Constructor-based Theorem Prover

Daniel Găină[1]

[1]Japan Advanced Institute of Science and Technology
daniel@jaist.ac.jp

Constructor-based Inductive Theorem Prover (CITP) is a proof management tool built on top of an algebraic specification language. This means that there exists a intimate connection between the modeling technique and the verification approach. The methodology supported by the tool is not intended for formalizing mathematics, but for the application to the development of software systems. In order to achieve the targeted goal, two important research directions are pursued: (1) proposing more expressive logical systems to allow engineers to specify easily and accurately the software systems, and (2) develop decision procedures that can reason efficiently about these more sophisticated logics.

The formal language of CITP is based on a variation of conditional equational logic [4] with sub-sorting relations [5], transitions [3] and constructor operators [6]. The logical formulas are constructed from atoms given as equations, membership relations [8] and transitions by applying logical implications and universal quantification. It follows that the specifications of CITP are executable by rewriting [1], which means a high degree of automations of the verification process. The formal language is equipped with specification building operators [10] in the algebraic specification tradition.

Inductive theorem provers are interactive, in general [2, 9, 7]. It is required a trained user to direct the theorem prover towards discharging the goals which cannot be proved by automatic techniques. In many cases, the tool needs the user's help to perform trivial proofs. One major direction of research in inductive theorem proving is improving and reducing the user interaction. This issue is tackled in the present contribution from the following angles: (1) better strategies for generating induction schemes, (2) improved decision procedures to perform automated reasoning, and (3) improvements of the proof assistant interface to help the user understand the current state of the proof and interact with the tool in a more natural way.

The CITP's source code has been (1) refactored to be more readable and (2) improved to make it a better platform for future extensions. The command parsing component was reimplemented to generate better error messages.

References

[1] F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, New York, NY, USA, 1998.

[2] Y. Bertot and P. Castéran. *Interactive Theorem Proving and Program Development - Coq'Art: The Calculus of Inductive Constructions.* Texts in Theoretical Computer Science. An EATCS Series. Springer, 2004.

[3] R. Diaconescu and K. Futatsugi. *Cafeobj Report - The Language, Proof Techniques, and Methodologies for Object-Oriented Algebraic Specification*, volume 6 of *AMAST Series in Computing.* World Scientific, 1998.

[4] J. A. Goguen and J. Meseguer. Completeness of many-sorted equational logic. *SIGPLAN Notices*, 17(1):9–17, 1982.

[5] J. A. Goguen and J. Meseguer. Order-sorted algebra I: equational deduction for multiple inheritance, overloading, exceptions and partial operations. *Theor. Comput. Sci.*, 105(2):217–273, 1992.

[6] D. Găină, K. Futatsugi, and K. Ogata. Constructor-based logics. *J. UCS*, 18(16):2204–2233, 2012.

[7] J. D. Hendrix. *Decision Procedures for Equationally Based Reasoning.* Technical report, UIUC, 2008.

[8] J. Meseguer. Membership algebra as a logical framework for equational specification. In F. Parisi-Presicce, editor, *Recent Trends in Algebraic Development Techniques, 12th International Workshop, WADT'97, Tarquinia, Italy, June 1997, Selected Papers*, volume 1376 of *Lecture Notes in Computer Science*, pages 18–61. Springer, 1997.

[9] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL - A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science.* Springer, 2002.

[10] D. Sannella and A. Tarlecki. Specifications in an arbitrary institution. *Inf. Comput.*, 76(2/3):165–210, 1988.