The Natural Style Provers of TH∃OREM∀: A Survey of Strategies for Different Mathematical Domains*

B. Buchberger, C. Dupré, T. Jebelean, B. Konev, F. Kriftner, T. Kutsia, K. Nakagawa, F. Piroi, D. Văsaru, W. Windsteiger

RISC–Linz, A-4232 Hagenberg, Austria
theorema@theorema.org

The TH∃OREM∀ project aims at supporting, within one consistent logic and one coherent software system, the entire mathematical exploration cycle consisting of: introducing new notions (functions, predicates, algorithms) by definitions or axioms; conjecturing and proving / disproving mathematical facts about the new notions; extracting methods and algorithms from proved facts; and applying algorithms to input data.

The TH∃OREM∀ system is implemented using the computer algebra system *Mathematica* and it is based on earlier work of Buchberger [1, 2]. The first version of the system is described in [5] and other papers (see www.theorema.org), and the current version (in particular with more details about the features presented in this summary) is presented in [4].

In this presentation we will concentrate on the various proving methods currently implemented in TH∃OREM∀. The objectives and features that distinguish proving in our system from other automatic reasoning projects are:

- Priority of "theory exploration" over "theorem proving", i.e. our goal is to support the formal development and proving of portions of coherent mathematical theories rather than in proving isolated theorems, see [3].
- We consider both computer-support in the "routine" reductions of proof problems to known black-box techniques and also the computer-supported invention of "ingenious" proofs by strong special techniques (e.g. algebraic techniques) based on nontrivial mathematics.
- We emphasize automated proof generation over automated proof checking.

*This research is partially supported by the INTAS project 76-960, the Austrian Science Foundation (FWF-SFB-project FO-1302), and the Upper Austrian Government (project "PROVE").

- We emphasize, in automated routine proofs, the importance of generating human-readable proof presentations in a “natural style”.
- We integrate many different general and special provers into one system, i.e. TH∃OREM∀ is a multi-method system.
- We emphasize the importance of convenient, two-dimensional, extensible syntax and graphical illustration.

The provers of the system are implemented using rewrite systems and operate under a general control mechanism (**meta-prover**) capable of combining various proving methods. The trace of the proof and the open subgoals are managed by the meta-prover using an internal representation (**proof-object**) organized as and AND–OR tree. At each step, the prover[s] are trying to solve a certain **proof-situation** (similar to sequent: has some current **assumptions**, but only one formula on the right-hand side: the **goal**).

The proving facilities which are currently implemented in TH∃OREM∀ are listed in the sequel.

The general propositional-logic prover uses a number of rules similar to sequent calculus and inspired by the natural style of proving by the working mathematician. As a general strategy, the assumptions and the goal are split into smaller subformulae, until the success or the failure is obvious (**final proof-situation**).

The general predicate-logic prover uses (besides the propositional rules from above) some “simple” rules involving quantifiers, which in general simplify the proof-situation (e. g. introducing new constants for the universal goal). The non-trivial proof-search starts when such “simple” rules are not applicable, and for this we implemented several strategies: level-saturation using a general inference mechanism - both for forward and for backward reasoning; usage of meta-variables for finding appropriate substitutions; as well as combinations of these.

The PCS prover is a new predicate logic prover that integrates proving, simplifying and solving steps in a systematic way. This prover uses first similar “simple” rules as the general predicate prover (the *prove* cycle), then it uses rewrite rules extracted from universal assumptions for rewriting the formulae from the proof-situation (the *compute* cycle), and finally it introduces special *solving constants* similar to meta-variables and finds appropriate relations on them, leading to the *solve* cycle.

Several inductive provers for equalities are constructed by combining induction and simplification over the domains of integers and lists. In addition, definitions using *case-distinction* are used by special inference rules.

The set theory prover extends the PCS prover by special inference rules appropriate for reasoning with sets (set-equality, inclusion, membership, etc.).

Some special provers are implemented for special domains (closed real fields, integers) by using special powerful techniques as black-box engines which are called at specific points in the proof development process: the Groebner Bases

prover for boolean combinations involving polynomial equalities and inequalities and the Gosper-Zeilberger prover [6] for problems involving combinatorial identities over integers.
Special interfaces are implemented to several other "external" predicate logic provers: Otter, EQP, Gandalf, Bliksem, and Mace.
The cascade strategy is used for extending the capabilities of existing provers. This strategy consists in inferring new subgoals from the information generated by failed proofs and calling recursively the same prover again. When applied to induction-simplification provers this strategy produces a powerful new prover which is able to generate new lemmata for a give theory.
Proof simplification is performed over the generated proof-object in order to produce more natural proofs. Simplification currently includes: elimination of unnecessary branches and inference steps, removal of duplicate steps which appear on parallel branches, combining often–used proof steps into one, etc.

The presentation will be accompanied by an on-line demonstration of the working system.

References

[1] BUCHBERGER, B. Mathematica: A System for Doing Mathematics by Computer? In *Advances in the Design of Symbolic Computation Systems*, A. Miola and M. Temperini, Eds. Springer New York - Vienna, 1993.

[2] BUCHBERGER, B. Using Mathematica for Doing Simple Mathematical Proofs. In *Proceedings of the 4th Mathematica Users' Conference, Tokyo, November 2, 1996* (1996), Wolfram Media Publishing. To appear.

[3] BUCHBERGER, B. Theory Exploration versus Theorem Proving. In *Proceedings of CALCULEMUS: Systems for Integrated Computation and Deduction* (Trento, Italy, 1999), A. Armando and T. Jebelean, Eds., Electronics Notes in Theoretical Computer Science, Elsevier.

[4] BUCHBERGER, B., DUPRÉ, C., JEBELEAN, T., KRIFTNER, F., NAKAGAWA, K., ASARU, D. V., AND WINDSTEIGER, W. The Theorema Project: A Progress Report. In *Proceedings of Calculemus 2000* (St. Andrews, UK, 2000), M. Kohlhase and M. Kerber, Eds., Electronic Notes in Computer Science.

[5] BUCHBERGER, B., JEBELEAN, T., KRIFTNER, F., MARIN, M., TOMUŢA, E., AND VĂSARU, D. An Overview of the *Theorema* Project. To appear in proceedings of ISSAC 1997, 1997.

[6] PAULE, P., AND SCHORN, M. A Mathematica Version of Zeilberger's Algorithm for Proving Binomial Coefficient Identities. *J. Symbolic Computation 20* (1995), 673–698.