CASC: Effective Evaluation having an Effect

Jeff Pelletier[*]
University of Alberta
jeffp@cs.ualberta.ca

Geoff Sutcliffe[†]
University of Miami
geoff@cs.miami.edu

Abstract

Having a set of benchmark problems is only half the battle in evaluating and improving (and evaluating the improvement of!) research in an area. Additionally it is necessary for the benchmark problems to be used appropriately, to evaluate the systems and produce improvements. In the field of ATP, this has in part been achieved by the CADE ATP System Competitions (CASC). We believe that having CASC at CADE has produced substantial increases in the performance of ATP systems, and has also significantly improved the methodologies used for evaluating ATP systems.

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 increase in performance sometimes seems to be a feature merely of the specific testing performed. 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 these considerations, researchers who make theoretical advances also need some way to demonstrate that the advance really does have general, overall positive consequences for their system's performance. One natural way to satisfy this desire is to have some set of benchmark problems to test the system, and to compare the performance with that of 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. This is 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.

The notion of a set of "benchmark problems" for areas of AI is often problematic, since in general it is not known what is the full extent of the problems we desire to solve in the area. Nonetheless, 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 represents 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 contribution to the performance of a system that embodies the advance, is to test the system on a set of benchmark problems. If there are enough benchmark problems, then we might be able to "randomly choose" only some of these to test the system's ability to deal adequately with the entire set of benchmarks. Further thoughts on these issues are in Section 12.

2 The TPTP Problem Library

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 and subsequently announced 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 published test problems are collected, and to which researchers are encouraged to contribute new problems as they are discovered or conceived. Many researchers took the opportunity to think of what it was that

[*]Dep't of Computing Science, University of Alberta, Edmonton, Alberta, Canada T6G 2H1. http://www.cs.ualberta.ca/∼jeffp.

[†]Dep't of Computer Science, University of Miami, P.O. Box 248154, Coral Gables, FL 33124. http://www.cs.miami.edu/∼geoff.

they believed to be a representative set of benchmark problems, and this has led to the situation where the TPTP contains 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, the TPTP 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 formed from the theoretical reflections and practical interests of researchers in the area. The TPTP is such a benchmark set. 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).

3 The CADE ATP System Competitions

Having a set of benchmark problems is only half the battle in evaluating and improving (and evaluating the improvement of!) research in an area. Additionally it is necessary for the benchmark problems to be used appropriately, to evaluate the systems and produce improvements. In the field of ATP, this has in part been achieved by the CADE ATP System Competitions (CASC). Each year since 1996, CASC has been run at the CADE conference (CADE, the Conference on Automated Deduction, is the major forum for the presentation of new research in automated deduction). We believe that having CASC at CADE has produced substantial increases in the performance of ATP systems, and has also significantly improved the methodologies used for evaluating ATP systems.

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 ATP community did not have a history of human tournaments to fall back on in designing its methodology, and this further exacerbated the organizational difficulties. 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. A second basic (and obvious) decision was that the systems should all be evaluated using the same hardware, supplied by the competition organizers. When a system cannot run on the machines supplied for the contest – as for instance when they ran over the web or they used special hardware, e.g., Lisp machines or Macintoshes – they are entered into a "demonstration version" of whatever division they would otherwise be competing in.

This remainder of this paper describes evolution of CASC, paying particular attention to its design, design changes, and implementation, since these are indicative of the sort of considerations that attempts to institute a contest in other areas would experience. We believe 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 salutary. 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 CASC.

4 Classes of Benchmark Problems

The problems in TPTP are characterized in three fundamentally different ways. The first dimension is concerned with how "difficult" they are, as judged by the number of known ATP systems that can provide solutions to them [Sutcliffe and Suttner, 2000]. On this dimension 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. CASC limits itself to problems that are (expected to be) solvable by some but not all of the systems, so as to provide differentiation between the systems.

The second dimension of categorization concerns 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 ATP systems are designed to operate only on CNF problems. This is not a logical weakness of these systems, since every "natural problem" has a CNF representation. At first CASC used only CNF problems, but very quickly was extended to include FOF problems.

The third dimension of characterization is by logical features of the problems. The TPTP has very many categories in this dimension, but they are not all relevant to this 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 (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).[1] The non-theorems are *satisfiable*, and hence are called SAT

[1] Of course, first-order logic is not decidable, so there can be no program that will universally make this judgment.

problems in CASC. Another important categorization concerns whether a problem is "really 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 (really first-order) or not (essentially propositional). In this paper we focus on the really first-order problems, although we will mention the essentially propositional problems in passing. Another categorization in this dimension concerns the use of equality in the problems, since this has a major impact on both the logical features of problems and the algorithms employed to solve them. A particular type of equality problem is that in which each piece of information is "an identity claim" or its negation. Attempting to determine whether a set of such claims is or is not satisfiable is a specialized task. These problems are called UEQ (for "unit equality") problems in CASC. Finally in this dimension, 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.

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 (a system that is 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), CASC is divided into divisions according to problem and system characteristics. At the beginning of the contests, and ever since that time, there was a MIX division, which consisted of a mixture of all types of non-UEQ theorems. Some contestants view this as the "central part" of CASC. All problems in this division are presented in CNF. Other divisions are mentioned in the following sections.

5 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 6). 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 up by several distinct monolithic systems, and a module chosen 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 scheme focussed on the ability to find as many solutions as possible, while the second scheme measured solutions-per-unit-time. It turned out that the two schemes identically ranked all the systems, for each division. As with the division structure, the scoring 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 Technische Universität München.
2. UEQ: Otter 3.0.4, by W. McCune of Argonne National Laboratories.

The major effects and observations of CASC-13 were:

- This was the first time that the relative capabilities of ATP systems had been seriously 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 excitement 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.

6 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 clause sets, and the FOF demonstration division containing problems in full first-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) thought this FOF division should be the "central part" of CASC.

The CASC-13 distinction drawn between compositional and monolithic 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 of CASC-13 showed no salient evidence that the compositional systems had any advantage over the monolithic 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 [Sutcliffe and Suttner, 1997] (see Section 12 for some details). 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] say how many problems it is necessary to select.

As mentioned in Section 5, 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 Kaiserslautern.
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 and observations of CASC-14 were:

- Introduction of the SAT and FOF divisions made it possible for those types of systems to enter.
- Two systems employing natural deduction were entered, marking the first time they competed against the more traditional resolution systems.
- Many of the systems were more refined at the control level. In particular, Gandalf introduced the notion of "time-slicing", where one strategy is attempted for a short while and if it doesn't produce a proof then it is jettisoned and the proof attempt is begun anew with a different strategy. Variants of this methodology have shown up in many of the stronger systems since. Several entrants produced "auto" modes for this and subsequent CASCs.
- Waldmeister began its stranglehold on the UEQ division.
- New entrants – people came out of the woodwork. There were both long-standing theorists in the field who decided to enter the competition, and new researchers who were initially introduced to the whole ATP area by the fact that there was a contest.
- The organizers had a better understanding of the evaluation process.

7 CASC-15 (Lindau, Germany, 1998)

In CASC-15 the PEQ (Pure Equality) category was added to the MIX division, and the Horn/Non-Horn categories HEQ and NEQ were limited to problems with some, but not pure, equality. This further allowed 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 for CASC-15 that the competition control scripts will be made available to the entrants, who would then ensure in advance that their systems behaved as required. This allowed more automation during the event itself. In an attempt to impose more standardization on the competition, all output was required to be to `stdout`, and no other output was deemed relevant.

In CASC-14 it was noticed that extremely high memory usage could cause a system to run for a very ling time due to swapping For CASC-15 it was decided that a "wall clock limit" should be employed. Such a scheme was designed for CASC-15, but actually not imposed because the organizers never got around to implementing it in the control scripts. It will finally become part of the contest in CASC-JC.

Winners:

1. MIX: Gandalf c-1.1, by T. Tammet of U. Göteborg.
2. UEQ: Waldmeister 798, by T. Hillenbrand of Universität Kaiserslautern.
3. SAT: SPASS 1.0.0a, by C. Weidenbach of Max Planck Institut für Informatik. Saarbrücken.
4. FOF: SPASS 1.0.0a, by C. Weidenbach of Max Planck Institut für Informatik.

The major new effects and observations of CASC-15 were:

- Some systems were tuned especially for CASC. Rather than entering systems that were to solve "all problems", or at least were designed to be general purpose, some contestants tuned their systems specifically to solve problems in the TPTP. This was achieved by adjusting overall system control parameters to maximize the number of TPTP problems that could be solved within the CASC time limit, and also by adjusting the control parameters according to fine grained features of the problem at hand. A particularly effective form of tuning was (and still is) employed by Waldmeister. Waldmeister examines the problem to determine the underlying "theory" of a problem (rings or groups or CD or ...) and chooses a search strategy based on this information. As is described in Section 8 this later led to complaints from other entrants.

- The influence of CASC was being acknowledged. Many contestants claimed that the particular research they carried out over the year was due to a desire to be competitive in future contests. Good performance in CASC was also affecting publications and grant funding. For the first time CASC attracted newspaper publicity, with an article appearing in the local press.
- There was a realization for some entrants 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.
- The success of Gandalf inspired more "competition systems" that ran an "internal competition" between strategies, e.g., E-SETHEO and SSCPA.
- The leading systems in CASC were too good for low rated problems, which were being solved in less-than-measurable time. This is due in part to the improved quality of the best systems, and in part due to "tuning" for the competition.

8 CASC-16 (Trento, Italy, 1999)

The competition division structure stayed the same for CASC-16. 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, to allow panel members and organizers to enter the competition.[2] The division-winning systems from CASC-15 were copied 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.

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 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 only inaccurately 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 the eligible problems.

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.[3] 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.

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 systems, it might seem that a winning strategy would be for the system to merely say 'proved' as soon as presented with a 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, the organizers expose the systems to a number of non-theorems prior to the contest.[4] Given the nature of this testing it is not surprising that some unsound systems are 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 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 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).

Winners:

1. MIX: Vampire 0.0, by A. Voronkov of Manchester University.
2. UEQ: Waldmeister 799, by T. Hillenbrand of Universität Kaiserslautern.
3. SAT: OtterMACE 437, by W. McCune of Argonne National Laboratories. Saarbrücken.
4. FOF: SPASS 1.0.0T, by C. Weidenbach of Max Planck Institut für Informatik.

The major new effects and observations of CASC-16 were:

- Before CASC-16, there was, for the first time, some acrimonious debate regarding the design and implementation of the competition. The problem started with a

[2] 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.

[3] These problems are first-order logic encodings of problems from multi-modal K-logics.

[4] 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.

complaint concerning the internal table of Waldmeister, used for recognizing particular theories, and from that deciding on a strategy to use. Some of the Waldmeister categories were small, so some people claimed 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 organizers, they could not formulate a rule to sort out the tuning issue. Related to this was the issue of the the skewing of results in CASC-15 because of the ALC problems, as discussed above.

- First steps towards a robust methodology for installation and execution were taken (but not all successful). Many systems became "TPTP clean".
- There was significantly more interest from within ATP community. As a result a session at CADE was dedicated to discussing the CASC results. There was an interesting debate regarding the desirability of a focus on implementation versus attention to theory development. It seems clear that much effort was being spent on carefully constructing and tuning systems for the eligible problems, and this was felt by some to be at the expense of basic research that had not yet been well-implemented.
- 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.

9 CASC-17 (CMU, USA, 2000)

At CASC-16 there was debate about the relevance of CASC, with claims that the "problems did 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 are biased towards certain specialized provers, and that they are 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 Technische Universität München.
2. UEQ: Waldmeister 600, by T. Hillenbrand of Universität Kaiserslautern.
3. SAT: GandalfSat 1.0, by T. Tammet of Tallin Technical University.
4. FOF: VampireFOF 1.0, by A. Voronkov of Manchester University.
5. SEM: E-SETHEO 2000csp, by S. Schultz of the Technische Universität München.

The major new effects and observations of CASC-17 were:

- Again, many researchers invested in significant, year long, development in preparation for CASC.
- Only the serious players were there at CASC-17. Is CASC getting too serious? Are new systems excluded due to level of competition?
- After CASC-17 there was a realization that most systems are now uniformly good enough that non-standard, non-biased problems are safe to use in CASC. The modifications that make them non-standard, which at the time would have biased the problems towards a chosen system, by now have no real biasing effect towards any specific system.
- At CASC-17 it became obvious that there is a real need to stop tuning. Entrants were beginning to fully understand the process whereby a problem becomes eligible for CASC, and were submitting TPTP performance data that was affecting eligibility in their favour.

10 CASC-JC (Siena, Italy, 2001)

In CASC-17 no systems were tuned specifically for the SEM division. The lack of interest, and the overlap between the SEM division and the existing syntactically defined divisions, led to the demise of the SEM division in CASC-JC.[5] 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 a specialized propositional system on this result. This prompted the introduction of the EPR (Effectively Propositional) division in CASC-JC, containing problems worded in first-order form but which have a finite Herbrand Universe. Such problems are not used in any of the other divisions. As mentioned in Section 9, non-standard, non-biased problems have become eligible for use.

Following the unsoundness discovery after CASC-16, and with continued interest at CASC-17, the CASC-18 MIX division has been 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

[5] The particular SEM problems selected could also be used in the FOF division.

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-JC, unseen problems will be used. 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.

11 Conclusion

The purpose of this paper has been two-fold:

- To give an informal account of the motivation and history of CASC.
- 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 subfield of AI
  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 have 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 before. 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." CASC has pointed a way to deal with this topic.

For entrants in CASC, there have been some useful side-effects:

- The requirement that systems run completely autonomously in CASC has made researchers develop systems that can tune themselves at run time to the problem, and attempt the problem without user intervention. As well as being useful in its own right, this feature has made it possible to perform extensive automatic testing on such systems, leading to further insights and improvements.
- Developers have been motivated to make their systems robust, so that they do not crash, etc, during CASC. It has also been necessary for the systems to include mechanisms that allows them to be stopped in a clean fashion.
- The decision by some developers to tune for CASC has led to the production of automatic tuning techniques and tools. This is a useful development, as it allows the system to be tuned for particular applications by submitting sample problems.
- By having CASC as a live event at CASC, it has brought interested researchers together in an inspiring environment. As one entrant has said "Digging for and reading papers is a lot more time-consuming (and has a higher entry barrier) than sitting round the desk at the CASC dinner and swapping war stories ;-)".

We believe that the competition has 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 contributes 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.

References

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

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

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

[Sutcliffe and Suttner, 1997] G. Sutcliffe and C.B. Suttner. Special Issue: The CADE-13 ATP System Competition. *Journal of Automated Reasoning*, 18(2), 1997.

[Sutcliffe and Suttner, 1998] G. Sutcliffe and C.B. Suttner. The TPTP Problem Library: CNF Release v1.2.1. *Journal of Automated Reasoning*, 21(2):177–203, 1998.

[Sutcliffe and Suttner, 2000] G. Sutcliffe and C.B. Suttner. Evaluating General Purpose Automated Theorem Proving Systems. Technical Report 2000/2, School of Information Technology, James Cook University, Townsville, Australia, 2000.

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

12 Appendix: Choosing Numbers of Problems

An initial decision facing the organizers was to determine how many of the problems needed to be used in order for there to be some degree of confidence that a system which could solve such-and-so percentage of the chosen problems would be able to solve that same percentage of the eligible problems. The idea is that we want to know how well a system will perform on the entire eligible set, but are restricted to using only a subset of the problems due to resource limits.

For CASC-13 and CASC-14 a simple statistical approach was using for determining the minimal number of problems that should be used, as follows. The minimal number of problems was required 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]. The minimal number of problems is then:

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

From CASC-15 onwards a more sophisticated approach has been used, as explained in Section 6.

The minimal number of problems is used in determining the time limit imposed on each solution attempt, as explained below. Once the time limit has been determined, a lower bound on the total number of problems to be used is 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.

A minimal time limit of 180 seconds has been used in most CASCs. This value is based on the organizers experience with ATP systems, ensuring that it gives all systems enough time to reach reasonably deeply into their search spaces. The maximal time limit is determined using the relationship used for determining the number of problems, with the minimal number of problems as the "Number of problems". The time limit is chosen as a reasonable value within the range allowed.