RESEARCH CENTRE

Inria Saclay Centre
at Université Paris-Saclay

IN PARTNERSHIP WITH:
Université Paris-Saclay

2023
ACTIVITY REPORT

Project-Team
DEDUCTEAM

DEDUCTEAM

IN COLLABORATION WITH: Laboratoire de Méthodes Formelles

DOMAIN

Algorithmics, Programming, Software and Architecture

THEME

Proofs and Verification

Inria

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]
- Anthony Bordg [INRIA, Advanced Research Position, from Sep 2023]
- Theo Winterhalter [INRIA, Researcher, from Oct 2023]

Faculty Member

- Catherine Dubois [ENSIIE, Professor, from Sep 2023, HDR]

Post-Doctoral Fellow

- Claude Stolze-Hubert [INRIA, Post-Doctoral Fellow]

PhD Students

- Luc Chabassier [ENS PARIS]
- Louise Dubois De Prisque [INRIA, from Nov 2023]
- Thiago Felicissimo Cesar [UNIV PARIS SACLAY]
- Amélie Ledein [INRIA, until Sep 2023]
- Nicolas Margulies [ENS PARIS-SACLAY, from Sep 2023]
- Thomas Traversie [CENTRALESUPELEC, from Oct 2023]
- Rishikesh Hirendu Vaishnav [INRIA, from Mar 2023]

Administrative Assistant

- Aissatou-Sadio Diallo [INRIA, from May 2023]

External Collaborators

- Guillaume Burel [ENSIIE]
- Alessio Coltellacchi [Loria, PhD]
- Catherine Dubois [ENSIIE, until Aug 2023, HDR]
- Yoan Geran [Mines Paris PSL]
- Olivier Hermant [Mines Paris PSL]
- Jean-Pierre Jouannaud [INRIA, HDR]
- Chantal Keller [Université Paris-Saclay]
- Amélie Ledein [U. Strasbourg, from Oct 2023, Ater]

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 2022: *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 [33]. 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 Euclid's 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* [35].

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, is a synthesis of the Edinburgh logical framework and of Deduction modulo theory, and 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, and from one system to another via Dedukti.

3.2 Interoperability, cross verification 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 investigate 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. We also develop tools for checking proofs in the TSTP and Alethe formats generated by automated theorem provers and 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

5.1 Awards

- Gilles Dowek has been awarded the Grand Prix Inria - Académie des Sciences 2023.
- Théo Winterhalter received the Amazing Reviewer Award for his work on the program committee of CPP 2024.

6 New software, platforms, open data

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 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

6.1.11 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

6.1.12 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

6.1.13 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: Amelie Ledein

6.1.14 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: Amelie Ledein

Participant: Elliot Butte

6.1.15 BiTTs

Keywords: Dependent types, Logical Framework

Functional Description: This is an implementation of the generic bidirectional typing algorithm presented in the paper "Generic bidirectional typing for dependent type theories".

URL: https://github.com/thiagofelicissimo/BiTTs

Contact: Thiago Felicissimo Cesar

6.1.16 pogtranslator

Keywords: Formal methods, Proof

Functional Description: Translator of Atelier B proof obligations (in the POG format) to the TPTP or SMT-LIB format.

Contact: Claude Stolze

6.1.17 sniper

Keywords: Coq, Automated deduction

Functional Description: Sniper is a Coq plugin that improves its automation.

URL: https://github.com/smtcoq/sniper

Contact: Chantal Keller

Partner: Université Paris-Saclay

6.1.18 hol2dk

Keywords: Interoperability, Proof

Functional Description: Tool making HOL-Light generate proofs, simplifying those proofs, and translating those proofs to Dedukti, Lambdapi and Coq.

URL: https://github.com/Deducteam/hol2dk

Contact: Frederic Blanqui

6.1.19 commutative-diagrams

Name: Commutative diagrams proof assistant

Keyword: Proof assistant

Functional Description: A coq plugin enabling to progress categoretical proofs graphically. It can infer the diagram from the proof context, and display it graphically to the user. The user's action on the diagram are then converted into Coq proofs.

URL: https://github.com/dwarfmaster/commutative-diagrams

Contact: Luc Chabassier

6.1.20 dkpltact

Keywords: Coq, Interoperability, Proof

Functional Description: A tool to translate proofs from a Dedukti encoding of Predicate logic to the tactic language of Coq. It takes Dedukti files whose terms comply with this encoding and produces the corresponding Coq files.

URL: https://gitlab.crans.org/geran/dkpltact

Contact: Yoan Geran

7 New results

7.1 Implementations of DEDUKTI

7.1.1 Lambdapi

Participants: Frédéric Blanqui.

Lambdapi has been improved and extended in various ways. The most notable novelties are:

- Lambdapi can now translate to Coq any Dedukti or Lambdapi files using a user-defined encoding of higher-order logic. Moreover, some Dedukti or Lambdapi symbol can be renamed or replaced by Coq expressions in order to align those symbols to the one already defined in the Coq standard library.
- Claudio Sacerdoti from the University of Bologna added commands for indexing constants, definitions and rewrite rules; searching such items matching some patterns and conditions; running a web server for answering such requests.

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

7.2.1 Confluence and levels

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

In a dependently typed lambda calculus, subject reduction, confluence and termination are inter-dependent, which makes difficult to add dependently typed higher-order rewrite rules, as needed is some complex encodings. It then makes sense to check confluence in the untyped lambda-calculus. The case of left-linear rewrite rules is treated in [36]: confluence is preserved by adding terminating rewrite rules whose critical pairs are joinable by Van Oostrom's decreasing diagrams. Unfortunately, the use of higher-order rewrite rules with non-linear left-hand sides destroys the confluence property of the untyped lambda-calculus, this is the case with very simple critical pair free rewrite rules like F(x) -> x. In [38], it is shown that confluence is preserved on a subset of *layered terms*, provided "nested critical pairs" are joinable by some "layer non-increasing" van Oostrom decreasing diagram. A yet open question is under which assumptions the set of layered terms contains all typable terms of interest, a property that happens to be true in some practical cases. In a yet unpublished work, we have described a way to layer even more terms by a simpler definition of layering which is at the same time more easily implementable, a first simple step towards a solution to this question.

7.2.2 Generic bidirectional typing for dependent type theories

Participants: Thiago Felicissimo, Frédéric Blanqui, Gilles Dowek.

Bidirectional typing is a discipline in which the typing judgment is decomposed explicitly into inference and checking modes, allowing to control the flow of type information in typing rules and to specify algorithmically how they should be used. Bidirectional typing has been fruitfully studied and bidirectional systems have been developed for many type theories. However, the formal development of bidirectional typing has until now been kept confined to specific theories, with general guidelines remaining informal. In this work [28], we give a generic account of bidirectional typing for a general class of dependent type theories.

As a practical outcome, we obtain a theory-independent bidirectional typechecker that has been implemented in a prototype and used in practice with many theories. The use of bidirectionality allows in particular the omission of many type annotations, proividing a much more succint syntax when compared with fully-annotated presentations of type theory as available in logical frameworks. As a result, we expect our implementation to provide important performance gains when compared with Dedukti. This work has been accepted at ESOP 2024.

7.3 Expressing theories in DEDUKTI

7.3.1 Universes

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

The imax operator defined by $\mathrm{imax}(x,0)\ \ 0$ and $\mathrm{imax}(x,s(y)\ \ \max(x,s(y))$ is used to represent universes in impredicative theories. Yoan Géran has given a canonical form for the term of the imax-successor algebra, leading to a decision procedure for the equality and inequality problem in this algebra [29]. Rishikesh Vaishnav implemented it in lean2dk (see below).

7.3.2 Set theory

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

We are currently expressing two set-based specification formalisms used in industry, B and TLA+ and their proof tools. A translator, called pogtranslator, has been developed: it translates proof obligations generated by Atelier B expressed in the framework of the B set theory, into TPTP proof obligations, expressed in first-order logic.

TLAPS, the TLA+ proof system, is a proof assistant that mechanically checks TLA+ proofs by calling automatic provers such as veriT, cvc4, cvc5, or Zenon, on proof obligations. In collaboration with Stephan Merz (Loria), we are developing a ckecker for these proofs by reconstructing a proof term from a trace in the new Alethe proof format[34]. The term produced uses the encoding of TLA+ in DEDUKTI as defined by Stephan Merz.

7.3.3 Cubical Type Theory

Participants: Nicolas Margulies, Bruno Barras.

During his internship supervised by Bruno Barras, Nicolas Margulies has integrated the various elements of an proof import procedure from Cubical to Dedukti. He has mainly worked on two components. Firstly, he has updated the encoding of Cubical Type Theory to follow the minor evolutions of this language. He also adapted the work of Luc Chabassier (translation from extensional to intensional type theory inside Dedukti), to translate Cubical proofs in their usual presentation into the encoding of Cubical Type Theory as a 2-level Type Theory. This adaptation appeared to be tedious but most of it has been implemented.

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) continued to improve isabelle_dedukti, the translator from Isabelle to Dedukti and Lambdapi. It is now possible to export most of the Isabelle/HOL standard library, as well as some libraries of the Archive of Formal Proofs (AFP). Frédéric Blanqui started also to work on the translation of the obtained Dedukti files to Coq.

7.4.2 From HOL-Light to Lambdapi and Coq

Participants: Frédéric Blanqui, Anthony Bordg.

hol2dk is a new software making HOL-Light to generate proofs, simplifying them, and translating them to Dedukti and Lambdapi, and in turn to Coq by using the new export feature of Lambdapi described above. To translate the proofs generated by HOL-Light, which can be quite big, it is necessary to simplify them, prune useless proofs and translate them in parallel. hol2dk can currently handle the whole base library of HOL-Light as well as some other libraries. Some further improvement is necessary to handle all the HOL-Light libraries. For the obtained Coq theorems to be usable, it is necessary to align the definitions of the types and functions of HOL-Light to those given in the Coq standard library. We did this for natural numbers and several common mathematical functions on natural numbers. The obtained Coq library is available in the Opam package coq-hol-light. Our goal is to make this alignement up to real numbers, so as to allow Coq users to import and reuse the large library of real analysis of HOL-Light to Coq.

7.4.3 From Lean to Dedukti

Participants: Rishikesh Vaishnav, Frédéric Blanqui.

For his PhD thesis started in March, Rishikesh Vaishnav started to write a framework implementation for translating Lean code to Dedukti (lean2dk) that reads in Lean code, elaborates, runs a translation function, and prints out the translated Dedukti code. He began the implementation of a Dedukti library for the encoding of Lean (generally following an interpretation of Lean as a Pure Type System). He added debugging utilities and various command line options to lean2dk control what code is translated from the input file, and wrote code to test the translation and various aspects of the rewrite systems. He worked with Yoan on the implemenation and theory of a rewrite system for deciding impredicative universe terms, and integrated this system into lean2dk. Finally, he implemented the translation of a number of features of Lean including universe impredicativity, let expressions, inductive types and recursors.

7.4.4 From impredicative to predicative type theory

Participants: Thiago Felicissimo, Frédéric Blanqui, Gilles Dowek.

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.

In 2022, we proposed 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 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.

In 2023, this work has been published at the conference Computer Science Logic 2023 (CSL 23) [19]. An extended version of this work is currently under submission for the special issue of CSL at Logical Methods in Computer Science [37].

7.4.5 From Predicate logic to the tactic language of Coq

Participants: Yoan Géran, Olivier Hermant, Gilles Dowek.

dkpltact is a software that translates proof from Predicate Logic, expressed in DEDUKTI, into the tactic language of Coq. Thus, it permits to obtain proof that are more readable and lighter than proof terms.

7.4.6 From the $\mathbb{K}$ Framework to DEDUKTI

Participants: Amélie Ledein, Valentin Blot, Catherine Dubois.

An encoding from $\mathbb{K}$ to DEDUKTI via Matching Logic using the $\mathbb{K}$ore language, in order to execute programs within DEDUKTI has been defined and implemented in the tool KaMeLo. We have contributed in particular to a paper formalization of the translation from $\mathbb{K}$ into $\mathbb{K}$ore [21]. We also defined a partial shallow embedding of Matching Logic. The latter is used to check the proof objects generated by $\mathbb{K}$'s automatic prover in the particular case of program execution.

7.4.7 From Metamath to DEDUKTI

Participants: Amélie Ledein, Valentin Blot.

The Metamath formal language for specification of mathematical proofs, comes with a proof checker. A deep and a shallow embedding of Metamath into DEDUKTI have been defined. With an extended version of the deep encoding, all the proofs of the Metamath standard library have been translated into DEDUKTI and checked by it using the tool MM2DK [23].

7.4.8 From a variant of extensional type theory using ghost types

Participants: Théo Winterhalter.

We have developed a new version of extensional type theory where equality reflection is restricted to certain types so that the type theory still enjoys desirable properties like type constructor discrimination (the ability to distinguish, *e.g.* the type of natural numbers from a function type) and termination (although this last point remains a conjecture for now). We show that this theory is conservative over an intensional type theory, without having to rely on the usual axioms of function extensionality and uniqueness of indentity proofs.

We build this restriction by considering ghost dependent types. Values in a ghost type can be safely erased for computation (for instance at extraction), but are nevertheless distinguishable. They thus have a spot in-between propositions (whose proofs are all equal) and relevant data. In the type theory we consider, reflection is restricted to those ghost values. We have written a preprint [32] containing two translations, one from ghost extensional type theory to ghost type theory and one for ghost type theory to the usual intensional one with a universe of definitionally proof-irrelevant propositions (for instance that of Coq or Agda), showing consistency of the two theories.

7.4.9 From rewrite rules to axioms

Participants: Thomas Traversié, Valentin Blot, Gilles Dowek, Théo Winterhalter.

We studied [25] the possibility to transform proofs of the $\lambda\Pi$-calculus modulo rewriting so that we replace the use of user-defined rewrite rules by the use of equational axioms instead. This work has been published in Foundations of Software Science and Computation Structures 2024 (FoSSaCS 2024). This result paves the way for its implementation in DEDUKTI, that would allow one to get rid of rewrite rules used for one encoding of a theory in order to produce a proof in a different system without these rules.

7.5 Other research projects

7.5.1 Automation for the Coq proof assistant

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

In order to automatize the Coq proof assistant, Valentin Blot and Louise Dubois de Prisque, with the external collaboration of Chantal Keller, develop the Sniper plugin [18] (see 6.1.17).

The plugin contains:

- a library of fine-grained certifying logical transformations
- a tactic that combines these transformations in order to translate a subset of Coq goals into first-order logic, then calls external SMT solvers (through the SMTCoq plugin).

The use of modular and independent transformations allows incremental development. A rewriting of the orchestrator that combines them, in order to make the tactic more powerful and more efficient, is under progress.

7.5.2 Extensions of proof assistants with rewrite rules

Participants: Yann Leray, Théo Winterhalter.

We have started an implementation of user-defined rewrite rules in the Coq proof assistant, which, although still at an experimental stage, is waiting to be integrated in a coming official release. This practical was conducted in parallel to a formalisation of the meta-theory of Coq extended with rewrite rules as part of the MetaCoq project which contains a specification of Coq's type theory, theorems about its meta-theory and a certified implementation of type checker. This is still work in progress, but we identified a criterion to ensure confluence of whole system and proceeded with its formal proof of correctness.

7.5.3 Diller-Nahm bar recursion

Participants: Valentin Blot.

Valentin Blot described an interpretation of the double-negation shift (and hence of the classical axiom of countable choice and of second-order arithmetic) in the Diller-Nahm variant of the Dialectica interpretation [17]. Using the Diller-Nahm variant allows in particular for non-decidable atomic formulas, and provides a naturally structured interpretation.

7.5.4 Quantum Computing

Participants: Gilles Dowek, Alejandro Díaz-Caro.

Alejandro Díaz-Caro and Gilles Dowek are investingting applications of proof-theory to the design of quantum progrmming languages. More precisely they try to understand in which way propositional logic must be extended or restricted in such a way that its proof-term language is a quantum programming language. First, their 2021 work on the extension of propostional logic with a non-harmonious connective "sup" (for "superposition") has been published in a journal. A linear restriction of this calculus has been presented in 2022. The final version of this paper has been published in a journal [13].

A new work has been started on new introduction rues for the disjunction, that allow a better elimination process for comuting cuts. The obtained calculus has some similarities with the sup-calculus. This work will be submitted for publication in 2024.

7.5.5 Category theory in Dedukti

Participants: Luc Chabassier, Bruno Barras.

Luc Chabassier and Bruno Barras have explored the potential of using dedukti powerful definitional equality to work around the complexities of the use of dependent types in category theory formalisations. Indeed, the intrisically dependent nature of category theory means any formalisation suffers from the drawbacks of dependent types. To work around that without going to a full extensional theory, they implemented some categories in dedukti such that the rewrite system of Dedukti perfectly captured the equality on morphisms of those categories. However, despite some success on simple categories, the approach failed to generalize to more complex categories.

7.5.6 Rewriting with graphs

Participants: Jean-Pierre Jouannaud.

Jean-Pierre Jouannaud and two external collaborators (Nachum Dershowitz, Tel Aviv University, and Fernando Orejas, Universitat Politecnica de Catalugna) have developed a new algebraic framework for rewriting term-graphs equiped with variables, seen as input ports, and roots, seen as output ports of some computation. Term-graphs are just graphs whose every vertex is labeled by a function symbol whose arity dictates the number of its outgoing edges. They describe an algorithm for unification and use it for deciding local confluence (hence confluence under a termination assumption) in [16]. They have recently improved their framework and shown that it now encodes faithfully first-order term rewriting, which has been a long standing open problem only solved in particular cases so far [26]. The next, ongoing step is the generalization of this framework to arbitrary graphs equiped with variables and roots, that is, whose number of outgoing edges at a given vertex can be arbitrary.

7.5.7 Ethics and logic

Participants: Gilles Dowek.

Gilles Dowek has published a paper [14] showing a paralelism between the notion of explanation used in ethics and the notion of cut used in logic.

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.

Gilles Dowek received a grant from Amazon to hire a post-doc working on checking proofs produced by SMT solvers. The post-doctoral researcher will start at the beginning of 2024.

9 Partnerships and cooperations

9.1 International initiatives

9.1.1 Participation in other International Programs

PHC Sakura project

Participants: Frédéric Blanqui.

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

Additionnal info/keywords: Frédéric Blanqui is the French PI of the Sakura project between France and Japan with 6000 euros/year for missions.

9.2 International research visitors

9.2.1 Visits of international scientists

Other international visits to the team

Geoff Sutcliffe

Status (professor)

Institution of origin: Miami University

Country: USA

Context of the visit: Geoff Sutcliffe, main developer of the TPTP framework, visited Deducteam for one month, and extended his tool GDV to check TSTP proofs by exporting Lambdapi proofs.

Dorel Lucanu

Status professor

Institution of origin: Universitatea Alexandru Ioan Cuza

Country: Romania

Context of the visit: Dorel Lucanu visited Deducteam for two weeks. With Amélie Ledein, Valentin Blot and Catherine Dubois, he studied the design a Dedukti proof checker for the $\mathbb{K}$ prover.

9.2.2 Visits to international teams

Research stays abroad

Frédéric Blanqui

Visited institution: Gunma University and AIST Tokyo

Country: Japan

Context of the visit: Frédéric Blanqui worked for one month in total on confluence and termination of higher-order rewrite systems, and the development of isabelle_dedukti.

Luc Chabassier

Visited institution: TU Delft

Country: Netherlands

Context of the visit: Luc Chabassier worked for two weeks on his Coq plugin for commutative diagrams with Benedikt Arhens.

9.3 European initiatives

9.3.1 Other european programs/initiatives

Frédéric Blanqui is the chair of the COST action CA20111 EuroProofNet 2022-2025 which is a research network on proofs gathering more than 400 members from 43 different countries.

9.4 National initiatives

9.4.1 ICSPA

Participants: Guillaume Burel, 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 organized several Dedukti developers meetings.

Frédéric Blanqui organized with Geoff Sutcliffe the 2023 TPTP Tea Party.

Member of steering committees Valentin Blot is the workshop chair and a member of the steering committee of the ACM/IEEE Symposium on Logic In Computer Science (LICS).

Catherine Dubois is the chair of the steering committee of the international conference Test and Proof (TAP).

10.1.2 Scientific events: selection

Chair of conference program committees Catherine Dubois was chairing with Manfred Kerber the program committee of the international conference on Intelligent Computer Mathematics, held in September 2023 (CICM 2023).

Member of the conference program committees Frédéric Blanqui was a member of the program committee of the international workshop on Logical Frameworks and Meta-Languages: Theory and Practice, held in July 2023 (LFMTP 2023).

Catherine Dubois was a member of the program comittee of the international Symposium on Applied Computing, Software Verification and Testing Track, (SAC-SVT 2024), that will be held in April 2024.

Théo Winterhalter was a member of the program committee of in the international conference on Certified Programs and Proofs (CPP 2024) held in January 2024.

10.1.3 Invited talks

Frédéric Blanqui gave an invited talk at the 16th Conference on Intelligent Computer Mathematics.

Frédéric Blanqui gave a talk at the annual meeting of the IFIP WG1.6 on rewriting.

10.1.4 Leadership within the scientific community

Frédéric Blanqui is member of the IFIP WG1.6 on rewriting.

10.1.5 Research administration

Frédéric Blanqui was elected member of the Evaluation committee of Inria until August 2023.

Frédéric Blanqui is member of the Scientific committee of Inria Saclay.

Frédéric Blanqui is the chair of the COST action CA20111 EuroProofNet 2022-2025 which is a research network on proofs gathering more than 400 members from 43 different countries.

Catherine Dubois is one of the two co-chairs of Groupement de Recherche Génie de la Programmation et du Logiciel (Gdr GPL).

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: Théo Winterhalter, Proof Assistants, 12h, M2, MPRI, France
- Master: Amélie Ledein, Software Engineering, 30h, M1, ENS Paris-Saclay, 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, Architecture et système, 22h30, L3, ENS Paris-Saclay, France
- License: Nicolas Margulies, Compilation project, 15h, L3, ENS Paris-Saclay, France
- License: Nicolas Margulies, Architecture et système, 22h30, L3, ENS Paris-Saclay, France
- License: Yoan Géran, Compilation project, 15h, L3, ENS Paris-Saclay, France
- License: Yoan Géran, Projet Programmation, 30h, L3, ENS Paris-Saclay, 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
- IUT: Claude Stolze, C++ R101-2, première année, 33h, IUT d'Orsay, France
- Engineering school: Thomas Traversié, Algorithmique et complexité, 18h, first-year engineering school, CentraleSupélec, France
- Engineering school: Thomas Traversié, Modélisation logique - Langages et automates, 12h, third-year engineering school, CentraleSupélec, France

10.2.2 Supervision

- Valentin Blot and Chantal Keller are supervising the PhD of Louise Dubois de Prisque.
- Catherine Dubois and Valentin Blot are supervising the PhD of Amélie Ledein.
- Catherine Dubois and Burkhart Wolff are supervising the PhD of Benoit Ballenghien.
- Bruno Barras and Gilles Dowek are supervising the PhD of Luc Chabassier.
- Bruno Barras and Gilles Dowek are supervising the PhD of Nicolas Margulies.
- Frédéric Blanqui is supervising the PhD of Rishikesh Vaishnav.
- Frédéric Blanqui and Gilles Dowek are supervising the PhD of Thiago Felicissimo.
- Olivier Hermant and Gilles Dowek are supervising the PhD of Yoan Géran.
- Marc Aiguier and Gilles Dowek are supervising the PhD of Thomas Traversié.
- Stephan Merz and Gilles Dowek are supervising the PhD of Alessio Coltellacci.

10.2.3 Juries

- Catherine Dubois was the president of the jury for the PhD defence of Rosalie Defourné (Université de Lorraine), on 7 November 2023.
- Catherine Dubois was the president of the jury for the PhD defence of Wendlasida Ouédrago (Institut Polytechnique de Paris), on 15 September 2023.
- Théo Winterhalter was an examiner for the PhD defence of Enzo Crance (Nantes Université), 21 December 2023.
- Gilles Dowek was an examiner for the PhD defence of Julie Cailler, December 2023 (Université de Montpellier).

10.3 Popularization

10.3.1 Education

Gilles Dowek has been appointed at the Conseil Supérieur des Programmes, in charge of defining the curricula for K-12 education on all topics.

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 λΠ-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 λΠ-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] F. Blanqui, G. Dowek, E. Grienenberger, G. Hondet and F. Thiré. 'A modular construction of type theories'. In: *Logical Methods in Computer Science* 19.1 (14th Feb. 2023). DOI: 10.46298/lmcs-19(1:12)2023. URL: https://inria.hal.science/hal-04317047.

[13] A. Díaz-Caro and G. Dowek. 'A New Connective in Natural Deduction, and its Application to Quantum Computing'. In: *Theoretical Computer Science* 957 (2023), p. 113840. DOI: 10.1016/j.tcs.2023.113840. URL: https://inria.hal.science/hal-04398691.

[14] G. Dowek. 'Explanation: from ethics to logic'. In: *Annals of the Japan Association for Philosophy of Science* 32 (2023), p. 16. URL: https://inria.hal.science/hal-04055141.

[15] P. Haselwarter, E. Rivas, A. van Muylder, T. Winterhalter, C. Abate, N. Sidorenco, C. Hriţcu, K. Maillard and B. Spitters. 'SSProve: A Foundational Framework for Modular Cryptographic Proofs in Coq'. In: *ACM Transactions on Programming Languages and Systems (TOPLAS)* 45.3 (30th Sept. 2023), pp. 1–61. DOI: 10.1145/3594735. URL: https://hal.science/hal-04273257.

[16] 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://inria.hal.science/hal-02562152.

International peer-reviewed conferences

[17] V. Blot. 'Diller-Nahm Bar Recursion'. In: FSCD 2023 - 8th International Conference on Formal Structures for Computation and Deduction. Vol. 260. LIPIcs. Rome, Italy, 3rd July 2023, 32:1–32:16. DOI: 10.4230/LIPIcs.FSCD.2023.32. URL: https://inria.hal.science/hal-04144888.

[18] 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, pp. 1–15. DOI: 10.1145/3573105.3575676. URL: https://inria.hal.science/hal-03901019.

[19] T. Felicissimo, F. Blanqui and A. Kumar Barnawal. 'Translating proofs from an impredicative type system to a predicative one'. In: 31st EACSL Annual Conference on Computer Science Logic (CSL 2023). Warsaw, Poland, 2023. DOI: 10.4230/LIPIcs.CSL.2023.19. URL: https://inria.hal.science/hal-03848584.

[20] G. Gilbert, Y. Leray, N. Tabareau and T. Winterhalter. 'The Rewster: The Coq Proof Assistant with Rewrite Rules'. In: TYPES 2023 - 29th International Conference on Types for Proofs and Programs. Valencia, Spain, 2023, pp. 1–3. URL: https://inria.hal.science/hal-04403667.

[21] A. Ledein, V. Blot and C. Dubois. 'A semantics of K into dedukti'. In: *TYPES 2022 - 28th International Conference on Types for Proofs and Programs*. TYPES 2022 - 28th International Conference on Types for Proofs and Programs (TYPES). Nantes, France: Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. DOI: 10.4230/LIPIcs.TYPES.2022.23. URL: https://inria.hal.science/hal-03895834.

National peer-reviewed Conferences

[22] A. Ledein, V. Blot and C. Dubois. 'Vers une traduction de K en Dedukti'. In: *JFLA 2022 - Journées Francophones des Langages Applicatifs*. JFLA 2022 - Journées Francophones des Langages Applicatifs (JFLA). Saint-Médard-d'Excideuil, France: Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. URL: https://hal.science/hal-03604962.

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

Reports & preprints

[24] F. Blanqui, A. Canteaut, H. de Jong, S. Imperiale, N. Mitton, G. Pallez, X. Pennec, X. Rival and B. Thirion. *Recommandations sur les « éditeurs de la zone grise »*. Inria, 25th Jan. 2023, pp. 1–3. URL: https://inria.hal.science/hal-04001505.

[25] V. Blot, G. Dowek, T. Traversié and T. Winterhalter. *From Rewrite Rules to Axioms in the $\lambda\Pi$-Calculus Modulo Theory*. 13th Feb. 2024. URL: https://hal.science/hal-04275229.

[26] N. Dershowitz, J.-P. Jouannaud and F. Orejas. *Drag Rewriting*. 1st June 2023. URL: https://inria.hal.science/hal-04143346.

[27] G. Dowek. *A theory independent Curry-De Bruijn-Howard correspondence*. 15th Apr. 2023. URL: https://inria.hal.science/hal-04070185.

[28] T. Felicissimo. *Generic bidirectional typing for dependent type theories*. 4th Nov. 2023. URL: https://inria.hal.science/hal-04270368.

[29] Y. Géran. *Encoding impredicative hierarchy of type universes with variables*. 28th Nov. 2023. URL: https://hal.science/hal-04311936.

[30] M. Sozeau, Y. Forster, M. Lennon-Bertrand, J. B. Nielsen, N. Tabareau and T. Winterhalter. *Correct and Complete Type Checking and Certified Erasure for Coq, in Coq*. 21st Apr. 2023. URL: https://inria.hal.science/hal-04077552.

[31] C. Stolze, O. Hermant and R. Guillaumé. *Towards Formalization and Sharing of Atelier B Proofs with Dedukti*. 16th Jan. 2024. URL: https://hal.science/hal-04398119.

[32] T. Winterhalter. *Extensionality of Ghost Dependent Types for Free*. 17th July 2023. URL: https://hal.science/hal-04163836.

11.3 Cited publications

[33] M. Boespflug. 'Conception d'un noyau de vérification de preuves pour le λΠ-calcul modulo'. PhD thesis. École Polytechnique, 2011.

[34] A. Coltellacci. 'Reconstruction of TLAPS Proofs Solved by VeriT in Lambdapi'. In: *Rigorous State-Based Methods*. Ed. by U. Glässer, J. Creissac Campos, D. Méry and P. Palanque. Cham: Springer Nature Switzerland, 2023, pp. 375–377.

[35] 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. URL: http://dx.doi.org/10.1007/978-3-642-31585-5.

[36] 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* (Jan. 2022), pp. 1–36. DOI: 10.1017/S0960129522000044. URL: https://inria.hal.science/hal-03126111.

[37] T. Felicissimo and F. Blanqui. *Sharing proofs with predicative theories through universe polymorphic elaboration*. 2023. arXiv: 2308.15465 [cs.LO].

[38] 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.