Integrating Automated and Interactive Theorem Proving*

Wolfgang Ahrendt[1], Bernhard Beckert[1], Reiner Hähnle[1], Wolfram Menzel[1], Wolfgang Reif[2], Gerhard Schellhorn[2] and Peter Schmitt[1]

[1] Universität Karlsruhe, Inst. für Logik, Komplexität und Deduktionssysteme, D-76128 Karlsruhe
[2] Universität Ulm, Abt. Programmiermethodik, D-89069 Ulm

We present a project to *integrate interactive and automated theorem proving.* Its aim is to combine the advantages of the two paradigms. We focus on one particular application domain, which is deduction for the purpose of software verification. Some of the reported facts may not be valid in other domains. We report on the integration concepts and on the experimental results with a prototype implementation.

Automatic provers are very fast for the majority of the problems they can solve at all. With increasing complexity, response time increases dramatically. Beyond a certain problem size, automated theorem provers produce reasonable results only in very exceptional cases. Interactive theorem provers on the other hand can be used even in very large case studies. For small problems, they do, however, require many user interactions, particularly when combinatorial exhaustive search has to be performed.

Concerning software verification and the typical proof tasks arising there, the gap between both methods (if applied naively) is even more dramatic. There are essentially two reasons for that phenomenon. First, the theories occurring in verification projects are very large (hundreds of axioms). Second, the majority of these axioms use equality. Such theories are not well handled by automatic provers. We present techniques to relieve these problems.

We investigate a conceptual integration of interactive and automated theorem proving for software verification that goes beyond a loose coupling of two proof systems. Our concrete application domain turned out to have an enormous influence on the integration concepts. We have implemented a prototype system combining the advantages of both paradigms. In large applications, the integrated system incorporates the proof engineering capabilities of an interactive system and, at the same time, eliminates user interactions for those goals that can be solved by the efficient combinatorial proof search embodied in an automated prover. We report on the integration concept, on the encountered problems, and on experimental results with the prototype implementation. Furthermore, the current directions of our ongoing research are described.

The technical basis for the integration are the systems KIV [3] and $_3T^AP$ [2], both of which were developed in the research groups of the authors at Ulm and Karlsruhe. KIV ("Karlsruhe Interactive Verifier") is an advanced verification

* Research supported by DFG, grant No. Re 828/2-2.

system which has been applied in large realistic case studies in academia and industry for many years now. $_3TAP$ ("Three-valued Tableau-based Automated Theorem Prover") is an automated tableau prover for full first-order logic with equality. It does not require normal forms, and it is easily extensible. Although we experimented with these particular systems, the conceptual results carry over to other provers.

Based on statistics from case studies in KIV, we estimate that in our application domain up to 30% of all user interactions needed by an interactive prover could be saved in principle by a first-order theorem prover. Current provers, however, are far from this goal, because they are in general not prepared for deduction in large software specifications (i.e., very large search spaces) or for typical domain specific reasoning. We describe these and other problems, and present the solutions we came up with so far.

Many of our decisions are based on experimental evidence. Therefore, we put a lot of effort in a sophisticated verification case study: Correct compilation of Prolog programs into Warren Abstract Machine code ([4]). We use it as a reference or benchmark. Parts of it are repeated every now and then to evaluate the success of our integration concepts.

In realistic applications in software verification, proof attempts are more likely to fail than to go through. This is because specifications, programs, or user-defined lemmas typically are erroneous. Correct versions usually are only obtained after a number of corrections and failed proof attempts. Therefore, the question is not only how to produce powerful theorem provers but also how to integrate proving and error correction. Current research on this and related topics is discussed.

References

1. W. Ahrendt, B. Beckert, R. Hähnle, W. Menzel, W.Reif, G. Schellhorn, and P. Schmitt. Integration of Automated and Interactive Theorem Proving. In W. Bibel and P. Schmitt, editors, *Automated Deduction — A Basis for Applications*, volume II, 4. Kluwer, 1998.
2. B. Beckert, R. Hähnle, P. Oel, and M. Sulzmann. The tableau-based theorem prover $_3TAP$, version 4.0. In M. McRobbie and J. Slanley, editors, *Proc. 13th CADE, New Brunswick/NJ, USA*, LNCS 1104, pages 303–307. Springer-Verlag, 1996.
3. W. Reif. The KIV-approach to software verification. In M. Broy and S. Jähnichen, editors, *KORSO: Methods, Languages, and Tools for the Construction of Correct Software—Final Report*, LNCS 1009. Springer-Verlag, 1995.
4. G. Schellhorn and W. Ahrendt. Reasoning about Abstract State Machines: The WAM Case Study. *Journal of Universal Computer Science (JUCS)*, 3(4):377–380, 1997. Available at the URL: http://hyperg.iicm.tu-graz.ac.at/jucs/.