RESEARCH CENTRE

Inria Saclay Center at Université Paris-Saclay

IN PARTNERSHIP WITH:

Université Paris-Saclay

2022

ACTIVITY REPORT

Project-Team

DEDUCTEAM

DEDUCTEAM

IN COLLABORATION WITH: Laboratoire de Méthodes Formelles

DOMAIN

Algorithmics, Programming, Software and Architecture

THEME

Proofs and Verification

Contents

Project-Team DEDUCTEAM

Creation of the 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]
- Frederic Blanqui [INRIA, Senior Researcher, HDR]
- Valentin Blot [INRIA, Researcher]
- Théo Winterhalter [INRIA, Researcher, from Oct 2022]

Post-Doctoral Fellows

- Claude Stolze [INRIA, from Oct 2022]
- Pierre Vial [INRIA, until Sep 2022]

PhD Students

- Luc Chabassier [ENS PARIS]
- Louise Dubois De Prisque [INRIA]
- Thiago Felicissimo Cesar [UNIV PARIS SACLAY]
- Émilie Grienenberger [ENS PARIS-SACLAY]
- Gabriel Hondet [INRIA, until Sep 2022]
- Amélie Ledein [INRIA]

Technical Staff

- Boris Djalal [INRIA, Engineer, until May 2022]

Interns and Apprentices

- Elliot Butte [INRIA, Intern, from Jun 2022 until Aug 2022]
- Quentin Buzet [Télécom Paris, Intern, from Jul 2022 until Aug 2022]
- Corentin Chabanol [CENTRALESUPELEC, Intern]
- Loris Cros [CENTRALESUPELEC, Intern, until Jul 2022]
- Yann Leray [ENS PARIS, Intern, from Nov 2022]
- Taïssir Marce [University Paris Saclay, Intern, from May 2022 until Aug 2022]
- Émile Oléon [ENS Paris-Saclay, Intern, from Mar 2022 until Jul 2022]
- Thomas Traversié [CENTRALESUPELEC, Intern]

Administrative Assistant

- Aissatou-Sadio Diallo [INRIA, from May 2022]

Visiting Scientist

- Cristian Sottile [UBA, from Oct 2022 until Oct 2022]

External Collaborators

- Guillaume Burel [ENSIIE]
- Catherine Dubois [ENSIIE, HDR]
- Yoan Géran [ENSMP]
- Olivier Hermant [ENSMP, HDR]
- Jean-Pierre Jouannaud [INRIA, HDR]

2 Overall objectives

2.1 Objectives

Deducteam investigates the design of logical frameworks, that is frameworks where various theories can be defined, and the use of such frameworks for interoperability between proof systems, cross verification of proofs, and the sustainability of proof libraries.

To achieve these goals, we develop

- a logical framework DEDUKTI, where various theories can be expressed,
- several implementations of this framework: DKCHECK, (formerly also called DEDUKTI), that is a small trust base, theory independent, proof-checker, LAMBDAPI, that is a system to develop DEDUKTI proofs interactively, and KONTROLI that is a fast parallel proof-checker for DEDUKTI,
- 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,
- libraries NUBO and LOGIPEDIA of proofs expressed in various DEDUKTI theories.

2.2 History

The development of computerized proof systems such as COQ, HOL LIGHT, or PVS is a major step forward in the quest of mathematical rigor. But it jeopardizes, once again, the universality of mathematical truth: we used to have proofs of Fermat's little theorem, we now have COQ proofs of Fermat's little theorem, HOL LIGHT proofs of Fermat's little theorem, PVS proofs of Fermat's little theorem, etc., as each proof system defines its own language for mathematical statements and its own truth conditions for these statements. See, for instance, our invited talk at IJCAR this year [18]: *From the Universality of Mathematical Truth to the Interoperability of Proof Systems.*

One way to address this issue is to express the theories implemented in these systems in a common logical framework and to determine, for each proof, which axioms it depends on. This way, a proof can be used in any system that supports these axioms, independently of the system it has been developed in.

The idea that systems such as Euclidean geometry, non-Euclidean geometries, set theory, with or without the axiom of choice, etc. should be expressed in the same logical framework appeared, in 1928, with the design of the first logical framework in the history of logic: predicate logic. Later, several more powerful logical frameworks have been designed: λ-Prolog, Isabelle, the Edinburgh logical framework, Pure type systems, Deduction modulo theory, etc.

The logical framework that we use is a simple λ-calculus with dependent types and rewrite rules, called the $\lambda\Pi$-calculus modulo theory, or the Martin-Löf logical framework. It generalizes all the mentioned frameworks. Its concrete syntax is the language DEDUKTI.

The first implementation of DEDUKTI, now called DKCHECK, was developed in 2011 by Mathieu Boespflug [30]. Then, new versions of this implementation 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.

We now focus on the translation of proofs from one DEDUKTI theory to another and on 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. In the same way, the first book of Euclids elements, formalized in COQ, has been translated to predicate logic and exported to several systems, and a proof of Bertrand's theorem, originally developed in MATITA, has been translated to predicative type theory, allowing its export to AGDA.

This led us to develop an on-line proof repository NUBO and an on-line encyclopedia LOGIPEDIA, allowing to share and browse this library.

We also focus on the development of new theories in DEDUKTI, such as Simple type theory with predicate subtyping, implemented in the system PVS, several formulations of homotopy type theory, various formulations of set theory, in particular those used in B and TLA+, matching logic, etc.

Finally, we develop an interactive theorem prover LAMBDAPI for DEDUKTI. This interactive theorem prover is also used as a tool in the process of translating proofs from PVS and from automated theorem provers.

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. In the same way, proof-search algorithms or the algorithmic interpretation of proofs should not depend on a theory, 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: λ-Prolog and Isabelle have allowed terms containing bound variables, the Edinburgh logical framework has allowed proofs to be expressed as λ-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. Our goal is to express as many theories as possible in DEDUKTI, express proofs in these theories and translate proofs from one theory to another.

3.2 Interoperability, cross verfication and sustainability of proof libraries

Using a single prover to check proofs coming from different systems and translating these proofs from one theory to another naturally leads to investigating how these proofs can be used in a system different from the one they have been developed in.

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 put little effort in the direction of standardization and interoperability.

A more recent trend is to use logical frameworks and proof translations for cross-checking. Checking a proof in several systems introduces some redundancy and hence reduces the probability that an incorrect proof is nevertheless successfully verified because of a bug in the proof-checker. This problem can be mitigated by developing proofs in systems that rely on a small and auditable trust base, that ensure a significantly lower probability for such undesirable events. In practice, however, this is not always possible, and our argument gets stronger when the proof has been developed in a theory that does not enjoy a small proof checker, but, instead, a complex, and sometimes heterogeneous, proof-construction system. This is for instance the case of B set theory, the theory on which the B method is based. There are several powerful tools to build proofs in this theory, but no small independent proof checker. Defining such a theory in a logical framework such as DEDUKTI and translating the proofs built by these tools into this theory permits to increase in a substantial way the trust we can have in these proofs.

Finally, on a more long-term perspective, we know that some proof-checking systems are not maintained anymore (this is, for instance the case of Automath and LCF, the two first proof checkers in history). When such a system disappears, its libraries often disappear with it. We can hope that expressing the proofs in a universal format in place of a system-specific one and preserving these proofs into a system-independent on-line repository such as NUBO or LOGIPEDIA will increase the sustainability of these libraries.

3.3 Interactive theorem proving

We also investigate how the $\lambda\Pi$-calculus modulo theory can be used as the basis of an interactive theorem prover. This leads to new scientific questions: first, how much can a tactic system be theory-independent, and then how does rewriting extend the possibility to write tactics.

This has led to the development of LAMBDAPI, which is an interactive theorem prover for the $\lambda\Pi$-calculus modulo theory. 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.

Such an interactive theorem prover happens to be very useful when we translate to DEDUKTI proofs coming from laconic systems that output a proof sketch rather than a full proof. In these cases, one first produces a proof skeleton with many gaps, that are filled, in a second step of the translation, with the help of automatic tactics.

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

Our research project has lead us to focus on applications directed to the proof-checking community itself rather than to users of proof-checking. Indeed, translating proofs from one system to another, or building a system-independent proof library is more a service to the proof-checking community than to the users of formal methods.

This situation is evolving fast, along with the rise of cross-verification.

Providing a complementary small-trust-base proof checker for B leads us to be in closer connection with the community using formal methods in the railways industry and more generally to the modelization of industrial system community.

This is materialized with the ICSPA ANR project. We also have a long-term collaboration with the air traffic control community through the PVS community.

5 Highlights of the year

- The COST action CA20111 EuroProofNet chaired by Frédéric Blanqui started its activities. In 2022, EuroProofNet funded 17 short-term scientific missions and organized 2 schools (Dedukti and VTSA), 2 workshops (PAAR and 1st workshop of proof library management), 1 conference (AITP), a workshop Women in EuroProofNet, and 3 other working group meetings. EuroProofNet now has more than 300 participants from 42 countries.
- The ANR project ICSPA organized its kick off meeting on February 17th.
- Gabriel Hondet defended his PhD on expressing predicate subtyping in computational logical frameworks [26].

6 New software and platforms

6.1 New software

6.1.1 Lambdapi

Keywords: Dependent types, Rewriting, Proof assistant

Functional Description: Lambdapi is an interactive proof development system featuring dependent types like in Martin-Lőf's type theory, but allowing to define objects and types using oriented equations, aka rewriting rules, and reason modulo those equations. This allows to simplify some proofs, and formalize complex mathematical objects that are otherwise impossible or difficult to formalize in more traditional proof systems.

Lambdapi comes with Emacs and VSCode support.

Lambdapi can also read and output Dedukti files, and can thus be used as an higher-level intermediate language for translating proofs from one system to Dedukti.

Lambdapi is a logical framework and does not come with a pre-defined logic. However, it is easy to define a logic by declaring a few symbols and rules. A library of pre-defined logic is also provided.

Here are some of the features of Lambdapi: - Emacs and VSCode plugins (based on LSP) - support for unicode (UTF-8) and user-defined infix operators - symbols can be declared commutative, or associative and commutative - some arguments can be declared as implicit: the system will try to find out their value automatically - symbol and rule declarations are separated so that one can easily define inductive-recursive types or turn a proved equation into a rewriting rule - support for interactive resolution of typing goals, and unification goals as well, using tactics - a rewrite tactic similar to the one of SSReflect in Coq - the possibility of calling external automated provers - a command is provided for automatically generating an induction principle for (mutually defined) strictly-positive inductive types - Lambdapi can call external provers for checking the confluence and termination of user-defined rewriting rules by translating them to the XTC and HRS formats used in the termination and confluence competitions

URL: `https://github.com/Deducteam/lambdapi`

Contact: Frederic Blanqui

6.1.2 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: Francois Thire

Participants: Francois Thire, Gaspard Ferey, Guillaume Genestier, Rodolphe Lepigre

6.1.3 personoj

Keywords: PVS, Automated theorem proving, Dedukti, Machine translation

Functional Description: Personoj comprises a set of PVS patches that may be used to export PVS specifications (propositions and definitions) or to export successive sequents of a proof to lambdapi. Another program is able to process these sequents and call automated theorem provers through Why3 to prove the implications of the successive sequents.

Contact: Gabriel Hondet

6.1.4 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

6.1.5 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

6.1.6 Krajono

Keyword: Proof

Functional Description: Krajono translates Matita proofs into Dedukti[CiC] (encoding of CiC in Dedukti) terms.

Contact: Francois Thire

6.1.7 Holide

Keyword: Proof

Functional Description: Holide translates HOL proofs to Dedukti[OT] proofs, using the OpenTheory standard (common to HOL Light and HOL4). Dedukti[OT] being the encoding of OpenTheory in Dedukti.

URL: https://github.com/Deducteam/Holide

Contact: Guillaume Burel

6.1.8 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: Francois Thire

6.1.9 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

6.1.10 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

6.1.11 Zenon Modulo

Keywords: First-order logic, Automated theorem proving, Deduction Modulo

Functional Description: Zenon Modulo is an extension of the automated theorem prover Zenon. Compared to Super Zenon, it can deal with rewrite rules both over propositions and terms. Like Super Zenon, Zenon Modulo is able to deal with any first-order theory by means of a similar heuristic.

URL: https://github.com/Deducteam/zenon_modulo

Contact: Pierre Halmagrand

6.1.12 CoLoR

Name: Coq Library on Rewriting and termination

Keywords: Coq, Formalisation

Functional Description: CoLoR is a Coq library on rewriting theory and termination. It provides many definitions and theorems on various mathematical structures (quasi-ordered sets, relations, ordered semi-rings, etc.), data structures (lists, vectors, matrices, polynomials, finite graphs), term structures (strings, first-order terms, lambda-terms, etc.), transformation techniques (dependency pairs, semantic labeling, etc.) and (non-)termination criteria (polynomial and matrix interpretations, recursive path ordering, computability closure, etc.).

URL: http://color.inria.fr/

Publications: inria-00543157, inria-00390902, inria-00084835

Authors: Frederic Blanqui, Sebastien Hinderer

Contact: Frederic Blanqui

6.1.13 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

6.1.14 SKonverto

Name: SKonverto

Keywords: Skolemization, First-order logic, Proof assistant

Functional Description: SKonverto is a tool that transforms Lambdapi proofs containing Skolem symbols into proofs without these symbols.

URL: https://github.com/Deducteam/SKonverto

Contact: Mohamed Yacine El Haddad

6.1.15 Predicativize

Name: Predicativize

Keywords: Dedukti, Proof assistant, Interoperability

Functional Description: Predicativize is a tool allowing for the translation of proofs from a core impredicative type theory to a core predicative theory featuring universe polymorphism. It works by calculating constraints between universe levels, which are then solved using universe level unification, generating then a predicative universe polymorphic definition. The theory behind the tool is provided in the paper "Translating proofs from an impredicative type system to a predicative one", by Thiago Felicissimo, Frédéric Blanqui and Ashish Kumar Barnawal. Predicativize was used to translate Matita's arithmetic library to Agda.

URL: https://github.com/Deducteam/predicativize

Contact: Thiago Felicissimo Cesar

6.1.16 KaMeLo

Name: KaMeLo

Keywords: K Framework, Matching Logic, Semantics, Rewriting systems

Functional Description: Translation of the K framework to the Logical Framework Dedukti. The input is written in Matching Logic.

URL: https://gitlab.com/semantiko/kamelo

Contact: Amélie Ledein

6.1.17 MM2DK

Keywords: Metamath, Logical Framework

Functional Description: Translation of the K framework to the Logical Framework Dedukti. The input is written in Matching Logic

URL: https://gitlab.com/semantiko/mm2dk/translator

Contact: Amélie Ledein

Participant: Elliot Butte

7 New results

7.1 Implementations of DEDUKTI

7.1.1 Lambdapi

Participants: Frédéric Blanqui, Bruno Barras, Gabriel Hondet, Quentin Buzet.

Many improvements have been made on Lambdapi in 2022: replacement of the type-checking algorithm by a refinement algorithm, extension of the why3 tactic to handle quantifiers, incremental local confluence checking, postfix operators, addition of a coercion mechanism based on rewrite rules, and export to Coq. Moreover, an open Opam repository for Lambdapi developments has been created.

Lambdapi standard library has been extended. Quentin Buzet (Télécom Paris intern) formalized many functions and properties on natural numbers and polymorphic lists inspired of the SSreflect library in Coq. Quentin Garchery's (EPI Toccata) formalization of binary integers has been integrated by Frédéric Blanqui who simplified the used logic from the calculus of constructions to first-order logic.

Lambdapi has also new users. Stephan Merz (Inria Nancy) formalized the set theory on which the TLA proof system is based, Claude Stolze started to formalize the B set theory, Jean-Paul Bodeveix (Toulouse) used it to formalize proofs coming from Rodin, Artur Kornilowicz (Poland) started to use it to translate the proofs of the Mizar proof checker.

Bruno Barras has improved the efficiency of a call-by-need abstract machine that implements β-reduction and rewriting. Some significative improvements have been carried out by using data-structures requiring less comparisons. However, memory allocation and garbage collection remains a source of inefficiency that shall be addressed in the future. This would allow to integrate this machine to the main development branch.

7.2 Theory of the lambda-Pi-calculus modulo rewriting and other logical formalisms

7.2.1 Sub-theories

Participants: Frédéric Blanqui, Gilles Dowek, Émilie Grienenberger.

The theory $\mathscr{U}$ is a theory that contains several theories as fragments, in particular Predicate logic, Ecumenical predicate logic, Simple type theory, Simple type theory with predicate subtyping, and the Calculus of Constructions. This theory has been introduced in 2021 in an extended abstract. The long version of this paper has been accepted to publication in Logical Methods in Computer Science in 2022.

For this theory, the *fragment theory* shows the modularity of the sub-theories of the theory $\mathscr{U}$, that is that each sub-theory can be used with no reference to the symbols and axioms not in the theory. However the application of this theorem to a specific sub-theory requires a proof that this theory is well-typed. This proof must be provided for each of the sub-theories — for example for each of the 13 fragments of the theory $\mathscr{U}$. A natural follow up of this work is to seach for well-typedness criteria.

Émilie Grienenberger has provided a sufficient condition for a fragment to be well-typed if the theory from which it has been extracted is. This condition is inspired by well-typedness criteria studied by R. Saillard [33]. She has proved that every fragment of a *strongly well-formed* theory is well typed.

She has then proved that if Saillard's algorithm establishing that a theory is *weakly well-formed* succeeds for one theory, then all its fragments are well typed.

This work based on the results of R. Saillard [33], has moreover led to a transformation of some of Saillard's proof that happened not to be fully correct.

7.2.2 Confluence and levels

Participants: Corentin Chabanol, Jean-Pierre Jouannaud, Gilles Dowek.

Confluence and termination of rewrite rules including beta-reduction in the presence of dependent types, depend on each other. In the general case, we must therefore prove the confluence of untyped computations, a problem that we have completely solved in the case of left linear rules [12]. In the left non-linear case, the confluence not being generally satisfied, we proposed a solution which proves the confluence of the only terms t having a level, integer which characterizes a certain level of nesting of the beta-redexes potentially belonging to a left non-linear rule instance applying to t [32]. Corentin Chabanol showed that the calculus of levels is reduced to a system of constraints in algebra (max, plus) on integers, constraints which also intervene in type inference, and are studied in this respect in the team . The calculation of levels is being implemented by Corentin Chabanol.

7.2.3 Completion of rewrite systems

Participants: Loris Cros, Bruno Barras.

In his last year of his programme recherche supervised by Bruno Barras, Loris Cros has finished the implementation of a completion algorithm in lambdapi. The main issue was to adapt an algorithm found in the literature to rewrite rules where the left handside is expressed as a pattern. When the algorithm terminates, it produces a confluent rewrite system which equational theory is the same as the system given as input. It remains to evaluate the cases where this algorithm produces a result. The main motivating example for this work is the de Morgan algebras.

7.3 Expressing theories in DEDUKTI

7.3.1 Predicate subtying and PVS

Participants: Gabriel Hondet, Frédéric Blanqui, Gilles Dowek.

Gabriel Hondet defended his PhD on expressing predicate subtyping in computational logical frameworks [26]. Safe programming as well as most proof systems rely on typing. The more a type system is expressive, the more these types can be used to encode invariants which are therefore verified mechanically through type checking procedures. Predicate subtyping extends simple type theory by allowing terms to be defined by predicates. A predicate subtype $\{x : A|P(x)\}$ is inhabited by terms t of type A for which $P(t)$ holds. This extension provides a rich and intuitive but undecidable type system. This work is dedicated to the encoding of predicate subtyping in DEDUKTI: a logical framework with computation rules. We begin by encoding explicit predicate subtyping for which terms of type A are syntactically different from terms of type $\{x : A|P(x)\}$. Predicate subtyping, is often used implicitly, with no syntactic difference between terms of type A and terms of type $\{x : A|P(x)\}$. We enrich our logical framework with a term refiner which can add these syntactic markers in order to make subtyping explicit in terms. This transformation is used to translate the standard library of PVS, a proof assistant using extensively predicate subtyping, to DEDUKTI.

7.3.2 New expression of Pure Type Systems

Participants: Thiago Felicissimo, Frédéric Blanqui, Gilles Dowek.

Thiago Felicissimo worked on the theory of Dedukti encodings. Conservativity is an important property that ensures that every proof that is checked correct in Dedukti corresponds to a correct proof in the encoded system. But proving the conservativity of encodings is generally hard, and because of this many encodings used in practice lack a conservativity result. In [20], he proposes a different encoding of Pure Type Systems in Dedukti which do not only allow for simple conservativity proofs, but also provides an adequacy theorem — i.e., a one-to-one syntactic correspondence between the (quotiented) framework syntax and the syntax of the encoded system. Unlike most conservativity proofs, his proof does not rely on the normalization of the rewrite rules of the encoding, a property that is known to be hard to show. Given that Pure Type Systems are the basis of many other type systems, his approach should be in general applicable to most encodings.

7.3.3 Universes

Participants: Yoan Géran, Olivier Hermant, Gilles Dowek, Frédéric Blanqui.

Universes are a feature of several type theories, such as those implemented in AGDA and COQ. These two theories have however different universe systems: the one of AGDA is predicative and the one of COQ impredicative. In the past, the predicative case has been studied by Thiago Felicissimo and the impredicative one by Gaspard Férey. In this last case, however the confluence of the theory was left as a conjecture.

Yoan Géran has given a new definition of impredicative universes, simplifying Férey's and proved termination and confluence.

Most definitions of universes in DEDUKTI use a unary successor symbol and a binary max symbol. To decide the word problem in this max-successor algebra, all the proposed definitions use rewriting with matching modulo associativity and commutativity (AC), which is of high complexity and difficult to add in standard algorithms for β-reduction and type-checking. Frédéric Blanqui has shown that it is possible to get rid of matching modulo AC by enforcing terms to be in some special canonical form with respect to AC, and by using rewriting rules taking advantage of this canonical form. This work has been presented at the 7th International Conference on Formal Structures for Computation and Deduction (FSCD) [14].

7.3.4 Set theory

Participants: Thomas Traversié, Valentin Blot, Gilles Dowek, Claude Stolze, Catherine Dubois, Olivier Hermant.

Thomas Traversié, during his internship supervised by Valentin Blot and Gilles Dowek, implemented set theory in Lambdapi, using an encoding of sets with a structure of pointed graphs [25]. This work has been presented in Haïfa, at the workshop LFMTP.

Claude Stolze started a post-doc in October 2022 on expressing in DEDUKTI the B set theory used by ATELIER B. He is also working on a translator from proof obligations generated by ATELIER B into LAMBDAPI. The translator can be found at B-pog-translator repository on github.

7.3.5 Cubical Type Theory

Participants: Émile Oléon, Bruno Barras.

In his internship supervised by Bruno Barras, Émile Oléon has translated by hand a proof expressed in Cubical Agda in the Dedukti encoding of Two Layers Type Theory instantiated with the primitives of Cubical Type Theory. He translated the proof that the loop space of the circle is $\mathbb{Z}$. This paves the way to an automatic translation procedure of Cubical Agda proofs to Dedukti.

7.3.6 Matching Logic

Participants: Amélie Ledein, Valentin Blot, Catherine Dubois.

Amélie Ledein defined a partial shallow embedding of Matching Logic. This embedding is partial because the positive occurrence criteria isn't check by the embedding itself, and it is difficult to find Matching Logic proofs using some rules of the Matching Logic proof system. This initial work began with the study of the formalization of Matching Logic in Metamath, and continues with the objective of better understanding Matching Logic. This is also the objective of other researchers doing similar work in Coq and Lean, with whom Amélie Ledein is collaborating, as well as the K team itself.

7.4 Translations

7.4.1 From Isabelle to DEDUKTI

Participants: Frédéric Blanqui.

In the framework of his PHC Sakura project, Frédéric Blanqui, together with Jérémy Dubut and Akihisa Yamada (AIST Tokyo, Japan) improved the translator from Isabelle to Dedukti and Lambdapi. It is now possible to export most of the Isabelle/HOL standard library.

7.4.2 From PVS to DEDUKTI

Participants: Gabriel Hondet, Frédéric Blanqui, Gilles Dowek.

Gabriel Hondet developed a tool, Personoj, to translate the terms and statements of PVS to Lambdapi.

7.4.3 From impredicative to predicative type theory

Participants: Thiago Felicissimo, Frédéric Blanqui, Gilles Dowek.

Thiago Felicissimo, Frédéric Blanqui and Ashish Kumar Barnawal (former intern from India) worked on the translation of proofs in impredicative type systems to predicative ones using universe polymorphism [28]. As the development of formal proofs is a time-consuming task, it is important to devise ways of sharing the already written proofs to prevent wasting time redoing them. One of the challenges in this domain is to translate proofs written in proof assistants based on impredicative logics, such as Coq, Matita and the HOL family, to proof assistants based on predicative logics like Agda, whenever impredicativity is not used in an essential way. They developed an algorithm to do such a translation between a core impredicative type system and a core predicative one allowing prenex universe polymorphism like in Agda. It consists in trying to turn a potentially impredicative term into a universe polymorphic term as general as possible. The use of universe polymorphism is justified by the fact that mapping an impredicative universe to a fixed predicative one is not sufficient in most cases. During the algorithm, one needs to solve unification problems modulo the max-successor algebra on universe levels. But, in this algebra, there are solvable problems having no most general solution. They however provide an incomplete algorithm whose solutions, when it succeeds, are most general ones. The proposed translation is of course partial, but in practice allows one to translate many proofs that do not use impredicativity in an essential way. Indeed, it was implemented in the tool Predicativize and then used to translate semi-automatically many non-trivial developments from Matita's arithmetic library to Agda, including Bertrand's Postulate and Fermat's Little Theorem, which were not available in Agda yet.

7.4.4 From Coq to Dedukti

Participants: Yoan Géran, Olivier Hermant, Gilles Dowek.

Yoan Géran worked on an encoding of the Coq universes in order to improve the translation from Coq to Dedukti. These universes are expressed with the functions max and successor, the natural numbers and variables together with another symbol R to handle impredicativity such that $R(x,0) = 0$ and $R(x,s(y)) = max(x,s(y))$. Yoan Géran showed that any term of this grammar can be written as the maximum of a list of sub-terms where the sub-terms are expressed in a sub-grammar, and the study of this sub-grammar led to a normal form for its terms.

7.4.5 From Predicate logic to the tactic language of Coq

Participants: Yoan Géran, Olivier Hermant, Gilles Dowek.

Yoan Géran has implemented a tool to translate proofs from a Dedukti expression of Predicate logic to the tactic language of Coq. As an example, he translated the proofs of the first book of Euclid Elements: dktactgeo on github.

These proofs were obtained in three steps. The original developement is a Coq library (GeoCoq/Elements/OriginalProofs on github). This library has been translated, by Gaspard Férey in a Dedukti expression of the type theory of Coq (GeoCoqInE-Euclid on github).

Yoan Géran then showed that these proofs can be expressed in Simple type theory with polymorphsm (which led to translations to Coq, HOL Light, Matita, Lean, Open Theory, and PVS using Logipedia (sttfa-geocoq-euclid on github) and to Predicate Logic (plth-geocoq-euclid on github).

The proofs obtained are then more readable than the one obtained using Logipedia, and translators from the encoding of the Predicate Logic to other proof system could be written in the same way, leading to the export of proofs in a readable format using Dedukti.

7.4.6 From the K Framework to Dedukti

Participants: Amélie Ledein, Valentin Blot, Catherine Dubois.

Amélie Ledein defined an encoding from K to Dedukti via Matching Logic, in order to execute programs within Dedukti. This work has been presented at JFLA2022 [23].

This work was then extended by a pen-and-paper formalization of the translation performed internally by K. This extension has been submitted to TYPES2022 post-proceedings [29].

Amélie Ledein defined a partial shallow embedding of Matching Logic. To validate it, Amélie Ledein checked the proof objects generated by K's automatic prover in the particular case of program execution. Ongoing work consists in isolating a simpler fragment of Matching Logic that is sufficient for expressing this kind of proofs.

7.4.7 From Metamath to Dedukti

Participants: Amélie Ledein, Valentin Blot, Catherine Dubois.

Metamath is an american logical framework from the 90's. Matching Logic (the theoretical foundation of K) was formalized within this framework. Moreover, 74 out of the 100 theorems of Freek Wiedijk's list were formalized in metamath.

Amélie Ledein defined both a deep and a shallow emdbeddings of Metamath into DEDUKTI [24]. During his M1 internship, Elliot Butte contributed to the implementation of the shallow embedding.

One future challenge consists in trying to determine the encoded features (subtyping, overloading, etc.) before translating them. Another challenge would be to define an interpretation, within DEDUKTI, from a deep embedding into a shallow one.

7.5 Other research projects

7.5.1 Automation for the Coq proof assistant

Participants: Valentin Blot, Louise Dubois de Prisque, Chantal Keller, Pierre Vial.

In order to automatize the Coq proof assistant, tactics which send a first-order goal to SMT solvers are available in the SMTCoq plugin.

Valentin Blot, Louise Dubois de Prisque and Pierre Vial, with the external collaboration of Denis Cousineau, Enzo Crance, Chantal Keller and Assia Mahboubi, developed a new Coq automatic tactic [16] which generates and proves first-order statements about Coq terms (datatypes and functions). This enriches the semantics of information transmitted to SMTCoq and increases automation in the Coq proof assistant.

This tactic snipe is modular and combines independent transformations, which allows incremental development.

7.5.2 Extensions of proof assistants with rewrite rules

Participants: Yann Leray, Théo Winterhalter.

Yann Leray has been working on an extension of the COQ proof assistant with user-defined rewrite rules: both on the practical level (as a fork of the COQ repository for now) and on the theoretical level. This latter point is performed as part of the METACOQ project aiming to specify and verify COQ as well as studying its meta-theory. In other words, the idea is to both be able to enjoy and test new features while figuring out the proper restrictions needed to make sure good properties of the system (such as confluence or type safety) are preserved.

Théo Winterhalter has been working on pushing this idea of rewrite rules even further by considering local computation instead. More precisely, a type theory can be extended with a local binder for new computational theories, the same way new hypotheses can currently be introduced. This would allow a user to locally rely on rewrite rules to perform a proof, without polluting the global theory, or the base of assumptions of the whole system.

7.5.3 Bar recursive interpretation of second-order arithmetic

Participants: Valentin Blot.

Valentin Blot defined a bar recursive computational interpretation of second-order arithmetic presented as a second-order theory with quantification on predicates [15], rather than a first-order theory equipped with the axiom scheme of comprehension. This brings closer together the two existing interpretations of second-order arithmetic: polymorphic λ-calculus and bar recursion.

7.5.4 Quantum Computing

Participants: Gilles Dowek, Alejandro Díaz-Caro.

Gilles Dowek and Alejandro Díaz-Caro have extended linear logic with two rules for addtion and multiplication and shown a purely syntactic linearity theorem for this logic: the closed proof-terms of a linear implication commute with addition and multiplication by a scalar. Together with the supperposition connective introduced in a previous work, the proof-terms of this logic form a complete quantum programming language. This work has been presented at FSCD 2022 [17].

7.5.5 Encoding the diagrammatic reasoning in Type Theory

Participants: Luc Chabassier, Bruno Barras.

In his PhD supervised by Bruno Barras, Luc Chabassier has carried out experiments to figure out how the difficulty of categoretical proof could be reduced in standard proof assistants, more specifically on Coq. He started developing a decision procedure for problems that were usually solved with diagrammatic reasoning, in an attempt to circumvent the difficulty of embedding diagrammatic reasoning in Coq. However, as this problem is undecidable. he designed a pretty good but incomplete algorithm for this problem, and he implemented it as a Coq plugin, released on opam. At that point it turned out to be too simple for real use-cases, but further work may extend it to cover more cases, making it more usable.

Another way to face the undecidability of the above problem is to create an interactive way to work with diagrams in proof assistants, that would use the previously created procedure as an helper. Luc Chabassier has started the design and implementation of such an idea. However it is not yet ready for release.

8 Bilateral contracts and grants with industry

8.1 Bilateral contracts with industry

Participants: Valentin Blot, Pierre Vial, Boris Djalal, Louise Dubois De Prisque.

Valentin Blot and Chantal Keller have funding for a 4-year project (2021–2025) 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.

9 Partnerships and cooperations

9.1 International initiatives

9.1.1 STIC/MATH/CLIMAT AmSud projects

QAPLA

Title: Qapla', Quantum aspects of programming langages

Program: STIC-AmSud

Duration: January 1, 2021 – December 31, 2022

Local supervisor: Gilles Dowek

Partners:

- Universidad de Chile (Chili)
- Universidad de la República (Uruguay)

Inria contact: Gilles Dowek

Summary: QAPLA investigates develops a logical approach to the development of quantum programming languages

9.1.2 Participation in other International Programs

PHC Sakura project

Participants: Frédéric Blanqui(PI) , Thiago Felicissimo.

Title: ADVANCED HIGH-ASSURANCE SOFTWARE TECHNOLOGY BY PROOF ASSISTANTS WITH HIGHER-ORDER REWRITING

Partner Institution(s):
- Gunma University, Kiryu, Japan
- AIST, Tokyo, Japan

Date/Duration: 2 years, 2022-2023

Funding: Partenariat Hubert Curien (PHC) franco-japonais

Web site: https://blanqui.gitlabpages.inria.fr/sakura.html

9.2 International research visitors

9.2.1 Visits to international teams

Research stays abroad Frédéric Blanqui and Thiago Felicissimo both visited for 2 weeks Akihisa Yamada and Jérémy Dubut, AIST Tokyo, Japan, and Makoto Hamana, Gunma University, Kiryu, Japan (PHC Sakura project).

Amélie Ledein visited for 2 weeks the K Framework team in Iasi and Bucarest (Romania).

9.3 European initiatives

9.3.1 Horizon Europe

COST action CA20111 EuroProofNet

Description: EuroProofNet is the European research network on digital proofs. It aims at boosting the interoperability and usability of proof systems. It has more than 300 participants from 42 different countries. It is chaired by Frédéric Blanqui.

Date/Duration: 4 years, 01/11/21 - 30/10/25

Funding: COST

Web site: https://europroofnet.github.io/

9.4 National initiatives

9.4.1 ICSPA

Participants: Gilles Dowek, Catherine Dubois, Olivier Hermant, Claude Stolze.

The ANR project (2022-2025) ICSPA (Interoperable and Confident Set-based Proof Assistants) has been accepted in the context of the AAPG 2021 call. It is coordinated by Catherine Dubois and has the following academic partners Samovar – Inria Grand Est – Inria Paris-Saclay – LIRMM – IRIT with the industrial partner Clearsy. The project starts on January 1st 2022. This project aims at reinforcing the confidence in proofs carried out mechanically for the set-based specification formalisms B, Event-B, and TLA+ that are used in industry.This will be done by verifying these proofs formally and independently with the proof verifier Dedukti. The project also aims at designing and implementing an exchange framework, through which those three systems can share their proofs and theories, making them effectively interoperable.

9.4.2 PROGRAMme

Participants: Gilles Dowek.

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.

10 Dissemination

10.1 Promoting scientific activities

10.1.1 Scientific events: organisation

General chair, scientific chair

- Frédéric Blanqui and Gilles Dowek organized the 1st Dedukti school at Nantes, France, on 24-25 June 2022.
- Frédéric Blanqui organized the 1st Workshop on the development, maintenance, refactoring and search of large libraries of proofs, at Tbilisi, Georgia, on 24-25 September 2022.
- Frédéric Blanqui organized the 1st Dedukti tools developers meeting at Val d’Ajol, France, on 16-18 October 2022.

Member of steering committee

- Valentin Blot is the workshop chair and a member of the steering committee of the ACM/IEEE Symposium on Logic In Computer Science (LICS).
- Frédéric Blanqui is member of the steering committee of the ACM/IEEE Symposium on Logic in Computer Science (LICS).
- Frédéric Blanqui is member of the steering committee of the international conference on types for proofs and programs (TYPES).
- Frédéric Blanqui is member of the steering committee of the international school on rewriting (ISR).

Local organizer

- Valentin Blot was the local organizer of the LHC days (Logique, Homotopie, Catégories), a working group of GDR-IM (Groupe De Recherche Informatique Mathématiques).

10.1.2 Scientific events: selection

Member of the conference program committees

- Frédéric Blanqui was PC member of the following conferences: 11th International Joint Conference on Automated Reasoning (IJCAR'22), 28th International Conference on Types for Proofs and Programs (TYPES'22).
- Théo Winterhalter was a PC member of the 50th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL 2023).
- Olivier Hermant was a PC member of the "Approches Formelles dans l'Assistance au Développement de Logiciels" conference (AFADL 2022).
- Catherine Dubois was a PC member of the "Approches Formelles dans l'Assistance au Développement de Logiciels" conference (AFADL 2022).

10.1.3 Invited talks

- Frédéric Blanqui gave a two days lecture on proof interoperability at the 2022 Summer School on Verification Technology, Systems & Applications (VTSA), Saarbrücken, Germany.
- Gilles Dowek has been an invited speaker at IJCAR 2022 [18].

10.1.4 Leadership within the scientific community

- Frédéric Blanqui is chair of the COST action CA20111 EuroProofNet, the European research network on digital proofs (>300 participants from 42 countries).
- Catherine Dubois is co-chair of the GDR Génie de la Programmation et du Logiciel (GDR GPL).
- Olivier Hermant is a member of the scientific board of the GDR GPL.

10.1.5 Research administration

- Frédéric Blanqui is co-director of the STIC doctoral school on information and computer science of the University Paris Saclay (>500 PhD students).
- Frédéric Blanqui is member of the INRIA Evaluation Committee.
- Frédéric Blanqui is member of the INRIA Saclay Scientific Committee.

10.2 Teaching - Supervision - Juries

10.2.1 Teaching

- Master: Frédéric Blanqui, formal languages, 21h, M1, ENSIIE, France
- Master: Frédéric Blanqui, rewriting theory, 14h, M1, ENS Paris-Saclay, France
- Master: Gilles Dowek, Foundations of proof systems, 22h30, M2, MPRI, France
- Master: Bruno Barras, Proof Assistants, 12h, M2, MPRI, France
- Master: Théo Winterhalter, Foundations of proof systems (master class), 1h30, M2, MPRI, France
- Master: Amélie Ledein, Software Engineering, 30h, M1, ENS Paris-Saclay, France

- License: Thiago Felicissimo, functional programming - TP, 18h, Polytech Paris-Saclay, France
- License: Thiago Felicissimo, logic - TD, 22h, L3, Faculté des Sciences d'Orsay, France
- License: Thiago Felicissimo, compilation - TD/TP, 24h, L3, Faculté des Sciences d'Orsay, France
- License: Thiago Felicissimo, object-oriented programming - TP, 21h, IUT d'Orsay, France
- License: Amélie Ledein, Compilation project, 15h, L3, ENS Paris-Saclay, France
- License: Amélie Ledein, Logic project, 22h30, L3, ENS Paris-Saclay, France
- License: Luc Chabassier, Logique, L3, 30h, ENS Paris-Saclay, France
- License: Luc Chabassier, Projet base de données, 22h30, L3, ENS Paris-Saclay, France
- License: Luc Chabassier, Archtecture et système, 22h30, L3, ENS Paris-Saclay, France
- IUT: Yoan Géran, Algorithmic and programmation, 45h, IUT d'Orsay, France
- IUT: Luc Chabassier, C++ R101-2, première année, 38h30, IUT d'Orsay, France
- IUT: Luc Chabassier, Projet c++ S102, première année, 10h30, IUT d'Orsay, France
- ENSTA Paris: Yoan Géran, Algorithmic, 16h

10.2.2 Supervision

- Frédéric Blanqui is the PhD supervisor of Gabriel Hondet and Thiago Felicissimo.
- Gilles Dowek is the PhD supervisor of Gabriel Hondet, Émilie Grienenberger, Thiago Felicissimo, and Yoan Géran.
- Olivier Hermant is the PhD supervisor of Yoan Géran.
- Chantal Keller is the PhD supervisor of Louise Dubois de Prisque.
- Catherine Dubois is the PhD supervisor of Amélie Ledein.
- Valentin Blot is the PhD supervisor of Amélie Ledein and Louise Dubois de Prisque.
- Bruno Barras is the PhD supervisor of Luc Chabassier.
- Théo Winterhalter supervises Yann Leray for a six months internship (post master).
- Gilles Dowek supervises Thomas Traversié and Corentin Chabanol for their programme recherche.
- Valentin Blot supervises Thomas Traversié for his programme recherche.
- Bruno Barras supervises the internship (master) of Émile Oléon and the programme recherche of Loris Cros.
- Jean-Pierre Jouannaud supervises Corentin Chabanol for his programme recherche.

10.2.3 Juries

- Frédéric Blanqui was reviewer of the PhD thesis of Hans-Jorg Schurr (University of Nancy) on "Stronger SMT solvers for proof assistants – Proofs, quantifier simplification, strategy schedules".

10.3 Popularization

10.3.1 Internal or external Inria responsibilities

Gilles Dowek is a member of the Conseil ational du numérique.

Gilles Dowek is a member of the Conseil National Pilote d'Éthique du numérique.

Gilles Dowek is a member of the Scientific board of the Société informatique de France.

10.3.2 Articles and contents

- Frédéric Blanqui published an article on proof system interoperability on Binaire on 6 May 2022: L'interopérabilité des systèmes de preuve.
- Inria published an article of EuroProofNet on 1st March 2022: Proof assistants: strengthening the position of the EU through EuroProofNet.

11 Scientific production

11.1 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).

11.2 Publications of the year

International journals

[12] G. Dowek, G. Férey, J.-P. Jouannaud and J. Liu. 'Confluence of left-linear higher-order rewrite theories by checking their nested critical pairs'. In: *Mathematical Structures in Computer Science* (22nd Jan. 2022), pp. 1–36. DOI: 10.1017/S0960129522000044. URL: https://hal.inria.fr/hal-03126111.

[13] J.-P. Jouannaud and F. Orejas. 'Unification of Drags and Confluence of Drag Rewriting'. In: *Journal of Logical and Algebraic Methods in Programming* 131 (Feb. 2023), p. 26. DOI: 10.1016/j.jlamp.2022.100845. URL: https://hal.inria.fr/hal-02562152.

International peer-reviewed conferences

[14] F. Blanqui. 'Encoding Type Universes Without Using Matching Modulo Associativity and Commutativity'. In: FSCD 2022 - 7th International Conference on Formal Structures for Computation and Deduction. Vol. 228. 7th International Conference on Formal Structures for Computation and Deduction (FSCD 2022). Haifa, Israel, 28th June 2022, p. 14. DOI: 10.4230/LIPIcs.FSCD.2022.24. URL: https://hal.inria.fr/hal-03708036.

[15] V. Blot. 'A direct computational interpretation of second-order arithmetic via update recursion'. In: LICS 2022 - 37th Annual ACM/IEEE Symposium on Logic in Computer Science. Haïfa, Israel, 2nd Aug. 2022. DOI: 10.1145/3531130.3532458. URL: https://hal.inria.fr/hal-03698879.

[16] V. Blot, D. Cousineau, E. Crance, L. Dubois de Prisque, C. Keller, A. Mahboubi and P. Vial. 'Compositional pre-processing for automated reasoning in dependent type theory'. In: CPP 2023 - Certified Programs and Proofs. Boston, United States, 2023. DOI: 10.1145/3573105.3575676. URL: https://hal.inria.fr/hal-03901019.

[17] A. Díaz-Caro and G. Dowek. 'Linear Lambda-Calculus is Linear'. In: *LIPICS*. Formal Structures for Computation and Deduction. Vol. 228. Haifa, Israel, 2nd Aug. 2022. DOI: 10.4230/LIPIcs.FSCD.2022.21. URL: https://hal.inria.fr/hal-03959343.

[18] G. Dowek. 'From the Universality of Mathematical Truth to the Interoperability of Proof Systems'. In: *LNAI*. International Joint Conference on Automated Reasoning. Vol. 13385. Haifa, Israel, 8th Aug. 2022. URL: https://hal.inria.fr/hal-03959359.

[19] M. Färber. 'Safe, Fast, Concurrent Proof Checking for the lambda-Pi Calculus Modulo Rewriting'. In: 11th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP '22). Philadelphia, PA, United States, 17th Jan. 2022. DOI: 10.1145/3497775.3503683. URL: https://hal.inria.fr/hal-03143359.

[20] T. Felicissimo. 'Adequate and Computational Encodings in the Logical Framework Dedukti'. In: FSCD 2022 - 7th International Conference on Formal Structures for Computation and Deduction. Haifa, Israel, 2nd Aug. 2022. DOI: 10.4230/LIPIcs.FSCD.2022.25. URL: https://hal.inria.fr/hal-03956666.

[21] A. Kissinger, R. Vilmart and J. van de Wetering. 'Classical simulation of quantum circuits with partial and graphical stabiliser decompositions'. In: 17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022). Vol. 232. 17th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2022) 5. Urbana-Champaign, United States, 4th July 2022, 5:1–5:13. DOI: 10.4230/LIPIcs.TQC.2022.5. URL: https://hal.science/hal-03606226.

National peer-reviewed Conferences

[22] V. Blot, L. Dubois de Prisque, C. Keller and P. Vial. 'Bécassine à la chasse au Coq (démonstration)'. In: *Journées Francophones des Langages Applicatifs*. JFLA 2022 - Journées Francophones des Langages Applicatifs. Saint-Médard-d'Excideuil, France, 28th June 2022. URL: https://hal.science/hal-03604902.

[23] A. Ledein, V. Blot and C. Dubois. 'Vers une traduction de K en Dedukti'. In: *Journées Francophones des Langages Applicatifs*. JFLA 2022 - Journées Francophones des Langages Applicatifs. Saint-Médard-d'Excideuil, France, 28th June 2022. URL: https://hal.science/hal-03604962.

[24] A. Ledein and E. Butte. 'Traduire l'univers des mathématiques en Dedukti, sans univers'. In: *Journées Francophones des Langages Applicatifs*. JFLA 2023 - 34èmes Journées Francophones des Langages Applicatifs. Praz-sur-Arly, France, 16th Jan. 2023, pp. 172–189. URL: https://hal.inria.fr/hal-03936696.

Conferences without proceedings

[25] V. Blot, G. Dowek and T. Traversié. 'An Implementation of Set Theory with Pointed Graphs in Dedukti'. In: LFMTP 2022 - International Workshop on Logical Frameworks and Meta-Languages : Theory and Practice. Haïfa, Israel, 1st Aug. 2022. URL: https://hal.inria.fr/hal-03740004.

Doctoral dissertations and habilitation theses

[26] G. Hondet. 'Expressing predicate subtyping in computational logical frameworks'. Université Paris-Saclay, 27th Sept. 2022. URL: https://theses.hal.science/tel-03855351.

Reports & preprints

[27] A. Díaz-Caro and G. Dowek. *A New Connective in Natural Deduction, and its Application to Quantum Computing* ⋆. 23rd Jan. 2022. URL: https://hal.inria.fr/hal-03540150.

[28] T. Felicissimo, F. Blanqui and A. Kumar Barnawal. *Translating proofs from an impredicative type system to a predicative one*. 10th Nov. 2022. URL: https://hal.inria.fr/hal-03848584.

[29] A. Ledein, V. Blot and C. Dubois. *A semantics of K into Dedukti*. 13th Dec. 2022. DOI: 10.4230/LIPIcs.TYPES.2022.23. URL: https://hal.inria.fr/hal-03895834.

11.3 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.

[32] G. Férey and J.-P. Jouannaud. 'Confluence in Non-Left-Linear Untyped Higher-Order Rewrite Theories'. In: *PPDP 2021 - 23rd International Symposium on Principles and Practice of Declarative Programming*. Tallin, Estonia, Sept. 2021. DOI: 10.1145/3479394.3479403. URL: https://hal.inria.fr/hal-03126115.

[33] R. Saillard. 'Typechecking in the $\lambda\Pi$-Calculus Modulo: Theory and Practice'. PhD thesis. MINES ParisTech, 2015.