

First Order Proof for Higher Order Logic Theorem Provers (abstract)

Joe Hurd*

Computing Laboratory
University of Oxford,
`joe.hurd@comlab.ox.ac.uk`

Interactive theorem provers are useful for modelling computer systems and then verifying properties of them by constructing a formal proof that the properties logically follow from the definition of the system. The expressivity of higher order logic makes it easy to model systems in a natural way, and there are many interactive theorem provers based on higher order logic, including HOL4 [4], Isabelle [12] and PVS [10]. In these theorem provers the system properties to be formally verified are statements of higher order logic, which are presented to the user as goals. The user proves goals by manually selecting tactics that reduce goals to simpler subgoals, until eventually the subgoals are simple enough that tactics can completely prove them. In general the initial goals corresponding to system properties require some higher order reasoning to prove them (typically an induction), but many subgoals require only first order reasoning and are efficiently proved by a standard first order calculus. Using first order provers to support interactive proof in higher order logic theorem provers has been a productive line of research, and the following is a chronological list of such combinations: FAUST in HOL [9]; SEDUCT in LAMBDA [3]; MESON in HOL [5]; 3TAP in KIV [1]; `blast` in Isabelle [11]; Gandalf in HOL [6]; and Bliksem in Coq [2].

There are two barriers to combining first order provers with interactive higher order theorem provers. The first is the incompatibility of the different logics: a method is required to convert a higher order logic goal to a set of first order clauses, and then to lift a refutation of the clauses to a higher order logic proof. Using the idea of an LCF kernel for first order refutations it is possible to make this logical interface into a module, allowing several different interfaces between first and higher order logic to co-exist [7]. The choice of interface to apply to a particular higher order logic goal depends on both the syntactic structure of the goal and which other interfaces have been tried.

The second barrier is an engineering one: the specifics of how to link up the first order prover and extract the information necessary to reconstruct the refutation and translate it to a higher order logic proof. The LCF kernel design of the logical interface makes it simple to convert refutations to a form in which they can be automatically translated to higher order logic proofs, and thus supports experimentation with a full range of first order calculi. Experiments have shown that resolution is more effective than model elimination for higher order logic

* Supported by a Junior Research Fellowship at Magdalen College, Oxford.

goals [8], and a calculus with specific rules for equality is also important for this application.

All the above ideas are implemented in the Metis proof tactic in the HOL4 theorem prover, which is separately presented as a system description.

References

1. Wolfgang Ahrendt, Bernhard Beckert, Reiner Hähnle, Wolfram Menzel, Wolfgang Reif, Gerhard Schellhorn, and Peter H. Schmitt. Integration of automated and interactive theorem proving. In W. Bibel and P. Schmitt, editors, *Automated Deduction: A Basis for Applications*, volume II, chapter 4, pages 97–116. Kluwer, 1998.
2. Marc Bezem, Dimitri Hendriks, and Hans de Nivelle. Automated proof construction in type theory using resolution. In David A. McAllester, editor, *Proceedings of the 17th International Conference on Automated Deduction (CADE-17)*, volume 1831 of *Lecture Notes in Computer Science*, pages 148–163, Pittsburgh, PA, USA, June 2000. Springer.
3. H. Busch. First-order automation for higher-order-logic theorem proving. In Tom Melham and Juanito Camilleri, editors, *Higher Order Logic Theorem Proving and Its Applications, 7th International Workshop*, volume 859 of *Lecture Notes in Computer Science*, Valletta, Malta, September 1994. Springer.
4. M. J. C. Gordon and T. F. Melham, editors. *Introduction to HOL (A theorem-proving environment for higher order logic)*. Cambridge University Press, 1993.
5. John Harrison. Optimizing proof search in model elimination. In Michael A. McRobbie and John K. Slaney, editors, *13th International Conference on Automated Deduction (CADE-13)*, volume 1104 of *Lecture Notes in Artificial Intelligence*, pages 313–327, New Brunswick, NJ, USA, July 1996. Springer.
6. Joe Hurd. Integrating Gandalf and HOL. In Yves Bertot, Gilles Dowek, André Hirschowitz, Christine Paulin, and Laurent Théry, editors, *Theorem Proving in Higher Order Logics, 12th International Conference, TPHOLS '99*, volume 1690 of *Lecture Notes in Computer Science*, pages 311–321, Nice, France, September 1999. Springer.
7. Joe Hurd. An LCF-style interface between HOL and first-order logic. In Andrei Voronkov, editor, *Proceedings of the 18th International Conference on Automated Deduction (CADE-18)*, volume 2392 of *Lecture Notes in Artificial Intelligence*, pages 134–138, Copenhagen, Denmark, July 2002. Springer.
8. Joe Hurd. First-order proof tactics in higher-order logic theorem provers. In Myla Archer, Ben Di Vito, and César Muñoz, editors, *Design and Application of Strategies/Tactics in Higher Order Logics*, number NASA/CP-2003-212448 in NASA Technical Reports, pages 56–68, September 2003.
9. R. Kumar, T. Kropf, and K. Schneider. Integrating a first-order automatic prover in the HOL environment. In Myla Archer, Jeffrey J. Joyce, Karl N. Levitt, and Phillip J. Windley, editors, *Proceedings of the 1991 International Workshop on the HOL Theorem Proving System and its Applications (HOL '91), August 1991*, pages 170–176, Davis, CA, USA, 1992. IEEE Computer Society Press.
10. S. Owre, N. Shankar, J. M. Rushby, and D. W. J. Stringer-Calvert. *PVS System Guide*. Computer Science Laboratory, SRI International, Menlo Park, CA, September 1999.

11. L. C. Paulson. A generic tableau prover and its integration with Isabelle. *Journal of Universal Computer Science*, 5(3), March 1999.
12. Lawrence C. Paulson. Isabelle: A generic theorem prover. *Lecture Notes in Computer Science*, 828:xvii + 321, 1994.