Evaluation of Automated Theorem Proving on the Mizar Mathematical Library

Josef Urban[1,⋆], Krystof Hoder[2], and Andrei Voronkov[2,⋆⋆]

[1] Radboud University, Nijmegen
[2] University of Manchester

Abstract. This paper investigates the strength of first-order automatic theorem provers (ATPs) in proving theorems and lemmas from the Mizar proof assistant's formal mathematical library. Several Mizar use-cases are described and evaluated, as well as various ATP systems and strategies. The new version of the leading Vampire ATP system is included in the evaluation, experiments with Mizar-specific strategy-selection are performed with E the prover, and the SInE axiom selection is evaluated on large Mizar problems with both E and Vampire. A rough mathematical division of the Mizar library is introduced, and the ATP performance is evaluated on it.

1 Introduction and Motivation

In the last five years there was a considerable increase of the use of fully automatic first-order theorem provers as assistants to interactive theorem provers (ITPs). A number of formal knowledge bases and core logics have been translated to first-order ATP formats such as TPTP.[1] For example, let us mention the related work on the Isabelle/Sledgehammer ATP link [MP09], and the export of the SUMO [PS07] and CYC [MJWD06] real-world formal knowledge bases.[2]

One of the main goals of the MPTP[3] project is to make the large Mizar Mathematical Library[4] (MML) accessible to ATP and AI experiments and techniques. The particular value of MML/MPTP in comparison to the above mentioned related projects is that this is a comparatively large library focused primarily on standard mathematics as done by mainstream mathematicians (using first-order logic and set theory as the foundations). The first-order setting allows a practically complete and reasonably efficient translation for first-order ATPs, which is harder to do for higher-order systems. The size of the library and its consistency on the symbol-naming and theorem-naming level also allows experimenting with

⋆ Supported by the NWO project "MathWiki a Web-based Collaborative Authoring Environment for Formal Proofs".
⋆⋆ Supported by an EPSRC grant.

[1] Thousands of Problems for Theorem Provers, see `www.tptp.org`
[2] `www.ontologyportal.org` and `www.cyc.com`
[3] Mizar Problems for Theorem Proving.
[4] `www.mizar.org`

K. Fukuda et al. (Eds.): ICMS 2010, LNCS 6327, pp. 155–166, 2010.

all kinds of "knowledge-based" ATP/AI techniques, which might be relevant for emulating the thinking of learned mathematicians, and bringing new insights to the fields of ATP and AI.

The first inclusion of the MML/MPTP problems in ATP benchmarks (the TPTP library) happened in 2006, when also the large-theory MPTP Challenge[5] was announced. Since then the CASC[6] Large Theory Batch (LTB) competition was introduced, and run already twice in 2008 and 2009. This influences the performance and tuning of existing ATP systems, and gives rise to new techniques and interesting metasystems.

The purpose of the current paper is to evaluate the progress made over the past five years in the area of reasoning in large formal mathematical theories, and particularly evaluate the strongest ATP systems and metasystems on sufficiently recent MML/MPTP and the mathematical subfields contained in it.

1.1 Recent Evolution of Mizar and MPTP

Despite its age, Mizar is a living and evolving system with a number of users around the world. Since the last published MPTP experiments done on MML version 938 (938 articles), a number of articles have been added to the library resulting in thousands of new "Mizar theorems"[7]. These range from a number of standard calculus results developing, e.g., the Riemann integral, to abstract algebra results like Sylow theorems [Ric07], to formalization of special fields like BCI/BCK algebras [Din07] to results from mathematical theory of social choice like Arrow's Impossibility Theorem [Wie07].

At least the following developments have been tried/done with the Mizar system between these two versions:

- The Mizar type system mechanisms (Horn-like mechanisms automatically inferring monadic adjectives about the objects of the set-theoretical universe) have been constantly strengthened, becoming one of the main automation tools in Mizar.
- Experiments have been done with strengthening the matching/unification mechanisms in the Mizar kernel module.
- Identifications (i.e., registered automated equalities applied implicitly by the system) have been introduced by Mizar and used in MML.
- Further elements of computer algebra have been introduced in the kernel module, to allow automated normalization and solving of systems of linear equations.

The development of MPTP has to reflect the Mizar/MML changes. Also, as a relatively young system, MPTP has a number of its own developments to do. Here is a short summary of the recent ones:

[5] `http://www.tptp.org/MPTPChallenge/`

[6] The CADE ATP System Competition, see `http://www.tptp.org/CASC/`

[7] We put Mizar theorems in quotes at least once to deliver the message that only very few of these "theorems" would be called a "theorem" by mathematicians. Large majority of these propositions are lemmas useful and re-usable for proving further results, and this property makes them "theorems" in the Mizar parlance.

- Probably the largest change is that initial methods for ATP-export of Mizar internal arithmetics have been implemented. This is a constant cat-and-mouse pursuit with the experiments done with computer algebra in the Mizar kernel,[8] however it is now possible to do ATP experiments over Mizar problems containing arithmetics. The export is correct, but not always complete.[9] However, as can be seen in Section 3, counter-satisfiability is detected only reasonably rarely in practice by ATPs.
- MPTP changes in ATP problem creation, accommodating the new developments in the Mizar type automations, and introduction of identifications.
- Changes making MPTP faster and more real-time, including:
- More advanced (graph-like) datastructures to speed-up the process of selecting necessary parts of the library for generating the ATP problems.
- Larger use of available Prolog indexing and the asserted database for various critical parts of the code.
- Instead of working always with the whole loaded MML, MPTP was refactored to allow working only with the (usually much smaller) part of the MML needed for the newly processed article. This is specifically required for the new ATP-for-Mizar (MizAR) service running now in real time at the RU Foundations' group server[10] [US10].

The summary of data from previous experiments with SPASS (version 2.1) and E (version 0.9) from 2005 on MML version 938 using MPTP 0.2 is given in Table 1 (see [Urb06] for details).

Table 1. Reproving of the theorems from non-numerical articles by MPTP 0.2 in 2005

description	proved	countersatisfiable	timeout or memory out	total
E 0.9	4309	0	8220	12529
SPASS 2.1	3850	0	8679	12529
together	4854	0	7675	12529

Note that these experiments have been done in 2005 only on "non-numerical" articles (containing 12529 theorems/problems), i.e., on Mizar articles guaranteed not to contain any arithmetical evaluations. The current experiments described in Section 3 are however performed on the whole MML, because a basic ATP-export of Mizar computer algebra is now available.

[8] This is the main reason why we choose a version of MML that is not completely recent at the time of writing this evaluation. The MML version 1011 that we choose for the evaluation here has now been sufficiently tested, and the ATP-export of Mizar internal arithmetics sufficiently debugged. It would be possible to experiment with a more recent MML version, however for a large-scale evaluation it is preferable to use a reasonable recent version for which MPTP is known to work well.

[9] This also really depends on the particular version of the Mizar kernel.

[10] `http://mws.cs.ru.nl/~mptp/MizAR.html`

2 Mizar Data, Experimental Setup

The experiments described in Section 3 are performed on three classes of data,[11] all coming from the proofs of all Mizar theorems from MML version 1011. There are 51424 theorems in this MML version. The classes differ by the average number of axioms (previous theorems and definitions from MML) included in the problems, coming from different Mizar use-cases. The classes and use-cases are as follows:

- *SMALL*: Problems with smallest number of included axioms. This use-case models a user who knows relatively well how a proof should proceed (what MML knowledge should roughly be used). In the HOL Light (established for its ITP/ATP inventions) terminology: MESON_TACTIC[12] . The average size of an MPTP problem in this class is 218 formulas. Many of these theorems have long Mizar proofs - tens to hundreds of lines - and can contain nontrivial mathematical ideas. See the listings at these web pages.[13]
- *ENVIRON*: Problems that include all axioms contained in article's environment (that is: articles imported by the current article). This use-case models Mizar authors who selected a particular combination of mathematical areas (previous articles) to base their articles on, and thus limited the Mizar knowledge to a smaller subset of MML. Inside this MML subset they however do not provide any additional guidance to the ATPs. Such problems can already be very large: their average size is 5830 formulas.
- *ALL*: Problems that include all of the Mizar knowledge available in the MML at the time of proving a particular theorem. This models users who do not want to limit their search to the articles imported in their environment, and provide no guidance to ATPs. The price for such intellectual laziness is obviously a large number of axioms in such ATP problems, the average size of a problem in this class is 40898 formulas.

All these three use-cases are interesting and relevant. As mentioned above, the *SMALL* case is used a lot in ITPs like HOL (Light) as a general method for solving a goal once the user feels that it is sufficiently simply derivable from other established premises. The Mizar system actually also works in a similar way (using a custom weak theorem prover for the "by" inference), however, the emphasis there is not on strength, but on capturing the notion of *obvious inference* [Dav81, Rud87]. Another advantage of the *SMALL* case is that the 218 average formulas (which means much less in a significant number of cases) can be reasonably attacked by existing standard resolution and tableaux techniques, and ATPs based on them, without introducing any novel techniques for dealing

[11] Available at `http://mws.cs.ru.nl/~mptp/mptp_1011/noint/`

[12] `http://www.cl.cam.ac.uk/~jrh13/hol-light/HTML/MESON_TAC.html`. Actually, MPTP does here more than MESON: it adds a lot of "background" formulas to the problem including knowledge used implicitly by Mizar (reflexivity of $\leq$, etc.).

[13] `http://mmlquery.mizar.org/mmlquery/fillin.php?filledfilename=mml-facts.mqt&argument=number+102`, and `http://www.cs.ru.nl/~freek/100/`

with a large number of axioms. Thus, for metasystems that combine custom axiom-selection methods with standard ATPs, the *SMALL* case can be thought of as a benchmark for the ATP component of such metasystems, i.e., telling how good the performance of the whole metasystem could be, if the axiom-selection component of the metasystem was perfect.

This is no longer true for the *ENVIRON* and *ALL* classes. As will be seen in Section 3, using standard ATP techniques on these problems is currently not productive, and axiom-preselection methods are necessary on the *ENVIRON* and *ALL* classes to make use of ATPs.

There are other possible classes of data and divisions of the *SMALL*, *ENVIRON* and *ALL* classes along various axes. A common objection to these three classes is that they are too hard: for example, the data for testing Isabelle/Sledgehammer come typically from goals that are easier than the "full Isabelle theorems". The answer is that using Mizar *simple justifications* ("by" steps – steps provable using the Mizar built-in limited checker [Wie00]) has with the development of MPTP and ATP methods over Mizar become too easy, and such data are no longer suitable as a Mizar/MPTP/ATP benchmark. The success rate of various combined ATP/AI methods on large pieces of "by" data is now around 99.9%, actually allowing for using such methods together with the GDV [Sut06] ATP-based verifier (enhanced to handle TPTP proofs with Jaskowski-like assumptions) to completely ATP-cross-verify large pieces of MML (see [US08] for details). Other classes of problems that could come to mind are:

- Internal Mizar sublemmas that serve to prove another theorem/lemma, but are not themselves "too easy" (i.e., are not proved by simple justification). Such sublemmas could be considered an easier dataset than theorems, but harder than the simple justifications.
- De-lemmatized theorems. This would be a dataset created from *SMALL*, where the references (other theorems) used to prove a theorem are (recursively, to some level of recursion) replaced by their own references, in the extreme case expanding them all the way to axioms and definitions. Such delemmatized theorems could be considered a harder dataset than the standard theorems.

The reason why it seems unnecessary to test also on such classes of data is that already the theorem dataset provides a variety of both easy and hard data. There are a number of Mizar theorems proved using a simple justification, and on the other hand, there are theorems (like ROLLE:1 in [KRS90] - Rolle's theorem) that take more than four hundred lines and a large number of references to prove, i.e., the amount of lemmatization varies greatly across various Mizar articles and with various Mizar authors.

The ATP success rates reported in Section 3 on the *SMALL* theorem dataset indicate that also from the practical "benchmark" point of having data that are not too easy and not always very hard, this dataset seem to work well with current off-the-shelf ATPs. Again, this is not yet true with the large *ENVIRON* and *ALL* datasets which, on the other hand, can be considered to be hard

Table 2. Evaluation of E, SPASS, and Vampire on all *SMALL* problems in 30s

description	proved	countersatisfiable	timeout or memory out	total
E 1.1-004	16191	4	35229	51424
SPASS 3.7	17550	12	33862	51424
Vampire 0.6	20109	0	31315	51424
together	22607	12	28817	51424

benchmarks for ATP/AI metasystems that complement standard ATP with systems for axiom selection.

Divisions along various further axes of these datasets are certainly possible. In section 4 we attempt to define a "reasonable" crude mathematical categorization of 80% of MML articles, and provide an initial evaluation across this division.

3 Experiments

3.1 Overall Evaluation on *SMALL* Problems

The large-scale experimental evaluation of the standard ATPs focuses on the *SMALL* class of problems (for the reasons mentioned above in Section 2). The three main evaluated ATPs are the latest versions of the SPASS [WDF+09] (version 3.7) system, the E prover [Sch02] (version 1.1-004 Balasun), and the Vampire [RV02] prover (version 0.6 - preliminary version for CASC-J5). SPASS and E are evaluated on the server of the Foundations group at Radboud University Nijmegen (RU), which is eight-core Intel Xeon E5520 2.27GHz with 8GB RAM and 8MB CPU cache. The time limit for the evaluations is 30s,[14] and the memory limit is 900MB for each problem. Vampire is evaluated on computers at the laboratory of the University of Manchester (UM), each of them being Intel Core2 Duo E7300 2.66GHz PC with 1G RAM and 3MB cache. The time limits used are again 30s. The relative performances of the two hardware platforms have been compared by evaluation on common ATP problems. The UM platform turns out to be approximately 10% faster.[15] No parallelization is used, each problem is always run serially. The Table 2 shows the results. Note that there are some counter-satisfiable problems (very likely arithmetical) however their number is insignificant. It turns out that E, SPASS, and Vampire can in 30s together (that is: if run in parallel) solve 44% of the MML *SMALL* problems.

In Table 3, the results of SPASS used in (the incomplete) SOS mode are shown, and compared to the results of standard SPASS. This is done on randomly selected 1000 *SMALL* problems. The number of countersatisfiable results is not relevant for SPASS-SOS however: it is incorrect when SPASS is used in

[14] The experience from previous experiments with E and SPASS is that only a small fraction of problems is solved after 30s. This is obviously different with strategy-scheduling ATP systems like Vampire.

[15] This difference obviously does not translate to 10% more solved problems, however particularly with strategy-scheduling ATP like Vampire, it is quite significant.

Table 3. Comparison of SPASS-SOS and SPASS on 1000 *SMALL* problems in 30s

description	proved	countersatisfiable	timeout or memory out	total
SPASS 3.7-SOS	292	55	653	1000
SPASS 3.7	345	0	655	1000
together	377	0	623	1000

SOS mode together with ordering-based ATP techniques. SPASS-SOS turns out to be significantly worse than SPASS on the same data (proving 345 of these 1000 problems), however the SOS strategy is reasonably complementary to the standard one: together, the both methods of running SPASS solve 377 problems from this dataset (Vampire solves 39% of these problems).

3.2 Overall Evaluation on *ENVIRON* and *ALL* Problems, SInE

To a smaller extent (the above mentioned dataset of 1000 problems) we also evaluate E and Vampire on the large *ENVIRON* and *ALL* problems, and focus on evaluation of a new heuristic axiom pre-selector SInE.[16]

The SInE selection algorithm (yet to be published) uses a syntactic approach based on symbol presence in formulas of the problem. SInE builds a relation D (as in "*D*efines") between symbols and axioms. The D-relation represents the fact that for a symbol there are some axioms that "give it its meaning." In order to construct the D-relation, for each symbol the number of axioms in which it appears is computed, this number is called the *generality index* of the symbol. Then each axiom is put into the D-relation with the least general symbol it contains.[17] After the relation is built, the actual axiom selection starts. All problem-specific formulas are selected, and in each iteration the selection is extended by all included formulas that are D-related to any of the symbols used in already selected formulas.[18] The iterating is done until the set of selected axioms becomes stable. The stable set of formulas is then passed to a theorem prover.

This standard fixpoint algorithm however tends to give too many axioms on MML problems.[19] To deal with this, we have introduced a depth limit parameter — a limit on the number of selection-extending iterations. With the depth limit equal to one ("-d1" parameter), for example, only the included axioms D-related to symbols in the problem-specific axioms are included.

[16] SUMO Inference Engine - originally developed by the second author for reasoning in the large SUMO knowledge base.

[17] If there are more symbols with lowest generality index, axiom is put in relation with all of them.

[18] Note that the current implementation relies on reasonable presentation of large-theory problems using TPTP-includes. This can be easily changed if needed.

[19] The structure of MML significantly differs from KBs like SUMO and CYC. There are many more nontrivial theorems in MML, while SUMO and CYC contain a lot of definitions.

Table 4. Evaluation of Vampire and E with SInE(-d1) on random 1000 *ENVIRON* and *ALL* MML.1011 problems in 30s

problems	Vampire+SInE	Vampire+SInE(-d1)	E	E+SInE	E+SInE(-d1)
ENVIRON	181	205	65	135	161
ALL	84	141	21	64	153

The Table 4 presents the results of evaluation of E and Vampire on 1000 *ENVIRON* and *ALL* problems run with 30s time limit on the UM PCs.[20] Note that the combination of Vampire and SInE run with -d1 solves 205 of the *ENVIRON* problems, and the combination of E and SInE with -d1 solves 153 of the *ALL* problems. This is a very good performance on problems with average size of 5830 formulas resp. 40898 formulas. E in this mode solves about half of the *SMALL* versions of the problems. This is very likely also due to improved E heuristics for dealing with large problems. The SInE preprocessing time needs to be added to the 30s given to E prover. For the *ENVIRON* problems this time is on average 1s, and for the *ALL* problems, this is on average 4s.

3.3 Evaluation of Strategy Selection and Combination

Design, selection and combination of sufficiently orthogonal useful ATP strategies has been for some time a well known technique significantly raising performance of ATPs. Both Vampire and E use strategy selection and machine learning on the TPTP library to select a collection of useful strategies. However, they use the found strategies in different ways. Vampire selects sequences of strategies, while E selects one "best" strategy when run in auto-mode. The effect of strategy combination in Vampire can be estimated by comparing Vampire's performance in shorter and longer times (here in 5s and 30s on a random set of 1000 *SMALL* problems). For E and SPASS (running a single strategy depending on the problem) this difference is relatively small. Only 41 problems out of 345 solved in 30s by SPASS are solved in time longer than 5s, which is approximately 13% increase. For Vampire, this improvement is much more significant: out of 384 problems solved in 30s, only 310 are solved in 5s, which gives a 24% increase. This significant difference in comparison to SPASS is due to Vampire running not only for a longer time, but also switching to a different strategy during proof-search.

This clue leads to a strategy-evaluation experiment done with E again on this random set of 1000 *SMALL* problems: Each of the 196 E strategies predefined by the E developer Stephan Schulz is tested on this set of problems with a 5s time limit. It turns out that the strategy selected by E as potentially the best solves 310 problems with the 30 seconds time limit, while the strategy that turns out to be the best in reality solves 317 problems with a 5s time limit.

This clearly demonstrates the potential of domain-based ATP strategy-tuning. All the 196 E strategies together solve 386 of the 1000 problems. This confirms

[20] The new version of Vampire uses SInE automatically, thus we do not provide data for Vampire without SInE.

a previous conjecture by the first author and the E developer that E with a suitably-tuned strategy combination mode will be considerably stronger. It also demonstrates that strategy combination is more robust on new problems. The strategies of E are defined using smaller building blocks and a special ("programming") language using a number of parameters. Given the performance potential gained by this strategy-tuning, it would be very interesting (and feasible) AI experiment to try to invent new E strategies by AI methods for (e.g., genetic) parameter-optimization/programming. Particularly on a large knowledge base like MML, this might lead to some surprising ATP-strategy inventions, in the same way as combining ATPs with learning on MML sometimes finds completely novel and significantly shorter proofs than those written by the Mizar authors.

4 Evaluation of ATPs on Different Mathematical Domains in MML

Formal mathematics can be clustered according to many aspects, and just thinking about organizing mathematics can lead to all kinds of theoretical investigations in Foundations, Category Theory, but also into practical investigations with various classification schemes like Mathematics Subject Classification 2000[21], and also into very pragmatic classifications based on the shape of the formal theories, important for performing automated reasoning in various domains[22].

For the purpose of ATP evaluation in this paper we attempt a manual division of the MML articles into (currently) seventeen subdomains of main MML developments. This is motivated by the curiosity to verify experimentally various intuitions developed over the years in the ATP field, like "algebra is ATP-easier than calculus". As mentioned above, there can be many approaches to this, and for example a proper MathSC2000 classification would certainly serve even better than the coarse-grained division started by us, however detailed classification of more than 1000 formal articles requires a non-trivial amount of work, and making fine-grained decisions would be beyond our resources. The (evolving) division can be viewed at our web page[23]. The categories and the numbers of articles in each category are shown in Table 5. The categorization now includes 804 articles out of the total 1011.

To the extent to which MML is approximation of "real mathematics", and to the extent to which this rough categorization is valid, these seventeen large classes of problems (with the three different problem sizes coming from the different use-cases described in Section 2) express a large mathematically-oriented (and particularly Mizar/MPTP-oriented) ATP benchmark.

[21] See `http://wiki.mizar.org/twiki/bin/view/Mizar/MathematicsSubjectClassification` for a so far 25%-successful attempt to classify MML according to MathSC2000.

[22] For example, the strategy-selection tuning typically works by defining suitable clustering based on the term and formula structure of the problems.

[23] `http://github.com/JUrban/MPTP2/raw/master/MMLdivision.1011`

Table 5. Categorization of MML 1011, 804 articles covered, SPASS, E, Vampire, and overall success rates on the categories

description	articles	probs	S	E	V	All	S %	E %	V %	All %
Algebra	50	2798	1182	1086	1314	1481	42.24	38.81	46.96	52.93
Algebraic Topology	5	215	52	50	89	94	24.19	23.26	41.4	43.72
Arithmetic, Number theory	70	4095	1587	1515	1741	1943	38.75	37	42.52	47.45
Calculus (real, complex)	54	3255	585	538	651	783	17.97	16.53	20	24.06
Category theory	21	1023	298	305	406	455	29.13	29.81	39.69	44.48
Computers, Algorithms	81	3809	971	932	1120	1304	25.49	24.47	29.4	34.23
Functional analysis	30	1320	395	330	445	507	29.92	25	33.71	38.41
General Topology	65	3191	1199	1115	1441	1594	37.57	34.94	45.16	49.95
Geometry	36	1593	666	659	806	876	41.81	41.37	50.6	54.99
Graph theory,Finite structs	43	2756	1186	1094	1331	1455	43.03	39.7	48.29	52.79
Lattices	50	2434	707	570	764	917	29.05	23.42	31.39	37.67
Linear Algebra	32	1752	496	493	630	700	28.31	28.14	35.96	39.95
Logic, Model theory	52	2832	1042	1084	1196	1369	36.79	38.28	42.23	48.34
Probability and Measure	23	1123	348	274	448	489	30.99	24.4	39.89	43.54
Real plane,Euclidean spaces	84	4555	1018	897	1290	1439	22.35	19.69	28.32	31.59
Set Theory	74	4060	2412	2278	2570	2735	59.41	56.11	63.3	67.36
Universal Algebra	34	1093	391	372	434	502	35.77	34.03	39.71	45.93
together	804	41904	14535	13592	16676	18643	34.69	32.44	39.8	44.49

The table indeed seems to confirm quite convincingly the "algebra is ATP-easier than calculus" theory. The set-theoretical domain outperforms even the algebraic and is the most ATP-friendly. This is quite likely because of two related factors:

- Set theoretical articles belong to the more basic ones, not much previous implicit knowledge is included in the articles and their size is thus likely smaller, making them easier for ATPs.
- As the needed implicit knowledge (encoding e.g. the type system) gets more involved in more advanced areas like calculus, the ATP-emulation of these Mizar type mechanisms becomes more costly, and the ATP performance suffers.

There are a number of ways that these data can be further analyzed, providing useful feedback to the MPTP algorithms and also to ATP (meta) systems.

5 Conclusions, Future Work

The most important message of this evaluation is that a combination of three recent ATP systems can solve 44% of the MML theorems coming from many different parts of mathematics, regardless of how much computer algebra is done in them. The leading Vampire ATP system alone can solve 39%. Another important message is that off-the-shelf ATPs are still weak in large theories, however fast axiom-selection heuristics like SInE can help them and improve this state very significantly. Other one-problem-at-atime large-theory heuristic methods similar

to SInE are used for axiom pruning in Isabelle/Sledgehammer [MP09], and also for axiom ordering in the SRASS system [SP07]. If these other methods could be run easily on arbitrary and large TPTP problems, it would be interesting to evaluate and compare them on our benchmarks, or at least in the CASC LTB competition which includes a small selection of the older MPTP problems in its MZR category. The problem of axiom selection in large theories and the possible gains from good solutions seem to be sufficiently important to warrant such benchmarking and further research in this field, possibly leading also to smarter clause-selection algorithms implemented directly inside ATPs.

In this evaluation, axiom-selection methods that use transfer of knowledge between problems (machine learning) like MaLARea [USPV08] are not considered. Methods using learning are very interesting and quite novel in the ATP context, and we are currently investigating various suitable methods for machine learning , characterizations of problems and proofs, and also suitable combinations with strategy-selection and with other axiom-selection methods working in the one-problem-at-a-time setting like SInE. We also plan to do a thorough strategy evaluation of a much larger set of strategies available with Vampire, and their Mizar/MPTP-oriented tuning similar to the current CASC-tuning of Vampire, possibly again with adding machine learning technology.

An important future work is translation of ATP proofs to a presentable ITP format. The (technically certainly admirable) solution used by Isabelle/ Sledgehammer (and obviously also of HOL Light) is inclusion of a reasonably strong ATP system directly into their cores. This is however against the philosophy of readable proofs and "obvious inferences" of Mizar, and with strong external ATPs also potentially causing all kinds of other problems: not all proofs can be internalized, and the hard internalized proof s make the library refactoring slow and fragile.[24] With the growing strength of ATPs, a proper human-readable presentation of ATP proofs is a more and more pressing (and also very interesting) AI task. Some previous work in this direction has been done in the context of the Omega and ILF systems. A recent initial effort in this direction is described in [VSU10].

References

[Dav81] Davis, M.: Obvious logical inferences. In: Hayes, P.J. (ed.) IJCAI, pp. 530–531. William Kaufmann, San Francisco (1981)

[Din07] Ding, Y.: Several classes of BCI-algebras and their properties. Formalized Mathematics 15(1), 1–9 (2007)

[KRS90] Kotowicz, J., Raczkowski, K., Sadowski, P.: Average value theorems for real functions of one variable. Formalized Mathematics 1(4), 803–805 (1990)

[24] Note that this philosophy is no longer just Mizar's: the Math Components project targeted at the large formalization of Feit-Thompson theorem in Coq is avoiding Coq mechanisms that keep "too much automation" inside proofs for very similar reasons as Mizar does.

[MJWD06] Matuszek, C., Cabral, J., Witbrock, M., DeOliveira, J.: An Introduction to the Syntax and Content of Cyc. In: Baral, C. (ed.) Proceedings of the 2006 AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering, pp. 44–49 (2006)

[MP09] Meng, J., Paulson, L.C.: Lightweight relevance filtering for machine-generated resolution problems. J. Applied Logic 7(1), 41–57 (2009)

[PS07] Pease, A., Sutcliffe, G.: First Order Reasoning on a Large Ontology. In: Urban, J., Sutcliffe, G., Schulz, S. (eds.) Proceedings of the CADE-21 Workshop on Empirically Successful Automated Reasoning in Large Theories (2007)

[Ric07] Riccardi, M.: The Sylow theorems. Formalized Mathematics 15(3), 159–165 (2007)

[Rud87] Rudnicki, P.: Obvious inferences. J. Autom. Reasoning 3(4), 383–393 (1987)

[RV02] Riazanov, A., Voronkov, A.: The design and implementation of VAMPIRE. Journal of AI Communications 15(2-3), 91–110 (2002)

[Sch02] Schulz, S.: E – a brainiac theorem prover. Journal of AI Communications 15(2-3), 111–126 (2002)

[SP07] Sutcliffe, G., Puzis, Y.: SRASS - a semantic relevance axiom selection system. In: Pfenning, F. (ed.) CADE 2007. LNCS (LNAI), vol. 4603, pp. 295–310. Springer, Heidelberg (2007)

[Sut06] Sutcliffe, G.: Semantic Derivation Verification. International Journal on Artificial Intelligence Tools 15(6), 1053–1070 (2006)

[Urb06] Urban, J.: MPTP 0.2: Design, implementation, and initial experiments. J. Autom. Reasoning 37(1-2), 21–43 (2006)

[US08] Urban, J., Sutcliffe, G.: ATP-based cross-verification of Mizar proofs: Method, systems, and first experiments. Mathematics in Computer Science 2(2), 231–251 (2008)

[US10] Urban, J., Sutcliffe, G.: Automated reasoning and presentation support for formalizing mathematics in Mizar. In: Autexier, S., Calmet, J., Delahaye, D., Ion, P.D.F., Rideau, L., Rioboo, R., Sexton, A.P. (eds.) AISC 2010. LNCS (LNAI), vol. 6167, pp. 132–146. Springer, Heidelberg (2010)

[USPV08] Urban, J., Sutcliffe, G., Pudlak, P., Vyskocil, J.: MaLARea SG1: Machine Learner for Automated Reasoning with Semantic Guidance. In: Armando, A., Baumgartner, P., Dowek, G. (eds.) IJCAR 2008. LNCS (LNAI), vol. 5195, pp. 441–456. Springer, Heidelberg (2008)

[VSU10] Vyskocil, J., Stanovsky, D., Urban, J.: Automated proof shortening by invention of new definitions. In: LPAR 2010. LNCS (LNAI). Springer, Heidelberg (to appear 2010)

[WDF+09] Weidenbach, C., Dimova, D., Fietzke, A., Kumar, R., Suda, M., Wischnewski, P.: SPASS version 3.5. In: Schmidt, R.A. (ed.) Automated Deduction – CADE-22. LNCS, vol. 5663, pp. 140–145. Springer, Heidelberg (2009)

[Wie00] Wiedijk, F.: CHECKER - notes on the basic inference step in Mizar (2000), `http://www.cs.kun.nl/~freek/mizar/by.dvi`

[Wie07] Wiedijk, F.: Arrow's impossibility theorem. Formalized Mathematics 15(4), 171–174 (2007)