Gianas developed a compiler from a subset of OCaml with effects to Coq. Possible effects are the exceptions, the global references and the non-termination. Guillaume Claret and Yann Régis-Gianas developed Pluto, a concurrent HTTP web server written in Gallina. They worked on techniques to certify such interactive programs, formalising the reasoning by use cases. Use cases are proven correct giving a scenario, a typed schema of interactions between a program and an environment, built using the tactic mode of Coq as a symbolic debugger.

5.5.8. Libraries

Sébastien Hinderer and Pierre Letouzey contributed an extended library of lists. Pierre Letouzey contributed an extended library about Peano numbers, that takes advantages of the "Numbers" modular framework done earlier.

6. Partnerships and Cooperations

6.1. National Initiatives

Alexis Saurin (coordinator) and Yann Régis-Gianas are members of the four-year RAPIDO ANR project accepted in 2014 and starting in January 2015. RAPIDO aims at investigating the use of proof-theoretical methods to reason and program on infinite data objects. The goal of the project is to develop logical systems capturing infinite proofs (proof systems with least and greatest fixed points as well as infinitary proof systems), to design and to study programming languages for manipulating infinite data such as streams both from a syntactical and semantical point of view. Moreover, the ambition of the project is to apply the fundamental results obtained from the proof-theoretical investigations (i) to the development of software tools dedicated to the reasoning about programs computing on infinite data, *e.g.* stream programs (more generally coinductive programs), and (ii) to the study of properties of automata on infinite words and trees from a proof-theoretical perspective with an eye towards model-checking problems. Other permanent members of the project are Christine Tasson from PPS, David Baedle from LSV, ENS-Cachan, and Pierre Clairambault, Damien Pous and Colin Riba from LIP, ENS-Lyon.

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. This project is with members of the LAGA (Laboratory of Mathematics, Univ. Paris 13).

Pierre-Louis Curien (coordinator), Yves Guiraud and Philippe Malbos are members of the four-years Cathre ANR project, started in January 2014. This project investigates 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 in 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 (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 in January 2012 and is a joint project with LRI, PPS and Inria Grenoble.

Yann Régis-Gianas collaborates with Mitsubishi Rennes on the topic of differential semantics. This collaboration led to the CIFRE grant for the PhD of Thibaut Girka.

Matthieu Sozeau is a member of the CoqHoTT project led by Nicolas Tabareau (Ascola team, École des Mines de Nantes), funded by an ERC Starting Grant.

6.2. European Initiatives

6.2.1. Collaborations with Major European Organisations

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 International Partners

The project-team has collaborations with Wroclaw University (Poland), University of Aarhus (Denmark), University of Oregon, University of Tokyo, University of Sovi Sad, University of Nottingham, Institute of Advanced Study, MIT and University of Cambridge.

6.3.2. 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).

6.4. International Research Visitors

6.4.1. Visits of International Scientists

Beta Ziliani (MPI Saarbrücken) visited πr^2 for one week in November 2014 to collaborate with Yann Régis-Gianas and Matthieu Sozeau.

Peter Aczel (Manchester Univ.), Steve Awodey (Carnegie Mellon University), Thierry Coquand (Univ. Göteborg), and Vladimir Voevodsky (Institute for Advanced Study) were Inria funded invited professors for the thematic IHP trimester Semantics of Proofs and Certified Mathemtatics.

6.4.1.1. Internships

Akira Yoshimizu is an international Inria intern, working on abstract machines for quantum programmming languages inspired from game semantics and linear logic.

6.4.2. Visits to International Teams

6.4.2.1. Research stays abroad

Pierre-Louis Curien visited Chili (Univ. of Talca) in March 2014 (collaborative work with Maria Ronco in operad theory).

7. Dissemination

7.1. Promoting Scientific Activities

7.1.1. Collective responsibilities

Pierre-Louis Curien was a member of the Conseil Scientifique of the INSII (CNRS), until September 2014. 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.

Frédéric Loulergue is a member of the editorial board of Scalable Computing: Practice and Experience, and Technique et Science Informatiques.

Hugo Herbelin, Pierre Letouzey and Matthieu Sozeau are co-editors of the post-proceedings of the conference TYPES 2014 which was held in Paris in May.

7.1.3. Program committees and organising committees

Alexis Saurin has been a PC member of GaLoP 2014 (International Workshop on Games and Logic for Programming) which took place during ETAPS 2014 in Grenoble, as well as FSTTCS 2014, which took place in Mumbai in December. He is in the PC of WoC 2015, the workshop on continuations, affiliated with ETAPS.

Alexis Saurin is member of the Education committee of the ACM Special Interest Group on Logic (SIGLOG, http://siglog.hosting.acm.org).

Hugo Herbelin, Pierre Letouzey and Matthieu Sozeau co-organised and co-chaired the TYPES'14 conference in Paris in May.

Yves Guiraud and Philippe Malbos were among the organisers of the 5-week programme Mathematical Structures of Computation, held in Lyon in January-February 2014, and supported by the Labex MILYON. They were 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.

Yves Guiraud is in the organising committee and PC co-chair of Higher-Dimensional Rewriting and Applications (HDRA), a workshop of the International Conference on Rewriting, Deduction and Programming (RDP) 2015.

Pierre-Louis Curien, Hugo Herbelin and Paul-André Melliès were the organisers of the IHP trimester Semantics of proofs and certified mathematics (cf. highlights). They also organised a spring school at CIRM preceding the trimester, in April 2014.

Pierre-Louis Curien and Hugo Herbelin are members of the steering committee of the conference Typed Lambda Calculi and Applications (TLCA).

Pierre-Louis Curien is a member of the steering committee of the international workshop Games for Logic and Programming Languages (GaLop).

Frédéric Loulergue is a member of the steering committee of the international symposium on High-Level Parallel Programming and Applications (HLPP), he is a member of the program committee of the International Conference on Computational Science (ICCS'15).

Yann Régis-Gianas, Pierre Letouzey, Matthieu Sozeau are the organisers of the "Ecole de Printemps d'Informatique Théorique 2015" about proof of programs. This school has been accepted as a "Ecole Thématique" of the CNRS.

Matthieu Sozeau is a member of the steering committee of the Dependently Typed Programming international workshop (DTP).

Matthieu Sozeau co-organised and chaired the first Coq for Programming Languages (CoqPL) workshop, collocated with POPL'15 in Mumbai, India, in January 2015.

7.1.4. Jury participation

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

Yann Régis-Gianas has been a member of the "Comité de Sélection" for an assistant professor position at the University of Paris Sud.

Pierre-Louis Curien has been a member of the "Comité de Sélection" for a professor position in computer science at the University Paris Diderot.

Matthieu Sozeau has been a member of the Student Research Competition Jury at ICFP 2014 in Gothenburg.

7.1.5. Invited talks

P.-L. Curien gave an invited talk at the workshop "Algebra and Computation" in Lyon (mentioned above).

J.-J. Lévy wrote a paper for P.-L. Curien's Festschrift Volume, entitled "On the length of Standard Reductions in the Lambda Calculus" [2] (corresponding to his talk back in September 2013 in Venice at the anniversary meeting in his honour).

P.-L. Curien and J.-J. Lévy participated to Luca Cardelli's Festschrift at Microsoft Research in Cambridge, UK (September 8-9). They gave talks entitled "Around formal parametricity" and "Simple Proofs for Simple Programs", respectively.

J.-J. Lévy participated to Matthew Hennessy's Festschrift at IMD in Lucca, Italy (October 15-16). He gave a talk on "Simple Proofs for Simple Programs".

J.-J. Lévy and A. Saurin participated to "Les Journées LAC (Logique, Algèbre, Calcul)" in Chambéry, France (November 20-21). They gave talks on "Simple Proofs for Simple Programs" and "On the dependencies of logical rules", respectively.

J.-J. Lévy participated to "The 2nd Locali workshop" between Institute of Software, Chinese Academy of Sciences and Inria/Paris 7 (November 24-26). He gave a talk on "Simple Proofs for Simple Programs".

M. Sozeau gave an invited talk on "Universe Polymorphism: Subtyping and unification" at the XIXth Agda meeting in Paris (May 2014).

L. Patey has been invited to give a talk at the Workshop on Computability Theory 2014, at a special session of the conference Computability in Europe 2015 and to a Dagstuhl Seminar 2015.

7.1.6. Presentation of papers

Philippe Malbos has presented [31] at IWC 2014.

Marc Lasson has presented [24] at MFPS 2014.

Hugo Herbelin has presented his joint work with Danko Ilik on the computational content of Gödel's completeness theorem at the workshop PSC 2014.

Matthieu Sozeau has presented [27] at ITP 2014 and [28] at UNIF 2014.

Ludovic Patey received the best student paper award for his paper about dichotomy theorems and an extended version of his paper submitted to the Computability journal has been accepted.

7.1.7. Other presentations

Yann Régis-Gianas gave a talk about Coq at the PPS-LIAFA "pedagogical" meeting.

Frédéric Loulergue gave a short talk at JFLA'15 and a talk at the International Symposium on Symbolic and Numeric Algorithms for Scientific Computing SYNASC'14 (Timisoara).

Matthieu Sozeau gave talks on the development of Coq at the Coq Workshop in Vienna (July 2014) and at the CoqPL workshop in Mumbai (Jan 2015). He also gave a talk on generalised rewriting strategies at TYPES 2014 and at the Coq Workshop.

7.1.8. Talks in seminars

Lourdes González gave a talk on "Incrémentalité dans le calcul" during the Journées PPS, September 2014.

Marc Lasson gave talks about parametricity for dependent types at the TYPES 2014 workshop, at the journéees nationales du GDR IM 2014, and the following team seminars: séminaire LCR, LIPN, Paris 13; séminaire Logique et Interactions, Institut de Mathématiques de Marseille; séminaire d'équipe ACADIE, IRIT Toulouse. He also presented his plugin during a Coq working group held in Paris.

Eric Finster gave a talk on Homotopy Type Theory at the "Notions of Identification" seminar at L'équipe ERC Philosophy of Canonical Quantum Gravity, Paris 7 and a talk entitled "Opetopic Diagrams as a Language for Higher Categorical Proofs" at the IHP Trimester in Paris.

Alexis Saurin gave a talk in PARSIFAL seminar at LIX on dependency and correctness of proof-nets.

Matthieu Sozeau gave a talk at the Deducteam seminar at Inria (place d'Italie) on the formalisation of the groupoid model of type theory.

Pierre-Louis Curien gave a seminar talk at the University of Talca (Chile) on "Revisiting the categorical interpretation of type theory" (March).

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

Lourdes González attended TYPES 2014 in Paris and the IHP trimester "Semantics of proofs and certified mathematics".

Marc Lasson attended TYPES 2014 in Paris, the IHP trimester, and the Homotopy Type Theory Workshop held at the Mathematical Institute, University of Oxford, in November 2014.

Frédéric Loulergue attended SYNASC 2014 in September (Timisoara), and the JFLA'15 in Val d'Ajol.

Eric Finster attended the IHP Trimester in Paris, and the Homotopy Type Theory Workshop at Oxford.

Alexis Saurin and Matthieu Sozeau attended JFLA 2014, the IHP trimester, Types 2014 and FLOC 2014.

Matthieu Sozeau attended the Heidelberg Laureate Forum in September 2014 and ICFP'14.

7.1.10. Groupe de travail Théorie des types et réalisabilité

This is one of the working groups of PPS, jointly organised by Hugo Herbelin and Paul-André Melliès, since September 2009. It is held weekly. Matthieu Sozeau joined the organisation this year.

Internal speakers this year were Eric Finster on Higher Dimensional Syntax and Ludovic Patey on an Introduction to reverse mathematics. The external speakers were Jean-Baptiste Jeannin (CMU), Matthias Puech (McGill University), Sylvain Schmitz (ENS Cachan), Timothy Bourke (Inria and ENS), Cătălin Hriţcu (Inria), Pierre-Evariste Dagand (Inria), Carsten Schürmann (ITU Copenhagen), Conor McBride (Strathclyde University), Nicolas Pouillard (IT University of Copenhagen, Denmark), Alois Brunel (LIPN).

7.2. Teaching - Supervision - Juries

7.2.1. Teaching

Licence: Lourdes González has a temporary research and teaching position (A.T.E.R) at University Paris 7 for the academic year 2014–2015. During the first semester (Sep-Dec 2014) she was in charge of TP (Travaux pratiques, 24 hours) on the subject "Principes de fonctionnement des machines binaires" (L1).

Licence: Étienne Miquey was in charge of TP (Travaux practiques, 24 hours) in the course "Introduction à la programmation" (L1) at University Paris 7 during the first semester 2014/15.

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).

Master: Alexis Saurin taught, jointly with Christine Tasson, a Master 2 course in “Logique Mathématique et Fondements de l’Informatique” (LMFI), Université Paris Diderot, entitled “Lambda-calcul: des abstractions aux applications”. He taught about 30H. In addition, Saurin chairs LMFI M2 from september 2013.

Master: Yann Régis-Gianas took part in the MPRI course entitled “Type systems”: he taught 12 hours about generalised algebraic data types, higher-order Hoare logic and dependently typed programming.

MOOC: In collaboration with Roberto Di Cosmo and Ralf Treinen, Yann Régis-Gianas wrote a proposal for a MOOC about the OCaml programming language. The proposal has been accepted and the course is in preparation.

Master: Matthieu Sozeau teaches in the MPRI course on Advanced uses of Proof Assistants (12 hours + a project), together with Assia Mahboubi (Inria SpecFun).

Master: Matthieu Sozeau taught an introducory course on software verification to M2 Pro students at University Paris 7 during the first semester (Lectures + Practical Works, 20h).

7.2.2. Supervision

Internship: Alexis Saurin has supervised the L3 internship of Paul Fermé.

Internship: Alexis Saurin has supervised the M1 internship of Simon Lunel.

Internship: Alexis Saurin has supervised, with David Baelde, the M2 internship of Amina Doumane.

Internship: Yann Régis-Gianas has supervised the M2 internship of Thibaut Girka.

Internship: Yann Régis-Gianas has supervised the M2 internship of Thomas Refis.

Internship: Yann Régis-Gianas has supervised the M1 internship of Grégoire Duchêne.

Internship: Yann Régis-Gianas has supervised the M1 internship of Alexandre Ly.

Internship: Matthieu Sozeau has supervised the M2 internship of Philipp Haselwarter.

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

PhD in progress: Guillaume Claret, Programmation avec effets en Coq, September 2012, supervised 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: Thibaut Girka, Sémantique différentielle, October 2014, supervised by Roberto DiCosmo and Yann Régis-Gianas.

PhD in progress: Cyrille Chenavier, Méthodes algébriques pour la réécriture linéaire, supervised by Yves Guiraud and Philippe Malbos.

PhD in progress: Maxime Lucas, Résolutions polygraphiques cubiques et théorie de Garside, supervised by Yves Guiraud and Pierre-Louis Curien.

PhD in progress: Jovana Obradović, Langages pour la description de différentes sortes d’opérades, supervised by Pierre-Louis Curien.

PhD in progress: Amina Doumane, Ludique, automates, points fixes, supervised by Alexis Saurin, David Baelde and Pierre-Louis Curien.

PhD in progress: Étienne Miquey, Réalisabilité classique et effets de bords, September 2014, supervised by Hugo Herbelin and Alexandre Miquel.

7.2.3. Juries

Pierre-Louis Curien was president of the thesis juries of Aloïs Brunel (Univ. Paris 13), Guillaume Jaber (Univ. Nantes), Valentin Blot (ENS Lyon), and Marc Bagnol(Univ. Marseille).

Pierre-Louis Curien was president of the habilitation jury of Philippe Malbos (Univ. Lyon 1).

He is a referee for the habilitation theses of Olivier Serre (LIAFA), Russ Harmer (ENS Lyon), and Paul-André Melliès.

Frédéric Loulergue was president of the thesis juries of Hélène Coullon (Univ. Orléans), Mouhamadou Sakho (Univ. Orléans) and Nader Khammassi (ENSTA Bretagne) ; referee for the theses of Nuno Gaspar (Univ. Nice), Nader Khammassi (ENSTA Bretagne) and Charif Mahmoudi (Univ. Paris-Est Créteil) ; supervisor of the thesis of Thomas Pinsard (Univ. Orléans).

7.3. Popularization

Pierre-Louis Curien wrote the editorial of a special “hors série” issue of the information letter of The Fondation Sciences Mathématiques, entitled “Des preuves et des programmes”, may 2014. He wrote an introductory article “Formalisation mathématique, certification logicielle, même combat!” in the journal Gazette des Mathématiciens 142, 83-86 (octobre 2014).

Lourdes González-Huesca and Étienne Miquey took part in the animation of the “Fête de la Science” event at the University Paris 7.

Étienne Miquey took part in the animation of several activities about mathematics in elementary and high schools of Paris.

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 organised the “Fête de la Science” event for the computer science department of the University Paris 7.

Yann Régis-Gianas and Pierre Letouzey took part in the “Salon Culture et Jeux mathématiques” at Saint Sulpice, Paris.

Yann Régis-Gianas gave several conferences about “What is programming?” in elementary and high 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] A. ASPERTI, J.-J. LÉVY. *On the length of Standard Reductions in the Lambda Calculus*

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

[4] 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/

[5] Y. GUIRAUD, P. MALBOS. *Higher-dimensional normalisation strategies for acyclicity*, in "Advances in Mathematics", 2012, vol. 231, n^{o} 3-4, pp. 2294-2351 [*DOI :* 10.1016/J.AIM.2012.05.010], https://hal.archives-ouvertes.fr/hal-00531242

[6] 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], https://hal.inria.fr/hal-00818253

[7] 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

[8] 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/

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

[10] J.-J. LÉVY. *Simple proofs of simple programs in Why3*

[11] 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

[12] 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

[13] 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

[14] A. SAURIN. *On the Relations between the Syntactic Theories of $\lambda\mu$-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/

[15] 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

[16] P. BOUTILLIER. *New tool to compute with inductive in Coq*, Université Paris-Diderot - Paris VII, February 2014, https://tel.archives-ouvertes.fr/tel-01054723

Articles in International Peer-Reviewed Journals

[17] P.-L. CURIEN, R. GARNER, M. HOFMANN. *Revisiting the categorical interpretation of dependent type theory*, in "Theoretical Computer Science", 2014, vol. 546, pp. 99-119 [*DOI :* 10.1016/J.TCS.2014.03.003], https://hal.archives-ouvertes.fr/hal-01114033

[18] S. GAUSSENT, Y. GUIRAUD, P. MALBOS. *Coherent presentations of Artin monoids*, in "Compositio Mathematica", December 2014, pp. 1-42 [*DOI :* 10.1112/S0010437X14007842], https://hal.archives-ouvertes.fr/hal-00682233

International Conferences with Proceedings

[19] M. BAGNOL, A. DOUMANE, A. SAURIN. *On the dependencies of logical rules*, in "FOSSACS, 18th International Conference on Foundations of Software Science and Computation Structures", London, United Kingdom, April 2015, https://hal.archives-ouvertes.fr/hal-01110340

[20] S. CASTELLAN, J. HAYMAN, M. LASSON, G. WINSKEL. *Strategies as Concurrent Processes*, in "MFPS 2014", Ithaca, United States, Proceedings of the 30th Conference on the Mathematical Foundations of Programming Semantics (MFPS XXX), June 2014, vol. 308, pp. 87–107 [*DOI :* 10.1016/J.ENTCS.2014.10.006], https://hal.archives-ouvertes.fr/hal-01105258

[21] F. DABROWSKI, F. LOULERGUE, T. PINSARD. *Nested atomic sections with thread escape: Compilation to threads and locks*, in "ACM Symposium on Applied Computing (SAC)", Salamanca, Spain, ACM, April 2015, https://hal.inria.fr/hal-01105093

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

[23] H. HERBELIN, A. SPIWACK. *The Rooster and the Syntactic Bracket* , in "19th International Conference on Types for Proofs and Programs (TYPES 2013)", Toulouse, France, Leibniz International Proceedings in Informatics (LIPIcs), Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, July 2014, vol. 26, pp. 169–187 [*DOI :* 10.4230/LIPICS.TYPES.2013.169], https://hal.inria.fr/hal-01097919

[24] M. LASSON. *Canonicity of Weak ω-groupoid Laws Using Parametricity Theory*, in "MFPS 2014", Ithaca, United States, Proceedings of the 30th Conference on the Mathematical Foundations of Programming Semantics (MFPS XXX), June 2014, vol. 308, pp. 229 - 244 [*DOI :* 10.1016/J.ENTCS.2014.10.013], https://hal.archives-ouvertes.fr/hal-01105252

[25] G. MUNCH-MACCAGNONI. *Models of a Non-Associative Composition*, in "FOSSACS 2014 - 17th International Conference on Foundations of Software Science and Computation Structures", Grenoble, France, A. MUSCHOLL (editor), Springer, April 2014, vol. 8412, pp. 396-410 [*DOI :* 10.1007/978-3-642-54830-7_26], https://hal.inria.fr/hal-00996729

[26] P.-M. PÉDROT. *A Functional Functional Interpretation*, in "CSL-LICS 2014 - Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science", Vienna, Austria, July 2014 [*DOI :* 10.1145/2603088.2603094], https://hal.archives-ouvertes.fr/hal-01111802

[27] M. SOZEAU, N. TABAREAU. *Universe Polymorphism in Coq*, in "Interactive Theorem Proving", Vienna, Austria, July 2014, https://hal.inria.fr/hal-00974721

[28] B. ZILIANI, M. SOZEAU. *Towards a better-behaved unification algorithm for Coq* , in "UNIF 2014 Workshop", Vienna, Austria, July 2014, pp. 74-87, https://hal.archives-ouvertes.fr/hal-01111193

National Conferences with Proceedings

[29] M. BAGNOL, A. DOUMANE, A. SAURIN. *Analyse de dépendances et correction des réseaux de preuve*, in "JFLA 2014 - Vingt-cinquièmes Journées Francophones des Langages Applicatifs", Fréjus, France, January 2014, https://hal.archives-ouvertes.fr/hal-01110338

[30] A. SAURIN, P.-M. PÉDROT. *Nécessité faite loi: de la réduction linéaire de tête à l'évaluation paresseuse*, in "JFLA 2014 - Vingt-cinquièmes Journées Francophones des Langages Applicatifs", Fréjus, France, January 2014, https://hal.archives-ouvertes.fr/hal-01110337

Conferences without Proceedings

[31] Y. GUIRAUD, E. HOFFBECK, P. MALBOS. *Confluence of linear rewriting and homology of algebras*, in "3rd International Workshop on Confluence", Vienna, Austria, July 2014, https://hal.archives-ouvertes.fr/hal-01105087

Scientific Books (or Scientific Book chapters)

[32] F. LOULERGUE, W. BOUSDIRA, J. TESSON. *Calcul de programmes parallèles avec Coq*, in "Informatique Mathématique", collection Alpha, CNRS Éditions, March 2015, https://hal.inria.fr/hal-01107296

Research Reports

[33] P. BOUTILLIER, S. GLONDU, B. GRÉGOIRE, H. HERBELIN, P. LETOUZEY, P.-M. PÉDROT, Y. RÉGIS-GIANAS, M. SOZEAU, A. SPIWACK, E. TASSI. *Coq 8.4 Reference Manual*, Inria, July 2014, The Coq Development Team, https://hal.inria.fr/hal-01114602

Scientific Popularization

[34] P.-L. CURIEN. *Formalisation mathématique, certification logicielle, même combat*, in "Gazette des Mathématiciens", October 2014, vol. 142, pp. 83-86, https://hal.archives-ouvertes.fr/hal-01114035

Other Publications

[35] Y. GUIRAUD, E. HOFFBECK, P. MALBOS. *Linear polygraphs and Koszulity of algebras*, June 2014, 42 pages, https://hal.archives-ouvertes.fr/hal-01006220

[36] Y. GUIRAUD, P. MALBOS. *Polygraphs of finite derivation type*, January 2014, 46 pages, https://hal.archives-ouvertes.fr/hal-00932845

References in notes

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

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

[39] 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° 3-4, pp. 475–511, Translated from the 1965 German original by Michael P. Abramson

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

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

[42] 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

[43] 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

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

[45] F. DABROWSKI, F. LOULERGUE, T. PINSARD. *Nested Atomic Sections with Thread Escape: A Formal Definition*, in "Symposium on Applied Computing (SAC)", ACM, 2014, pp. 1585-1592, http://dx.doi.org/10.1145/2554850.2554996

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

[47] P. DELIGNE. *Action du groupe des tresses sur une catégorie*, in "Invent. Math.", 1997, vol. 128, n° 1, pp. 159–175

[48] K. EMOTO, F. LOULERGUE, J. TESSON. *A Verified Generate-Test-Aggregate Coq Library for Parallel Programs Extraction*, in "Interactive Theorem Proving (ITP)", LNCS, Springer, 2014, n° 8558, pp. 258-274, http://dx.doi.org/10.1007/978-3-319-08970-6_17

[49] 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

[50] 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

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

[52] 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° 63, pp. 63-92

[53] 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

[54] 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

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

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

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

[58] 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

[59] F. LOULERGUE, S. ROBILLARD, J. TESSON, J. LÉGAUX, Z. HU. *Formal Derivation and Extraction of a Parallel Program for the All Nearest Smaller Values Problem*, in "Symposium on Applied Computing (SAC)", ACM, 2014, pp. 1577-1584, http://dx.doi.org/10.1145/2554850.2554912

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

[61] 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

[62] 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

[63] 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

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

[65] J. TITS. *A local approach to buildings*, in "The geometric vein", New York, Springer, 1981, pp. 519–547

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