RESEARCH CENTRE

**Saclay - Île-de-France**

**IN PARTNERSHIP WITH:**

**ENS Paris-Saclay**

2020

ACTIVITY REPORT

Project-Team

# DEDUCTEAM

**DEDUCTEAM**

**IN COLLABORATION WITH: Laboratoire specification et vérification (LSV)**

**DOMAIN**

**Algorithmics, Programming, Software and Architecture**

**THEME**

**Proofs and Verification**

# Contents

# Project-Team DEDUCTEAM

*Creation of the Team: 2011 December 01, updated into Project-Team: 2017 January 01*

## Keywords

### Computer sciences and digital sciences

A2.1.4. – Functional programming

A2.1.11. – Proof languages

A2.4.3. – Proofs

A3.1.1. – Modeling, representation

A7. – Theory of computation

A7.2. – Logic in Computer Science

### Other research topics and application domains

B7. – Transport and logistics

## 1 Team members, visitors, external collaborators

### Research Scientists

- Gilles Dowek [Team leader, Inria, Senior Researcher, HDR]
- Bruno Barras [Inria, Researcher]
- Frédéric Blanqui [Inria, Researcher, HDR]
- Valentin Blot [Inria, Researcher]
- Jean-Pierre Jouannaud [Inria, Emeritus, HDR]
- Renaud Vilmart [Inria, from Nov 2020, Starting Faculty Position]

### Faculty Member

- Pablo Arrighi [Aix-Marseille Université, Associate Professor, Sept 2020 until Aug 2020, Cana/LIS, HDR]

### Post-Doctoral Fellows

- Michael Farber [Inria, until Sep 2020]
- Rehan Malak [CNRS, until Oct 2020]
- Étienne Miquey [CNRS, until Oct 2020]
- Pierre Vial [Inria, from Apr 2020]

### PhD Students

- Louise Dubois De Prisque [Inria, from Nov 2020]
- Mohamed Yacine El Haddad [CNRS]
- Gaspard Ferey [École Nationale Supérieure des Mines de Paris ]
- Guillaume Genestier [École Normale Supérieure de Paris-Saclay]
- Emilie Grienenberger [École Normale Supérieure de Paris-Saclay]
- Gabriel Hondet [Inria]
- Amelie Ledein [Inria, from Oct 2020]

### Technical Staff

- Boris Djalal [Inria, Engineer, from Jun 2020]

### Interns and Apprentices

- Tristan Delort [Ecole nationale supérieure d'informatique pour l'industrie et l'entreprise, from Jun 2020 until Jul 2020]
- Louise Dubois De Prisque [Inria, from May 2020 until Aug 2020]
- Nathan Guermond [Inria, from Jul 2020 until Sep 2020]
- Amelie Ledein [Inria, from Mar 2020 until Sep 2020]

- Francois Lefoulon [Ecole nationale supérieure d'informatique pour l'industrie et l'entreprise, from Jun 2020 until Jul 2020]
- Valentin Maestracci [Inria, from Apr 2020 until Aug 2020]
- Simon Mirwasser [Ecole normale supérieure Paris-Saclay, from Mar 2020 until Jul 2020]

### Administrative Assistants

- Alexandra Merlin [Inria, from Oct 2020]
- Emmanuelle Perrot [Inria, until Aug 2020]

### External Collaborators

- Pablo Arrighi [Univ Paris-Saclay, from Sep 2020, LRI/Vals puis LMF/QuaCS, HDR]
- Guillaume Burel [Ecole nationale supérieure d'informatique pour l'industrie et l'entreprise]
- Catherine Dubois [Ecole nationale supérieure d'informatique pour l'industrie et l'entreprise, HDR]
- Olivier Hermant [École Nationale Supérieure des Mines de Paris ]
- Stéphane Lengrand [SRI International - USA, from Jul 2020]

## 2 Overall objectives

### 2.1 Objectives

The project-team investigates the design of logical frameworks, in order to ensure interoperability between proof systems, and to the development of system-independent proof libraries. To achieve these goals, we develop

- a logical framework DEDUKTI, where several theories can be expressed,
- tools to import proofs developed in external proof systems to DEDUKTI theories,
- tools to translate proofs from one DEDUKTI theory to another,
- tools to export proofs expressed in DEDUKTI theories to an external proof system,
- tools to prove the confluence, the termination, and the consistency of theories expressed in DEDUKTI,
- tools to develop proofs directly in DEDUKTI,
- an encyclopedia LOGIPEDIA of proofs expressed in various DEDUKTI theories.

### 2.2 History

The idea that systems such as Euclidean geometry or set theory should be expressed, not as independent systems, but in a logical framework appeared with the design of the first logical framework: predicate logic, in 1928. Later, several more powerful logical frameworks have been designed: $\lambda$-prolog, Isabelle, the Edinburgh logical framework, Pure type systems, and Deduction modulo theory.

The logical framework that we use is a simple $\lambda$-calculus with dependent types and rewrite rules, called the $\lambda\Pi$-calculus modulo theory, and also the Martin-Löf logical framework, and it generalizes all the mentioned frameworks. It is implemented in the system DEDUKTI.

The first version of DEDUKTI was developed in 2011 by Mathieu Boespflug [30]. From 2012 to 2015, new versions of DEDUKTI were developed and several theories were expressed in DEDUKTI, allowing to import proofs developed in MATITA (with the tool KRAJONO), HOL LIGHT (with the tool HOLIDE), FoCaLiZe (with the tool Focalide), iProver, and Zenon, totalizing several hundred of megabytes of proofs.

From 2015 to 2018, we focused on the translation of proofs from one Dedukti theory to another and to the exporting of proofs to other proof systems. In particular the Matita arithmetic library has been translated to a much weaker theory: constructive simple type theory, allowing to export it to Coq, Lean, PVS, HOL Light, and Isabelle/HOL. This led us to develop, in 2018, an online proof encyclopedia Logipedia, allowing to share and browse this library. We also focused on the development of new theories in Dedukti, and on an interactive theorem prover on top of Dedukti.

## 3 Research program

### 3.1 Logical Frameworks

A thesis, which is at the root of our research effort, is that logical systems should be expressed as theories in a logical framework. As a consequence, proof-checking systems should not be focused on one theory, such as Simple type theory, Martin-Löf's type theory, or the Calculus of constructions, but should be theory independent. On the more theoretical side, the proof search algorithms, or the algorithmic interpretation of proofs should not depend on the theory in which proofs are expressed, but this theory should just be a parameter. This is for instance expressed in the title of our invited talk at ICALP 2012: *A theory independent Curry-De Bruijn-Howard correspondence* [31].

Various limits of Predicate logic have led to the development of various families of logical frameworks: $\lambda$-prolog and Isabelle have allowed terms containing free variables, the Edinburgh logical framework has allowed proofs to be expressed as $\lambda$-terms, Pure type systems have allowed propositions to be considered as terms, and Deduction modulo theory has allowed theories to be defined not only with axioms, but also with computation rules.

The $\lambda\Pi$-calculus modulo theory, that is implemented in the system Dedukti and that is a synthesis of the Edinburgh logical framework and of Deduction modulo theory, subsumes them all. Part of our research effort is focused on improving the $\lambda\Pi$-calculus modulo theory, for instance allowing to define congruences with associative and commutative rewriting. Another part of our research effort is focused on the automatic analysis of theories to prove their confluence, termination, and consistency either by pencil and paper proofs or automatically [4].

### 3.2 Interoperability and proof encyclopediae

Using a single prover to check proofs coming from different systems naturally leads to investigate how these proofs can be translated from one theory to another and used in a system different from the system in which they have been developed. This issue is of prime importance because developments in proof systems are getting bigger and, unlike other communities in computer science, the proof checking community has given little effort in the direction of standardization and interoperability.

For each proof, independently of the system in which it has been developed, we should be able to identify the systems in which it can be expressed. For instance, we have shown that many proofs developed in the Matita prover did not use the full strength of the logic of Matita and could be exported, for instance, to the systems of the HOL family, that are based on a weaker logic.

Rather than importing proofs from one system, transforming them, and exporting them to another system, we can use the same tools to develop system-independent proof encyclopedia called Logipedia. In such a library, each proof is labeled with the theories in which it can be expressed and so with the systems in which it can be used.

System independent proofs translated from the libraries of formal proof assistants are stored and classified into Nubo, a repository for interoperable formal proofs. Nubo allows to archive formal developments translated into Dedukti and ensure they are correct, so that translated libraries may be available anytime, and correct.

### 3.3 Interactive theorem proving

If our main goal with DEDUKTI is to import, transform, and export proofs developed in other systems, we also want to investigate how DEDUKTI can be used as the basis of an interactive theorem prover. This leads to two new scientific questions: first, how much can a tactic system be theory independent, and then how does rewriting extends the possibility to write tactics.

This has led to the development of a new version of DEDUKTI, which supports metavariables. Several tactics have been developed for this system, which are intended to help a human user to write proofs in our system instead of writing proof terms by hand. This work is a continuation of the previous work the team did on DEMON, which was an extension of DEDUKTI, whereas the support for interactive theorem proving is now native in DEDUKTI.

### 3.4 Proof automation

Interoperability between interactive and automatic theorem provers can be fruitful to both systems: results coming from automatic solvers can be checked by a third-party software with an identified kernel, and interactive provers can benefit from more automation. We are pushing towards this last application by extending the SMTCoq plugin for the Coq proof assistant with new logical transformations that encode Coq goals into first-order logic, which is the input logic of the class of automatic provers called SMT solvers.

## 4 Application domains

### 4.1 Interoperability

Our main impact applications, for instance to proofs of programs, or to air traffic control, are through our cooperation with other teams.

As a matter of fact, we view our work on interoperability and on the design of a formal proof encyclopedia as a service to the formal proof community.

## 5 New software and platforms

### 5.1 New software

#### 5.1.1 Dedukti

**Keyword:** Logical Framework

**Functional Description:** Dedukti is a proof-checker for the LambdaPi-calculus modulo. As it can be parametrized by an arbitrary set of rewrite rules, defining an equivalence relation, this calculus can express many different theories. Dedukti has been created for this purpose: to allow the interoperability of different theories.

Dedukti's core is based on the standard algorithm for type-checking semi-full pure type systems and implements a state-of-the-art reduction machine inspired from Matita's and modified to deal with rewrite rules.

Dedukti's input language features term declarations and definitions (opaque or not) and rewrite rule definitions. A basic module system allows the user to organize his project in different files and compile them separately.

Dedukti features matching modulo beta for a large class of patterns called Miller's patterns, allowing for more rewriting rules to be implemented in Dedukti.

**URL:** `https://deducteam.github.io/`

**Publications:** hal-01086609, hal-01176715, hal-01441751

**Contact:** François Thiré

**Participants:** François Thiré, Gaspard Ferey, Guillaume Genestier, Rodolphe Lepigre

### 5.1.2 Logipedia

**Name:** Logipedia

**Keywords:** Formal methods, Web Services, Logical Framework

**Functional Description:** Logipedia is composed of two distinct parts: 1) A back-end that translates proofs expressed in a theory encoded in Dedukti to other systems such as Coq, Lean or HOL 2) A front-end that prints these proofs in a "nice way" via a website. Using the website, the user can search for a definition or a theorem then, download the whole proof into the wanted system.

Currently, the available systems are: Coq, Matita, Lean, PVS and OpenTheory. The proofs comes from a logic called STTForall.

In the long run, more systems and more logic should be added.

**Release Contributions:** This is the beta version of Logipedia. It implements the functionalities mentioned above.

**URL:** `http://www.logipedia.science`

**Contact:** François Thiré

### 5.1.3 nubo

**Name:** Nubo

**Keywords:** Interoperability, Proof

**Functional Description:** Nubo is a repository of formal proofs for computer scientists and mathematicians. Nubo aims to leverage the interoperability issues raised by the substantial quantity of proof systems. To do so, it relies on a formalism in which many proofs of other systems can be stated. This formalism allows to translate formal developements to and fro foreign systems. Nubo stores, classifies and serves those formal developments expressed in this general formalism. As such, developers may exchange their proofs, whatever their favourite system is.

**URL:** `https://github.com/Deducteam/nubo`

**Contact:** Gabriel Hondet

### 5.1.4 Kontroli

**Keywords:** Rewriting systems, Higher-order logic

**Functional Description:** Kontroli is an alternative implementation of the logical framework Dedukti, exploring efficient parallel verification of proofs.

**Contact:** Michael Farber

### 5.1.5 Agda2Dedukti

**Keywords:** Compilation, Proof assistant, Higher-order logic, Rewriting systems

**Functional Description:** Translation of Agda proofs to the Logical Framework Dedukti.

**URL:** `https://github.com/Deducteam/Agda2Dedukti`

**Contact:** Guillaume Genestier

**Partner:** Chalmers University

### 5.1.6 Coqine

**Name:** Coq In dEdukti

**Keywords:** Higher-order logic, Formal methods, Proof

**Functional Description:** CoqInE is a plugin for the Coq software translating Coq proofs into Dedukti terms. It provides a Dedukti signature file faithfully encoding the underlying theory of Coq (or a sufficiently large subset of it). Current development is mostly focused on implementing support for Coq universe polymorphism. The generated ouput is meant to be type-checkable using the latest version of Dedukti.

**URL:** `http://www.ensiie.fr/~guillaume.burel/blackandwhite_coqInE.html.en`

**Contact:** Guillaume Burel

### 5.1.7 HOT

**Name:** Higher-Order Termination

**Functional Description:** HOT is an automated termination prover for higher-order rewriting, based on the notion of computability closure.

**URL:** `http://rewriting.gforge.inria.fr/hot.html`

**Contact:** Frédéric Blanqui

### 5.1.8 SizeChangeTool

**Keywords:** Rewriting systems, Proof assistant, Termination

**Functional Description:** A termination-checker for higher-order rewriting with dependent types.

Took part in the Termination Competition 2018 ( `http://termination-portal.org/wiki/Termination_Competition_2018` ) in the "Higher-Order Rewriting (union Beta)" category.

**URL:** `https://github.com/Deducteam/SizeChangeTool`

**Contact:** Guillaume Genestier

**Partner:** Mines ParisTech

### 5.1.9 ekstrakto

**Keywords:** TPTP, TSTP, Proof assistant, Dedukti

**Functional Description:** Extracting TPTP problems from a TSTP trace. Proof reconstruction in Dedukti from TSTP trace.

**URL:** `https://github.com/elhaddadyacine/ekstrakto`

**Contact:** Mohamed Yacine El Haddad

### 5.1.10 lrat2dk

**Keywords:** Automated theorem proving, Proof

**Functional Description:** Take as input a SAT proof trace in LRAT format, which can be obtained from the de facto standard format DRAT using drat-trim. Output a proof checkable by Dedukti, in a shallow encoding of propositional logic.

**URL:** `https://github.com/gburel/lrat2dk`

**Contact:** Guillaume Burel

**Participant:** Guillaume Burel

**Partner:** ENSIIE

#### 5.1.11 iProver Modulo

**Keywords:** Automated deduction, Automated theorem proving

**Scientific Description:** Integration of ordered polarized resolution modulo theory into the prover iProver.

**Functional Description:** iProver Modulo is an extension of the automated theorem prover iProver originally developed by Konstantin Korovin at the University of Manchester. It implements ordered polarized resolution modulo theory, a refinement of the resolution method based on deduction modulo theory. It takes as input a proposition in predicate logic and a clausal rewriting system defining the theory in which the formula has to be proved. Normalization with respect to the term rewriting rules is performed very efficiently through translation into OCaml code, compilation and dynamic linking. Experiments have shown that ordered polarized resolution modulo dramatically improves proof search compared to using raw axioms.

**News of the Year:** Maintenance of Dedukti output

**URL:** `https://github.com/gburel/iProverModulo`

**Publications:** hal-01126321, hal-01125858

**Contact:** Guillaume Burel

**Participant:** Guillaume Burel

**Partner:** ENSIIE

#### 5.1.12 Autotheo

**Keyword:** Automated deduction

**Scientific Description:** Transformation of axiomatic theories into rewriting systems that can be used by iProverModulo.

**Functional Description:** Autotheo is a tool that transforms axiomatic theories into polarized rewriting systems, thus making them usable in iProverModulo. It supports several strategies to orient the axioms, some of them being proved to be complete, in the sense that ordered polarized resolution modulo the resulting systems is refutationally complete, some others being merely heuristics. In practice, Autotheo takes a TPTP input file and produces an input file for iProverModulo.

**News of the Year:** Maintenance.

**URL:** `http://www.ensiie.fr/~guillaume.burel/blackandwhite_autotheo.html.en`

**Publication:** inria-00614040

**Contact:** Guillaume Burel

**Participant:** Guillaume Burel

**Partner:** ENSIIE

## 6 New results

### 6.1 Development of Dedukti

Diego Diverio (engineer SED), François Lefoulon (intern) and Ashish Kumar Barnawal (intern) developed the Emacs and VSCode interfaces of Dedukti v3 aka Lambdapi.

In her internship with Frédéric Blanqui and Catherine Dubois, Amélie Ledein added in Lambdapi, a tool for generating induction principles for first-order mutual data types.

Gabriel Hondet and Frédéric Blanqui published in [17] a description of the new rewriting engine of Lambdapi. Dedukti is a type-checker for the $\lambda\Pi$-calculus modulo rewriting, an extension of Edinburgh's logical framework LF where functions and type symbols can be defined by rewrite rules. It therefore contains an engine for rewriting LF terms and types according to the rewrite rules given by the user. A key component of this engine is the matching algorithm to find which rules can be fired. In this paper, we describe the class of rewrite rules supported by Dedukti and the new implementation of the matching algorithm. Dedukti supports non-linear rewrite rules on terms with binders using higher-order pattern-matching as in Combinatory Reduction Systems (CRS). The new matching algorithm extends the technique of decision trees introduced by Luc Maranget in the OCaml compiler to this more general context.

During his postdoc, Rehan Malak extended Lambdapi so that the interactive proof mode can also be used for defining types and definitions.

During his postdoc, Michael Färber developed a small multi-threaded type-checker for Dedukti files [24]. The lambda-Pi calculus modulo rewriting is a framework to uniformly express a multitude of logical systems. The reference proof checker for this calculus, Dedukti, has a relatively large kernel, making its correctness difficult to verify. This work deals with the question how small one can make a kernel that is sufficiently powerful to verify most Dedukti theories, such as those generated from proof assistants such as Isabelle or automated theorem provers such as iProver Modulo. The result of this work is a new proof checker called Kontroli, implementing a kernel that is more than five times smaller than Dedukti's. Furthermore, unlike Dedukti, Kontroli allows for concurrent checking of theorems independently of the theory structure. Despite its small size, Kontroli is faster than Dedukti on all of five evaluated datasets obtained from automated and interactive theorem provers.

### 6.2 Theory of $\lambda\Pi$-calculus modulo rewriting and other logical formalisms

Frédéric Blanqui published in [15] a new criterion for checking the type safety of rewriting rules in the $\lambda\Pi$-calculus modulo rewriting. The expressiveness of dependent type theory can be extended by identifying types modulo some additional computation rules. But, for preserving the decidability of type-checking or the logical consistency of the system, one must make sure that those user-defined rewriting rules preserve typing. In this paper, he gives a new method to check that property using Knuth-Bendix completion.

Gaspard Férey has proposed, in his doctoral thesis prepared under the supervision of Gilles Dowek and Jean-Pierre Jouannaud, new confluence results for untyped higher-order rewrite systems including functional reductions, both left-linear ones and non-left-linear ones. These results have been presented at various specialized workshops and are now under evaluation by journals [26] [25] [23].

Guillaume Genestier defended his PhD thesis on dependently-typed termination and embedding of extensional universe-polymorphic type theory using rewriting. Dedukti is a logical framework in which the user encodes the theory she wants to use via rewriting rules. To ensure the decidability of typing, the rewriting system must be terminating. After recalling some properties of pure type systems and their extension with rewriting, a termination criterion for higher-order rewriting with dependent types is presented. It is an extension of the dependency pairs to the $\lambda\Pi$-calculus modulo rewriting. This result features two main theorems. The first one states that the well-foundedness of the call relation defined from dependency pairs implies the strong normalization of the rewriting system. The second result of this part describes decidable sufficient conditions to use the first one. This decidable version of the termination criterion is implemented in "SizeChange Tool". The second part of this thesis is dedicated to the use of the logical framework Dedukti to encode a rich type theory. We are interested in a fragment of the logic beyond Agda which includes two widely used features: extension of conversion with the eta rule and universe polymorphism. Once again, this work includes a theoretical part, with correct encodings of both features in the lambda-pi-calculus modulo rewriting, and a prototypical translator from Agda to Dedukti.

During his internship supervised by Bruno Barras and Valentin Blot, Valentin Maestracci has developed a Dedukti theory to encode Two-Layer Type Theories. He also extended this theory with a significant subset of the primitives of Cubical Type Theories [14]

Bruno Barras and Rehan Malak have developed a library of semi-simplicial sets in Dedukti. As an application, they built a formal model of Girard's system F in semi-simplicial sets. This works has been accepted for a presentation at the conference TYPES 2020.

During his internship supervised by Bruno Barras, Nathan Guermond has studied encodings of set theories inside type theory, following the work of Aczel. He has shown that a weaker form of replacement (called *functional replacement*) can be derived in Zermelo set theory. This is an important step towards the formulation of a constructive set theory, encoded in type theory, and in which type theory can be encoded.

During his post-doc, Étienne Miquey worked with Valentin Blot on a new computational interpretation of the axiom of countable choice in a classical setting. This interpretation uses memoization, a technique developed by Étienne in his previous works, as well as bar induction principles [16].

Étienne Miquey, Valentin Blot and Alexandre Miquel supervised the internship of Simon Mirwasser. During this internship, Simon studied several variants of morphisms in the context of implicative algebras, a categorical model of Krivine's classical realizability developed by Alexandre Miquel.

## 6.3 New theories in the $\lambda\Pi$-calculus modulo rewriting

Frédéric Blanqui and Gabriel Hondet submitted a paper on the encoding in the $\lambda\Pi$-calculus modulo rewriting of predicate subtyping and proof irrelevance as used in proof assistants like PVS. The $\lambda\Pi$-calculus modulo theory is a logical framework in which various logics and type systems can be encoded, thus helping the cross-verification and interoperability of proof systems based on those logics and type systems. In this paper, they show how to encode predicate subtyping and proof irrelevance, two important features of the PVS proof assistant. They prove that this encoding is correct and that encoded proofs can be mechanically checked by Dedukti. The paper is available on `https://blanqui.gitlabpages.inria.fr/papers/types20.pdf`.

Frédéric Blanqui, Gilles Dowek, Emilie Grienenberger, Gabriel Hondet and François Thiré worked on a new set of axioms for mathematics. They developed a theory in the $\lambda\Pi$-calculus modulo theory, the theory U, where all the proofs of Minimal predicate logic, Constructive predicate logic, Ecumenical predicate logic, Minimal simple type theory, Constructive simple type theory, Ecumenical simple type theory, Simple type theory with predicate subtyping, the Calculus of constructions, Simple type theory with prenex predicative polymorphism, and the Calculus of constructions with prenex predicative polymorphism can be expressed. The proofs in the theory U, can be classified into proofs in Minimal predicate logic, Constructive predicate logic, etc. just by identifying the axioms they use. We identify sub-theories of U that correspond to each of these theories, and we prove that when a proof in U uses only symbols of a sub-theory, then it is a proof in that sub-theory.

Gaspard Férey has proposed, in his doctoral thesis prepared under the supervision of Gilles Dowek and Jean-Pierre Jouannaud, a new formalization, in the $\lambda\Pi$-calculus modulo theory, of the Calculus of constructions with universe polymorphism.

## 6.4 Translation from one theory to another

François Thiré has defended his doctoral thesis prepared under the supervision of Gilles Dowek and Stéphane Graham-Lengrand, presenting a methodology to translate proofs from one theory to another within Dedukti. He has applied this methodology to the translation of a library of arithmetic results originally developed in the Calculus of constructions with inductive types and universes to a much weaker theory : Simple type theory with prenex object level polymorphism. This has permitted to export this proof to several systems including HOL Light, Isabelle / HOL, HOL 4, Coq, Lean, PVS and, of course, Matita.

Guillaume Genestier has published in `http://doi.org/10.4230/LIPIcs.FSCD.2020.31` his work on the encoding of two common features: universe polymorphism and eta-convertibility, and its application to the translation of Agda programs into Dedukti.

During his internship supervised by Frédéric Blanqui and Guillaume Genestier, Tristan Delort studied how to translate proofs in the STTfa logic to Agda [29].

### 6.5 Alignment of logical connectives

Émilie Grienenberger has developed a new presentation of Ecumenical logic, that has permitted to develop a first version of Ecumenical simple type theory and to express it is Dedukti.

### 6.6 Automation for the Coq proof assistant

In a new contract with the society Nomadic Labs, Valentin Blot, Louise Dubois de Prisque and Pierre Vial started to design and implement automatic tactics for the Coq proof assistant based on external automatic solvers. The idea is to provide independent logical transformations that encode various aspects of Coq logic into first-order logic. Together, they will allow users to automatically encode some Coq goals into the input logic of SMT solvers, then rely on the SMTCoq project to discharge them.

Valentin Blot and Boris Djalal are working on the engineering part of this project, and started to design continuous integration for it and for the SMTCoq plugin.

This work is done in collaboration with Chantal Keller (Université Paris-Saclay).

### 6.7 Quantum Computing

Renaud Vilmart arrived late in 2020 (in November). He worked with former LRI collaborators on a kind of geometry of interaction for a graphical language for quantum computing called ZX-Calculus. He also works more generally on the development of this graphical language.

## 7 Bilateral contracts and grants with industry

### 7.1 Bilateral contracts with industry

Valentin Blot obtained with Chantal Keller funding for a 4-year project involving a PhD student, a research engineer (2 years) and a post-doctoral researcher (2 years). This funding is part of the Inria - Nomadic labs partnership for Tezos blockchain.

## 8 Partnerships and cooperations

### 8.1 National initiatives

The ANR PROGRAMme is an ANR for junior researcher Liesbeth Demol (CNRS, UMR 8163 STL, University Lille 3) to which G. Dowek participates. The subject is: "What is a program? Historical and Philosophical perspectives". This project aims at developing the first coherent analysis and pluralistic understanding of "program" and its implications to theory and practice.

## 9 Dissemination

### 9.1 Promoting scientific activities

#### 9.1.1 Scientific events: organisation

Frédéric Blanqui is Workshop Chair of the ACM/IEEE Symposium on Logic in Computer Science (LICS). He also has been member of the panel of the 9th Confluence Competition (CoCo'20).

Pablo Arrighi organized the conferences QPL 2020 and MFPS 2020.

### 9.1.2 Scientific events: selection

**Member of the conference program committees** Gilles Dowek has been member of the program committees of AUTOMATA, TYPES and IJCAR conferences.

Frédéric Blanqui has been member of the program committees of the 17th International Colloquium on Theoretical Aspects of Computing (ICTAC'20), the 13th Conference on Intelligent Computer Mathematics (CICM'20), the 15th International Workshop on Logical Frameworks and Meta Languages: Theory and Practice (LFMTP'20), the 23rd International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'20).

**Reviewer** Frédéric Blanqui has been reviewer for the TYPES 2019 conference (abstract and post-proceedings).

Bruno Barras has been a reviewer for the POPL 2021 conference.

### 9.1.3 Invited talks

Frédéric Blanqui has been invited to give a talk at the 9th International Workshop on Confluence (IWC'20).

Pablo Arrighi has given an invited talk at the 9th international conference on quantum walks and quantum simulation (QWQS'20).

### 9.1.4 Leadership within the scientific community

Frédéric Blanqui is member of the steering committees of ACM/IEEE Symposium on Logic in Computer Science (LICS), and of the TYPES conference.

### 9.1.5 Scientific expertise

Gilles Dowek is a member of the scientific committee of "Société informatique de France", "Main à la pâte","Comité national pilote d'éthique du numérique", IRT SystemX, IHEST and member of "Comité national français d'histoire et de philosophie des sciences et des techniques".

## 9.2 Teaching - Supervision - Juries

### 9.2.1 Teaching

- Master: Frédéric Blanqui, formal languages, 21h, M1, ENSIIE, France
- Master: Frédéric Blanqui, rewriting theory, 14h, M1, ENS Paris-Saclay, France
- Master: Frédéric Blanqui, $\lambda$-calculus and theories in first-order logic, 18h, M1/M2, ENS Paris-Saclay, France
- Master: Bruno Barras, Proof Assistants, 12h, M2, MPRI, France
- Master: Gabriel Hondet, rewriting theory TD, 14h, M1, ENS Paris-Saclay, France
- Master: Amélie Ledein, programming - TD/project, 30h, M1, ENS Paris-Saclay, France
- Licence: Amélie Ledein, software engineering - project, 30h, L3, ENS Paris-Saclay, France
- Licence: Gabriel Hondet, operating systems and architecture TD, 22.5h, L3, ENS Paris-Saclay, France
- Licence: Gabriel Hondet, programming TD, 12h, L3, ENS Paris-Saclay, France
- Licence: Gabriel Hondet, projet bases de données, 22.5h, L3, ENS Paris-Saclay, France
- License: Gaspard Férey, Théorie des Langages, 44h, L3, EISTI
- IUT: Yacine El Haddad, Programmation Web Côté Serveur, 62h, IUT of Orsay

#### 9.2.2 Supervision

- PhD defended on 03/12/20: François Thiré,Interoperability between proof systems using the Dedukti logical framework, started on 01/10/17 under the supervision of Gilles Dowek and Stéphane Graham-Lengrand.
- PhD defended on 10/12/20: Guillaume Genestier, Dependently-Typed Termination and Embedding of Extensional Universe-Polymorphic Type Theory using Rewriting, started 01/10/17 under the supervision of Frédéric Blanqui and Olivier Hermant.
- PhD in progress: Gaspard Férey, started on 01/10/17 under the supervision of Gilles Dowek and Jean-Pierre Jouannaud.
- PhD in progress: Mohamed Yacine El Haddad, using automated provers in proof assistants, 05/01/18, Frédéric Blanqui and Guillaume Burel.
- PhD in progress: Gabriel Hondet, translating PVS proofs to Dedukti, 01/10/19, Frédéric Blanqui and Gilles Dowek.
- PhD in progress: Émilie Grienenberger, started on 01/10/19, under the supervision of Gilles Dowek and Dale Miller.
- PhD in progress: Amélie Ledein, Definitions and proofs by induction in dependent type theory, started on 01/10/20 under the supervision of Catherine Dubois and Frédéric Blanqui.

## 10 Scientific production

### 10.1 Major publications

[1] A. Assaf, G. Burel, R. Cauderlier, D. Delahaye, G. Dowek, C. Dubois, F. Gilbert, P. Halmagrand, O. Hermant and R. Saillard. 'Expressing theories in the $\lambda\Pi$-calculus modulo theory and in the Dedukti system'. In: *22nd International Conference on Types for Proofs and Programs, TYPES 2016*. Novi SAd, Serbia, May 2016. URL: https://hal-mines-paristech.archives-ouvertes.fr/hal-01441751.

[2] B. Barras, T. Coquand and S. Huber. 'A generalization of the Takeuti-Gandy interpretation'. In: *Mathematical Structures in Computer Science* 25.5 (2015), pp. 1071–1099. DOI: 10.1017/S0960129514000504. URL: https://doi.org/10.1017/S0960129514000504.

[3] F. Blanqui. 'Definitions by rewriting in the Calculus of Constructions'. Anglais. In: *Mathematical Structures in Computer Science* 15.1 (2005), pp. 37–92. DOI: 10.1017/S0960129504004426. URL: http://hal.inria.fr/inria-00105648/en/.

[4] F. Blanqui, J.-P. Jouannaud and A. Rubio. 'The Computability Path Ordering'. In: *Logical Methods in Computer Science* (Oct. 2015). DOI: 10.2168/LMCS-11(4:3)2015. URL: https://hal.inria.fr/hal-01163091.

[5] V. Blot. 'An interpretation of system F through bar recursion'. In: *32nd ACM/IEEE Symposium on Logic in Computer Science*. IEEE, 2017.

[6] G. Burel, G. Bury, R. Cauderlier, D. Delahaye, P. Halmagrand and O. Hermant. 'First-Order Automated Reasoning with Theories: When Deduction Modulo Theory Meets Practice'. In: *Journal of Automated Reasoning* (2019). DOI: 10.1007/s10817-019-09533-z. URL: https://hal.archives-ouvertes.fr/hal-02305831.

[7] D. Cousineau and G. Dowek. 'Embedding Pure Type Systems in the $\lambda\Pi$-calculus modulo'. In: *Typed lambda calculi and applications*. Ed. by S. R. della Rocca. Vol. 4583. Lecture Notes in Computer Science. Springer-Verlag, 2007, pp. 102–117.

[8] G. Dowek, T. Hardin and C. Kirchner. 'Theorem proving modulo'. In: *Journal of Automated Reasoning* 31 (2003), pp. 33–73.

[9] O. Hermant. 'Resolution is Cut-Free'. In: *Journal of Automated Reasoning* 44.3 (Mar. 2010), pp. 245–276.

[10] M. Jacquel, K. Berkani, D. Delahaye and C. Dubois. 'Tableaux Modulo Theories Using Superdeduction'. In: *Global Journal of Advanced Software Engineering (GJASE)* 1 (Dec. 2014), pp. 1–13. DOI: 10.1007/978-3-642-31365-3_26. URL: https://hal.archives-ouvertes.fr/hal-01099338.

[11] M. Jacquel, K. Berkani, D. Delahaye and C. Dubois. 'Verifying B Proof Rules using Deep Embedding and Automated Theorem Proving'. In: *Software and Systems Modeling (SoSyM)* (June 2013).

## 10.2 Publications of the year

### International journals

[12] G. Burel. 'Linking Focusing and Resolution with Selection'. In: *ACM Transactions on Computational Logic* 21.3 (22nd May 2020), pp. 1–30. DOI: 10.1145/3373276. URL: https://hal.archives-ouvertes.fr/hal-02908808.

[13] G. Burel, G. Bury, R. Cauderlier, D. Delahaye, P. Halmagrand and O. Hermant. 'First-Order Automated Reasoning with Theories: When Deduction Modulo Theory Meets Practice'. In: *Journal of Automated Reasoning* 64.6 (2020), pp. 1001–1050. DOI: 10.1007/s10817-019-09533-z. URL: https://hal.archives-ouvertes.fr/hal-02305831.

### International peer-reviewed conferences

[14] B. Barras and V. Maestracci. 'Implementation of Two Layers Type Theory in Dedukti and Application to Cubical Type Theory'. In: Logical Frameworks and Meta-Languages: Theory and Practice 2020. Paris, France, 29th June 2020. URL: https://hal.inria.fr/hal-03138145.

[15] F. Blanqui. 'Type safety of rewrite rules in dependent types'. In: FSCD 2020 - 5th International Conference on Formal Structures for Computation and Deduction. Vol. 167. Paris, France, 28th June 2020, p. 14. DOI: 10.4230/LIPIcs.FSCD.2020.13. URL: https://hal.inria.fr/hal-02981528.

[16] H. Herbelin and É. Miquey. 'A calculus of expandable stores: Continuation-and-environment-passing style translations'. In: *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS '20), July 8–11, 2020, Saarbrücken, Germany*. LICS 2020 - 35th ACM/IEEE Symposium on Logic in Computer Science. Saarbrücken / Virtual, Germany, 8th July 2020, pp. 564–577. DOI: 10.1145/3373718.3394792. URL: https://hal.archives-ouvertes.fr/hal-02557823.

[17] G. Hondet and F. Blanqui. 'The New Rewriting Engine of Dedukti'. In: FSCD 2020 - 5th International Conference on Formal Structures for Computation and Deduction. 167. Paris, France, 28th June 2020, p. 16. DOI: 10.4230/LIPIcs.FSCD.2020.35. URL: https://hal.inria.fr/hal-02981561.

### Scientific book chapters

[18] O. Bournez, G. Dowek, R. Gilleron, S. Grigorieff, J.-Y. Marion, S. Perdrix and S. Tison. 'Theoretical Computer Science: Computability, Decidability and Logic'. In: *A Guided Tour of Artificial Intelligence Research - Volume III: Interfaces and Applications of Artificial Intelligence (10.1007/978-3-030-06170-8)*. 2020. URL: https://hal.archives-ouvertes.fr/hal-03173193.

[19] O. Bournez, G. Dowek, R. Gilleron, S. Grigorieff, J.-Y. Marion, S. Perdrix and S. Tison. 'Theoretical Computer Science: Computational Complexity'. In: *A Guided Tour of Artificial Intelligence Research - Volume III: Interfaces and Applications of Artificial Intelligence (10.1007/978-3-030-06170-8)*. 2020. URL: https://hal.archives-ouvertes.fr/hal-02995771.

### Doctoral dissertations and habilitation theses

[20] G. Genestier. ‘Dependently-Typed Termination and Embedding of Extensional \linebreak Universe-Polymorphic Type Theory using Rewriting’. Université Paris-Saclay, 10th Dec. 2020. URL: https://tel.archives-ouvertes.fr/tel-03167579.

### Reports & preprints

[21] K. Chardonnet, B. Valiron and R. Vilmart. *Geometry of Interaction for ZX-Diagrams*. 1st Mar. 2021. DOI: 10.4230/LIPIcs.CVIT.2016.23. URL: https://hal.archives-ouvertes.fr/hal-03154573.

[22] N. Dershowitz, J.-P. Jouannaud and Q. Wang. *The Algebra of Infinite Sequences: Notations and Formalization*. 11th May 2020. URL: https://hal.inria.fr/hal-02569232.

[23] G. Dowek, G. Férey, J.-P. Jouannaud and J. Liu. *Confluence of Non-Terminating Left-Linear Higher-Order Rewrite Theories*. 2nd Feb. 2021. DOI: 10.1145/nnnnnnn.nnnnnnn. URL: https://hal.inria.fr/hal-03126111.

[24] M. Färber. *Small, Fast, Concurrent Proof Checking for the lambda-Pi Calculus Modulo Rewriting*. 17th Feb. 2021. URL: https://hal.inria.fr/hal-03143359.

[25] G. Férey and J.-P. Jouannaud. *Confluence in non-left-linear higher-order theories*. 3rd Mar. 2021. URL: https://hal.inria.fr/hal-03126115.

[26] G. Férey and J.-P. Jouannaud. *Confluence in UnTyped Higher-Order Theories by means of Critical Pairs*. 1st Feb. 2021. DOI: 10.1145/nnnnnnn.nnnnnnn. URL: https://hal.inria.fr/hal-03126102.

[27] J.-P. Jouannaud and F. Orejas. *Unification of drags*. 4th May 2020. URL: https://hal.inria.fr/hal-02562152.

[28] J.-P. Jouannaud and F. Orejas. *Unificatiuon of Drags and Confluence of Drag Rewriting*. 1st Feb. 2021. URL: https://hal.inria.fr/hal-02562463.

### Other scientific publications

[29] T. Delort. ‘Importing Logipedia proofs in Agda’. Inria Saclay Ile de France, 2nd Nov. 2020. URL: https://hal.inria.fr/hal-02985530.

## 10.3 Cited publications

[30] M. Boespflug. ‘Conception d’un noyau de vérification de preuves pour le $\lambda\Pi$-calcul modulo’. PhD thesis. École Polytechnique, 2011.

[31] G. Dowek. ‘A Theory Independent Curry-de Bruijn-howard Correspondence’. In: *Proceedings of the 39th International Colloquium Conference on Automata, Languages, and Programming - Volume Part II*. ICALP’12. Warwick, UK: Springer-Verlag, 2012, pp. 13–15. DOI: 10.1007/978-3-642-31585-5_2. URL: http://dx.doi.org/10.1007/978-3-642-31585-5%5C_2.