Activity Report 2015

Team DEDUCTEAM

Deduction modulo, interopérabilité et démonstration automatique

Inria teams are typically groups of researchers working on the definition of a common project, and objectives, with the goal to arrive at the creation of a project-team. Such project-teams may include other partners (universities or research institutions).

RESEARCH CENTER
Paris - Rocquencourt

THEME
Proofs and Verification

Table of contents

Team DEDUCTEAM

Creation of the Team: 2011 December 01

Keywords:

Computer Science and Digital Science:

2. - Software
3. - Data and knowledge
7. - Fundamental Algorithmics

Other Research Topics and Application Domains:

7. - Transport and logistics

1. Members

Research Scientists

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

PhD Students

Simon Cruanes [École Polytechnique, until Sep 2015]
Ali Assaf [École Polytechnique, until Sep 2015]
Guillaume Bury [ENS Paris, Université Paris Diderot]
Frédéric Gilbert [Ecole des Ponts ParisTech]
Kailiang Ji [ANR Locali, until Oct 2015]
Raphaël Cauderlier [CNAM]
Pierre Halmagrand [CNAM]
Ronan Saillard [ENSM Paris, until Sep 2015]

Post-Doctoral Fellows

Vaston Goncalves Da Costa [Universidade Federal de Goias, until Feb 2015]
Simon Martiel [Université Joseph Fourier, since Oct 2015]
Arnaud Spiwack [ENSM Paris, until Oct 2015]

Visiting Scientists

Bruno Bernardo [Until Sep 2015]
Ying Jiang [ISCAS, Aug-Sep 2015]
Guillaume Burel [MdC ENSIIE]
David Delahaye [MdC CNAM, until Aug 2015, HdR]
Catherine Dubois [Pr ENSIIE, HdR]
Olivier Hermant [ENSM Paris]
Jean-Pierre Jouannaud [Université Paris-Sud, École Polytechnique, HdR]

Administrative Assistant

Virginie Collette [Inria]

Others

Gaetan Gilbert [Intern, until Aug 2015]
Shuai Wang [Intern, ENS Cachan, Mar-Aug 2015]
Éric Uzena [Intern, Denis Diderot]

2. Overall Objectives

2.1. Objectives

The team investigates applications of recent results in proof theory to the design of logical frameworks and automated theorem proving systems. It develops the Dedukti logical framework 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 [41]. 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 [40], HOL [47], 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 [53].

A thesis, which is at the root of our research effort, and which was already formulated in [46] 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*. Such a theory independent proof-checker is called a *Logical Framework*.

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 [52] 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 SuperZenon [49] [44] and is an extension to superdeduction (which is a variant of Deduction modulo), and the second one is called ZenonModulo [42], [43] 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 SuperZenon has been successfully applied to verify B proof rules of Atelier B (work in collaboration with Siemens). Similarly, we plan to apply ZenonModulo 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 [45] 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 (SuperZenon and ZenonModulo). 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 [48]. From this work presented in the Doctoral dissertation of Mélanie Jacquel, the SuperZenon tool [49] [44] 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 ZenonModulo [42], [43] (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 ZenonModulo 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. Highlights of the Year

5.1. Highlights of the Year

Deducteam released a new version of Dedukti, more efficient, and with new features (e.g. higher-order patterns, confluence checking).

6. New Software and Platforms

6.1. Introduction

Deducteam develops several kinds of tools or libraries:

- Proof checkers:
  - Dedukti: proof checker for the $\lambda\Pi$-calculus modulo rewriting
  - Sukerujo: extension of Dedukti with syntactic constructions for records, strings, lists, etc.
  - Rainbow: CPF termination certificate verifier
- Tools for translating into Dedukti's proof format proofs coming from various other provers:
  - Coqine translates Coq proofs
  - Focalide translates Focalize proofs
  - Holide translates OpenTheory proofs (HOL-Light, HOL4, ProofPower)
  - Krajono translates Matita proofs
  - Sigmaid translates ς-calculus
- Automated theorem provers:
  - iProverModulo: theorem prover based on polarized resolution modulo

  - SuperZenon: extension of Zenon using superdeduction
  - ZenonArith: extension of Zenon using the simplex algorithm for arithmetic
  - ZenonModulo: extension of Zenon using deduction modulo and producing Dedukti proofs
  - Zipperposition: superposition prover featuring arithmetic and induction
  - HOT: automated termination prover for higher-order rewrite systems
- Libraries or generation tools:
  - CoLoR: Coq library on rewriting theory and termination
  - Logtk: library for first-order automated reasoning
  - mSat: modular SAT/SMT solver with proof output
  - Moca: generator of construction functions for types with relations on constructors

In the following, we only details software that received improvements in 2015.

In addition, Shuai Wang developed the ProofCloud prototype, a proof retrieval engine for verified higher order proofs. ProofCloud provides a fast proof searching service for mathematicians and computer scientists for the reuse of proofs and proof packages. Using ProofCloud, he conducted a statistical analysis of the OpenTheory repository.

6.2. Autotheo

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

- Contact: Guillaume Burel
- URL: http://www.ensiie.fr/~guillaume.burel/blackandwhite_autotheo.html.en

In 2015, we extended Autotheo so that it prints a derivation of the transformation of the axioms into rewriting rules. This derivation is in TSTP format and includes the CNF conversions obtained from the prover E.

6.3. CoLoR

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

- Contact: Frédéric Blanqui
- URL: http://color.inria.fr/

In 2015, CoLoR has been enriched and improved in various ways:

- Its compilation time has been improved by about 20%.
- The results on computability have been extended to η-reduction.
- It has been enriched by a library on finite and infinite sets, and a proof of the infinite Ramsey's theorem [54].
- CoLoR is now available on OPAM.

6.4. Coqine

Coqine translates Coq proofs into Dedukti proofs.

- Contact: Guillaume Burel
- URL: http://www.ensiie.fr/~guillaume.burel/blackandwhite_coqInE.html.en

The addition of higher-order pattern matching in Dedukti allowed the encoding of universes.

6.5. 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 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.

- Contact: Olivier Hermant
- URL: http://dedukti.gforge.inria.fr/

The new version of Dedukti (v2.5) brings two major improvements.

First the typing of rewrite rules has been completely reworked. It can now check a large class of rewrite rules including rules whose left-hand sides are not algebraic nor well-typed. Moreover the typing context do not need to be given with the rewrite rule anymore, as it is inferred by Dedukti, and therefore it is more convenient for the user.

Second, Dedukti can now be interfaced with automatic confluence checkers in order to check that the rewrite system generated by the rewrite rules together with beta reduction is confluent. This verification is important as the soundness of the program relies on this hypothesis.

6.6. Focalide

Focalide is an extension of the FoCaLize compiler which produces Dedukti files.

- Contact: Raphaël Cauderlier
- URL: http://deducteam.gforge.inria.fr/focalide/

Focalide has been improved to support FoCaLiZe proofs found by Zenon using the Dedukti backend for Zenon. 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.

6.7. Holide

Holide translates HOL proofs to Dedukti proofs, using the OpenTheory standard (common to HOL Light and HOL4).

- Contact: Guillaume Burel
- URL: http://deducteam.gforge.inria.fr/holide/

Shuai Wang fixed a number of problems, especially in the translation of type variables, allowing us to translate more libraries.

6.8. iProverModulo

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.

- Contact: Guillaume Burel
- URL: http://www.ensiie.fr/~guillaume.burel/blackandwhite_iProverModulo.html.en

In 2015, we improved its integration with Autotheo.

6.9. Krajono

Krajono translates Matita proofs into Dedukti proofs.

- Contact: Guillaume Burel
- URL: http://deducteam.gforge.inria.fr/krajono/

First working version able to translate the Matita library on arithmetics.

6.10. 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.

- Contact: Guillaume Bury
- URL: https://github.com/Gbury/mSAT

mSAT now provides a functor for generating a McSat solver, outputs a model or a proof, and provides a push/pop functionality.

6.11. ZenonModulo

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.

- Contact: Pierre Halmagrand
- URL: http://deducteam.gforge.inria.fr/zenonmodulo/

In 2015, we extended Zenon Modulo to polymorphism. Moreover, it can now take TPTP-TFF1 problems as input, and output Dedukti's proofs.

Guillaume Bury continued to improve an extension of Zenon with arithmetic.

6.12. Zipperposition

Zipperposition is an implementation of the superposition method that 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.

- Contact: Simon Cruanes
- URL: http://deducteam.gforge.inria.fr/zipperposition/

In 2015, we extended Zipperposition to structural induction.

7. New Results

7.1. Termination

In [15], Frédéric Blanqui showed how to extend the notion of reducibility introduced by Girard for proving the termination of β-reduction in the polymorphic λ-calculus, to prove the termination of various kinds of rewrite relations on λ-terms, including rewriting modulo some equational theory and rewriting with matching modulo $\beta\eta$, by using the notion of computability closure. This provides a powerful termination criterion for various higher-order rewriting frameworks, including Klop's Combinatory Reductions Systems with simple types and Nipkow's Higher-order Rewrite Systems.

In [16], Frédéric Blanqui, together with Jean-Pierre Jouannaud and Albert Rubio, introduced the computability path ordering (CPO), a recursive relation on terms obtained by lifting a precedence on function symbols. A first version, core CPO, is essentially obtained from the higher-order recursive path ordering (HORPO) by eliminating type checks from some recursive calls and by incorporating the treatment of bound variables as in the so-called computability closure. The well-foundedness proof shows that core CPO captures the essence of computability arguments à la Tait and Girard, therefore explaining its name. We further show that no more type check can be eliminated from its recursive calls without loosing well-foundedness, but one for which we found no counterexample yet. Two extensions of core CPO are then introduced which allow one to consider: the first, higher-order inductive types; the second, a precedence in which some function symbols are smaller than application and abstraction.

Another extension of CPO, to dependently typed terms, has been developed by Jean-Pierre Jouannaud and Jianqi Li in [50].

Jean-Pierre Jouannaud and Albert Rubio showed in [51] how to modify recursive path orders for higher-order terms which, like CPO, include $\beta\eta$-reductions, into orders that are compatible with $\beta\eta$-conversion. The result is a powerful order for proving termination of higher-order rewrite rules based on higher-order pattern matching.

Gaëtan Gilbert and Olivier Hermant have introduced a constructive way to perform proof normalization through completeness proofs [23].

Frédéric Blanqui formalized Ramsey's proof of the (infinite) Ramsey's theorem [54] (see http://color.inria.fr/).

7.2. Confluence

Jean-Pierre Jouannaud, in collaboration with Jiaxiang Liu, has started a program in order to enable confluence proofs in $\lambda\Pi$ modulo, investigating several open confluence problems for non-terminating relations. In [27], together with Mizuhito Ogawa, they introduced the new class of layered rewrite systems, and showed that their confluence can be reduced to that of their critical pairs computed by using unification over infinite rational terms when they do not increase the layer-depth of terms. This shows why an old example of non-terminating, left non-linear, critical pair free rewrite system due to Klop was non-confluent: it indeed had a critical pair in infinite rational trees. In the same paper, they also give an example of a non-confluent, layer-depth increasing system which has no critical pairs, hence showing that layer-depth plays a key role.

7.3. Automated theorem proving

In [25], Guillaume Bury, Raphaël Cauderlier and Pierre Halmagrand presented the extension of the automated theorem prover Zenon to ML-style polymorphism.

In [20], Guillaume Bury, David Delahaye, Damien Doligez, Pierre Hamalgrand and Olivier Hermant introduced an encoding of the set theory of the B method using polymorphic types and deduction modulo, used for the automated verification of proof obligations in the framework of the BWare project.

In [24], Kailiang Ji designed a strategy to translate model-checking problems into proving the satisfiability of a set of first-order formulas. The focus is to give an encoding of temporal properties expressed in CTL as first-order formulas, by translating the logical equivalence between temporal operators into rewrite rules. In this way, proof-search algorithms designed for Deduction Modulo, such as Resolution Modulo or Tableaux Modulo, can be used to verify temporal properties of finite transition systems. This strategy is implemented in iProver Modulo, and the testing results show that Resolution Modulo can be considered as a new way to quickly determine whether a temporal property is violated or not in transition system models.

7.4. $\lambda\Pi$ modulo and Dedukti

Gaëtan Gilbert, supervised by Arnaud Spiwack, wrote a prototype of a principle unification and type inference mechanism for Dedukti, based on a monadic API. This prototype separates with an abstraction barrier a unifier kernel which implements correct unification primitives from the unification algorithm and heuristics. The unification algorithm is written in a style which closely mirrors a pen-and-paper deduction rule presentation.

Éric Uzena, supervised by David Delahaye and Arnaud Spiwack, wrote a prototype of an extension of Dedukti with associative and commutative symbols and rewriting modulo associativity and commutativity of these symbols.

7.5. Encodings into Dedukti and interoperability

Ali Assaf, Guillaume Burel, Raphaël Cauderlier, David Delahaye, Gilles Dowek, Catherine Dubois, Frédéric Gilbert, Pierre Hamalgrand, Olivier Hermant, and Ronan Saillard have written a synthetic paper on the Dedukti system and on the expression of theories in this system. This paper is submitted to publication.

Ali Assaf [32] proved that Cousineau and Dowek's embedding of functional pure type systems [41] is conservative with respect to the original systems, using a new notion of reducibility called relative normalization. Together with Cousineau and Dowek's original result on the preservation of typing, this result justifies the use of the $\lambda\Pi$-calculus modulo as a logical framework.

Ali Assaf's translation of the calculus of inductive constructions to the $\lambda\Pi$-calculus modulo, which was presented at the TYPES conference in 2014, has been published in the postproceedings of TYPES 2014 [39]. This translation, which is based on the translation of pure type systems by Cousineau and Dowek [41], is implemented in the automated translation tool Coqine.

Ali Assaf and Guillaume Burel presented their translation of HOL to Dedukti at the PxTP 2015 workshop [18]. This translation, which is based on the translation of pure type systems by Cousineau and Dowek [41], is implemented in the automated translation tool Holide.

Raphaël Cauderlier and Catherine Dubois' translation of object calculus and subtyping to Dedukti, which was presented at the TYPES conference in 2014, has been published in the post-proceedings of TYPES 2014 [34].

In [26], Raphaël Cauderlier and Pierre Halmagrand presented a shallow embedding into Dedukti of proofs produced by ZenonModulo, an extension of the tableau-based first-order theorem prover Zenon to deduction modulo and typing.

In [33], Ali Assaf and Raphaël Cauderlier have combined simple developments written in Coq and HOL using Dedukti and the existing translation tools Coqine and Holide. This work is a first step towards using Dedukti as a framework for proof interoperability.

7.6. Proof theory

Guillaume Burel, Gilles Dowek and Ying Jiang have introduced a general framework to prove the decidability of reachability and provability problems. This framework uses an analogy between the objects recognized by an automaton and cut-free proofs. Various aspects of this work have been published at FroCoS [19], LPAR [21], and another paper is in preparation.

Gilles Dowek's paper on the definition of the classical connectives and quantifiers has been published [30].

Arnaud Spiwack gave a predicative shallow embedding of a weak version of system U^- in dependent type theory, for Hurkens's paradox to hold. He also showed that a variety of incarnations of Hurkens's paradox are straightforward instantiations of this encoding, greatly simplifying existing proofs.

Arnaud Spiwack developped a topos-theoretic methodology to reason equationally on circuit languages. Results that hold for combinational circuits are lifted to sequential circuits thanks to a transfer principle. This approach allows, in particular, to simplify reasoning about more complex temporal gates than the unit delay. These results aim at enriching the compiler of the Faust audio signal processing programming language, which features such complex temporal gates.

For the sake of reliability, the kernels of Interactive Theorem Provers (ITPs) are kept relatively small in general. On top of the kernel, additional symbols and inference rules are defined. Some dependency analysis of symbols of HOL Light indicates that the depth of dependency could be reduced by introducing a few more symbols to the kernel. Shuai Wang showed that extending the kernel of HOL Light is a successful attempt to reduce proof size and speed up proof-checking. More specifically, symbols and inference rules of universal quantification

and implication were added to the kernel. This approach has been proved to give equivalent proof-checking results with the size of the proof files reduced to 24% on average and a speedup of 38% for proof-checking overall.

7.7. Computation models

Pablo Arrighi and Gilles Dowek have studied the expression of mecanic motions in cellular automata. Part of this work has been published in TPNC [17] and another paper is in preparation.

Arnaud Spiwack developped a variant of Turing machine where the tape is replaced by an unlabeled tree. The additional structure makes combining machines much easier, making it tractable to give explicit descriptions of rather complex machines. The cost model of these machines models that of purely functional programming languages, making it possible to compare mathematically the complexity of imperative algorithms and of purely functional algorithms.

8. Partnerships and Cooperations

8.1. National Initiatives

8.1.1. ANR Locali

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

8.1.2. 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.

8.1.3. ANR Tarmac

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

8.2. International Research Visitors

8.2.1. Visits of International Scientists

Jim Lipton, professor at Wesleyan University (USA) has visited Deducteam from 9 to 14 March 2015.

8.2.1.1. Internships

Gaetan Gilbert did an internship with Arnaud Spiwack and Olivier Hermant.

Shuai Wang did an internship with Gilles Dowek.

Éric Uzena did an internship with Arnaud Spiwack and David Delahaye.

8.2.2. Visits to International Teams

8.2.2.1. Sabbatical programme

Olivier Hermant is a visiting professor at Wesleyan University (USA) since September 2015.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific events selection

9.1.1.1. Chair of conference program committees

Gilles Dowek was PC chair of TLCA-RTA.

9.1.1.2. Member of the conference program committees

Gilles Dowek was PC member of CADE, ICTAC and eMoocs.

Guillaume Burel was PC member of PxTP'15 and IWIL'15.

9.1.1.3. Reviewer

Frédéric Blanqui has reviewed papers for TYPES'14 post-proceedings and LICS'15.

Guillaume Burel has reviewed a paper for Tableaux'15.

Olivier Hermant has reviewed papers for Tableaux'15 and CADE-25, a project for ANR (second phase) and pre-projects for ANR (first phase).

9.1.2. Journal

9.1.2.1. Member of the editorial boards

Gilles Dowek is an editor of TCS.

9.1.2.2. Reviewer - Reviewing activities

Frédéric Blanqui reviewed a paper for TCS.

Guillaume Burek reviewed a paper for Formal Aspects of Computing.

Olivier Hermant reviewed a paper for TCS.

9.1.3. Invited talks

Gilles Dowek was invited to DCM and Tools for teaching logic.

9.1.4. Scientific expertise

Gilles Dowek has been a consultant for the Conseil Scientifique des Programmes.

Gilles Dowek is the President of the Scientific board of the Société Informatique de France.

Gilles Dowek is a member of the Commission de réflexion sur l'éthique de la recherche en sciences et technologies du numérique d'Allistène (CERNA).

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

Master: Pierre Halmagrand, Initiation à la Méthode B, 54 HETD, M2, CNAM.

License: Frédéric Gilbert, Les principes des langages de programmation, 40, L3, Ecole Polytechnique.

License: Raphaël Cauderlier, Introduction aux Bases de Données Relationnelles, 21, L2, UPMC.

License: Raphaël Cauderlier, Projet (Application) : Android, 42, L2, UPMC.

License: Raphaël Cauderlier, Eléments de programmation 1, 58, L1, UPMC.

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

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

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

Master: Guillaume Burel, Systèmes et langages formels, 8.75 HETD, M1, ENSIIE.

Master: Guillaume Burel, Compilation, 24.5 HETD, M1, ENSIIE.

Master: Guillaume Burel, Preuve, Analyse statique, Vérification run-time, 13 HETD, M2 CILS, Paris-Saclay.

Licence: Guillaume Burel is in charge of the 4th and 5th semesters of the engineering degree at ENSIIE, and was responsible for the final engineer internship until September, 2015.

Gilles Dowek has given a course at the MPRI.

Gilles Dowek has been teaching at ENS-Cachan.

Gilles Dowek has given various talks about teaching informatics in primary and secondary education.

Gilles Dowek has participated to several training sessions for high school teachers with La Main à la Pâte.

Licence: Olivier Hermant, Introduction to Programming in Python, 100 HETD, L1-L3, Wesleyan University, USA.

9.2.2. Supervision

PhD: Simon Cruanes, Extending Superposition with Integer Arithmetic, Structural Induction, and Beyond [13], defended at École polytechnique on September the 10th, supervised by Guillaume Burel and Gilles Dowek.

PhD: Bruno Bernardo, An implicit Calculus of Constructions with dependent sums and decidable type inference [12], defended at École polytechnique on September the 18th, supervised by Bruno Barras and Gilles Dowek.

PhD: Ali Assaf, A framework for defining computational higher-order logics [11], defended at École polytechnique on September 28, 2015, supervised by Gilles Dowek and Guillaume Burel.

PhD: Kailiang Ji, Model Checking and Theorem Proving [14], defended at Paris Diderot on September 25, 2015, supervised by Gilles Dowek.

PhD: Ronan Saillard, Typechecking in the $\lambda\Pi$-Calculus Modulo: Theory and Practice [55], defended at MINES ParisTech on September 25, 2015, supervised by Olivier Hermant and Pierre Jouvelot.

PhD in progress: Guillaume Bury, Deduction Modulo Theory, started October 1st, 2015, supervised by David Delahaye and Gilles Dowek.

PhD in progress: Raphaël Cauderlier, Object-oriented mechanisms for interoperability of proof systems, started September 1st, 2013, supervised by Catherine Dubois.

9.2.3. Juries

Gilles Dowek is a member of the prix Le Monde de la Recherche Universitaire.

9.3. Popularization

Gilles Dowek has given various popular science talks.

Gilles Dowek writes a monthy chronicle in Pour la Science.

Gilles Dowek is a a member of the Scientific board of La Main à la Pâte.

10. 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^{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^{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^{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] A. ASSAF. *A framework for defining computational higher-order logics*, École polytechnique, September 2015, https://pastel.archives-ouvertes.fr/tel-01235303

[12] B. BERNARDO. *An implicit Calculus of Constructions with dependent sums and decidable type inference*, École polytechnique, October 2015, Version soutenance, https://hal.inria.fr/tel-01197380

[13] S. CRUANES. *Extending Superposition with Integer Arithmetic, Structural Induction, and Beyond*, École polytechnique, September 2015, https://hal.archives-ouvertes.fr/tel-01223502

[14] K. JI. *Model Checking and Theorem Proving*, Paris Diderot, September 2015, https://hal.inria.fr/tel-01251073

Articles in International Peer-Reviewed Journals

[15] F. BLANQUI. *Termination of rewrite relations on λ-terms based on Girard's notion of reducibility*, in "Journal of Theoretical Computer Science (TCS)", December 2015, vol. 611, n° 50-86, 37 p. [*DOI :* 10.1016/J.TCS.2015.07.045], https://hal.inria.fr/hal-01191693

[16] F. BLANQUI, J.-P. JOUANNAUD, A. RUBIO. *The Computability Path Ordering*, in "Logical Methods in Computer Science", October 2015 [*DOI :* 10.2168/LMCS-11(4:3)2015], https://hal.inria.fr/hal-01163091

International Conferences with Proceedings

[17] P. ARRIGHI, G. DOWEK. *Discrete Geodesics and Cellular Automata*, in "Theory and Practice of Natural Computing", Mieres, Spain, December 2015 [*DOI :* 10.1007/978-3-319-26841-5_11], https://hal.inria.fr/hal-01252131

[18] A. ASSAF, G. BUREL. *Translating HOL to Dedukti*, in "Fourth Workshop on Proof eXchange for Theorem Proving, PxTP'15", Berlin, Germany, C. KALISZYK, A. PASKEVICH (editors), EPTCS, August 2015, vol. 186, pp. 74-88 [*DOI :* 10.4204/EPTCS.186.8], https://hal.archives-ouvertes.fr/hal-01097412

[19] G. BUREL, G. DOWEK, Y. JIANG. *A Completion Method to Decide Reachability in Rewrite Systems*, in "International Symposium on Frontiers of Combining Systems FroCoS'15", Wroclaw, Poland, C. LUTZ, S. RANISE (editors), Lecture Notes in Computer Science, Springer, September 2015, vol. 9322, pp. 205-219 [*DOI :* 10.1007/978-3-319-24246-0_13], https://hal.inria.fr/hal-01252138

[20] G. BURY, D. DELAHAYE, D. DOLIGEZ, P. HALMAGRAND, O. HERMANT. *Automated Deduction in the B Set Theory using Typed Proof Search and Deduction Modulo*, in "LPAR 20 : 20th International Conference on Logic for Programming, Artificial Intelligence and Reasoning", Suva, Fiji, November 2015, https://hal-mines-paristech.archives-ouvertes.fr/hal-01204701

[21] G. DOWEK, Y. JIANG. *Decidability, Introduction Rules and Automata*, in "International Conferences on Logic for Programming, Artificial Intelligence and Reasoning", Bula, Fiji, November 2015 [*DOI :* 10.1007/978-3-662-48899-7_8], https://hal.inria.fr/hal-01252135

[22] F. GILBERT. *A Lightweight Double-negation Translation*, in "LPAR-20. 20th International Conference on Logic for Programming, Artificial Intelligence and Reasoning", Suva, Fiji, EasyChair Proceedings in Computing, November 2015, https://hal.inria.fr/hal-01245021

[23] G. GILBERT, O. HERMANT. *Normalization by Completeness with Heyting Algebras*, in "LPAR 20 : 20th International Conference on Logic for Programming, Artificial Intelligence and Reasoning", Suva, Fiji, November 2015, https://hal-mines-paristech.archives-ouvertes.fr/hal-01204599

[24] K. JI. *CTL Model Checking in Deduction Modulo*, in "Automated Deduction - CADE-25", Berlin, Germany, August 2015, pp. 295-310 [*DOI :* 10.1007/978-3-319-21401-6_20], https://hal.inria.fr/hal-01241132

Conferences without Proceedings

[25] G. BURY, R. CAUDERLIER, P. HALMAGRAND. *Implementing Polymorphism in Zenon*, in "11th International Workshop on the Implementation of Logics (IWIL)", Suva, Fiji, November 2015, https://hal.inria.fr/hal-01243593

[26] R. CAUDERLIER, P. HALMAGRAND. *Checking Zenon Modulo Proofs in Dedukti*, in "Fourth Workshop on Proof eXchange for Theorem Proving (PxTP)", Berlin, Germany, August 2015, https://hal.inria.fr/hal-01171360

[27] J. LIU, J.-P. JOUANNAUD, M. OGAWA. *Confluence of layered rewrite systems*, in "24th EACSL Annual Conference on Computer Science Logic (CSL 2015)", Berlin, Germany, September 2015, vol. 41, pp. 423–440 [*DOI :* 10.4230/LIPICS.CSL.2015.423], https://hal.inria.fr/hal-01199062

[28] R. SAILLARD. *Rewriting Modulo β in the $\lambda\Pi$-Calculus Modulo*, in "Logical Frameworks and Meta Languages: Theory and Practic, Affiliated with CADE-25", Berlin, Germany, August 2015, https://hal-mines-paristech.archives-ouvertes.fr/hal-01176715

[29] S. WANG. *Higher Order Proof Engineering: Proof Collaboration, Transformation, Checking and Retrieval*, in "AITP 2016 - Conference on Artificial Intelligence and Theorem Proving", Obergurgl, Austria, April 2016, https://hal.inria.fr/hal-01250197

Scientific Books (or Scientific Book chapters)

[30] G. DOWEK. *On the definition of the classical connectives and quantifiers*, in "Why is this a Proof?, Festschrift for Luiz Carlos Pereira", E. H. HAEUSLER, W. DE CAMPOS SANZ, B. LOPES (editors), College Publications, 2015, https://hal.inria.fr/hal-01252221

Scientific Popularization

[31] S. ALAYRANGUES, G. DOWEK, E. KERRIEN, J. MAIRESSE, T. VIÉVILLE. *Médiation en sciences du numériques : un levier pour comprendre notre quotidien ?*, in "Science & You", Nancy, France, June 2015, https://hal.inria.fr/hal-01211457

Other Publications

[32] A. ASSAF. *Conservativity of embeddings in the lambda-Pi calculus modulo rewriting (long version)*, 2015, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01084165

[33] A. ASSAF, R. CAUDERLIER. *Mixing HOL and Coq in Dedukti (Rough Diamond)*, 2015, working paper or preprint, https://hal.inria.fr/hal-01141789

[34] R. CAUDERLIER, C. DUBOIS. *Objects and subtyping in the $\lambda\Pi$-calculus modulo*, June 2015, working paper or preprint, https://hal.inria.fr/hal-01097444

[35] G. DOWEK. *Models and termination of proof-reduction in the $\lambda\Pi$-calculus modulo theory*, January 2015, working paper or preprint, https://hal.inria.fr/hal-01101834

[36] G. DOWEK. *Rules and derivations in an elementary logic course*, January 2016, working paper or preprint, https://hal.inria.fr/hal-01252124

[37] G. DOWEK, Y. JIANG. *Cut-elimination and the decidability of reachability in alternating pushdown systems*, January 2015, working paper or preprint, https://hal.inria.fr/hal-01101835

[38] K. JI. *Resolution in Solving Graph Problems*, December 2015, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01245138

References in notes

[39] A. ASSAF. *A calculus of constructions with explicit subtyping*, 2014, working paper or preprint, https://hal.archives-ouvertes.fr/hal-01097401

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

[41] 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

[42] 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

[43] 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

[44] 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

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

[46] R. HARPER, F. HONSELL, G. PLOTKIN. *A Framework for Defining Logics*, in "Journal of the association for computing machinery", 1993, pp. 194–204

[47] 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

[48] 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

[49] 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

[50] J.-P. JOUANNAUD, J.-Q. LI. *Termination of dependently typed rewrite rules*, in "Proc. TLCA 2015", Varsaw, Poland, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, July 2015, vol. LIPIcs, n^{o} 38 [*DOI :* 10.4230/LIPICS.TLCA.2015.X], https://hal.inria.fr/hal-01239083

[51] J.-P. JOUANNAUD, A. RUBIO. *Normal Higher-Order Termination*, in "ACM Transactions on Computational Logic", March 2013 [*DOI :* 10.1145/26999.13], https://hal.inria.fr/hal-01239068

[52] 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

[53] 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

[54] F. P. RAMSEY. *On a problem of formal logic*, in "Proceedings of the London Mathematical Society", 1930, vol. s2-30, n^{o} 1, pp. 264-286, http://dx.doi.org/10.1112/plms/s2-30.1.264

[55] R. SAILLARD. *Typechecking in the $\lambda\Pi$-Calculus Modulo: Theory and Practice*, MINES ParisTech, 2015