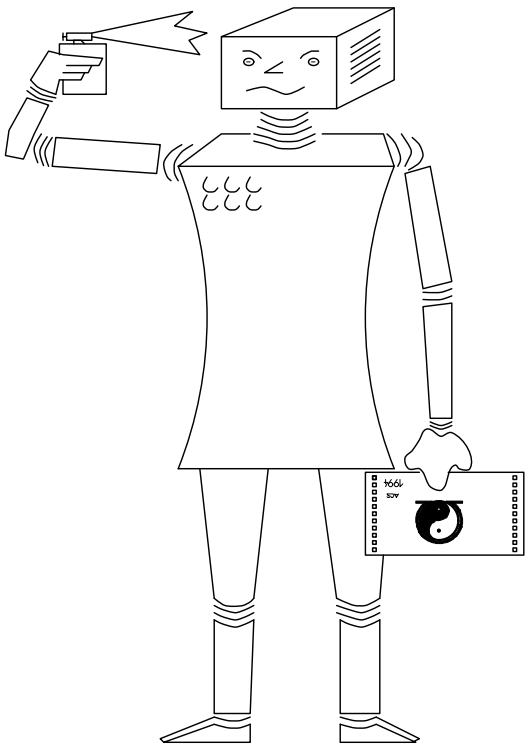

A Test for Evaluating the Practical Usefulness of Deduction Systems

Xiaorong Huang Manfred Kerber
Michael Kohlhase Daniel Nesmith
Jörn Richts

Published as: In Geoff Sutcliffe, Christian B. Suttner, eds., *Proceedings of the CADE-Workshop on Evaluation of Automated Theorem Proving Systems*, Nancy, France, p.33–36, 1994.

A Test for Evaluating the Practical Usefulness of Deduction Systems

Xiaorong Huang, Manfred Kerber, Michael Kohlhase,
Daniel Nesmith, Jörn Richts
FB Informatik, Universität des Saarlandes
66041 Saarbrücken, Germany
{huang,kerber,kohlhase,nesmith,richts}@cs.uni-sb.de

The discussion on the evaluation of theorem provers, especially in the community concentrating on fully automatic systems has in the past primarily been concerned with evaluating only two parameters:

- The success on sets of classical benchmark examples (like the sets of Pelletier, Wos, or the TPTP library[SSY94])
- and the minimal time used for the proof of these examples.

While these parameters certainly are of great interest to the community of deduction system engineers, they do not give a complete or meaningful account of a given deduction system for practical applications. For this purpose the evaluation should be centered more around the needs of the user of a deduction system. It seems that the field of deduction systems is now mature enough (systems like NQTHM, HOL, VSE (INKA/KIV) are migrating into commercial applications) that other evaluation criteria should be discussed and tried.

We propose a comprehensive test PUTEST (Test of Practical Usefulness) for the practical usefulness of deduction systems. This test is modeled after tests for commercial software, such as word processors, desktop publishing systems and database programs commonly found in computer periodicals. These tests[Die93] give a typical application problem to a team consisting of a typical user and the software product and then evaluate the performance of the team in terms of elapsed time until a solution is found, and the quality of the result.

We will now briefly discuss the application of this test plan to deduction systems and discuss the merits and shortcomings of this approach.

For a concrete PUTEST situation we would give a deduction problem formalized in natural language (e.g. think of a problem from a typical mathematics textbook or a riddle from today's newspaper) to a team consisting of a user and a deduction system. Then we would ask the user to come up with a proof of the problem in some

proof formalism. Note that we do not insist that the proof necessarily is in the proof formalism provided by the deduction system.

- Our deduction problems are stated in a human-oriented way (natural language, or mathematical vernacular), which is the way most practical problems occur in the world. The need to formalize the problem in the specific input language of a deduction system includes a test of adequacy of the language for the task at hand. In contrast to tests that only rely on the search time our test is fair to deduction systems that invest on adequate input languages at the cost of a reduced efficiency (in terms of time) of the proof search. It seems that our test will find systems that have found a good trade-off between the expressive power of the working language and the speed of proof search.

- In our view the key merit of the proposed test is that it evaluates a deduction system from the point of view of a potential user by giving an assessment of user productivity by measuring the time of the whole problem solving cycle. The test addresses the questions, "If I want to use the system, how long will it take me to get results?" and "what will the quality of the results be?". User productivity is the measure which in the long run will determine the propagation of deduction systems and ensure the future of the community.

- Last but not least, PUTEST allows the comparison of interactive systems with fully automatic ones. A comparison on terms of search speed or the set of provable theorems cannot be meaningful for this task, since it does not apply to interactive systems. Even for fully automated systems, a ranking by search time alone sometimes gives a misleading picture, since very fast search times can often only be reached by investing a great amount of time into fine-tuning a system interactively for the special example at hand. Incidently these times are not published, but would be measured by PUTEST. On the other hand the rate of success on standardized sets of benchmark examples cannot be a measure for interactive systems, since with the help of the complete user, they can prove all theorems (if the underlying calculus is complete). Here the PUTEST situation allows the user of an automatic system to break up the theorems into pieces manageable for the automatic system and prove them separately.

As we have stated in the introduction, PUTEST is only a test schema, which has to be instantiated to get meaningful results. Therefore we want to mention some of the problems that have to be overcome in practical test situations.

- In recent times there has been some awareness of the problems of different input languages for deduction systems and has lead to attempts to standardize the input languages. However the much more difficult problem of different proof formalisms (which clearly contains the language problem) has to be solved for an evaluation of the results in PUTEST.

- The evaluation of the (proof) results is a central component of PUTest. This paradigm is not feasible in some application areas of deduction systems like for instance program verification, where great numbers of "junk-theorems" have to be proven fast and fully automatically. Here other tests may give more meaningful results.

- There must be some criterion which specifies which teams of user/system may enter PUTest in order to prevent the team consisting of a good typist and `emacs` from competing. We believe that the purpose of the use of deduction systems is to obtain a level of formalization and of machine support that makes the resulting problem solving session with a deduction system more credible than traditional proofs on paper. Thus we need some minimal criteria for systems to be called deduction systems.

- The test does not yield a linear scale by which to give a ranking of theorem provers. However, the multidimensional result of a PUTest evaluation is also an asset, since it allows one to quantify relative strengths and weaknesses of very heterogenous sets of deduction systems. Furthermore it is not surprising that comprehensive tests of a class of software systems that is as complex and heterogenous as that of deduction systems can only result in a multidimensional result.

- A jury is needed to judge the quality of the results. This fact makes evaluation of deduction systems with PUTest a social process with all its problems and advantages. One of the problems here will be to find competent juries to judge the results delivered by the deduction system. Furthermore the choice of test users of the deduction systems is critical for the test. We think that tests should be conducted both with gurus (to see what is possible in principle) as well as with beginners (in order to be able to judge the time need to become productive with the systems).

References

[Die93] Stanford Diehl. Acrobat vs. Common Ground. *BYTE*, pages 133–136, October 1993.

[SSY94] Geoff Stucliffe, Christian Suttner, and Theodor Yemenis. The TPTP problem library. In *Proceedings of the 12th Conference on Automated Deduction*, LNCS, Nancy, France, 1994.