IN PARTNERSHIP WITH:
CNRS

Ecole Polytechnique

Activity Report 2016

Project-Team PARSIFAL

Proof search and reasoning with logic specifications

IN COLLABORATION WITH: Laboratoire d'informatique de l'école polytechnique (LIX)

RESEARCH CENTER
Saclay - Île-de-France

THEME
Proofs and Verification

Table of contents

Project-Team PARSIFAL

Creation of the Project-Team: 2007 July 01

Keywords:

Computer Science and Digital Science:

2.1.1. - Semantics of programming languages
2.1.11. - Proof languages
2.4.2. - Model-checking
2.4.3. - Proofs
7.4. - Logic in Computer Science

Other Research Topics and Application Domains:

9.4.1. - Computer science
9.4.2. - Mathematics
9.6. - Reproducibility

1. Members

Research Scientists

Beniamino Accattoli [Inria, Researcher]
Kaustuv Chaudhuri [Inria, Researcher]
Stéphane Graham-Lengrand [CNRS, Researcher, HDR]
François Lamarche [Inria, Researcher]
Dale Miller [Team leader, Inria, Researcher]
Lutz Straßburger [Inria, Researcher, HDR]

Engineers

Danko Ilik [Inria, until Aug 2016, funded by FP7 ERC PROOFCERT]
Tomer Libal [Inria, funded by FP7 ERC PROOFCERT]
Marco Volpe [Inria, funded by FP7 ERC PROOFCERT]
Noam Zeilberger [Inria, until Oct 2016, funded by FP7 ERC PROOFCERT]

PhD Students

Roberto Blanco Martinez [Inria, funded by FP7 ERC PROOFCERT]
Ulysse Gerard [Inria, funded by FP7 ERC PROOFCERT]
Quentin Heath [Inria, funded by FP7 ERC PROOFCERT]
Sonia Marin [Inria, funded by FP7 ERC PROOFCERT]
François Thiré [ENS Cachan, from Apr 2016]

Post-Doctoral Fellows

Taus Brock-Nannestad [Inria, until Sep 2016, funded by FP7 ERC PROOFCERT]
Giselle Machado Nogueira Reis [Inria, until Jul 2016, funded by FP7 ERC PROOFCERT]
Matthias Puech [Inria, funded by FP7 ERC PROOFCERT]
Hugh Paul Steele [Inria, until Nov 2016, funded by FP7 ERC PROOFCERT]

Visiting Scientists

Jose Carlos Espirito Santo [University of Minho, Portugal, Nov 2016]
Chuck Liang [Hofstra University, NY, USA, Jun 2016]

Administrative Assistant

Christine Aklouche [Inria]

Others

Ameni Chtourou [Inria, intern, from May 2016 until Jul 2016]

Leonardo Augusto Lima Ferreira Da Silva [Inria, Intern, until Feb 2016]

2. Overall Objectives

2.1. Main themes

The aim of the Parsifal team is to develop and exploit *proof theory* and *type theory* in the specification, verification, and analysis of computational systems.

- *Expertise*: the team conducts basic research in proof theory and type theory. In particular, the team is developing results that help with automated deduction and with the manipulation and communication of formal proofs.
- *Design*: based on experience with computational systems and theoretical results, the team develops new logical principles, new proof systems, and new theorem proving environments.
- *Implementation*: the team builds prototype systems to help validate basic research results.
- *Examples*: the design and implementation efforts are guided by examples of specification and verification problems. These examples not only test the success of the tools but also drive investigations into new principles and new areas of proof theory and type theory.

The foundational work of the team focuses on *structural* and *analytic* proof theory, *i.e.*, the study of formal proofs as algebraic and combinatorial structures and the study of proof systems as deductive and computational formalisms. The main focus in recent years has been the study of the *sequent calculus* and of the *deep inference* formalisms.

An important research question is how to reason about computational specifications that are written in a *relational* style. To this end, the team has been developing new approaches to dealing with induction, co-induction, and generic quantification. A second important question is of *canonicity* in deductive systems, *i.e.*, when are two derivations “essentially the same”? This crucial question is important not only for proof search, because it gives an insight into the structure and an ability to manipulate the proof search space, but also for the communication of *proof objects* between different reasoning agents such as automated theorem provers and proof checkers.

Important application areas currently include:

- Meta-theoretic reasoning on functional programs, such as terms in the λ-calculus
- Reasoning about behaviors in systems with concurrency and communication, such as the π-calculus, game semantics, *etc.*
- Combining interactive and automated reasoning methods for induction and co-induction
- Verification of distributed, reactive, and real-time algorithms that are often specified using modal and temporal logics
- Representing proofs as documents that can be printed, communicated, and checked by a wide range of computational logic systems.
- Development of cost models for the evaluation of proofs and programs.

3. Research Program

3.1. General overview

There are two broad approaches for computational specifications. In the *computation as model* approach, computations are encoded as mathematical structures containing nodes, transitions, and state. Logic is used to *describe* these structures, that is, the computations are used as models for logical expressions. Intensional operators, such as the modals of temporal and dynamic logics or the triples of Hoare logic, are often employed to express propositions about the change in state.

The *computation as deduction* approach, in contrast, expresses computations logically, using formulas, terms, types, and proofs as computational elements. Unlike the model approach, general logical apparatus such as cut-elimination or automated deduction becomes directly applicable as tools for defining, analyzing, and animating computations. Indeed, we can identify two main aspects of logical specifications that have been very fruitful:

- *Proof normalization*, which treats the state of a computation as a proof term and computation as normalization of the proof terms. General reduction principles such as β-reduction or cut-elimination are merely particular forms of proof normalization. Functional programming is based on normalization [69], and normalization in different logics can justify the design of new and different functional programming languages [41].
- *Proof search*, which views the state of a computation as a a structured collection of formulas, known as a *sequent*, and proof search in a suitable sequent calculus as encoding the dynamics of the computation. Logic programming is based on proof search [75], and different proof search strategies can be used to justify the design of new and different logic programming languages [73].

While the distinction between these two aspects is somewhat informal, it helps to identify and classify different concerns that arise in computational semantics. For instance, confluence and termination of reductions are crucial considerations for normalization, while unification and strategies are important for search. A key challenge of computational logic is to find means of uniting or reorganizing these apparently disjoint concerns.

An important organizational principle is structural proof theory, that is, the study of proofs as syntactic, algebraic and combinatorial objects. Formal proofs often have equivalences in their syntactic representations, leading to an important research question about *canonicity* in proofs – when are two proofs "essentially the same?" The syntactic equivalences can be used to derive normal forms for proofs that illuminate not only the proofs of a given formula, but also its entire proof search space. The celebrated *focusing* theorem of Andreoli [43] identifies one such normal form for derivations in the sequent calculus that has many important consequences both for search and for computation. The combinatorial structure of proofs can be further explored with the use of *deep inference*; in particular, deep inference allows access to simple and manifestly correct cut-elimination procedures with precise complexity bounds.

Type theory is another important organizational principle, but most popular type systems are generally designed for either search or for normalization. To give some examples, the Coq system [85] that implements the Calculus of Inductive Constructions (CIC) is designed to facilitate the expression of computational features of proofs directly as executable functional programs, but general proof search techniques for Coq are rather primitive. In contrast, the Twelf system [80] that is based on the LF type theory (a subsystem of the CIC), is based on relational specifications in canonical form (*i.e.*, without redexes) for which there are sophisticated automated reasoning systems such as meta-theoretic analysis tools, logic programming engines, and inductive theorem provers. In recent years, there has been a push towards combining search and normalization in the same type-theoretic framework. The Beluga system [81], for example, is an extension of the LF type theory with a purely computational meta-framework where operations on inductively defined LF objects can be expressed as functional programs.

The Parsifal team investigates both the search and the normalization aspects of computational specifications using the concepts, results, and insights from proof theory and type theory.

3.2. Inductive and co-inductive reasoning

The team has spent a number of years in designing a strong new logic that can be used to reason (inductively and co-inductively) on syntactic expressions containing bindings. This work is based on earlier work by McDowell, Miller, and Tiu [71] [70] [76] [86], and on more recent work by Gacek, Miller, and Nadathur [4] [56]. The Parsifal team, along with our colleagues in Minneapolis, Canberra, Singapore, and Cachen, have been building two tools that exploit the novel features of this logic. These two systems are the following.

- Abella, which is an interactive theorem prover for the full logic.
- Bedwyr, which is a model checker for the "finite" part of the logic.

We have used these systems to provide formalize reasoning of a number of complex formal systems, ranging from programming languages to the λ-calculus and π-calculus.

Since 2014, the Abella system has been extended with a number of new features. A number of new significant examples have been implemented in Abella and an extensive tutorial for it has been written [1].

3.3. Developing a foundational approach to defining proof evidence

The team is developing a framework for defining the semantics of proof evidence. With this framework, implementers of theorem provers can output proof evidence in a format of their choice: they will only need to be able to formally define that evidence's semantics. With such semantics provided, proof checkers can then check alleged proofs for correctness. Thus, anyone who needs to trust proofs from various provers can put their energies into designing trustworthy checkers that can execute the semantic specification.

In order to provide our framework with the flexibility that this ambitious plan requires, we have based our design on the most recent advances within the theory of proofs. For a number of years, various team members have been contributing to the design and theory of *focused proof systems* [45] [48] [49] [50] [59] [67] [68] and we have adopted such proof systems as the corner stone for our framework.

We have also been working for a number of years on the implementation of computational logic systems, involving, for example, both unification and backtracking search. As a result, we are also building an early and reference implementation of our semantic definitions.

3.4. Deep inference

Deep inference [61], [63] is a novel methodology for presenting deductive systems. Unlike traditional formalisms like the sequent calculus, it allows rewriting of formulas deep inside arbitrary contexts. The new freedom for designing inference rules creates a richer proof theory. For example, for systems using deep inference, we have a greater variety of normal forms for proofs than in sequent calculus or natural deduction systems. Another advantage of deep inference systems is the close relationship to categorical proof theory. Due to the deep inference design one can directly read off the morphism from the derivations. There is no need for a counter-intuitive translation.

The following research problems are investigated by members of the Parsifal team:

- Find deep inference system for richer logics. This is necessary for making the proof theoretic results of deep inference accessible to applications as they are described in the previous sections of this report.
- Investigate the possibility of focusing proofs in deep inference. As described before, focusing is a way to reduce the non-determinism in proof search. However, it is well investigated only for the sequent calculus. In order to apply deep inference in proof search, we need to develop a theory of focusing for deep inference.

3.5. Proof nets and atomic flows

Proof nets and atomic flows are abstract (graph-like) presentations of proofs such that all "trivial rule permutations" are quotiented away. Ideally the notion of proof net should be independent from any syntactic formalism, but most notions of proof nets proposed in the past were formulated in terms of their relation to the sequent calculus. Consequently we could observe features like “boxes” and explicit “contraction links”. The latter appeared not only in Girard's proof nets [58] for linear logic but also in Robinson's proof nets [83] for classical logic. In this kind of proof nets every link in the net corresponds to a rule application in the sequent calculus.

Only recently, due to the rise of deep inference, new kinds of proof nets have been introduced that take the formula trees of the conclusions and add additional “flow-graph” information (see e.g., [6], [5] and [62]. On one side, this gives new insights in the essence of proofs and their normalization. But on the other side, all the known correctness criteria are no longer available.

This directly leads to the following research questions investigated by members of the Parsifal team:

- Finding (for classical logic) a notion of proof nets that is deductive, i.e., can effectively be used for doing proof search. An important property of deductive proof nets must be that the correctness can be checked in linear time. For the classical logic proof nets by Lamarche and Straßburger [6] this takes exponential time (in the size of the net).
- Studying the normalization of proofs in classical logic using atomic flows. Although there is no correctness criterion they allow to simplify the normalization procedure for proofs in deep inference, and additionally allow to get new insights in the complexity of the normalization.

3.6. Cost Models and Abstract Machines for Functional Programs

In the *proof normalization* approach, computation is usually reformulated as the evaluation of functional programs, expressed as terms in a variation over the λ-calculus. Thanks to its higher-order nature, this approach provides very concise and abstract specifications. Its strength is however also its weakness: the abstraction from physical machines is pushed to a level where it is no longer clear how to measure the complexity of an algorithm.

Models like Turing machines or RAM rely on atomic computational steps and thus admit quite obvious cost models for time and space. The λ-calculus instead relies on a single non-atomic operation, β-reduction, for which costs in terms of time and space are far from evident.

Nonetheless, it turns out that the number of β-steps is a reasonable time cost model, i.e.,it is polynomially related to those of Turing machines and RAM. For the special case of *weak evaluation* (i.e., reducing only β-steps that are not under abstractions)—which is used to model functional programming languages—this is a relatively old result due to Blelloch and Greiner [46] (1995). It is only very recently (2014) that the strong case—used in the implementation models of proof assistants—has been solved by Accattoli and Dal Lago [42].

With the recent recruitment of Accattoli, the team's research has expanded in this direction. The topics under investigations are:

1. *Complexity of Abstract Machines*. Bounding and comparing the overhead of different abstract machines for different evaluation schemas (weak/strong call-by-name/value/need λ-calculi) with respect to the cost model. The aim is the development of a complexity-aware theory of the implementation of functional programs.
2. *Reasonable Space Cost Models*. Essentially nothing is known about reasonable space cost models. It is known, however, that environment-based execution model—which are the mainstream technology for functional programs—do not provide an answer. We are exploring the use of the non-standard implementation models provided by Girard's Geometry of Interaction to address this question.

4. Application Domains

4.1. Integrating a model checker and a theorem prover

The goal of combining model checking with inductive and co-inductive theorem is appealing. The strengths of systems in these two different approaches are strikingly different. A model checker is capable of exploring a finite space automatically: such a tool can repeatedly explore all possible cases of a given computational space. On the other hand, a theorem prover might be able to prove abstract properties about a search space. For example, a model checker could attempt to discover whether or not there exists a winning strategy for, say, tic-tac-toe while an inductive theorem prover might be able to prove that if there is a winning strategy for one board then there is a winning strategy for any symmetric version of that board. Of course, the ability to combine proofs from these systems could drastically reduce the amount of state exploration and verification of proof certificates that are needed to prove the existence of winning strategies.

Our first step to providing an integration of model checking and (inductive) theorem proving was the development of a strong logic, that we call $\mathcal{G}$, which extends intuitionistic logic with notions of least and greatest fixed points. We had developed the proof theory of this logic in earlier papers [4] [56]. We have now recently converted the Bedwyr system so that it formally accepts almost all definitions and theorem statements that are accepted by the inductive theorem prover Abella. Thus, these two systems are proving theorems in the same logic and their results can now be shared.

Bedwyr's tabling mechanism has been extended so that its it can make use of previously proved lemmas. For instance, when trying to prove that some board position has a winning strategy, an available stored lemma can now be used to obtain the result if some symmetric board position is already in the table.

Heath and Miller have shown how model checking can be seen as constructing proof in (linear) logic [64]. For more about recent progress on providing checkable proof certificates for model checking, see the web site for Bedwyr http://slimmer.gforge.inria.fr/bedwyr/.

4.2. Implementing trusted proof checkers

Traditionally, theorem provers—whether interactive or automatic—are usually monolithic: if any part of a formal development was to be done in a particular theorem prover, then the whole of it would need to be done in that prover. Increasingly, however, formal systems are being developed to integrate the results returned from several, independent and high-performance, specialized provers: see, for example, the integration of Isabelle with an SMT solver [55] as well as the Why3 and ESC/Java systems.

Within the Parsifal team, we have been working on foundational aspects of this multi-prover integration problem. As we have described above, we have been developing a formal framework for defining the semantics of proof evidence. We have also been working on prototype checkers of proof evidence which are capable of executing such formal definitions. The proof definition language described in the papers [52], [51] is currently given an implementation in the λProlog programming language [74]. This initial implementation will be able to serve as a "reference" proof checker: others who are developing proof evidence definitions will be able to use this reference checker to make sure that they are getting their definitions to do what they expect.

Using λProlog as an implementation language has both good and bad points. The good points are that it is rather simple to confirm that the checker is, in fact, sound. The language also supports a rich set of abstracts which make it impossible to interfere with the code of the checker (no injection attacks are possible). On the negative side, the performance of our λProlog interpreters is lower than that of specially written checkers and kernels.

4.3. Trustworthy implementations of theorem proving techniques

Instead of integrating different provers by exchanging proof evidence and relying on a backend proof-checker, another approach to integration consists in re-implementing the theorem proving techniques as proof-search strategies, on an architecture that guarantees correctness. Focused systems can serve as the basis of such an architecture, identifying points for choice and backtracking, and providing primitives for the exploration of the search space. These form a trusted *Application Programming Interface* that can be used to program and experiment various proof-search heuristics without worrying about correctness. No proof-checking is needed if one trusts the implementation of the API.

This approach has led to the development of the Psyche engine.

Two major research directions are currently being explored, based on the above:

- The first one is about understanding how to deal with quantifiers in presence of one or more theories: On the one hand, traditional techniques for quantified problems, such as *unification* [40] or *quantifier elimination* are usually designed for either the empty theory or very specific theories. On the other hand, the industrial techniques for combining theories (Nelson-Oppen, Shostak, MCSAT [79], [84], [89], [65]) are designed for quantifier-free problems, and quantifiers there are dealt with incomplete *clause instantiation* methods or *trigger*-based techniques [54]. We are working on making the two approaches compatible.

- The above architecture's modular approach raises the question of how its different modules can safely cooperate (in terms of guaranteed correctness), while some of them are trusted and others are not. The issue is particularly acute if some of the techniques are run concurrently and exchange data at unpredictable times. For this we explore new solutions based on Milner's *LCF* [77]. In [60], we argued that our solutions in particular provide a way to fulfil the "Strategy Challenge for SMT-solving" set by De Moura and Passmore [90].

5. Highlights of the Year

5.1. Highlights of the Year

D. Miller gave invited talks at the following two regularly held international meetings.

- TYPES 2016: 22nd International Conference on Types for Proofs and Programs (Novi Sad, Serbia, 23-26 May 2016) and
- Linearity 2016: 4th International Workshop on Linearity (Porto, 25 June 2016).

D. Miller gave invited talks at the following research oriented meetings.

- Workshop on linear logic, mathematics and computer science as part of "LL2016-Linear Logic: interaction, proofs and computation", 7-10 November 2016, Lyon. France.
- Research seminar titled "Interactions between logic, computer science and linguistics: history and philosophy", Université de Lille 3, 15 June 2016.
- CIPPMI (Current issues in the philosophy of practice of mathematics and informatics) Workshop on Proofs, justifications and certificates. 3-4 June 2016, Toulouse, France.

A seminar in honor of the 60th birthday of Professor Miller was held on 15-16 December at Université Paris Diderot-Paris 7 in Paris, France. Several members of the team contributed talks and original research papers.

- Tomer Libal and Marco Volpe, *A general proof certification framework for modal logic*.
- Roberto Blanco and Zakaria Chihani, *An interactive assistant for the definition of proof certificates*. Preprint available as [36].
- Lutz Straßburger, *Combinatorial flows as proof certificates with built-in proof compression*.
- Taus Brock-Nannestad, *Substructural cut elimination*.

B. Accattoli gave an invited talk at the following regularly held international meeting.

- WPTE 2016: 3rd International Workshop on Rewriting Techniques for Program Transformations and Evaluation (Porto, 23 June 2016).

S. Graham-Lengrand gave an invited talk at the following international conference.

- CLAM 2016: 5th Latin American Congress of Mathematicians, thematic session on Logic and Computability (Barranquilla, Colombia, 15th July 2016).

6. New Software and Platforms

6.1. Abella

FUNCTIONAL DESCRIPTION

Abella is an interactive theorem prover for reasoning about computations given as relational specifications. Abella is particuarly well suited for reasoning about binding constructs.

- Participants: Dale Miller, Olivier Savary-Bélanger, Mary Southern, Yuting Wang, Kaustuv Chaudhuri, Matteo Cimini and Gopalan Nadathur
- Partner: Department of Computer Science and Engineering, University of Minnesota
- Contact: Kaustuv Chaudhuri
- URL: http://abella-prover.org/

6.2. Bedwyr

Bedwyr - A proof search approach to model checking
FUNCTIONAL DESCRIPTION

Bedwyr is a generalization of logic programming that allows model checking directly on syntactic expression that possibly contain bindings. This system, written in OCaml, is a direct implementation of two recent advances in the theory of proof search.

It is possible to capture both finite success and finite failure in a sequent calculus. Proof search in such a proof system can capture both may and must behavior in operational semantics. Higher-order abstract syntax is directly supported using term-level lambda-binders, the nabla quantifier, higher-order pattern unification, and explicit substitutions. These features allow reasoning directly on expressions containing bound variables.

The distributed system comes with several example applications, including the finite pi-calculus (operational semantics, bisimulation, trace analyses, and modal logics), the spi-calculus (operational semantics), value-passing CCS, the lambda-calculus, winning strategies for games, and various other model checking problems.

- Participants: Quentin Heath, Roberto Blanco, and Dale Miller
- Contact: Quentin Heath
- URL: http://slimmer.gforge.inria.fr/bedwyr/

6.3. Checkers

Checkers - A proof verifier
KEYWORDS: Proof - Certification - Verification
FUNCTIONAL DESCRIPTION

Checkers is a tool in Lambda-prolog for the certification of proofs. Checkers consists of a kernel which is based on LKF and is based on the notion of ProofCert.

- Participants: Tomer Libal, Giselle Machado Nogueira Reis and Marco Volpe
- Contact: Tomer Libal
- URL: https://github.com/proofcert/checkers

6.4. Psyche

Proof-Search factorY for Collaborative HEuristics
FUNCTIONAL DESCRIPTION

Psyche is a modular platform for automated or interactive theorem proving, programmed in OCaml and built on an architecture (similar to LCF) where a trusted kernel interacts with plugins. The kernel offers an API of proof-search primitives, and plugins are programmed on top of the API to implement search strategies. This architecture is set up for pure logical reasoning as well as for theory-specific reasoning, for various theories.
The major effort in 2016 was the release of version 2.1 that allows the combination of theories, integrating and subsuming both the Nelson-Oppen methodology [79] and the *model constructing satisfiability* (MCSAT) methodology recently proposed by De Moura and Jovanovic [89], [65].

- Participants: Assia Mahboubi, Jean-Marc Notin and Stéphane Graham-Lengrand
- Contact: Stéphane Graham-Lengrand
- URL: http://www.lix.polytechnique.fr/~lengrand/Psyche/

7. New Results

7.1. Linear rewriting systems for Boolean logic

Participant: Lutz Straßburger.

Last year's result on the nonexistence of a complete linear term rewriting system for propositional logic [53] has been generalized and some applications to proof theory have been invesigated. For example, we have found that the medial rule which plays a central role in deep inference systems is canonical in a strong sense: It is minimal, and every rule that reduce contraction to an atomic form is indeed derivable via medial. This is published in [15] (joint work with Anupam Das).

7.2. Non-crossing Tree Realizations of Ordered Degree Sequences

Participant: Lutz Straßburger.

We investigate the enumeration of non-crossing tree realizations of integer sequences, and we consider a special case in four parameters, that can be seen as a four-dimensional tetrahedron that generalizes Pascal's triangle and the Catalan numbers. This work is motivated by the study of ambiguities arising in the parsing of natural language sentences using categorial grammars. This is joint work with Laurent Méhats and published in [31].

7.3. Focusing for Nested Sequents

Participants: Kaustuv Chaudhuri, Sonia Marin, Lutz Straßburger.

Focusing is a general technique for transforming a sequent proof system into one with a syntactic separation of non-deterministic choices without sacrificing completeness. This not only improves proof search, but also has the representational benefit of distilling sequent proofs into synthetic normal forms. We have shown how to apply the focusing technique to nested sequent calculi, a generalization of ordinary sequent calculi to tree-like instead of list-like structures. We thus improve the reach of focusing to the most commonly studied modal logics, the logics of the modal S5 cube. Among our key contributions is a focused cut-elimination theorem for focused nested sequents. This is published in [25].

Then we further extend our results to intuitionistic nested sequents, which can capture all the logics of the intuitionistic S5 cube in a modular fashion. We obtained an internal cut-elimination procedure for the focused system which in turn is used to show its completeness. This is published in [26]

7.4. Combining inference systems: a generalization of Nelson-Oppen and MCSAT

Participant: Stéphane Graham-Lengrand.

Nelson-Oppen [79] and Model-Constructing Satisfiability (MCSAT) [89], [65] are two methodologies that allow the reasoning mechanisms of different theories to collaborate, in order to tackle hybrid problems. While these methodologies are often used and implemented for the practical applications of Automated Reasoning, their rather sophisticated foundations are traditionally explained in terms of model theory. SRI International pioneered some work providing such methodologies with new and more general foundations in terms of *inference systems* [57], closer to proof theory and to Parsifal's research. The more recent MCSAT methodology was not captured, more generally lacked any kind of theorem about the generic combination of arbitrary theories, and was also thought to be incompatible with the Nelson-Oppen approach, so that SMT-solvers are either working with one methodology or the other, unable to get the best of both worlds.

In 2016 we designed a combination methodology, based on *inference systems*, that supersedes both Nelson-Oppen and MCSAT [34]. We showed its soundness and completeness, and identified for this the properties that the theories to combine are required to satisfy. This generalized MCSAT with the generic combination mechanism that it lacked, and showed that it is perfectly compatible with the Nelson-Oppen methodology, which can now cohabit within the same solver.

7.5. Linear lambda terms as invariants of rooted trivalent maps

Participant: Noam Zeilberger.

Recent studies of the combinatorics of linear lambda calculus have uncovered some unexpected connections to the old and well-developed theory of graphs embedded on surfaces (also known as "maps") [47], [87], [88]. In [19], we aimed to give a simple and conceptual account for one of these connections, namely the correspondence (originally described by Bodini, Gardy, and Jacquot [47]) between α-equivalence classes of closed linear lambda terms and isomorphism classes of rooted trivalent maps on compact oriented surfaces without boundary. One immediate application of this new account was a characterization of trivalent maps which are *bridgeless* (in the graph-theoretic sense of having no disconnecting edge) as linear lambda terms with no closed proper subterms. In turn, this lead to a surprising but natural reformulation of the Four Color Theorem as a statement about typing in lambda calculus.

7.6. A bifibrational reconstruction of Lawvere's presheaf hyperdoctrine

Participant: Noam Zeilberger.

In joint work with Paul-André Melliès, we have been investigating the categorical semantics of type refinement systems, which are type systems built "on top of" a typed programming language to specify and verify more precise properties of programs. The fibrational view of type refinement we have been developing (cf. [72]) is closely related to the categorical perspective on first-order logic introduced by Lawvere [66], but with some important conceptual and technical differences that provide an opportunity for reflection. For example, Lawvere's axiomatization of first-order logic (his theory of so-called "hyperdoctrines") was based on the idea that existential and universal quantification can be described respectively as left and right adjoints to the operation of substitution, this giving rise to a family of *adjoint triples* $\Sigma_f \dashv \mathcal{P}_f \dashv \Pi_f$ (one such triple for every function $f : A \to B$). On the other hand, a bifibration only induces a family of *adjoint pairs* $\mathsf{push}_f \dashv \mathsf{pull}_f$ (again, one such pair for every $f : A \to B$). In [33], we resolved this and other apparent mismatches by applying ideas inspired by the semantics of linear logic and the shift from the cartesian closed category $\mathbf{Set}$ to the symmetric monoidal closed category $\mathbf{Rel}$. Two other applications of our analysis include an axiomatic treatment of *directed* equality predicates (which can be modelled as "hom" presheaves, realizing an early vision of Lawvere), as well as a simple calculus of string diagrams that is highly reminiscent of C. S. Peirce's "existential graphs" for predicate logic.

7.7. Towards a link between CPS and focusing

Participant: Matthias Puech.

Continuation-passing style translations make a functional program more explicit by sequentializing its computations and reifying its control. They have been used as an intermediate language in many compilers. They are also understood as classical-to-intuitionistic proof embedding (so-called double negation translations). Matthias Puech studied a novel correspondence between CPS and focusing: to each CPS transform corresponds a focused proof system that is identifiable as a particular polarization of classical statements. Since, after Miller's and others work, we know the full design space of focused sequent calculi, we expect to understand the full design space of CPS translation.

The first step of this goal is to study the syntax and typing of variants of the CPS translation. Puech designed and implemented in OCaml a compacting, optimizing CPS translation, while using OCaml's type system to verify that it maps well-typed terms to well-typed terms in a tightly restricted syntactical form (the "typeful" approach to formalization) [82]. The resulting type system is in Curry-Howard isomorphism with a weakly focused proof system: LJQ.

7.8. Proof Checking and Logic Programming

Participants: Roberto Blanco, Tomer Libal, Dale Miller, Marco Volpe.

In a world where trusting software systems is increasingly important, formal methods and formal proofs can help provide some basis for trust. Proof checking can help to reduce the size of the *trusted base* since we do not need to trust an entire theorem prover: instead, we only need to trust a (smaller and simpler) proof checker. Many approaches to building proof checkers require embedding within them a full programming language. In most modern proof checkers and theorem provers, that programming language is a functional programming language, often a variant of ML. In fact, aspects of ML (e.g., strong typing, abstract data types, and higher-order programming) were designed to make ML a trustworthy "meta-language" for checking proofs. While there is considerable overlap between logic programming and proof checking (e.g., both benefit from unification, backtracking search, efficient term structures, etc), the discipline of logic programming has, in fact, played a minor role in the history of proof checking. Miller has been pushing the argument that logic programming can have a major role in the future of this important topic [18]. Many aspects of the ProofCert project are based on this perspective that logic programming techniques and methods can have significant utility within proof checking. This perspective stands in constrast to the work on the Dedukti proof checking framework [44] where functional programming principles are employed for proof checking.

7.9. Proof Certificates for First-Order Equational Logic

Participants: Dale Miller, Zakaria Chihani.

The kinds of inference rules and decision procedures that one writes for proofs involving equality and rewriting are rather different from proofs that one might write in first-order logic using, say, sequent calculus or natural deduction. For example, equational logic proofs are often chains of replacements or applications of oriented rewriting and normal forms. In contrast, proofs involving logical connectives are trees of introduction and elimination rules. Chihani and Miller have shown [13] how it is possible to check various equality-based proof systems with a programmable proof checker (the *kernel* checker) for first-order logic. That proof checker's design is based on the implementation of *focused proof search* and on making calls to (user-supplied) *clerks and experts* predicates that are tied to the two phases found in focused proofs. This particular design is based on the work of Chihani, Miller, and Renaud [14].

The specification of these clerks and experts provide a formal definition of the structure of proof evidence and they work just as well in the equational setting as in the logic setting where this scheme for proof checking was originally developed. Additionally, executing such a formal definition on top of a kernel provides an actual proof checker that can also do a degree of proof reconstruction. A number of rewriting based proofs have been defined and checked in this manner.

7.10. Extended Pattern Unification

Participants: Tomer Libal, Dale Miller.

Unification is a central operation in the construction of a range of computational logic systems based on first-order and higher-order logics. First-order unification has a number of properties that dominates the way it is incorporated within such systems. In particular, first-order unification is decidable, unary, and can be performed on untyped term structures. None of these three properties hold for full higher-order unification: unification is undecidable, unifiers can be incomparable, and term-level typing can dominate the search for unifiers. The so-called *pattern* subset of higher-order unification was designed to be a small extension to first-order unification that respected the basic laws governing λ-binding (the equalities of α, β, and η-conversion) but which also satisfied those three properties. While the pattern fragment of higher-order unification has been popular in various implemented systems and in various theoretical considerations, it is too weak for a number of applications. Libal and Miller [28] have defined an extension of pattern unification that is motivated by some existing applications and which satisfies these three properties. The main idea behind their extension is that the arguments to a higher-order, free variable can be more than just distinct bound variables: they can also be terms constructed from (sufficient numbers of) such variables using term constructors and where no argument is a subterm of any other argument. This extension to pattern unification satisfies the three properties mentioned above. R. Blanco is currently adding this extended unification to the Abella theorem prover.

7.11. Focused proofs for modal logics

Participants: Tomer Libal, Sonia Marin, Dale Miller, Marco Volpe.

Several deductive formalisms (e.g., sequent, nested sequent, labeled sequent, hypersequent calculi) have been used in the literature for the treatment of modal logics, and some connections between these formalisms are already known. Marin, Miller, and Volpe [30] have propose a general framework, which is based on a focused version of the labeled sequent calculus by Negri [78], augmented with some parametric devices allowing to restrict the set of proofs. By properly defining such restrictions and by choosing an appropriate polarization of formulas, one can obtain different, concrete proof systems for the modal logic K and for its extensions by means of geometric axioms. The expressiveness of the labeled approach and the control mechanisms of focusing allow a clean emulation of a range of existing formalisms and proof systems for modal logic. These results make it possible to write Foundational Proof Certificate definitions of common modal logic proof systems.

7.12. Preserving differential privacy under finite-precision semantics

Participant: Dale Miller.

(Joint work with Ivan Gazeau and Catuscia Palamidessi). The approximation introduced by finite-precision representation of continuous data can induce arbitrarily large information leaks even when the computation using exact semantics is secure. Such leakage can thus undermine design efforts aimed at protecting sensitive information. Gazeau, Miller, and Palamidessi [16] have applied differential privacy—an approach to privacy that emerged from the area of statistical databases—to this problem. In their approach, privacy is protected by the addition of noise to a true (private) value. To date, this approach to privacy has been proved correct only in the ideal case in which computations are made using an idealized, infinite-precision semantics. An analysis of implementation levels, where the semantics is necessarily finite-precision, i.e. the representation of real numbers and the operations on them are rounded according to some level of precision. In general there are violations of the differential privacy property but a limited (but, arguably, totally acceptable) variant of the property can be used insteand, under only a minor degradation of the privacy level. In fact, two cases of noise-generating distributions can be employed: the standard Laplacian mechanism commonly used in differential privacy, and a bivariate version of the Laplacian recently introduced in the setting of privacy-aware geolocation.

7.13. Certification of Prefixed Tableau Proofs for Modal Logic

Participants: Tomer Libal, Marco Volpe.

This work [29] describes the theory and implementation of a proof checker for tableau theorem provers for modal logics. The tool supports proofs in both the traditional tableau format as well as the free variable variant. The implentation can be found at https://github.com/proofcert/checkers under the gandalf2016 branch.

7.14. Towards a Substitution Tree Based Index for Higher-order Resolution Theorem Provers

Participant: Tomer Libal.

First-order resolution theorem provers depend on efficient data structures for redundancy elimination. These data structures do not exist for higher-order resolution theorem provers. In [32] we discuss a new approach to this problem. (Joint work with Alexander Steen).

7.15. Open Call-by-Value

Participant: Beniamino Accattoli.

Functional programming languages are often based on the call-by-value λ-calculus, whose elegant theory relies on weak evaluation and closed terms, that are natural hypotheses in the study of programming languages. To model proof assistants, however, strong evaluation and open terms are required, and it is well known that the operational semantics of call-by-value becomes problematic in this case. In this joint work with Giulio Guerrieri we studied the intermediate setting—that we call Open Call-by-Value—of weak evaluation with open terms, on top of which Gregoire and Leroy designed the abstract machine of Coq. Various calculi for Open Call-by-Value already exist, each one with its pros and cons. We did a detailed comparative study of the operational semantics of four of them, coming from different areas such as the study of abstract machines, denotational semantics, linear logic proof nets, and sequent calculus. We showed that these calculi are all equivalent from a termination point of view, justifying the slogan Open Call-by-Value. The work has been published in the proceedings of the international conference APLAS 2016 [22].

7.16. A Reasonable Abstract Machine for the Strong λ-Calculus

Participant: Beniamino Accattoli.

We provided a new proof that the strong λ-calculus is a reasonable computational model. The original proof is by B. Accattoli and H. Dal Lago uses a calculus with explicit substitutions while the new one relies on a new sophisticated abstract machine, the Useful MAM. The work has been published in the proceeding of the international conference WoLLIC 2016 [21].

7.17. Space-efficient Acyclicity Constraints

Participant: Taus Brock-Nannestad.

Acyclicity constraints can be used to encode a large variety of useful constraints on graphs. The basic constraint itself can be encoded in terms of simpler constraints (e.g. integer linear constraints) in a straightforward and intuitive way, associating to each vertex of the (fixed) input graph a variable with domain linear in the size of the graph. For large graphs, this quickly becomes inefficient.

In [24], we show that in the case of planar graphs, a more efficient encoding (using a two-valued variable per vertex) is possible.

7.18. Exp-log normal form of types and the axioms for η-equality of the λ-calculus with sums

Participant: Danko Ilik.

In the presence of sum types, the λ-calculus has but one implemented (and incomplete) heuristic for deciding $\beta\eta$-equality of terms, in spite of a dozen of meta-theoretic works showing that the equality is decidable.

In the work discussed here, we first used the exp-log decomposition of the arrow type—inspired from the analytic transformation $a^b = \exp(b \times \log a)$—to obtain a type normal form for the type languages $\{\rightarrow, \times, +\}$. We then made a quotient of the $\beta\eta$-equality of terms modulo the terms coerced into their representation at the exp-log normal form of their type. This allows to obtain a *simplification* of the so far standard axioms for $\beta\eta$-equality.

Moreover, we provided a Coq implementation of a heuristic decision procedure for this equality. Although a heuristic, this implementation manages to tackle examples of equal terms that need a complex program analysis in the only previously implemented heuristic of Vincent Balat.

This work is described in a paper accepted for presentation at POPL 2017, [27].

7.19. Invertible-rule-free sequent calculi and an intuitionistic arithmetical hierarchy

Participants: Taus Brock-Nannestad, Danko Ilik.

In sequent calculi, proof rules can be divided into two groups: invertible (asynchronous) proof rules and non-invertible (synchronous) proof rules. Even in focusing sequent calculi the two groups of rules are present, albeit grouped together in synthetic rules (we speak of the synchronous and asynchronous phase).

In this work, we used the exp-log decomposition (described above) in the context of logic in order to obtain a version of sequent caclulus which contains synchronous rules only, a first such formalism for intuitionistic logic.

We extended the picture from the setting of propositional to the one of first-order intuitionistic logic, where the exp-log decomposition provided us with an intuitionistic hierarchy of formulas analogous to the classical arithmetical hierachy; although the classical arithmetical hierarchy exists since the 1920s, a correspondingly versatile notion for intuitionistic logic has been elusive up to this day.

This work is described in the manuscript [37], submitted to an academic journal.

8. Partnerships and Cooperations

8.1. European Initiatives

8.1.1. FP7 & H2020 Projects

8.1.1.1. Proofcert

Title: ProofCert: Broad Spectrum Proof Certificates

Programm: FP7

Type: ERC

Duration: January 2012 - December 2016

Coordinator: Inria

Inria contact: Dale Miller

There is little hope that the world will know secure software if we cannot make greater strides in the practice of formal methods: hardware and software devices with errors are routinely turned against their users. The ProofCert proposal aims at building a foundation that will allow a broad spectrum of formal methods—ranging from automatic model checkers to interactive theorem provers—to work together to establish formal properties of computer systems. This project starts with a wonderful gift to us from decades of work by logicians and proof theorist: their efforts on logic and proof has given us a universally accepted means of communicating proofs between people and computer systems. Logic can be used to state desirable security and correctness properties of software and hardware systems and proofs are uncontroversial evidence that statements are, in fact, true. The current state-of-the-art of formal methods used in academics and industry shows, however, that the notion of logic and proof is severely fractured: there is little or no communication between any two such systems. Thus any efforts on computer system correctness is needlessly repeated many time in the many different systems: sometimes this work is even redone when a given prover is upgraded. In ProofCert, we will build on the bedrock of decades of research into logic and proof theory the notion of proof certificates. Such certificates will allow for a complete reshaping of the way that formal methods are employed. Given the infrastructure and tools envisioned in this proposal, the world of formal methods will become as dynamic and responsive as the world of computer viruses and hackers has become.

8.1.2. Collaborations in European Programs, Except FP7 & H2020

8.1.2.1. FISP: ANR blanc International

Participants: Kaustuv Chaudhuri, François Lamarche, Sonia Marin, Dale Miller, Lutz Straßburger.

Title: The Fine Structure of Formal Proof Systems and their Computational Interpretations

Duration: 01/01/2016 – 31/12/2018

Partners:

University Paris VII, PPS (PI: Michel Parigot)

Inria Saclay–IdF, EPI Parsifal (PI: Lutz Straßburger)

University of Innsbruck, Computational Logic Group (PI: Georg Moser)

Vienna University of Technology, Theory and Logic Group (PI: Matthias Baaz)

Total funding by the ANR: 316 805 EUR

The FISP project is part of a long-term, ambitious project whose objective is to apply the powerful and promising techniques from structural proof theory to central problems in computer science for which they have not been used before, especially the understanding of the computational content of proofs, the extraction of programs from proofs and the logical control of refined computational operations. So far, the work done in the area of computational interpretations of logical systems is mainly based on the seminal work of Gentzen, who in the mid-thirties introduced the sequent calculus and natural deduction, along with the cut-elimination procedure. But that approach shows its limits when it comes to computational interpretations of classical logic or the modelling of parallel computing. The aim of our project, based on the complementary skills of the teams, is to overcome these limits. For instance, deep inference provides new properties, namely full symmetry and atomicity, which were not available until recently and opened new possibilities at the computing level, in the era of parallel and distributed computing.

8.1.2.2. COCA HOLA: ANR JCJC Project

Participant: Beniamino Accattoli.

Title: COst model for Complexity Analyses of Higher-Order programming LAnguages.

Collaborators: Ugo Dal Lago (University of Bologna & Inria), Delia Kesner (Paris Diderot University), Damiano Mazza (CNRS & Paris 13 University), Claudio Sacerdoti Coen (University of Bologna).

Duration: 01/10/2016 – 31/09/2019

Total funding by the ANR: 155 280 EUR

The COCA HOLA project aims at developing complexity analyses of higher-order computations, i.e. that approach to computation where the inputs and outputs of a program are not simply numbers, strings, or compound data-types, but programs themselves. The focus is not on analysing fixed programs, but whole programming languages. The aim is the identification of adequate units of measurement for time and space, i.e. what are called reasonable cost models. The problem is non-trivial because the evaluation of higher-order languages is defined abstractly, via high-level operations, leaving the implementation unspecified. Concretely, the project will analyse different implementation schemes, measuring precisely their computational complexity with respect to the number of high-level operations, and eventually develop more efficient new ones. The goal is to obtain a complexity-aware theory of implementations of higher-order languages with both theoretical and practical downfalls.

The projects stems from recent advances on the theory of time cost models for the lambda-calculus, the computational model behind the higher-order approach, obtained by the principal investigator and his collaborators (who are included in the project).

COCA HOLA will span over three years and is organised around three work packages, essentially:

1. extending the current results to encompass realistic languages;
2. explore the gap between positive and negative results in the literature;
3. use ideas from linear logic to explore space cost models, about which almost nothing is known.

8.2. International Initiatives

8.2.1. Participation in Other International Programs

8.2.1.1. PHC Amadeus: Analytic Calculi for Modal Logics

Participants: Kaustuv Chaudhuri, Sonia Marin, Giselle Reis, Lutz Straßburger.

Title: Analytic Calculi for Modal Logics

Duration: 01/01/2016 – 31/12/2017

Austrian Partner: TU Wien, Institute for Computer Science (Department III)

Modal logics are obtained from propositional logics by adding modalities $\Box$ and $\diamond$, meaning necessity and possibility. Originally studied by philosophers in order to reason about knowledge and belief, modal logics have nowadays many applications in computer science. Well known examples are epistemic logics, which allow to formally reason about the knowledge of independently acting and interacting agents, temporal logics, which allow to reason about temporal properties of processes, and authentication logics, which are used to formally reason about authentication protocols.

The purpose of this project is to develop a proof theory for variants of modal logic that have applications in modern computer science but that have been neglected by traditional proof theory so far.

8.3. International Research Visitors

8.3.1. Visits of International Scientists

Professor Chuck Liang (from Hofstra University, NY, USA) visited the team from 5 June to 25 June 2016 in order to continue his collaborations with team members on basic questions of proof theory. In particular, he worked with Miller on identifying possible means to allow classical and intuitionistic logic to be mixed in a common proof system. Miller is exploring how the resulting ideas might be able to reorganize the notion of kernel logic used within the ProofCert project.

8.3.1.1. Internships

Ameni Chtourou was an intern funded by ProofCert during May, June, and July 2016. She was advised by Accattoli and worked with using the Abella theorem prover to formalize connections various connections between λ-term evaluation and abstract machine models.

8.3.2. Visits to International Teams

8.3.2.1. Research Stays Abroad

Stéphane Graham-Lengrand spent 8 months, from January 2016 to August 2016, at SRI International, Computer Science Lab. This visit developed a collaboration with N. Shankar, MP Bonacina, D. Jovanovic, and Martin Schaeff on new algorithms and new architectures for automated and interactive theorem proving, as well as on new programme verification techniques.

9. Dissemination

9.1. Promoting Scientific Activities

9.1.1. Scientific Events Organisation

9.1.1.1. Member of the Organizing Committees

D. Miller was on the Steering Committee for the FSCD series of International Conference on Formal Structures for Computation and Deduction.

D. Miller was a member of the jury for selecting the 2016 Ackermann Award (the EACSL award for outstanding doctoral dissertation in the field of Logic in Computer Science).

D. Miller was an ex officio member of the Executive Committee of the ACM Special Interest Group on Logic and Compuation (SIGLOG), from April 2014 to June 2016. He was also a member of the SIGLOG advisory board, starting November 2015.

9.1.2. Scientific Events Selection

9.1.2.1. Member of the Conference Program Committees

D. Miller was on the Program Committee of the following meetings.

FSCD'16: First International Conference on Formal Structures for Computation and Deduction, Porto, Portugal, 22-26 June.

IJCAR 2016: International Joint Conference on Automated Reasoning, Coimbra, Portugal, 27 June - 2 July.

CPP 2016, Fifth International Conference on Certified Programs and Proofs, 18-19 January, Saint Petersburg, Florida.

B. Accattoli was one of the two Program Committee chairs of the 5th International Workshop on Confluence (IWC 2016).

N. Zeilberger served on the program committee for workshops Computational Logic and Applications (CLA 2016) and Off the Beaten Track (OBT 2016).

N. Zeilberger served on external review committee for POPL 2017

L. Straßburger was on the Program Committee for LICS 2016.

9.1.2.2. Reviewer

D. Miller was a reviewer for CONCUR 2016: the International Conference on Concurrency Theory.

B. Accattoli was a reviewer for the international conferences ICTAC 2016, FSCD 2016, LICS 2016 (twice), FOSSACS 2017.

S. Graham-Lengrand was a reviewer for the international conferences FSCD 2016 (twice), LICS 2016 (three times), Concur 2016, VSTTE 2017, HATT 2017, FSTTCS 2017.

L. Straßburger was a reviewer for the international conferences LICS 2016 (8 times), FLOPS 2016.

M. Volpe was a reviewer for the international conferences IJCAR 2016 and CSL 2016.

H. Steele was a reviewer for the internal conference LICS 2016.

9.1.3. Journal

9.1.3.1. Member of the editorial boards

D. Miller is on the editorial board of the following journals: *ACM Transactions on Computational Logic*, *Journal of Automated Reasoning* (Springer), *Theory and Practice of Logic Programming* (Cambridge University Press), and *Journal of Applied Logic* (Elsevier).

9.1.3.2. Reviewer - Reviewing Activities

S. Graham-Lengrand has been a reviewer for the journals *Fundamenta Informaticae*, *Transactions on Computational Logic*, *Journal of Logic and Computation*, *Logical Methods in Computer Science*, *Journal of Automated Reasoning*.

Danko Ilik was a reviewer for Mathematical Reviews and Zentralblatt MATH.

F. Lamarche was a reviewer for *Mathematical Structures in Computer Science*.

Lutz Straßburger was a reviewer for the journals *Theoretical Computer Science* and *Logical Methods in Computer Science*.

Marco Volpe was a reviewer for the journal *Annals of Mathematics and Artificial Intelligence*.

Beniamino Accattoli was a reviewer for the journals *Theoretical Computer Science* and *Logical Methods in Computer Science*.

9.1.4. Invited Talks

D. Miller was an invited speaker at the following conferences and workshops.

Workshop on linear logic, mathematics and computer science as part of "LL2016-Linear Logic: interaction, proofs and computation", 7-10 November 2016, Lyon. France.

Linearity 2016. Porto, 25 June 2016.

CIPPMI (Current issues in the philosophy of practice of mathematics and informatics) Workshop on Proofs, justifications and certificates. 3-4 June 2016, Toulouse, France.

TYPES 2016: 22nd International Conference on Types for Proofs and Programs. Novi Sad, Serbia, 23-26 May 2016.

D. Miller was an invited speaker at the research seminar titled "Interactions between logic, computer science and linguistics: history and philosophy", Université de Lille 3, 15 June 2016.

D. Miller was an invited speaker at the ACADIA research centre, Ca' Foscari University, Venice, 27 April 2016.

B. Accattoli was invited speaker at WPTE 2016: 3rd International Workshop on Rewriting Techniques for Program Transformations and Evaluation (Porto, 23 June 2016).

S. Graham-Lengrand gave an invited talk at CLAM 2016: 5th Latin American Congress of Mathematicians, thematic session on Logic and Computability (Barranquilla, Colombia, 15th July 2016).

N. Zeilberger was an invited lecturer at OPLSS 2016: Oregon Programming Languages Summer School on Types, Logic, Semantics, and Verification.

9.1.5. Leadership within the Scientific Community

D. Miller was a member of the ACM SIGLOG Advisory Board, the LICS Organizing Board, the CPP Steering Committee, and the ACM SIGLOG Executive Committee Nominating Committee.

S. Graham-Lengrand is the head of the National Workgroup on "Logic, Algebra, and Computation", within the Informatique Mathématique section of CNRS.

9.1.6. Research Administration

L. Straßburger serves on the "commission développement technologique (CDT)" for Inria Saclay–Île-de-France since June 2012

9.2. Teaching - Supervision - Juries

9.2.1. Teaching

Master: D. Miller, "*MPRI 2-1: Logique linéaire et paradigmes logiques du calcul*", 12 hours, M2, Master Parisien de Recherche en Informatique, France.

Licence: S. Graham-Lengrand, "*INF412: Fondements de l'Informatique: Logique, Modèles, Calcul*", 32 hours eq. TD, L3, École Polytechnique, France.

Master: S. Graham-Lengrand, "*INF551: Computational Logic*", 45 hours eq. TD, M1, École Polytechnique, France.

Master: S. Graham-Lengrand, "*MPRI 2-1: Logique linéaire et paradigmes logiques du calcul*", 6 hours, M2, Master Parisien de Recherche en Informatique, France.

Undergraduate: K. Chaudhuri, R. Blanco, M. Volpe, G. Reis, T. Libal all taught or tutored exercises for first and second year undergrad courses, mostly at École Polytechnique.

9.2.2. Supervision

PhD in progress: Sonia Marin, 1 Nov 2014, supervised by L. Straßburger and D. Miller

PhD in progress: Roberto Blanco, Ulysse Gérard, and Quentin Heath, supervised by D. Miller

PhD in progress: François Thiré (since 1st October 2016), supervised by S. Graham-Lengrand (joint with G. Dowek)

9.2.3. Juries

Miller was a reporter for the PhD juries of Raphaël Cauderlier (CNAM, 10 October 2016) and Gabriel Scherer (Université Paris-Diderot, 30 March 2016).

Graham-Lengrand was a reporter for the PhD juries of Pierre Halmagrand (CNAM, 10 December 2016).

10. Bibliography

Major publications by the team in recent years

[1] D. Baelde, K. Chaudhuri, A. Gacek, D. Miller, G. Nadathur, A. Tiu, Y. Wang. *Abella: A System for Reasoning about Relational Specifications*, in "Journal of Formalized Reasoning", 2014, vol. 7, n^{o} 2, pp. 1-89 [*DOI :* 10.6092/issn.1972-5787/4650], https://hal.inria.fr/hal-01102709

[2] K. Chaudhuri, N. Guenot, L. Strassburger. *The Focused Calculus of Structures*, in "Computer Science Logic: 20th Annual Conference of the EACSL", Leibniz International Proceedings in Informatics (LIPIcs), Schloss Dagstuhl–Leibniz-Zentrum für Informatik, September 2011, pp. 159–173, http://drops.dagstuhl.de/opus/frontdoor.php?source_opus=3229

[3] M. Farooque, S. Graham-Lengrand, A. Mahboubi. *A bisimulation between DPLL(T) and a proof-search strategy for the focused sequent calculus*, in "Proceedings of the 2013 International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice (LFMTP 2013)", A. Momigliano, B. Pientka, R. Pollack (editors), ACM Press, September 2013 [*DOI :* 10.1145/2503887.2503892]

[4] A. Gacek, D. Miller, G. Nadathur. *Nominal abstraction*, in "Information and Computation", 2011, vol. 209, n^{o} 1, pp. 48–73, http://arxiv.org/abs/0908.1390

[5] A. Guglielmi, T. Gundersen, L. Strassburger. *Breaking Paths in Atomic Flows for Classical Logic*, in "Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science (LICS 2010)", Edinburgh, United Kingdom, July 2010, pp. 284–293 [*DOI :* 10.1109/LICS.2010.12], http://www.lix.polytechnique.fr/~lutz/papers/AFII.pdf

[6] F. Lamarche, L. Strassburger. *Naming Proofs in Classical Propositional Logic*, in "Typed Lambda Calculi and Applications, TLCA 2005", P. Urzyczyn (editor), LNCS, Springer-Verlag, 2005, vol. 3461, pp. 246–261

[7] C. Liang, D. Miller. *Focusing and Polarization in Linear, Intuitionistic, and Classical Logics*, in "Theoretical Computer Science", 2009, vol. 410, n^{o} 46, pp. 4747–4768

[8] C. Liang, D. Miller. *A Focused Approach to Combining Logics*, in "Annals of Pure and Appl. Logic", 2011, vol. 162, n^{o} 9, pp. 679–697 [*DOI :* 10.1016/j.apal.2011.01.012], http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/lku.pdf

[9] D. Miller. *A proposal for broad spectrum proof certificates*, in "CPP: First International Conference on Certified Programs and Proofs", J.-P. Jouannaud, Z. Shao (editors), LNCS, 2011, vol. 7086, pp. 54–69, http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/cpp11.pdf

[10] L. STRASSBURGER. *On the Axiomatisation of Boolean Categories with and without Medial*, in "Theory and Applications of Categories", 2007, vol. 18, n^o^ 18, pp. 536–601, http://arxiv.org/abs/cs.LO/0512086

[11] A. TIU, D. MILLER. *Proof Search Specifications of Bisimulation and Modal Logics for the π-calculus*, in "ACM Trans. on Computational Logic", 2010, vol. 11, n^{o} 2, http://arxiv.org/abs/0805.2785

Publications of the year

Articles in International Peer-Reviewed Journals

[12] B. ACCATTOLI, U. DAL LAGO. *(Leftmost-outermost) beta reduction is invariant, indeed*, in "Logical Methods in Computer Science", 2016 [*DOI :* 10.2168/LMCS-12(1:4)2016], https://hal.inria.fr/hal-01337712

[13] Z. CHIHANI, D. MILLER. *Proof Certificates for Equality Reasoning*, in "Electronic Notes in Theoretical Computer Science", 2016, vol. 323, pp. 93 - 108 [*DOI :* 10.1016/J.ENTCS.2016.06.007], https://hal.inria.fr/hal-01390919

[14] Z. CHIHANI, D. MILLER, F. RENAUD. *A Semantic Framework for Proof Evidence*, in "Journal of Automated Reasoning", 2016 [*DOI :* 10.1007/S10817-016-9380-6], https://hal.inria.fr/hal-01390912

[15] A. DAS, L. STRASSBURGER. *On linear rewriting systems for Boolean logic and some applications to proof theory*, in "Logical Methods in Computer Science", December 2016, https://hal.inria.fr/hal-01422611

[16] I. GAZEAU, D. MILLER, C. PALAMIDESSI. *Preserving differential privacy under finite-precision semantics*, in "Journal of Theoretical Computer Science (TCS)", 2016, https://hal.inria.fr/hal-01390927

[17] W. HEIJLTJES, L. STRASSBURGER. *Proof nets and semi-star-autonomous categories*, in "Mathematical Structures in Computer Science", June 2016, vol. 26, n^{o} 5, pp. 789-828 [*DOI :* 10.1017/S0960129514000395], https://hal.inria.fr/hal-01417643

[18] D. MILLER. *Proof Checking and Logic Programming*, in "Formal Aspects of Computing", 2016 [*DOI :* 10.1007/S00165-016-0393-Z], https://hal.inria.fr/hal-01390901

[19] N. ZEILBERGER. *Linear lambda terms as invariants of rooted trivalent maps*, in "Journal of Functional Programming", November 2016, vol. 26 [*DOI :* 10.1017/S095679681600023X], https://hal.archives-ouvertes.fr/hal-01247757

Invited Conferences

[20] B. ACCATTOLI. *The Complexity of Abstract Machines*, in "Third International Workshop on Rewriting Techniques for Program Transformations and Evaluation (WPTE 2016)", Porto, Portugal, June 2016, vol. 235, pp. 1 - 15 [*DOI :* 10.4204/EPTCS.235.1], https://hal.inria.fr/hal-01425560

International Conferences with Proceedings

[21] B. ACCATTOLI. *The Useful MAM, a Reasonable Implementation of the Strong λ-Calculus*, in "23rd International Workshop on Logic, Language, Information, and Computation (WoLLIC 2016)", Puebla, Mexico, August 2016, pp. 1 - 21 [*DOI :* 10.1007/978-3-662-52921-8_1], https://hal.inria.fr/hal-01425534

[22] B. ACCATTOLI, G. GUERRIERI. *Open Call-by-Value*, in "14th Asian Symposium on Programming Languages and Systems (APLAS)", Hanoi, Vietnam, November 2016, pp. 206 - 226 [*DOI :* 10.1007/978-3-319-47958-3_12], https://hal.inria.fr/hal-01425465

[23] S. AZAIEZ, D. DOLIGEZ, M. LEMERRE, T. LIBAL, S. MERZ. *Proving Determinacy of the PharOS Real-Time Operating System*, in "Abstract State Machines, Alloy, B, TLA, VDM, and Z - 5th International Conference, ABZ 2016", Linz, Austria, M. J. BUTLER, K.-D. SCHEWE, A. MASHKOOR, M. BIRÓ (editors), LNCS - Lecture Notes in Computer Science, Springer, May 2016, vol. 9675, pp. 70-85 [*DOI :* 10.1007/978-3-319-33600-8_4], https://hal.inria.fr/hal-01322335

[24] T. BROCK-NANNESTAD. *Space-efficient Planar Acyclicity Constraints - A Declarative Pearl*, in "FLOPS 2016 - 13th International Symposium on Functional and Logic Programming", Kochi, Japan, March 2016, https://hal.inria.fr/hal-01426753

[25] K. CHAUDHURI, S. MARIN, L. STRASSBURGER. *Focused and Synthetic Nested Sequents*, in "FOSSACS 2016 - 19th International Conference Foundations of Software Science and Computation Structures", Eindhoven, Netherlands, B. JACOBS, C. LÖDING (editors), LNCS - Lecture Notes in Computer Science, Springer, April 2016, vol. 9634, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016 [*DOI :* 10.1007/978-3-662-49630-5_23], https://hal.inria.fr/hal-01417618

[26] K. CHAUDHURI, S. MARIN, L. STRASSBURGER. *Modular Focused Proof Systems for Intuitionistic Modal Logics*, in "FSCD 2016 - 1st International Conference on Formal Structures for Computation and Deduction", Porto, Portugal, June 2016 [*DOI :* 10.4230/LIPIcs.FSCD.2016.16], https://hal.inria.fr/hal-01417603

[27] D. ILIK. *On the exp-log normal form of types: Decomposing extensional equality and representing terms compactly*, in "Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages", Paris, France, January 2017, pp. 387-399, https://hal.inria.fr/hal-01167162

[28] T. LIBAL, D. MILLER. *Functions-as-constructors Higher-order Unification*, in "1st International Conference on Formal Structures for Computation and Deduction (FSCD 2016)", Porto, Portugal, Delia Kesner and Brigitte Pientka, June 2016, pp. 1 - 17 [*DOI :* 10.4230/LIPIcs.FSCD.2016.26], https://hal.inria.fr/hal-01379683

[29] T. LIBAL, M. VOLPE. *Certification of Prefixed Tableau Proofs for Modal Logic*, in "the Seventh International Symposium on Games, Automata, Logics and Formal Verification (GandALF 2016)", Catania, Italy, September 2016, vol. 226, pp. 257 - 271 [*DOI :* 10.4204/EPTCS.226.18], https://hal.archives-ouvertes.fr/hal-01379625

[30] S. MARIN, D. MILLER, M. VOLPE. *A focused framework for emulating modal proof systems*, in "11th conference on "Advances in Modal Logic"", Budapest, Hungary, L. BEKLEMISHEV, S. DEMRI, A. MÁTÉ (editors), Advances in Modal Logic, College Publications, August 2016, n^{o} 11, pp. 469-488, https://hal.archives-ouvertes.fr/hal-01379624

[31] L. MEHATS, L. STRASSBURGER. *Non-crossing Tree Realizations of Ordered Degree Sequences*, in "LACL 2016 - 9th International Conference Logical Aspects of Computational Linguistics - Celebrating 20 Years of LACL (1996–2016)", Nancy, France, M. AMBLARD, P. DE GROOTE, S. POGODALLA, C. RETORÉ (editors), LNCS - Lecture Notes in Computer Science, Springer, December 2016, vol. 10054 [*DOI :* 10.1007/978-3-662-53826-5_13], https://hal.inria.fr/hal-01417567

Conferences without Proceedings

[32] T. LIBAL, A. STEEN. *Towards a Substitution Tree Based Index for Higher-order Resolution Theorem Provers*, in "5th Workshop on Practical Aspects of Automated Reasoning", Coimbra, Portugal, Proceedings of the 5th Workshop on Practical Aspects of Automated Reasoning co-located with International Joint Conference on Automated Reasoning (IJCAR 2016), Coimbra, Portugal, July 2nd, 2016, July 2016, https://hal.archives-ouvertes.fr/hal-01424749

[33] P.-A. MELLIÈS, N. ZEILBERGER. *A bifibrational reconstruction of Lawvere's presheaf hyperdoctrine*, in "IEEE/ACM Logic in Computer Science (LICS) 2016", New York, United States, Proceedings of LICS'16, June 2016, https://hal.archives-ouvertes.fr/hal-01261955

Research Reports

[34] M. P. BONACINA, S. GRAHAM-LENGRAND, N. SHANKAR. *A model-constructing framework for theory combination*, Universita degli Studi di Verona, September 2016, n^o RR-99/2016, https://hal.inria.fr/hal-01425305

[35] K. CHAUDHURI, S. MARIN, L. STRASSBURGER. *Focused and Synthetic Nested Sequents (Extended Technical Report)*, Inria, April 2016, https://hal.inria.fr/hal-01251722

Other Publications

[36] R. BLANCO, Z. CHIHANI. *An interactive assistant for the definition of proof certificates*, December 2016, working paper or preprint, https://hal.inria.fr/hal-01422829

[37] T. BROCK-NANNESTAD, D. ILIK. *An Intuitionistic Formula Hierarchy Based on High-School Identities*, August 2016, working paper or preprint, https://hal.inria.fr/hal-01354181

[38] Z. CHIHANI, D. ILIK, D. MILLER. *Classical polarizations yield double-negation translations*, August 2016, working paper or preprint, https://hal.inria.fr/hal-01354298

[39] D. ILIK. *Perspectives for proof unwinding by programming languages techniques*, August 2016, Book chapter to appear, https://hal.inria.fr/hal-01354180

References in notes

[40] J. A. ROBINSON, A. VORONKOV (editors). *Handbook of Automated Reasoning*, Elsevier and MIT press, 2001

[41] S. ABRAMSKY. *Computational Interpretations of Linear Logic*, in "Theoretical Computer Science", 1993, vol. 111, pp. 3–57

[42] B. ACCATTOLI, U. DAL LAGO. *Beta reduction is invariant, indeed*, in "Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014", 2014, pp. 8:1–8:10, http://doi.acm.org/10.1145/2603088.2603105

[43] J.-M. ANDREOLI. *Logic Programming with Focusing Proofs in Linear Logic*, in "Journal of Logic and Computation", 1992, vol. 2, n^{o} 3, pp. 297–347

[44] A. ASSAF, G. BUREL, R. CAUDERLIER, D. DELAHAYE, G. DOWEK, C. DUBOIS, F. GILBERT, P. HALMAGRAND, O. HERMANT, R. SAILLARD. *Dedukti: a Logical Framework based on the $\lambda\Pi$-Calculus Modulo Theory*, 2016, Unpublished, http://www.lsv.ens-cachan.fr/~dowek/Publi/expressing.pdf

[45] D. BAELDE, D. MILLER, Z. SNOW. *Focused Inductive Theorem Proving*, in "Fifth International Joint Conference on Automated Reasoning (IJCAR 2010)", J. GIESL, R. HÄHNLE (editors), LNCS, 2010, n^{o} 6173, pp. 278–292 [*DOI :* 10.1007/978-3-642-14203-1], http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/ijcar10.pdf

[46] G. E. BLELLOCH, J. GREINER. *Parallelism in Sequential Functional Languages*, in "Proceedings of the seventh international conference on Functional programming languages and computer architecture, FPCA 1995, La Jolla, California, USA, June 25-28, 1995", 1995, pp. 226–237, http://doi.acm.org/10.1145/224164.224210

[47] O. BODINI, D. GARDY, A. JACQUOT. *Asymptotics and random sampling for BCI and BCK lambda terms*, in "Theoretical Computer Science", 2013, vol. 502, pp. 227–238

[48] K. CHAUDHURI. *The Focused Inverse Method for Linear Logic*, Carnegie Mellon University, December 2006, Technical report CMU-CS-06-162, http://reports-archive.adm.cs.cmu.edu/anon/2006/CMU-CS-06-162.pdf

[49] K. CHAUDHURI, N. GUENOT, L. STRASSBURGER. *The Focused Calculus of Structures*, in "Computer Science Logic: 20th Annual Conference of the EACSL", Leibniz International Proceedings in Informatics (LIPIcs), Schloss Dagstuhl–Leibniz-Zentrum für Informatik, September 2011, pp. 159–173 [*DOI :* 10.4230/LIPICS.CSL.2011.159], http://drops.dagstuhl.de/opus/volltexte/2011/3229/pdf/16.pdf

[50] K. CHAUDHURI, S. HETZL, D. MILLER. *A Multi-Focused Proof System Isomorphic to Expansion Proofs*, in "Journal of Logic and Computation", June 2014 [*DOI :* 10.1093/LOGCOM/EXU030], http://hal.inria.fr/hal-00937056

[51] Z. CHIHANI, D. MILLER, F. RENAUD. *Checking Foundational Proof Certificates for First-Order Logic (extended abstract)*, in "Third International Workshop on Proof Exchange for Theorem Proving (PxTP 2013)", J. C. BLANCHETTE, J. URBAN (editors), EPiC Series, EasyChair, 2013, vol. 14, pp. 58–66

[52] Z. CHIHANI, D. MILLER, F. RENAUD. *Foundational proof certificates in first-order logic*, in "CADE 24: Conference on Automated Deduction 2013", M. P. BONACINA (editor), Lecture Notes in Artificial Intelligence, 2013, n^{o} 7898, pp. 162–177

[53] A. DAS, L. STRASSBURGER. *No complete linear term rewriting system for propositional logic*, in "26th International Conference on Rewriting Techniques and Applications (RTA 2015)", Warsaw, Poland, 26th International Conference on Rewriting Techniques and Applications (RTA 2015), June 2015 [*DOI :* 10.4230/LIPICS.RTA.2015.127], https://hal.inria.fr/hal-01236948

[54] C. DROSS, S. CONCHON, J. KANIG, A. PASKEVICH. *Reasoning with Triggers*, in "10th Intern. Worksh. on Satisfiability Modulo Theories, SMT 2012", P. FONTAINE, A. GOEL (editors), EPiC Series, EasyChair, June 2012, vol. 20, pp. 22–31, http://www.easychair.org/publications/?page=2135488790

[55] P. FONTAINE, J.-Y. MARION, S. MERZ, L. P. NIETO, A. TIU. *Expressiveness + Automation + Soundness: Towards Combining SMT Solvers and Interactive Proof Assistants*, in "TACAS: Tools and Algorithms for the Construction and Analysis of Systems, 12th International Conference", H. HERMANNS, J. PALSBERG (editors), LNCS, Springer, 2006, vol. 3920, pp. 167–181 [*DOI :* 10.1007/11691372_11]

[56] A. GACEK, D. MILLER, G. NADATHUR. *Combining generic judgments with recursive definitions*, in "23th Symp. on Logic in Computer Science", F. PFENNING (editor), IEEE Computer Society Press, 2008, pp. 33–44, http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/lics08a.pdf

[57] H. GANZINGER, H. RUEB, N. SHANKAR. *Modularity and Refinement in Inference Systems*, SRI, 2004, n^{o} SRI-CSL-04-02

[58] J.-Y. GIRARD. *Linear Logic*, in "Theoretical Computer Science", 1987, vol. 50, pp. 1–102

[59] S. GRAHAM-LENGRAND, R. DYCKHOFF, J. MCKINNA. *A Focused Sequent Calculus Framework for Proof Search in Pure Type Systems*, in "Logical Methods in Computer Science", 2011, vol. 7, n^{o} 1, http://www.lix.polytechnique.fr/~lengrand/Work/Reports/TTSC09.pdf

[60] S. GRAHAM-LENGRAND. *Slot Machines: an approach to the Strategy Challenge in SMT solving (presentation only)*, in "13th International Workshop on Satisfiability Modulo Theories", San Francisco, United States, July 2015, https://hal.inria.fr/hal-01211209

[61] A. GUGLIELMI. *A System of Interaction and Structure*, in "ACM Trans. on Computational Logic", 2007, vol. 8, n^{o} 1

[62] A. GUGLIELMI, T. GUNDERSEN. *Normalisation Control in Deep Inference Via Atomic Flows*, in "Logical Methods in Computer Science", 2008, vol. 4, n^{o} 1:9, pp. 1–36, http://arxiv.org/abs/0709.1205

[63] A. GUGLIELMI, L. STRASSBURGER. *Non-commutativity and MELL in the Calculus of Structures*, in "Computer Science Logic, CSL 2001", L. FRIBOURG (editor), LNCS, Springer-Verlag, 2001, vol. 2142, pp. 54–68

[64] Q. HEATH, D. MILLER. *A framework for proof certificates in finite state exploration*, in "Proceedings of the Fourth Workshop on Proof eXchange for Theorem Proving", C. KALISZYK, A. PASKEVICH (editors), Electronic Proceedings in Theoretical Computer Science, Open Publishing Association, August 2015, n^{o} 186, pp. 11–26 [*DOI :* 10.4204/EPTCS.186.4], http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/pxtp2015.pdf

[65] D. JOVANOVIĆ, C. BARRETT, L. DE MOURA. *The Design and Implementation of the Model Constructing Satisfiability Calculus*, in "Proc. of the 13th Int. Conf. on Formal Methods In Computer-Aided Design (FMCAD '13)", FMCAD Inc., 2013, Portland, Oregon, http://www.cs.nyu.edu/~barrett/pubs/JBdM13.pdf

[66] F. W. LAWVERE. *Adjointness in Foundations*, in "Dialectica", 1969, vol. 23, pp. 281–296

[67] C. LIANG, D. MILLER. *Focusing and Polarization in Linear, Intuitionistic, and Classical Logics*, in "Theoretical Computer Science", 2009, vol. 410, n^{o} 46, pp. 4747–4768 [*DOI :* 10.1016/J.TCS.2009.07.041], http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/tcs09.pdf

[68] C. LIANG, D. MILLER. *A Focused Approach to Combining Logics*, in "Annals of Pure and Applied Logic", 2011, vol. 162, n° 9, pp. 679–697 [*DOI :* 10.1016/J.APAL.2011.01.012]

[69] P. MARTIN-LÖF. *Constructive Mathematics and Computer Programming*, in "Sixth International Congress for Logic, Methodology, and Philosophy of Science", Amsterdam, North-Holland, 1982, pp. 153–175

[70] R. MCDOWELL, D. MILLER. *Reasoning with Higher-Order Abstract Syntax in a Logical Framework*, in "ACM Trans. on Computational Logic", 2002, vol. 3, n° 1, pp. 80–136, http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/mcdowell01.pdf

[71] R. MCDOWELL, D. MILLER. *A Logic for Reasoning with Higher-Order Abstract Syntax*, in "Proceedings, Twelfth Annual IEEE Symposium on Logic in Computer Science", Warsaw, Poland, G. WINSKEL (editor), IEEE Computer Society Press, July 1997, pp. 434–445

[72] P.-A. MELLIÈS, N. ZEILBERGER. *Functors are Type Refinement Systems*, in "42nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2015)", Mumbai, India, January 2015 [*DOI :* 10.1145/2676726.2676970], https://hal.inria.fr/hal-01096910

[73] D. MILLER. *Forum: A Multiple-Conclusion Specification Logic*, in "Theoretical Computer Science", September 1996, vol. 165, n° 1, pp. 201–232

[74] D. MILLER, G. NADATHUR. *Programming with Higher-Order Logic*, Cambridge University Press, June 2012 [*DOI :* 10.1017/CBO9781139021326]

[75] D. MILLER, G. NADATHUR, F. PFENNING, A. SCEDROV. *Uniform Proofs as a Foundation for Logic Programming*, in "Annals of Pure and Applied Logic", 1991, vol. 51, pp. 125–157

[76] D. MILLER, A. TIU. *A Proof Theory for Generic Judgments: An extended abstract*, in "Proc. 18th IEEE Symposium on Logic in Computer Science (LICS 2003)", IEEE, June 2003, pp. 118–127, http://www.lix.polytechnique.fr/Labo/Dale.Miller/papers/lics03.pdf

[77] R. MILNER. *LCF: A Way of Doing Proofs with a Machine*, in "Proc. of the 8th Intern. Symp. on the Mathematical Foundations of Computer Science", J. BECVÁR (editor), LNCS, Springer, 1979, vol. 74, pp. 146-159

[78] S. NEGRI. *Proof Analysis in Modal Logic*, in "Journal of Philosophical Logic", 2005, vol. 34, n° 5-6, pp. 507–544 [*DOI :* 10.1007/S10992-005-2267-3]

[79] G. NELSON, D. C. OPPEN. *Simplification by Cooperating Decision Procedures*, in "ACM Press Trans. on Program. Lang. and Syst.", October 1979, vol. 1, n° 2, pp. 245–257 [*DOI :* 10.1145/357073.357079]

[80] F. PFENNING, C. SCHÜRMANN. *System Description: Twelf — A Meta-Logical Framework for Deductive Systems*, in "16th Conference on Automated Deduction", Trento, H. GANZINGER (editor), LNAI, Springer, 1999, n° 1632, pp. 202–206

[81] B. PIENTKA, J. DUNFIELD. *Beluga: A Framework for Programming and Reasoning with Deductive Systems (System Description)*, in "Fifth International Joint Conference on Automated Reasoning", J. GIESL, R. HÄHNLE (editors), LNCS, 2010, n° 6173, pp. 15–21

[82] M. PUECH. *Typeful Continuations*, in "Journées Francophones des Langages Applicatifs (JFLA) 2017", Gourette, France, January 2017, https://hal.archives-ouvertes.fr/hal-01419473

[83] E. P. ROBINSON. *Proof Nets for Classical Logic*, in "Journal of Logic and Computation", 2003, vol. 13, pp. 777–797

[84] R. E. SHOSTAK. *Deciding Combinations of Theories*, in "J. ACM", 1984, vol. 31, n^{o} 1, pp. 1–12 [*DOI :* 10.1145/2422.322411]

[85] THE COQ DEVELOPMENT TEAM. *The Coq Proof Assistant Version 8.3 Reference Manual*, Inria, October 2010

[86] A. TIU, D. MILLER. *Proof Search Specifications of Bisimulation and Modal Logics for the π-calculus*, in "ACM Trans. on Computational Logic", 2010, vol. 11, n^{o} 2, http://arxiv.org/abs/0805.2785

[87] N. ZEILBERGER, A. GIORGETTI. *A correspondence between rooted planar maps and normal planar lambda terms*, in "Logical Methods in Computer Science", September 2015, vol. 11, n^{o} 3:22, pp. 1-39 [*DOI :* 10.2168/LMCS-11(3:22)2015], https://hal.inria.fr/hal-01057269

[88] N. ZEILBERGER. *Counting isomorphism classes of β-normal linear lambda terms*, September 2015, 5 pages, https://hal.archives-ouvertes.fr/hal-01214121

[89] L. M. DE MOURA, D. JOVANOVIC. *A Model-Constructing Satisfiability Calculus*, in "Proc. of the 14th Int. Conf. on Verification, Model Checking, and Abstract Interpretation (VMCAI'13)", R. GIACOBAZZI, J. BERDINE, I. MASTROENI (editors), LNCS, Springer-Verlag, 2013, vol. 7737, pp. 1–12 [*DOI :* 10.1007/978-3-642-35873-9_1]

[90] L. M. DE MOURA, G. O. PASSMORE. *The Strategy Challenge in SMT Solving*, in "Automated Reasoning and Mathematics - Essays in Memory of William W. McCune", M. P. BONACINA, M. E. STICKEL (editors), LNCS, Springer, 2013, vol. 7788, pp. 15–44 [*DOI :* 10.1007/978-3-642-36675-8_2], http://dx.doi.org/10.1007/978-3-642-36675-8