The State of CASC

Geoff Sutcliffe [a] Christian Suttner [b]
[a] *Department of Computer Science, University of Miami, USA*
E-mail: geoff@cs.miami.edu
[b] *Cirrus Management, Germany*
E-mail: christian@suttner.info

The CADE ATP System Competition (CASC) is an annual evaluation of fully automatic, first order Automated Theorem Proving systems – the world championship for such systems. This paper captures the state of CASC after CASC-20, the tenth CASC, held in 2005. It provides a summarized history of CASC, details of the current design of the competition, observations and discussion of the effects of CASC on ATP, lessons learnt during CASC, and remarks regarding the past, present, and future of CASC.

Keywords: competition, automated theorem proving

1. Introduction

The International Conference on Automated Deduction (CADE) is the major forum for the presentation of new research in all aspects of automated deduction. In order to stimulate Automated Theorem Proving (ATP) system development, and to expose ATP systems to interested researchers, the CADE ATP System Competition (CASC) is held at each CADE. CASC evaluates the performance of sound, fully automatic, classical 1st order ATP systems – the world championship for such systems. The evaluation is in terms of:

– the number of problems solved
– the number of solutions output
– the average runtime for problems solved

in the context of:

– a bounded number of eligible problems, chosen from the TPTP Problem Library [29]
– a CPU time limit for each solution attempt.

The primary purpose of CASC is a public evaluation of the relative capabilities of ATP systems. Additionally, CASC aims to stimulate ATP research in general, to stimulate ATP research towards autonomous systems, to motivate implementation of robust ATP systems, to provide an inspiring environment for personal interaction between ATP researchers, and to expose ATP systems within and beyond the ATP community. Fulfillment of these objectives provides stimulus and insight for the development of more powerful ATP systems, leading to increased and more effective usage.

Over the years CASC has been a catalyst for impressive improvements in ATP, stimulating both theoretical and implementation advances. It has provided a forum at which empirically successful implementation efforts are acknowledged and applauded, and at the same time provides a focused meeting at which novice and experienced developers exchange ideas and techniques. While the positive effects have been appreciated, there have been some concerns regarding excessive emphasis on CASC, which may have detracted from underlying theoretical research and from the overarching need for application. Through successive refinement of the competition design, CASC has managed to minimize the negative effects, and overall has been of significant benefit to the development of ATP [11].

This paper captures the state of CASC after CASC-20, the tenth CASC, held at CADE-20 in 2005. It provides: a summarized history of CASC (Section 2), details of the current design of the competition (Section 3), observations and discussion of the effects of CASC on ATP (Section 4), lessons learnt during CASC (Section 5), and concluding remarks regarding the past, present, and future of CASC (Section 6).

2. The History of CASC

This section provides a summarized history of CASC, highlighting important developments, design changes, results, and effects. A full history of CASC up to CASC-JC is provided in [14].

Table 1
The Growth of CASC

Year		Divisions	Prob-	Sys-	New	CPUs
	CASC		lems	tems	sys.	
2005	20	FOF MIX SAT EPR UEQ	660	12	2	44
2004	J2	FOF MIX SAT EPR UEQ	580	15	3	60
2003	19	FOF MIX SAT EPR UEQ	420	13	2	44
2002	18	FOF MIX SAT EPR UEQ	455	13	3	42
2001	JC	FOF MIX SAT EPR UEQ	440	12	2	25
2000	17	FOF MIX SAT SEM UEQ	215	11	0	20
1999	16	FOF MIX SAT UEQ	165	14	3	10
1998	15	FOF MIX SAT UEQ	180	16	4	8
1997	14	MIX SAT UEQ	182	18	11	9
1996	13	MIX UEQ	100	13	–	8

CASC has been run at each CADE conference since 1996, including the Federated Logic Conference in 2002 and the International Joint Conferences on Automated Reasoning in 2001 and 2004. The scope of the competition has grown, from two divisions in CASC-13 (the first CASC, in 1996) to five divisions in CASC-20 (the most recent CASC, in 2005), with eight trophies being awarded at CASC-20 (see Section 3 for an explanation of the division and category structure of CASC). Thanks to the generous support of various institutions, the number of computers used has grown from 8 in CASC-13, to a peak of 60 in CASC-J2, and 44 in CASC-20. The availability of more computers has made it possible to take advantage of the continual growth of the TPTP, and the number of problems used in CASC has increased from 100 in CASC-13 to 660 in CASC-20. Table 1 provides an overview of the expansion and stabilization of CASC. Over the years 41 different systems (variants and successive versions of a system being considered to be the same system) have been entered. These are shown in Table 2, ordered by their year of last entry and the range of CASCs entered. Table 3 lists the winners in the competition divisions of the CASCs so far.

The first CASC was CASC-13 [28]. Two divisions were run: the MIX division and the UEQ division, and two ranking schemes were used in each division: the first scheme focused on the ability to find as many solutions as possible, while the second scheme measured solutions-per-unit-time. As it turned out, the two ranking schemes identically ranked the systems, in both divisions – systems that solve many problems also solve them quickly. A distinction was made between "compositional" and "monolithic" systems. The idea was that compositional systems could be made up from several distinct subsystems, and a subsystem chosen based on the given problem's characteristics, while monolithic systems ran a single calculus and strategy. As it turned out, monolithic systems solved the most problems in both divisions. Additionally, it became clear that it is hard to distinguish between monolithic and compositional systems. CASC-13 stimulated much implementation research and effort – the entrants had to produce stable, ready-to-use implementations of their systems, and most entrants made special efforts to improve the autonomous performance of their systems.

The success of CASC-13 motivated expansion in CASC-14 [32], adding the SAT division and a FOF demonstration division. The solutions-per-unit-time ranking scheme was abandoned, and ranking according to the number of problems solved, with ties decided by the average CPU times taken over problems solved, was established as the CASC ranking scheme. The compositional-monolithic distinction was abandoned, and instead problem categories were introduced in order to observe specialist capabilities of individual systems and components of compositional systems. There were many new entrants in CASC-14 – people came out of the woodwork. Many of the systems were more refined at the control level: several entrants produced an "automatic" mode, which autonomously adapts the system to the given problem according to its characteristics. Gandalf [33,34] introduced use of *strategy scheduling*, which has been used by many CASC winners since. In strategy scheduling a schedule is formed by allocating some fraction of the CPU time limit to each of several selected strategies, which are then run in succession until one finds a solution (or they all fail). In CASC-14 Waldmeister [6] began its stranglehold on the UEQ division.

In CASC-15 [30] the PEQ (Pure Equality) category was added to the MIX division, and the FOF division was promoted to a competition division. The organizational infrastructure for the competition was significantly strengthened in several ways. First, the control scripts for the competition were made publicly available, and entrants had to ensure conformance with the scripts' expectations. Second, the minimal numbers of problems to be

Table 2
CASC Entrants

System	Main entrant	13	14	15	16	17	JC	18	19	J2	20
Barcelona	Robert Nieuwenhuis	*									
CLIN	Geoff Alexander	*									
LINUS	Reinhold Letz	*									
SPTHEO	Christian Suttner	*									
Violet	Steve Greenbaum	*									
Allpaths	Joe Horton		*								
I-THOP	Koji Iwanuma		*								
OSCAR	John Pollock		*								
PROTEIN	Peter Baumgartner		*								
THINKER	Jeff Pelletier		*								
RRTP	M. Paramasivam	*	*								
Herby	Mohammed Almulla			*							
Satchmo	Tim Geisler	*		*							
DISCOUNT	Jörg Denzinger	*	*	*							
SETHEO	Reinhold Letz	*	*	*							
FDP	Peter Baumgartner				*						
SSCPA	Geoff Sutcliffe				*						
Fiesta	Robert Nieuwenhuis		*		*						
SPASS	Christoph Weidenbach	*	*	*	*	*					
Exlog	Ivan Kossey							*			
GrAnDe	Geoff Sutcliffe						*	*			
Bliksem	Hans de Nivelle			*	*	*	*	*			
SCOTT	Kahil Hodgson		*	*	*	*	*	*			
CARINE	Paul Haroun								*		
CiME	Benjamin Monate							*	*		
Dilemma	Magnus Björk									*	
SOS	John Slaney									*	
E-SETHEO	Reinhold Letz		*	*	*	*	*	*	*	*	
Gandalf	Tanel Tammet	*	*	*	*	*	*	*	*	*	
MathServ	Jürgen Zimmer										*
Prover9	William McCune										*
Darwin	Peter Baumgartner									*	*
Mace4 (aka ICGNS)	William McCune							*		*	*
Paradox	Koen Claessen								*	*	*
MUSCADET	Dominique Pastre				*	*	*		*		*
Octopus	Monty Newborn		*	*					*	*	*
THEO (aka TGTP)	Monty Newborn		*	*					*	*	*
DCTP	Gernot Stenz						*	*	*	*	*
Mace2	William McCune		*	*	*	*	*			*	*
E/EP	Stephan Schulz			*	*	*	*	*	*	*	*
Vampire	Andrei Voronkov			*	*	*	*	*	*	*	*
Otter	William McCune	*	*	*	*	*	*	*	*	*	*
Waldmeister	Thomas Hillenbrand	*	*	*	*	*	*	*	*	*	*

There is not enough space to name *all* the people who have contributed to the development of each system.

Table 3
CASC Division Winners

	FOF	MIX	SAT	EPR	UEQ
20	Vampire 8.0 (A&P)	Vampire 8.0 (A&P)	Paradox 1.3 (A&M)	DCTP 10.21p	Waldmeister 704
J2	Vampire 7.0 (A&P)	Vampire 7.0 (A&P)	Gandalf c-2.6-SAT (A) Paradox 1.0 (M)	DCTP 10.21p	Waldmeister 704
19	Vampire 5.0	Vampire 6.0 (A&P)	Gandalf c-2.6-SAT (A) Paradox 1.0 (M)	DCTP 1.3-EPR	Waldmeister 702
18	Vampire 5.0	Vampire 5.0 (A&P)	Gandalf c-2.5-SAT	E-SETHEO csp02	Waldmeister 702
JC	E-SETHEO csp01	Vampire 2.0 (A&P) E-SETHEO csp01 (A)	GandalfSat 1.0	E-SETHEO csp01	Waldmeister 601
17	VampireFOF 1.0	E 0.6	GandalfSat 1.0	-	Waldmeister 600
16	SPASS 1.00T	Vampire 0.0	OtterMACE 437	-	Waldmeister 799
15	SPASS 1.0.0a	Gandalf c-1.1	SPASS 1.0.0a	-	Waldmeister 798
14	SPASS 0.77	Gandalf	SPASS 0.77	-	Waldmeister
13	-	E-SETHEO	-	-	Otter 3.0.4z

The (...) bracketed tags refer to the assurance, proof, and model ranking classes - see Section 3.

used in each category and division, in order to attain a required degree of confidence in the representativeness of the results, was calculated using a new statistical approach [5], and this approach has been in use since. Third, all output was required to be on `stdout`, and no other output was deemed relevant. Finally, a wall-clock time limit was introduced to limit very high memory usage that causes excessive swapping. The results of CASC-15 prompted a realization for some entrants that their systems had fallen behind the rapidly improving state-of-the-art. At the same time, it was observed that rather than entering systems that were designed to be general purpose, some entrants had spent considerable time tuning their systems for only the eligible problems. This "over-tuning" was considered by some to be unproductive in the broader context of ATP development, and it continued to be a contentious issue until CASC-JC. 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 CASCs, and good performance in CASC was starting to affect publications and grant funding. For the first time CASC attracted newspaper publicity, with an article appearing in the local press. Some time after CASC-15 it was found that one of the entrants was unsound, and was retrospectively disqualified. This was the first, but not the last, retrospective disqualification from CASC.

Before CASC-16 [21] there was some acrimonious email exchange regarding the design of the competition. The exchange started with a complaint concerning the way that Waldmeister adapts to the given problem, but quickly expanded to a range of issues. Stimulated by, and in response to, the debate, some changes were introduced to limit tuning for the eligible problems. First, the lists of eligible problems were not published until after the systems had been installed on the competition machines. Second, 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. Third, a limit was imposed on the number of very similar problems in any division or category. Despite these changes, the debate over the design of CASC continued during the CASC results presentation at CADE-16, and extended to a discussion regarding the desirability of a focus on implementation versus attention to theory development. It seemed clear that much effort was being spent on carefully constructing and tuning systems, and this was felt by some to be at the expense of basic research. An important feature of CASC, adopted from CASC-16 onwards, was to enter the winning systems from the previous competition in their respective divisions (the competition archive provides access to those systems' executables and source codes). This provides benchmarks against which the performance of the new systems can be judged, making it possible to make definitive statements about the progress of ATP (see Section 4). About a month after CADE-16 two of the entered systems, including the announced winner of the MIX division, were found to be unsound in certain rare circumstances. The competition panel thus retrospectively disqualified the two systems from being ranked in the competition. This led to a revival of interest in proof output and verification.

In CASC-17 [22] the organizers introduced an emphasis on the usability of the systems, in terms of their installation and execution. System installation for CASC-17 required that the entrants supply the organizers with installation packages. The motivation was to encourage developers to make installation and practical usage easier for potential users. For only CASC-17 a "semantic" division, the SEM division, was run. The SEM division used FOF problems based on a specified encoding of a chosen semantic domain, to evaluate how well systems could be tuned to perform in a realistic application domain. In CASC-17 theorems based on the TPTP's von Neumann-Bernays-Gödel set theory axiomatization [15] were used. Despite enthusiasm expressed the previous year by CASC-16 entrants, no systems were especially tuned for the SEM division in CASC-17, and the division was never run again. The performance of the systems in CASC-16, on the "ALC" problems in the `SYN` domain, highlighted the issue of effectively propositional problems - problems with a finite Herbrand universe. It was clear that these problems are particularly well suited to specialized systems, and not suitable for the evaluation of general purpose first-order systems. Motivated by these observations, effectively propositional problems were made ineligible for CASC-17 (but were reintroduced in a separate division in CASC-JC). For CASC-17 many researchers invested in significant, year long, development in preparation for the competition. Some entrants made special efforts to make their systems effective for TPTP problems. The tuning was not only for problems that were predicted to be eligible for the competition, but also for TPTP problems in general. The entrants were beginning to understand that tuning for the TPTP in general, and submitting corresponding performance data, affected the eligibility of problems for CASC, thus providing leverage for the particular strengths of their systems.

In CASC-JC [27] the EPR (Effectively Propositional) division was added, and the SAT division was divided into two categories: SEQ (SAT with equality) and SNE (SAT without equality). In the MIX division, the use of separate assurance and proof ranking classes was introduced. An assurance class is ranked according to the number of problems solved, while a proof class is ranked according to the number of problems solved with an acceptable proof output. This change aimed to encourage research into proof presentation, and the implementation of proof generation and verification as part of an integrated reasoning process. The effort required to produce proofs was evident. In particular, some systems solved some problems within the time limit but ran overtime while building the proofs. Up to CASC-17, non-standard problems (problems based on an incomplete or explicitly redundant axiomatization) had been excluded from CASC, as there was a perceived danger that such problems might be biased towards a particular ATP system. Between CASC-17 and CASC-JC it was noted that problems explicitly biased towards or against any particular calculus or system are tagged as such in the TPTP, and only they need to be excluded from CASC. Therefore non-standard problems became eligible from CASC-JC, while biased problems remained ineligible (there was much discussion about the `FLD` problems, which had been tagged as biased, but were retagged as non-standard before CASC-JC). An important change introduced from CASC-JC was to take the problems from an unreleased version of the TPTP. This makes tuning for new problems in that TPTP version impossible. Overtuning for the problems in the preceding TPTP version is potentially disadvantageous because it can degrade performance on the new problems, with a consequent degradation in overall performance. At CASC-JC there was generally strong performance on the new problems, countering the concern that systems had been overtuned for the competition: in the case that systems were tuned using TPTP problems, then that tuning also worked for the new problems, and therefore seemed to be effective in general. In the environment of the combined IJCAR conference, observers with a broad range of perspectives were evidently interested in the competition and its outcomes. In particular, it was pleasing to see some commercial interest in the best performing systems.

In CASC-18 [25] the rules regarding limitations on tuning for TPTP problems were clarified, and have remained in effect since then. The changes required that strategies and strategy selection be transparently general purpose, and expected to be useful beyond the TPTP. In order to expose strategies and strategy selection techniques, entrants are required to describe the strategies used, why they are effective, and how they are selected for given problems. CASC-18 introduced an escape clause for systems that might be found to be unsound after the competition, and hence liable for retrospective disqualification: the entrants can convince the competition panel that the unsoundness did not manifest itself in the competition. Additionally, a statute of limitations was adopted for retrospective disqualification: no retrospective disqualification is possible after the competition report has been published. Neither rule had to be invoked for CASC-18. CASC-18 saw the clear emergence of strategy scheduling as an important technique for solving the most problems within a known, not-too-small, time limit. The top systems in all except the UEQ division employed strategy scheduling. Classic examples of strategy scheduling systems are E-SETHEO [20], Gandalf, and Vampire [16]. The extensive use of strategy scheduling produced a divergence between "slow" strategy scheduling systems that solve a few more problems, and "fast" single strategy systems such as E [17] and Paradox [2]. In CASC-18 the EPR division contained some very large new `SYN` problems, converted from modal logic problems [8]. Several of the systems were unable to parse and store these, thus focusing attention on the need for efficient input parsing and large capacity data structures.

In CASC-19 [26] the SAT division became ranked in two classes, an assurance class, and a model class ranked by the number of problems solved with an acceptable model output. An important change was made to the way the numbers of problems to be used in each division and category are decided. Prior to CASC-19 equal numbers had been used in each category, but it had become evident that certain categories and divisions, especially the NNE category of the MIX division, are more important than others. From CASC-19 onwards, the numbers of problems in the categories in the various divisions have been (roughly) proportional to the numbers of eligible problems. The results of CASC-19 confirmed the value of strategy scheduling for solving as many problems as possible, and again every division except UEQ was won by a strategy scheduling system. In the MIX division the separation between the top performing systems and the "also-rans" became very obvious. Analysis showed that the top group includes SPASS [35], E, E-SETHEO, Gandalf, and Vampire. In the UEQ division, for the first time since CASC-16, Waldmeister did not completely dominate. This was due to Waldmeister's inability to solve some new lattice theory problems, whose algebraic structure it could not recognize.

In CASC-J2 [23] the FOF division became ranked in two classes, an assurance class and a proof class. The FOF division contained some new, very large `ALG` problems that none of the systems in the FOF division could convert to CNF. This indicated a need for research and development of better FOF to CNF converters. The divergence between the top systems and the other systems in the MIX division, noted in CASC-19, remained salient, and was also observed in the FOF and SAT divisions. After CASC-J2 the entrants of one system discovered that their system was unsound with respect to unsatisfiability, and it was thus retrospectively disqualified by the competition panel. A repaired version of the system was retrospectively entered into the demonstration division. CASC-J2 was a most orderly and uncontentious CASC. The years of controlled refinement of the design, coupled with an increasing number of returning entrants who already understood the expectations and possible outcomes, yielded a smooth running competition.

CASC-20 was, at the time of writing, the most recent CASC. For the first time the FOF division equaled the MIX division in size, and thus stature. This change reflects the increased number of FOF contributions to the TPTP (550 new FOF problems between TPTP v3.0.1 and TPTP v3.1.0, in contrast with only 168 new CNF problems), and the corresponding increased use of FOF in applications. For CASC-20 there was a drop in the number of systems entered, with some of the "regulars", e.g., E-SETHEO and DCTP, absent. At the same time four new systems were entered (although two withdrew due to lack of development). This turnover of systems indicated continued activity in ATP system development, and appropriate recognition when a system may have reached the end of it's development path. After CASC-20 the entrant of one system discovered that the system was unsound for syntactically identifiable FOF problems, and the system was thus retrospectively disqualified by the competition panel. A repaired version of the system was retrospectively entered into the demonstration division. Although retrospective disqualification is an undesirable necessity, it does provide motivation for developers to repair and carefully test their systems for soundness. The consistent strong performances of a few systems in the last few CASCs established these systems as the preferred general purpose ATP systems for research and application. CASC makes these systems publicly available, and thus provides starting points from which new developers can leverage the knowledge of highly experienced ATP system developers.

3. The Design of CASC

In order to obtain the full benefits of a competition, a thoroughly organized event, with an unambiguous and motivated design, is necessary. In 1993 Ross Overbeek ran a very specialized competition at CADE-11, using small sets of specifically selected problems [12]. His competition allowed a detailed analysis and comparison of the performances of the ATP systems on the selected problems, but did not aim to evaluate the general usefulness of the systems. No other formally organized competitions for ATP have taken place, thus the design of CASC had to be developed from first principles. This section reviews the design of CASC, explaining the philosophical underpinning of the competition and the design process, and technical details of the current design.

In order for a comparison of different ATP systems to make sense, it is necessary that all the systems should be attempting to capture a common notion of truth, as is described in the Realist viewpoint in [13], whereby all the differing proof systems are viewed merely as different ways of demonstrating facts about the same abstract realm of logic. Given this commonality across all systems, it is possible to design an ATP competition that determines winners, relative to some clearly specified constraints. The CASC design has several aspects: how the competition is divided into divisions and categories, what problems are eligible for use, how many and which problems are used, what resource limits are imposed, how the systems are ranked, the properties required of the ATP systems that compete, and what organizational rules apply. In some cases inevitable constraints have emerged, while for others there have been several choices, and decisions have had to be made.

For some aspects of the CASC design, the original decisions have not changed over the years, while for others there has been expansion and adaptation (as doumented in Section 2). Each year has revealed further issues, arising from changes in ATP techniques and usage, the experience of the organizers (a few decisions about the design were not optimal the first time!), and the changing hardware and software infrastructure that is available. Entrants have also been forthcoming with ideas, criticisms, and suggestions regarding the design. Every change to the design has been the result of careful consideration and discussion. Some ideas, which at first glance seemed to offer improvement to the design, have been rejected, while others have been adopted. A guiding principle has been to avoiding perturbing the fundamental nature of the competition. As a result the general design has been reasonably stable, and through the continuity of the event, the results allow performance comparisons with previous and future years.

As is the case in all competitions, and regardless of the care with which the competition has been designed, unforeseen circumstances arise. In order to provide for impartial resolution of such matters, CASC is overseen by a panel of knowledgeable researchers who are not participating in the event. Once the design details of each CASC have been finalized, only the panel has the right to make changes or exceptions.

3.1. Divisions and Categories

CASC is divided into divisions according to problem and system characteristics. There are competition divisions in which systems are explicitly ranked, and a demonstration division in which systems demonstrate their abilities without being formally ranked.

Each competition division uses problems that have certain logical, language, and syntactic characteristics, so that the ATP systems that compete in the division are, in principle, able to attempt all the problems in the division. Some divisions are further divided into problem categories. The categories make it possible to analyze, at a more fine grained level, which systems work well for what types of problems. Section 3.2 explains the source and selection of the problems that are eligible and that are used, in each division and category. The categories have no effect on the competition rankings, which are made at only the division level, as explained in Section 3.4.

The characteristics used to define the divisions and categories of CASC include:

- Whether the problems are presented in first-order form (FOF problems) or in clause normal form (CNF problems).
- Whether or not the problem is a theorem. For CNF problems this separates unsatisfiable and satisfiable clause sets.
- Whether a problem is really non-propositional or effectively propositional. Really non-propositional means with an infinite Herbrand universe, while effectively propositional means non-propositional with a finite Herbrand universe.
- The extent to which equality is present in the problem.
- For CNF problems, whether or not the clauses are all Horn.

Since there are important differences in the types of problems, and significant differences in the techniques required to solve the different types of problems (e.g., a system designed to show that a set of clauses is satisfiable is typically not intended to also prove FOF theorems, and so on), CASC is run in divisions based on these characteristics.

The **MIX** division uses mixed CNF really non-propositional theorems (unsatisfiable clause sets). *Mixed* means Horn and non-Horn problems, with or without equality, but not unit equality problems (see the UEQ division below). The MIX division has five problem categories: HNE - Horn problems with no equality, HEQ - Horn problems with some (but not pure) Equality, NNE - Non-Horn problems with No Equality, NEQ - Non-Horn problems with some (but not pure) Equality, and PEQ - Pure Equality problems. The **FOF** division uses FOF non-propositional theorems (axioms with a provable conjecture). The FOF division has two problem categories: FNE - FOF problems with No Equality, and FEQ - FOF problems with Equality. The **SAT** division uses mixed CNF really-non-propositional non-theorems (satisfiable clause sets). The SAT division has two problem categories: SNE - SAT problems with No Equality, and SEQ - SAT with Equality. The **EPR** division uses CNF effectively propositional theorems and non-theorems (unsatisfiable and satisfiable clause sets). The EPR division has two problem categories: EPT - Effectively Propositional Theorems (unsatisfiable clauses), and EPS - Effectively Propositional non-theorems (Satisfiable clauses). The **UEQ** division uses unit equality CNF really non-propositional theorems (unsatisfiable clause sets).

The demonstration division is available for systems that must be run on specialist hardware (e.g., a network of computers), or cannot be entered into the competition divisions for any other reason (e.g., the entrant is a competition organizer or panel member). Demonstration division entries specify which competition divisions' problems are to be used. The results are presented along with the competition divisions' results, but may not be comparable with those results. No ranking is done in the demonstration division.

3.2. Problems

The problems for CASC are taken from the TPTP. The TPTP version used for the competition is not released until after the system installation deadline, so that new problems in that TPTP version have not been seen by the entrants. The problems must have a TPTP difficulty rating [31] in the range 0.21 to 0.99, as computed using performance data from systems submitted by an announced deadline. Problems in that difficulty range are expected to be solved by some but not all of the systems, thus providing differentiation between the systems. Problems that were specifically designed to be biased towards or against any particular calculus or system, as documented in the TPTP, are excluded.

The minimal numbers of problems that have to be used in each division and category, to ensure sufficient confidence in the competition results, are determined statistically from the numbers of eligible problems in each division and category [5]. This minimal numbers of problems is used in determining the CPU time limit imposed on each solution attempt - see Section 3.3. The CPU time limit is in turn used in computing a lower bound on the total number of problems to be used, according to the following relationship:

$$\#Problems = \frac{\#Computers * TimeAvailable}{\#Systems * CPULimit}$$

It is a lower bound on the total number of problems because it assumes that every system uses all of the CPU time limit for each problem. Since some solution attempts succeed before the CPU time limit is reached, more problems can be used. The numbers of problems used in each division and category are determined according to the judgement of the competition organizers. The numbers of problems used in the categories in the various divisions are (roughly) proportional to the numbers of eligible problems in the categories (after taking into account the limitation on very similar problems, as described below).

The problems used are randomly selected from the eligible problems at the start of the competition, based on a seed supplied by the competition panel. The selection is constrained so that no division or category contains an excessive number of very similar problems. This is achieved using the very-similar-problems lists that are distributed with the TPTP. The limit on the number of very similar problems used from any one list is:

$$\frac{\#NumberOfProblemsInCategory}{\#Lists + 1}$$

provided that the number of eligible problems in each very-similar-problems list is at least that size. If the number of eligible problems in a very-similar-problems list is less than that size, then that very-similar-problems list is dropped from consideration (i.e., the problems in that list are all eligible) and the number is recalculated. The selection mechanism is also biased to select problems that are new in the TPTP version used, until 50% of the problems in each category have been selected, after which random selection (from old and new problems) continues. The actual percentage of new problems used depends on how many new problems are eligible and the limitation on very similar problems.

To ensure that no system receives an advantage or disadvantage due to the specific presentation of the problems in the TPTP, the `tptp2X` utility (distributed with the TPTP) is used to replace all predicate and function symbols with new symbols, randomly reorder the formulae and the clauses' literals, randomly reverse the unit equalities in UEQ problems, add and remove equality axioms as required by the ATP systems, and output the problems in the formats required by the ATP systems (the formula type information, one of `axiom`, `hypothesis`, `conjecture`, or `negated_conjecture`, may be included in the final output of each formula). Further, to prevent systems from recognizing problems from their file names, symbolic links are made to the selected problems, using meaningless names for the symbolic links. The problems are specified to the ATP systems using the symbolic link names.

3.3. Resource Limits

In the competition divisions a CPU time limit is imposed on each solution attempt. A minimal CPU time limit of 240 seconds is used. The maximal CPU time limit is determined using the relationship used for determining the number of problems, with the minimal number of problems as the $\#Problems$. The CPU time limit is chosen as a reasonable value within the range allowed, and is announced at the competition. A wall clock time limit is imposed in addition to the CPU time limit, to constrain very high memory usage that causes swapping. The wall clock time limit is double the CPU time limit.

In the demonstration division, each entrant can choose to use either a CPU or a wall clock time limit, whose value is the CPU time limit of the competition divisions.

3.4. System Evaluation

All the divisions have an assurance ranking class, ranked according to the number of problems solved (a "yes" output, giving an assurance of the existence of a proof). The MIX, FOF, and SAT divisions additionally have a proof/model ranking class, ranked according to the number of problems solved with an acceptable proof/model output on `stdout`. Ties are broken according to the average CPU times over problems solved. All systems are automatically ranked in the assurance classes, and are ranked in the proof/model classes if they output acceptable solutions. In the assurance classes, and the EPR and UEQ divisions, the ATP systems are not required to output solutions. However, systems that do output solutions on `stdout` are highlighted in the presentation of results.

The competition panel judges whether or not the systems' proofs/models are acceptable. The criteria include:

- Derivations must be complete, starting at formulae from the problem, and ending at the conjecture (for axiomatic proofs) or a false formula (for proofs by contradiction, including CNF refutations). For proofs of FOF problems by CNF refutation, the conversion from FOF to CNF must be adequately documented.
- Derivations must show only relevant inference steps.
- Inference steps must document the parent formulae, the inference rule used, and the inferred formula.
- Inference steps must be reasonably fine-grained.
- An unsatisfiable set of ground instances of clauses is acceptable for establishing the unsatisfiability of a set of clauses.
- Models must be complete, documenting the domain, function maps, and predicate maps. The domain, function maps, and predicate maps may be specified by explicit ground lists (of mappings), or by any clear, terminating algorithm.

During the competition, for each ATP system, for each problem attempted, three items of data are recorded: whether or not a solution was found, the CPU time taken, and whether or not a solution was output on `stdout`. The systems are ranked from this performance data, and trophies are awarded to division winners.

3.5. System Entry

To be entered into CASC, systems have to be registered, and an entrant has to be nominated to handle all issues arising before and during the competition. Systems can be entered at only the division level, and can be entered into more than one division (a system that is not entered into a competition division is assumed to perform worse than the entered systems, for that type of problem). Entering many similar versions of the same system is deprecated, and entrants may be required to limit the number of system versions that they enter. The division winners from the previous CASC are automatically entered into their divisions, to provide benchmarks against which progress can be judged. After the competition all systems' source code is made publically available on the CASC WWW site.

Systems are required to have the following properties:

- Competition division systems have to run on a single locally provided standard UNIX computer supplied by the competition organizers (the *general hardware*). ATP systems that cannot run on the general hardware can be entered into the demonstration division.
- Systems have to be fully automatic, i.e., any command line switches have to be the same for all problems.
- Systems have to be sound. At some time before the competition all the systems in the competition divisions are tested for soundness. Non-theorems are submitted to the systems in the MIX, FOF, EPR, and UEQ divisions, and theorems are submitted to the systems in the SAT and EPR divisions. Claiming to have found a proof of a non-theorem or a disproof of a theorem indicates unsoundness. The soundness testing eliminates the possibility of an ATP system simply delaying for some amount of time and then claiming to have found a solution. (Since first order logic is semidecidable, there can be no absolute test of soundness. Empirical testing provides strong evidence of soundness.)
- Systems do not have to be complete in any sense, including calculus, search control, implementation, or resource requirements.
- The precomputation and storage of any information specifically about TPTP problems is not allowed. Strategies and strategy selection based on the characteristics of a few specific TPTP problems are not allowed, i.e., strategies and strategy selection must be general purpose and expected to extend usefully to new unseen problems.
- For every problem solved, the system's solution process must be reproducible by running the system again.

It is assumed that each entrant has read and understood the competition design, and has complied with the competition rules. A "catch-all" rule is used to deal with any unforeseen circumstances: *No cheating is allowed.* The competition panel is allowed to disqualify entrants due to unfairness, and to adjust the competition rules in case of misuse.

4. The Effects of CASC

The effects of CASC can be seen from two perspectives: that of ATP system development, and that of the image of ATP.

CASC has had two main effects on ATP system development. First, new strategies and techniques have been developed to increase the range of problems that can be solved by individual systems, and second, the quality of implementations has improved. Possibly the most important improvement has been in the selection of strategies according to the characteristics of the given problem – the "auto-mode"s now available in almost all ATP systems. There have been significant developments in this area, including deeper understanding of what problem characteristics are important for what aspects of strategy selection, the examination of the input to detect the domain structure of the problem (e.g., in Waldmeister [7] and Vampire), the use of machine learning techniques to optimize the choice of strategy (e.g., in E-SETHEO [19] and E), and the use of strategy scheduling (e.g., in E-SETHEO, Gandalf, and Vampire). The selection of strategies according to problem characteristics has been a contentious issue in CASC, as explained in Section 2, and various measures have been added to the competition design to counter excessive tuning, e.g., the emphasis placed on new, unseen problems (see Section 3). Although excessive tuning for CASC is undesirable, research into tuning has led to the production of automatic tuning techniques and tools. This is a useful development, as it allows a system to be tuned for particular applications by submitting sample problems.

There have been other developments stimulated by CASC. The publicized unsoundness of systems after CASCs 15 and 16 generated interest in the production and verification of ATP system output, leading to the progressive introduction of proof/model ranking classes to the MIX, FOF, and SAT divisions. This has prompted several developers to produce and improve their systems' output. The increasing importance of the FOF division has encouraged the development and refinement of FOF to CNF converters, that are used to preprocess FOF problems before handing the clauses to the core inference engine. An important aspect is the adequate documentation of the conversion, as required for the FOF division's proof ranking class. The failure of the CNF-based systems in the FOF division of CASC-J2 to cope with the large formulae in the new `ALG` problems further stimulated research and development of better FOF to CNF converters. Techniques that can deal directly with FOF problems may also be improved.

In addition to improvements in the logical features of ATP systems, CASC has obliged developers to deliver systems that are robust in terms of installation and execution. Prior to participation in CASC, many systems' installation required a complicated sequence of steps, individually controlled from a terminal. The demand for installation packages since CASC-17 has vastly improved this situation. The runtime behavior of systems has also had to be debugged and engineered, so that systems can be easily started and stopped by control software, as would be expected when the systems are embedded in larger projects. The use of the very large `SYN` problems in CASC-18 compelled some developers to improve the parsing capabilities of their systems. As well as being useful in its own right, the improved stability and autonomy of ATP systems has made it possible to perform extensive automatic testing of ATP systems, leading to further insights and improvements.

The CASC events and resources have in themselves had an effect on the development of ATP. Interested researchers have been brought together at each CASC in an inspiring environment, and there have been fruitful exchanges of ideas. 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 ;-)". The online CASC archives, including the competition problems and the systems' source and binary codes, allows developers and users to experiment with the systems. The CASC control software has been adopted for use in other projects, e.g., [4].

The image of ATP has been influenced by CASC, consistent with the stated aim "to expose ATP systems within and beyond the ATP community". Within the ATP community the competition is an established and well known event. At the conference it attracts attention and interest, and the annual reports expose ATP to a much wider audience. The competition shows how theoretical advances are embodied in real implementations. Prior to CASC there was almost no effective way to gain recognition for the significant knowledge and effort required to produce a high-performance ATP system, and this void has been well filled by CASC. The competition has provided a platform from which the importance of implementation has been argued, and there is evidence of increased recognition for implementations in calls for papers and accepted publications.

Beyond the ATP community, many ATP system users base their choice of which systems to use on the performance of the systems at CASC. In some cases industrial contracts have resulted from such choices. Within academia, performance in CASC is used as motivating evidence in grant applications, promotion and tenure applications, and requests for computing infrastructure. As well as publicizing ATP systems, CASC has promoted awareness of the TPTP as a common resource. ATP system users have made contributions to the TPTP so that developers can tackle these problems and develop techniques for efficiently solving them. As a result the ATP developers and systems specifically provide the services required by contributing users - see, e.g., [3], [10], and [4].

In each CASC since CASC-16 the previous year's division winners have been automatically entered, thus providing benchmarks against which the new systems can be judged. Table 4 shows that, in general, each year new systems outperform the previous winners. This provides strong evidence that there is progress in ATP systems. (Further evidence of progress in ATP, e.g., declining TPTP problem ratings and the solution of previously unsolved problems, is given in [24]. CASC is a contributing cause of this improvement.) It is evident that there are some "big name" players in CASC, who have been successful over multiple years, and in some cases in multiple divisions. Although the effect of CASC on these systems is highly visible, it is certainly the case that less lauded systems have also been affected, and their performance has been improved through participation in CASC.

5. The Lessons of CASC

There have been several keys to the success of CASC, and several lessons learnt along the way. This section documents some of the main points.

One of the very core reasons why CASC became possible, and why it was successful right from the start, was that the TPTP problem library was already established and in accepted use by the ATP community. This provided a platform upon which to build CASC, providing problems that had already passed community scrutiny. The `tptp2X` utility was already battle-hardened, and available to prepare problems for the ATP systems. It is clear that the availability and ongoing maintenance of a commonly accepted library of test problems gives the development and ongoing organization of a competition an advantage. In contrast, competitions that are not supported by such a library have an added burden of finding and preparing suitable problems, e.g., the SMT-COMP [1], which was developed in parallel with the SMT-LIB that provides the problems, and the SAT competition [18], which at times has some difficulties finding suitable problems. In the extreme, the lack of a commonly accepted source of problems may make it hard for a competition to be run at all, e.g., CISC [9]. An important feature of the TPTP for CASC is the problem ratings. The ratings provide an accurate measure of how difficult the problems are for state-of-the-art ATP systems, which in turn makes it possible to select appropriately difficult problems for CASC to differentiate between the systems. Additionally, the carefully crafted rating scheme [31] provides the principles for the CASC rating scheme, which provides a realistic and stable ranking of the systems. A key to sustaining the value of CASC in the future is continued support for, and growth of, the TPTP. Developers and users are strongly encouraged to contribute to the TPTP, particularly problems from emerging commercial applications of ATP.

An important facet of the CASC design is the use of unseen problems. The decision to hold back the release of the TPTP version used in each CASC until the event has started, was motivated by and has contributed to convincing developers to limit their tuning for existing TPTP problems. Improvements to systems are thus general purpose and expected to extend usefully to new problems and applications. The regular success of systems on new problems in CASC has provided evidence that using the TPTP for testing newly implemented ideas, and gauging the quality of the ideas based on the results, does not just lead to systems that can solve only TPTP problems.

The CASC decision to maintain an "open source" policy has many benefits. From the point

Table 4

Performance of previous CASC division winners

	Division winner Problems/Solved by winner/Solved by previous winner (Ranking of previous winner) FOF	MIX	SAT	EPR	UEQ
20	Vampire 8.0 150/131/129 (2nd)	Vampire 8.0 150/137/133 (2nd)	Paradox 1.3 120/117/–	DCTP 10.21p 120/117/117 (1st)	Waldmeister 704 120/110/110 (1st)
J2	Vampire 7.0 88/80/75 (2nd)	Vampire 7.0 200/180/157 (4th)	Gandalf c-2.6-SAT 100/95/95 (1st)	DCTP 10.21p 80/79/72 (3rd)	Waldmeister 704 100/100/94 (2nd)
19	Vampire 5.0 70/57/57 (1st)	Vampire 6.0 140/120/113 (4th)	Gandalf c-2.6-SAT 70/63/49 (3rd)	DCTP 1.3-EPR 70/66/57 (3rd)	Waldmeister 702 70/56/56 (1st)
18	Vampire 5.0 70/55/–	Vampire 5.0 175/158/152 (3rd)	Gandalf c-2.5-SAT 70/61/44 (3rd)	E-SETHEO csp02 70/60/–	Waldmeister 702 70/70/70 (2nd)
JC	E-SETHEO csp01 90/75/72 (2nd)	Vampire 2.0 120/93/81 (4th)	GandalfSat 1.0 90/48/48 (1st)	E-SETHEO csp01	Waldmeister 601 90/69/69 (2nd)
17	VampireFOF 1.0 60/53/51 (2nd)	E 0.6 75/57/37 (5th)	GandalfSat 1.0 30/25/21 (4th)		Waldmeister 600 30/30/29 (2nd)
16	SPASS 1.00T 30/22/19 (3rd)	Vampire 0.0 75/51/39 (4th)	OtterMACE 437 30/16/9 (3rd)		Waldmeister 799 30/30/19 (2nd)
15	SPASS 1.0.0a 40/39/–	Gandalf c-1.1 80/61/–	SPASS 1.0.0a 30/22/–		Waldmeister 798 30/30/–

Due to special software requirements E-SETHEO csp01 could not be installed on the CASC-18 hardware, and thus there is no data for the previous winner in the CASC-18 FOF and EPR divisions. Due to unsoundness of Gandalf c-2.6 on the new problems in the TPTP version used for CASC-20, Gandalf c-2.6 was not entered into CASC-20, and thus there is no data for the previous winner in the CASC-20 SAT division.

of view of the evaluation, it is important that it is possible to check that a system is not violating any of the competition design rules. Although checking through source for violations is a daunting task, the fact that the source is publicly available provides a strong disincentive for cheating. Access to the source also ensures reproducibility of the competition performance - researchers are assured of the possibility of downloading, recompiling for the local environment, and obtaining at least similar results as in the competition. The source forms the only authoritative reference for a particular version of a system. Many practically important techniques are never published, and if they are, the published descriptions are necessarily simplified and often ambiguous. Access to the source allows other developers to understand in detail how a particular technique works and can be effectively implemented. From an academic and scientific point of view, making source code available provides stepping stones for researchers to build better systems. New developers can leverage on the experience of others, and as a result progress in the field is supported. The primary argument for a "closed source" approach is to protect commercial interests. Although it is possible for a system's source to be taken from the CASC WWW site and integrated (without compensation to the developer) into commercial software, this seems highly improbable. With an appropriate license agreement in the source, a serious commercial enterprise would not take such a step. Indeed, the real value is in the developer, not in the source, and it would be far more likely that the enterprise would want to obtain use of the ATP system in (financial) cooperation with the developer.

In the face of the desirability of more extensive competitions, it is important to avoid being over-ambitious. The development of CASC has shown that contentious changes, or changes that were proactive rather than reactive, did rarely last. In contrast, changes introduced to account for issues noted and discussed by many members of the community, led to continuous strengthening of the conceptual foundation of CASC. Therefore the development of CASC focuses on continuous, conservative refinements, driven by insights for maintaining neutral and relevant evaluations of ATP systems, and minimizing bias of the direction in which research should go. A successful limited competition with meaningful results is preferred over a larger competition which is not accepted by the ATP community.

6. Conclusion

The history of ATP is rich with theoretical research results and system developments that required evaluation, in order to determine which ideas and techniques are viable, and to integrate them into systems that are more flexible and powerful than before. For all these goals, empirical system evaluation is a crucial research tool. This paper has described how the underlying need for empirical evaluation has been translated into the annual ATP system competition. The underlying need motivated the development of a communally accepted benchmark set – the TPTP, and the specification of formal schemes for evaluating ATP systems. These in turn provided the practical foundations for the design and development of CASC. CASC is now an established and influential event in the ATP calendar.

CASC fulfills its aims: evaluation of the relative capabilities of ATP systems, stimulation of ATP research, providing motivation for improving implementations, and having an exciting event that exposes ATP systems to researchers within and beyond the ATP community. The significant efforts that go into developing the ATP systems receive public recognition. The competition provides an overview of which researchers and research groups have decent, running, fully automatic ATP systems.

The divergence between the top systems and the "also-rans" in the MIX divisions, noted in CASC-19 [26], and subsequently also observed in CASC-J2 and CASC-20, is noteworthy. The top few systems are the product of deep theoretical and practical knowledge, coupled with significant development effort. While the developers of these top systems reap the rewards of their unique situation, the field as a whole would benefit from a wider range of available state-of-the-art systems. This is an issue that can be addressed through explicit support and recognition of implementation efforts. CASC benefits from the entry of new systems, which illustrate the potential of new calculi and strategies. It is hoped that more developers will realize that the benefits of being a part of CASC far outweigh any perceived disadvantages of not being one of the top few performers.

References

[1] C. Barrett, L. de Moura, and A. Stump. SMT-COMP: Satisfiability Modulo Theories Competition. In K. Etessami and S. Rajamani, editors, *Proceedings of the 17th International Conference on Computer Aided Verification*, number 3576 in Lecture Notes in Computer Science, pages 20–23. Springer-Verlag, 2005.

[2] K. Claessen and N. Sorensson. New Techniques that Improve MACE-style Finite Model Finding. In P. Baumgartner and C. Fermueller, editors, *Proceedings of the CADE-19 Workshop: Model Computation - Principles, Algorithms, Applications*, 2003.

[3] S. Colton and G. Sutcliffe. Automatic Generation of Benchmark Problems for Automated Theorem Proving Systems. In B. Faltings, editor, *Proceedings of the 7th International Symposium on Artificial Intelligence and Mathematics*, pages AI–M 4–2002., 2002.

[4] E. Denney, B. Fischer, and J. Schumann. Using Automated Theorem Provers to Certify Auto-generated Aerospace Software. In M. Rusinowitch and D. Basin, editors, *Proceedings of the 2nd International Joint Conference on Automated Reasoning*, number 3097 in Lecture Notes in Artificial Intelligence, pages 198–212, 2004.

[5] 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.

[6] T. Hillenbrand. Citius altius fortius: Lessons Learned from the Theorem Prover Waldmeister. In I. Dahn and L. Vigneron, editors, *Proceedings of the 4th International Workshop on First-Order Theorem Proving*, number 86.1 in Electronic Notes in Theoretical Computer Science. Elsevier Science, 2003.

[7] T. Hillenbrand, A. Jaeger, and B. Löchner. Waldmeister - Improvements in Performance and Ease of Use. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction*, number 1632 in Lecture Notes in Artificial Intelligence, pages 232–236. Springer-Verlag, 1999.

[8] U. Hustadt and R. Schmidt. Using Resolution for Testing Modal Satisfiability and Building Models. *Journal of Automated Reasoning*, 28(2):205–232, 2002.

[9] D. Hutter and A. Bundy. The Design of the CADE-16 Induction Theorem Prover Contest. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction*, number 1632 in Lecture Notes in Artificial Intelligence, pages 374–377. Springer-Verlag, 1999.

[10] J. Meng and L. Paulson. Experiments on Supporting Interactive Proof Using Resolution. In M. Rusinowitch and D. Basin, editors, *Proceedings of the 2nd International Joint Conference on Automated Reasoning*, number 3097 in Lecture Notes in Artificial Intelligence, pages 372–384, 2004.

[11] R. Nieuwenhuis. The Impact of CASC in the Development of Automated Deduction Systems. *AI Communications*, 15(2-3):77–78, 2002.

[12] R. Overbeek. The CADE-11 Competitions: A Personal View. *Journal of Automated Reasoning*, 11(3):315–316, 1993.

[13] F.J. Pelletier. The Philosophy of Automated Theorem Proving. In Mylopolous J. and R. Reiter, editors, *Proceedings of the 12th International Joint Conference on Artificial Intelligence* , pages 1039–1045. Morgan-Kaufmann, 1991.

[14] F.J. Pelletier, G. Sutcliffe, and C.B. Suttner. The Development of CASC. *AI Communications*, 15(2-3):79–90, 2002.

[15] A. Quaife. *Automated Development of Fundamental Mathematical Theories*. Kluwer Academic Publishers, 1992.

[16] A. Riazanov and A. Voronkov. The Design and Implementation of Vampire. *AI Communications*, 15(2-3):91–110, 2002.

[17] S. Schulz. E: A Brainiac Theorem Prover. *AI Communications*, 15(2-3):111–126, 2002.

[18] L. Simon, D. Le Berre, and E. Hirsch. The SAT2002 Competition Report. *Annals of Mathematics and Artificial Intelligence*, 43(1-4):307–342, 2005.

[19] G. Stenz and A. Wolf. Strategy Selection by Genetic Programming. In A. Kumar and I. Russell, editors, *Proceedings of the 12th Florida Artificial Intelligence Research Symposium*, pages 346–350. AAAI Press, 1999.

[20] G. Stenz and A. Wolf. E-SETHEO: An Automated Theorem Prover. In R. Dyckhoff, editor, *Proceedings of the International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX-2000)*, number 1847 in Lecture Notes in Artificial Intelligence, pages 436–440. Springer-Verlag, 2000.

[21] G. Sutcliffe. The CADE-16 ATP System Competition. *Journal of Automated Reasoning*, 24(3):371–396, 2000.

[22] G. Sutcliffe. The CADE-17 ATP System Competition. *Journal of Automated Reasoning*, 27(3):227–250, 2001.

[23] G. Sutcliffe. The IJCAR-2004 Automated Theorem Proving Competition. *AI Communications*, 18(1):33–40, 2005.

[24] G. Sutcliffe, M. Fuchs, and C. Suttner. Progress in Automated Theorem Proving, 1997-1999. In H. Hoos and T. Stützle, editors, *Proceedings of the IJCAI'01 Workshop on Empirical Methods in Artificial Intelligence*, pages 53–60, 2001.

[25] G. Sutcliffe and C. Suttner. The CADE-18 ATP System Competition. *Journal of Automated Reasoning*, 31(1):23–32, 2003.

[26] G. Sutcliffe and C. Suttner. The CADE-19 ATP System Competition. *AI Communications*, 17(3):103–182, 2004.

[27] G. Sutcliffe, C. Suttner, and F.J. Pelletier. The IJCAR ATP System Competition. *Journal of Automated Reasoning*, 28(3):307–320, 2002.

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

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

[30] G. Sutcliffe and C.B. Suttner. The CADE-15 ATP System Competition. *Journal of Automated Reasoning*, 23(1):1–23, 1999.

[31] G. Sutcliffe and C.B. Suttner. Evaluating General Purpose Automated Theorem Proving Systems. *Artificial Intelligence*, 131(1-2):39–54, 2001.

[32] C.B. Suttner and G. Sutcliffe. The CADE-14 ATP System Competition. *Journal of Automated Reasoning*, 21(1):99–134, 1998.

[33] T. Tammet. Gandalf. *Journal of Automated Reasoning*, 18(2):199–204, 1997.

[34] T. Tammet. Towards Efficient Subsumption. 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 427–440. Springer-Verlag, 1998.

[35] C. Weidenbach, U. Brahm, T. Hillenbrand, E. Keen, C. Theobald, and D. Topic. SPASS Version 2.0. In A. Voronkov, editor, *Proceedings of the 18th International Conference on Automated Deduction*, number 2392 in Lecture Notes in Artificial Intelligence, pages 275–279. Springer-Verlag, 2002.