CASC: Effective Evaluation having an Effect

Jeff Pelletier and Geoff Sutcliffe

March 25, 2001

1 Introduction

Although advances in the underlying theory of a subdiscipline of AI can result in impressive increases in the performance of systems that employ such an underlying theory, this sometimes seems to be almost "by accident." The reason for this impression is that the impressive new advance in performance sometimes seems to be a feature merely of the specific example tests that are being demonstrated. And this impression is further strengthened when one notes that, in general, a localized theoretical advance is only rarely sufficient to increase the overall performance of any complex system. As a result of all these considerations, researchers who make theoretical advances are left desiring some method that would demonstrate that the advance really does have general, overall positive consequences for their system's performance.

One natural way to satisfy this desire would be to have some set of benchmark problems to test the system on and compare it to the older system using these benchmarks. An alternative suggestion, that a "randomly-chosen" set of problems from the intended problem domain be selected and the new system be compared to the older system on this set, seems not to be appropriate in most areas of AI because there is generally no notion of what the entire problem space is and therefore there is no well-defined notion of a "randomly chosen" problem set.

Of course, the notion of a set of "benchmark problems" for areas of AI is also problematic, since in general it is not known what is the full extent of the problems we desire to solve in the area. Nonetheless, because the notion of a set of benchmark problems is, in principle, something that can be agreed upon, at least to some extent, by researchers in the relevant area, it is easier to construct this sort of test apparatus than it is to get an approximation to a "randomly chosen set" of problems that represent the underlying reality of the problems in the area. Thus a good general strategy to be used in determining whether an (apparent) theoretical advance in some area of AI really represents an important advance in the performance of a system that embodies the advance is to test the system on a set of benchmark problems. Naturally enough, this requires an agreement among the researchers in the field as to what the benchmark problems are. But if there are enough benchmark problems, then we might be able to “randomly choose” only some of these to test fully the system’s ability to deal adequately with the entire set of benchmarks.

2 Background

The foregoing picture gives a perspective from which to view the field of Automated Theorem Proving (ATP) as it has developed over the last four decades. Prior to six years ago, research in ATP was characterized by researchers who did self-evaluation of their systems by announcing that some new theoretical advance could, when added to their system, prove such-and-so problem. Although there were various attempts to construct lists of “test problems” (e.g., [Pelletier, 1986]), these were themselves constructed by individual researchers and did not have the general backing of the field (nor, it must be said, even of their own authors) as true benchmark problems. Given all this, it was difficult to determine which theoretical research efforts in the field of ATP were really promising and which were merely “accidentally interesting.” Impressive claims were made about individual systems, but since there was no real way to compare systems overall on problems agreed to be central, there was no way to impartially evaluate these claims.

This state of affairs changed with the construction of TPTP (Thousands of Problems for Theorem Provers, [Sutcliffe and Suttner, 1998]), where all the published test problems were collected and into which researchers were encouraged to add new problems as they were discovered or conceived. Many researchers took the opportunity to think of what it was that they believed to be a representative set of benchmark problems, and this has led TPTP to contain pretty much all the problems that current researchers think of as benchmarks. Of course, since the realm of “theorems in logic” is infinite and not well-categorized into a finite set of representative groups, it cannot really contain *all* benchmark problems about “theorems of logic.” Instead it contains those problems that are seen by today’s researchers as being important tests. And of course it is growing continually as researchers add more and more problems.

Thus, although there is no such thing as a “representative random sample” of theorems of first-order logic, there can be a set of benchmark problems that are set by the theoretical reflections and practical interests of researchers in the area, and it is from these sources that the TPTP claims to be a set of benchmark problems. As mentioned, the TPTP is large enough that there can also be representative sampling of the problems such that solving a certain percentage of them justifies one in believing that a system could solve that same percentage of all the problems in TPTP (within a certain confidence level).

Having a set of benchmark problems is only half the battle in evaluating and improving (and evaluating the improvement!) research in an area. Additionally we need to demonstrate that there has been incremental improvement within the field *as a function of* the existence of the benchmark problems. In the field of ATP, this has been demonstrated by the annual CADE (Conference on Automated Deduction) ATP System Competitions, CASC. CADE is the major forum for the presentation of new research in automated deduction, and having the CASC theorem-proving contest at the general meetings of this conference has produced substantial increase in the performance of ATP systems.

Or at least, this is what the present paper will set out to demonstrate. Our goal in this paper, then, is to give empirical evidence that the combination of the benchmark TPTP problems together with their use in the CASC tournaments has resulted in major improvements in important parameters of evaluation. This paper presents quantitative measures that show the progress in ATP, from mid-1996 to the present. The measures are based on collected performance data from ATP systems and their performance at the CASC tournaments over the years.

3 Classes of Benchmark Problems in TPTP

The problems in TPTP are characterized in three fundamentally different ways. One way concerns how "difficult" they are, as judged by the number of known ATP systems that can provide proofs of them. On this dimension of characterization we find problems ranging from "solvable essentially by all systems" to "not solvable by any current system" and even beyond this to "not known whether or not it is solvable." The rankings here are real numbers from 0 to 1; and of course this numerical feature changes over time as the ATP systems improve.

A second dimension of categorization is about the language, or format, in which the problems are stated: whether they are presented to the theorem proving system in "natural form" (which is called FOF, for 'first-order form') or they are presented in "clausal form" (also called 'negated-conclusion conjunctive normal form' and labelled CNF here). Most of the theorem-proving systems were designed to operate only on CNF problems. (This is not a logical weakness of these systems, since every "natural problem" has a CNF representation).

The third dimension of characterization is by logical features of the problems. TPTP has very many categories in this dimension, but they are not all relevant to the present paper. The following categories are relevant to what follows, and a few more will be mentioned as they are introduced. One fundamental categorization concerns whether or not the formula is a theorem, this being thought to be a decision that an ATP system should be able to determine.[1] (And perhaps, if it is not a theorem it should give a "model" that demonstrates this, whereas if it is a theorem it should provide a proof of this.) The non-theorems are *satisfiable*, and hence are called SAT. Another important category concerns whether a problem is "essentially first-order" or whether it is "essentially a propositional logic problem." This difference can more precisely expressed as whether the problem's Herbrand Universe is infinite (first-order) or not (propositional). In

[1]Of course, first-order logic is not decidable, so there can be no program that will universally make this judgment. But the TPTP focuses on the comparative extent to which theorem provers can determine this for a group of particular problems.

this paper we focus on the first-order problems, although we will mention the SAT problems in passing.

Another categorization in this dimension concerns the use of equality in the problems, since this turns out to have a major impact on both the logical features of problems and the algorithms employed by implementations. A particular type of equality problem occurs when each piece of information is "an identity claim" or its negation ("unit equalities"). Attempting to determine whether a set of such claims is or is not satisfiable is a specialized problem, and this has formed a separate contest from the very beginning, called UEQ (for "unit equality").

Finally in this dimension, we should mention that there is an important logical distinction between problems that can be represented with "horn clauses" versus ones that cannot be so represented. Although this distinction was not made in the problems presented at the early contests, it was incorporated later, as we will see. At the beginning of the contests, and ever since that time, there was a category MIX, which consisted of a mixture of all types of problems. Some contestants view this as the "central part" of the contests on the grounds that it did not favor any one type of problem. (Although all problems in this group were presented as CNF, at least in the beginning, rather than in FOF.)

4 Some Brief Historical Remarks about CASC

The purpose of this section is to show how the foregoing idea of having a collection of benchmark problems was turned into a series of competitions involving these problems, and what changes have occurred in the competitions over the years. Our claim is that any other effort to employ benchmark problems in a competitive environment will face similar experiences, and that for the most part these experiences are salutory. Not only does the contest evaluate the relative capabilities of the systems in the area – CASC accurately describes the relative capabilities of the various ATP systems – but a contest, no matter what the field, has other effects both on the relevant community (where it provides an inspiring environment for personal interaction among the researchers) and also on the wider community (by exposing the systems to researchers outside the narrow group, and by introducing new or isolated researchers to the mainline of research). It also has the effect of stimulating research in the general area because of this wider exposure. All these consequences have been evidenced in the ATP community as a result of the contests.

The overall point for which we are arguing is that *the most effective way to empirically evaluate (algorithmetically intractable) AI systems is by means of a contest.* Not only does it foster improvement in the existing systems, and not only does it entice new researchers into the field, and not only does it garner attention to the area from outside, but in fact *a contest accurately measures, in an empirically respectable way, relative strength of systems and improvement over time.*

Any contest is difficult to organize in the first instance and then to run over the years. Unlike, say, the experience of the chess community in initiating their computer chess contests, the theorem proving community did not have a history of human tournaments to fall back on in designing its methodology, and this further exacerbated the organizational dificulties. One important decision made at the very beginning was that all systems were to run autonomously, that is, without any human intervention. Arguably this rules out a number of very interesting research efforts that involve cooperative investigation of the proof process; but it was decided that since part of the goal of the contest was to evaluate the systems themselves, it would not be an accurate investigation if different people were also involved in the search for a proof by different systems.

Each year since 1996, the CASC Competition has been run at the CADE conference. We will characterize the evolution of CASC, paying particular attention to its design, design changes, and implmentation, since these are indicative of the sort of considerations that attempts to institute a contest in other areas would experience.

As we mentioned above, ATP problems have identifiable logical, language, and syntactic characteristics. Since there are important differences in the types of problems, as well as there being practical differences in the reasons to be able to solve such problems,[2] CASC is divided into divisions according to problem and system characteristics.[3] Different systems enter the different divisions, although some systems enter all divisions.

4.1 CASC-13 (Rutgers University, USA, 1996)

CASC-13 had only two divisions, the MIX division containing CNF theorems that were "really first order", and the UEQ division containing unit equality problems. Although adequate for a first competition, this division structure was refined for CASC-14 (see Section 4.2).

An initial puzzle facing the organizers was to determine how many of the problems in TPTP (or, in one of its subgroups) needed to be asked of the contestants in order for us to have some particular degree of confidence that a systme which could solve such-and-so percentage of the chosen problems would be able to solve that same percentage of all the TPTP problems (or, of all the problems in a chosen subgroup). The Appendix below gives the reasoning employed in the first two contests.

Additionally, in an attempt to evaluate individual specialized techniques and systems, a distinction was made between "monolithic" and "compositional" systems. The idea was that in the former systems, no special subprogram would be chosen just because the problem manifested a certain style or characteristic, whereas the latter type of systems would choose distinct submodules based on the given problem's characteristics. So a compositional system might be made

[2]A system that was able to solve SAT problems, for example, is intended to be used for different purposes than one able to solve unit equality problems, and so on.

[3]When a system could not run on the machines supplied for the contest – as for instance when they ran over the web or they used special hardware (Lisp machines or Macintoshes) – they were entered into a "demonstration version" of whatever category they would otherwise be competing in.

up by several distinct monolithic systems, and the module that was to be chosen would be based on the given problem's characteristics.

There was a lot of discussion in the year leading up to the competition, and especially in the month or two prior, concerning how a "winner" in any one category would be chosen. The discussions led the organizers to propose two different scoring schemes. The first ranking scheme focussed on the ability to find as many solutions as possible, while the second ranking scheme measured solutions-per-unit-time.

It turned out that the scoring schemes identically ranked all the systems, for each division. As with the division structure, the ranking schemes were changed for CASC-14, basically by deleting the solutions-per-unit-time measure.
WINNERS:

1. MIX: E-SETHEO, by R. Letz of the Techniche Universität München.
2. UEQ: Otter 3.0.4, by B. McClune of Argonne National Laboratories.

The major effects of this competition were:

- This was the first time that the relative capabilities of ATP systems were evaluated.
- The competition stimulated ATP research – most entrants made special efforts to improve the autonomous performance of their systems, and all the entrants had to ensure that the implementation and debugging of their systems was complete.
- The competition provided an inspiring environment for personal interaction between ATP researchers – there was more excitment and activity than the organizers expected!
- Many of the conference delegates came to see the competition – the competition thus exposed the ATP systems to researchers both within and outside the ATP community.

4.2 CASC-14 (Townsville, Australia, 1997)

The overall success of CASC-13 motivated expansion in CASC-14. Two new divisions were added: the SAT division containing satisfiable clauses sets, and the FOF demonstration division containing problems in full 1st order form. This latter division was designed to give Natural Deduction theorem proving systems, which typically operate with the full range of connectives of the first-order logic (the "natural form") rather than with the restricted subset of CNF, a chance to compete. Of course, systems that operated on the restricted subsets could also compete in this division, so long as part of their proof time involved converting the natural form into their restricted form. Some contestants (and other theorists) think this FOF division should be the *main* competition.

The CASC-13 distinction drawn between compositional and monolothic systems was not very successful or meaningful. It is hard to clearly distinguish the two types of systems – almost all intuitively "monolithic" systems have some internal runtime tuning, and it is not clear when such tuning makes the system "compositional". And anyway, the results showed no salient evidence that compositional systems had any advantage over monoloithic systems. At the same time, the results (as well as feedback from the entrants) suggested that it would be interesting to separate the systems' performances within the MIX division according to finer-grained problem types. It was realized that with an appropriately fine-grained categorization, compositional systems would invoke the same monolithic component for every problem in such a category. This would then enable the individual components of a compositional system to be evaluated fairly against monolithic systems. Therefore, in CASC-14 the MIX division was divided into 4 categories: HNE (Horn problems with No Equality), HEQ (Horn problems with Equality), NNE (Non-Horn problems with No Equality), and NEQ (Non-Horn problems with Equality).

In CASC-13 the minimal number of problems used in each division and category was based on the simple statistical measures mentioned in Section 2. The problem of confidence in the results aroused the interest of some statisticians at the Technische Universität München, who developed a more elegant model for determining the minimal number of problems to be chosen from a population in order to have some specified degree of confidence that the percentage of actually-solved problems projects to the entire population [Greiner and Schramm, 1996]. This new scheme has been in use in all the more recent competitions. The organizers can give the number of eligible problems and a required average confidence level that the CASC results represent the results that would be obtained using all eligible problems, and the tables in [Greiner and Schramm, 1996] will say how many problems it is necessary to select.

As mentioned in the last section, the two scoring schemes of CASC-13 did not produce different rankings. As it happens, systems that solve many problems also solve them quickly. So a decision was made to rank simply by number of problems solved, using average solution time over problems solved in order to break ties.
WINNERS:

1. MIX: Gandalf, by T. Tammet of U. Göteborg.
2. UEQ: Waldmeister, by A. Buch and T. Hillenbrand of Universität Kaiserslartern.
3. SAT: SPASS 0.77, by C. Weidenbach of Universität Saarbrücken.
4. FOF: SPASS 0.77, by C. Weidenbach of Universität Saarbrücken.

The major new effects of CASC-14 were:

- Introduction of the SAT and FOF systems.
- Systems more refined at control level.

- New ideas appear, especially with Gandalf and Waldmeister. In particular, Gandalf introduced the notion of "time-slicing", where one strategy was attempted for a short while and if it didn't produce a proof then it was jettisoned and the proof was begun anew with a different strategy. Variants of this methodology have shown up in almost all the stronger systems afterwards. As will be seen below, Waldmeister has a stranglehold on the UEQ competition.

- New entrants – people coming out of the woodwork. There were both long-standing theorists in the field who decided to enter the competition and also new researchers who were initially introduced to the whole ATP area by the fact that there was a contest.

- Better understanding of the evaluation.

4.3 CASC-15 (Lindau, Germany, 1998)

In CASC-15 a further category was added to the MIX division, PEQ (Pure Equality), and the Horn/Non-Horn categories HEQ and NEQ were limited to problems with some, but not pure, equality. It was thought that this would further allow specialized systems to show their particular abilities.

In CASC-14 the organizers took on the task of ensuring that the submitted system ran correctly in the competition environment. It was decided that in the future the competition control scripts will be made available to the entrants, who can then ensure in advance that their systems behave as required. This was first implemented at CASC-15, which allowed more auotmation during the event itself. In CASC-14 it was noticed that extremely high memory usage could be employed as a method to circumvent the cpu time allotment, since the time spent in swapping would not therefore be counted against the contestant. So it was decided that a "wall clock limit" should be employed. Such a scheme was designed for CASC-15, but not imposed. It will finally become part of the contest in CASC-18. In an attempt to impose more standardization on the competition, all output required to be to stdout, and no other output was deemed relevant.
WINNERS:

1. MIX: Gandalf c-1.1, by T. Tammet of U. Göteborg.

2. UEQ: Waldmeister798, by T. Hillenbrand of Universität Kaiserslartern.

3. SAT: SPASS 1.0.0a, by C. Weidenbach of Max Plank Institut für Informatik. Saarbrücken.

4. FOF: SPASS 1.0.0a, by C. Weidenbach of Max Plank Institut für Informatik.

The major new effects of CASC-15 were:

- Advent of tuning to CASC. Rather than writing theorem proving systems that were to solve "all problems", or at least were designed to be general purpose, some contestants thought to write systems that could prove all the problems in TPTP. And to do this they had many specific subroutines that were designed especially for some small group of problems in TPTP.
- Newspaper publicity. It is expected that this will influence graduate students to enter the area.
- Influence acknowledged. Many contestants claim that the particular research they carried out over the year was due to a desire to be competitive in future contests.
- Realization of some players that they their systems were no longer competitive.
- No natural deduction systems were entered; CNF conversion systems won. There was speculation on whether this shows that natural deduction systems cannot compete with CNF conversion systems.
- Success of Gandalf inspired more "competition systems" (E-SETHEO, SSCPA) – ones that were designed so that there was an "internal competition" to see which of several different proof strategies would succeed.
- Realization that CASC matters. The improvement that can be seen in the winning systems over the last-year winners can be parlayed into publications and grant funding. And it was the competition that caused this improvement.
- Leading systems in CASC too good for low rated problems, which are solved in less-than-measurable time. This is due in part to the improved system quality of the best systems, and in part due to "tuning" for the competition.

4.4 CASC-16 (Trento, Italy, 1999)

The competition division structure stayed the same for CASC-16. But the "demonstration division", which was initially conceived as a place for systems requiring special hardware (e.g., LISP machines, or parallel machines, or web access, etc.), was expanded to a more general concept, with the explicit purpose of allowing panel members and organizers to enter the competition.[4]

The early access to the lists of the eligible problems in CASC-15 lead to entrants spending considerable time tuning their systems specifically for the eligible problems. This is unproductive in the broader context of ATP development, and so in CASC-16 the lists of eligible problems were not published

[4]C. Suttner, one of the organizers of CASC-13, had entered his system, SPTHEO, in that competition. And some contestants had questioned the wisdom of allowing this (but there were no concerns about improprieties at that competition). Furthermore, G. Sutcliffe wanted some way to enter his SSCPA system in the CASC-16 competition.

until after the systems had been installed on the competition machines. Further, the most up-to-date TPTP problem difficulty ratings, which have a role in determining which problems are eligible, were not released before the competition. So, although entrants were able to determine which TPTP problems have the right syntactic characteristics for each division and category, they had access to only the difficulty ratings supplied with an earlier release of TPTP. Thus entrants could not accurately determine exactly which problems were eligible, and they could then only tune their systems for problems with the right general characteristics, rather than specifically to a class of eligible problems. However, public access to TPTP means people can fairly accurately work it out themselves: the only thing they do not know is updated difficulty rating of the problems, which affects whether a problem is eligible or not.

A particular problem encountered in CASC-15 was skewing caused by large numbers of very similar problems, in particular the 'ALC' problems within the TPTP "pure syntax" (SYN) domain.[5] For CASC-16, lists of such very similar problems in the TPTP were identified, and a limit was imposed on the number of very similar problems in any division or category.

Many of the problems used in CASC-15 were relatively easy, as is indicated by the large numbers of very low proof times in the CASC-15 results. For CASC-16 the TPTP problem difficulty rating scheme was improved, and the minimal difficulty rating for eligible problems was increased to 0.21. These changes provided a more appropriate selection of problems, both in terms of breadth and of difficulty.

The division-winning programs from CASC-15 were copies and saved, and were entered into CASC-16 to provide benchmarks (the competition archive provides access to those systems' executables and source code). By comparing the results of the "old" systems with the results of the "new" systems, definitive statements about the progress of ATP can be made.

Although there had been some discussion of the issue prior to the very first CASC, no system is required to print out or otherwise display a proof or any other assurance that it in fact actually has proved a problem. Since in most divisions of the competition only theorems are presented to the contestant programs, it might seem that a winning strategy would be for the system to merely say 'proved' as soon as asked any problem. Of course, such systems would be unsound – they would declare non-theorems to be theorems ...if they were asked. In order to try to establish soundness of the entrants,[6] the organizers exposed the programs to a number of non-theorems prior to the contest. Given the nature of this testing it is not surprising that some unsound systems were not noticed. In CASC-16 this was particularly noteworthy because in August 1999, i.e., about a month after the competition, E 0.5 and E-SETHEO(-FLOTTER) 99csp were found to be unsound in certain rare circumstances. The unsoundness was due to a bug in E, which was also used as a component of E-SETHEO. Unsoundness is unacceptable, and the competition panel retrospectively disqualified the

[5]These problems are encodings of problems from multi-modal K-logics.

[6]Since first-order logic is undecidable, there can be no absolute test of soundness. The best that can be done is to "empirically test" the systems.

two systems from being ranked in the competition. (It must be noted that the unsoundness was entirely accidental, and that there was no attempt to deceive). The circumstances in which the two systems are unsound did not occur in the competition, and therefore the performance of these systems in the competition was not affected. Subsequent testing of the repaired systems verified that the same problems could be solved.
WINNERS:

1. MIX: Vampire 0.0, by A. Voronkov of Manchester University.
2. UEQ: Waldmeister799, by T. Hillenbrand of Universität Kaiserslartern.
3. SAT: OtterMACE 437, by B. McClune of Argonne National Laboratories. Saarbrücken.
4. FOF: SPASS 1.0.0T, by C. Weidenbach of Max Plank Institut für Informatik.

The major new effects of CASC-16 were:

- For the first time, before CASC-16 there was some acrimonious debate regarding the design and implementation of the competition. The problem starts with a complaint concerning the internal table of Waldmeister, which recognizes the "theory" of a problem (rings or groups or CD or ...) and chooses a search strategy based on this information. Some of the Waldmeister categories are small, so people said it was the same as storing information about individual problems. Having warmed up, people then started complaining about other forms of excessive tuning. Despite lots of thought on the part of the organizing committee, they could not formulate a rule to sort out the tuning issue. Related to this were the issues about the skewing of results in CASC-15 because of the ALC problems. This was solved by imposing a limit on the number of similar problems.
- First steps towards a robust methodology for installation and execution are taken (but not all successful). Most systems become "TPTP clean".
- More interest from within ATP community.
- Debate at panels regarding desirability of the focus on implementation, versus theory development. It seems clear that much effort is being spent on carefully constructing and tuning systems for the eligible problems, and this is felt by some to be at the expense of basic research that has not yet been well-implemented.
- Session at CADE dedicated to discussing CASC results.
- The issue of the initially-announced winner being unsound highlighted the (unavoidable) cracks in the soundness testing. This in turn led to a revival in interest in proof output, which had been discussed at length prior to CASC-13.

4.5 CASC-17 (Carnegie Mellon University, USA, 2000)

At CASC-16 there was debate about the relevance of CASC, with claims that the "problems do not reflect real-world usage". Although this debate is complex, and not all agreed with this change, it was decided to react to this apparent interest in applications, by adding a SEM division ("semantically interpreted") in CASC-17. The aim was to provide a group of problems from a chosen application domain, and to allow systems to be explicitly tuned for the type of problem. The particular problems used for this group were set-theoretical theorems formulated within Gödel-von Neuman-Bernays set theory.

The debate at CASC-16 also raised the issue of "effectively propositional" problems, i.e., problems with a finite Herbrand Universe that can be directly translated to propositional form and solved using specialized propositional techniques. It was considered that these problems were biased towards certain specialized provers, and that they were not suitable for the evaluation of first-order systems. Although this view point is consistent with the underlying motivation for divisions, and although there are other venues where these sorts of problems are to be solved, not everyone agreed with the decision to exclude effectively propositional problems from CASC-17.

System installation for CASC-17 required installation packages, to encourage developers to provide "industrial strength installation". However, more than in previous competitions, the systems required modification after installation before they could execute in the 'production environment' of the competition.
WINNERS:

1. MIX: E 0.6, by S. Schultz of the Techniche Universität München.
2. UEQ: Waldmeister600, by T. Hillenbrand of Universität Kaiserslartern.
3. SAT: OtterMACE 437, by B. McClune of Argonne National Laboratories.
4. FOF: VampireFOF 1.0, by A. Voronkov of Manchester University.
5. SEM: E-SETHEO 2000csp, by S. Schultz of the Techniche Universität München.

The major new effects of CASC-17 were:

- Significant, year long, development in preparation for CASC.
- Only the serious players there - is CASC getting too serious? Are new systems excluded due to level of competition?
- Realization that systems are now uniformly good enough that non-standard, non-biased problems are safe to use. Initially, such non-standard problems were thought to favor particular ATP systems, since they were alterations made to existing problems. So long as these problems are non-biased (as determined by the maintainers of TPTP), they should be included.

- It is becoming obvious that there is a real need to stop tuning. Systems are submitting data to affect the eligibility of problems in TPTP, since the degree of difficulty, etc., of problems is influenced by the reported results of different theorem proving systems over the course of the year.

4.6 CASC-18 (Siena, Italy, 2001)

In CASC-17, FOF problems on set theory were used in the SEM domain, but no systems were tuned specifically for the division. The lack of interest, and the overlap between the SEM division and the existing syntactically defined divisions,[7] led to the demise of the SEM division in CASC-18. At CASC-17 some entrants expressed a continued interest in effectively propositional problems, claiming that first order techniques could be more effective than translation to propositional form and the use of specialized hardware and software on this result. The theoretical implications of this thought prompted the introduction of the EPR - Effectively Propositional division in CASC-18, containing problems worded in first-order form but which have a finite Herbrand Universe. Such problems are not given in any of the other divisions.

Non-standard, non-biased problems become eligible.

Following unsoundness issue in CASC-16, and continued interest at CASC-17, MIX division divided into two classes, the "proof" class and the "assurance" class. The aim is to encourage research into proof and model presentation, and the implementation of proof generation and verification as part of an integrated reasoning process. The requirement will not exclude from the competition those systems that do not, for whatever reason, generate proofs or models, for they are entered into the "assurance" class.

In an effort to stymie tuning of systems to the details of the problems being used at CASC-18, the organizers have resorted to unseen problems. It remains to be seen how successful this will be.

The wall clock limit, designed for CASC-15, was eventually implemented, and a new set of "clean execution requirements" has been established.

5 Conclusion

The purpose of this paper has been two-fold:

- to give an informal account of the motivation and history of the CASC competitions for automated theorem proving.

- to argue for the usefulness of competitions in providing an empirically accurate evaluation of both

    1. the relative success of different systems and methodologies in a sub-field of AI

[7]The particular SEM problems selected could also have occurred in the "ordinary competition" in the FOF category.

2. an absolute measure of the overall improvement of systems and methodologies in a subfield of AI

Although the example subfield of AI that we discussed is automated theorem proving, we think these conclusions will hold for any similarly-organized subfield of AI.

There were 35 years of theoretical research and individually-evaluated systems in automated theorem proving. In that time many techniques and ideas were generated that needed to be evaluated in order to determine which were viable, and to integrate them into systems that were more flexible and powerful than the ones created by their inventors. For all these goals, experimental system evaluation is a crucial research tool, and competitions provide stimulus and insight that can lay the basis for the development of future ATP systems.

Like most areas of AI, automated theorem proving is not tractable. This means that it is impossible to give a theoretical analysis of relative success or progress in the field, except in certain very simple cases. Therefore some other means is necessary if we are to convince ourselves (and funding agencies) that there has been substantial and important progress. One such way is through the construction of benchmark problems, but as we are all too aware, the non-tractability of these interesting areas of AI means that there cannot be any "truly representative" set of benchmark problems. The best we can do is construct an ever-expanding set of problems that current researchers take as benchmarks.

But having a set of "benchmark problems" is only a part of the issue of empirical evaluation. There must also be some way to determine that AI systems *really can* solve these problems, as opposed to merely being "tuned" to specific problems. And it is here that competitions enter the picture. With a large enough set of benchmark problems, we can statistically control issues of "tuning." The CASC has pointed a way to deal with this topic.

We believe that the competition fulfilled its main motivations: stimulation of research, motivation for improving implementations, evaluation of relative capabilities of ATP systems, and providing an exciting event. The competition contributed to ensuring that ATP systems meet "the basic requirements" suggested in [Kaufmann, 1998]. For the entrants, their research groups, and their systems, there has been substantial publicity both within and outside the ATP community. The significant efforts that have gone into developing the ATP systems have received public recognition; publications, which adequately present theoretical work, have not been able to expose such practical efforts appropriately. The competition has provided an overview of which researchers and research groups have decent, running, fully automatic ATP systems.

In the face of the desirability of more extensive competitions, it is also important to avoid being overambitious. A successful limited competition with meaningful results is preferred over a larger competition which is not accepted by the ATP community. Large events require large resources, which are currently not available. If the CASC competitions continue to provide useful information about the competing systems, it would be reasonable for those who benefit from access to the information to help provide the resources required for bigger and better competitions.

APPENDIX: **Choosing numbers of Problems**

An initial difficulty was to determine the number of problems that the theorem provers would be asked in the competition. The idea was that we wanted to know how good a system would do on the entire TPTP corpus, but were restricted to asking only a subset of the problems. So the question was, how many problems do we need to pose?

The minimal number of problems should ensure sufficient confidence (say 85%) that the competition results are the same as would be obtained using all eligible problems. For an evaluation based on the number of problems solved, as is essentially the case for the ranking schemes used, and assuming a worst case proportion of problems solved (50%), the minimal number of problems is computed by first computing n_0, the minimal number assuming an infinite number of eligible problems:

$$n_0 = \frac{1.44^2 \times 0.5 \times (1 - 0.5)}{0.15^2} = 23.04$$

(the 0.15 in 0.15^2 comes from 85%). We then adjust for the actual number of eligible problems as follows, in accordance with [Wadsworth, 1990, page 9.17].

Decision : The minimal number of problems is:

$$min\_number\_of\_problems = \left\lceil \frac{23.04 \times number\_of\_eligible\_problems}{23.04 + number\_of\_eligible\_problems} \right\rceil$$

After that determination, the choice of the number of problems in a competition is figured like this (which is taken from the "rules page"). The minimal numbers of problems that have to be used in each division and category, to ensure sufficient confidence in the competition results, will be determined from the numbers of eligible problems in each division and category (the competition organizers have to ensure that there is sufficient CPU time available to run the ATP systems on this minimal number of problems). This minimal numbers of problems will then be used in determining the time limit imposed on each solution attempt.

A lower bound on the total number of problems that will be used will be determined from the number of workstations available, the time allocated to the competition, the number of ATP systems to be run on the general hardware over all the divisions, and the time limit, according to the following relationship:

$$\#Problems = \frac{(\#Workstations \times TotalTime)}{(\#Entrants \times TimeLimit)}$$

It is a lower bound on the total number of problems because it assumes that every system will use all of the time limit for each problem. Since some solution attempts will obviously succeed before the time limit is reached, more problems can actually be used. The actual numbers used in each division and category will be determined according to the judgement of the competition organizers.

References

[Greiner and Schramm, 1996] Greiner, M. and Schramm, M. (1996). A Probabilistic Stopping Criterion for the Evaluation of Benchmarks. Technical Report I9638, Institut für Informatik, Technische Universität München, München, Germany.

[Kaufmann, 1998] Kaufmann, M. (1998). ACL2 Support for Verification Projects. In Kirchner, C. and Kirchner, H., editors, *Proceedings of the 15th International Conference on Automated Deduction*, number 1421 in Lecture Notes in Computer Science, pages 220–238. Springer-Verlag.

[Pelletier, 1986] Pelletier, F. (1986). Seventy-five Problems for Testing Automatic Theorem Provers. *Journal of Automated Reasoning*, 2(2):191–216.

[Sutcliffe and Suttner, 1998] Sutcliffe, G. and Suttner, C. (1998). The TPTP Problem Library: CNF Release v1.2.1. *Journal of Automated Reasoning*, 21:177–203.

[Wadsworth, 1990] Wadsworth, H. (1990). *Handbook of Statistical Methods for Engineers and Scientists*. McGraw-Hill.