Activity Report 2014

Exploratory Action DEDUCTEAM

Deduction modulo, interopérabilité et démonstration automatique

RESEARCH CENTER
Paris - Rocquencourt

Table of contents

Exploratory Action DEDUCTEAM

Keywords: Type Systems, Proof Theory, Automated Theorem Proving, Model Of Computation, Safety

Creation of the Team: 2011 December 01*, updated into exploratory Action:* 2013 January 01, end of the Exploratory Action: 2014 December 31.

1. Members

Research Scientists

Gilles Dowek [Team leader, Inria, Senior Researcher, HdR]
Frédéric Blanqui [Inria, Researcher, HdR]

Faculty Member

David Delahaye [CNAM, Associate Professor, HdR]

PhD Students

Guillaume Bury [ENS Paris, from Mar 2014]
Ali Assaf [École Polytechnique]
Simon Cruanes [École Polytechnique]
Frédéric Gilbert [École des Ponts ParisTech]
Kailiang Ji [Inria]
Kim Quyen Ly [Inria, until Oct 2014]

Post-Doctoral Fellows

Hugo Dos Santos Macedo [Inria, until May 2014]
Vaston Goncalves Da Costa [Universidade Federal de Goias, from Mar 2014]
Alejandro Díaz-Caro [ATER Univ. Paris X, until Sep 2014]
Arnaud Spiwack [ENSM Paris, from Nov 2014]

Visiting Scientists

Edward Hermann Häusler [PUC Rio, Professor, until Apr 2014]
Ying Jiang [ISCAS, Senior Researcher, from Apr 2014 until Jun 2014]

Administrative Assistants

Hélène Milome [Inria, until May 2014]
Virginie Collette [Inria, from June 2014]

Others

Gaëtan Gilbert [ENS Lyon, Intern, from Feb 2014 until Jun 2014]
Julien Rateau [ENSIIE, Intern, from Jun 2014 until Aug 2014]
Guillaume Burel [ENSIIE, Associate Professor]
Raphaël Cauderlier [CNAM]
Catherine Dubois [ENSIIE, Professor, HdR]
Pierre Halmagrand [CNAM]
Olivier Hermant [ENSM Paris, Associate Professor]
Ronan Saillard [ENSM Paris]

2. Overall Objectives

2.1. Objectives

The team investigates applications of recent results in proof theory to the design of proof-checkers and automated theorem proving systems. It develops the Dedukti proof-checker and the iProver modulo and Zenon modulo automated theorem proving systems.

2.2. History

Deduction modulo is a formulation of predicate logic where deduction is performed modulo an equivalence relation defined on propositions. A typical example is the equivalence relation relating propositions differing only by a re-arrangement of brackets around additions, relating, for instance, the propositions $P((x+y)+z)$ and $P(x+(y+z))$. Reasoning modulo this equivalence relation permits to drop the associativity axiom. Thus, in Deduction modulo, a theory is formed with a set of axioms and an equivalence relation. When the set of axioms is empty the theory is called *purely computational*.

Deduction modulo was proposed at the end of the 20th century as a tool to simplify the completeness proof of equational resolution. Soon, it was noticed that this idea was also present in other areas of logic, such as Martin-Löf's type theory, where the equivalence relation is definitional equality, Prawitz' extended natural deduction, etc. More generally, Deduction modulo gives an account on the way reasoning and computation are articulated in a formal proof, a topic slightly neglected by logic, but of prime importance when proofs are computerized.

The early research on Deduction modulo focused on the design of general proof search methods—Resolution modulo, tableaux modulo, etc.—that could be applied to any theory formulated in Deduction modulo, to general proof normalization and cut elimination results, to the definitions of models taking the difference between reasoning and computation into account, and to the definition of specific theories—simple type theory, arithmetic, some versions of set theory, etc.—as purely computational theories.

3. Research Program

3.1. From proof-checking to Interoperability

A new turn with Deduction modulo was taken when the idea of reasoning modulo an arbitrary equivalence relation was applied to typed λ-calculi with dependent types, that permits to express proofs as algorithms, using the Brouwer-Heyting-Kolmogorov interpretation and the Curry-de Bruijn-Howard correspondence [32]. It was shown in 2007, that extending the simplest λ-calculus with dependent types, the $\lambda\Pi$-calculus, with an equivalence relation, led to a calculus we called the $\lambda\Pi$-calculus modulo, that permitted to simulate many other λ-calculi, such as the Calculus of Constructions, designed to express proofs in specific theories.

This led to the development of a general proof-checker based on the $\lambda\Pi$-calculus modulo [3], that could be used to verify proofs coming from different proof systems, such as Coq [30], HOL [39], etc. To emphasize this versatility of our proof-system, we called it Dedukti —"to deduce" in Esperanto. This system is currently developed together with companion systems, Coqine, Holide, Focalide, and Zenonide, that permits to translate proofs from Coq, HOL, Focalize, and Zenon, to Dedukti. Other tools, such as Zenon Modulo, directly output proofs that can be checked by Dedukti.

Dedukti proofs can also be exported to other systems, in particular to the MMT format [47].

A thesis, which is at the root of our research effort, and which was already formulated by the team of the Logical Framework [38] is that proof-checkers should be theory independent. This is for instance expressed in the title of our invited talk at Icalp 2012: *A theory independent Curry-De Bruijn-Howard correspondence*.

Using a single prover to check proofs coming from different provers naturally led to investigate how these proofs could interact one with another. This issue is of prime importance because developments in proof systems are getting bigger and, unlike other communities in computer science, the proof-checking community has given little effort in the direction of standardization and interoperability. On a longer term we believe that, for each proof, we should be able to identify the systems in which it can be expressed.

3.2. Automated theorem proving

Deduction modulo has originally been proposed to solve a problem in automated theorem proving and some of the early work in this area focused on the design of an automated theorem proving method called *Resolution modulo*, but this method was so complex that it was never implemented. This method was simplified in 2010 [5] and it could then be implemented. This implementation that builds on the iProver effort [46] is called iProver modulo.

iProver modulo gave surprisingly good results [4], so that we use it now to search for proofs in many areas: in the theory of classes—also known as *B* set theory—, on finite structures, etc. Similar ideas have also been implemented for the tableau method with in particular several extensions of the *Zenon* automated theorem prover. More precisely, two extensions have been realized: the first one is called *Super Zenon* [13] [35] and is an extension to superdeduction (which is a variant of Deduction modulo), and the second one is called *Zenon Modulo* [33], [34] and is an extension to Deduction modulo. Both extensions have been extensively tested over first order problems (of the TPTP library), and also provide good results in terms of number of proved problems. In particular, these tools provide good performances in set theory, so that *Super Zenon* has been successfully applied to verify *B* proof rules of *Atelier B* (work in collaboration with *Siemens*). Similarly, we plan to apply *Zenon Modulo* in the framework of the *BWare* project to verify *B* proof obligations coming from the modeling of industrial applications.

More generally, we believe that proof-checking and automated theorem proving have a lot to learn from each other, because a proof is both a static linguistic object justifying the truth of a proposition and a dynamic process of proving this proposition.

3.3. Models of computation

The idea of Deduction modulo is that computation plays a major role in the foundations of mathematics. This led us to investigate the role played by computation in other sciences, in particular in physics. Some of this work can be seen as a continuation of Gandy's [36] on the fact that the physical Church-Turing thesis is a consequence of three principles of physics, two well-known: the homogeneity of space and time, and the existence of a bound on the velocity of information, and one more speculative: the existence of a bound on the density of information.

This led us to develop physically oriented models of computations.

4. Application Domains

4.1. Safety of aerospace systems

In parallel with this effort in logic and in the development of proof checkers and automated theorem proving systems, we always have been interested in using such tools. One of our favorite application domain is the safety of aerospace systems. Together with César Muñoz' team in Nasa-Langley, we have proved the correctness of several geometric algorithms used in air traffic control.

This has led us sometimes to develop such algorithms ourselves, and sometimes to develop tools for automating these proofs.

4.2. B-set theory

Set theory appears to be an appropriate theory for automated theorem provers based on Deduction modulo, in particular the several extensions of *Zenon* (*Super Zenon* and *Zenon Modulo*). Modeling techniques using set theory are therefore good candidates to assess these tools. This is what we have done with the *B* method whose formalism relies on set theory. A collaboration with *Siemens* has been developed to automatically verify the *B* proof rules of *Atelier B* [42]. From this work presented in the Doctoral dissertation of Mélanie Jacquel, the *Super Zenon* tool [13] [35] has been designed in order to be able to reason modulo the *B* set theory. As a sequel of this work, we contribute to the *BWare* project whose aim is to provide a mechanized framework to support the automated verification of *B* proof obligations coming from the development of industrial applications. In this context, we have recently designed *Zenon Modulo* [33], [34] (Pierre Halmagrand's PhD thesis, which has started on October 2013) to deal with the *B* set theory. In this work, the idea is to manually transform the *B* set theory into a theory modulo and provide it to *Zenon Modulo* in order to verify the proof obligations of the *BWare* project.

4.3. Termination certificate verification

Termination is an important property to verify, especially in critical applications. Automated termination provers use more and more complex theoretical results and external tools (e.g. sophisticated SAT solvers) that make their results not fully trustable and very difficult to check. To overcome this problem, a language for termination certificates, called CPF, has been developed since several years now. Deducteam develops a formally certified tool, Rainbow, based on the Coq library CoLoR, that is able to automatically verify the correctness of such termination certificates.

5. New Software and Platforms

5.1. Dedukti

Dedukti is a proof-checker for the $\lambda\Pi$-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 [29] for type-checking semi-full pure type systems and implements a state-of-the-art reduction machine inspired from Matita's [28] 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 its project in different files and compile them separately.

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

Dedukti has been developed by Mathieu Boespflug, Olivier Hermant, Quentin Carbonneaux and Ronan Saillard. It is composed of about 2500 lines of OCaml.

5.2. Coqine, Holide, Focalide and Sigmaid

Dedukti comes with four companion tools: Holide, an embedding of HOL proofs through the OpenTheory format [41], Coqine, an embedding of Coq proofs, Focalide, an embedding of FoCaLiZe certified programs, and Sigmaid, a type-checker for the simply-typed ς-calculus with subtyping and a translator to Dedukti. All of the OpenTheory standard library and a part of Coq's and FoCaLiZe's libraries are checked by Dedukti.

A preliminary version of Coqine supports the following features of Coq: the raw Calculus of Constructions, inductive types, and fixpoint definitions. Coqine is currently being rewritten to support universes. Coqine has been developed by Mathieu Boespflug, Guillaume Burel, and Ali Assaf.

Holide supports all the features of HOL, including ML-polymorphism, constant definitions, and type definitions. It is able to translate all of the OpenTheory standard theory library. Holide has been developed by Ali Assaf.

Focalide has been improved to support FoCaLiZe proofs found by Zenon using the Dedukti backend for Zenon developped by Frédéric Gilbert. This backend has been improved by a simple typing mechanism in order to work with Focalide. Focalide has also been updated again to work with the latest version of FoCaLiZe.

Sigmaid implements a shallow embedding of the simply-typed ς-calculus of Abadi and Cardelli with subtyping in the $\lambda\Pi$-calculus modulo. This translation has been proved[21] to preserve the typing judgments and the semantics of the simply-typed ς-calculus and tested on the examples of Abadi and Cardelli.

Focalide and Sigmaid have been developed by Raphaël Cauderlier.

Translators from Version 2.0 of the SMT-LIB standard and from the SMT-solver veriT have been initiated. They are currently developed by Frédéric Gilbert.

5.3. iProver Modulo

iProver Modulo is an extension of the automated theorem prover iProver originally developed by Konstantin Korovin at the University of Manchester. It implements Ordered polarized resolution modulo, a refinement of the Resolution method based on Deduction modulo. It takes as input a proposition in predicate logic and a clausal rewriting system defining the theory in which the formula has to be proved. Normalization with respect to the term rewriting rules is performed very efficiently through translation into OCaml code, compilation and dynamic linking. Experiments have shown that Ordered polarized resolution modulo dramatically improves proof search compared to using raw axioms. iProver modulo is also able to produce proofs that can be checked by Dedukti, therefore improving confidence. iProver modulo is written in OCaml, it consists of 1,200 lines of code added to the original iProver.

A tool that transforms axiomatic theories into polarized rewriting systems, thus making them usable in iProver Modulo, has also been developed. Autotheo supports several strategies to orient the axioms, some of them being proved to be complete, in the sense that Ordered polarized resolution modulo the resulting systems is refutationally complete, some others being merely heuristics. In practice, autotheo takes a TPTP input file and transforms the axioms into rewriting rules, and produces an input file for iProver Modulo.

iProver Modulo and autotheo have been developed by Guillaume Burel. iProver Modulo is released under a GPL license.

5.4. Super Zenon and Zenon Modulo

Several extensions of the *Zenon* automated theorem prover (developed by Damien Doligez at *Inria* in the *Gallium* team) to Deduction modulo have been studied. These extensions intend to be applied in the context of the automatic verification of proof rules and obligations coming from industrial applications formalized using the *B* method.

The first extension, developed by Mélanie Jacquel and David Delahaye, is called *Super Zenon* and is an extension of *Zenon* to superdeduction, which can be seen as a variant of Deduction modulo. This extension is a generalization of previous experiments [42] together with Catherine Dubois and Karim Berkani (*Siemens*), where *Zenon* has been used and extended to superdeduction to deal with the *B* set theory and automatically prove proof rules of *Atelier B*. This generalization consists in allowing us to apply the extension of *Zenon* to superdeduction to any first order theory by means of a heuristic that automatically transforms axioms of the theory into rewrite rules. This work is described in [13] [35], which also proposes a study of the possibility of recovering intuition from automated proofs using superdeduction.

The second extension, developed by Pierre Halmagrand, David Delahaye, Damien Doligez, and Olivier Hermant, is called *Zenon Modulo* and is an extension of *Zenon* to Deduction modulo. Compared to *Super Zenon*, this extension allows us to 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. To assess the approach of *Zenon Modulo*, we have applied this extension to the first order problems coming from the TPTP library. An increase of the number of proved problems has been observed, with in particular a significant increase in the category of set theory. Over these problems of the TPTP library, we have also observed a significant proof size reduction, which confirms this aspect of Deduction modulo. These results are gathered into two publications [33], [34].

The third extension, developed by Guillaume Bury and David Delahaye, is an extension of *Zenon* to (rational and integer) linear arithmetic (using the simplex algorithm), that has been integrated to *Zenon Modulo* by Guillaume Bury and Pierre Halmagrand, in order to be applied in the framework of the *B* set theory to the verification of proof obligations of *Atelier B* [17]. Experiments have been conducted over the benchmarks of the *BWare* project, and it turns out that more than 95% of the proof obligations are proved thanks to this extension.

5.5. Zipperposition (and extensions) and Logtk

Zipperposition is an implementation of the superposition method; it relies on the library Logtk for basic logic data structures and algorithms. Zipperposition is designed as a testbed for extensions to superposition, and can currently deal with polymorphic typed logic, integer arithmetic, and total orderings; an extension to handle structural induction is being worked on by Simon Cruanes.

Those pieces of software also depend on many smaller tools and libraries developped by Simon Cruanes in OCaml. In particular, efficient iterators were key to implementing arithmetic rules successfully, and a lightweight extension to the standard library has been developped steadily and released regularly.

5.6. CoLoR

CoLoR is a Coq library on rewriting theory and termination of more than 83,000 lines of code [2]. It provides definitions and theorems for:

- Mathematical structures: relations, (ordered) semi-rings.
- Data structures: lists, vectors, polynomials with multiple variables, finite multisets, matrices, finite graphs.
- Term structures: strings, algebraic terms with symbols of fixed arity, algebraic terms with varyadic symbols, pure and simply typed λ-terms.
- Transformation techniques: conversion from strings to algebraic terms, conversion from algebraic to varyadic terms, arguments filtering, rule elimination, dependency pairs, dependency graph decomposition, semantic labelling.
- Termination criteria: polynomial interpretations, multiset ordering, lexicographic ordering, first and higher order recursive path ordering, matrix interpretations.

CoLoR is distributed under the CeCILL license. It is currently developed by Frédéric Blanqui and Kim-Quyen Ly, but various people participated to its development since 2006 (see the website for more information).

5.7. HOT

HOT is an automated termination prover for higher-order rewrite systems based on the notion of computability closure and size annotation [31]. It won the 2012 competition in the category “higher-order rewriting union beta”. The sources (5000 lines of OCaml) are not public. It is developed by Frédéric Blanqui.

5.8. Moca

Moca is a construction functions generator for OCaml data types with invariants.

It allows the high-level definition and automatic management of complex invariants for data types. In addition, it provides the automatic generation of maximally shared values, independently or in conjunction with the declared invariants.

A relational data type is a concrete data type that declares invariants or relations that are verified by its constructors. For each relational data type definition, Moca compiles a set of construction functions that implements the declared relations.

Moca supports two kinds of relations:

- predefined algebraic relations (such as associativity or commutativity of a binary constructor),
- user-defined rewrite rules that map some pattern of constructors and variables to some arbitrary user's define expression.

The properties that user-defined rules should satisfy (completeness, termination, and confluence of the resulting term rewriting system) must be verified by a programmer's proof before compilation. For the predefined relations, Moca generates construction functions that allow each equivalence class to be uniquely represented by their canonical value.

Moca is distributed under QPL. It is written in OCaml (14,000 lines) It is developed by Frédéric Blanqui, Pierre Weis (EPI Pomdapi) and Richard Bonichon (CEA).

5.9. Rainbow

Rainbow is a set of tools for automatically verifying the correctness of termination certificates expressed in the CPF format used in the termination competition. It contains:

- a tool `xsd2coq` for generating Coq data types for representing XML files valid wrt some XML Schema,
- a tool `xsd2ml` for generating OCaml data types and functions for parsing XML files valid wrt some XML Schema,
- a tool for translating a CPF file into a Coq script,
- a standalone Coq certified tool for verifying the correctness of a CPF file.

Rainbow is distributed under the CeCILL license. It is developed in OCaml (10,000 lines) and Coq (19,000 lines). It is currently developed by Frédéric Blanqui and Kim-Quyen Ly. See the website for more information.

5.10. mSAT

mSAT is a modular, proof-producing, SAT and SMT core based on Alt-Ergo Zero, written in OCaml. The solver accepts user-defined terms, formulas and theory, making it a good tool for experimenting. This tool produces resolution proofs as trees in which the leaves are user-defined proof of lemmas.

An encoding of tableaux rules as a theory for SMT solvers has been implemented and tested in mSAT. mSat has also been extended to implement model constructing satisfiability calculus, a variant of SMT solvers in which assignment of variables to values are propagated along with the usual boolean assignment of litterals.

6. New Results

6.1. Highlights of the Year

In the framework of the *BWare* project, Pierre Halmagrand, David Delahaye, Damien Doligez, and Olivier Hermant designed a new version of the *B* set theory using deduction modulo, in order to automatically verify a large part of the proof obligations of the benchmark of *BWare*, which consists of proof obligations coming from the modeling of industrial applications (about 13,000 proof obligations). Using this *B* set theory modulo with *Zenon Modulo*, as well as some other extensions of *Zenon*, such as typed proof search and arithmetic (implemented by Guillaume Bury), we are able to automatically verify more than 95% of the proof obligations of *BWare*, while the regular version of *Zenon* is only able to prove less than 1% of these proof obligations. This is a real breakthrough for the *BWare* project, but also for automated deduction in general, as it tends to show that deduction modulo is the way to go when reasoning modulo theories.

6.2. Termination

Frédéric Blanqui, together with Jean-Pierre Jouannaud (Univ. Paris 11) and Albert Rubio (Technical University of Catalonia), have finished their work on a new version of the higher-order recursive path ordering (HORPO) [44], [43], a decidable monotone well-founded relation that can be used for proving the termination of higher-order rewrite systems by checking that rules are included in it. This new version, called the computability path ordering (CPO), appears to be the ultimate improvement of HORPO in the sense that this definition captures the essence of computability arguments *à la* Tait and Girard [37], therefore explaining the name of the improved ordering. It has been shown that CPO allows to consider higher-order rewrite rules in a simple type discipline with inductive types, that most of the guards present in the recursive calls of its core definition cannot be relaxed in any natural way without losing well-foundedness, and that the precedence on function symbols cannot be made more liberal anymore. This new result is described in a 41-pages papers available on Frédéric Blanqui's web page which has been submitted to a journal for publication. A Prolog implementation of CPO is also available on Albert Rubio's web page.

Frédéric Blanqui revised his work on the compatibility of Tait and Girard's notion of computability for proving the termination of higher-order rewrite systems when matching is done modulo $\beta\eta$-equivalence. In particular, he showed that computability is preserved by leaf-β-expansion, a key property for dealing with higher-order pattern-matching. This work is described in a 46-pages paper available on his web page which has been submitted to a journal for publication.

Frédéric Blanqui did some historical investigations on fixpoint theorems in posets used for instance for defining the semantics of non-basic inductive types (i.e. types with constructors taking functions as arguments) and the termination of functions defined by induction on such non-basic inductive types. These theorems assume the function either extensive or monotone. However, as shown by Salinas in [48], these two conditions can be subsumed by a more general one. Frédéric Blanqui slightly improved this condition further by using results by Hartogs, Rubin and Rubin, and Abian and Brown. This work is described in a 10-pages note available on his web page [20].

Kim Quyen Ly finished the development of a new version faster, safer (proved correct in Coq) and standalone version of Rainbow, based on Coq extraction mechanism. She defended her PhD thesis [11] on the automated verification of termination certificates in October.

6.3. Proof and type theory modulo rewriting

Ali Assaf defined a sound and complete embedding of the cumulative universe hierarchy of the *calculus of inductive constructions* (CIC) in the $\lambda\Pi$-calculus modulo rewriting [18]. By reformulating universes in the Tarski style, he showed that we can make cumulativity explicit without losing any typing power. This result refines the translation used by Coqine, which was unsound because it collapsed the universe hierarchy to a single type universe. It also sheds some light on the metatheory of Coq and its connection to Martin-Löf's intuitionistic type theory. This work was presented at the TYPES meeting in Paris.

Frédéric Gilbert and Olivier Hermant defined new encodings from classical to intuitionistic first-order logic. These encodings, based on the introduction of double negations in formulas, are tuned to satisfy two purposes jointly: basing their specifications on the definition of *classical connectives* inside intuitionistic logic – which is the property of *morphisms*, and reducing their impact on the shape and size of formulas, by limiting as much as possible the number of negations introduced. This paper has been submitted.

Raphael Cauderlier and Catherine Dubois defined a shallow embedding of an object calculus (formalized by Abadi and Cardelli), in the $\lambda\Pi$-calculus modulo rewriting . The main result concerns the encoding of subtyping. This encoding shows that rewriting is an effective help for handling of subtyping proofs. The implementation in Dedukti, Sigmaid. This work has been presented at the TYPES 2014 meeting in Paris. A paper has been submitted.

Ali Assaf, Olivier Hermant and Ronan Saillard defined a rewrite system such that all strongly normalizable proof term can be typed in Natural Deduction modulo this rewrite system. This work is inspired by Statman's work [49], and can be understood as an encoding of intersection types.

Guillaume Burel showed how to get rewriting systems that admit cut by using standard saturation techniques from automated theorem proving, namely ordered resolution with selection, and superposition. This work relies on a view of proposition rewriting rules as oriented clauses, like term rewriting rules can be seen as oriented equations. This also lead to introduce an extension of deduction modulo with *conditional* term rewriting rules. This work was presented at the RTA-TLCA conference in Vienna [15].

Gilles Dowek, has generalized the notion of super-consistency to the lambda-Pi-calculus modulo theory and proved this way the termination of the embedding of various formulations of Simple Type Theory and of the Calculus of Constructions in the Lambda-Pi calculs modulo theory.

Gilles Dowek and Alejandro Díaz-Caro have finished their work on the extension of Simply Typed Lambda-Calculus with Type Isomorphisms. This work has been presented at the Types meeting and recently accepted for publication in the Theoretical Computer Science journal [26].

Gilles Dowek and Ying Jiang have given a new proof of the decidability of reachability in alternating pushdown systems, based on a cut-elimination theorem.

Vaston Costa presented to the group a new structure to represent proofs through references rather than copy. The structure, called Mimp-graph, was initially developed for minimal propositional logic but the results have been extended to first-order logic. Mimp-graph preserves the ability to represent any Natural Deduction proof and its minimal formula representation is a key feature of the mimp-graph structure, it is easy to distinguish maximal formulas and an upper bound in the length of the reduction sequence to obtain a normal proof. Thus a normalization theorem can be proved by counting the number of maximal formulas in the original derivation. The strong normalization follow as a direct consequence of such normalization, since that any reduction decreases the corresponding measures of derivation complexity. Sharing for inference rules is performed during the process of construction of the graph. This feature is very important, since we intend to use this graph in automatic theorem provers.

6.4. Automated theorem proving

Guillaume Bury defined a sound and complete extension of the tableaux method to handle linear arithmetic. The rules are based on a variant of the simplex algorithm for rational and real linear arithmetic, and a Branch&Bound algorithm for integer arithmetic.

Guillaume Bury defined an encoding of analytical tableaux rules as a theory for smt solvers. The theory acts like a lazy cnf conversion during the proof search and allows to integrate the cnf conversion into the resolution proof for unsatisfiable formulas. This work was implemented in mSAT.

Simon Cruanes added many improvements to Logtk, in particular a better algorithm to reduce formulas to Clausal Normal Form. A presentation of its design and implementation has been made at PAAR 2014[16]. He also used Zipperposition as a testbed for integer linear arithmetic; a sophisticated inference system for this fragment of arithmetic was designed and implemented in Zipperposition, including many redundancy criteria and simplification rules that make it efficient in practice. The arithmetic-enabled Zipperposition version entered CASC-J7, the annual competition of Automated Theorem Provers, in the first-order theorems with linear arithmetic division where it had very promising results (on integer problems only, since Zipperposition doesn't handle rationals).

Another extension of Zipperposition has been performed by Julien Rateau, Simon Cruanes, and David Delahaye, in order to deal with a fragment of set theory in the same vein as the $STR{+}VE{\subseteq}$ prover [40]. This extension relies on a specific normal form of literal, which only involves the ⊆, ∩, ∪, and complement set operators. In the future, the idea is to use this extension in the framework of the *BWare* project to verify *B* proof obligations coming from industrial benchmarks.

The current effort of research on Zipperposition focuses on extending superposition to handle structural induction, following the work from [45]. The current prototype is able to prove simple properties on natural numbers, binary trees and lists.

Kailiang Ji defined a set of rewrite rules for the equivalence between CTL formulas (denote them as R_{CTL}), by taking them as terms of designed predicates. For a given transition system model, we transform it into a set of rewrite rules (denote them as R_m). Then any CTL property of the transition system can be proved in deduction modulo $R_{CTL} \cup R_m$, by specifying the model checking problems into designed first-order formulas. This method was implemented in iProver Modulo, and the experimental evaluation was reported in workshop of Locali 2014.

6.5. Algebraic λ-calculus

Ali Assaf, Alejandro Díaz-Caro, Simon Perdrix, Christine Tasson, and Benoit Valiron completed a journal paper covering results on different algebraic extensions of the λ-calculus [12]. These extensions equip the calculus with an additive and a scalar-multiplicative structure, and their set of terms is closed under linear combinations. Two such extensions, the *algebraic λ-calculus* and the *linear-algebraic λ-calculus* arise independently in different contexts – the former is a fragment of the differential λ-calculus, the latter is a candidate λ-calculus for quantum computation – and have different operational semantics. In this paper, the authors showed how the two approaches relate to each other. They showed that the the first calculus follows a call-by-name strategy while the second follows a call-by-value strategy. They proved that the two can simulate each other using algebraic extensions of *continuation passing style* (CPS) translations that are sound and complete.

7. Partnerships and Cooperations

7.1. National Initiatives

7.1.1. ANR Locali

We are coordinators of the ANR-NFSC contract Locali with the Chinese Academy of Sciences.

7.1.2. ANR BWare

We are members of the ANR *BWare*, which started on September 2012 (David Delahaye is the national leader of this project). The aim of this project is to provide a mechanized framework to support the automated verification of proof obligations coming from the development of industrial applications using the *B* method. The methodology used in this project consists in building a generic platform of verification relying on different theorem provers, such as first order provers and SMT solvers. We are in particular involved in the introduction of Deduction modulo in the first order theorem provers of the project, i.e. *Zenon* and *iProver*, as well as in the backend for these provers with the use of *Dedukti*.

The ANR mid-term review of the project took place in October 2014 and the members of the project received very positive feedbacks from the reviewers. A more detailed report is expected from the reviewers in early 2015.

7.1.3. ANR Tarmac

We are members of the ANR Tarmac on models of computation, coordinated by Pierre Valarcher.

7.2. International Research Visitors

7.2.1. Visits to International Teams

7.2.1.1. Research stays abroad

Olivier Hermant was an invited researcher at the Natal University (UFRN, Brazil) in December 2014.

8. Dissemination

8.1. Promoting Scientific Activities

8.1.1. Scientific events organisation

8.1.1.1. General chair, scientific chair

Catherine Dubois was a general chair of the GDR GPL national days, and the AFADL'14, CAL'14, and CIEL'14 conferences, organized at *Cnam* in Paris.

8.1.1.2. Member of the organizing committee

David Delahaye was a member of the organizing committee of the GDR GPL national days, and the AFADL'14, CAL'14, and CIEL'14 conferences, organized at *Cnam* in Paris.

Olivier Hermant and Florian Rabe organized the 2nd KWARC-Deducteam workshop in Bremen, Germany (May 2014).

Gilles Dowek is a member of the steering committees of RTA and TLCA. Catherine Dubois is the chair of the steering committee of TAP.

8.1.2. Scientific events selection

8.1.2.1. Chair of conference program committee

Gilles Dowek has been the PC chair of RTA-TLCA.

David Delahaye and Catherine Dubois were co-chairs of the SETS'14 workshop (affiliated to ABZ'14).

David Delahaye was a co-chair of the $\forall X.X\Pi$'14 workshop (affiliated to VSL'14).

Catherine Dubois was a co-chair of the F-IDE'14 workshop (affiliated to ETAPS'14). She was also a PC co-chair of the AFADL'2014 national conference.

8.1.2.2. Member of the conference program committee

Olivier Hermant was member of the program committee of the RTA-TLCA 2014 conference (July 2014).

8.1.2.3. Reviewer

Frédéric Blanqui reviewed a paper submitted to the post-proceedings of TYPES'14.

Alejandro Díaz-Caro reviewed a paper submitted to the FSTTCS'14 conference.

Ronan Saillard reviewed a paper submitted to the RTA'14 conference.

8.1.3. Journal

8.1.3.1. Member of the editorial board

David Delahaye is a member of the editorial board of the Global Journal of Advanced Software Engineering (GJASE).

8.1.3.2. Reviewer

David Delahaye reviewed one paper for the Global Journal of Advanced Software Engineering (GJASE) and one paper for the Requirements Engineering (RE) journal.

Alejandro Díaz-Caro reviewed a paper submitted to Elsevier's "Science of Computer Programming" journal.

Olivier Hermant reviewed a paper submitted to the journal "Theoretical Computer Science".

8.2. Teaching - Supervision - Juries

8.2.1. Teaching

CPGE: Simon Cruanes, Mathématiques-Informatiques en MPSI, 64 HETD, MPSI, Lycée Saint-Louis, France.

Licence : Raphaël Cauderlier, Eléments de programmation 2, 63 HETD, L1, UPMC, France.

License: Ali Assaf, Algorithmique et programmation, 36 HETD, L3, Ecole Polytechnique, France.

License: Ali Assaf, Les bases de la programmation et de l'algorithmique, 36 HETD, L3, Ecole Polytechnique, France.

Licence : Alejandro Díaz-Caro, Mathématiques 2, 18 HETD, L1, Université Paris-Ouest Nanterre La Défense, France.

Licence : Alejandro Díaz-Caro, Méthodologie de la mesure en sciences humaines, 48 HETD, L1, Université Paris-Ouest Nanterre La Défense, France.

Licence : Alejandro Díaz-Caro, Statistiques et probabilités, 18 HETD, L2, Université Paris-Ouest Nanterre La Défense, France.

Licence : Alejandro Díaz-Caro, Probabilités, 36 HETD, L2, Université Paris-Ouest Nanterre La Défense, France.

Licence : Alejandro Díaz-Caro, Probabilités, 36 HETD, L2, Université Paris-Ouest Nanterre La Défense, France.

Licence: Ronan Saillard, Programmation Orientée Objet en Java, 27 HETD, Telecom ParisTech, France.

Licence: Guillaume Burel, Programmation avancée, 25.5 HETD, L3, ENSIIE, France

Licence: Guillaume Burel, Logique, 10.5 HETD, L3, ENSIIE, France

Licence: Guillaume Burel, Projet informatique, 22.75 HETD, L3, ENSIIE, France

Master: Guillaume Burel, Systèmes et langages formels, 17.5 HETD, M1, ENSIIE, France

Master: Guillaume Burel, Compilation, 33.25 HETD, M1, ENSIIE, France

Licence: Guillaume Burel is in charge of the 4th, 5th, and 6th semesters of the engineering degree at ENSIIE.

Licence: Pierre Halmagrand, Initiation à la Programmation en JAVA, 48.75 HETD, L1, CNAM, France

Master: Pierre Halmagrand, Initiation à la Méthode B, 53.7 HETD, M2, CNAM, France

Master: Gilles Dowek has given invited classes at Supelec, Centrale, INSA-Lyon.

8.2.2. Supervision

PhD : Kim-Quyen Ly, Automated verification of termination certificates [11], University Joseph Fourier, Grenoble, 9 October 2014, Jean-François Monin and Frédéric Blanqui.

PhD in progress : Ronan Saillard, Systèmes de Types Modulo, Oct 2012, Olivier Hermant and Pierre Jouvelot.

PhD in progress : Pierre Halmagrand, "Automated Deduction Modulo", from Oct. 2013, David Delahaye, Olivier Hermant, and Damien Doligez.

PhD in progress : Guillaume Bury, "SMT Modulo", Oct. 2014, David Delahaye.

PhD in progress : Ali Assaf, Interoperability of Proof Systems, Oct. 2012, Gilles Dowek and Guillaume Burel.

PhD in progress : Simon Cruanes, Automated reasoning modulo theories, August 2012, Gilles Dowek and Guillaume Burel.

PhD in progress : Kailiang Ji, Model Checking and Automated Theorem Proving, October 2012, Gilles Dowek.

8.2.3. Juries

David Delahaye reviewed the PhD thesis of Hernán Vanzetto, supervised by Stephan Merz and Kaustuv Chaudhuri, and defended in Dec. 2014 in Nancy.

David Delahaye was part of the HDR jury of Stéphane Lengrand, defended in Dec. 2014 at École Polytechnique in Palaiseau.

Gilles Dowek has been part of the HDR committee of Philippe Malbos.

Catherine Dubois reviewed the HDR thesis of Oum-El-Kheir Aktouf, defended in July 2014 in Valence.

8.3. Popularization

Gilles Dowek is the president of the scientific board of the “Société Informatique de France” (SIF).

Gilles Dowek is a member of the scientific board of “La Main à la Pâte”.

Raphaël Cauderlier, Simon Cruanes, Pierre Halmagrand et Ronan Saillard ont participé à l’animation du stand Inria au Salon Culture et Jeux Mathématiques les 24 et 25 mai

Ali Asaf, Raphaël Cauderlier, Simon Cruanes et Pierre Halmagrand ont participé à l’animation du stand Inria lors de la Fête de la Science les 9 et 10 octobre 2014 à Paris

Gilles Dowek has given a talk at Mathenjeans.

Alejandro Díaz-Caro is member of the scientific board of “Ensemble”, a journal of the Maison Argentina at Cité Universitaire in Paris, ISSN 1852–5911.

9. Bibliography

Major publications by the team in recent years

[1] F. BLANQUI. *Definitions by rewriting in the Calculus of Constructions*, in "Mathematical Structures in Computer Science", 2005, vol. 15, n$^{\mathrm{O}}$ 1, pp. 37-92 [*DOI :* 10.1017/S0960129504004426], http://hal.inria.fr/inria-00105648/en/

[2] F. BLANQUI, A. KOPROWSKI. *CoLoR: a Coq library on well-founded rewrite relations and its application to the automated verification of termination certificates*, in "Mathematical Structures in Computer Science", 2011, vol. 21, n$^{\mathrm{O}}$ 4, pp. 827-859, http://hal.inria.fr/inria-00543157/en/

[3] M. BOESPFLUG. *Conception d’un noyau de vérification de preuves pour le lambda-Pi-calcul modulo*, École Polytechnique, 2011

[4] G. BUREL. *Experimenting with Deduction Modulo*, in "CADE 2011", V. SOFRONIE-STOKKERMANS, N. BJØRNER (editors), Lecture Notes in Artificial Intelligence, Springer, 2011, vol. 6803, pp. 162–176

[5] G. DOWEK. *Polarized Resolution Modulo*, in "IFIP Theoretical Computer Science", 2010

[6] G. DOWEK, T. HARDIN, C. KIRCHNER. *Theorem proving modulo*, in "Journal of Automated Reasoning", 2003, vol. 31, pp. 33-73

[7] C. DUBOIS, T. HARDIN, V. DONZEAU-GOUGE. *Building certified components within FOCAL*, in "Revised Selected Papers from the Fifth Symposium on Trends in Functional Programming, TFP 2004, München, Germany, 25-26 November 2004", H.-W. LOIDL (editor), Trends in Functional Programming, Intellect, 2006, vol. 5, pp. 33-48

[8] O. HERMANT. *Resolution is Cut-Free*, in "Journal of Automated Reasoning", March 2010, vol. 44, n$^{\mathrm{O}}$ 3, pp. 245-276

[9] M. JACQUEL, K. BERKANI, D. DELAHAYE, C. DUBOIS. *Verifying B Proof Rules using Deep Embedding and Automated Theorem Proving*, in "Software and Systems Modeling (SoSyM)", June 2013

[10] M. JACQUEL, K. BERKANI, D. DELAHAYE, C. DUBOIS. *Tableaux Modulo Theories Using Superdeduction*, in "Global Journal of Advanced Software Engineering (GJASE)", December 2014, vol. 1, pp. 1 - 13 [*DOI :* 10.1007/978-3-642-31365-3_26], https://hal.archives-ouvertes.fr/hal-01099338

Publications of the year

Doctoral Dissertations and Habilitation Theses

[11] K. Q. LY. *Automated verification of termination certificates*, Université Joseph Fourier, October 2014, https://hal.inria.fr/tel-01097793

Articles in International Peer-Reviewed Journals

[12] A. ASSAF, A. DÍAZ-CARO, S. PERDRIX, C. TASSON, B. VALIRON. *Call-by-value, call-by-name and the vectorial behaviour of the algebraic λ-calculus*, in "Logical Methods in Computer Science", December 2014, vol. 10:4, n^{o} 8, 40 p. [*DOI :* 10.2168/LMCS-10(4:8)2014], https://hal.inria.fr/hal-01097602

[13] M. JACQUEL, K. BERKANI, D. DELAHAYE, C. DUBOIS. *Tableaux Modulo Theories Using Superdeduction*, in "Global Journal of Advanced Software Engineering (GJASE)", December 2014, vol. 1, pp. 1 - 13 [*DOI :* 10.1007/978-3-642-31365-3_26], https://hal.archives-ouvertes.fr/hal-01099338

Invited Conferences

[14] G. DOWEK. *Deduction modulo theory*, in "All about proofs. Proofs for all.", Wien, Austria, July 2014, https://hal.inria.fr/hal-01101829

International Conferences with Proceedings

[15] G. BUREL. *Cut Admissibility by Saturation*, in "Joint International Conference, RTA-TLCA", Vienna, Austria, G. DOWEK (editor), Lecture Notes in Computer Science, Springer, July 2014, vol. 8560, pp. 124-138 [*DOI :* 10.1007/978-3-319-08918-8_9], https://hal.inria.fr/hal-01097428

[16] S. CRUANES. *Logtk : A Logic ToolKit for Automated Reasoning and its Implementation*, in "4th Workshop on Practical Aspects of Automated Reasoning", Vienna, Austria, July 2014, https://hal.inria.fr/hal-01101057

Conferences without Proceedings

[17] P. HALMAGRAND. *Using Deduction Modulo in Set Theory*, in "SETS14, 1st International Workshop about Sets and Tools", Toulouse, France, June 2014, 12 p. , https://hal.inria.fr/hal-01100512

Other Publications

[18] A. ASSAF. *A calculus of constructions with explicit subtyping*, 2014, https://hal.archives-ouvertes.fr/hal-01097401

[19] A. ASSAF, G. BUREL. *Translating HOL to Dedukti*, 2014, https://hal.archives-ouvertes.fr/hal-01097412

[20] F. BLANQUI. *A point on fixpoints in posets*, January 2014, https://hal.inria.fr/hal-01097809

[21] R. CAUDERLIER, C. DUBOIS. *Objects and subtyping in the $\lambda\Pi$-calculus modulo*, December 2014, https://hal.inria.fr/hal-01097444

[22] G. Dowek. *Models and termination of proof-reduction in the $\lambda\Pi$-calculus modulo theory*, January 2015, https://hal.inria.fr/hal-01101834

[23] G. Dowek, Y. Jiang. *A Logical Approach to CTL*, January 2014, https://hal.inria.fr/hal-00919467

[24] G. Dowek, Y. Jiang. *Axiomatizing truth in a finite model*, January 2014, https://hal.inria.fr/hal-00919469

[25] G. Dowek, Y. Jiang. *Cut-elimination and the decidability of reachability in alternating pushdown systems*, January 2015, https://hal.inria.fr/hal-01101835

[26] A. Díaz-Caro, G. Dowek. *Simply Typed Lambda-Calculus Modulo Type Isomorphisms*, August 2014, https://hal.inria.fr/hal-01109104

[27] R. Saillard. *Dedukti : un vérificateur de preuves universel*, June 2014, Journées de seconde année de l'Ecole Doctorale à Mines ParisTech, https://hal-mines-paristech.archives-ouvertes.fr/hal-01100519

References in notes

[28] A. Asperti, W. Ricciotti, C. Sacerdoti Coen, E. Tassi. *A compact kernel for the calculus of inductive constructions*, in "Sadhana", 2009, vol. 34, n^{o} 1, pp. 71-144, http://dx.doi.org/10.1007/s12046-009-0003-3

[29] L. Benthem Jutting, J. McKinna, R. Pollack. *Checking algorithms for Pure Type Systems*, in "Types for Proofs and Programs", H. Barendregt, T. Nipkow (editors), Lecture Notes in Computer Science, Springer Berlin Heidelberg, 1994, vol. 806, pp. 19-61

[30] Y. Bertot, P. Castéran. *Interactive Theorem Proving and Program Development Coq'Art: The Calculus of Inductive Constructions*, Springer-Verlag, 2004

[31] F. Blanqui. *Terminaison des systèmes de réécriture d'ordre supérieur basée sur la notion de clôture de calculabilité*, Université Paris-Diderot - Paris VII, July 2012, HDR, http://tel.archives-ouvertes.fr/tel-00724233

[32] D. Cousineau, G. Dowek. *Embedding Pure Type Systems in the lambda-Pi-calculus modulo*, in "Typed lambda calculi and applications", S. Ronchi della Rocca (editor), Lecture Notes in Computer Science, Springer-Verlag, 2007, vol. 4583, pp. 102-117

[33] D. Delahaye, D. Doligez, F. Gilbert, P. Halmagrand, O. Hermant. *Proof Certification in Zenon Modulo: When Achilles Uses Deduction Modulo to Outrun the Tortoise with Shorter Steps*, in "IWIL - 10th International Workshop on the Implementation of Logics - 2013", Stellenbosch, South Africa, S. Schulz, G. Sutcliffe, B. Konev (editors), EasyChair, December 2013, https://hal.inria.fr/hal-00909688

[34] D. Delahaye, D. Doligez, F. Gilbert, P. Halmagrand, O. Hermant. *Zenon Modulo: When Achilles Outruns the Tortoise using Deduction Modulo*, in "LPAR - Logic for Programming Artificial Intelligence and Reasoning - 2013", Stellenbosch, South Africa, K. McMillan, A. Middeldorp, A. Voronkov (editors), Springer, December 2013, vol. 8312, pp. 274-290 [*DOI :* 10.1007/978-3-642-45221-5_20], https://hal.inria.fr/hal-00909784

[35] D. Delahaye, M. Jacquel. *Recovering Intuition from Automated Formal Proofs using Tableaux with Superdeduction*, in "electronic Journal of Mathematics and Technology", February 2013, vol. 7, n^{o} 2, pp. 1 - 20, https://hal.archives-ouvertes.fr/hal-01099371

[36] R. Gandy. *Church's Thesis and Principles for Mechanisms*, in "The Kleene Symposium", North-Holland, 1980

[37] J.-Y. Girard, Y. Lafont, P. Taylor. *Proofs and Types*, Cambridge University Press, 1988

[38] R. Harper, F. Honsell, G. Plotkin. *A Framework for Defining Logics*, in "Journal of the association for computing machinery", 1993, pp. 194–204

[39] J. Harrison. *HOL Light: An Overview*, in "Theorem Proving in Higher Order Logics", S. Berghofer, T. Nipkow, C. Urban, M. Wenzel (editors), Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2009, vol. 5674, pp. 60-66, http://dx.doi.org/10.1007/978-3-642-03359-9_4

[40] L. M. Hines. *STR$\dotplus$VE$\subseteq$: The STR$\dotplus$VE-Based Subset Prover*, in "Conference on Automated Deduction (CADE)", Kaiserslautern (Germany), LNCS, Springer, July 1990, pp. 193–206

[41] J. Hurd. *The OpenTheory Standard Theory Library*, in "NASA Formal Methods", M. Bobaru, K. Havelund, G. Holzmann, R. Joshi (editors), Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2011, vol. 6617, pp. 177-191, http://dx.doi.org/10.1007/978-3-642-20398-5_14

[42] M. Jacquel, K. Berkani, D. Delahaye, C. Dubois. *Verifying B Proof Rules using Deep Embedding and Automated Theorem Proving*, in "Software Engineering and Formal Methods", November 2011, vol. 7041, pp. 253-268 [*DOI :* 10.1007/978-3-642-24690-6_18], https://hal.archives-ouvertes.fr/hal-00722373

[43] J.-P. Jouannaud, A. Rubio. *Polymorphic Higher-Order Recursive Path Orderings*, in "Journal of the ACM", 2007, vol. 54, n^{o} 1, pp. 1-48

[44] J.-P. Jouannaud, A. Rubio. *The Higher-Order Recursive Path Ordering*, in "Proceedings of the 14th IEEE Symposium on Logic in Computer Science", 1999

[45] A. Kersani, N. Peltier. *Combining Superposition and Induction: A Practical Realization*, in "FroCoS 2013 - 9th International Symposium on s of Combining Systems (co-located with TABLEAUX 2013)", Nancy, France, P. Fontaine, C. Ringeissen, R. A. Schmidt (editors), Lecture Notes in Computer Science (LNCS), Springer, September 2013, vol. 8152, pp. 7-22, Session: Inductive Theorem Proving [*DOI :* 10.1007/978-3-642-40885-4_2], https://hal.archives-ouvertes.fr/hal-00934610

[46] K. Korovin. *iProver – An Instantiation-Based Theorem Prover for First-Order Logic (System Description)*, in "IJCAR", A. Armando, P. Baumgartner (editors), Lecture Notes in Artificial Intelligence, Springer, 2008, vol. 5195, pp. 292-298

[47] F. Rabe, M. Kohlhase. *A Scalable Module System*, in "Inf. Comput.", September 2013, vol. 230, pp. 1–54, http://dx.doi.org/10.1016/j.ic.2013.06.001

[48] B. R. SALINAS. *Puntos fijos en conjuntos ordenados*, Publicaciones del Seminario Matematico Garcia de Galdeano, Consejo Superior de Investigationes Cientificas, Facultad de Ciencas de Zaragoza, 1969, English summary on http://www.ams.org/mathscinet-getitem?mr=250943

[49] R. STATMAN. *A New Type Assignment for Strongly Normalizable Terms*, in "Computer Science Logic 2013 (CSL 2013)", Dagstuhl, Germany, S. R. D. ROCCA (editor), Leibniz International Proceedings in Informatics (LIPIcs), Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013, vol. 23, pp. 634–652