The CADE-19 ATP System Competition

Geoff Sutcliffe [a] and 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 (ATP) systems. CASC-19 was the eighth competition in the CASC series. Twenty-five ATP system variants competed in the various competition and demonstration divisions. An outline of the design and a commentated summary of the results are presented.

Keywords: competition, automated theorem proving

1. Introduction

The CADE ATP System Competition (CASC) is an annual evaluation of fully automatic, first order Automated Theorem Proving (ATP) systems. In addition to the primary aim of evaluating the relative capabilities of ATP systems, CASC aims to stimulate ATP research in general, to stimulate ATP research towards autonomous systems, to motivate implementation and fixing of systems, to provide an inspiring environment for personal interaction between ATP researchers, and to expose ATP systems both 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. CASC-19 was held on 31st July 2003, as part of the 19th International Conference on Automated Deduction, in Miami, USA.

CASC-19 was the eighth competition in the CASC series - see [12] and citations therein. Twenty five ATP system variants, listed in Table 1, competed in the various competition and demonstration divisions. The division winners of CASC-18 were automatically entered to provide benchmarks against which progress can be judged. Details of the CASC-19 design, and system descriptions for the entered systems, are in [11] and on the CASC-19 WWW site:

`http://www.cs.miami.edu/~tptp/CASC/19/`

The WWW site also provides access to all systems and competition resources. CASC-19 was organized by Geoff Sutcliffe and Christian Suttner, and was overseen by a panel consisting of Uli Furbach, Don Loveland, and Jeff Pelletier. The competition machines were supplied by the University of Manchester.

This paper is organized as follows: Sections 2 and 3 describe the divisions and organization of CASC-19. Section 4 provides a commentated summary of the results, and short descriptions of the division winners are given in Section 5.

2. Divisions

CASC is run in divisions according to system and problem characteristics. In CASC-19 there were five *competition divisions*, in which the systems were ranked according to the numbers of problems solved, with ties decided by average CPU times over problems solved.

– The **MIX** division used mixed CNF really non-propositional theorems. *Mixed* means Horn and non-Horn problems, with or without equality, but not unit equality problems (see the UEQ division below). *Really non-propositional* means with an infinite Herbrand universe. The MIX division had five problem categories: **HNE** - Horn with No Equality, **HEQ** - Horn with some (not pure) Equality, **NNE** - Non-Horn with No Equality, **NEQ** - Non-Horn with some (not pure) Equality, and **PEQ** - Pure Equality. The MIX division had two ranking classes: the **assurance** class - ranked according to the number of problems solved (a "yes" output, giving an *assurance* of the

AI Communications
ISSN 0921-7126, IOS Press.

Table 1
The ATP systems and entrants

ATP System	Divisions	Entrants	Affiliation
CARINE 0.72	MIX*	Paul Haroun	McGill University
CiME 2.01	UEQ	Evelyne Contejean, Benjamin Monate	LRI, Universite Paris-Sud
DCTP 1.3	MIX	Gernot Stenz	Max-Planck-Institut für Informatik
DCTP 1.3-SAT	SAT		*DCTP 1.3 variant*
DCTP 1.3-EPR	EPR		*DCTP 1.3 variant*
DCTP 10.2p	MIX FOF EPR	Gernot Stenz	Max-Planck-Institut für Informatik
DCTP 10.2p-SAT	SAT		*DCTP 10.2p variant*
E 0.8	MIX EPR UEQ	Stephan Schulz	Technische Universität München
EP 0.8	MIX*		*E 0.8 variant*
E-SETHEO csp02	EPR		*CASC-18 EPR winner*
E-SETHEO csp03	MIX FOF EPR	Gernot Stenz, Reinhold Letz,	Max-Planck-Institut für Informatik
	UEQ	Stephan Schulz	Technische Universität München
E-S'O csp03-SAT	SAT		*E-SETHEO csp03 variant*
Gandalf c-2.5-SAT	SAT		*CASC-18 SAT winner*
Gandalf c-2.6	MIX EPR	Tanel Tammet	Tallinn Technical University
Gandalf c-2.6-PRF	MIX*		*Gandalf c-2.6 variant*
Gandalf c-2.6-SAT	SAT EPR		*Gandalf c-2.6 variant*
MUSCADET 2.4	FOF	Dominique Pastre	Université René Descartes
Octopus N	MIX (demo)	Monty Newborn, Zongyan Wang	McGill University
Otter 3.2	MIX* FOF UEQ	William McCune	Argonne National Laboratory
Paradox 1.0	SAT* EPR	Koen Claessen, Niklas Srensson	Chalmers University of Technology
THEO J2003	MIX*	Monty Newborn, Zongyan Wang	McGill University
Vampire 5.0	MIX* FOF		*CASC-18 MIX and FOF winner*
Vampire 6.0	MIX* FOF EPR	Andrei Voronkov, Alexandre Riazanov	The University of Manchester
	UEQ		
Waldmeister 702	UEQ		*CASC-18 UEQ winner*
Waldmeister 703	UEQ	Thomas Hillenbrand, Jean-Marie	Max-Planck-Institut für Informatik
		Gaillourdet, Bernd Lchner	Universität Kaiserslautern

MIX* indicates participation in the MIX division proof class - see Section 2.

existence of a proof), and the **proof** class - ranked according to the number of problems solved with an *acceptable proof output*. The competition panel judged whether or not each system's proof format is acceptable.

– The **FOF** division used non-propositional first-order form theorems. The FOF division had two problem categories: **FNE** - FOF with No Equality, and **FEQ** - FOF with Equality.
– The **SAT** division used CNF really non-propositional non-theorems. The SAT division had two problem categories: **SNE** - SAT with No Equality, and **SEQ** - SAT with Equality. The SAT division had two ranking classes: the **assurance** class - ranked according to the number of problems solved (a "yes" output, giving an *assurance* of the existence of a model), and the **model** class - ranked according to the number of problems solved with an *acceptable model output*. The competition panel judged whether or not each system's model format is acceptable.
– The **EPR** division used CNF effectively propositional theorems and non-theorems. *Effectively propositional* means syntactically non-propositional but with a finite Herbrand universe. The EPR division had two problem categories: **EPT** - Effectively Propositional Theorems (unsatisfiable clause sets), and **EPS** - Effectively Propositional non-theorems (Satisfiable clause sets).
– The **UEQ** division used unit equality CNF really non-propositional theorems.

Additionally, CASC has a *demonstration division*, in which systems demonstrate their abili-

ties without being formally ranked, using the same problems as in the competition divisions.

3. Organization

The CASC-19 competition divisions were run on 44 Dell Precision 330 workstations, each having an Intel P4 993 MHz CPU, 512 MB memory, and the Linux 2.4.9-34 operating system. In the demonstration division, Octopus ran on a network of 60 workstations, each having an Intel P2 or P3 CPU, between 128 MB and 256 MB memory, and either the FreeBSD or Linux operating system.

The problems were taken from the TPTP problem library [13], v2.6.0. TPTP v2.6.0 was not released until after the competition, so that new problems had not previously been seen by the entrants. Unbiased TPTP problems with a TPTP difficulty rating in the range 0.21 to 0.99 were eligible for use. The problems used were randomly selected from the eligible problems, based on a seed provided by the panel at the start of the competition. The random selection was subject to a limitation on the number of very similar problems in each division and category [10], and biased to ensure (if possible) the selection of at least 50% new problems in each division and category. In CASC-19, due to only a small growth of difficult problems in the TPTP since CASC-18, only the UEQ division had a significant number of new problems. Table 2 gives the numbers of eligible problems, the maximal numbers that could be used after taking into account the limitation on very similar problems, and the numbers of problems used, in each division and category. A change for CASC-19 was to take into account the maximal numbers of usable problems in each category, when deciding the numbers of problems to be used. As a result, in the MIX division the NEQ category had 60 problems while the other categories had 20. Due to the small maximal numbers of usable problems in the EPT and EPS categories, the limitation on the number of very similar problems could not be fully imposed there.

To ensure that no system received an advantage or disadvantage due to the specific presentation of the problems in the TPTP, the `tptp2X` utility was used to replace all predicate and function symbols with new symbols, randomly reorder the formulae and the clauses' literals, and randomly reverse the unit equalities in the UEQ problems.

The ATP systems were required to be sound and fully automatic. The organizers tested the systems for soundness by submitting non-theorems to the systems participating in the MIX, UEQ, FOF, and EPR divisions, and theorems to the systems participating in the SAT and EPR divisions. Claiming to have found a proof of a non-theorem or a disproof of a theorem indicates unsoundness. No system failed this test. Fully automatic operation meant that any command line switches had to be the same for all problems. A 600 second CPU time limit was imposed on each solution attempt. A wall clock time limit of double the CPU time limit was imposed in all divisions, to limit very high memory usage that causes swapping.

4. Results

For each ATP system, for each problem, three items of data were recorded: whether or not the problem was solved, the CPU time taken, and whether or not a solution (proof or model) was output. This section summarizes the results, and provides some commentary. Detailed results, including the systems' output files, are available from the CASC-19 WWW site. In each of the results summary tables below, the CASC-18 winner is highlighted in *italics*.

4.1. The MIX Division

Tables 3 and 4 summarize the results in the MIX division. As Vampire outputs proofs, Vampire was the winner of both the Assurance and Proof classes. The improved performance of the top three systems over Vampire 5.0, the CASC-18 winner, indicates progress in the area. The improved performance of Vampire 6.0 over Vampire 5.0 is due to improved indexes and constraint checking procedures, as well as a better selection of strategies (see Section 5). The average solution time of Vampire 6.0 is higher than that of Vampire 5.0 because the strategy scheduling (see below) in Vampire 6.0 was optimized only for solving as many problems as possible, and was not optimized for time taken.

Of the top eight systems, only E and EP (EP is a pipeline of E and a postprocessing program that

Table 2
Numbers of eligible and used problems

Division	MIX					FOF		SAT		EPR		UEQ
Category	HNE	HEQ	NNE	NEQ	PEQ	FNE	FEQ	SNE	SEQ	EPT	EPS	
Eligible	186	114	71	623	241	73	466	149	129	58	111	138
Max usable	51	63	65	498	121	18	429	149	129	7	12	138
Max new	2	0	0	0	0	0	4	2	0	0	0	33
Used	20	20	20	60	20	15	55	35	35	35	35	70
New used	2	0	0	0	0	0	4	2	0	0	0	33

generates proofs) are monolithic, i.e., select a single strategy for each problem. The others all employ *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). The monolithic nature of E is a major reason for E's lowest average time taken among the top eight systems. Note that for three problems, the postprocessing in EP could not be completed within the time limit, even though it had been determined that a proof exists. Gandalf c-2.6-PRF was also unable to solve as many problems as the plain variant that does not always produce a proof, confirming the overhead of proof production for these systems. (The plain variant of Gandalf c-2.6 has the same inference engine as the PRF variant; the only difference is that it does not maintain the data structures required for complete proof production. This similarity does not exist between distinct versions of systems, e.g., between Vampires 5.0 and 6.0.)

The rankings in the categories align quite closely with the division ranking, with the exception of E and EP in the Horn categories. E and EP are relatively weaker on HNE problems and stronger on HEQ problems. E's stronger performance in the HEQ category comes from the combination of its strength in handling unit-equational theories and good literal selection heuristics. The other systems are reasonably unspecialized, which is expected when strategy scheduling is used.

The individual problem results show that three problems were solved by all the systems, and four problems were unsolved (they were eligible because they had been solved by systems that were not entered). These low numbers of undifferentiating problems indicate that appropriately difficult problems were eligible for the division. Three problems were solved by (essentially) only one system: `GRP198-1` (a PEQ problem) was solved by only both Gandalf variants, and `NUM007-1` and `SET014-3` (both NEQ problems) were solved by only THEO. The latter uniqueness is interesting in the light of THEO's weak overall performance, and indicates some unique capabilities in that system.

Table 3
MIX division results

ATP System	MIX /140	Average time	Prfs out
Vampire 6.0	120	65.6	120
E-SETHEO csp03	119	34.7	0
E 0.8	113	20.9	0
Vampire 5.0	113	23.1	113
EP 0.8	113	25.4	110
Gandalf c-2.6	102	67.8	0
Gandalf c-2.6-PRF	79	30.9	79
DCTP 10.2p	72	43.3	0
DCTP 1.3	55	18.2	0
THEO J2003	49	45.6	47
Otter 3.2	34	99.1	34
CARINE 0.72	7	104.9	7
Demonstration division			
Octopus N	59	-	-

None of the systems solved more than a very few problems close to the 600s time limit, and the ranking would have been the same for any time limit from 250s to 600s. This indicates that the 600s time limit was sufficient.

4.2. The FOF Division

Table 5 summarizes the results in the FOF division. All the systems except MUSCADET work by converting to CNF and producing a refutation. The winner, Vampire 5.0, was the winner of the FOF division in CASC-18. The slightly weaker performance of the new Vampire 6.0 is believed to be an artifact of the problem selec-

Table 4
MIX category results

ATP System	HNE /20	HEQ /20	NNE /20	NEQ /60	PEQ /20
Vampire 6.0	18	14	18	54	16
E-S'O csp03	17	17	15	53	17
E 0.8	14	18	15	49	17
Vampire 5.0	18	11	18	51	15
EP 0.8	14	18	15	49	17
Gandalf c-2.6	18	10	13	48	13
G'lf c-2.6-PRF	12	10	8	37	12
DCTP 10.2p	15	2	13	36	6
DCTP 1.3	9	2	9	29	6
THEO J2003	9	0	7	32	1
Otter 3.2	8	2	3	17	4
CARINE 0.72	0	0	0	7	0
Demonstration division					
Octopus N	10	2	8	37	2

tion for the division combined with inappropriate strategy scheduling for the selected problems. The difference in performance between the FNE and FEQ categories for Otter and MUSCADET are noticeable - both performed significantly better in the FEQ category. The individual problem results show that six FEQ problems were solved by only MUSCADET and one was solved by only Otter, indicating some unique capabilities in these generally weaker systems. Two further FEQ problems, `SET169+3` and `SET171+3`, were solved by only Vampire and MUSCADET, with Vampire requiring 400s and 530s to solve them respectively, while MUSCADET solved them in no time.

Table 5
FOF division and category results

ATP System	FOF /70	Avg time	Prfs out	FNE /15	FEQ /55
Vampire 5.0	57	15.2	57	14	43
Vampire 6.0	56	56.5	56	14	42
E-S'O csp03	48	66.1	0	11	37
DCTP 10.2p	42	74.0	0	14	28
Otter 3.2	14	72.0	14	1	13
M'DET 2.4	13	0.5	0	0	13

4.3. The SAT Division

Table 6 summarizes the results in the SAT division. The performances of Gandalf c-2.6-SAT and Paradox are significantly better than that of the CASC-18 winner Gandalf c-2.5-SAT, indicating progress in the area. Gandalf c-2.6-SAT benefited (relative to Gandalf c-2.5-SAT) from improved handling of problems with more than 128 predicate and function symbols, an additional literal ordering for resolution, improvements to the MACE-style strategy, and an ability to interrupt some complex processing within the Falcon/SEM-style strategy.

A highlight of the SAT division was the strong performance of the new system, Paradox. While Gandalf gave more assurances of satisfiability (and thus won the Assurance class), Gandalf failed to output models for 23 of its 63 solved problems. Paradox output models for all 60 of its solved problems, and thus won the Model class. The significantly lower average solution time of Paradox is also noteworthy. Paradox is a monolithic system, while all the other systems in the SAT division, except DCTP 1.3, employ strategy scheduling. Gandalf and Paradox solved the same number of problems in the SNE category, and it was Gandalf's better performance in the SEQ division that gave it the overall edge.

The individual problem results show that eight problems were solved by all the systems. Five problems, all SEQ, were solved by only E-SETHEO, and were all solved by the "e-iterator" within E-SETHEO's strategy scheduling. The "e-iterator" sequentially alternates between a model elimination system and E, with a carefully filtered exchange of unit lemmas at each switch.

Table 6
SAT division and category results

ATP System	SAT /70	Avg time	Mdls out	SNE /35	SEQ /35
G'lf c-2.6-SAT	63	142.2	40	34	20
Paradox 1.0	60	6.5	60	34	26
G'lf c-2.5-SAT	49	93.9	0	27	22
E-S'O csp03-SAT	31	14.2	0	20	11
DCTP 10.2p-SAT	22	28.0	0	17	5
DCTP 1.2-SAT	19	2.5	0	14	5

4.4. The EPR Division

Table 7 summarizes the results in the EPR division. DCTP performed well on both EPT and EPS problems, in contrast to other systems that performed well in one category but not the other,

e.g., Vampire performed well in only EPT and Paradox performed well in only EPS. DCTP is also used as a component of the strategy scheduling E-SETHEO, just as last year's DCTP 1.2-SAT was used as a component of the CASC-18 EPR winner, E-SETHEO csp02. The natural expectation (and the developers' expectations!) that E-SETHEO would again outperform the component system was invalidated because of DCTP's better individual performance in the EPS category. The single strategy used by DCTP was appropriate for the large `SYN8XX` problems in the EPS category, while the DCTP and other strategies used within E-SETHEO were not allocated enough time in the strategy schedule to solve those problems.

The individual problem results show that 18 EPT problems and 4 EPS problems were solved by all the systems. This indicates that some of the eligible problems were too easy for comparing the systems in the competition. An interesting feature of Gandalf's performance is that 37 problems were solved in exactly 259s. This is an artifact of Gandalf's strategy scheduling - for most of the 37 problems Gandalf switched at 259s to a ground-and-decide style strategy that converts the problem to an equiconsistent propositional problem and runs ZChaff [6] on it. This turns out to be an appropriate strategy for many of the problems selected for the EPR division.

Table 7
EPR division and category results

ATP System	EPR /70	Avg time	Ps/Ms output	EPT /35	EPS /35
DCTP 1.3-EPR	66	95.8	0/0	32	34
G'lf c-2.6-SAT	61	248.1	0/0	33	28
E-S'O csp02	57	25.0	0/0	31	26
E-S'O csp03	57	71.2	0/0	31	26
DCTP 10.2p	55	47.4	0/0	30	25
Paradox 1.0	48	60.7	0/26	22	26
Vampire 6.0	47	49.7	32/0	32	15
E 0.8	47	80.7	0/0	30	17

4.5. The UEQ Division

Table 8 summarizes the results in the UEQ division. The winner, Waldmeister 702, was the winner of the UEQ division in CASC-18. The marginally lower average time taken by the older system is insignificant. The main development in Waldmeister over the past year has been to implement the "new Waldmeister loop" [2], which dramatically reduces Waldmeister's memory usage. This allows Waldmeister to solve harder problems that require longer searches, but does not improve performance on the competition level problems.

A noteworthy feature of the UEQ division is that, in contrast to the UEQ division in recent years, Waldmeister did not completely dominate. There is a relatively gentle drop in the number of problems solved down the ranking. This is due mainly to Waldmeister's relatively weak performance on the 33 new problems in the UEQ division – on the new problems it was outperformed by three other systems. In particular, Waldmeister was unable to solve 14 new lattice theory problems, as it was unable to recognize the algebraic structure (a key feature of Waldmeister's search parameter selection scheme - see Section 5), and hence used non-specialized search parameters for these problems.

Table 8
UEQ division results

ATP System	UEQ /70	Avg time	Prfs out	New /33
Waldmeister 702	56	6.9	56	17
Waldmeister 703	56	7.0	56	19
E 0.8	53	26.4	0	28
E-SETHEO csp03	52	45.7	0	25
Vampire 6.0	48	52.6	48	31
Gandalf c-2.6	45	73.3	0	24
CiME 2.01	21	93.9	0	12
Otter 3.2	11	44.1	11	4

5. Descriptions of the Winning Systems

Vampire 6.0 [7], the MIX and FOF divisions winner, is an automatic prover for first order classical logic. Its kernel implements ordered binary resolution and superposition with several standard simplification techniques. A number of efficient indexing techniques are used to implement all search related operations on sets of terms and clauses. The saturation process can be controlled by the *limited resource strategy*, which aims to maximize the effectiveness of the proof search in the presence of a time limit. The kernel works only with the clausal normal form, but the preprocessor com-

ponent accepts a problem in full first order form, clausifies it, and calls the kernel on the clauses. The kernel provides a fairly large number of features for specifying strategies, including choice of the saturation procedure (several variants of the given-clause algorithm), a variety of optional simplifications, parameterized reduction orderings (not used in CASC-19), and a number of built-in (parameterized) literal selection functions. Strategy scheduling is used to try several kernel strategies on each given problem. The main changes since Vampire 5.0 are: a new mode of query answering for large knowledge bases, the possibility of adding evaluated functions, availability of answer literals, output of first order proofs, set-of-support, and built-in special treatment of transitive relations. Most of these changes were driven by applications and are not CASC-oriented. Changes useful for CASC (such as built-in transitivity) have not been properly tested, and were not used in CASC. The system is implemented in C++. Further information may be obtained from:

`http://www.cs.man.ac.uk/~riazanoa/Vampire`

Gandalf c-2.6-SAT, the SAT division Assurance class winner, is a member of the Gandalf family of automated theorem provers [14], which includes systems for classical logic, type theory, intuitionistic logic, and linear logic. Gandalf c-2.6 contains the classical logic prover and a finite model builder, for clause form input. One of the basic ideas used in Gandalf is strategy scheduling. During each run Gandalf typically modifies its strategy as it approaches the time limit for the run. Additionally, selected clauses from unsuccessful strategies are sometimes used in later strategies. The following strategies are used for satisfiability checking: finite model building by incremental search through function and predicate symbol interpretations, ordered (by term depth) binary resolution for problems not containing equality, and finite model building using MACE-style flattening [5]. Gandalf is implemented in Scheme and compiled to C using the Hobbit Scheme-to-C compiler. The finite model building uses the Zchaff propositional logic solver [6] as an external program for the MACE-style strategy. Gandalf is available at:

`http://www.ttu.ee/it/gandalf`

Paradox 1.0 [1], the SAT division Model class winner, is a finite-domain model generator that produces human readable models. Paradox is based on a MACE-style [4] flattening and instantiation of the first order clauses into propositional clauses that encode the existence of a model of a fixed size. These propositional problems are generated for increasing domain sizes and given to a SAT solver. In some cases, most notably when there are no functions in the problem, an upper bound on the size of a model can be determined. This allows Paradox to deduce that the problem is contradictory when no models up to this size bound are found. The following novel features are included in Paradox: New polynomial-time clause splitting heuristics, the use of incremental SAT, static symmetry reduction techniques, and the use of sort inference. The main part of Paradox is implemented in Haskell using the GHC compiler. Paradox also has a built-in incremental SAT solver which is written in C++. The two parts are linked together on the object level using Haskell's Foreign Function Interface.

DCTP 1.3 [8] is an automated theorem prover for first order clause logic. It is an implementation of the disconnection calculus described in [9]. The disconnection calculus is an instantiation based, proof confluent and inherently cut-free tableau calculus with a weak connectedness condition. The inherently depth-first proof search is guided by a literal selection based on literal instantiatedness or literal complexity, and a heavily parameterized link selection. The pruning mechanisms mostly rely on different forms of variant deletion and unit based strategies. Additionally, the calculus has been augmented by full tableau pruning. The new DCTP 1.3 has been enhanced with respect to clause preprocessing, selection functions and closure heuristics. Most prominent among the improvements are the introduction of a unification index for finding connections, which also replaces the connection graph hitherto used, and the introduction of an enhanced algorithm for deterministically resolving isolated connections in the input set. As the disconnection calculus provides a decision procedure for the Bernays-Schönfinkel class of problems, DCTP-1.3 had been expected to perform well in the EPR category. Moreover, the enhancements described above specifically improved DCTP's performance in this class while keeping the proof procedure sufficiently generic. DCTP is available at:

`http://www.mpi-sb.mpg.de/~stenz/dctp-sb.html`

Waldmeister 703 is a system for unit equational deduction. Its theoretical basis is unfailing comple-

tion with refinements toward ordered completion. See [3] for a recent in-depth description. Since last year's competition, the "new WALDMEISTER loop" has been implemented, and is now operational [2]. This notion captures a novel organization of the saturation-based proof procedure into a system architecture, featuring a highly compact representation of the search state which exploits its inherent structure. With this architecture one can now solve problems that previously were out of reach. For example, two new minimal-length single axioms have been found for Boolean algebra in terms of the Sheffer stroke. The focus of recent developments has been more on coping with large search states. Therefore it is not astonishing that, within the competition setting, this year's system version is roughly equivalent to last year's. The performance on the unseen problems indicates that the general-purpose strategy for unknown algebraic structures should be fanned out into a multitude of strategies. The new search-state representation renders possible a novel realization of time-slicing via true multiplexing. Finally, visit the thoroughly rewritten Web pages at:

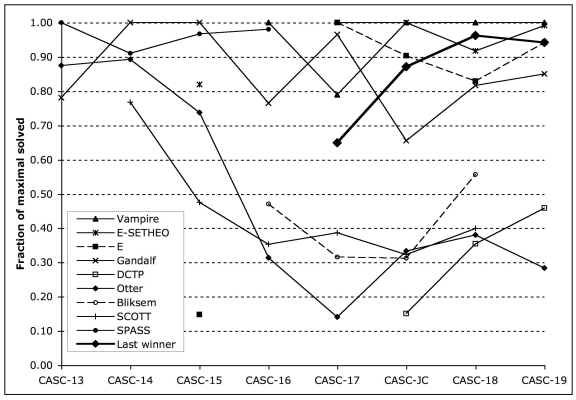
`http://www.mpi-sb.mpg.de/ hillen/waldmeister`

6. Conclusion

The CADE-19 ATP System Competition was the eighth large scale competition for first order ATP systems. Improved performances (relative to the CASC-18 winners) in the MIX, SAT, and EPR divisions were complemented by strong performances by the runners up in the MIX, SAT, and UEQ divisions. For the first time since CASC-16, the UEQ division was not overwhelmed by Waldmeister. At the same time, the divergence between the top systems and the "also-rans" in the MIX divisions remains clear. Figure 1 illustrates the history of this divergence since CASC-13. The graph plots the fractions of the maximal number of problems solved in the MIX division, for each CASC, for systems that have been entered into several CASCs. The top group includes SPASS (when it was entered into CASC), E, E-SETHEO, Gandalf, and Vampire.

CASC-18 fulfilled its main objectives, by evaluating the relative abilities of current ATP systems, and stimulating development of and interest in ATP systems. The competition highlighted areas of ATP where progress was (and possibly was not) made in the last year, and through the continuity of the event the results allow performance comparisons with previous and future years. The competition provided exposure for system builders both within and outside of the community, and provided an overview of the implementation state of running, fully automatic, first order ATP systems.

Fig. 1. History of MIX division performances



References

[1] 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.

[2] J-M. Gaillourdet, T. Hillenbrand, B. Löchner, and H. Spies. The New Waldmeister Loop at Work. In F. Baader, editor, *Proceedings of the 19th International Conference on Automated Deduction*, number 2741 in Lecture Notes in Artificial Intelligence, pages 317–321. Springer-Verlag, 2003.

[3] 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.

[4] W.W. McCune. A Davis-Putnam Program and its Application to Finite First-Order Model Search: Quasigroup Existence Problems. Technical Report ANL/MCS-TM-194, Argonne National Laboratory, Argonne, USA, 1994.

[5] W.W. McCune. MACE 2.0 Reference Manual and Guide. Technical Report ANL/MCS-TM-249, Argonne National Laboratory, Argonne, USA, 2001.

[6] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff: Engineering an Efficient SAT Solver. In Blaauw D. and L. Lavagno, editors, *Proceedings of the 39th Design Automation Conference*, pages 530–535, 2001.

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

[8] G. Stenz. DCTP 1.2 - System Abstract. In C. Fermüller and U. Egly, editors, *Proceedings of TABLEAUX 2002: Automated Reasoning with Analytic Tableaux and Related Methods*, number 2381 in Lecture Notes in Artificial Intelligence, pages 335–340. Springer-Verlag, 2002.

[9] G. Stenz. *The Disconnection Calculus*. Logos Verlag, 2002. Dissertation, Fakultät für Informatik, Technische Universität München.

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

[11] G. Sutcliffe. Proceedings of the CADE-19 ATP System Competition. Miami, USA, 2003.

[12] G. Sutcliffe and C. Suttner. The CADE-18 ATP System Competition. *Journal of Automated Reasoning*, page To appear, 2003.

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

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