The CADE-25 Automated Theorem Proving System Competition – CASC-25

Geoff Sutcliffe[a] and Josef Urban[b]
[a] *Department of Computer Science University of Miami, USA*
geoff@cs.miami.edu
[b] *Intelligent Systems ICIS Radboud Universiteit Nijmegen, The Netherlands*

The CADE ATP System Competition (CASC) is an annual evaluation of fully automatic, classical logic Automated Theorem Proving (ATP) systems. CASC-25 was the twentieth competition in the CASC series. Twenty-seven ATP systems and system variants competed in the various competition divisions. An outline of the competition design, and a commentated summary of the results, are presented.

Keywords: automated theorem proving, competition

1. Introduction

The CADE ATP System Competition (CASC) is the annual evaluation of fully automatic, classical logic Automated Theorem Proving (ATP) systems – the world championship for such systems. One purpose of CASC is to provide a public evaluation of the relative capabilities of ATP systems. Additionally, CASC aims to stimulate ATP research, to motivate development and implementation of robust ATP systems that are useful and easily deployed in applications, 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 insight and stimulus for the development of more powerful ATP systems, leading to increased and more effective use. CASC-25 was held on 4th August 2015 in Berlin, Germany, as part of the 25th International Conference on Automated Deduction (CADE-25). CASC-25 was the twentieth competition in the CASC series; see [14] and citations therein for information about individual previous competitions. The CASC-25 web site provides access to all competition resources: `http://www.tptp.org/CASC/25`.

CASC is divided into divisions according to problem and system characteristics. There are competition divisions in which the systems are explicitly ranked, and a demonstration division in which systems demonstrate their abilities without being ranked. (The demonstration division is for systems that cannot be entered into the competition divisions for any reason, e.g., the system runs on specialist hardware, or the entrant is a competition organizer or panel member. In CASC-25 there were no systems in the demonstration division.) Each competition division uses problems that have certain logical, language, and syntactic characteristics, so that the 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 that make it possible to analyze, at a more fine-grained level, which systems work well for what types of problems. The demonstration division uses the same problems as the competition divisions – the entry specifies which competition divisions' problems are to be used. Table 1 catalogs the divisions and problem categories of CASC-25. The example problems can be viewed online at `http://www.tptp.org/cgi-bin/SeeTPTP?Category=Problems`.

Twenty-seven ATP systems and system variants, listed in Table 2, competed in the various competition divisions. The division winners of CASC-J7 (the previous CASC) were automatically entered to provide benchmarks against which progress can be judged. (As the LTB division had been suspended since CASC-24 in 2013, the CASC-24 LTB winner was entered.) Additionally, Prover9 2009-11A is entered every year as a fixed point against which progress can be judged (this is the fourth year that Prover9 has been used as the fixed point - in prior years Otter was used). System descriptions of the entered systems are in [13] and on the CASC-25 web site.

Division	Problems	Problem Categories
THF	**T**yped **H**igher-order **F**orm theorems (axioms with a provable conjecture).	**TNE** – **T**HF with **N**o **E**quality, e.g., `NUM738^1`. **TEQ** – **T**HF with **EQ**uality, e.g., `SET171^3`.
THN	**T**yped **H**igher-order form **N**on-theorems (axioms with a countersatisfiable conjecture, and satisfiable axiom sets).	**TNN** – **THN** with **N**o equality, e.g., `CSR142^3`. **TNQ** – **THN** with e**Q**uality, e.g., `PHI003^3`.
TFA	**T**yped **F**irst-order form theorems with **A**rithmetic (axioms with a provable conjecture).	**TFI** – **TF**A with only **I**nteger arithmetic, e.g., `DAT016=1`. **TFR** – **TF**A with only **R**ational arithmetic, e.g., `ARI621=2`. **TFE** – **TF**A with only r**E**al arithmetic, e.g., `MSC022=2`.
TFN	**T**yped **F**irst-order form **N**on-theorems with arithmetic (axioms with a countersatisfiable conjecture, and satisfiable axiom sets).	**TIN** – **TFN** with only **I**nteger arithmetic, e.g., `ARI127=1`. **TRN** – **T**NN with only **R**ational arithmetic, e.g., `ARI575=2`. **TEN** – **TFN** with only r**E**al arithmetic, e.g., `ARI575=3`.
FOF	**F**irst-**O**rder **F**orm theorems (axioms with a provable conjecture).	**FNE** – **F**OF with **N**o **E**quality, e.g., `COM003+1`. **FEQ** – **F**OF with **EQ**uality, e.g., `SEU147+3`.
FNT	**F**OF **N**on-**T**heorems (axioms with a countersatisfiable conjecture, and satisfiable axioms sets).	**FNN** – **FNT** with **N**o equality, e.g., `KRS173+1`. **FNQ** – **FNT** with e**Q**uality, e.g., `MGT033+2`.
EPR	**E**ffectively **PR**opositional theorems and non-theorems in clause normal form (unsatisfiable and satisfiable clause sets). *Effectively propositional* means that the problems are known to be reducible to propositional problems, e.g., CNF problems that have no functions with arity greater than zero.	**EPT** – **E**ffectively **P**ropositional **T**heorems (unsatisfiable clause sets), e.g., `PUZ037-3`. **EPS** – **E**ffectively **P**ropositional non-theorems (**S**atisfiable clause sets), e.g., `GRP123-2.005`.
LTB	First-order form theorems (axioms with a provable conjecture) from **L**arge **T**heories, presented in **B**atches. A *large theory* has many functors and predicates, and many axioms of which typically only a few are required for the proof of a theorem. Problems in a *batch* all use a common core set of axioms, and the problems in a batch are given to the ATP system all at once.	**HLL** – Problems exported from **HOL L**ight. **HL4** – Problems exported from **HOL 4**. **ISA** – Problems exported from **ISA**belle. **MZR** – Problems exported from the **Miz**a**r** Mathematical Library. (See Section 2.2 for details of these problems' sources.)

Table 1
Divisions and Problem categories

CASC-25 was organized by Geoff Sutcliffe, assisted by Josef Urban (who was responsible for the LTB division), and overseen by a panel consisting of Pascal Fontaine, Aart Middeldorp, and Neil Murray. The competition was run on computers provided by the StarExec project [9] at the University of Iowa, and by the Department of Computer Science at the University of Manchester.

This paper is organized as follows: Section 2 outlines the design and organization of CASC-25. Section 3 provides a commentated summary of the results. Section 4 contains short descriptions of three of the ATP systems. Section 5 concludes and discusses plans for future CASCs.

2. Outline of Design and Organization

The design and organization of CASC has evolved over the years to a sophisticated state. An outline of the CASC-25 design and organization is provided here. The details are in [13] and on the CASC-25 web site. Important changes for CASC-25 were:

- The Typed Higher-order form Non-theorems (THN) and Typed First-order form Non-theorems (TFN) divisions were added.
- The Large Theory Batch (LTB) division returned from its one year hiatus.
- The Unit Equality (UEQ) division returned to its hiatus state.

ATP System	Divisions	Entrant (Associates)	Entrant's Affiliation
Beagle 0.9.22	TFA TFN	Peter Baumgartner (Joshua Bax)	NICTA and ANU
CVC4 1.4	TFA	CASC	CASC-J7 TFA winner
CVC4 1.5	TFA TFN FOF FNT	Andrew Reynolds (Clark Barrett, Cesare Tinelli, Tim King)	EPFL
E 1.9.1	FOF FNT EPR LTB	Stephan Schulz	DHBW Stuttgart
ePrincess 1.0	FOF	Peter Backeman (Philipp Rümmer)	Uppsala University
ET 0.2	FOF	Josef Urban	Radboud University Nijmegen
Geo-III 2015E	FOF FNT EPR	Hans de Nivelle	University of Wrocław
iProver 0.9	EPR	CASC	CASC-J7 EPR winner
iProver 1.0	FNT	CASC	CASC-J7 FNT winner
iProver 2.0	FOF FNT EPR LTB	Konstantin Korovin	University of Manchester
iProverModulo 0.7-0.3	FOF	Guillaume Burel	ENSIIE/Cedric/Deducteam
Isabelle 2015	THF	Jasmin Blanchette (Lawrence Paulson, Tobias Nipkow, Makarius Wenzel)	Inria Nancy
leanCoP 2.2	FOF	Jens Otten	University of Potsdam
LEO-II 1.6.2	THF	Christoph Benzmüller	Freie Universität Berlin
MaLARea 0.5	LTB	CASC	CASC-24 LTB winner
Muscadet 4.5	FOF	Dominique Pastre	University Paris Descartes
Nitpick 2015	THN	Jasmin Blanchette	Inria Nancy
Princess 150706	TFA TFN	Philipp Rümmer (Peter Backeman)	Uppsala University
Prover9 2009-11A	FOF	CASC (William McCune, Bob Veroff)	CASC fixed point
Refute 2015	THN	Jasmin Blanchette (Tjark Weber)	Inria Nancy
Satallax 2.8	THF THN	Nik Sultana (Chad Brown)	Cambridge University
Satallax-MaLeS 1.3	THF	CASC	CASC-J7 THF winner
SPASS+T 2.2.22	TFA	Uwe Waldmann	Max-Planck-Institut für Informatik
Vampire 2.6	FOF	CASC	CASC-J7 FOF winner
Vampire 4.0	TFA FOF FNT EPR LTB	Giles Reger (Andrei Voronkov, Martin Suda, Laura Kovacs)	University of Manchester
VampireZ3 1.0	TFA	Giles Reger (Andrei Voronkov, Martin Suda)	University of Manchester
ZenonArith 0.1.0	TFA	Guillaume Bury (David Delahaye)	Inria

Table 2: The ATP systems and entrants

2.1. System Delivery, Execution, and Evaluation

The non-LTB divisions' systems were delivered to the competition organizer as StarExec installation packages, which the organizer installed and tested on StarExec. Source code was delivered separately for archiving on the competition web site. The LTB division's systems were delivered to the competition coorganizer as source code installation packages, which the coorganizer installed and tested on the Manchester cluster. (The Manchester cluster was used for the LTB division because the StarExec execution model does not support the batch mode of the LTB division.)

The ATP systems entered into CASC are required to be sound and fully automatic. They are executed as black boxes, on one problem (non-LTB) or one problem batch (LTB) at a time. Any command line parameters have to be the same for all problems/batches in each division. The systems are tested for soundness by submitting non-theorems to the systems in the THF, TFA, FOF, EPR, and LTB divisions, and theorems to the systems in the THN, TFN, FNT, and EPR divisions. Claiming to have found a proof of a non-theorem or a disproof of a theorem indicates unsoundness. For the second year in a row, no systems failed this soundness testing. Note that the soundness testing for CASC-25 was more extensive than in previous years, using 50 problems in each problem category, thanks to the available computing power in StarExec.

The THF, TFA, FOF, FNT, and LTB divisions were ranked according to the number of problems solved with an acceptable proof/model output. (The models are "counter-models" for problems with an unprovable conjecture, and "models" for satisfiable axiom sets.) The THN, TFN, and EPR divisions were ranked according to the number of problems solved, but not necessarily accompanied by a proof or model (but systems that do output proofs/models are highlighted in the presentation of results). Ties were broken according to the average time taken over problems solved (CPU time for non-LTB divisions, wall clock time for the LTB division). Trophies were awarded to the division winners.

In addition to the ranking criteria, three other measures were made and are presented in the results: The *state-of-the-art (SoTA) contribution* quantifies the unique abilities of each system. For each problem solved by a system, its SoTA contribution for the problem is the reciprocal of the number of systems that solved the problem, so that if a system is the only one to solve a problem then its SoTA contribution for the problem is 1.00, and if all the systems solve a problem their SoTA contribution for the problem is the inverse of the number of systems. A system's overall SoTA contribution is its average SoTA contribution over the problems it solved. The *efficiency measure* is a combined measure that balances the time taken for each problem solved against the number of problems solved. It is the average solution rate over the problems solved (the solution rate for one problem is the inverse of the time taken to solve it), multiplied by the fraction of problems solved. In the LTB division, which imposes a wall clock time limit (see Section 2.3), the *core usage* measures the extent to which the systems take advantage of the multiple cores. It is the average ratio of CPU time to wall clock time taken, over the problems solved. The LTB division ran on quad-core computers, thus the maximal core usage was 4.0.

2.2. Problems

The problems for the non-LTB divisions were taken from the Thousands of Problems for Theorem Provers (TPTP) problem library [11], v6.2.0. The TPTP version used for CASC is not released until after the competition has started, so that new problems have not been seen by the entrants. The problems have to meet certain criteria to be eligible for selection, as listed below. The problems used are randomly selected from the eligible problems based on a seed supplied by the competition panel.

- The TPTP tags problems that are designed specifically to be suited or ill-suited to some ATP system, calculus, or control strategy as *biased*, and they are not eligible.
- The problems have to be syntactically non-propositional to be eligible. (There are very few propositional problems in the TPTP anyway.)
- Each TPTP problem has a difficulty rating [15] that is based on ATP system performance data in the Thousands of Solutions from Theorem Provers (TSTP) solution library [12]. The ratings range from 0.00 (easy) to 1.00 (unsolved). Difficult problems with a rating in

the range 0.21 to 0.99 are eligible. Problems of lesser and greater ratings might also be eligible in some divisions if there are not enough problems with ratings in that range. Developers can submit their systems in advance of CASC (by a published deadline) so that the systems' performance data is used when computing the difficulty ratings.

– The selection is constrained so that no division or category contains an excessive number of very similar problems [10].
– The selection is biased to select problems that are new in the TPTP version used, until 50% of the problems in each problem category have been selected, after which random selection (from old and new problems) continues. The number of new problems used depends on how many new problems are eligible and the limitation on very similar problems.

The numbers of problems used in each division and problem category are roughly proportional to the numbers of eligible problems, subject to the limitation on the number of very similar problems. The problems are given to the ATP systems in TPTP format, in increasing order of TPTP difficulty rating.

The problems for the LTB division were taken from publicly available problem sets: the HLL problem category used the HH7150 problem set[1]; the HL4 problem category used the H4H13897 problem set[2]; the ISA problem category used the SH5795 problem set[3]; the MZR problem category used the MPTP2078 problem set[4]. The problems in each category have consistent symbol usage, and almost consistent axiom naming, between problems. In order to facilitate and promote learning from previous proofs, each problem category was accompanied by a set of 1000 training problems and their solutions, which could be used for tuning and training during (typically at the start of) the competition. Entrants were expected to honestly not use any other of the (publicly available) problems for tuning or training before the competition. The problems had to meet certain criteria to be eligible for selection, as listed below. The problems used were randomly selected from the eligible problems based on a seed supplied by the competition panel.

– Problems that are solvable in 60s by at least one of the non-LTB versions of the systems were used for the training sets. One thousand such problems were selected at random for the training set in each problem category. Problems that were used in training sets were not eligible.
– Problems that are solvable in 60s by at least one of the non-LTB versions of the systems were eligible for selection. 70% of the selected problems were from this group.
– Problems that are not solvable in 60s by any of the non-LTB versions of the systems were eligible for selection. 30% of the selected problems were from this group.

The problems are given to the ATP systems in TPTP format, in the natural order of their export from the underlying library. (This models the work of LTB-like systems when used to assist in formal proof development).

Table 3 gives the numbers of eligible problems, the maximal numbers that could be used after taking into account the limitation on very similar problems (in the context of the numbers of problems used), and the numbers of problems used, in each division and category. There were very few new problems in CASC-25, due to limited growth of the TPTP in the preceding year. In the TFR problem category there were very few usable problems, so an excess of very similar problems was selected.

2.3. Resources

The non-LTB divisions' computers had four (a quad-core chip) Intel(R) Xeon(R) E5-2609, 2.40GHz CPUs, 256GB memory, and ran the Red Hat Enterprise Linux Workstation release 6.3 (Santiago) operating system, kernel 2.6.32-431.1.2.el6.x86_64. A 300s CPU time limit was imposed for each system on each problem. A wall clock time limit of 600s was also imposed to limit very high memory usage that causes swapping.

The LTB division's computers had four (a quad-core chip) Intel(R) Xeon(R) L5410, 2.333GHz CPUs, 12GB memory, and ran the Linux 2.6.29.4-167.fc11.x86_64 operating system. A 60s wall clock time limit was imposed for each system on each problem, and an overall wall clock time limit of 24000s (400 problems times 60s) was imposed on each problem category. The use of a wall clock time limit encourages the use of the multiple cores that are available.

[1]`https://github.com/JUrban/HH7150`
[2]`https://github.com/JUrban/H4H13897`
[3]`http://mws.cs.ru.nl/~urban/isaltb/SH5795.tar.gz`
[4]`https://github.com/JUrban/MPTP2078`

Division	THF		THN		TFA			TFN			FOF		FNT		EPR		LTB			
Category	TNE	TEQ	TNN	TNQ	TFI	TFR	TFE	TIN	TRN	TEN	FNE	FEQ	FNN	FNQ	EPT	EPS	HLL	HL4	ISA	MZR
Eligible	124	595	25	228	207	42	16	19	3	2	373	3456	114	92	174	91	7150	13897	5795	2078
Usable	124	595	25	228	207	19	15	19	3	2	373	3456	114	92	174	24	-	-	-	-
New	0	7	0	5	5	0	0	0	0	0	1	0	2	0	0	0	-	-	-	-
Used	100	300	25	175	150	35	15	15	3	2	150	250	110	90	125	75	400	400	400	400
New	0	7	0	5	5	0	0	0	0	0	1	0	2	0	0	0	-	-	-	-

Table 3
Numbers of eligible and used problems

3. Results

For each ATP system, for each problem, four items of data were recorded: whether or not the problem was solved, the CPU time taken, the wall clock time taken, and whether or not a proof or model was output. This section summarizes the results, and provides commentary. The result tables below give the number of problems solved in the division, the average time over problems solved, whether or not proofs or models were output, the state-of-the-art contribution, the micro-efficiency, the core usage (LTB only), the number of new problems solved, and the number of problems solved in each problem category. In each of the results tables, the CASC-J7 winner is *emphasized*. Detailed results, including the systems' output files, are available from the CASC-25 web site.

3.1. The THF Division

Table 4 summarizes the results of the THF division. Satallax 2.8 differed from the previous version Satallax 2.7 only in its proof output. Satallax-MaLeS solved more problems than Satallax 2.7 in CASC-J7, so it is not surprising that Satallax-MaLeS solved more problems than Satallax 2.8 in CASC-25. However, in CASC-J7 the THF division's ranking was based on problems solved, without requiring proofs to be output, and Satallax-MaLeS does not output proofs. Thus Satallax was the winner of this THF division. Generally, there has not been significant progress in the THF systems in the last year. Next year it is expected that the new LEO-III system [17] will be entered, which might produce some new results.

The SoTA contribution values are all very similar, indicating that all the systems have similar degrees of unique capability. Isabelle has a significantly lower micro-efficiency, reflecting its higher average CPU time caused by a long start-up time, and a strategy schedule in which often the sixth or seventh strategy (the z3 and cvc4 strategies) solves the problem, i.e., the time spent on earlier strategies in the schedule is "wasted". A better tuned strategy schedule would improve Isabelle's performance. The systems' abilities to solve the new problems, and their performances in the two problem categories, align with their overall performances. This suggests that there is no specific tuning to existing TPTP problems, or specialization to problems with or without equality.

ATP System	THF	Avg	Prfs	SOTA	μ	New	TNE	TEQ
	/400	CPU	out	Con.	Eff.	/7	/100	/300
Satallax 2.8	271	15.0	268	0.36	390	4	71	200
LEO-II 1.6.2	195	12.3	191	0.36	366	3	49	146
Sat'x-MaLeS 1.3	285	21.7	0	0.35	240	4	71	214
Isabelle 2015	267	61.0	0	0.38	30	4	66	201

Table 4
THF division results

The individual problem results show that 32 problems were unsolved, 95 problems were solved by all the systems, and 49 problems were solved by only one system. Of those 49 unique solutions, 17 were by Isabelle, 13 by LEO-II, 12 by Satallax, and 7 by Satallax-MaLeS. The unique solutions suggest that a portfolio approach would be effective for these THF problems. Giving Isabelle, LEO-II, and Satallax 100s each would solve 352 problems.

3.2. The THN Division

Table 5 summarizes the results of the THN division. Nitpick's dominant performance is a consequence of Nitpick's overwhelming influence on the TPTP problem ratings for this type of problem, and hence the eligibility of the problems for CASC. There is very little diversity of systems for this type of problem, and this division will be placed in

ATP System	THN /200	Avg CPU	Mdls out	SOTA Con.	μ Eff.	New /5	TNN /25	TNQ /175
Nitpick 2015	200	7.9	✓	0.70	130	5	25	175
Refute 2015	74	24.3	×	0.49	29	2	25	49
Satallax 2.8	49	0.1	×	0.48	242	2	5	44

Table 5
THN division results

a hiatus state until there has been further development. However, credit given where credit due, and it is worth noting that Nitpick uniquely solved 82 problems, and output models for all the problems.

3.3. The TFA Division

Table 6 summarizes the results of the TFA division. The winner, VampireZ3, is a newcomer to CASC. It benefits from being "built on the shoulders of giants", the well known Vampire [6] and Z3 [5] systems. VampireZ3 leveraged the new AVATAR architecture in Vampire [16], using Z3 to guide splitting decisions so that the first-order solver deals with sets of clauses whose ground part is consistent with the theories of equality and arithmetic. This approach seems to have great potential. A brief system description of VampireZ3 is provided in Section 4. CVC4 1.4, which won the TFA division of CASC-J7, was relegated to the bottom of the table because it does not output proofs - this was not required in CASC-J7. In terms of numbers of problems solved there is clear evidence of progress in this important area of ATP, with four of this year's systems solving more problems than CVC4 1.4. In particular, CVC4 1.5 has improved greatly over CVC4 1.4, due to the addition of a new instantiation-based technique [7] that was optimized for quantifiers in pure linear arithmetic, and improved implementations of the various E-matching techniques that are effective on problems combining arithmetic with function symbols. The importance of TFA problems in application domains is growing, and this division will be emphasized in CASC-J8 (the next CASC).

The SOTA contributions are all very similar, indicating similar levels of unique capability. Princess is the only system whose efficiency is notably lower. The main differences between the performances of the systems are in the TFI problem category. The TFI category had the most eligible problems, indicating that this kind of problem is important to users and developers. The five new problems are all TFI problems that verify properties of the SPARCT2 RTL chip. These problems were hard for Beagle and Princess because they have a large combinatorial search space, and make little use of arithmetic. The need for proof output for users (both human and machine) of these ATP systems is well understood, and it is pleasing to note that six of the systems output proofs, including the newcomer ZenonArith. Like VampireZ3, ZenonArith is "built on the shoulders of a giant", viz. Zenon [3]. A brief system description of ZenonArith is provided in Section 4.

The individual problem results show that 4 problems were unsolved, 41 problems were solved by all the systems, and 7 problems were solved by only one system. Of those 7 unique solutions, 3 were by CVC4 1.5, 2 by Beagle, and 1 by each of VampireZ3 and CVC4 1.4. A portfolio of these five systems, giving each 60s, would solve 196 problems.

3.4. The TFN Division

Table 7 summarizes the results of the TFN division. The obvious deficiency of this division was the small number of eligible problems, which will hopefully be remedied in the TPTP in time for CASC-J8 – users and developers are encouraged to submit suitable problems to the TPTP. CVC4 solved all of the 10 problems that any of the systems solved, and solved them all in 0.1s. These 10 problems are all purely arithmetical without function symbols, and the new instantiation-based technique that was effective in the TFA division (see Section 3.3) was again effective here. There were 7 software verification problems that none of the systems could solve, but which are considered to be relatively simple problems for infinite state model checkers. They were eligible for CASC because they have been solved by Z3, which was not entered into the competition.

None of the systems output models, which is an important capability in applications. A future TFN division will, like the FNT division, require models to be output. Like TFA problems, the importance of this type of problem in application domains is growing, and this division will be emphasized alongside the TFA division in CASC-J8.

ATP System	TFA /200	Avg CPU	Prfs out	SOTA Con.	μ Eff.	New /5	TFI /150	TFR /35	TFE /15
VampireZ3 1.0	172	11.9	172	0.21	478	4	126	34	12
CVC4 1.5	163	17.3	163	0.21	424	3	116	35	12
Vampire 4.0	160	10.8	160	0.20	422	4	114	34	12
Beagle 0.9.22	131	21.8	131	0.18	275	0	81	35	15
SPASS+T 2.2.22	108	10.0	108	0.14	186	2	60	35	13
ZenonArith 0.1.0	60	2.9	60	0.14	281	2	12	35	13
Princess 150706	143	17.4	0	0.18	152	0	100	33	10
CVC4 1.4	131	10.7	0	0.17	395	4	84	35	12

Table 6

TFA division results

ATP System	TFN /20	Avg CPU	Mdls out	SOTA Con.	μ Eff.	TIN /15	TRN /3	TEN /2
CVC4 1.5	10	0.0	×	0.50	500	8	1	1
Princess 150706	6	1.0	×	0.42	200	6	0	0
Beagle 0.9.22	6	1.3	×	0.42	183	4	1	1

Table 7

TFN division results

3.5. The FOF Division

Table 8 summarizes the results of the FOF division. Vampire 4.0 beat Vampire 2.6, which was the winner of the FOF division for the last three years. Vampire 4.0 had a low average proof time, the highest SOTA contribution, and the second highest efficiency. Vampire 4.0 is clearly the new powerhouse in FOF ATP. However, the old Vampire 2.6 still had the highest efficiency, due to lower CPU times for the easier problems, and it also solved the most FNE problems. Both Vampires failed to output proofs for some of the problems solved. For Vampire 2.6, as was noted in the CASC-J7 report [14], some proofs were not output within the time limit because they are very large. For Vampire 4.0 some proofs were not output due to a bug, which has been repaired in Vampire 4.1. ePrincess, the winner of the "best newcomer" award for CASC-25, performed reasonably well in this tough division. Like the other newcomers (VampireZ3 and ZenonArith in the TFA division) ePrincess is "built on the shoulders of a giant", viz. Princess [8]. A brief system description of ePrincess is provided in Section 4. The problem category results align well with the overall results, with only iProver having a notable preference for FNE problems.

The individual problem results show that 7 problems were unsolved, 2 problems were solved by all the systems, and 10 problems were solved by only one system. Of those 10 unique solutions, 4 were by each of Vampire 4.0 and E, and 2 were by Vampire 2.6. Although ET did not have any unique solutions, it would provide a useful contribution to a portfolio; the two Vampires and ET, giving each 100s, would solve 387 problems.

3.6. The FNT Division

Table 9 summarizes the results of the FNT division. This is the first time Vampire has won this division. The main reason for Vampire's improved performance is the addition of a finite model builder, which solves the majority of problems – 91 of the 106 problems solved in the FNN problem category, and 61 of the 89 problems solved in the FNQ problem category. It is also pleasing to see that the new version of iProver beat last year's winning version by a substantial margin. The improvements made to iProver include incremental finite model finding, improved sort inference using a propositional SAT solver, improved strategies, and the use of competition parallelism (as opposed to sequential strategy scheduling). There has clearly been progress in this area of ATP, in contrast to the apparent stagnation last year.

Vampire does not only solve the most problems, it also has the highest SOTA contribution and the highest efficiency. The problem category rankings align with the overall ranking, indicating no specialization to problems with or without equality. It is pleasing to note that all the systems output models.

The individual problem results show that only 2 problems were unsolved, 9 problems were solved by all the systems, and 17 problems were solved by only one system. Of those 17 unique solutions,

ATP System	FOF /400	Avg CPU	Prfs out	SOTA Con.	μ Eff.	New /1	FNE /150	FEQ /250
Vampire 4.0	380	12.2	374	0.20	552	1	139	241
Vampire 2.6	371	14.9	368	0.19	574	1	144	227
E 1.9.1	316	20.2	316	0.17	450	1	122	194
ET 0.2	303	21.0	303	0.16	344	0	119	184
CVC4 1.5	257	33.4	256	0.15	287	0	111	146
iProver 2.0	222	21.1	217	0.14	215	1	125	97
leanCoP 2.2	159	46.8	159	0.12	150	0	85	74
iProverM'lo 0.7-0.3	127	30.2	127	0.12	136	1	91	36
Prover9 1109a	111	28.0	111	0.14	138	0	29	82
ePrincess 1.0	113	48.4	88	0.13	19	0	35	78
Muscadet 4.5	37	7.3	37	0.11	70	0	18	19
Geo-III 2015E	37	38.8	37	0.14	50	0	15	22

Table 8

FOF division results

16 were by Vampire, and 1 by CVC4. A portfolio approach does not help.

ATP System	FNT /200	Avg CPU	Mdls out	SOTA Con.	μ Eff.	New /2	FNN /110	FNQ /90
Vampire 4.0	195	39.0	195	0.37	368	2	106	89
iProver 2.0	163	44.1	163	0.29	226	2	92	71
iProver 1.0	134	80.0	134	0.28	39	2	77	57
CVC4 1.5	71	57.8	71	0.25	121	2	21	50
E 1.9.1	51	9.6	51	0.33	127	0	32	19
Geo-III 2015E	38	21.9	38	0.20	112	0	8	30

Table 9

FNT division results

3.7. The EPR Division

Table 10 summarizes the results of the EPR division. As for the FNT division, this is the first time that Vampire has won the division. General improvements in the implementation of AVATAR and its SAT solvers have improved Vampire's ability to find refutations (in the EPT category) and saturations (in the EPS category) for EPR problems. In the EPS category, Vampire's new finite model builder solved 15 of the 69 problems that Vampire solved, and a subsequent experiment showed that it could have solved 38 problems if placed earlier in the strategy schedule. iProver 0.9 was the winner of the last five years of the EPR division, and earlier versions won for two years before that. It is disappointing that the new iProver 2.0 was beaten by the old version, due to a weaker performance in the EPT problem category.

The individual problem results show that 6 problems were unsolved, 7 problems were solved by all the systems, and 24 problems were solved by only one system. Of those 24 unique solutions, 23 were by Vampire, and 1 by iProver 0.9. A portfolio approach does not help.

ATP System	EPR /200	Avg CPU	Pr out	M	SOTA Con.	μ Eff.	EPT /125	EPS /75
Vampire 4.0	192	27.6	✓	✓	0.39	445	123	69
iProver 0.9	161	27.9	×	✓	0.30	250	106	55
iProver 2.0	153	36.6	✓	✓	0.28	242	96	57
E 1.9.1	101	11.1	✓	✓	0.25	327	61	40
Geo-III 2015E	9	86.7	✓	✓	0.21	11	1	8

Table 10

EPR division results

3.8. The LTB Division

Table 11 summarizes the results of the LTB division. Vampire clearly dominated the division. It was noted in Section 3.5 that Vampire 4.0 had a bug that caused some problems to be solved but with no proof output - this bug was fixed in the version used in the LTB division. The previous winner, MaLARea, suffered from some component software failures in the HL4 problem category, but otherwise still performed well. MaLARea solved the most problems in the MZR problem category, where it benefited from machine learning on the large number of proofs provided with the 1000 training problems. The top three systems made use of the available cores, with core usages over 3.0.

The individual problem results show that 280 problems were unsolved (82 HLL, 68 HL4, 64 ISA, 66 MZR), 237 problems were solved by all the systems (53 HLL, 8 HL4, 19 ISA, 157 MZR), and 313 problems were solved by only one system (86 HLL, 125 HL4, 54 ISA, 48 MZR). Of those 313 unique solutions, 214 were by Vampire, 77 by MaLARea, 16 by E, and 6 by iProver. The HLL category has the most to gain from a portfolio approach; giving Vampire and MaLARea 30s each would solve 299 HLL problems.

4. System Descriptions

There were three newcomers in CASC-25, all of which are "built on the shoulders of giants", as noted in Sections 3.3 and 3.5. They are VampireZ3 1.0, ZenonArith 0.1.0, and ePrincess 1.0 (which won the best newcomer prize). It is interesting to understand how these systems have been constructed, taking advantage of the existing "giant" capabilities. The following brief descriptions of these systems were written by their developers.

VampireZ3 1.0 is a combination of Vampire 4.0 [6] and Z3 4.3.1 [5]. A key to the recent success of Vampire 4.0 is its usage of AVATAR [16], which uses a SAT solver to make splitting decisions. VampireZ3 leverages this architecture. In "plain" Vampire AVATAR works by representing the first-order search space in the SAT solver, using propositional symbols to consistently name variable-disjoint components. The model produced by the SAT solver is then used to (iteratively) select a subset of components for proof search. In VampireZ3, the Z3 SMT solver is used in place of the SAT solver, and the *ground* components are translated into Z3 terms (the non-ground components continue to be represented by propositional symbols). This means Z3's efficient methods for ground reasoning with equality and theories are exposed by AVATAR, as the SMT solver only produces theory-consistent models. Of the 172 problems that VampireZ3 solved in the TFA division, only 81 were solved by strategies making use of Z3. VampireZ3 1.0 can use any strategy available to Vampire 4.0, including using a SAT solver for splitting, and evidently these strategies proved useful. For reasoning with theories Vampire 4.0 uses a combination of evaluation, theory axiom introduction and simple instantiation. VampireZ3's success can be attributed to the combination of Vampire's efficient quantifier reasoning with Z3's powerful ground theory reasoning.

ZenonArith 0.1.0 [4] is an extension of the tableau-based Zenon ATP system [3] to linear arithmetic. The extension uses the simplex algorithm as a decision procedure to solve problems over rationals, and a branch-and-bound approach to solve problems over integers. It is also able to handle real linear arithmetic, which coincides with the rational fragment of linear arithmetic due to the syntactical restrictions of the TPTP input format for reals. The extension consists of a smooth integration of arithmetic deductive rules to the usual tableau rules, so that there is a natural interleaving between the arithmetic and regular analytic rules. Universally quantified formulae are handled by translating the unsatisfiability explanations from the simplex algorithm into inference rules. Instantiation for the existentially quantified formulae is achieved by trying to solve sets of formulae that cover all branches of a derivation tree. ZenonArith is implemented in OCaml, and uses the Zarith library to handle arbitrary precision integers and rationals. An incremental implementation of the simplex algorithm in OCaml is used as the decision procedure over linear arithmetic. The underlying Zenon system provides a built-in notion of types for terms, and inference rules that discriminate over the types of expressions. ZenonArith can automatically output Coq proof scripts, which can be checked by the Coq proof assistant. While quite simple, the approaches of ZenonArith work quite well over rational and real systems, as can be seen in the competition results. The branch-and-bound strategy used for integer systems seems less effective.

ePrincess 1.0 [2] is a theorem prover for first-order logic with equality. It is an extension of the Princess theorem prover [8], which uses a combination of techniques from the areas of first-order reasoning and SMT solving. The main underlying calculus is a free-variable, backtracking-free, tableau calculus. ePrincess extends the calculus in Princess with rules for handling equality, based on a complete procedure utilizing Bounded Rigid E-Unification (BREU) as a foundation of the unification step. The two different procedures for solving BREU problems from [1] have been implemented. ePrincess 1.0 competed in the FOF division where its performance was promising enough

ATP System	LTB /1600	Avg CPU	Avg WC	Prfs out	SOTA Con.	μ Eff.	Core Usage	HLL /400	HL4 /400	ISA /400	MZR /400
Vampire 4.0	1208	24.1	6.5	1208	0.49	387	3.7	279	322	319	288
MaLARea 0.5	837	27.9	8.3	837	0.41	96	3.4	243	9	280	305
E 1.9.1	799	27.8	7.4	799	0.36	214	3.8	154	214	202	229
iProver 2.0	352	30.4	13.0	352	0.29	51	2.3	59	77	21	195

Table 11
LTB division results

to earn the “best newcomer” award. It was shown to be stronger on problems that include equality, which is to be expected since the BREU calculus focusses on first-order logic with equality. While the number of problems solved is not yet close to the top-performers, it was enough to show that ePrincess is competitive with other tableaux systems on problems with equality.

5. Conclusion

CASC-25 was the twentieth large scale competition for fully automatic, classical logic ATP systems. CASC-J7 fulfilled its objectives by evaluating the relative abilities of current ATP systems, and stimulating development and interest in ATP.

The highlight of CASC-25 was the dominance of Vampire 4.0, and the developers have set a new high standard for others to follow. This dominant performance by Vampire does raise some concern that other developers will lose hope of winning, and CASC will lose some of its attraction. However, the TFA and TFN divisions will be emphasized next year in CASC-J8, and the THF division is still in relative infancy. There’s a lot of space out there for new ideas and implementations!

While the design of CASC is mature and stable, each year’s experiences lead to ideas for changes and improvements. Some changes that are being considered for CASC-J8 are:

- The TFA and TFN divisions will be emphasized (hopefully with real cash prizes).
- The THN division will go into hiatus state.
- In order to motivate the development of systems that take advantage of multiple cores, a ranking based on wall clock time will be done in the FOF and FNT divisions.

As always, the ongoing success and utility of CASC depends on ongoing contributions of problems to the TPTP. The automated reasoning community is encouraged to continue making contributions of all types of problems. In particular, typed first-order problems with arithmetic, i.e., suitable for the TFA and TFN divisions, are needed.

References

[1] P. Backeman and P. Rümmer. Efficient Algorithms for Bounded Rigid E-Unification. In H. de Nivelle, editor, *Proceedings of the 24th Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, number 9323 in Lecture Notes in Computer Science, pages 70–85. Springer-Verlag, 2015.

[2] P. Backeman and P. Rümmer. Theorem Proving with Bounded Rigid E-Unification. In A. Felty and A. Middeldorp, editors, *Proceedings of the 25th International Conference on Automated Deduction*, number 9195 in Lecture Notes in Computer Science, pages 572–587. Springer-Verlag, 2015.

[3] R. Bonichon, D. Delahaye, and D. Doligez. Zenon : An Extensible Automated Theorem Prover Producing Checkable Proofs. In N. Dershowitz and A. Voronkov, editors, *Proceedings of the 14th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning*, number 4790 in Lecture Notes in Artificial Intelligence, pages 151–165, 2007.

[4] G. Bury and D. Delahaye. Integrating Simplex with Tableaux. In H. de Nivelle, editor, *Proceedings of the 24th Conference on Automated Reasoning with Analytic Tableaux and Related Methods*, number 9323 in Lecture Notes in Computer Science, pages 86–101. Springer-Verlag, 2015.

[5] L. de Moura and N. Bjorner. Z3: An Efficient SMT Solver. In C. Ramakrishnan and J. Rehof, editors, *Proceedings of the 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, number 4963 in Lecture Notes in Artificial Intelligence, pages 337–340. Springer-Verlag, 2008.

[6] L. Kovacs and A. Voronkov. First-Order Theorem Proving and Vampire. In N. Sharygina and H. Veith, editors, *Proceedings of the 25th International Conference on Computer Aided Verification*, number 8044 in Lecture Notes in Artificial Intelligence, pages 1–35. Springer-Verlag, 2013.

[7] A. Reynolds, M. Deters, V. Kuncak, C. Barrett, and C. Tinelli. Counterexample Guided Quantifier Instan-

tiation for Synthesis in CVC4. In D. Kroening and C. Pasareanu, editors, *Proceedings of the 27th International Conference on Computer Aided Verification*, number 9207 in Lecture Notes in Computer Science, pages 198–216. Springer-Verlag, 2015.

[8] P. Rümmer. A Constraint Sequent Calculus for First-Order Logic with Linear Integer Arithmetic. In I. Cervesato, H. Veith, and A. Voronkov, editors, *Proceedings of the 15th International Conference on Logic for Programming Artificial Intelligence and Reasoning*, number 5330 in Lecture Notes in Artificial Intelligence, pages 274–289. Springer-Verlag, 2008.

[9] A. Stump, G. Sutcliffe, and C. Tinelli. StarExec: a Cross-Community Infrastructure for Logic Solving. In S. Demri, D. Kapur, and C. Weidenbach, editors, *Proceedings of the 7th International Joint Conference on Automated Reasoning*, number 8562 in Lecture Notes in Artificial Intelligence, pages 367–373, 2014.

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

[11] G. Sutcliffe. The TPTP Problem Library and Associated Infrastructure. The FOF and CNF Parts, v3.5.0. *Journal of Automated Reasoning*, 43(4):337–362, 2009.

[12] G. Sutcliffe. The TPTP World - Infrastructure for Automated Reasoning. In E. Clarke and A. Voronkov, editors, *Proceedings of the 16th International Conference on Logic for Programming Artificial Intelligence and Reasoning*, number 6355 in Lecture Notes in Artificial Intelligence, pages 1–12. Springer-Verlag, 2010.

[13] G. Sutcliffe. Proceedings of the CADE-25 ATP System Competition. Berlin, Germany, 2015. http://www.tptp.org/CASC/25/Proceedings.pdf.

[14] G. Sutcliffe. The 7th IJCAR Automated Theorem Proving System Competition - CASC-J7. *AI Communications*, 28(4):683–692, 2015.

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

[16] A. Voronkov. AVATAR: The New Architecture for First-Order Theorem Provers. In A. Biere and R. Bloem, editors, *Proceedings of the 26th International Conference on Computer Aided Verification*, number 8559 in Lecture Notes in Computer Science, pages 696–710, 2014.

[17] M. Wisniewski, A. Steen, and C. Benzmüller. The Leo-III Project. In A. Bolotov and M. Kerber, editors, *Proceedings of the Joint Automated Reasoning Workshop and Deduktionstreffen*, page 38, 2014.