RESEARCH CENTRE

Inria Saclay Centre at Institut Polytechnique de Paris

IN PARTNERSHIP WITH:

CNRS, Institut Polytechnique de Paris

2024

ACTIVITY REPORT

Project-Team

PARTOUT

Proof Automation and RepresenTation: a fOundation of compUtation and deducTion

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

DOMAIN

Algorithmics, Programming, Software and Architecture

THEME

Proofs and Verification

Inria

Contents

Project-Team PARTOUT

Creation of the Project-Team: 2019 December 01

Keywords

Computer sciences and digital sciences

A2.1. – Programming Languages

A2.2. – Compilation

A2.4. – Formal method for verification, reliability, certification

A4.5. – Formal methods for security

A7.2. – Logic in Computer Science

A7.2.1. – Decision procedures

A7.2.2. – Automated Theorem Proving

A7.2.3. – Interactive Theorem Proving

A7.2.4. – Mechanized Formalization of Mathematics

A7.3.1. – Computational models and calculability

A8.1. – Discrete mathematics, combinatorics

A8.11. – Game Theory

Other research topics and application domains

B6.1. – Software industry

1 Team members, visitors, external collaborators

Research Scientists

- Lutz Strassburger [Team leader, INRIA, Senior Researcher]
- Beniamino Accattoli [INRIA, Researcher]
- Kaustuv Chaudhuri [INRIA, Researcher]
- Dale Miller [INRIA, Senior Researcher]

Faculty Members

- Ambroise Lafont [Ecole Polytechnique, Associate professor]
- Benjamin Werner [Ecole polytechnique, Professor, LIX, HDR]
- Noam Zeilberger [Ecole Polyetchnique, Associate Professor, LIX]

Post-Doctoral Fellows

- Victoria Barrett [INRIA, Post-Doctoral Fellow, from Sep 2024]
- Bryce Clarke [INRIA, Post-Doctoral Fellow, until Apr 2024]

PhD Students

- Farah Al Wardani [INRIA]
- Pablo Donato [INRIA, until Feb 2024]
- Arunava Gantait [ECOLE POLY PALAISEAU]
- Adrienne Lancelot [INRIA]
- Niyousha Najmaei [ECOLE POLY PALAISEAU, from Nov 2024]
- Adonis Rima [INRIA]
- Jui-Hsuan Wu [IP PARIS, until Sep 2024]

Interns and Apprentices

- Samy Avrillon [ENS DE LYON, Intern, from Apr 2024 until Aug 2024]
- Yoan Bouniard [INRIA, Intern, from Apr 2024 until Sep 2024]
- Mathis Bouverot-Dupuis [ENS PARIS]
- Alizee Marquardt [ECOLE POLY PALAISEAU, Intern, from Jun 2024 until Aug 2024]
- Aya Matmata [INRIA, Intern, from Jul 2024 until Aug 2024]
- Samar Rahmouni [INRIA, Intern, from Apr 2024 until Jul 2024]

Administrative Assistant

- Michael Barbosa [INRIA]

External Collaborator

- Olivier Martinot [Inria Paris, until Apr 2024]

2 Overall objectives

There is an emerging consensus that formal methods must be used as a matter of course in software development. Most software is too complex to be fully understood by one programmer or even a team of programmers, and requires the help of computerized techniques such as testing and model checking to analyze and eliminate entire classes of bugs. Moreover, in order for the software to be maintainable and reusable, it not only needs to be bug-free but also needs to have fully specified behavior, ideally accompanied with formal and machine-checkable proofs of correctness with respect to the specification. Indeed, formal specification and machine verification is the only way to achieve the highest level of assurance (EAL7) according to the ISO/IEC Common Criteria.[1]

Historically, achieving such a high degree of certainty in the operation of software has required significant investment of manpower, and hence of money. As a consequence, only software that is of critical importance (and relatively unchanging), such as monitoring software for nuclear reactors or fly-by-wire controllers in airplanes, has been subjected to such intense scrutiny. However, we are entering an age where we need trustworthy software in more mundane situations, with rapid development cycles, and without huge costs. For example: modern cars are essentially mobile computing platforms, smart-devices manage our intensely personal details, elections (and election campaigns) are increasingly fully computerized, and networks of drones monitor air pollution, traffic, military arenas, etc. Bugs in such systems can certainly lead to unpleasant, dangerous, or even life-threatening incidents.

The field of formal methods has stepped up to meet this growing need for trustworthy general purpose software in recent decades. Techniques such as computational type systems and explicit program annotations/contracts, and tools such as model checkers and interactive theorem provers, are starting to become standard in the computing industry. Indeed, many of these tools and techniques are now a part of undergraduate computer science curricula. In order to be usable by ordinary programmers (without PhDs in logic), such tools and techniques have to be high level and rely heavily on automation. Furthermore, multiple tools and techniques often need to marshaled to achieve a verification task, so theorem provers, solvers, model checkers, property testers, etc. need to be able to communicate with—and, ideally, trust—each other.

With all this sophistication in formal tools, there is an obvious question: what should we trust? Sophisticated formal reasoning tools are, generally speaking, complex software artifacts themselves; if we want complex software to undergo rigorous formal analysis we must be prepared to formally analyze the tools and techniques used in formal reasoning itself. Historically, the issue of trust has been addressed by means of relativizing it to *small* and *simple* cores. This is the basis of industrially successful formal reasoning systems such as Coq, Isabelle, HOL4, and ACL2. However, the relativization of trust has led to a balkanization of the formal reasoning community, since the Coq kernel, for example, is incompatible with the Isabelle kernel, and neither can directly cross-validate formal developments built with the other. Thus, there is now a burgeoning cottage industry of translations and adaptations of different formal proof languages for bridging the gap. A number of proposals have also been made for universal or retargetable proof languages (e.g., Dedukti, ProofCert) so that the cross-platform trust issues can be factorized into single trusted checkers.

Beyond mutual incompatibility caused by relativized trust, there is a bigger problem that the proof evidence that is accepted by small kernels is generally far too detailed to be useful. Formal developments usually occurs at a much higher level, relying on algorithmic techniques such as unification, simplification, rewriting, and controlled proof search to fill in details. Indeed, the most reusable products of formal developments tend to be these algorithmic techniques and associated collections of hand-crafted rules. Unfortunately, these techniques are even less portable than the fully detailed proofs themselves, since the techniques are often implemented in terms of the behaviors of the trusted kernels. We can broadly say that the problem with relativized trust is that it is based on the *operational* interpretation of implementations of trusted kernels. There still remains the question of *meta-theoretic correctness*. Most formal reasoning

[1] http://www.commoncriteriaportal.org/cc/

systems implement a variant of a well known mathematical formalism (e.g., Martin-Löf type theory, set theory, higher-order logic), but it is surprising that hardly any mainstream system has a formalized meta-theory.[2] Furthermore, formal reasoning systems are usually associated with complicated checkers for side-conditions that often have unclear mathematical status. For example, the Coq kernel has a built-in syntactic termination checker for recursive fixed-point expressions that is required to work correctly for the kernel to be sound. This termination checker evolves and improves with each version of Coq, and therefore the most accurate documentation of its behavior is its own source code. Coq is not special in this regard: similar trusted features exist in nearly every mainstream formal reasoning system.

The PARTOUT project is interested in the principles of deductive and computational formalisms. In the broadest sense, we are interested in the question of *trustworthy and verifiable meta-theory*. At one end, this includes the well studied foundational questions of the meta-theory of logical systems and type systems: cut-elimination and focusing in proof theory, type soundness and normalization theorems in type theory, etc. The focus of our research here is on the fundamental relationships behind the the notions of *computation* and *deduction*. We are particularly interested in relationships that go beyond the well known correspondences between proofs and programs.[3] Indeed, interpreting *computation in terms of deduction* (as in logic programming) or *deduction in terms of computation* (as in rewrite systems or in model checking) can often lead to fruitful and enlightening research questions, both theoretical and practical.

From another end, PARTOUT works on the question of the *essential nature* of deductive or computational formalisms. For instance, we are interested in the question of *proof identity* that attempts to answer the following question: when are two proofs of the same theorem the same? Surprisingly, this very basic question is left unanswered in *proof theory*, the branch of mathematics that supposedly treats proofs as algebraic objects of interest. We also pay particular attention to the combinatorial and complexity-theoretic properties of the formalisms. Indeed, it is surprising that until very recently the λ-calculus, which is the de facto basis of every functional programming language, lacked a good complexity-theoretic foundation, i.e., a cost model that would allow us to use the λ-calculus directly to define complexity classes.

To put trustworthy meta-theory to use, the PARTOUT project also works on the design and implementations of formal reasoning tools and techniques. We study the mathematical principles behind the representations of formal concepts (λ-terms, proofs, abstract machines, etc.), with the goal of identifying the relationships and trade-offs. We also study computational formalisms such as higher-order relational programming that is well suited to the specification and analysis of systems defined in the *structural operational semantics* (SOS) style. We also work on foundational questions about induction and co-induction, which are used in intricate combinations in metamathematics.

3 Research program

Software and hardware systems perform *computation* (systems that process, compute and perform) and *deduction* (systems that search, check or prove). The makers of those systems express their intent using various frameworks such as programming languages, specification languages, and logics. The PARTOUT project aims at developing and using mathematical principles to design better frameworks for computation and reasoning. Principles of expression are researched from two directions, in tandem:

- Foundational approaches, from theories to applications: studying fundamental problems of programming and proof theory.

  Examples include studying the complexity of reduction strategies in lambda-calculi with sharing, or studying proof representations that quotient over rule permutations and can be adapted to many different logics.

- Empirical approaches, from applications to theories: studying systems currently in use to build a theoretical understanding of the practical choices made by their designers.

[2] A prominent exception is HOL-Light, whose implementation has been self-certified—in HOL-Light itself—up to a strong assumption necessary to side-step incompleteness.

[3] The *Curry-Howard* correspondence.

Examples include studying realistic implementations of programming languages and proof assistants, which differ in interesting ways from their usual high-level formal description (regarding of sharing of code and data, for example), or studying new approaches to efficient automated proof search, relating them to existing approaches of proof theory, for example to design proof certificates or to generalize them to non-classical logics.

One of the strengths of PARTOUT is the co-existence of a number of different expertise and points of view. Many dichotomies exist in the study of computation and deduction: functional programming *vs* logic programming, operational semantics *vs* denotational semantics, constructive logic *vs* classical logic, proof terms *vs* proof nets, etc. We do not identify with any one of them in particular, rather with them as a whole, believing in the value of interaction and cross-fertilization between different approaches. PARTOUT defines its scope through the following core tenets:

- An interest in both computation and logic.
- The use of mathematical formalism as our core scientific method, paired with practical implementations of the systems we study.
- A shared belief in the importance of good *design* when creating new means of expression, iterating towards simplicity and elegance.

More concretely, the research in PARTOUT will be centered around the following four themes:

1. **Foundations of proof theory as a theory of proofs.** Current proof theory is not a theory of proofs but a theory of proof systems. This has many practical consequences, as a proof produced by modern theorem provers cannot be considered independent from the tool that produced it. A central research topic here is the quest for proof representations that are independent from the proof system, so that proof theory becomes a proper theory of proofs.
2. **Program Equivalence** We intend to use our proof theoretical insights to deepen our understanding of the structure of computer programs by discovering canonical representations for functional programming languages, and to apply these to the problems of program equivalence checking and program synthesis.
3. **Reasoning with relational specifications of formal systems.** Formal systems play a central role for proof checkers and proof assistants that are used for software verification. But there is usually a large gap between the specification of those formal systems in concise informal mathematical language and their implementation in ML or C code. Our research goal is to close that gap.
4. **Foundations of complexity analysis for functional programs.** One of the great merits of the functional programming paradigm is the natural availability of high-level abstractions. However, these abstractions jeopardize the programmer's predictive control on the performance of the code, since many low-level steps are abstracted away by higher-order functions. Our research goal is to regain that control by developing models of space and time costs for functional programs.

4 Application domains

4.1 Automated Theorem Proving

The Partout team studies the structure of mathematical proofs, in ways that often makes them more amenable to automated theorem proving – automatically searching the space of proof candidates for a statement to find an actual proof – or a counter-example.

(Due to fundamental computability limits, fully-automatic proving is only possible for simple statements, but this field has been making a lot of progress in recent years, and is in particular interested with the idea of generating verifiable evidence for the proofs that are found, which fits squarely within the expertise of Partout.)

4.2 Proof-assistants

Our work on the structure of proofs also suggests ways how they could be presented to a user, edited and maintained, in particular in “proof assistants”, automated tool to assist the writing of mathematical proofs with automatic checking of their correctness.

4.3 Programming language design

Our work also gives insight on the structure and properties of programming languages. We can improve the design or implementation of programming languages, help programmers or language implementors reason about the correctness of the programs in a given language, or reason about the cost of execution of a program.

5 Social and environmental responsibility

- Benjamin Werner is an elected member of the executive boards (conseils d’administration) of both Ecole polytechnique (X) and Institut Polytechnique de Paris (IPP)

6 Highlights of the year

- Accattoli joint paper with Dal Lago and Vanoni in LICS 2022 was selected for the special issue of the conference. The journal version was published in 2024 [5]. It solves the long-standing problem of finding a reasonable space cost model for the lambda calculus accounting for logarithmic space.
- Lutz Strassburger won a PHC Sophie Germaine grant over 33.500 EUR, joint with UCL, UK on "Formal Verification for Large Language Models" (Period: 1. September 2024 – 31. December 2025)

7 New software, platforms, open data

7.1 New software

7.1.1 Abella

Keyword: Proof assistant

Functional Description: Abella is an interactive theorem prover for reasoning about computations given as relational specifications. Abella is particularly well suited for reasoning about binding constructs.

Release Contributions: This release includes a major refactoring of the Abella documentation generator. Abella developments can now be easily converted into interactive web-based presentations that can be used without having to run Abella by the reader.

This release also fixes a number of outstanding issues with the 2.0.7 and earlier releases. At least two of these fixes involve soundness issues with regard to higher-order arguments.

Abella is now also independently packaged for MacOS (homebrew), FreeBSD, and OpenBSD.

URL: https://abella-prover.org/

Contact: Kaustuv Chaudhuri

Participants: Dale Miller, Gopalan Nadathur, Kaustuv Chaudhuri, Mary Southern, Mattéo Cimini, Olivier Savary-Belanger, Yuting Wang

Partner: Department of Computer Science and Engineering, University of Minnesota

7.1.2 Actema

Name: Actema

Keywords: Higher-order logic, First-order logic, Proof assistant, GUI (Graphical User Interface), Man-machine interfaces, User Interfaces

Functional Description: This is a new approach, aiming at making the building of formal proofs more intuitive and convenient. The system is currently at a prototype stage. An interfacing with the Coq proof-system has been developed in 2023 and 2024 and is now freely available. The system runs through an html/JS serve.

Release Contributions: This version can be used online at actema.xyz and comes with explanation videos.

URL: http://actema.xyz

Publication: 03823357

Contact: Benjamin Werner

Participants: Mathis Bouverot-Dupuis, Benjamin Werner, Pablo Donato, Pierre-Yves Strub

Partner: Ecole Polytechnique

7.1.3 DAMF Dispatch

Keywords: Interactive Theorem Proving, Distributed systems, Verification

Scientific Description: The Distributed Assertion Management Framework (DAMF) is a proposed collection of formats and techniques to enable heterogeneous formal reasoning systems and users to communicate assertions in a decentralized, reliable, and egalitarian manner. An assertion is a unit of mathematical knowledge—think lemmas, theorems, corollaries, etc.—that is cryptographically signed by its originator.

DAMF is based on content-addressable storage using the InterPlanetary File System (IPFS) network, and uses the InterPlanetary Linked Data (IPLD) data model to represent assertions and all their components.

Functional Description: Dispatch is an intermediary tool for publishing, retrieval, and trust analysis in the Distributed Assertion Management Framework (DAMF). Dispatch specifies a family of JSON-based formats for DAMF objects and implements the main DAMF processes. It is intended to be usable by both human users and tools.

Dispatch is being developed as part of the exploratory action W3Proof.

Release Contributions: This initial version has a demonstration proof of a theorem using a combination of Coq, LambdaProlog, and Abella.

URL: https://distributed-assertions.github.io

Publication: hal-04167922

Contact: Kaustuv Chaudhuri

Participants: Farah Al Wardani, Kaustuv Chaudhuri, Dale Miller

7.1.4 MOIN

Name: MOdal Intuitionistic Nested sequents

Keywords: Logic programming, Modal logic

Functional Description: MOIN is a SWI Prolog theorem prover for classical and intuitionstic modal logics. The modal and intuitionistic modal logics considered are all the 15 systems occurring in the modal S5-cube, and all the decidable intuitionistic modal logics in the IS5-cube. MOIN also provides a protptype implementation for the intuitionistic logics for which decidability is not known (IK4,ID5 and IS4). MOIN is consists of a set of Prolog clauses, each clause representing a rule in one of the three proof systems. The clauses are recursively applied to a given formula, constructing a proof-search tree. The user selects the nested proof system, the logic, and the formula to be tested. In the case of classic nested sequent and Maehara-style nested sequents, MOIN yields a derivation, in case of success of the proof search, or a countermodel, in case of proof search failure. The countermodel for classical modal logics is a Kripke model, while for intuitionistic modal logic is a bi-relational model. In case of Gentzen-style nested sequents, the prover does not perform a countermodel extraction.

A system description of MOIN is available at https://hal.inria.fr/hal-02457240

URL: http://www.lix.polytechnique.fr/Labo/Lutz.Strassburger/Software/Moin/MoinProver.html

Publication: hal-02457240

Contact: Lutz Strassburger

7.1.5 OCaml

Keywords: Programming language, Functional programming, Compilers

Functional Description: The OCaml language is a functional programming language that combines safety with expressiveness through the use of a precise and flexible type system with automatic type inference. The OCaml system is a comprehensive implementation of this language, featuring two compilers (a bytecode compiler, for fast prototyping and interactive use, and a native-code compiler producing efficient machine code for x86, ARM, PowerPC, RISC-V and System Z), a debugger, and a documentation generator. Many other tools and libraries are contributed by the user community and organized around the OPAM package manager.

URL: https://ocaml.org/

Publications: hal-04884634, hal-04681703, hal-04794404, hal-03917754, hal-03947986, hal-04407119, hal-03146495, hal-03510931, hal-03145030, hal-01929508, hal-03125031, hal-00772993, hal-00914493, hal-00914560, inria-00074804, hal-01499973, hal-01499946

Contact: Florian Angeletti

Participants: Florian Angeletti, Damien Doligez, Xavier Leroy, Luc Maranget, Gabriel Scherer, David Allsopp, Stephen Dolan, Alain Frisch, Jacques Garrigue, Anil Madhavapeddy, Kc Sivaramakrishnan, Nicolas Ojeda Bar, Leo White

7.1.6 ocaml-boxroot

Keywords: Interoperability, Library, Ocaml, Rust

Scientific Description: Boxroot is an implementation of roots for the OCaml GC based on concurrent allocation techniques. These roots are designed to support a calling convention to interface between Rust and OCaml code that reconciles the latter's foreign function interface with the idioms from the former.

Functional Description: Boxroot implements fast movable roots for OCaml in C. A root is a data type which contains an OCaml value, and interfaces with the OCaml GC to ensure that this value and its transitive children are kept alive while the root exists. This can be used to write programs in other languages that interface with programs written in OCaml.

URL: https://gitlab.com/ocaml-rust/ocaml-boxroot

Publication: hal-03910313

Contact: Guillaume Munch

Participants: Guillaume Munch, Gabriel Scherer

7.1.7 Profound-Intuitionistic

Name: Interactive theorem proving by direct manipulation for Intuitionistic Logic

Keywords: Interactive Theorem Proving, First-order logic

Functional Description: Profound-Intuitionistic (Profint) is a tool for building formal proofs in intuitionistic logic using an interactive direct manipulation based web-interface. The tool can transform the interactive proof into formal proof objects in a variety of backend provers including: Coq, Lean 3, Lean 4, Isabelle/HOL, HOL4, and Abella.

Release Contributions: This release adds support for inductive theorem proving using sized relations in the style of Abella.

This release also adds preliminary support for three-dimensional representations of the proof state (using WebGL and the Three.js library).

URL: https://github.com/direct-manipulation/profint

Contact: Kaustuv Chaudhuri

7.1.8 YADE

Name: Yet Another Diagram Editor

Keyword: Diagram

Functional Description: This diagram editor can help mechanising diagrammatic categorical proofs by generating Coq proof scripts from a drawn diagram. This is part of the Coreact ANR Project (started in March 2023), which aims at developing a methodology for diagrammatic reasoning in Coq.

URL: https://amblafont.github.io/graph-editor/index.html

Contact: Ambroise Lafont

Participant: Ambroise Lafont

8 New results

8.1 Loopchecking in proof search for intuitionistic modal logics

Participants: Lutz Straßburger.

External Collaborators: Marianna Girlando (Amsterdam), Sonia Marin (Birmingham), Marianela Morales (Madrid), Roman Kuznets (Vienna)

Following our work [24] from last year, we present an algorithm for establishing decidability and finite model property of intuitionistic modal logic IK. These two results have been previously established independently by proof theoretic and model theoretic techniques respectively. Our algorithm, by contrast, enables us to establish both properties at the same time and simplifies previous approaches. It implements root-first proof search in a labelled sequent calculus that employs two binary relations: one corresponding to the modal accessibility relation and the other to the preorder relation of intuitionistic models. As a result, all the rules become invertible, hence semantic completeness could be established directly by extracting a (possibly infinite) countermodel from a failed proof attempt. To obtain the finite model property, we rather introduce a simple loopcheck ensuring that root-first proof search always terminates. The resulting finite countermodel displays a layered structure akin to that of intuitionistic first-order models.

These results have been presented at the WoLLIC 2024 conference [15].

8.2 Lambek Calculus with Banged Atoms for Parasitic Gaps

Participants: Lutz Straßburger.

External Collaborators: Mehrnoosh Sadrzadeh (London)

Lambek Calculus is a non-commutative substructural logic for formalising linguistic constructions. However, its domain of applicability is limited to constructions with local dependencies. We propose here a simple extension that allows us to formalise a range of relativised constructions with long distance dependencies, notably medial extractions and the challenging case of parasitic gaps. In proof theoretic terms, our logic combines commutative and non-commutative behaviour, as well as linear and non-linear resource management. This is achieved with a single restricted modality. But unlike other extensions of Lambek Calculus with modalities, our logic remains decidable, and the complexity of proof search (i.e., sentence parsing) is the same as for the basic Lambek calculus. Furthermore, we provide not only a sequent calculus, and a cut elimination theorem, but also proof nets. This result has been published in [16].

8.3 LA strictly linear subatomic proof system

Participants: Victoria Barrett.

External Collaborators: Alessio Guglielmi (Bath), Benjamin Ralph(Bath)

We present a subatomic deep-inference proof system for a conservative extension of propositional classical logic with decision trees that is strictly linear. In a strictly linear subatomic system, a single linear rule shape subsumes not only the structural rules, such as contraction and weakening, but also the unit equality rules. An interpretation map from subatomic logic to propositional classical logic recovers the usual semantics and proof theoretic properties. By using explicit substitutions that indicate the substitution of one derivation into another, we are able to show that the unit-equality inference steps can be eliminated from a subatomic system for propositional classical logic with only a polynomial complexity cost in the size of the derivation, from which it follows that the system p-simulates Frege systems, and we show cut elimination for the resulting strictly linear system. This result will be presented at the CSL 2025 conference [14]

8.4 Property-Based Testing by Elaborating Proof Outlines

Participants: Dale Miller.

External Collaborators: Alberto Momigliano (Milan)

Property-based testing (PBT) is a technique for validating code against an executable specification by automatically generating test-data. We present a proof-theoretical reconstruction of this style of testing for relational specifications and employ the Foundational Proof Certificate framework to describe test generators. We do this by encoding certain kinds of "proof outlines" as proof certificates that can describe various common generation strategies in the PBT literature, ranging from random to exhaustive, including their combination. We also address the shrinking of counterexamples as a first step toward their explanation. Once generation is accomplished, the testing phase is a standard logic programming search. This approach is applied to simple, first-order (algebraic) data structures as well as to data structures containing bindings by using the λ-tree syntax approach to encode bindings. The λProlog programming language can perform both generating and checking of tests using this approach to syntax. We also show how to extend PBT to specifications in a fragment of linear logic.

These results have been published in the Theory and Practice of Logic Programming (TPLP) [8].

8.5 About Trust and Proof: An Experimental Framework for Heterogeneous Verification

Participants: Farah Al Wardani, Kaustuv Chaudhuri, Dale Miller.

Information and opinions come to us daily from a wide range of actors, including scientists, journalists, and pundits. Some actors may be biased or malicious, while others rely on physical measurements, statistics, or in-depth research. Some sources may be signed or edited, while others are anonymous and unmoderated. Trusting information from such diverse sources is a serious challenge facing society today. In this paper, we will describe another domain-the world of machinechecked logic and mathematics-in which many similar issues can appear but in which tractable solutions are possible. Many actors (people or software systems) assert that certain logical statements are theorems in this domain. We describe the Distributed Assertion Management Framework (DAMF) that explicitly manages claims by theorem provers that they have proved certain theorems from associated contexts. Provers willing to trust other provers will be able to avoid rechecking proofs.

This work on DAMF has been published in the Festschrift for Cliff Jones [19].

8.6 Peano Arithmetic and μMALL

Participants: Dale Miller, Matteo Manighetti.

Formal theories of arithmetic have traditionally been based on either classical or intuitionistic logic, leading to the development of Peano and Heyting arithmetic, respectively. We propose to use μMALL as a formal theory of arithmetic based on linear logic. This formal system is presented as a sequent calculus proof system that extends the standard proof system for multiplicative-additive linear logic (MALL) with the addition of the logical connectives universal and existential quantifiers (first-order quantifiers), term equality and non-equality, and the least and greatest fixed point operators. We first demonstrate how functions defined using μMALL relational specifications can be computed using a simple proof search algorithm. By incorporating weakening and contraction into μMALL, we obtain μLK+, a natural candidate for a classical sequent calculus for arithmetic. While important proof theory results are still lacking for μLK+ (including cut-elimination and the completeness of focusing), we prove that μLK+ is consistent and that it contains Peano arithmetic. We also prove some conservativity results regarding μLK+ over μMALL.

This paper has been submitted to a journal for publication: it is available as the technical report [23].

8.7 The categorical contours of the Chomsky-Schützenberger representation theorem

Participants: Noam Zeilberger

External Collaborators: Paul-André Melliès (Université Paris Cité, CNRS, Inria)

This paper [26], to appear in Logical Methods in Computer Science, is a thoroughly revised and expanded version of the work [25] that we presented at MFPS 2022. In this work, we analyze the Chomsky-Schützenberger Representation Theorem, a classic result that in its now-standard formulation states that a language is context-free if and only if it is a homomorphic image of the intersection of a Dyck language of well-bracketed words with a regular language. One reason the theorem is interesting is that it invokes a non-trivial closure property of context-free languages, namely closure under intersection with regular languages. Another reason is that it suggests, intuitively, that Dyck languages are in some sense "universal" context-free languages. It turns out that both aspects of the theorem have good categorical explanations, and that this leads to a generalization of the classical theory of context-free and regular languages, to languages of arrows in arbitrary categories.

These perspectives seem to have some real explanatory power and have already been taken up by others, including in recent PhD theses. Román showed that a "contour / splicing" adjunction of the form that we introduced in [25] could be used to characterize some aspects of process algebra (Mario Román, *Monoidal context theory*, PhD dissertation, November 2023), while Earnshaw demonstrated that there is a meaningful theory of regular and context-free languages of string diagrams in monoidal categories (Matthew Earnshaw, *Languages of string diagrams*, PhD dissertation, January 2025).

8.8 The free bifibration over a functor

Participants: Bryce Clarke, Noam Zeilberger

External Collaborators: Gabriel Scherer (Inria, IRIF)

A functor $p : D \to C$ between two categories is a *bifibration* when, roughly speaking, objects of D may be pushed and pulled along arrows of C. The categorical notion of bifibration was originally introduced by Grothendieck, together with the weaker notion of fibration where one only has the ability to pull objects of the category above along arrows of the category below. One reason for the special interest of bifibrations from the perspective of logic and computer science is that the operations of pushing forward or pulling back along an arrow may be seen as generalizations of existential and universal quantification, and hence by alternating these operations one can in some sense define objects of arbitrary quantifier complexity. The pushforward and pullback operations may also be seen as generalizations of strongest postconditions and weakest preconditions in specification logics.

In this work, we have studied the fundamental problem of building the free bifibration over a functor, using both proof-theoretic and categorical techniques. The basic idea is to construct the morphisms of the free bifibration as derivations in a sequent calculus modulo a natural notion of permutation equivalence, and then to establish a normal form theorem based on an appropriate notion of focusing.

We are currently writing up the results of this work as a long article with the intention of submitting it directly to a journal. Zeilberger presented some of these results at the LICS-affiliated Structure Meets Power 2024 workshop, while Scherer gave a short presentation at Journées PPS 2024.

8.9 Light Genericity

Participants: Beniamino Accattoli, Adrienne Lancelot.

This work is about the semantical theory of untyped lambda-calculi.

To better understand Barendregt's genericity for the untyped call-by-value lambda-calculus, we start by first revisiting it in call-by-name, adopting a lighter statement and establishing a connection with contextual equivalence. Then, we use it to give a new, lighter proof of maximality of head contextual equivalence, i.e. that H^* is a maximal consistent equational theory. We move on to call-by-value, where we adapt these results and also introduce a new notion dual to light genericity, that we dub co-genericity

. Lastly, we give alternative proofs of (co-)genericity based on applicative bisimilarity. This work was published in the proceedings of the international conference FoSSaCS 2024 [10] (it also has an associated technical report [20]).

8.10 Mirroring Call-By-Need, or Values Acting Silly

Participants: Beniamino Accattoli, Adrienne Lancelot.

This work is about evaluation strategies for the lambda-calculus.

Call-by-need evaluation for the λ-calculus can be seen as merging the best of call-by-name and call-by-value, namely the wise erasing behaviour of the former and the wise duplicating behaviour of the latter. To better understand how duplication and erasure can be combined, we design a degenerated calculus, dubbed call-by-silly, that is symmetric to call-by-need in that it merges the worst of call-by-name and call-by-value, namely silly duplications by-name and silly erasures by-value. We validate the design of the call-by-silly calculus via rewriting properties and multi types. In particular, we mirror the main theorem about call-by-need - that is, its operational equivalence with call-by-name - showing that call-by-silly and call-by-value induce the same contextual equivalence. This fact shows the blindness with respect to efficiency of call-by-value contextual equivalence. We also define a call-by-silly strategy and measure its length via tight multi types. Lastly, we prove that the call-by-silly strategy computes evaluation sequences of maximal length in the calculus.

This work was published in the proceedings of the international conference FSCD 2024 [11].

8.11 Positive Focusing is Directly Useful

Participants: Beniamino Accattoli, Jui-Hsuan Wu.

This work is about sharing for lambda-calculi and functional languages.

Recently, Miller and Wu introduced the positive lambda-calculus, a call-by-value lambda-calculus with sharing obtained by assigning proof terms to the positively polarized focused proofs for minimal intuitionistic logic. The positive lambda-calculus stands out among lambda-calculi with sharing for a compactness property related to the sharing of variables. We show that – thanks to compactness – the positive calculus neatly captures the core of useful sharing, a technique for the study of reasonable time cost models.

This work was published in the proceedings of the international conference MFPS 2024 [13] (it also has an associated technical report [21]).

8.12 IMELL Cut Elimination with Linear Overhead

Participants: Beniamino Accattoli.

External Collaborator: Claudio Sacerdoti Coen (Università di Bologna).

This work is about an abstract machine for performing cut elimination in linear logic proofs.

Recently, Accattoli introduced the Exponential Substitution Calculus (ESC) given by untyped proof terms for Intuitionistic Multiplicative Exponential Linear Logic (IMELL), endowed with rewriting rules at-a-distance for cut elimination. He also introduced a new cut elimination strategy, dubbed the good strategy, and showed that its number of steps is a time cost model with polynomial overhead for ESC/IMELL, and the first such one. Here, we refine Accattoli's result by introducing an abstract machine for ESC and proving that it implements the good strategy and computes cut-free terms/proofs within a linear overhead.

This work was published in the proceedings of the international conference FSCD 2024 [12].

8.13 Semantic Bounds and Multi Types, Revisited

Participants: Beniamino Accattoli.

This work is about extracting evaluation lengths from a specific kind of type system for the lambda calculus.

Intersection types are a standard tool in operational and semantical studies of the lambda-calculus. De Carvalho showed how multi types, a quantitative variant of intersection types providing a handy presentation of the relational denotational model, allows one to extract precise bounds on the number of beta-steps and the size of normal forms. In the last few years, de Carvalho's work has been extended and adapted to a number of lambda-calculi, evaluation strategies, and abstract machines. These works, however, only adapt the first part of his work, that extracts bounds from multi type derivations, while never consider the second part, which deals with extracting bounds from the multi types themselves. The reason is that this second part is more technical, and requires to reason up to type substitutions. It is however also the most interesting, because it shows that the bounding power is inherent to the relational model (which is induced by the types, without the derivations), independently of its presentation as a type system. Here we dissect and clarify the second part of de Carvalho's work, establishing a link with principal multi types, and isolating a key property independent of type substitutions.

This work was published in the proceedings of the international conference CSL 2024 [9].

8.14 Reasonable Space for the Lambda-Calculus, Logarithmically

Participants: Beniamino Accattoli.

External Collaborators: Ugo Dal Lago (Università di Bologna & Inria OLAS team), Gabriele Vanoni (IRIF, Université Paris Cité).

This work [5] is about reasonable space cost models for the lambda-calculus. It is the considerably extended journal version of the LICS 2022 conference paper with the same title and authors solving the long-standing problem of finding a reasonable space cost model for the lambda calculus accounting for logarithmic space. It was an invited submission to the special issue of LICS 2022.

8.15 Categorical semantics of pattern unification

Participants: Ambroise Lafont.

External Collaborators: Neel Krishnaswami (University of Cambridge).

We propose a notion of syntax with metavariables that generalises Miller's decidable pattern fragment of second-order unification for simply-typed λ-calculus. Using categorical semantics, we show that, under some conditions, a generalisation of Miller's unification algorithm applies. To illustrate our semantic analysis, we implemented our generic unification algorithm implemented in Agda. The syntax with metavariables given as input of the algorithm is specified by a notion of signature generalising binding signatures, covering a wide range of examples, including ordered λ-calculus and (intrinsic) polymorphic syntax such as System F. Although we do not explicitly handle equations, we also tackle simply-typed λ-calculus modulo β- and η-equations (Miller's original setting) by working on the syntax of normal forms.

This work was submitted to the Journal of Functional Programming.

8.16 A diagram editor to mechanise categorical proofs

Participants: Ambroise Lafont.

Diagrammatic proofs are ubiquitous in certain areas of mathematics, especially in category theory. Mechanising such proofs is a tedious task because proof assistants (such as Coq) are text based. We developed a prototypical diagram editor to make this process easier, building upon the vscode extension coq-lsp for the Coq proof assistant and a web application.

This was presented at JFLA 2024 [18].

8.17 Lifting twisted coreflections against delta lenses

Participants: Bryce Clarke

Delta lenses are functors equipped with a suitable choice of lifts, generalising the notion of split opfibration. In recent work, delta lenses were characterised as the right class of an algebraic weak factorisation system. In this paper, we show that this algebraic weak factorisation system is cofibrantly generated by a small double category, and characterise the left class as split coreflections with a certain property; we call these twisted coreflections. We demonstrate that every twisted coreflection arises as a pushout of an initial functor from a discrete category along a bijective-on-objects functor. Throughout the article, we take advantage of a reformulation of algebraic weak factorisation systems, due to Bourke, based on double-categorical lifting operations.

This work was posted as a preprint [22] in January 2024, and was accepted for publication in the journal *Theory and Applications of Categories*, appearing in July 2024.

8.18 The Flower Calculus

Participants: Pablo Donato.

We introduce the flower calculus, a deep inference proof system for intuitionistic first-order logic inspired by Peirce's existential graphs. It works as a rewriting system over inductive objects called "flowers", that enjoy both a graphical interpretation as topological diagrams, and a textual presentation as nested sequents akin to coherent formulas. Importantly, the calculus dispenses completely with the traditional notion of symbolic connective, operating solely on nested flowers containing atomic predicates. We prove both the soundness of the full calculus and the completeness of an analytic fragment with respect to Kripke semantics. This provides to our knowledge the first analyticity result for a proof system based on existential graphs, adapting semantic cut-elimination techniques to a deep inference setting. Furthermore, the kernel of rules targetted by completeness is fully invertible, a desirable property for both automated and interactive proof search. (published in [7])

9 Partnerships and cooperations

9.1 International initiatives

9.1.1 Inria associate team not involved in an IIL or an international program

COMPRONOM

Title: Combinatorial Proof Normalization

Duration: 2020 – 2024

Coordinator: Willem Heijltjes (w.b.heijltjes@bath.ac.uk)

Partners:

- Université de Bath (Royaume-Uni)

Inria contact: Dale Miller, Lutz Strassburger

Summary: This project teams up three research groups, one at Inria Saclay, one at the University of Bath, and one at University College London, who are driven by their joint interest in the development of a combinatorial proof theory which is able to treat formal proofs independently from syntactic proof systems.

We plan to focus our research in two major directions: First, study the normalization of combinatorial proofs, with possible applications for the implementation of functional programming languages, and second, study combinatorial proofs for the logic of bunched implications, with the possible application for separation logic and its use in the verification of imperative programs.

9.1.2 STIC/MATH/CLIMAT AmSud projects

DLR

Title: Dynamic logics (reloaded)

Program: STIC-AmSud

Duration: January 1, 2024 – December 31, 2025

Local supervisor: Lutz Strassburger

Partners:

- *Carlos Areces* (AFacultad de Matemática, Astronomía, Física y Computación, Universidad Nacional de Córdoba, Argentine)
- *(Mario R. F. Benevides* (Instituto de Computação, Universidade Federal Fluminense, Brésil)
- *Pablo Barceló* (Institute for Mathematical and Computational Engineering, Universidad Católica de Chile y Millennium Institute for Foundational Research on Data, Chili)

Inria contact: Lutz Strassburger

Summary: During the project we will advance our understanding of a novel family of logics called dynamic logics. Dynamic logics are characterized by the inclusion of operators that can modify the model in which they are being evaluated. This characteristic made them especially well suited for the description of evolving scenarios like, for example, the temporal evolution of a communication network, where connections are dynamically created and eliminated, constantly changing the actual topology. A number of different dynamic logics have been investigated by members of the project, but a general perspective is still missing, and a number of important open questions remains, ranging from adequate model theoretic characterizations, to a proper understanding of how to define proof calculi for these logics. In recent years, the potential applications of dynamic logics have grown, with the recent rise of AI techniques based on knowledge represented as large graphs. The project aims to pull together the strengths of the five international research teams, to unify existing results and attempt to answer these open problems.

9.1.3 Participation in other International Programs

PHC Sophie Germain funding scheme

Title: Formal Verification for Large Language Models

Inria contact: Lutz Strassburger

Partner Institution(s): University College London (UCL), UK

Date/Duration: 1. September 2024 – 31. December 2025

Summary: Methods from proof theory and formal verification can significantly enhance the reliability and trustworthiness of Large Language Models (LLMs). By applying formal verifi- cation, we can systematically ensure that LLMs adhere to specified properties and constraints, addressing several key challenges in their deployment.

9.2 International research visitors

9.2.1 Visits of international scientists

Other international visits to the team

Ryoma Sin'ya

Status researcher

Institution of origin: Akita University

Country: Japan

Dates: 12-13 September, 2024

Context of the visit: research visit

Benedikt Ahrens

Status Assistant professor

Institution of origin: Delft University of Technology

Country: Netherlands

Dates: 5-9 February, 2024

Context of the visit: research visit

9.3 National initiatives

LambdaComb

Title: LambdaComb: a cartographic quest between lambda-calculus, logic, and combinatorics

Duration: 2022 – 2026 (4 years)

Coordinator: Noam Zeilberger

Partners:

- LIX (Ecole Polytechnique), LIPN (Paris Nord), LIS (Marseille), LIGM (Marne-la-Vallée)
- Jagiellonian University (Poland)

Summary: LambdaComb is an interdisciplinary project financed by the Agence Nationale de la Recherche (PRC grant ANR-21-CE48-0017). Broadly, the project aims to deepen connections between lambda calculus and logic on the one hand and combinatorics on the other. One important motivation for the project is the discovery over recent years of a host of surprising links between subsystems of lambda calculus and enumeration of graphs on surfaces, or "maps", the latter being an active subfield of combinatorics with roots in W. T. Tutte's work in the 1960s. Using these new links and other ideas and tools, the LambdaComb project aims to:

- develop rigorous logical perspectives on maps and related combinatorial objects; and
- develop precise quantitative perspectives on lambda calculus and related systems.

The project also intersects with and aims to shed new light on other established connections between logic and geometry, notably Joyal and Street's categorical framework of string diagrams as well as Girard's proof nets for linear logic.

REPRO

Title: REPRO: searching for canonical REpresentations of PROgrams.

Duration: 2021 – 2025 (4 years)

Coordinator: Gabriel Scherer

Summary: The REPRO project aims to

1. deepen our understanding of the structure of computer programs by discovering canonical representations for fundamental programming languages, and to
2. explore the application of canonical representations to the problems of program equivalence checking and program synthesis.

CoREACT

Title: CoREACT: Coq-based Rewriting: towards Executable Applied Category Theory

Duration: 2023 – 2027 (4 years)

Coordinator: Nicolas Behr

Partners: IRIF (Université Paris Cité), LIP (ENS-Lyon), LIX (Ecole Polytechnique), Sophia-Antipolis (Inria)

Local participants: Ambroise Lafont, Benjamin Werner, Noam Zeilberger

Summary: The main objectives of the CoREACT project include:

1. Development of a methodology for diagrammatic reasoning in Coq
2. Formalization and certification of a representative collection of axioms and theorems for compositional categorical rewriting theory
3. Development of a Coq-enabled interactive database and wiki system
4. Development of a CoREACT wiki-based "proof-by-pointing" engine
5. Executable reference prototype algorithms from categorical structures in Coq (via the use of SMT solvers/theorem provers such as Z3)

10 Dissemination

10.1 Promoting scientific activities

10.1.1 Scientific events: organisation

General chair, scientific chair

- Lutz Strassburger was coorganizing (together with Anupam Das, Elaine Pimentel, Carlos Areces) the Dagstuhl Seminar 24341: *Proof Representations: From Theory to Applications*, Schloss Dagstuhl, Germany, 18–23 August 2024.

Member of the organizing committees

- Dale Miller was a member of the Steering Committee for both LICS and LFMTP.
- Beniamino Accattoli was a member of the Steering Committee of PPDP.

10.1.2 Scientific events: selection

Chair of conference program committees

- Dale Miller was a PC Chair for FLOPS 2024: 17th International Symposium on Functional and Logic Programming.

Member of the conference program committees

- Lutz Strassburger was PC member for LICS 2024 and FSCD 2024
- Dale Miller was a PC member of the following meetings: HCVS-2024: Horn Clauses for Verification and Synthesis, Luxembourg, 7 April. • LPAR 2024: 25th International Conference on Logic for Programming, Artificial Intelligence and Reasoning, Mauritius, 26-31 May 2024. • CICM 2024: 17th Conference on Intelligent Computer Mathematics, Montréal, 5-9 August • RAMICS 2024: 21st Conference on Relational and Algebraic Methods in Computer Science, Prague, 19-23 August • NCL 2024: 11th edition of the conference Non-Classical Logics in Łódź, Poland, September 5-8.
- Beniamino Accattoli was a PC member of ICFP 2024 and APLAS 2024, as well as of the Summer School ESSLLI 2024.
- Ambroise Lafont was a PC member of CPP 2025 and JFLA 2025.
- Kaustuv Chaudhuri was a PC member of TABLEAUX 2024 and LFMTP 2024.

Reviewer

- Lutz Strassburger was reviewer for CSL 2024.
- Beniamino Accattoli was reviewer for CSL 2025.
- Ambroise Lafont was reviewer for LICS 2024.

10.1.3 Journal

Member of the editorial boards

- Dale Miller has editorial duties with the following journals:
  - Journal of Automated Reasoning, published by Springer (member of Editorial Board since May 2011).
  - TheoretiCS is a new Diamond Open Access electronic journal covering all areas of Theoretical Computer Science. Member of the Advisory Board, November 2019 – August 2024.
  - Science of Computer Programming, guest co-editor for a special issues of papers selected from FLOPS 2024.

Reviewer - reviewing activities

- Lutz Strassburger was reviewer for the jounals ACM ToCL, JLC, LMCS (2x).
- Beniamino Accattoli was reviewer for LMCS and JFP.

10.1.4 Invited talks

- Lutz Strassburger was invited speaker at the Workshop *Logic at the Interface: Modal Logic and AI*, University of Montpellier, France, 11–12 July 2024.
- Lutz Strassburger was invited speaker at the *6th Proof Society International School and Workshop*, University of Birmingham, UK, 9–13 September 2024.
- Dale Miller gave an invited tutorial at the Days in Logic 2024, IST, University of Lisbon, 1-3 February 2024.
- Dale Miller and Kaustuv Chaudhuri were invited speakers at the Dagstuhl Seminar on "Proof Representations: From Theory to Applications"', 18-23 August 2024 and the PPLV Research Seminar, University College London, 21 March 2024.

10.1.5 Research administration

- Lutz Strassburger is member of the BCEP at Inria Saclay.
- Beniamino Accattoli is member of the "commission scientifique" at Inria Saclay.
- Kaustuv Chaudhuri is a member of the *conseil de laboratoire* (lab council) of LIX.

10.2 Teaching - Supervision - Juries

10.2.1 Teaching

- Werner teaches is in charge of the course "Foundations of Proof Systems" in the joint MPRI Master program (28 hours). He is in charge of the course "Mécanismes d'un langage de programmation orienté-objet" in the cycle ingénieur polytechnicien (10 weeks, 250 students) and of the course "Logic and Proofs" of the BSc program of Ecole polytechnique (16 weeks, 60 students).
- Lafont taught 90 hours in the Bachelors program of Ecole Polytechnique for the following courses: Introduction to algorithms, Logic and proofs, Concurrent and Distributed Computing, Introduction to Formal Languages.
- Zeilberger taught 128 hours in the Bachelors and Polytechniciens programs of Ecole Polytechnique for the following courses: Computer Programming, Introduction to Formal Languages, Functional Programming, Fondements de l'informatique.
- Beniamino Accattoli taught 15h at MPRI.

10.2.2 Supervision

- Lutz Strassburger supervised Samar Rahmouni (stage M1) from Apr 2024 until Jul 2024
- Dale Miller and Noam Zeilberger supervised Aarrya Saraf (bachelor thesis) during Spring 2024.
- Noam Zeilberger supervised Yoshimi-Théophile Etienne (bachelor thesis) during Spring 2024.
- Beniamino Accattoli was the PhD advisor of Adrienne Lancelot.
- Beniamino Accattoli and Dale Miller were the PhD co-advisors of Jui-Hsuan Wu.
- Beniamino Accattoli supervised the internship of Yoan Bouniard (stage M2) from Apr 2024 until September 2024.
- Beniamino Accattoli co-supervised (main advisor Giulio Manzonetto at IRIF) the internship of Marco Santamaria (stage M2) from March 2024 until July 2024.
- Kaustuv Chaudhuri and Dale Miller were the PhD co-advisors of Farah Al Wardani and Arunava Gantait.
- Ambroise Lafont supervised the internship of Samy Avrillon (stage M2) during summer 2024.
- Ambroise Lafont and Benjamin Werner were the PhD co-advisors of Niyousha Najmaei.
- Kaustuv Chaudhuri supervised the bachelor internship of Aya Matmata at the IP Paris.
- Benjamin Werner supervised the internship of Mathis Bouverot (stage M2) from Apr 2024 until September 2024.

10.2.3 Juries

- Lutz Strassburger was external reviewer for the PhD thesis of Victoria Barrett (University of Bath)
- Noam Zeilberger was an external reviewer for the PhD theses of Dimitrios Economou (Queen's University, "Focusing on modular refinement typing"), Jonathan Prieto-Cubides (University of Bergen, "Investigations into Graph-theoretical Constructions in Homotopy Type Theory"), and Matthew Earnshaw (Tallinn University of Technology, "Languages of String Diagrams").
- Dale Miller was a member of ACM's Heidelberg Laureate Forum Young Researcher Selection Committee for three years starting 2023.

11 Scientific production

11.1 Major publications

[1] B. Accattoli, U. D. Lago and G. Vanoni. 'Reasonable Space for the Lambda-Calculus, Logarithmically'. In: *Logical Methods in Computer Science* 20.4 (20th Nov. 2024). DOI: 10.46298/lmcs-20(4:15)2024. URL: https://inria.hal.science/hal-04835903.

[2] F. Al Wardani, K. Chaudhuri and D. Miller. 'About Trust and Proof: An experimental framework for heterogeneous verification'. In: *The Practice of Formal Methods*. Vol. LNCS 14781. Essays in Honour of Cliff Jones, Part II. Springer Nature Switzerland, 1st Sept. 2024, pp. 162–183. DOI: 10.1007/978-3-031-66673-5_9. URL: https://hal.science/hal-04710949.

[3] M. Girlando, R. Kuznets, S. Marin, M. Morales and L. Straßburger. 'A Simple Loopcheck for Intuitionistic K'. In: Logic, Language, Information, and Computation. WoLLIC 2024. Vol. 14672. Lecture Notes in Computer Science. Bern, Switzerland, Switzerland: Springer Nature Switzerland, 8th June 2024, pp. 47–63. DOI: 10.1007/978-3-031-62687-6_4. URL: https://inria.hal.science/hal-04569308.

[4] M. Sadrzadeh and L. Straßburger. 'Lambek Calculus with Banged Atoms for Parasitic Gaps'. In: Logic, Language, Information, and Computation. WoLLIC 2024. Vol. 14672. Lecture Notes in Computer Science. Bern, Switzerland: Springer Nature Switzerland, 8th June 2024, pp. 193–209. DOI: 10.1007/978-3-031-62687-6_13. URL: https://inria.hal.science/hal-04569184.

11.2 Publications of the year

International journals

[5] B. Accattoli, U. D. Lago and G. Vanoni. 'Reasonable Space for the Lambda-Calculus, Logarithmically'. In: *Logical Methods in Computer Science* 20.4 (20th Nov. 2024). DOI: 10.46298/lmcs-20(4:15)2024. URL: https://inria.hal.science/hal-04835903 (cit. on pp. 6, 14).

[6] C. Allain, F. Bour, B. Clément, F. Pottier and G. Scherer. 'Tail Modulo Cons, OCaml, and Relational Separation Logic'. In: *Proceedings of the ACM on Programming Languages* 9.POPL (9th Jan. 2025), pp. 2337–2363. DOI: 10.1145/3704915. URL: https://inria.hal.science/hal-04884634.

[7] P. Donato. 'The Flower Calculus'. In: *Leibniz International Proceedings in Informatics* 9th International Conference on Formal Structures for Computation and Deduction (FSCD 2024).299 (5th July 2024), 5:1–5:24. URL: https://hal.science/hal-04472717 (cit. on p. 15).

[8] D. Miller and A. Momigliano. 'Property-Based Testing by Elaborating Proof Outlines'. In: *Theory and Practice of Logic Programming* (14th June 2024). URL: https://hal.science/hal-04710992. In press (cit. on p. 11).

International peer-reviewed conferences

[9] B. Accattoli. 'Semantic Bounds and Multi Types, Revisited'. In: CSL 2024 - 32nd EACSL Annual Conference on Computer Science Logic. Naples, Italy: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. DOI: 10.4230/LIPIcs.CSL.2024.7. URL: https://inria.hal.science/hal-04837714 (cit. on p. 14).

[10] B. Accattoli and A. Lancelot. 'Light Genericity'. In: FoSSaCS 2024 - 27th International Conference on Foundations of Software Science and Computation Structures. Vol. 14575. Lecture Notes in Computer Science. Luxembourg City, Luxembourg: Springer Nature Switzerland, 6th Apr. 2024, pp. 24–46. DOI: 10.1007/978-3-031-57231-9_2. URL: https://inria.hal.science/hal-04886354 (cit. on p. 13).

[11] B. Accattoli and A. Lancelot. 'Mirroring Call-By-Need, or Values Acting Silly'. In: FSCD 2024 - 9th International Conference on Formal Structures for Computation and Deduction. Tallin, Estonia: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. DOI: 10.4230/LIPIcs.FSCD.2024.23. URL: https://inria.hal.science/hal-04837722 (cit. on p. 13).

[12] B. Accattoli and C. Sacerdoti Coen. 'IMELL Cut Elimination with Linear Overhead'. In: 9th International Conference on Formal Structures for Computation and Deduction (FSCD 2024). Tallin, Estonia: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. DOI: 10.4230/LIPICS.FSCD.2024.24. URL: https://inria.hal.science/hal-04837719 (cit. on p. 13).

[13] B. Accattoli and J.-H. Wu. 'Positive Focusing is Directly Useful'. In: *Proceedings of MFPS XL*. MFPS 2024 - 40th Conference on Mathematical Foundations of Programming Semantics. Oxford, United Kingdom, 11th Dec. 2024. DOI: 10.46298/entics.14758. URL: https://inria.hal.science/hal-04886376 (cit. on p. 13).

[14] V. Barrett, A. Guglielmi and B. Ralph. 'A strictly linear subatomic proof system'. In: CSL 2025 - 33rd EACSL Annual Conference on Computer Science Logic. Amsterdam, Netherlands, 10th Feb. 2025. URL: https://inria.hal.science/hal-04845619 (cit. on p. 10).

[15] M. Girlando, R. Kuznets, S. Marin, M. Morales and L. Straßburger. 'A Simple Loopcheck for Intuitionistic K'. In: Logic, Language, Information, and Computation. WoLLIC 2024. Vol. 14672. Lecture Notes in Computer Science. Bern, Switzerland, Switzerland: Springer Nature Switzerland, 8th June 2024, pp. 47–63. DOI: 10.1007/978-3-031-62687-6_4. URL: https://inria.hal.science/hal-04569308 (cit. on p. 10).

[16] M. Sadrzadeh and L. Straßburger. 'Lambek Calculus with Banged Atoms for Parasitic Gaps'. In: Logic, Language, Information, and Computation. WoLLIC 2024. Vol. 14672. Lecture Notes in Computer Science. Bern, Switzerland: Springer Nature Switzerland, 8th June 2024, pp. 193–209. DOI: 10.1007/978-3-031-62687-6_13. URL: https://inria.hal.science/hal-04569184 (cit. on p. 10).

National peer-reviewed Conferences

[17] C. Allain and G. Scherer. 'Correct tout seul, sûr à plusieurs'. In: 35es Journées Francophones des Langages Applicatifs (JFLA 2024). Saint-Jacut-de-la-Mer, France, 30th Jan. 2024. URL: https://inria.hal.science/hal-04406412.

[18] A. Lafont. 'A diagram editor to mechanise categorical proofs'. In: JFLA 2024 - 35es Journées Francophones des Langages Applicatifs. Saint-Jacut-de-la-Mer, France, 30th Jan. 2024. URL: https://hal.science/hal-04407118 (cit. on p. 15).

Scientific book chapters

[19] F. Al Wardani, K. Chaudhuri and D. Miller. 'About Trust and Proof: An experimental framework for heterogeneous verification'. In: *The Practice of Formal Methods*. Vol. LNCS 14781. Essays in Honour of Cliff Jones, Part II. Springer Nature Switzerland, 1st Sept. 2024, pp. 162–183. DOI: 10.1007/978-3-031-66673-5_9. URL: https://hal.science/hal-04710949 (cit. on p. 11).

Reports & preprints

[20] B. Accattoli and A. Lancelot. *Light Genericity*. 19th Jan. 2024. URL: https://hal.science/hal-04406343 (cit. on p. 13).

[21] B. Accattoli and J.-H. Wu. *Positive Focusing is Directly Useful*. 10th June 2024. URL: https://hal.science/hal-04606194 (cit. on p. 13).

[22] B. Clarke. *Lifting twisted coreflections against delta lenses*. 30th Jan. 2024. URL: https://hal.science/hal-04430822 (cit. on p. 15).

[23] M. Manighetti and D. Miller. *Peano Arithmetic and μMALL*. 4th Nov. 2024. DOI: 10.48550/arXiv.2312.13634. URL: https://hal.science/hal-04824175 (cit. on p. 11).

11.3 Cited publications

[24] M. Girlando, R. Kuznets, S. Marin, M. Morales and L. Strassburger. 'Intuitionistic S4 is decidable'. In: *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. 2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS). Boston, United States: IEEE, June 2023, pp. 1–13. DOI: 10.1109/LICS56636.2023.10175684. URL: https://inria.hal.science/hal-04267899 (cit. on p. 10).

[25] P.-A. Melliès and N. Zeilberger. 'Parsing as a lifting problem and the Chomsky-Schützenberger representation theorem'. In: *MFPS 2022 - 38th conference on Mathematical Foundations for Programming Semantics*. Vol. Volume 1 - Proceedings of... Ithaca, NY, United States, July 2022. DOI: 10.46298/entics.10508. URL: https://hal.science/hal-03702762 (cit. on p. 12).

[26] P.-A. Melliès and N. Zeilberger. 'The categorical contours of the Chomsky-Schützenberger representation theorem'. Dec. 2023. URL: https://hal.science/hal-04399404 (cit. on p. 12).