First-Order Proof Reconstruction (Research Proposal – 2016)

Andrés Sicard-Ramírez

Seminar of the PhD in Mathematical Engineering
EAFIT University
21 September 2015

Abstract

In a previous research, we proposed a first-order theory for reasoning about functional programs by combining interactive proofs performed in the Agda proof assistant and automatic proofs performed by off-the-shelf first-order automatic theorem provers (ATPs). Our approach can be used with other first-order theories too. We have used it with other first-order theories such as Group Theory and Peano Arithmetic, and we had encouraging results. In our approach, we use the ATPs as oracles via a Haskell program called Apia, that is, we trust the ATPs when they tell us that a proof exists. In consequence, the consistency of our approach relies on the correct implementation of both the Apia program and the ATPs. We propose strengthen the consistency of our approach by reconstructing in Agda the first-order proofs automatically produced.

Project Information

Team work

- Andrés Sicard-Ramírez (main researcher)
- Juan Fernando Ospina-Giraldo (co-researcher)
- Research assistant (student of the Master in Applied Mathematics)

Project Information

Team work

- Andrés Sicard-Ramírez (main researcher)
- Juan Fernando Ospina-Giraldo (co-researcher)
- Research assistant (student of the Master in Applied Mathematics)

Subject

Formalisation of proofs, verification of functional programs, type theory, proof assistants and automatic theorem provers.

Context of Our Research Problem

- Verification of programs

Context of Our Research Problem

- Verification of programs
  - Verification of operational systems

    Example: Gerwin Klein et al. [2010]. seL4: Formal Verification of an Operating-system Kernel. Communications of ACM 53.6, pp. 107–115.

Context of Our Research Problem

- Verification of programs
  - Verification of operational systems

    Example: Gerwin Klein et al. [2010]. seL4: Formal Verification of an Operating-system Kernel. Communications of ACM 53.6, pp. 107–115.
  - Verification of compilers

    Example: CompCert Project (2008 - current)

    Xavier Leroy [2009]. Formal Verification of a Realistic Compiler. Communications of the ACM 52.7, pp. 107–115.

Context of Our Research Problem

- Verification of programs
  - Verification of operational systems

    Example: Gerwin Klein et al. [2010]. seL4: Formal Verification of an Operating-system Kernel. Communications of ACM 53.6, pp. 107–115.
  - Verification of compilers

    Example: CompCert Project (2008 - current)

    Xavier Leroy [2009]. Formal Verification of a Realistic Compiler. Communications of the ACM 52.7, pp. 107–115.
  - Programming logic (a logic in which programs and specifications can be expressed and in which it can be proved or disproved that a certain program meets a certain specification).

Context of Our Research Problem

- Formalisation of proofs / verification of programs

Context of Our Research Problem

- Formalisation of proofs / verification of programs
  - Proof assistant (an interactive computer system which helps with the development of formal proofs).

Context of Our Research Problem

- Formalisation of proofs / verification of programs
  - Proof assistant (an interactive computer system which helps with the development of formal proofs).
  - Dependent types (a dependent type is a type that depend on a value).
  - Π-types

    $\Pi x : A.B(x)$ is the type of terms f such that, for every $a : A$ then $f\ a : B(a)$.
  - Σ-types

    $\Sigma x : A.B(x)$ is the type of pairs (m, n) such that $m : A$ and $n : B(m)$.

Context of Our Research Problem

- Interaction with automatic theorem provers (ATPs)

Context of Our Research Problem

- Interaction with automatic theorem provers (ATPs)
  - ATPs for first-order logic

    The TPTP world (http://www.cs.miami.edu/~tptp/).

Context of Our Research Problem

- Interaction with automatic theorem provers (ATPs)
  - ATPs for first-order logic

    The TPTP world (http://www.cs.miami.edu/~tptp/).
  - Satisfiability modulo theories solvers (SMT Solvers)

Context of Our Research Problem

- Interaction with automatic theorem provers (ATPs)
  - ATPs for first-order logic

    The TPTP world (http://www.cs.miami.edu/~tptp/).
  - Satisfiability modulo theories solvers (SMT Solvers)
  - Apia

    A Haskell program which:

    (i) provides a translation of our Agda representation of first-order formulae into TPTP languages (FOF, TFF0) and

    (ii) calls the ATPs.

Research Problem

Problem

In our approach to the verification of functional programs [Bove, Dybjer, and Sicard-Ramírez 2009, 2012; Sicard-Ramírez 2014], we use the ATPs as oracles via the Apia program, that is, we trust the ATPs when they tell us that a proof exists.

The consistency of our approach relies on the correct implementation of both the Apia program and the ATPs.

We propose strengthen the consistency of our approach by reconstructing in Agda the first-order proofs automatically produced.

Research Problem

Problem

In our approach to the verification of functional programs [Bove, Dybjer, and Sicard-Ramírez 2009, 2012; Sicard-Ramírez 2014], we use the ATPs as oracles via the Apia program, that is, we trust the ATPs when they tell us that a proof exists.

The consistency of our approach relies on the correct implementation of both the Apia program and the ATPs.

We propose strengthen the consistency of our approach by reconstructing in Agda the first-order proofs automatically produced.

Goal

Reconstruct first-order proofs produced by one ATP using Agda as an logical framework.

State of Art

- We do not know of any existing first-order proof reconstruction in the Agda proof assistant.

State of Art

- We do not know of any existing first-order proof reconstruction in the Agda proof assistant.
- Sledgehammer provides a full integration of automatic theorem provers including ATPs for first-order logic and SMT solvers [Blanchette, Böhme, and Paulson 2013] with Isabelle/HOL.

State of Art

- We do not know of any existing first-order proof reconstruction in the Agda proof assistant.
- Sledgehammer provides a full integration of automatic theorem provers including ATPs for first-order logic and SMT solvers [Blanchette, Böhme, and Paulson 2013] with Isabelle/HOL.
- Foster and Struth [2011] integrate Waldmeister into Agda. This integration uses a proof reconstruction step. The approach is restricted to pure equational logic—FOL with equality but no other predicate symbols and no functions symbols [Appel 1959].

State of Art

- We do not know of any existing first-order proof reconstruction in the Agda proof assistant.
- Sledgehammer provides a full integration of automatic theorem provers including ATPs for first-order logic and SMT solvers [Blanchette, Böhme, and Paulson 2013] with Isabelle/HOL.
- Foster and Struth [2011] integrate Waldmeister into Agda. This integration uses a proof reconstruction step. The approach is restricted to pure equational logic—FOL with equality but no other predicate symbols and no functions symbols [Appel 1959].
- SMTCoq [Armand et al. 2011] is a tool for the Coq proof assistant which provides a certified checker for proof witnesses coming from the SMT solver veriT and adds a new tactic named verit, that calls veriT on any Coq goal.

State of Art

- We do not know of any existing first-order proof reconstruction in the Agda proof assistant.
- Sledgehammer provides a full integration of automatic theorem provers including ATPs for first-order logic and SMT solvers [Blanchette, Böhme, and Paulson 2013] with Isabelle/HOL.
- Foster and Struth [2011] integrate Waldmeister into Agda. This integration uses a proof reconstruction step. The approach is restricted to pure equational logic—FOL with equality but no other predicate symbols and no functions symbols [Appel 1959].
- SMTCoq [Armand et al. 2011] is a tool for the Coq proof assistant which provides a certified checker for proof witnesses coming from the SMT solver veriT and adds a new tactic named verit, that calls veriT on any Coq goal.
- Given a fixed but arbitrary first-order signature, Bezem, Hendriks, and de Nivelle [2002] transform a proof produced by the first-order ATP Bliksem in a Coq proof term.

References I

- Appel, K. I. (1959). Horn Sentences in Identity Theory. The Journal of Symbolic Logic 24.4, pp. 306–310.
- Armand, Michael, Germain Faure, Benjamin Grégoire, Chantal Keller, Laurent Théry, and Benjamin Werner (2011). A Modular Integration of SAT/SMT Solvers to Coq through Proof Witnesses. In: Certified Programs and Proofs (CPP 2011). Ed. by Jean-Pierre Jouannaud and Zhong Shao. Vol. 7080. Lecture Notes in Computer Science. Springer, pp. 135–150.
- Bezem, Marc, Dimitri Hendriks, and Hans de Nivelle (2002). Automated Proof Construction in Type Theory Using Resolution. Journal of Automated Reasoning 29, pp. 253–275.
- Blanchette, Jasmin Christian, Sascha Böhme, and Lawrence C. Paulson (2013). Extending Sledgehammer with SMT Solvers. Journal of Automated Reasoning 51.1, pp. 109–128.

References II


Bove, Ana, Peter Dybjer, and Andrés Sicard-Ramírez (2009). Embedding a Logical Theory of Constructions in Agda. In: Proceedings of the 3rd Workshop on Programming Languages Meets Program Verification (PLPV 2009), pp. 59–66.


– (2012). Combining Interactive and Automatic Reasoning in First Order Theories of Functional Programs. In: Foundations of Software Science and Computation Structures (FoSSaCS 2012). Ed. by Lars Birkedal. Vol. 7213. Lecture Notes in Computer Science. Springer, pp. 104–118.


Foster, Simon and Georg Struth (2011). Integrating an Automated Theorem Prover in Agda. In: NASA Formal Methods (NFM 2011). Ed. by Mihael Bobaru et al. Vol. 6617. Lecture Notes in Computer Science. Springer, pp. 116–130.


Klein, Gerwin, June Andronick, Kevin Elphinstone, Gernot Heiser, David Cock, Philip Derrin, Dhammika Elkaduwe, Kai Engelhardt, Rafal Kolanski, Michael Norrish, Thomas Sewell, Harvey Tuch, and Simon Winwood (2010). seL4: Formal Verification of an Operating-system Kernel. Communications of ACM 53.6, pp. 107–115.

References III


Leroy, Xavier (2009). Formal Verification of a Realistic Compiler. Communications of the ACM 52.7, pp. 107–115.


Sicard-Ramírez, Andrés (2014). Reasoning about Functional Programs by Combining Interactive and Automatic Proofs. No publicada. PhD thesis. PEDECIBA Informática. Universidad de la República. Uruguay.

Thanks!