Progress in Automated Theorem Proving, 1997–2001

Geoff Sutcliffe[1], Matthias Fuchs[2], and Christian Suttner[3]

[1] University of Miami, Miami, USA
geoff@cs.miami.edu
[2] Australian National University, Canberra, Australia
fuchs@arp.anu.edu.au
[3] Antfactory, München, Germany
csuttner@antfactory.com

Abstract. Despite some impressive individual achievements, the extreme difficulty of Automated Theorem Proving (ATP) means that progress in ATP is slow relative to, e.g., some aspects of commercial information technology. The (relatively) slow progress has two distinct disadvantages. First, for the researchers, it is difficult to determine if a direction of investigation is making a meaningful contribution. Second, for unaware observers, a lack of progress leads to a loss of interest and confidence in the field. In this context it is important that progress in ATP be measured, monitored, and recognized. This paper presents quantitative measures that show progress in ATP, from mid-1997 to mid-2001. The measures are based on collected performance data from ATP systems.

1 Introduction

Automated Theorem Proving (ATP) is concerned with the development and use of systems (computer programs) that automate sound reasoning: the derivation of conclusions that follow inevitably from facts. This capability lies at the heart of many important computational tasks, e.g., software verification [19, 7], and the development of mathematical theories [16, 10]. ATP systems are presented with problems written in some logic. Classical 1st order logic is widely used because of its semi-decidability, and all references to ATP systems and problems in this work are for classical 1st order logic. The ideas presented can, however, readily be transferred to other cases.

The development of useful ATP systems started in the mid-1960s, and has progressed to a point now where current ATP systems are capable of solving non-trivial problems, e.g., EQP solved the Robbins problem [15]. This progress is impressive, given that ATP is "possibly the hardest subfield of Computer Science" [23]. Noteworthy landmarks in this history include:

- The resolution inference rule [21].
- The series of early ATP systems developed at the Argonne National Laboratories [13], which, among other contributions, introduced the "given clause" control loop.

– Paramodulation as an alternative to the explicit use of equality axioms [20].
– Subsumption as an effective means for controlling redundant information [33].
– The tableau and model elimination strategies [5, 11], which are effective ATP strategies and also the basis for Prolog [9].
– The Knuth-Bendix completion procedure [8] and related methods for unit equality reasoning [34, 2].
– Indexing techniques for highly efficient storage and access to the data structures used by ATP systems [24, 18].
– The superposition inference rule [1].

There have also been some impressive implementations of the various calculi and search strategies, such as Otter [14], Gandalf [30], Waldmeister [6], SPASS [32], Vampire [31], and E [22].

Despite these individual achievements, the extreme difficulty of ATP means that progress in ATP is slow relative to, e.g., some aspects of commercial information technology. The (relatively) slow progress has two distinct disadvantages. First, for the researchers, it is difficult to determine if a direction of investigation is making a meaningful contribution. This is troublesome both in terms of motivation (obvious progress is always encouraging) and in terms of focus (expend more energy in directions that are successful). Second, for unaware observers, a lack of progress leads to a loss of interest and confidence in the field. A serious outcome of this loss of interest and confidence has been the withdrawal of significant funding for ATP research, e.g., the need for revitalized funding in the USA was highlighted in [12], and in Germany the DFG "Schwerpunktprogramm Deduktion" ended in 1998 and has not been replaced.

In this context of slow progress, it is important that progress in ATP be measured, monitored, and recognized. This paper presents quantitative measures that show progress in ATP, from mid-1997 to mid-2001. The measures are based on collected performance data from ATP systems. Section 2 describes the source, organization, and features of the performance data, which is then analyzed in Section 3. Section 4 concludes the paper.

2 Performance Data

In order to demonstrate progress in ATP, it is necessary to evaluate ATP over time. Evaluation of individual theoretical results, implementation techniques, etc, is possible, but from a user perspective these separate contributions are of little interest. Evaluation of the final product of ATP research, that is, the combination of theoretical results, implementation techniques, etc, into ATP systems, satisfies both user and developer perspectives of progress. This work thus demonstrates progress in ATP through evaluation of ATP systems over time. Analytic approaches to ATP system evaluation provide insights into theoretical system capabilities. However, complete analysis of the search space at the 1st order level is of course impossible (or P=NP). It is therefore necessary to make empirical evaluations of the ATP systems. Empirical evaluation using artifacts

specific to ATP, e.g., inference steps or formulae generated, is not possible because the different calculi and systems mostly have incomparable features. In any case, such an evaluation would provide little useful information for potential ATP users. The evaluation methodologies are thus based on the simple measure of whether or not systems solve the problems (a full explanation of this basis for empirical evaluation of ATP systems and problems is in [29]). The results provide information that is relevant to both developers and potential users, and also encapsulate the more fine grained features.

2.1 Specialist Problem Classes

An empirical evaluation of ATP systems requires a selection of ATP problems for the systems to attempt. ATP problems have easily identifiable logical, language, and syntactic characteristics. Various ATP systems and techniques have been observed to be particularly well suited to problems with certain characteristics, e.g., everyone agrees that special techniques are deserved for problems with equality. Due to this specialization, empirical evaluation of ATP systems must be done in the context of problem sets that are reasonably homogeneous with respect to the systems. These problem sets are called *Specialist Problem Classes* (SPCs), and are based on problem characteristics. The choice of what problem characteristics are used to form the SPCs is based on community input and on analysis of system performance data [3]. The range of characteristics that have so far been identified as relevant are: theoremhood (theorems vs non-theorems), order (essentially propositional vs real 1st order), equality (no equality vs some equality vs pure equality), form (CNF vs FOF), Horness (Horn vs non-Horn), and unit equality (unit equality vs non-unit pure equality). Based on these characteristics, 16 SPCs have been defined, as indicated by the leaves of the tree in Figure 1.

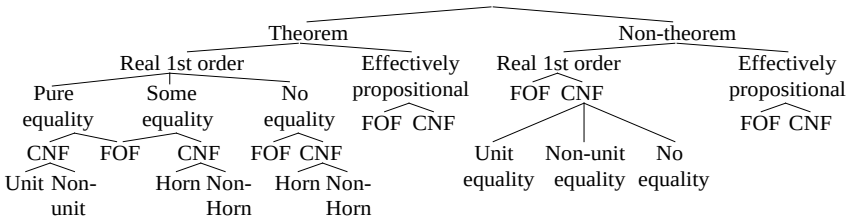


Fig. 1. Specialist Problem Classes

The SPCs are named using mnemonic acronyms, abbreviating theorem to `THM`, non-theorem to `SAT`, real 1st order to `RFO`, essentially propositional to `EPR`, pure equality to `PEQ`, some equality to `SEQ`, no equality to `NEQ`, unit equality to `UEQ`, non-unit pure equality to `NUE`, Horn to `HRN`, and non-Horn to `NHN`. `CNF` and `FOF` are retained as is.

2.2 The TPTP Problem Library

Currently there are not many "real" applications of 1st order ATP (current applications of ATP, such as software and hardware verification, largely use propositional techniques). There is therefore no corpus of application problems that can be used for testing 1st order ATP systems. The TPTP (Thousands of Problems for Theorem Provers) problem library is a library of test problems for ATP systems [27]. The TPTP is large enough to obtain statistical significance, and spans a diversity of subject matters. The TPTP is regularly updated with new problems, including problems from "real" applications of ATP. The TPTP is the best available collection of problems representing general purpose applications of ATP, and thus is the best source of problems for evaluating ATP systems. Since the first release of the TPTP in 1993, many researchers have used the TPTP as an appropriate and convenient basis for testing their ATP systems. Although other test problems do exist and are sometimes used, the TPTP is now the de facto standard for testing classical 1st order ATP systems.

Some researchers who have tested their ATP systems over the entire TPTP problem library have contributed their performance data to the TPTP results collection [28]. The results are for various ATP systems, various system versions, and various TPTP versions. The results collection thus provides snapshots of ATP systems' performances over time, and forms a basis for measuring progress in ATP.

2.3 System Performance Curves

The performance data in the TPTP results collection is provided by the individual system developers, which means that the systems have been tested using a range of CPU and memory resource limits. Analysis shows that the differences in resource limits do not significantly affect how many problems are solved by each ATP system. Figure 2 plots the CPU times taken by several well known systems to solve problems in the SPC `THM_RFO_SEQ_CNF_NHN`, for each solution found, in increasing order of time taken.[1] The relevant feature of these *performance curves* is that they are exponential in nature, as would be expected for search in an exponentially growing search space (the performance curves in other SPCs have the same feature). Each system has a point at which the time taken to find solutions starts to increase dramatically. This point is called the system's *Peter Principle Point* (PPP), as it is the point at which the system has reached its level of incompetence.[2] A linear increase in the CPU resources beyond the PPP would not lead to the solution of significantly more problems. The PPP thus defines a realistic CPU resource limit for the system. From an ATP perspective, after the PPP the search space has typically grown to a size where the

[1] The numbers of solutions found are not comparable, as the systems attempted the SPC in different TPTP versions

[2] The Peter Principle is "The theory that employees within an organization will advance to their highest level of competence and then be promoted to and remain at a level at which they are incompetent." [17]

system is unable to find a solution within the space. The PPP thus also defines a realistic memory resource limit for the system. Provided that enough CPU time and memory are allowed for the ATP system to pass its PPP, a usefully accurate measure of what problems it can solve within realistic resource limits is achieved. Performance curves provide a basis for evaluating the progress in ATP over time. This is described in Sections 3.1, 3.2, and 3.3.

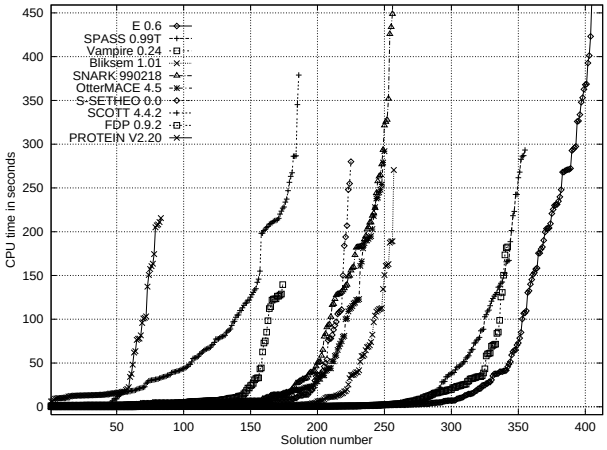


Fig. 2. Proof number vs CPU time

2.4 ATP System and Problem Evaluation

[29] presents methodologies for the empirical evaluation of ATP systems and problems, within individual SPCs. The methodologies may be summarized as follows. Initially a partial ordering of the systems is determined by *subsumption*: a system that solves (with realistic resource limits) a strict superset of the problems solved by another system subsumes, and is better than, the other system. The non-subsumed systems are designated *rating contributors*. If the number of rating contributors is less than a threshold, then other high performing but subsumed systems are also made rating contributors. (This use of subsumed rating contributors improves the ratings produced, as is explained in [29]). A problem is rated according to the fraction of rating contributors that fail to solve the problem. Problems with a rating of 0.00 are *easy*, with a rating between 0.00 and 1.00 are *difficult*, and with a rating of 1.00 are *unsolved*. Finally, the ATP systems are rated according to the fraction of difficult problems they can solve.

The TPTP results collection is used to rate the systems and the problems in the TPTP. The change of a problem rating from unsolved to difficult captures the point at which a problem is solved for the first time by an ATP system

(according to the collected data), which is an indication of progress in ATP. This is described in Section 3.4. Overall reductions in problem ratings over time are also a measure of progress in ATP. This is described in Section 3.5.

Each year since 1996, an empirical evaluation of ATP systems has been performed at CADE [25].[3] The CADE ATP System Competition (CASC) evaluates the performance of fully automatic ATP systems for classical 1st order logic. The evaluation is in terms of the number of problems solved and the average runtime for successful solutions, in the context of a bounded number of eligible problems chosen from the TPTP, and a specified CPU time limit for each solution attempt. The CPU time limit, and the memory in the computers used, are adequate for the ATP systems to reach their PPPs. The CASC results can be influential with regard to funding and other recognition for the ATP system developers. As a result, most of the decent contemporary ATP systems are entered, and the CASC results provide a way to show relative progress of ATP systems over time. This is described in Section 3.6.

3 Progress in ATP

To measure the progress in ATP, the performance of ATP systems has been analyzed in two ways. First, the performance data in the TPTP results collection, over a four year period, has been analyzed. The results analyzed are for TPTP versions v2.0.0, released on 5th June 1997, v2.1.0, released on 17th December 1997, v2.2.0, released on 11th February 1999, v2.3.0, released on 16th November 1999, and v2.4.0, released in April 2001. Second, the performance of ATP systems in CASC over a four year period has been analyzed. In all cases, as is explained above, the analysis is in the context of individual SPCs.

3.1 SOTA System Performance, part I

To evaluate overall quality and progress in ATP, the individual ATP systems tested on an SPC in a TPTP version are combined to form a *state-of-the-art* (SOTA) system. For any problem, a SOTA system has the performance of the best available individual system for the problem, i.e., the time taken by the SOTA system to solve a problem is the minimum of the times taken by the available individual systems. A SOTA system can really be built, by running the individual systems in competition parallel, as done in the SSCPA system [26]. A SOTA system's performance is thus a realistic measure of the combined quality of the ATP systems of the time. A comparison of the SOTA systems' performances for an SPC in two TPTP versions provides evidence of progress in ATP for that SPC. Note that the contributions of the individual systems to a SOTA system are dependent on the problems being attempted, but like the individual system performance curves, the performance curve of a SOTA system has an exponential shape.

[3] CADE, the Conference on Automated Deduction, is the major forum for the presentation of new research in automated deduction.

An initial comparison of the SOTA systems for two TPTP versions can be made by comparing their raw performance on problems that were in both TPTP versions. Part I of Table 1 gives the results for TPTP versions v2.0.0 and v2.4.0. For each SPC, the second column gives the number of problems in both TPTP versions, the third and fourth columns give the number of problems solved and the maximal time taken by the v2.0.0 SOTA system, and the fifth and sixth columns give that information for the v2.4.0 SOTA system (the last column and Part II give the results of the analysis described in Section 3.3). In most of the SPCs there is significant increase in the number of problems solved between the TPTP versions, indicating progress in ATP. Section 3.3 refines this analysis to compensate for the small dependence on resources available.

Part I: SPC	#	v2.0.0		v2.4.0			HIF
SAT_EPR_CNF	17	14	1s	17	1s		1.00
SAT_RFO_NEQ_CNF	45	26	1s	44	156s		1.00
SAT_RFO_EQU_CNF_NUE	17	10	4s	16	13s		1.07
SAT_RFO_EQU_CNF_UEQ	12	8	4s	11	15s		1.30
SAT_EPR_FOF	13	11	1s	13	1s		1.00
SAT_RFO_FOF	10	5	1s	10	4s		1.00
THM_EPR_CNF	304	301	11s	304	1s		1.02
THM_RFO_NEQ_FOF	15	15	1s	15	1s		1.00
THM_RFO_EQU_FOF	20	7	1s	19	9s		1.00
THM_RFO_NEQ_CNF_HRN	326	265	364s	289	225s		1.80
THM_RFO_NEQ_CNF_NHN	125	105	81s	113	124s		1.25
THM_RFO_SEQ_CNF_HRN	213	176	434s	197	165s		1.43
THM_RFO_SEQ_CNF_NHN	314	181	831s	221	204s		2.49
THM_RFO_PEQ_CNF_NUE	111	69	875s	97	129s		3.75
THM_RFO_PEQ_CNF_UEQ	357	309	329s	333	118s		1.59
THM_EPR_FOF	157	145	4s	157	292s		0.99
Part II: SPC	#	v2.4.0 ↓		v2.0.0 ↑		v2.0.0 requires	
THM_RFO_NEQ_CNF_HRN	326	288	202s	267	405s	1938s	8.61
THM_RFO_NEQ_CNF_NHN	125	113	65s	107	155s	1848s	14.91
THM_RFO_SEQ_CNF_HRN	213	200	304s	174	236s	662011s	401.22
THM_RFO_SEQ_CNF_NHN	314	229	334s	178	507s	306675s	1503.31
THM_RFO_PEQ_CNF_NUE	111	99	233s	67	484s	1610855s	12487.25
THM_RFO_PEQ_CNF_UEQ	357	336	207s	307	187s	2244s	18.02
THM_EPR_FOF	157	155	4s	152	290s	6941s	23.77

Table 1. Performance and exponential curve analysis

3.2 Exponential Curve Fitting

The increase in the number of problems solved by SOTA systems, from one TPTP version to another, may be extrapolated to resource limits beyond those found in the performance data. This is achieved by fitting *exponential curves* of the form $f(x) = ae^{bx}$ to the SOTA systems' performance curves. For the purpose of demonstrating progress in ATP, it is important that the exponential curves fit the performance curves most accurately at the higher solution numbers, as these data points correspond to harder ATP problems.

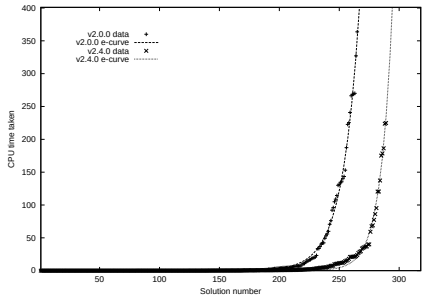
Exponential curves have been fitted to the performance curves of the SOTA systems for each of the SPCs in TPTP versions v2.0.0 and v2.4.0. For the SPCs

above the dividing line in Part I of Table 1, there is insufficient data for meaningful curve fitting and analysis. The performance and exponential curves for the other seven SPCs are shown in Figure 3. The gaps between the steep parts of the exponential curves are visual evidence of the progress in ATP in that period, for those SPCs.

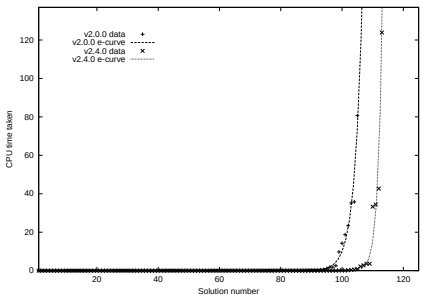
3.3 SOTA System Performance, part II

In the period between TPTP versions there may be hardware improvements that would make even an unchanged system perform better. Such hardware improvements may undermine confidence in the conclusion drawn in Section 3.1. However, hardware improvements can be taken into account, as follows. In order to estimate the *hardware improvement factor* (HIF) between two TPTP versions, the times taken to solve those problems solved by both versions' SOTA systems are extracted. The geometric average of the ratios of the times is computed, and used as an upper bound on the HIF. The computed HIF is an upper bound because it assumes that all changes in the times taken are caused by hardware improvements, while in reality some portion of the changes is caused by improvements in the systems. The HIFs are shown in the last column of Part I of Table 1. The HIF is used to scale the SOTA systems' exponential curves, and a comparison of the results then provides a lower bound on the progress in ATP. First, an estimate of the number of problems the v2.4.0 system would solve, if run on the v2.0.0 system's hardware with a time limit of the maximal time taken by the v2.0.0 system, is computed. To do this, the maximal time taken by the v2.0.0 system is scaled down by the HIF, and the inverse of the v2.4.0 system's exponential curve is applied. The result can be compared to the number of problems solved by the v2.0.0 system. Conversely, an estimate of the number of problems the v2.0.0 system would solve, if run on the v2.4.0 system's hardware with a time limit of the maximal time taken by the v2.4.0 system, is computed. The result can be compared to the number of problems solved by the v2.4.0 system. Finally, an estimate of the CPU time required by the v2.0.0 system to solve the number of problems solved by the v2.4.0 system is computed. The ratio of this time and the maximal time taken by the v2.4.0 system is the required HIF for the v2.0.0 system to solve the same number of problems as the v2.4.0 system, within the same maximal time. The required HIF can be compared to the actual computed HIF and to reality.

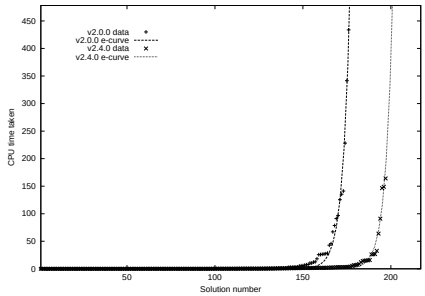
This hardware sensitive analysis has been applied to the performance curves shown in Figure 3, and the results are given in Part II of Table 1. The third and fourth columns give the scaled down v2.0.0 maximal time and the estimated number of problems that would be solved by the v2.4.0 system in this time. These values can be compared with the figures directly above in Part I of the table. The fifth and sixth columns give scaled up v2.4.0 maximal time and the estimated number of problems that would be solved by the v2.0.0 system in this time. The seventh and eight columns give CPU time and HIF required for the v2.0.0 system to solve the same number of problems as the v2.4.0. The extra



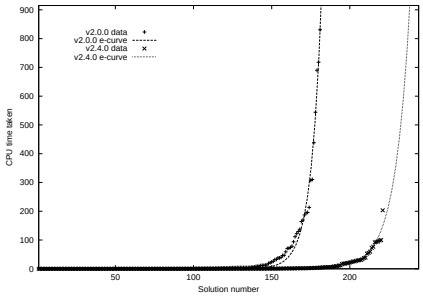
THM_RFO_NEQ_CNF_HRN



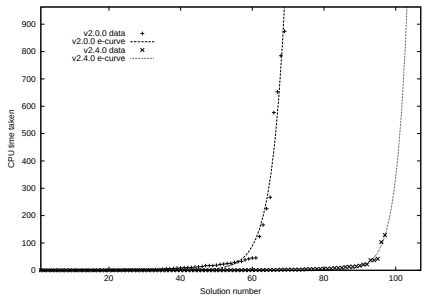
THM_RFO_NEQ_CNF_NHN



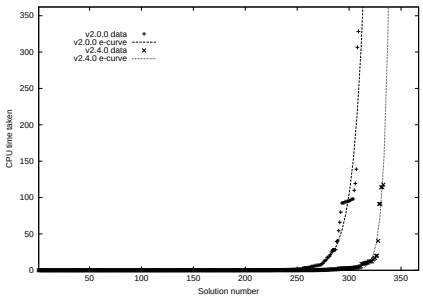
THM_RFO_SEQ_CNF_HRN



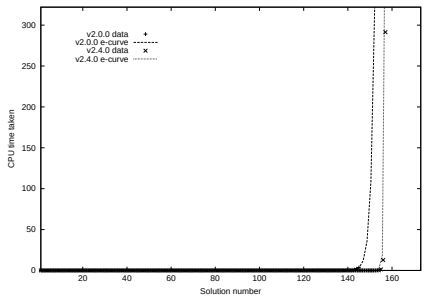
THM_RFO_SEQ_CNF_NHN



THM_RFO_PEQ_CNF_NUE



THM_RFO_PEQ_CNF_UEQ



THM_EPR_FOF

Fig. 3. Performance and exponential curves

problems solved by the v2.4.0 systems is evidence of progress in ATP, in these SPCs.

3.4 Problems Solved for the First Time

The first time solution of a problem that ATP systems had previously failed to solve is an indication of progress at the leading edge of ATP, and indicates that the solving system defines that part of the edge. This is particularly noticeable when the problem is one that humans have an interest in, but have failed to solve, e.g., EQP's solution of the Robbins problem [15], and MGTP's characterization of quasigroups [4]. As is the case with humans, major breakthroughs are few and far between. This is partly because it is hard to solve hard problems, and partly because hard problems that ATP systems have solved have not been a focus of human attention. It is therefore necessary to make a more senstive analysis of first time solutions, as a measure of progress in ATP.

The first time solution of a TPTP problem is easily detected from the TPTP problem ratings, which have been included in the TPTP since version v2.0.0. When a problem rating changes from 1.00 (unsolved) to less than 1.00 (difficult), the problem has been solved for the first time. Table 2 gives data about the first time solution of problems that were unsolved when TPTP v2.0.0 was released. The second column gives the number of such problems, and the subsequent columns give the numbers of problems that were solved for the first time in the periods between the TPTP versions. The numbers in Table 2 show regular first time solution of problems, indicating regular progress at the leading edge of ATP. Note that the first time solution of the problems becomes more impressive as the TPTP version increases, because the "easier" problems have already been solved. The analysis of Section 3.3 shows that hardware can account for only some small fraction of the first solutions.

3.5 Decreasing Problem Ratings

While the change of a problem rating from 1.00 to less than 1.00 shows progress at the leading edge of ATP, overall reductions in problem ratings show general improvement of ATP systems. A problem's rating decreases when a higher fraction of the rating contributors, as described in Section 2.4, are able to solve the problem. Either the number of rating contributors stays the same but a new system that can solve the problem replaces a previous rating contributor, or the number of rating contributors increases with a new system that can solve the problem. In both cases the new system has improved the overall quality of the available ATP systems, which is progress in ATP. Note that a subsumed system cannot cause a rating change.

The average problem ratings have been computed for those problems that have been in all the TPTP versions being considered. For some SPCs the problems that were in v2.0.0, and hence all subsequent versions, are almost all easy. Thus there is no meaningful change in the average rating. For the SPCs that

SPC	1.00 in v2.0.0	v2.0.0 ↓ v2.1.0	v2.1.0 ↓ v2.2.0	v2.2.0 ↓ v2.3.0	v2.3.0 ↓ v2.4.0
SAT_EPR_CNF	53	2	50	0	0
SAT_RFO_NEQ_CNF	11	3	0	1	7
SAT_RFO_EQU_CNF_NUE	10	6	1	1	1
SAT_RFO_EQU_CNF_UEQ	11	7	1	1	1
SAT_EPR_FOF	0	0	0	0	0
SAT_RFO_FOF	1	0	0	0	1
THM_EPR_CNF	2	0	0	2	0
THM_RFO_NEQ_CNF_HRN	52	7	9	14	1
THM_RFO_NEQ_CNF_NHN	18	2	0	4	0
THM_RFO_SEQ_CNF_HRN	41	15	3	6	3
THM_RFO_SEQ_CNF_NHN	130	18	5	15	6
THM_RFO_PEQ_CNF_NUE	36	17	1	6	2
THM_RFO_PEQ_CNF_UEQ	54	16	1	18	0
THM_EPR_FOF	0	0	0	0	0
THM_RFO_NEQ_FOF	0	0	0	0	0
THM_RFO_EQU_FOF	1	0	0	1	0

Table 2. Numbers of problems solved for the first time

have some meaningful change in average rating, the change is shown as a function of time in Figure 4. There is a clear overall downward trend in the problem ratings, which means that the systems are getting better, i.e., there is progress in ATP. The most marked decrease is between TPTP versions 2.2.0 and v2.3.0. The analysis of Section 3.3 shows that only a small part of the progress can be attributed to hardware improvements.

In some SPCs there is an increased average rating between some TPTP versions. This is caused by the introduction of new rating contributors that could not solve problems that were solved by the existing rating contributors, and is to be expected in the experimental environment of ATP system development. Such fluctuations should thus not be interpreted as a deterioration of the state-of-the-art.

3.6 CASC Fixed Points

CASC is organized into divisions, which correspond closely to the SPCs. The divisions are `MIX` – mixed CNF real 1st order theorems ("mixed" means Horn and non-Horn problems, with or without equality, but not unit equality problems), `UEQ` – unit equality CNF real 1st order theorems, `SAT` – mixed CNF real 1st order non-theorems, and `FOF` – mixed FOF theorems. A winner is announced in each division of each CASC. For the last three CASCs (-16, -17, and -JC), the CASC organizers have entered the previous CASC's division winners into their divisions. The previous winners provide fixed points against which the new systems, using the same resources, can be judged.

Table 3 shows the performance of the previous CASC's winners in CASCs-16, -17, and -JC. With the exception of the `SAT` division of CASC-JC, the previous winners have been beaten by one or more of the new systems. These results

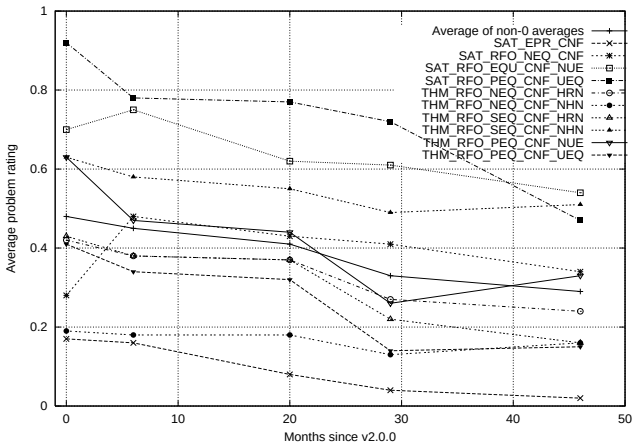


Fig. 4. Average problem ratings over time

indicate that there is progress in ATP between each CASC. It may be claimed that the different eligible and randomly selected problems of each CASC is the cause of a new system beating the previous winner, but the consistency with which previous winners are outperformed makes this claim most unlikely. Conversely, the addition of new problems to the TPTP each year, and the different selection of problems used in the competition, means that the improvements in the new systems cannot be attributed to simple tuning for the previous CASC's problems.

	Division winner Problems/Solved by winner/Previous winner (Place)			
	CASC-15	CASC-16	CASC-17	CASC-JC
`MIX`	Gandalf c-1.1	Vampire 0.0	E 0.6	VampireJC 2.0/ E-SETHEO csp01
	80/61/–	75/51/39 (4th)	75/57/37 (5th)	120/93/81 (4th)
`UEQ`	Waldmeister 798	Waldmeister 799	Waldmeister 600	Waldmeister 601
	30/30/–	30/30/19 (2nd)	30/30/29 (2nd)	90/69/69 (2nd)
`SAT`	SPASS 0.95T	OtterMACE 437	GandalfSat 1.0	GandalfSat 1.0
	30/22/–	30/16/9 (3rd)	30/25/21 (4th)	90/48/48 (1st)
`FOF`	SPASS 0.95T	SPASS 1.00T	VampireFOF 1.0	E-SETHEO csp01
	40/39/–	30/22/19 (3rd)	60/53/51 (2nd)	90/75/72 (2nd)

Table 3. Performance of previous CASC division winners

4 Conclusion

This paper presents quantitative measures that show progress in ATP, from mid-1997 to mid-2001. The measures are based on collected performance data from ATP systems, and from the results of the CADE ATP System Competitions. The performance data comes from testing ATP systems on TPTP problems, which

are divided into 16 Specialist Problem Classes. The performance data has been analyzed in five different ways, and for six of the SPCs the analyses consistently indicate significant progress in ATP. The six SPCs are the `THM_RFO_*_CNF_*` SPCs that represent the mainstream of ATP and ATP applications. There is also some evidence of progress in the other SPCs. The comparisons of ATP systems based on the CASC results similarly provide convincing evidence of progress in ATP, especially in the mainstream divisions.

A concern that has been discussed in the ATP community, and that has not been addressed in this paper, is that ATP systems are being tuned to get better at solving TPTP problems, but are not really increasing in deductive power. In one sense this concern may be viewed as somewhat self-contradictory; tuning is a way of increasing the deductive power of a system, and tuning for the TPTP is most likely to improve performance in the general sense. Anyway, an initial investigation into this claim was made at CASC-JC, where some problems previously unseen by the entrants were used. Table 4 shows, for each system, the fractions of old (in the TPTP before the competition, and thus available for system tuning), new, and all problems solved. Only VampireJC did not solve a higher fraction of new problems than old or all problems, which may be expected because VampireJC was explicitly tuned for the old TPTP problems that were predicted to be eligible for the MIX division. CHECK THIS CLAIM WITH ANDREI. The regular solution of new problems suggests that any tuning towards the existing TPTP problems is effective in general. A more conclusive experiment would be to run old and new versions of ATP systems on a larger set of unseen problems, and compare their performances. It is expected that the new versions would outperform the old versions, and this evidence would further support the conclusion that there is progress in ATP.

The conclusion that ATP is making progress sends out messages to users, researchers, and observers. Take heed ...

- To users: ATP research is steadily producing more powerful systems that can solve your problems.
- To researchers: The long hard effort is paying off.
- To funding bodies: Your money is being well spent, as support for ATP research is producing real results.

System	Old	New	All	System	Old	New	All
E-SETHEO	0.77	0.79	0.78	Gandalf	0.47	0.74	0.51
VampireJC	0.79	0.68	0.78	Otter	0.21	0.53	0.26
E 0.62	0.69	0.74	0.70	SCOTT	0.22	0.42	0.25
E 0.6	0.67	0.68	0.68	Bliksem	0.22	0.37	0.24
Vampire	0.62	0.68	0.63	DCTP	0.07	0.37	0.12
EP	0.60	0.63	0.61				

Table 4. Fractions of old, new, and all problems solved, in the CASC-JC MIX division

References

[1] L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic Paramodulation and Superposition. In Kapur D., editor, *Proceedings of the 11th International Conference on Automated Deduction*, number 607 in Lecture Notes in Artificial Intelligence, pages 462–476. Springer-Verlag, 1992.

[2] N. Dershowitz and L. Vigneron. Rewriting Home Page. http://rewriting.loria.fr/, 2000.

[3] M. Fuchs and G. Sutcliffe. Homogeneous Sets of ATP Problems. Technical Report TR-ARP-09-00, Automated Reasoning Project, Australian National University, Canberra, Australia, 2000.

[4] M. Fujita, J.K. Slaney, and F. Bennett. Automatic Generation of Some Results in Finite Algebra. In R. Bajcsy, editor, *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 52–57. Morgan Kaufmann, 1993.

[5] U. Furbach, B. Beckert, R. Hähnle, R. Letz, P. Baumgartner, U. Egly, W. Bible, S. Brüning, J. Otten, T. Rath, and T. Schaub. Tableau and Connection Calculi. In W. Bibel and P.H. Schmitt, editors, *Automated Deduction - A Basis for Applications, Volume 1: Foundations - Calculi and Methods*, pages 3–179. Kluwer, 1998.

[6] 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.

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

[8] D.E. Knuth and P.B. Bendix. Simple Word Problems in Universal Algebras. In Leech J., editor, *Computational Problems in Abstract Algebras*, pages 263–297. Pergamon Press, 1970.

[9] R.A. Kowalski. Predicate Logic as a Programming Language. In Rosenfeld J.L., editor, *Proceedings of the IFIP Congress* , pages 569–574. Elsevier Science, 1974.

[10] K. Kunen. Quasigroups, Loops, and Associative Laws. *Journal of Algebra*, 185:194–204, 1996.

[11] D.W. Loveland. A Simplified Format for the Model Elimination Theorem-Proving Procedure. *Journal of the ACM*, 16(3):349–363, 1969.

[12] D.W. Loveland. Automated Deduction - Looking Ahead. *AI Magazine*, 20(1):77–98, 1999.

[13] E.L. Lusk. Controlling Redundancy in Large Search Spaces: Argonne-Style Theorem Proving Through the Years. In Voronkov A., editor, *Proceedings of the 3rd International Conference on Logic Programming and Automated Reasoning* , number 624 in Lecture Notes in Artificial Intelligence. Springer-Verlag, 1992.

[14] W.W. McCune. Otter 3.0 Reference Manual and Guide. Technical Report ANL-94/6, Argonne National Laboratory, Argonne, USA, 1994.

[15] W.W. McCune. Solution of the Robbins Problem. *Journal of Automated Reasoning*, 19(3):263–276, 1997.

[16] W.W. McCune and R. Padmanabhan. *Automated Deduction in Equational Logic and Cubic Curves*, volume 1095 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 1996.

[17] L.J. Peter and R. Hull. *The Peter Principle*. Souvenir Press, 1969.

[18] I.V. Ramakrishnan, R. Sekar, and A. Voronkov. Term Indexing. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier Science, 1999.

[19] W. Reif. The KIV-approach to Software Verification. In M. Broy and S. Jähnichen, editors, *KORSO: Methods, Languages, and Tools for the Construction of Correct Software - Final Report*, number 1009 in Lecture Notes in Computer Science. 1995.

[20] G.A. Robinson and L. Wos. Paramodulation and Theorem Proving in First-Order Theories with Equality. *Machine Intelligence*, 4:135–150, 1969.

[21] J.A. Robinson. A Machine-Oriented Logic Based on the Resolution Principle. *Journal of the ACM*, 12(1):23–41, 1965.

[22] S. Schulz. System Abstract: E 0.3. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction*, number 1632 in Lecture Notes in Artificial Intelligence, pages 297–301. Springer-Verlag, 1999.

[23] J.K. Slaney. CADE-12 invited talk: The Crisis in Finite Mathematics: Automated Reasoning as Cause and Cure. 1994.

[24] M.E. Stickel. A Prolog Technology Theorem Prover: A New Exposition and Implementation in Prolog. Technical Report Technical Note 464, SRI International, Menlo Park, USA, 1989.

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

[26] G. Sutcliffe and D. Seyfang. Smart Selective Competition Parallelism ATP. In A. Kumar and I. Russell, editors, *Proceedings of the 12th Florida Artificial Intelligence Research Symposium*, pages 341–345. AAAI Press, 1999.

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

[28] G. Sutcliffe and C.B. Suttner. ATP System Results for the TPTP Problem Library. http://www.cs.jcu.edu.au/~tptp/TPTP/Results.html, 2000.

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

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

[31] A. Voronkov. The Anatomy of Vampire. *Journal of Automated Reasoning*, 15(2):237–265, 1995.

[32] C. Weidenbach, B. Afshordel, U. Brahm, C. Cohrs, T. Engel, E. Keen, C. Theobalt, and D. Tpoic. System Description: SPASS Version 1.0.0. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction*, number 1632 in Lecture Notes in Artificial Intelligence, pages 378–382. Springer-Verlag, 1999.

[33] L. Wos, R. Overbeek, and E. Lusk. Subsumption, a Sometimes Undervalued Procedure. Technical Report MCS-P93-0789, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, USA, 1993.

[34] L. Wos, G.A. Robinson, D.F. Carson, and L. Shalla. The Concept of Demodulation in Theorem Proving. *Journal of the ACM*, 14(4):698–709, 1967.