Scaling Up Polarized Deduction Modulo with Machine Learning

Research Topic for the ParisTech/CSC PhD Program

Subfield: Computer Science, Applied Mathematics

ParisTech School: MINES ParisTech

Title: Scaling Up Polarized Deduction modulo with Machine Learning

Advisor: Olivier Hermant, `http://cri.ensmp.fr/~hermant/`

Keywords: proofs, machine learning, rewriting, theorem provers, proof assistants, first-order logic, type systems, semantics

1 Formal Methods for Safer Software

Software industry produces the most complex objects ever seen. Managing program complexity is hard and frequently results misfunctionments (bugs) and vulnerabilities, whose origins are a misconception or a faulty implementation.

Formal methods help increase quality, especially of critical software (driveless cars, trains, and planes, embarked software). Several techniques exist, like model checking, abstract interpretation, static analysis, proof assistants and automated theorem proving. They have drawn more attention in the past decades, but still lack automation.

This proposal is at the crossroad of proof assistants, automated theorem proving and machine learning, it concerns both implementation, proof theory and learning over a large body of proofs. It aims at enhancing two tools with polarized rewriting: Dedukti (proof checker) and Zenon Modulo (automated theorem prover). Those tools are recent and already used in an industrial platform for safe-by-construction software [The12].

2 The Framework: Deduction Modulo Theory

The gist of Deduction Modulo Theory is to embed computation, via *rewrite rules*, within proof systems. This speeds up provers by avoiding the need for axioms and emphasizing the computational and deterministic nature of parts of proofs. It also offers a versatile and efficient way to express proof assistants in a shallow way, when combine with a type system like LF.

Currently, Deduction Modulo Theory has been successfully implemented in a resolution-based prover [Bur11] and in a tableau-based prover [BDD$^+$15] and has given very promising results [BBC$^+$17, Ji15].

Polarized Deduction Modulo Theory [Dow10] is an improvement over Deduction Modulo Theory [DHK03], that allows rewrite rules to be selectively applied to the hypothesis or to conclusion side of proofs. One of the avantages of this approach is the possibility to express asymetric axioms. Moreover, the generated rewrite system lends itself well to Skolemization, in particular in classical logic, which would further speeds up proof-seach. This is the direction we want to push forward.

3 Research Directions

3.1 Implementing Polarized Deduction Modulo Theory

The implementation of Polarized Deduction Modulo Theory in a tableau-based theorem prover is one goal. The chosen tool is Zenon Modulo [DDG+13], that currently implements *unpolarized* Deduction Modulo Theory. This implementation will be assessed by intensive, industrial, benchmarks, through the TPTP library and the BWare platform [The12].

3.2 Learning Heuristics

To automatically solve a problem, it is of crucial importance to recognize patterns in the problems and to trigger the optimal proof mechanism.

A large body of problems is available, through the TPTP problem set or the BWare industrial platform [The12], we will train machine learning techniques on those sets.

A particular interest is to generate term instances that correspond to the problem to solve.

3.3 Proof Theory

This part involves the definition of models of Polarized Deduction Modulo Theory, for instance by refining the order relation that can be found in common algebras, but also in generalizing Kripke Structures, etc. The impact is soundness and completeness of the calculus wrt to the semantics, and in a second time to derive cut admissibility theorems in the spirit of [Oka99, BH06b, BH06a, LDM05].

Expressiveness of the logic, in particular the possibility to embed constraint [LN07] or higher-order systems [LDM05], can also be envisioned. This can result in the definition of new translations of logics in Dedukti, our proof-checker. A middle-term perspective is to adapt superconsistency [Dow06] to the polarized case.

3.4 Higher-Order Polarization

The polarized approach can be lifted to *type theory*. The goal here is to introduce polarized rewrite rules in a framework like the $\lambda\Pi$-calculus.

The asymetry brought up by polarized rewriting could be used to express subtyping, which is currently missing, and prevents us from expressing some systems. Moreover, it will make possible to double-check the proofs produced by Zenon Modulo.

Therefore an implementation of polarized rewriting in Dedukti [BCH12] is extremely desirable.

4 Required Background of the Student

This proposal are broad and deep, we do not expect a single PhD thesis to cover all these subjects. The focus on a specific area will depend on the applicant and his/her wishes.

An M.Sc.-level specialization in any field of computer science or mathematics is a strong requirement. More specialized courses, among which machine learning, logics (proof systems, proof assistants), rewriting, or functional programming are a plus.

References

[BBC+17] Guillaume Burel, Guillaume Bury, Raphaël Cauderlier, David Delahaye, Pierre Halmagrand, and Olivier Hermant. Automated deduction: When deduction modulo theory meets the practice. 44p. Submitted to Journal of Automated Reasoning., 2017.

[BCH12] Mathieu Boespflug, Quentin Carbonneaux, and Olivier Hermant. The $\lambda\Pi$-calculus modulo as a universal proof language. In *In Second Workshop on Proof Exchange for Theorem Proving (PxTP)*, volume 878, pages 28–43. CEUR-WS.org, 2012.

[BDD+15] Guillaume Bury, David Delahaye, Damien Doligez, Pierre Halmagrand, and Olivier Hermant. Automated deduction in the B set theory using typed proof search and deduction modulo. In Ansgar Fehnker, Annabelle McIver, Geoff Sutcliffe, and Andrei Voronkov, editors, *20th International Conferences on Logic for Programming, Artificial Intelligence and Reasoning - Short Presentations, LPAR 2015, Suva, Fiji, November 24–28, 2015*, volume 35 of *EPiC Series in Computing*, pages 42–58. EasyChair, 2015.

[BH06a] Richard Bonichon and Olivier Hermant. On constructive cut admissibility in deduction modulo. In Thorsten Altenkirch and Conor McBride, editors, *TYPES for proofs and programs*, volume 4502 of *Lecture Notes in Computer Science*, pages 33–47. Springer, 2006.

[BH06b] Richard Bonichon and Olivier Hermant. A semantic completeness proof for tableaux modulo. In Miki Hermann and Andrei Voronkov, editors, *LPAR 2006*, volume 4246 of *Lecture Notes in Computer Science*, pages 167–181, Phom Penh, Cambodia, November 2006. Springer-Verlag.

[Bur11] Guillaume Burel. Experimenting with deduction modulo. In Nikolaj Bjørner and Viorica Sofronie-Stokkermans, editors, *CADE*, volume 6803 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 2011.

[DDG+13] David Delahaye, Damien Doligez, Frédéric Gilbert, Pierre Halmagrand, and Olivier Hermant. Zenon modulo: When achilles outruns the tortoise using deduction modulo. In Ken McMillan, Aart Middledorp, and Andreï Voronkov, editors, *LPAR*, volume 8312 of *LNCS ARCoSS*, pages 274–290. Springer, 2013.

[DHK03] Gilles Dowek, Thérèse Hardin, and Claude Kirchner. Theorem proving modulo. *Journal of Automated Reasoning*, 31:33–72, 2003.

[Dow06] Gilles Dowek. Truth values algebras and normalization. In Thorsten Altenkirch and Conor McBride, editors, *TYPES for proofs and programs*, volume 4502 of *Lecture Notes in Computer Science*, pages 110–124. Springer, 2006.

[Dow10] Gilles Dowek. Polarized deduction modulo. In *IFIP Theoretical Computer Science*, 2010.

[Ji15] Kailiang Ji. CTL model checking in deduction modulo. In Amy P. Felty and Aart Middeldorp, editors, *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings*, volume 9195 of *Lecture Notes in Computer Science*, pages 295–310. Springer, 2015.

[LDM05] James Lipton and Mary De Marco. Completeness and cut elimination in Church's intuitionistic theory of types. *Journal of Logic and Computation*, 15:821–854, December 2005.

[LN07] James Lipton and Susana Nieva. Higher-order logic programming languages with constraints: A semantics. In Simona Ronchi Della Rocca, editor, *TLCA*, volume 4583 of *Lecture Notes in Computer Science*, pages 272–289. Springer, 2007.

[Oka99] Mitsuhiro Okada. Phase semantic cut-elimination and normalization proofs of first- and higher-order linear logic. *Theoretical Computer Science*, 227:333–396, 1999.

[The12] The BWare Project, 2012. `http://bware.lri.fr/`.