IN PARTNERSHIP WITH:
CNRS

Université Denis Diderot (Paris 7)

Activity Report 2016

Project-Team PI.R2

Design, study and implementation of languages for proofs and programs

IN COLLABORATION WITH: Institut de Recherche en Informatique Fondamentale

RESEARCH CENTER
Paris

THEME
Proofs and Verification

Table of contents

Project-Team PI.R2

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

Keywords:

Computer Science and Digital Science:

2.1.1. - Semantics of programming languages
2.1.3. - Functional programming
2.1.11. - Proof languages
2.4.3. - Proofs
7.2. - Discrete mathematics, combinatorics
7.4. - Logic in Computer Science
7.6. - Computer Algebra

Other Research Topics and Application Domains:

6.1. - Software industry
6.6. - Embedded systems

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 [Inria, Emeritus Senior Researcher, HDR]
Alexis Saurin [CNRS, Researcher]
Matthieu Sozeau [Inria, Researcher]

Faculty Members

Thierry Coquand [Inria International Chair, Professor, Göteborg University]
Pierre Letouzey [Univ. Paris 7, Assistant Professor]
Yann Régis-Gianas [Univ. Paris 7, Assistant Professor]

Engineers

Daniel de Rauglaudre [Inria]
Matej Kosik [Inria, until September 2016]

PhD Students

Amina Doumane [Univ. Paris 7]
Cyrille Chenavier [Univ. Paris 7until December 2016, ATER 2016-2017]
Maxime Lucas [Univ. Paris 7]
Thibaut Girka [Univ. Paris 7, CIFRE grant (with Mitsubishi Rennes)]
Étienne Miquey [Univ. Paris 7]
Jovana Obradović [Univ. Paris 7]
Cyprien Mangin [Univ. Paris 7]
Guillaume Claret [Univ. Paris 7]
Gabriel Lewertowski [Univ. Paris 7, until September 2016]
Théo Zimmermann [Univ. Paris 7]

Visiting Scientist

Paul Aaron Steckler [MIT & Inria, from Apr 2016 until May 2016]

Administrative Assistant
Lindsay Polienor [Inria]

Others
Bruno Barras [External collaborator, Inria Saclay, Research Scientist]
Philippe Malbos [External collaborator, Univ. Lyon 1, Faculty Member]
Samuel Mimram [External collaborator, Ecole Polytechnique, Faculty Member]
Arnaud Spiwack [External collaborator, Tweag I/O]

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 [57] 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 [52], then by Howard and de Bruijn at the end of the 60's [70], [89], 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 and Huet's Calculus of Constructions [49], [50] stemmed out – a formalism that is both a logic and a programming language and that is at the source of the Coq system [87].

3.1.2. Towards the calculus of constructions

The λ-calculus, defined by Church [48], 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 [44].

To explain 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 [80].

In 1985, Coquand and Huet [49], [50] in the Formel team of Inria-Rocquencourt explored an alternative approach based on Girard-Reynolds' system F [58], [83]. 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 lists). 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 [51] 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 the formalisation of mathematics in Coq, while contributing importantly to the development of Coq, in particular nowadays for what regards user interfaces.

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. Standard library

The standard library is 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 have their core 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 combining logic and programming 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 [59] 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 [76] and Reynolds [82] and started to be studied in an abstract way in the 80's by Felleisen *et al* [55], leading to Parigot's $\lambda\mu$-calculus [81], 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 [75] and Krivine's abstract machine for call-by-name evaluation [72], [71]. 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 [56]: 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.

3.5. Effective higher-dimensional algebra

3.5.1. Higher-dimensional algebra

Like ordinary categories, higher-dimensional categorical structures originate in algebraic topology. Indeed, ∞-groupoids have been initially considered as a unified point of view for all the information contained in the homotopy groups of a topological space X: the *fundamental ∞-groupoid* $\Pi(X)$ of X contains the elements of X as 0-dimensional cells, continuous paths in X as 1-cells, homotopies between continuous paths as 2-cells, and so on. This point of view translates a topological problem (to determine if two given spaces X and Y are homotopically equivalent) into an algebraic problem (to determine if the fundamental groupoids $\Pi(X)$ and $\Pi(Y)$ are equivalent).

In the last decades, the importance of higher-dimensional categories has grown fast, mainly with the new trend of *categorification* that currently touches algebra and the surrounding fields of mathematics. Categorification is an informal process that consists in the study of higher-dimensional versions of known algebraic objects (such as higher Lie algebras in mathematical physics [43]) and/or of "weakened" versions of those objects, where equations hold only up to suitable equivalences (such as weak actions of monoids and groups in representation theory [54]).

Since a few years, the categorification process has reached logic, with the introduction of homotopy type theory. After a preliminary result that had identified categorical structures in type theory [69], it has been observed recently that the so-called "identity types" are naturally equiped with a structure of ∞-groupoid: the 1-cells are the proofs of equality, the 2-cells are the proofs of equality between proofs of equality, and so on. The striking ressemblance with the fundamental ∞-groupoid of a topological space led to the conjecture that homotopy type theory could serve as a replacement of set theory as a foundational language for different fields of mathematics, and homotopical algebra in particular.

3.5.2. Higher-dimensional rewriting

Higher-dimensional categories are algebraic structures that contain, in essence, computational aspects. This has been recognised by Street [86], and independently by Burroni [47], when they have introduced the concept of *computad* or *polygraph* as combinatorial descriptions of higher categories. Those are directed presentations of higher-dimensional categories, generalising word and term rewriting systems.

In the recent years, the algebraic structure of polygraph has led to a new theory of rewriting, called *higher-dimensional rewriting*, as a unifying point of view for usual rewriting paradigms, namely abstract, word and term rewriting [73], [79], [60], [61], and beyond: Petri nets [63] and formal proofs of classical and linear logic have been expressed in this framework [62]. Higher-dimensional rewriting has developed its own methods to analyse computational properties of polygraphs, using in particular algebraic tools such as derivations to prove termination, which in turn led to new tools for complexity analysis [46].

3.5.3. Squier theory

The homotopical properties of higher categories, as studied in mathematics, are in fact deeply related to the computational properties of their polygraphic presentations. This connection has its roots in a tradition of using rewriting-like methods in algebra, and more specifically in the work of Anick [41] and Squier [85], [84] in the 1980s: Squier has proved that, if a monoid M can be presented by a *finite*, *terminating* and *confluent* rewriting system, then its third integral homology group $H_3(M, \mathbb{Z})$ is finitely generated and the monoid M has *finite derivation type* (a property of homotopical nature). This allowed him to conclude that finite convergent rewriting systems were not a universal solution to decide the word problem of finitely generated monoids. Since then, Yves Guiraud and Philippe Malbos have shown that this connection was part of a deeper unified theory when formulated in the higher-dimensional setting [9], [10], [66], [67], [68].

In particular, the computational content of Squier's proof has led to a constructive methodology to produce, from a convergent presentation, *coherent presentations* and *polygraphic resolutions* of algebraic structures, such as monoids [9] and algebras [65]. A coherent presentation of a monoid M is a 3-dimensional combinatorial object that contains not only a presentation of M (generators and relations), but also higher-dimensional cells, each of which corresponding to two fundamentally different proofs of the same equality: this is, in essence, the same as the proofs of equality of proofs of equality in homotopy type theory. When this process of "unfolding" proofs of equalities is pursued in every dimension, one gets a polygraphic resolution of the starting monoid M. This object has the following desirable qualities: it is free and homotopically equivalent to M (in the canonical model structure of higher categories [74], [42]). A polygraphic resolution of an algebraic object X is a faithful formalisation of X on which one can perform computations, such as homotopical or homological invariants of X. In particular, this has led to new algorithms and proofs in representation theory [7], and in homological algebra [64], [65].

4. New Software and Platforms

4.1. Coq

KEYWORDS: Proof - Certification - Formalisation
FUNCTIONAL DESCRIPTION

Coq provides both a dependently-typed functional programming language and a logical formalism, which, altogether, support the formalisation of mathematical theories and the specification and certification of properties of programs. Coq also provides a large and extensible set of automatic or semi-automatic proof methods. Coq's programs are extractible to OCaml, Haskell, Scheme, ...

- Closest participants: Benjamin Grégoire, Enrico Tassi, Bruno Barras, Yves Bertot, Pierre Courtieu, Maxime Dénès, Hugo Herbelin, Matej Košík, Pierre Letouzey, Assia Mahboubi, Cyprien Mangin, Guillaume Melquiond, Jean-Marc Notin, Pierre-Marie Pédrot, Yann Régis-Gianas, Matthieu Sozeau, Arnaud Spiwack, Théo Zimmermann.
- Partners: CNRS - ENS Lyon - Université Paris-Diderot - Université Paris-Sud
- Contact: Matthieu Sozeau
- URL: http://coq.inria.fr/

4.1.1. Coq 8.6

The 8.6 version of Coq was released in December 2016. It initiates a time-based release cycle and concentrates on a smaller set of features than Coq 8.5 for which compatibility and testing were done more intensively. In the πr^2 team, Hugo Herbelin, Cyprien Mangin, Théo Zimmermann and Matthieu Sozeau contributed to the coordination of the development, to the discussion of the roadmap, to the implementation of the features, and to many bugfixes.

Matthieu Sozeau followed up his work on universe polymorphism making the explicit annotation system more accessible and resolving issues in the minimization algorithm used during refinement, resulting in a more predictable system. These improvements were used in the Coq/HoTT library for Homotopy Type Theory, which is described in an upcoming article [26].

Matthieu Sozeau implemented a new variant of the proof-search tactic for typeclasses that is set to replace the existing auto and eauto tactics in the following version. The new variant fully benefits from the features of the underlying proof engine, and allows much more control on proof-search (patterns used consistently, modes for triggering hints, ...). It is at the basis of the work of Théo Zimmermann described below.

4.1.2. The Equations plugin

Cyprien Mangin and Matthieu Sozeau continued work on the Equations plugin, modularizing it so that the use of axioms can be minimized, and making it compatible with developments in Homotopy Type Theory. To achieve this, it has moved to a simplification engine in ML based on telescopes and is able to produce axiom-free proofs of the examples that were previously implicitly using them. This work will be presented at the POPL workshop Type-Theoretic Tools (TTT), next January 2017.

4.1.3. Maintenance

Among other contributions, Hugo Herbelin, Pierre Letouzey, Matej Košík and Matthieu Sozeau worked at the maintenance of the system.

In particular, Pierre Letouzey vastly reworked the build mechanism of Coq, taking advantage of code evolutions driven by Pierre-Marie Pédrot. Pierre Letouzey also administrated (and improved) several machines or systems that are critical for the Coq community (web server, build test server, git repositories ...), in coordination with Inria's SIC support team.

Matej Košík developed a new benchmarking infrastructure based on Jenkins and continuous integration (http://ci.inria.fr). It allows easily testing any developer branch on the benchmark suite prior to integration to the main archive.

4.1.4. Coordination and animation

After 10 years coordinating the Coq development team, Hugo Herbelin handed over the coordination to Matthieu Sozeau.

A Coq working group is organised every two months (5 times a year). Discussions about the development happen, in particular, on `coq-dev@inria.fr`, Coq's GitHub http://github.com/coq and http://coq.inria.fr/bugs. This year, a week-long working group organized in Sophia-Antipolis was devoted to the 8.6 roadmap discussion.

4.1.5. Documentation and stabilization of Coq's programming interface

Matej Košík worked on the programming interfaces of Coq, starting to isolate a subset of key functions to be used by Coq plugin developers.

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 develops the "Hacking Dojo", a web platform to automatically grade programming exercises. The platform is now used in several courses of the University Paris Diderot. Yann Régis-Gianas develops a reference implementation of a syntactic analyzer for the POSIX shell programming language. This analyzer is used by the Colis project to analyze the scripts embedded in the packages of the Debian GNU/Linux distribution. In collaboration with Beta Ziliani (LIIS, Cordoba, Argentine), Yann Régis-Gianas, Béatrice Carré and Jacques-Pascal Deplaix develop MetaCoq, an extension of Coq to use Coq as a metalanguage for itself.

Yves Guiraud has updated the Catex tool for Latex, whose purpose is to automatise the production of string diagrams from algebraic expressions http://www.irif.fr/~guiraud/catex/catex.zip. Yves Guiraud collaborates with Samuel Mimram (LIX) to develop the prototype Rewr that implements several algorithms developed in the "Effective higher-dimensional algebra" research direction, including the homotopical completion-reduction procedure of [10]. An online version is available at http://www.lix.polytechnique.fr/Labo/Samuel.Mimram/rewr.

5. New Results

5.1. Effects in proof theory and programming

Participants: Hugo Herbelin, Gabriel Lewertowski, Étienne Miquey, Alexis Saurin, Matthieu Sozeau.

5.1.1. A classical sequent calculus with dependent types

Dependent types are a key feature of type systems, typically used in the context of both richly-typed programming languages and proof assistants. Control operators, which are connected with classical logic along the proof-as-program correspondence, are known to misbehave in the presence of dependent types [11], unless dependencies are restricted to values. As a step in his work to develop a sequent-calculus version of Hugo Herbelin's dPA_ω system [13], Étienne Miquey proposed a sequent calculus with classical logic and dependent types. His calculus—named dL—is an extension of the $\mu\tilde{\mu}$-calculus with a syntactical restriction of dependent types to the fragment of *negative-elimination free* proofs. The corresponding type system includes a list of explicit dependencies, which maintains type safety. He showed that a continuation-passing style translation can be derived by adding delimited continuations, and how a chain of dependencies can be related to a manipulation of the return type of this continuations. This work has been accepted for publication at ESOP 2017 [39].

5.1.2. Logical foundations of call-by-need evaluation

Alexis Saurin, in collaboration with Pierre-Marie Pédrot, extended their reconstruction of call-by-need based on linear head reduction with control. They showed how linear head reduction could be adapted to the $\lambda\mu$-calculus. This classical linear head reduction lifts the usual properties of the intuitionistic one (with respect to σ-equivalence) to the $\lambda\mu$-calculus (and its σ-equivalence already formulated by Olivier Laurent in his PhD thesis). Moreover, they showed that substitution sequences of the $\lambda\mu$-calculus' linear head reduction are in correspondence with the classical Krivine abstract machine substitution sequences, validating the known fact that the KAM implements linear head reduction. This work has been published at ESOP'16 [29]. They plan to lift to the $\lambda\mu$-calculus their three-step transformation from linear head reduction to call-by-need, and to study the correspondence with Ariola, Herbelin and Saurin's classical call-by-need.

5.1.3. Call-by-name forcing for Dependent Type Theory

Guilhem Jaber, Gabriel Lewertowski, Pierre-Marie Pédrot, Matthieu Sozeau, and Nicolas Tabareau studied a variant of the forcing translation for dependent type theory, moving from the call-by-value variant to a call-by-name version which naturally preserves definitional equalities, avoiding the coherence pitfalls of the former one. This new version was inspired by Pierre-Marie Pédrot's former decomposition of forcing in call-by-push-value. It allows to show various metatheoretical results in a succint fashion, notably for the independence of axioms. Work is ongoing to produce more positive results including abstracting reasoning on step-indexing using this technique. This work was presented at LICS 2016 [28].

5.1.4. Classical realizability and implicative algebras

Étienne Miquey has been working with Alexandre Miquel in Montevideo on the topic of implicative algebras. Implicative algebras are an algebraization of the structure needed to develop a realizability model. In particular, they give rise to the usual ordered combinatory algebras and thus to the triposes used to model classical realizability. An implicative algebra is given by an implicative structure (which consists of a complete semi-lattice with a binary operation $\rightarrow$) together with a separator containing the element interpreted as true in the structure. Étienne Miquey has been working on a formalization of implicative algebras theory in Coq. Following the work of Guillaume Munch-Maccagnoni on focalization and classical realizability, he also worked on alternative presentations within structures based on other connectives, (negation, "par", tensor), rather than $\rightarrow$. Such connectives correspond to the decomposition of the arrow according to the strategy of evaluation (call-by-name/call-by-value). The aim of this work is to obtain a classification of the possible algebraic structures to interpret classical realizability, in order to prove that different strategies of evaluation actually provide us with equivalent models.

5.2. Reasoning and programming with infinite data

Participants: Amina Doumane, Yann Régis-Gianas, Alexis Saurin.

This theme is part of the ANR project Rapido (see the National Initiatives section).

5.2.1. Proof theory of infinitary and circular proofs

In collaboration with David Baelde, Amina Doumane and Alexis Saurin developed further the theory of infinite proofs. In their study of the proof theory of circular and infinitary proofs in $\mu MALL$, they established two fundamental proof-theoretical and computational results, namely cut-elimination and focalisation. This result appeared in CSL 2016 (long version in [33]).

The usual result of focalisation for linear logic can actually be extended to circular proofs, but, contrarily to finitary $\mu MALL$ proofs where fixed-points operators can be given an arbitrary polarity, the least fixed-points must be set to be a positive construction and the greatest fixed-points to be negative, which is consistent with intuition from programming with inductive and co-inductive datatypes. An interesting phenomenon arising with focalisation is that some infinite but regular proofs may not have any regular focused proofs. This is similar to what happens for cut-elimination of regular proofs.

The proof of cut-elimination is quite involved and proceeds in two steps relying on semantic arguments, even though the paper actually proves a cut-elimination result and not only a cut-admissibility result as usual semantic arguments provide. A first part of the proof shows that some cut-reduction strategy is actually productive while a second part of the proof shows that the proof-object produced is actually a correct proof in the sense that it satisfies the validity condition of $\mu MALL$ infinite proofs. Previous cut-elimination results were only known for the restricted additive fragment of linear logic with fixed points, a result due to Santocanale and Fortier.

Baelde, Doumane and Saurin are currently working with Jaber to extend the cut-elimination result to a more expressive validity condition for $\mu MALL$ infinite proofs.

5.2.2. Automata theory meets proof theory: proof certificates for Büchi inclusion

In a joint work with David Baelde and Lucca Hirschi, Amina Doumane and Alexis Saurin carried out a proof-theoretical investigation of the linear-time μ-calculus, proposing well-structured proof systems and showing constructively that they are complete for inclusions of Büchi automata suitably encoded as formulas.

They do so in a way that combines the advantages of two lines of previous work: Kaivola gave a proof of completeness for an axiomatisation that amounts to a finitary proof system, but his proof is non-constructive and yields no reasonable procedure. On the other hand, Dax, Hofmann and Lange recently gave a deductive system that is appropriate for algorithmic proof search, but their proofs require a global validity condition and do not have a well understood proof theory.

They work with well-structured proof systems, effectively constructing proofs in a finitary sequent calculus that enjoys local correctness and cut elimination. This involves an intermediate circular proof system in which one can obtain proofs for all inclusions of parity automata, by adapting Safra's construction. In order to finally obtain finite proofs of Büchi inclusions, a translation result from circular to finite proofs is designed.

These results appeared in LICS 2016 (long version in [37]). Since then, Doumane extended the result and obtained a constructive proof of completeness for the full linear-time μ-calculus.

5.2.3. Co-patterns

In collaboration with Paul Laforgue (Master 1, University Paris Diderot), Yann Régis-Gianas studied the mechanisms of co-patterns introduced by Abel and Pientka from a programming language perspective. More precisely, they defined an untyped version of this calculus as well as an abstract machine to efficiently evaluate cofunctions. In addition, they designed several (type preserving) encodings of co-patterns using generalized algebraic datatypes and purely functional objects. Finally, they started to revisit an optimisation called "stream fusion" in a purely equational way by application of copattern-based program definitions.

5.2.4. Functional reactive programming

In collaboration with Sylvain Ribstein (Master 1, University Paris Diderot), Yann Régis-Gianas defined an OCaml library for differential functional reactive programming (DFRP). This framework extends standard functional reactive programming with the possibility to modify past events and to compute the consequences of this modification in all the events that depend on it. A paper is in preparation.

Saurin and Tasson co-advised in the spring/summer of 2016 the master internship of Rémi Nollet who started his PhD thesis under their supervision in September 2016. The topic of his thesis is the extension of Curry-Howard correspondence between FRP and LTL as recently noticed by Jeffrey and Jeltsch. During his internship, Nollet studied various proof systems for LTL and compared them to type systems for FRP. He notably studied various translations between natural deduction and sequent calculus, which led him to study precisely the role played by structural rules in those translations and preparing the work for future extensions to classical constructive LTL, and to work out the foundations for an extension of Curien-Herbelin's system L, closer to abstract machines, for LTL.

5.3. Effective higher-dimensional algebra

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

5.3.1. Rewriting and Garside theory

Yves Guiraud has collaborated with Patrick Dehornoy (LNO, Univ. Caen) to develop an axiomatic setting for monoids with a special notion of quadratic normalisation map with good computational properties. This theory generalises the normalisation procedure known for monoids that admit a special family of generators called a Garside family [53] to a much wider class that also includes the plactic monoids. It is proved that good quadratic normalisation maps correspond to quadratic convergent presentations, together with a sufficient condition for this to happen, based on the shape of the normalisation paths on length-three words. This work has been published in the International Journal of Algebra and Computation [21].

Building on this last article, Yves Guiraud currently collaborates with Matthieu Picantin (IRIF, Univ. Paris 7) to generalise the main results of Gaussent, Guiraud and Malbos on coherent presentations of Artin monoids [7], to monoids with a Garside family. This will allow an extension of the field of application of the rewriting methods to other geometrically interesting classes of monoids, such as the dual braid monoids.

Still in collaboration with Matthieu Picantin, Yves Guiraud develops an improvement of the classical Knuth-Bendix completion procedure, called the KGB completion procedure. The original algorithm tries to compute, from an arbitrary terminating rewriting system, a finite convergent presentation by adding relations to solve confluence issues. Unfortunately, this algorithm fails on standard examples, like most Artin monoids with their usual presentations. The KGB procedure uses the theory of Tietze transformations, together with Garside theory, to also add new generators to the presentation, trying to reach the convergent Garside presentation identified in [21]. The KGB completion procedure is partially implemented in the prototype Rewr, developed by Yves Guiraud and Samuel Mimram.

5.3.2. Higher-dimensional linear rewriting

With Eric Hoffbeck (LAGA, Univ. Paris 13), Yves Guiraud and Philippe Malbos have introduced in [65] the setting of linear polygraphs to formalise a theory of linear rewriting, generalising Gröbner bases. They have adapted the method of Guiraud and Malbos [9] to compute polygraphic resolutions of associative algebras, with applications to the decision of the Koszul homological property. They are currently finishing the major overhaul of this work, started in 2015, whose main goal is to ease the adaptation of the results to other algebraic varieties, like commutative algebras or Lie algebras.

Cyrille Chenavier, supervised by Yves Guiraud and Philippe Malbos, explored the use of Berger's theory of reduction operators [45] to improve the theory of Gröbner bases for associative algebras. This work has permitted to unveil two interesting algebraic structures that are hidden in rewriting theory. First, the operations that associate a normal form to an arbitrary word admit a structure of lattice, that gives a new algebraic characterisation of confluence and a new algorithm for completion, based on an iterated use of the meet-operation of the lattice. Second, under mild technical conditions, the different normalisation strategies are related through braid-like relations, as in Artin monoids, that have been used to propose a new method for a particular problem in homological algebra (namely, the construction of a contracting homotopy for the Koszul complex). The second result is published in Algebra and Representation Theory [20], the first one is submitted for publication [35], and both are contained in Cyrille Chenavier's PhD thesis [19].

5.3.3. Rewriting methods for coherence

Yves Guiraud and Philippe Malbos have written a survey on the use of rewriting methods in algebra, centered on a formulation of Squier's homotopical and homological theorems in the modern language of higher-dimensional categories. This article is intended as an introduction to the domain, mainly for graduate students, and will appear in Mathematical Structures in Computer Science [23].

Maxime Lucas, supervised by Yves Guiraud and Pierre-Louis Curien, has applied the rewriting techniques of Guiraud and Malbos [68] to prove coherence theorems for bicategories and pseudofunctors. He obtained a coherence theorem for pseudonatural transformations thanks to a new theoretical result, improving on the former techniques, that relates the properties of rewriting in 1- and 2-categories. This result is published in the Journal of Pure and Applied Algebra [25]. Maxime is currently engaged into a major rework of the results of [9], that will produce improved methods to build Squier's polygraphic resolution from a convergent presentation, based on the use of cubical higher categories instead of globular ones. He has already achieved a first result in this direction [77], and conducted a major foundational work towars the full result [78], which have just been submitted for publication.

Pierre-Louis Curien and Jovana Obradović pursued their work on cyclic operads (started in [36], now acepted in the Journal Applied Categorical Structures). They established the notion of categorified cyclic operad. Categorification involves weakening the axioms of cyclic operads (from equalities to natural isomorphisms) and formulating conditions concerning these isomorphisms which ensure coherence. For entries-only cyclic operads, this coherence is of the same kind as the coherence of symmetric monoidal categories: all diagrams made of associator and commutator isomorphisms are required to commute. However, in the setting of cyclic operads, where the existence of objects and morphisms depends on the shape of a fixed unrooted tree, these arrows do not always exist. In other words, the coherences that Mac Lane established for symmetric monoidal categories do not solve the coherence problem of categorified cyclic operads. They exhibited the appropriate conditions of this setting and proved the coherence theorem, relying on a result of Došen and

Petrić, coming from the coherence of categorified operads. Additionally, by the equivalence between the two possible characterisations of cyclic operads, for cyclic operads introduced as operads with extra structure (that exchanges the output of an operation with one of its inputs), i.e. for exchangeable-output cyclic operads, they examined which of the axioms of the extra structure needs to be weakened (in order to lift that equivalence to weakened structures), and they exhibited the appropriate coherence conditions in this setting as well.

5.4. Incrementality

Participants: Thibaut Girka, Yann Régis-Gianas.

5.4.1. Incrementality in proof languages

In collaboration with Paolo Giarrusso and Yufei Cai (Univ Marburg, Allemagne), Yann Régis-Gianas developed a new method to incrementalise higher-order programs using formal derivatives and static caching. Yann Régis-Gianas has developed a mechanized proof for this transformation. A paper will be submitted to ICFP 2017.

5.4.2. Difference languages

In collaboration with David Mentré (Mitsubishi), Thibaut Girka and Yann Régis-Gianas have developed a theoretical framework to define a notion of differential operational semantics: a general mathematical object to characterise the difference of behavior of two close programs. A paper is under submission. A technical report is available [8].

Thibaut Girka and Yann Régis-Gianas presented this work in several working groups: Gallium (Paris), “Journée annuelle du groupe LTP” of the GDR GPL (Saclay), LIMA (Nantes), IRIF (Paris).

5.5. Metatheory and development of Coq

Participants: Hugo Herbelin, Pierre Letouzey, Yann Régis-Gianas, Matthieu Sozeau.

5.5.1. Dependent pattern-matching

Hugo Herbelin supervised the internship of Meven Bertrand on compiling dependent pattern-matching using a combination of techniques known as small inversion and generalization, as a following of Pierre Boutillier’s PhD.

5.5.2. Transferring theorems along isomorphisms

Théo Zimmermann has developed a tool for transferring theorems along isomorphic structures. The long-term objective is to provide a language of proof methods matching the level of abstraction common in mathematics. Théo Zimmermann is applying his tool to introduce higher "mathematical" levels of abstraction to the basic Coq method for applying theorems. The proof of concept of this idea will be presented at the TTT POPL workshop in January.

5.5.3. Unification

Matthieu Sozeau worked in collaboration with Beta Ziliani (assistant professor at Córdoba, Argentina) on a journal version of the formalisation of the unification algorithm used in Coq, which is central for working with advanced type inference features like Canonical Structures. The presentation of this journal version is incremental (it is presented feature by feature), with an aim of easing the understanding of how the algorithm actually works for users who want to take advantage of it. It has been accepted for publication in the Journal of Functional Programming.

5.5.4. Explicit Cumulativity

Pierre Letouzey started exploring with the help of Matthieu Sozeau a version of Coq's logic (CIC) where the cumulativity rule would be explicit. This cumulativity rule is a form of coercion between Coq universes, and is done silently in Coq up to now. Having a version of CIC where the use of the cumulativity bewteen Prop and Type is traceable woud be of great interest. In particular this would lead to a solid ground for the Coq extraction tool and solve some of its current limitations. Moreover, an explicit cumulativity would also help significantly the studies of Coq theoretical models. Preliminary results are encouraging, but this work has not been finalized yet. This work is related to the studies of Ali Assaf (Google Zurich, formerly PhD student in the team Deducteam), but uses different technical choices for different goals. This work is now pursued by Gae¨tan Gilbert (PhD student of Nicolas Tabareau and Matthieu Sozeau at the E´cole des Mines in Nantes), with the goal of providing a version of the calculus of constructions with definitional proof-irrelevance. The absence of explicit cumulativity between Prop and Type was identified in earlier work by Benjamin Werner and Giesik Lee as an important obstacle to building models of the theory, we hence expect this work to simplify the (relative) consistency proof of the theory.

5.6. Formalisation work

Participants: Jean-Jacques Lévy, Daniel de Rauglaudre.

5.6.1. Proofs of algorithms on graphs

Jean-Jacques Lévy and Chen Ran (a PhD student of the Institute of Software, Beijing, visiting the Toccata team) pursue their work about formal proofs of algorithms. Their goal is to provide proofs of algorithms which ought to be both checked by computer and easily human readable. If these kinds of proofs exist for algorithms on inductive structures or recursive algorithms on arrays, they seem less easy to design for combinatorial structures such as graphs. In 2016, they completed proofs for algorithms computing the strongly connected components in graphs. There are mainly two algorithms: one by Kosaraju (1978) working in two phases (some formal proofs of it have already been achieved by Pottier with Coq-classic and by Théry and Gonthier with Coq-ssreflect), one by Tarjan (1972) working in a single pass.

Their proofs use a first-order logic with definitions of inductive predicates. This logic is the one defined in Why3 (research-team Toccata, Saclay). They widely use automatic provers interfaced by Why3. A very minor part of these proofs is also achieved in Coq. The difficulty of this approach is to combine automatic provers and intuitive design.

Part of this work (Tarjan 1972) is presented at JFLA 2017 in Gourette [30] A more comprehensive version is under submission to another conference [34]. Scripts of proofs can be found at http://jeanjacqueslevy.net/why3.

5.6.2. Formalization of theorems in Coq

This section reports on formalisation work by Daniel de Rauglaudre.

5.6.2.1. Puiseux' Theorem

Puiseux' theorem states that the set of Puiseux series (series with rational powers) is an algebraically closed field, i.e. every non-constant polynomial with Puiseux series coefficients admits a zero. This theorem was formalized in Coq a couple of years ago, but it depended on five ad hoc axioms. This year, all these axioms have been grouped together into the only axiom LPO (Limited Principle of Omniscience), stating that for each sequence of booleans, we can decide whether it is always false or if there is at least one true element. This formalized theorem now depends only on this axiom.

5.6.2.2. Banach-Tarski Paradox

Banach-Tarski Paradox states that, if we admit the axiom of choice, a sphere is equidecomposable into two spheres identical to the initial one. The equidecomposability is a property of geometric objects: two objects (sets) are equidecomposable if we can partition them into a same finite number of sets, and each set of the first object is mapped to a set of the second object by only rotations and translations. In other words, we break the first object into a finite number of pieces, and with them, we reconstitute the second object. Its pen and paper proof was done in 1924 by Banach and Tarski.

Its formal proof in Coq has been started this year. About 80% of the proof has been done. The already proved part includes a lemma which says that the sphere without some specific countable number of points is equidecomposable into twice itself. It also includes a formal proof that equidecomposability is an equivalence relation. This makes about 7000 lines of Coq. The remaining part is to formalize the proof that the sphere is equidecomposable into the sphere without this countable set of points.

The version of axiom of choice used for this proof is named TTCA (Type Theoretical Axiom of Choice, introduced by Benjamin Werner [88]), stating that for each equivalence relation, there exists a function mapping each relation class to one of its elements.

6. Partnerships and Cooperations

6.1. National Initiatives

Alexis Saurin (coordinator) and Yann Régis-Gianas are members of the four-year RAPIDO ANR project, started 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 IRIF (PPS team), David Baelde from LSV, ENS-Cachan, and Pierre Clairambault, Damien Pous and Colin Riba from LIP, ENS-Lyon.

Pierre-Louis Curien (coordinator), Yves Guiraud (local coordinator), Philippe Malbos and Samuel Mimram have been members of the three-year Focal project of the IDEX Sorbonne Paris Cité (July 2013 to June 2016). 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 joint with mathematicians form LAGA (Univ. Paris 13).

Pierre-Louis Curien (coordinator), Yves Guiraud (local coordinator), Philippe Malbos and Samuel Mimram are members of the four-year Cathre ANR project, started in January 2014. This project, giving the support for the PhD grant of Maxime Lucas, 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 linear algebra, combinatorial group theory and theoretical computer science. This project is joint with mathematicians and computer scientists from LAGA (Univ. Paris 13), LIX (École Polytechnique), ICJ (Univ. Lyon 1 and Univ. Saint-Étienne), I2M (Univ. Aix-Marseille) and IMT (Univ. Toulouse 3).

Pierre-Louis Curien, Yves Guiraud, Hugo Herbelin, Philippe Malbos, Samuel Mimram and Alexis Saurin are members of the GDR Informatique Mathématique, in the Géocal (Geometry of computation) and LAC (Logic, algebra and computation) working groups.

Pierre-Louis Curien, Yves Guiraud (local coordinator), Philippe Malbos, Samuel Mimram and Matthieu Sozeau are members of the GDR Topologie Algébrique, federating French researchers working on classical topics of algebraic topology and homological algebra, such as homotopy theory, group homology, K-theory, deformation theory, and on more recent interactions of topology with other themes, such as higher categories and theoretical computer science.

Hugo Herbelin was the coordinator of the PPS site for the ANR Récré (January 2012 to mid 2016). Récré is about realisability and rewriting, with applications to proving with side-effects and concurrency.

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.

Yann Régis-Gianas is a member of the ANR COLIS dedicated to the verification of Linux Distribution installation scripts. This project is joint with members of VALS (Univ Paris Sud) and LIFL (Univ Lille).

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. The PhD grant of Gabriel Lewertowski was funded by the CoqHoTT ERC.

6.2. European Initiatives

6.2.1. FP7 & H2020 Projects

Hugo Herbelin is a deputy representative of France in the COST action EUTYPES.

6.3. International Initiatives

6.3.1. Inria Associate Teams Not Involved in an Inria International Labs

Pierre-Louis Curien participates to the Associated Team CRECOGI (Concurrent, Resourceful and Effectful Computation, by Geometry of Interaction) between the project-team Focus (Bologna) and the University of Tokyo (principal investigators Ugo dal Lago and Ichiro Hasuo, started in 2015).

6.3.2. Inria International Partners

6.3.2.1. Informal International Partners

The project-team has collaborations with University of Aarhus (Denmark), University of Oregon, University of Tokyo, University of Sovi Sad and the Institute of Mathematics of the Serbian Academy of Sciences, University of Nottingham, Institute of Advanced Study, MIT, University of Cambridge, and Universidad Nacional de Córdoba.

6.3.3. Participation in Other International Programs

Pierre-Louis Curien participates to the ANR International French-Chinese project LOCALI (Logical Approach to Novel Computational Paradigms), coordinated by Gilles Dowek (Deducteam).

6.4. International Research Visitors

6.4.1. Visits of International Scientists

Paolo Giarrusso (Univ. of Marburg) visited Yann Régis-Gianas in February 2016.

Lourdes del Carmen Gonzalez Huesca (Univ. of Mexico) visited Yann Régis-Gianas in December 2016.

6.4.2. Visits to International Teams

6.4.2.1. Research Stays Abroad

Pierre-Louis Curien visited the Category Theory group at Macquarie University in June-July 2016 (collaborative work on the combinatorial structure of type dependency).

As a part of his joint PhD, Étienne Miquey worked most of the year in Montevideo within the Logic group of the Universidad de la República of Uruguay.

7. Dissemination

7.1. Promoting Scientific Activities

7.1.1. Scientific Events Organisation

7.1.1.1. Member of the Organizing Committees

Yann Régis-Gianas is multimedia chair of the organizing committee of POPL 2017 that will be held in Paris in January 2017.

Yves Guiraud, Philippe Malbos and Samuel Mimram have organised the second edition of the Higher-Dimensional Rewriting and Applications (HDRA) workshop of the Formal Structures for Computation and Deduction conference (FSCD), held in Porto in June 2016. They plan to organise the third edition of HDRA, still with FSCD, in September 2017 in Oxford.

Yves Guiraud and Alexis Saurin, with Christine Tasson (IRIF), have organised the annual meeting of the Géocal and LAC working groups of the GDR Informatique Mathématique in Paris, in November 2016.

Yves Guiraud and Samuel Mimram, with Dimitri Ara (Univ. Aix-Marseille) are currently organising the Categories in Homotopy and Rewriting one-week conference, that will be held at the CIRM, in Marseille, in September 2017.

7.1.2. Scientific Events Selection

7.1.2.1. Member of the Conference Program Committees

Matthieu Sozeau was member of the program committees of FSCD'16, ITP'16 and CoqPL'16.

7.1.2.2. Member of the Conference Steering Committees

Hugo Herbelin is a member of the steering committee of the conference *Formal Structures for Computation and Deduction* (FSCD).

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

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

7.1.3. Journal

7.1.3.1. Member of the Editorial Boards

Pierre-Louis Curien is editor in chief of the Cambridge University Press journal Mathematical Structures in Computer Science (since January 2016).

7.1.3.2. Reviewer - Reviewing Activities

The members of the team reviewed papers for numerous journals and international conferences.

7.1.4. Invited Talks

Pierre-Louis Curien and Samuel Mimram gave invited talks at the annual meeting of the Géocal and LAC working groups of the GDR Informatique Mathématique (Paris, November).

Pierre-Louis Curien gave an invited talk at the annual meeting of the international ANR project Pace (between Univ. of Bologna, ENS Lyon and Shanghai Jiaotong University) on "Categorified cyclic operads" (Shanghai, November).

Hugo Herbelin gave an invited talk on “Proving with side-effects” at the Days in Logic meeting in Lisbon, Portugal.

Jean-Jacques Lévy gave an invited talk about “Strongly connected components in graphs, Formal proof of Tarjan 1972 algorithm” at the LTP (Langages, Types et Preuves) day, Saclay [38].

Matthieu Sozeau gave invited talks at the DeepSpec kickoff meeting in Princeton, NJ, USA, June 8th 2016, on “Coq 8.6” (together with Maxime Dénès), at the International Conference on Mathematical Software in Berlin, Germany, July 14th 2016, on “Coq for HoTT”, at the Categorical Logic and Univalent Foundations workshop, Leeds, UK, July 28th 2016, on “Forcing Translations in Type Theory”, and at the Coq Workshop in Nancy, France, August 26th 2016, on “Coq 8.6”.

7.1.5. Scientific Expertise

Pierre-Louis Curien has been member of the “Comité de Sélection” for a professor position in discrete mathematics at the University Paul Sabatier in Toulouse.

Yann Régis-Gianas and Hugo Herbelin have been members of the “Comité de Sélection” for an assistant professor position at CNAM in Paris.

Yann Régis-Gianas has been member of the “Comité de Sélection” for an assistant professor position at IRIF in Paris.

Hugo Herbelin has been member of the “Comité de Sélection” for a starting researcher position at Inria Saclay.

7.1.6. Scientific expertise

Pierre-Louis Curien is a member of the Scientific Committee of the CIRM (since June 2013).

7.1.7. Research Administration

Pierre-Louis Curien, Hugo Herbelin and Yves Guiraud are members of the scientific council of the Computer Science deparment of University Paris 7.

Yves Guiraud is the head of the Preuves, Programmes and Systèmes (PPS) team of the IRIF laboratory (since April 2016), and a member of the IRIF council (since January 2016).

7.1.8. Presentation of papers

Étienne Miquey gave a talk on a computational reduction of dependent choice in classical logic to system F at TYPES’16 (Novi Sad, Serbia, May 2016).

Étienne Miquey gave a talk on realizability games for the specification problem during the workshop Realizability in Uruguay 2016 (Piriápolis, Uruguay, July 2016).

Cyrille Chenavier gave a talk at the workshop IWC, Obergurgl, Austria (September 2016).

Cyrille Chenavier, Maxime Lucas and Jovana Obradović gave talks at the workshop Categories, Homotopy and Rewriting (Toulouse, January) and at the workshop HDRA (Porto, June).

Jovana Obradović presented her works on cyclic operads at the Types Conference 2016 (Novi Sad, Serbia, May 2016) and at the Conference Logic and Applications 2016 (Dubrovnik, Croatia, September).

Hugo Herbelin gave a talk on proving Gödel’s completeness theorem with side-effects at the Mathematics for Computation workshop in Niederalteich, Germany, May 2016.

7.1.9. Talks in seminars

Pierre-Louis Curien gave a talk at the Séminaire de Topologie of the University of Angers on the semantics of dependent types (January).

Yves Guiraud gave a talk in the Séminaire de Combinatoire of the University Paris 7 on an introduction to Squier’s theory (November).

Hugo Herbelin gave a talk on a proof-as-program interpretation of the classical axiom of dependent choice at the Séminaire "Logique et Interactions" of the "Logique de la Programmation" team of the "Institut de Mathématiques de Marseille" (University Aix-Marseille, February).

Yann Régis-Gianas gave a talk about control operators in the history of programming at the Séminaire "Code Sources" organized by Baptiste Mélès.

Yann Régis-Gianas gave a talk about the writing style in programming at the conference "Current issues in the philosophy of practice of mathematics and informatics" (University of Toulouse, April).

Thibaut Girka gave a talk about difference languages at the Gallium seminar (Paris, September 2016) and at the TLP group of the GDR GPL (Saclay, November 2016).

Yann Régis-Gianas gave a talk about difference languages at the LIMA laboratory (Nantes, October 2016) and at the Semantic Working Group of IRIF (Paris, December 2016).

Matthieu Sozeau gave a talk about Equations: a function definition toolbox for Coq at Dagstuhl in March 2016.

Cyrille Chenavier gave a talk about confluence algebras at the Algebra working group of the LMPA, Calais, in February 2016.

Jovana Obradović gave a talk about categorified cyclic operads at the Proof Theory Seminar of the Mathematical Institute of the Serbian Academy of Sciences and Arts (Belgrade, December 2016).

7.1.10. Attendance to conferences, workshops, schools,...

Pierre-Louis Curien attended the conferences Types 2016 in Novi Sad (Serbia, May) and Logic and Applications in Dubrovnik (Croatia, September).

Cyrille Chenavier, Pierre-Louis Curien, Yves Guiraud, Maxime Lucas, Philippe Malbos, Samuel Mimram and Jovana Obradović attended the Category, Homotopy and Rewriting workshop in Toulouse (January 2016).

Cyrille Chenavier, Maxime Lucas, Philippe Malbos and Samuel Mimram attended the HDRA workshop in Lisbon (June 2016).

Hugo Herbelin attended the Days in Logic meeting in Lisbon (Portugal, January), the Mathematics for Computation workshop in Niederalteich (Germany, May), the conferences Types 2016 in Novi Sad (Serbia, May), the Coq coding sprint in Sophia-Antipolis (May-June), the DeepSpec kick-off meeting in Princeton (USA, June), the FSCD conference in Porto (Portugal, June), the Coq workshop and ITP 2016 (Nancy, August), as well as the Dagstuhl seminar on universality of proofs (October).

Jean-Jacques Lévy participated to CPP and POPL 2016 conferences, Saint Petersburgh, USA, January 18-22, and the Robin Milner Award reception, the Royal Society, London, November 24 (X. Leroy (research team Gallium) was awarded).

Matthieu Sozeau attended POPL 2016, ICMS 2016, ITP 2016, the Coq coding sprint, the DeepSpec kick-off meeting in Princeton as well as the Dagstuhl seminar on proofs of functional programs (March).

Théo Zimmermann attended the conference CICM 2016 in Białystok (Poland, July). He gave a talk there to present his PhD subject. He also attended the Coq coding sprint.

7.1.11. Groupe de travail Théorie des types et réalisabilité

This is one of the working groups of PPS, jointly organised by Hugo Herbelin and Matthieu Sozeau.

7.1.12. Groupe de travail Catégories supérieures, polygraphes et homotopie

Several members of the team participate actively in this weekly working group of PPS, organised by François Métayer (IRIF) since 2009.

7.2. Teaching - Supervision - Juries

7.2.1. Teaching

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: Hugo Herbelin teaches the course on the proof-as-program correspondence for classical logic and beyond at the LMFI.

Master: Pierre Letouzey teaches two short courses to the LMFI Master 2 students : “Models of programming” and ”Introduction to computed-aided formal proofs”. These two courses come in addition to Pierre Letouzey’s regular duty as teacher in the Computer Science department of Paris 7 (including a course on Compilation to M2-Pro students).

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

Master: Matthieu Sozeau taught the MPRI course on Advanced uses of proof assistants (12 hours + a project), together with Assia Mahboubi (Inria SpecFun).

MOOC: In collaboration with Roberto Di Cosmo and Ralf Treinen, Yann Régis-Gianas has created a MOOC about the OCaml programming language. The first edition took place in 2015, the second edition in 2016.

7.2.2. Supervision

Internship: Yves Guiraud has supervised the M2 internship of Amina Bendjaafar.

Internship: Hugo Herbelin has supervised the L3 internship of Meven Bertrand.

Internship: Hugo Herbelin has supervised the pre-doctoral internship of Théo Zimmermann.

Internship: Yann Régis-Gianas has supervised the M1 internship of Paul Laforgue.

Internship: Yann Régis-Gianas has supervised the M1 internship of Sylvain Ribstein.

PhD (completed): Cyrille Chenavier, supervised by Yves Guiraud and Philippe Malbos, successfully defended in December 2016

PhD in progress: Guillaume Claret, Programmation avec effets en Coq, (started in September 2012), supervised by Hugo Herbelin and Yann Régis-Gianas, defense planned in February 2017.

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

PhD in progress: Thibaut Girka, Differential semantics (started in January 2014), supervised by Roberto Di Cosmo and Yann Régis-Gianas.

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

Phd in progress: Cyprien Mangin, Dependent Pattern-Matching, induction-induction and higher inductive types, September 2015, supervised by Matthieu Sozeau and Bruno Barras.

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

PhD in progress: Jovana Obradović, Cyclic operads: syntactic, algebraic and categorified aspects, supervised by Pierre-Louis Curien.

PhD stopped: Gabriel Lewertowski, On forcing in type theory, supervised by Matthieu Sozeau and Nicolas Tabareau. Gabriel stopped his PhD in september 2016 and is now working at la Pitié Salpêtrière as an engineer.

PhD starting: Gae¨tan Gilbert, Definitional Proof Irrelevance, supervised by Nicolas Tabareau and Matthieu Sozeau.

PhD starting: Théo Zimmermann, supervised by Hugo Herbelin.

7.2.3. Juries

Pierre-Louis Curien was referee for the habilitations of Emmanuel Haucourt (Paris 7, September) and Samuel Mimram (Paris 7, September). He was president of the jury of the thesis of Matteo Acclavio (Univ. de la Méditerranée, December).

Pierre-Louis Curien (president), Yves Guiraud and Philippe Malbos were members of the jury of the thesis of Cyrille Chenavier (Univ. Paris 7, December).

Hugo Herbelin was referee for the habilitation of Nicolas Tabareau (Nantes, November). He was a referee of the jury of the thesis of Jirka Maršík (LORIA, December).

Matthieu Sozeau was a member of the jury of the thesis of Kevin Quirin (EMN Nantes, December).

Yann Régis-Gianas is a member of the jury of the competitive examination for the entrance to the Écoles Normales Supérieures and the École Polytechnique.

7.3. Popularization

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, and Étienne Miquey took part in the animation of the (computer science part of the) “Fête de la Science” event at the University Paris 7. Yann Régis-Gianas gave several presentations about “What is programming?” in primary and high schools of Paris and its region.

8. Bibliography

Major publications by the team in recent years

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

[2] 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

[3] P.-L. CURIEN. *Operads, clones, and distributive laws*, in "Operads and Universal Algebra : Proceedings of China-France Summer Conference", Tianjin, China, L. G. CHENGMING BAI, J.-L. LODAY (editors), Nankai Series in Pure, Applied Mathematics and Theoretical Physics, Vol. 9, World Scientific, July 2010, pp. 25-50, https://hal.archives-ouvertes.fr/hal-00697065

[4] 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, http://dx.doi.org/10.1007/s10990-007-9006-0

[5] 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/

[6] P.-L. CURIEN, H. HERBELIN. *Abstract machines for dialogue games*, in "Interactive models of computation and program behavior", Panoramas et Synthèses, Société Mathématique de France, 2009, pp. 231-275, https://hal.archives-ouvertes.fr/hal-00155295

[7] S. GAUSSENT, Y. GUIRAUD, P. MALBOS. *Coherent presentations of Artin monoids*, in "Compositio Mathematica", 2015, vol. 151, n^{o} 5, pp. 957-998 [*DOI :* 10.1112/S0010437X14007842], https://hal.archives-ouvertes.fr/hal-00682233

[8] T. GIRKA, D. MENTRÉ, Y. RÉGIS-GIANAS. *Oracle-based Dierential Operational Semantics (long version)*, Université Paris Diderot / Sorbonne Paris Cité, October 2016, https://hal.inria.fr/hal-01419860

[9] 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

[10] 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

[11] H. HERBELIN. *On the Degeneracy of Sigma-Types in Presence of Computational Classical Logic*, in "Proceedings of TLCA 2005", P. URZYCZYN (editor), Lecture Notes in Computer Science, Springer, 2005, vol. 3461, pp. 209–220

[12] 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/

[13] H. HERBELIN. *A Constructive Proof of Dependent Choice, Compatible with Classical Logic*, in "LICS 2012 - 27th Annual ACM/IEEE Symposium on Logic in Computer Science", Dubrovnik, Croatia, Proceedings of the 27th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2012, 25-28 June 2012, Dubrovnik, Croatia, IEEE Computer Society, June 2012, pp. 365-374, https://hal.inria.fr/hal-00697240

[14] G. JABER, N. TABAREAU, M. SOZEAU. *Extending Type Theory with Forcing*, in "LICS 2012 : Logic In Computer Science", Dubrovnik, Croatia, June 2012, https://hal.archives-ouvertes.fr/hal-00685150

[15] 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

[16] 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

[17] 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

[18] 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

[19] C. CHENAVIER. *The lattice of reduction operators: applications to noncommutative Gröbner bases and homological algebra*, Université paris Diderot, December 2016, https://tel.archives-ouvertes.fr/tel-01415910

Articles in International Peer-Reviewed Journals

[20] C. CHENAVIER. *Confluence Algebras and Acyclicity of the Koszul Complex*, in "Algebras and Representation Theory", 2016 [*DOI :* 10.1007/s10468-016-9595-6], https://hal.archives-ouvertes.fr/hal-01141738

[21] P. DEHORNOY, Y. GUIRAUD. *Quadratic normalisation in monoids*, in "International Journal of Algebra and Computation", August 2016, vol. 26, n° 5, pp. 935-972 [*DOI :* 10.1142/S0218196716500399], https://hal.archives-ouvertes.fr/hal-01141226

[22] M. GUILLERMO, É. MIQUEY. *Classical realizability and arithmetical formulæ*, in "Mathematical Structures in Computer Science", 2016 [*DOI :* 10.1017/S0960129515000559], https://hal.inria.fr/hal-01247989

[23] Y. GUIRAUD, P. MALBOS. *Polygraphs of finite derivation type*, in "Mathematical Structures in Computer Science", September 2016, in press, 46 pages [*DOI :* 10.1017/S0960129516000220], https://hal.archives-ouvertes.fr/hal-00932845

[24] F. LOULERGUE, W. BOUSDIRA, J. TESSON. *Calculating Parallel Programs in Coq using List Homomorphisms*, in "International Journal of Parallel Programming", 2016, 20 p. , https://hal.inria.fr/hal-01159182

[25] M. LUCAS. *A coherence theorem for pseudonatural transformations*, in "Journal of Pure and Applied Algebra", 2017, vol. 221, n° 5, pp. 1146-1217 [*DOI :* 10.1016/J.JPAA.2016.09.005], https://hal.archives-ouvertes.fr/hal-01191867

International Conferences with Proceedings

[26] A. BAUER, G. JASON, P. LUMSDAINE, M. SHULMAN, M. SOZEAU, B. SPITTERS. *The HoTT Library: A Formalization of Homotopy Type Theory in Coq*, in "CPP'17", Paris, France, CPP'17, ACM, January 2017, 9 p. [*DOI :* 10.1145/3018610.3018615], https://hal.inria.fr/hal-01421212

[27] P.-L. CURIEN, M. FIORE, G. MUNCH-MACCAGNONI. *A theory of effects and resources: adjunction models and polarised calculi*, in "Principles of Programmming Languages", Saint-Petersbourg, Florida, United States, Proceedings POPL 2016, January 2016 [*DOI :* 10.1145/2837614.2837652], https://hal.archives-ouvertes.fr/hal-01256092

[28] G. JABER, G. LEWERTOWSKI, P.-M. PÉDROT, M. SOZEAU, N. TABAREAU. *The Definitional Side of the Forcing*, in "Logics in Computer Science", New York, United States, May 2016 [*DOI :* 10.1145/2933575.2935320], https://hal.archives-ouvertes.fr/hal-01319066

[29] P.-M. PÉDROT, A. SAURIN. *Classical by-need*, in "European Symposium on Programming", Eindhoven, Netherlands, European Symposium on Programming, April 2016, https://hal.archives-ouvertes.fr/hal-01257348

National Conferences with Proceedings

[30] R. CHEN, J.-J. LÉVY. *Une preuve formelle de l'algorithme de Tarjan-1972 pour trouver les composantes fortement connexes dans un graphe*, in "JFLA 2017 - Vingt-huitièmes Journées Francophones des Langages Applicatifs", Gourette, France, Vingt-huitièmes Journées Francophones des Langages Applicatifs, January 2017, https://hal.inria.fr/hal-01422215

Conferences without Proceedings

[31] H. HERBELIN, É. MIQUEY. *A continuation-passing-style interpretation of simply-typed call-by-need λ-calculus with control within System F*, in "CL&C'16. Sixth International Workshop on. Classical Logic and Computation", Porto, Portugal, June 2016, https://hal.inria.fr/hal-01302696

Research Reports

[32] T. GIRKA, D. MENTRÉ, Y. RÉGIS-GIANAS. *Oracle-based Dierential Operational Semantics (long version)*, Université Paris Diderot / Sorbonne Paris Cité, October 2016, https://hal.inria.fr/hal-01419860

Other Publications

[33] D. BAELDE, A. DOUMANE, A. SAURIN. *Infinitary proof theory : the multiplicative additive case* , June 2016, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01339037

[34] R. CHEN, J.-J. LEVY. *Formal proofs of two algorithms for strongly connected components in graphs*, November 2016, working paper or preprint, https://hal.inria.fr/hal-01422216

[35] C. CHENAVIER. *Reduction Operators and Completion of Rewriting Systems*, June 2016, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01325907

[36] P.-L. CURIEN, J. OBRADOVIC. *On the various definitions of cyclic operads*, January 2016, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01254649

[37] A. DOUMANE, D. BAELDE, L. HIRSCHI, A. SAURIN. *Towards Completeness via Proof Search in the Linear Time mu-Calculus: The case of Büchi inclusions*, January 2016, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01275289

[38] J.-J. LEVY, R. CHEN. *Strongly Connected Components in graphs, formal proof of Tarjan1972 algorithm*, November 2016, Groupe de travail LTP du GDR GPL , Travail présenté à JFLA 2017, https://hal.inria.fr/hal-01422227

[39] É. MIQUEY. *A Classical Sequent Calculus with Dependent Types* , December 2016, working paper or preprint, https://hal.inria.fr/hal-01375977

[40] J. OBRADOVIC. *The Bénabou-Roubaud monadic descent theorem via string diagrams*, January 2016, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01254637

References in notes

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

[42] D. ARA, F. MÉTAYER. *The Brown-Golasiński Model Structure on strict ∞-groupoids revisited*, in "Homology, Homotopy and Applications", 2011, vol. 13, n^{o} 1, pp. 121–142

[43] J. BAEZ, A. CRANS. *Higher-dimensional algebra. VI. Lie 2-algebras*, in "Theory Appl. Categ.", 2004, vol. 12, pp. 492–538

[44] H. P. BARENDREGT. *The Lambda Calculus: Its Syntax and Semantics*, North Holland, Amsterdam, 1984

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

[46] G. BONFANTE, Y. GUIRAUD. *Polygraphic Programs and Polynomial-Time Functions*, in "Logical Methods in Computer Science", 2009, vol. 5, n^{o} 2, pp. 1–37

[47] A. BURRONI. *Higher-dimensional word problems with applications to equational logic*, in "Theoretical Computer Science", jul 1993, vol. 115, n^{o} 1, pp. 43–62

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

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

[50] 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

[51] 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

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

[53] P. DEHORNOY, F. DIGNE, E. GODELLE, D. KRAMMER, J. MICHEL. *Foundations of Garside theory*, EMS Tracts Math., European Mathematical Society, 2015, vol. 22, xiv + 689 pages p.

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

[55] 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

[56] 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

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

[58] 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

[59] 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

[60] Y. GUIRAUD. *Présentations d'opérades et systèmes de réécriture*, Univ. Montpellier 2, 2004

[61] Y. GUIRAUD. *Termination Orders for 3-Dimensional Rewriting*, in "Journal of Pure and Applied Algebra", 2006, vol. 207, n^{o} 2, pp. 341–371

[62] Y. GUIRAUD. *The Three Dimensions of Proofs*, in "Annals of Pure and Applied Logic", 2006, vol. 141, n^{o} 1–2, pp. 266–295

[63] Y. GUIRAUD. *Two Polygraphic Presentations of Petri Nets*, in "Theoretical Computer Science", 2006, vol. 360, n^{o} 1–3, pp. 124–146

[64] 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

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

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

[67] Y. GUIRAUD, P. MALBOS. *Identities among relations for higher-dimensional rewriting systems*, in "Séminaires et Congrès, Société Mathématique de France", 2011, vol. 26, pp. 145-161

[68] Y. GUIRAUD, P. MALBOS. *Coherence in monoidal track categories*, in "Math. Structures Comput. Sci.", 2012, vol. 22, n^{o} 6, pp. 931–969

[69] M. HOFMANN, T. STREICHER. *The groupoid interpretation of type theory*, in "Twenty-five years of constructive type theory (Venice, 1995)", Oxford Logic Guides, Oxford Univ. Press, New York, 1998, vol. 36, pp. 83–111

[70] 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

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

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

[73] Y. LAFONT. *Towards an Algebraic Theory of Boolean Circuits*, in "Journal of Pure and Applied Algebra", 2003, vol. 184, pp. 257-310

[74] Y. LAFONT, F. MÉTAYER, K. WORYTKIEWICZ. *A Folk Model Structure on Omega-Cat*, in "Advances in Mathematics", 2010, vol. 224, n^{o} 3, pp. 1183–1231

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

[76] 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

[77] M. LUCAS. *A cubical Squier's theorem*, 2016, 6 pages, arXiv:1612.06541

[78] M. LUCAS. *Cubical (omega,p)-categories*, 2016, 38 pages, arXiv:1612.07050

[79] P. MALBOS. *Critères de finitude homologique pour la non convergence des systèmes de réécriture de termes*, Univ. Montpellier 2, 2004

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

[81] 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

[82] 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

[83] 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

[84] 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

[85] C. C. SQUIER. *Word problems and a homological finiteness condition for monoids*, in "J. Pure Appl. Algebra", 1987, vol. 49, n^{o} 1-2, pp. 201–217

[86] R. STREET. *Limits Indexed by Category-Valued 2-Functors*, in "Journal of Pure and Applied Algebra", 1976, vol. 8, pp. 149–181

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

[88] B. WERNER. *Sets in types, types in sets*, M. ABADI, T. ITO (editors), Springer Berlin Heidelberg, Berlin, Heidelberg, 1997, pp. 530–546, http://dx.doi.org/10.1007/BFb0014566

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