The Design of the CADE-16 Inductive Theorem Prover Contest

Citation for published version:
Bundy, A & Hutter, D 1999, The Design of the CADE-16 Inductive Theorem Prover Contest. in Automated Deduction — CADE-16: 16th International Conference on Automated Deduction Trento, Italy, July 7–10, 1999 Proceedings. Lecture Notes in Computer Science, vol. 1632, Springer-Verlag GmbH, pp. 374-377. DOI: 10.1007/3-540-48660-7_33

Digital Object Identifier (DOI):
10.1007/3-540-48660-7_33

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Early version, also known as pre-print

Published In:
Automated Deduction — CADE-16

The Design of the CADE-16 Inductive Theorem Prover Contest

Dieter Hutter[1] and Alan Bundy[2]

[1] German Research Center for Artificial Intelligence, Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany, hutter@dfki.uni-sb.de
[2] The University of Edinburgh, Division of Informatics, 80 South Bridge, Edinburgh EH1 1HN, Scotland, bundy@dai.ed.ac.uk

> *"Progress in science is only achieved through careful analysis of methods and their power. Designing without analysis is idle speculation. Implementation without analysis is tinkering. Alone they have no research value. All too often we read of major pieces of AI work that stop with the engineering steps. But we need to know how the implemented program embodies the designs and that the program works well because of the design."* B.G. Buchanan [Bu88]

1 Introduction

It has become a tradition at CADE to run a competition for first-order automated theorem provers based on the TPTP problem library. This competition (CASC) [SuSu96] aims at fully automatic ATP systems and provides various categories dedicated to different problem classes of first-order logic. The number of problems solved and the runtime is used to assess the winners in the individual categories of the competition.

In the past considerable effort has been spent to discuss the issues of an inductive theorem proving competition but problems intrinsic to inductive reasoning have prevented a CASC-like competition. Inductive reasoning introduces additional search control problems to those already present in first-order theorem proving. The need for the selection of an appropriate induction order or for the speculation of intermediate lemmata results in infinite branching points in the search space because (technically speaking) the cut rule cannot be eliminated in inductive theories. As a consequence, all current inductive theorem proving systems are necessarily more or less interactive systems. For instance, recursively defined functions can be used to introduce new well-founded orderings to the prover, necessary intermediate lemmata have to be proven as theorems beforehand in order to make them known to the prover. Furthermore, many systems allow the user to complement the problem description by additional strategic information (like the so-called hints in NQTHM [BoMo79]) or allow him to build up a proof in an interactive way. In contrast to first-order theorem proving, inductive theorem proving still depends on the users help. Proving complex inductive theorems depends on the experience and the skills of the human user. Thus besides their degree of automation, inductive theorem proving systems should also be judged on their abilities to assist the user in the organisation and tracking of complex proof systems [PrSl94].

The time which is necessary to adjust the problem set to the individual inductive theorem provers, to perform the proofs, and to evaluate the results, implies that the competition must take place already before the CADE-conference. Thus, the contest is held over the internet. Various problem sets are emailed to the contest participants. Each participant of the competition treats the problem sets individually and has to provide a detailed report on the behaviour of its system wrt. the individual problems. Each team is also expected to write a detailed report explaining their results and allowing to compare the abilities of the systems wrt. different issues.

The success of an inductive theorem proving system depends on the appropriate formulation of the problems. Different systems provide different aids for the user to control the proof search. They also value differently the individual issues in automating inductive theorem proving. As a consequence, there is no winner of the competition since it seems to be impossible to weigh the different degrees of help, the user has given the system by his individual formulation of the problems. Instead the competition results in a detailed comparison of the participating systems with respect to the several issues. During the FLoC-workshop 13 "Automation of Proofs by Mathematical Induction" selected problems are tackled again (online) by all of the systems in order to study the behaviour of the systems in more detail.

2 Problem Selection and System Evaluation

The competition aims at an evaluation of the capabilities of inductive theorem proving systems with respect to various research problems. A "perfect" system would have to cope with a large variety of issues, such as for instance program synthesis, proving the termination of programs, or providing a sophisticated human-computer interface. Since existing systems usually support only parts of these issues, the selection of specific problem classes to be part of the competition would seriously affect the result of the competition. Hence the design of the competition takes this problem into account by the following means:

The problem set is divided into two categories. The first category aims at "core" issues common to all inductive theorem proving systems. This comprises the selection of an appropriate induction scheme or case analysis and the guidance of the resulting inductive proof obligation. These problems are adapted from a modified version of the NQTHM-92 release. This corpus is based on an $\exists$-quantifier free language and does not incorporate any synthesis problems. Within this category, about a hundred problems from this database are chosen to measure the automation of the various systems. The competition focuses on the number of problems which are solved automatically by the systems and the time they require to solve them.

The second category is dedicated to specific issues of inductive theorem proving mentioned earlier in this paper. Examples are program synthesis (or proving existentially quantified formulas respectively), proving the termination of programs, computing appropriate case analyses and induction schemes, or general issues of first-order theorem proving within inductive proofs. The competition provides specific (but small) problem sets dedicated to these problem classes. In contrast to the first, this category focuses on a comparison *how* different systems tackle these problems. Therefore the participants are requested to fill in a detailed questionnaire to illustrate the degree of automation, the used heuristics and methods, and the limitations of the chosen approach.

3 Problem Presentation

As mentioned before, the competition aims at a variety of different issues in inductive theorem proving. In order to cope with such problems (e.g. computing induction schemes, lemma speculation or proving the termination of algorithms), strategic knowledge of the user is incorporated into the way a particular problem is specified inside a system. Unfortunately the systems vary on the way how (and how much) proof knowledge is given to them. Additionally different provers are based on different logics. While for example NQTHM [BoMo79] is based on a non-sorted logic, CLAM [BvHHS91] uses Martin-Löf type theory. There is no uniform language which suits all of the provers. Thus, the participants are allowed to translate the problem sets into a logic accepted by their system but they are also requested to record this translation in their report.

3.1 Input Language

The principle of induction is strongly correlated to the semantic notion of generated algebras. These generated algebras are typically specified by providing a signature (a set of so-called constructor functions), the terms of which define the objects under consideration. In case of freely generated algebras two different (constructor-) ground terms always denote different objects. For instance natural numbers or lists are typically specified as freely-generated datatypes (based on the signatures $\{0, s\}$ and $\{nil, cons\}$ respectively). The specification of other datatypes like integers or finite sets requires the identification of different constructor-terms. Dealing with non-freely generated datatypes complicates the search for well-founded orderings which is one reason why only a small number of systems allow for these datatypes. Therefore in the first problem category the contest is restricted to freely-generated datatypes. For similar reasons mutually recursive data structures (like for instance terms and termlists) are not used within this category. Nevertheless there are special problems in the second category which are dedicated to these problems.

Inductive theorem proving systems have to create appropriate induction schemes for the given proof obligations. In general this constitutes an infinite branching point. In order to overcome this problem many systems use the recursion ordering of constructively defined functions to formulate appropriate induction schemes. Thus these systems provide schemes to specify functions in a constructive way which provides the systems with both, a set of logical axioms and a specification of well-founded orderings if the denoted algorithm can be proven to be terminating. Although the problem sets makes no use of such specific definition principles but consists of first-order axiomatisation, most of them can be encoded into such a framework. The participants are free to use such definition principles but again have to state this translation in their reports.

4 Conclusion

The CADE-16 contest on inductive theorem proving system is a first approach to compare and evaluate the different systems in this area. The motivations for running such a competition are to evaluate inductive theorem provers, to identify possible improvements and open research problems, and to construct a problem library for inductive theorem provers. Although we are aware of the limitations of this contest we see a clear potential for future competitions which will be improved with respect to the problem library and also with respect to a more granular evaluation of the experimental results.

Acknowledgements. We would like to thank David McAllester for initiating the discussion about a contest and for providing a first version of the problem library. We are also indebted to Ian Green who translated this first version of a library into a sorted language.

References

[BoMo79] B. Boyer and J S. Moore A Computational Logic. *Academic Press*, 1979

[Bu88] B.G. Buchanan: Artificial Intelligence as an Experimental Science. *In J.H. Fetzer (ed.): Aspects of Artificial Intelligence*, pp. 209-250, Kluwer Academic Publishers, 1988

[BvHHS91] A. Bundy, F. van Harmelen, J. Hesketh, and A. Smaill. Experiments with proof plans for induction. *Journal of Automated Reasoning*, 7:303-324, 1991

[PrSl94] C. Prehofer and K. Slind: Theorem Proving in Interactive Verification Systems. *Proceedings of the CADE-12 Workshop: Evaluation of Automated Theorem Proving Systems*, 1994

[SuSu96] C. Suttner and G. Suttcliff: The Design of the CADE-13 ATP System Competition. *Proceedings of the 13th International Conference on Automated Deduction, CADE-13*, Springer LNAI 1104, New Brunswick, USA, 1996