The Tableau–Based Theorem Prover $_3T^AP$ for Multiple–Valued Logics[*]

Bernhard Beckert   Stefan Gerberding
Reiner Hähnle   Werner Kernig

Institute for Logic, Complexity and Deduction Systems
University of Karlsruhe, 7500 Karlsruhe, Germany
{`beckert,gerberding,haehnle,kernig`}`@ira.uka.de`

$_3T^AP$ is an acronym for **3**–valued **ta**bleau–based theorem **p**rover. It is based on the method of analytic tableaux. $_3T^AP$ has been developed at the University of Karlsruhe in cooperation with the Institute for Knowledge Based Systems of IBM Germany in Heidelberg. Despite its name $_3T^AP$ is able to deal with "classical" — i.e. two–valued — first–order predicate logic as well as with any finite–valued first–order logic, provided the semantics is specified by truth–tables. Currently implemented versions are working for two–valued and for a certain three–valued first–order predicate logic, which is a variant of the strong Kleene logic, see [3]. The multiple–valued version implements the concept of *generalized signs*. These may be seen as sets of ordinary tableau signs or prefixes, see [6] and [7] for details. Without generalized signs one has to build a separate tableau for each non–designated sign to refute a formula. $_3T^AP$ needs to close only one tableau using generalized signs. The system has been implemented in Quintus Prolog and is running on SUN and IBM PS/2. The use of Prolog and the modular design makes it easy to extend or modify the prover.

$_3T^AP$'s input is given by a set of axioms and theorems contained in a database file, which can be precompiled. The user may specify a theorem to be proved or (s)he proves the consistency of the database. The formulae need not to be in normal form, hence the database remains readable. The use of a sort hierarchy in the database to distinguish terms of different nature is supported. For inspection of proofs some output formatting utilities allow the indentation of a proof or the conversion to LaTeX–syntax. During the compilation of a database static links are being built. These links are used to decide whether a formula can possibly lead to a branch closure and needs to be used in the tableau or not.

The free variable tableau calculus used by $_3T^AP$ is basically the one described in [10] or [4] with a few modifications concerning the handling of free variables and the extension to the multiple–valued case.

Through branch closures free variables become instantiated and they are used up. Since a formula is often needed with a different substitution of free variables, a mechanism has been employed that allows to mark formulae as *universal* with respect to a variable occurring freely in it. Basically, this is the case when the formula can

[*]This work has been supported by IBM Germany.

be deduced from the γ–formula that created the free variable without causing any branching of the tableau. Variables with respect to which formulae are marked as universal do not become instantiated when they are used in a branch closure. This feature allows $_3T^AP$ to find shorter proofs for many theorems.

The quantified variable x in a δ–formula is substituted by a new Skolem function, which depends on exactly the free variables in that formula except x. This version of the δ–rule generalizes the one in [4], where the newly introduced Skolem function depends on all free variables introduced so far on the current branch, i.e. our Skolem function depends on fewer variables and — most important for implementation — one needs not to keep track of the variables introduced on the branch in focus. That the liberalized δ–rule preserves completeness is obvious, soundness is proved in [8].

Another feature of $_3T^AP$ may also shorten proofs: The generation of lemmata. Lemmata may be generated for tableau rules with more than one extension in their conclusions. If the extensions $E_1, \dots, E_n$ are not logically disjoint and some of the corresponding branches have already been closed, one can retrieve information from the intersection of the already closed branches corresponding to, say, $E_{i_1}, \dots, E_{i_k}$. From this information one may extract a formula, which may be added to the next branch, the one corresponding to $E_{i_{k+1}}$ say, which possibly rules out some models for that branch and thus shortens the proof. The generation of lemmata can also be seen as a kind of analytic cut rule. It lifts proof length complexity in the propositional case, as is proved for a similar system in [2].

$_3T^AP$ is able to deal with equality [1], which is treated as a two–valued predicate in the multiple–valued case. Many–valued similarity predicates are not yet supported.

For efficiency reasons not the whole tableau is kept in memory, but only the branch in focus is stored. If a branch has been closed it is discarded.

The classical version of $_3T^AP$ is able to solve most of the problems stated in [9] in less than one second on a SUN–4.

References

[1] Bernhard Beckert. Konzeption und Implementierung von Gleichheit für einen tableau–basierten Theorembeweiser. Studienarbeit, Fakultät für Informatik, Universität Karlsruhe, July 1991.

[2] Marcello D'Agostino. *Investigations into the Complexity of some Propositional Calculi.* PhD thesis, Oxford University Computing Laboratory, Programming Research Group, November 1990. Also Technical Monograph PRG–88, Oxford University Computing Laboratory.

[3] Jens Erik Fenstad, Per-Kristian Halvorsen, Tore Langholm, and Johan F.A.K. van Benthem. Equations, schemata and situations: A framework for linguistic semantics. Technical Report CSLI–85–29, Center for the Studies of Language and Information Stanford, 1985.

[4] Melvin C. Fitting. *First–Order Logic and Automated Theorem Proving.* Springer, New York, 1990.

[5] Reiner Hähnle. Spezifikation eines Theorembeweisers für dreiwertige First–Order Logik. IWBS Report 136, Wissenschaftliches Zentrum, IWBS, IBM Deutschland, 1990.

[6] Reiner Hähnle. Towards an efficient tableau proof procedure for multiple–valued logics. In *Proceedings Workshop on Computer Science Logic, Heidelberg*. Springer, LNCS 533, 1990.

[7] Reiner Hähnle. Uniform notation of tableaux rules for multiple–valued logics. In *Proc. International Symposium on Multiple–Valued Logic, Victoria*. IEEE Press, 1991.

[8] Reiner Hähnle and Peter H. Schmitt. The liberalized δ–rule in free variable semantic tableaux. *To appear*, 1991.

[9] Francis Jeffry Pelletier. Seventy-five problems for testing automatic theorem provers. *Journal of Automated Reasoning*, 2:191 – 216, 1986.

[10] Peter H. Schmitt. The THOT theorem prover. Technical Report TR–87.09.007, IBM Heidelberg Scientific Center, 1987.