

# Theorem Proving in Large Formal Mathematics as an Emerging AI Field

Josef Urban<sup>1,\*</sup> and Jiří Vyskočil<sup>2,\*\*</sup>

<sup>1</sup> Radboud University Nijmegen, The Netherlands

<sup>2</sup> Czech Technical University

**Abstract.** In the recent years, we have linked a large corpus of formal mathematics with automated theorem proving (ATP) tools, and started to develop combined AI/ATP systems working in this setting. In this paper we first relate this project to the earlier large-scale automated developments done by Quaife with McCune’s Otter system, and to the discussions of the QED project about formalizing a significant part of mathematics. Then we summarize our adventure so far, argue that the QED dreams were right in anticipating the creation of a very interesting semantic AI field, and discuss its further research directions.

## 1 OTTER and QED

Twenty years ago, in 1992, Art Quaife’s book *Automated Development of Fundamental Mathematical Theories* [30] was published. In the conclusion to his JAR paper [29] about the development of set theory Quaife cites Hilbert’s “*No one shall be able to drive us from the paradise that Cantor created for us*”, and makes the following bold prediction:

The time will come when such crushers as Riemann’s hypothesis and Goldbach’s conjecture will be fair game for automated reasoning programs. For those of us who arrange to stick around, endless fun awaits us in the automated development and eventual enrichment of the corpus of mathematics.

Quaife’s experiments were done using an ATP system that has left so far perhaps the greatest influence on the field of Automated Theorem Proving: Bill McCune’s Otter. Bill McCune’s opinion on using Otter and similar Automated Reasoning methods for general mathematics was much more sober. The Otter manual [22] (right before acknowledging Quaife’s work) states:

Some of the first applications that come to mind when one hears “automated theorem proving” are number theory, calculus, and plane geometry, because these are some of the first areas in which math students try to

---

\* Supported by The Netherlands Organization for Scientific Research (NWO) grants *Knowledge-based Automated Reasoning* and *MathWiki*.

\*\* Supported by the Czech institutional grant MSM 6840770038.

prove theorems. Unfortunately, OTTER cannot do much in these areas: interesting number theory problems usually require induction, interesting calculus and analysis problems usually require higher-order functions, and the first-order axiomatizations of geometry are not practical.

Yet, Bill McCune was also a part of the QED<sup>1</sup> discussions about making a significant part of mathematics computer understandable, verified, and available for a number of applications. These discussions started around 1993, and gained considerable attention of the theorem proving and formalization communities, leading to the publication of the anonymous QED manifesto [1] and two QED workshops in 1994 and 1995. And ATP systems such as SPASS [50], E [33], and Vampire [32] based on the ideas that were first developed in Otter have now been really used for several years to prove lemmas in general mathematical developments in the large ATP-translated libraries of Mizar [13] and Isabelle [25].

This paper summarizes our experience so far with the QED-inspired project of developing automated reasoning methods for large general computer understandable mathematics, particularly in the large Mizar Mathematical Library.<sup>2</sup> A bit as Art Quaipe did, we believe (and try to argue below) that automated reasoning in general mathematics is one of the most exciting research fields, where a number of new and interesting topics for general AI research emerge today. We hope that the paper might be of some interest to those QED-dreamers who remember the great minds of the recently deceased Bill McCune, John McCarthy, and N.G. de Bruijn.

## 2 Why Link Large Formal Mathematics with AI/ATP Methods?

Formalization of Mathematics and Automated Theorem Proving are often considered to be two subfields of the field of Automated Reasoning. In the former, the focus is on developing methods for (human-assisted) computer verification of more and more complex parts of mathematics, while in ATP, the focus is on developing stronger and stronger methods for finding proofs automatically. The QED Manifesto has the following conservative opinion about the usefulness of automated theorem proving (and other AI) methods to a QED-like project:

It is the view of some of us that many people who could have easily contributed to project QED have been distracted away by the enticing lure of AI or AR. It can be agreed that the grand visions of AI or AR are much more interesting than a completed QED system while still believing that there is great aesthetic, philosophical, scientific, educational, and technological value in the construction of the QED system, regardless of whether its construction is or is not largely done ‘by hand’ or largely automatically.

---

<sup>1</sup> <http://www.rbjones.com/rbjpub/logic/qedres00.htm>

<sup>2</sup> [www.mizar.org](http://www.mizar.org)

Our opinion is that formalization and automation are two sides of the same coin. There are three kinds of benefits in linking formal proof assistants like Mizar and their libraries with the Automated Reasoning technology and particularly ATPs:

1. *The obvious benefits for the proof assistants and their libraries.* Automated Reasoning and AI methods can provide a number of tools and strong methods that can assist the formalization, provide advanced search and hint functions, and prove lemmas and theorems (semi-)automatically. The QED Manifesto says:

The QED system we imagine will provide a means by which mathematicians and scientists can scan the entirety of mathematical knowledge for relevant results and, using tools of the QED system, build upon such results with reliability and confidence but without the need for minute comprehension of the details or even the ultimate foundations of the parts of the system upon which they build

2. *The (a bit less obvious) benefits for the field of Automated Reasoning.* Research in automated reasoning over very large formal libraries is painfully theoretical (and practically useless) until such libraries are really available for experiments. Mathematicians (and scientists, and other human “reasoners”) typically know a lot of things about the domains of discourse, and use the knowledge in many ways that include many heuristic methods. It thus seems unrealistic (and limiting) to develop the automated reasoning tools solely for problems that contain only a few axioms, make little use of previously accumulated knowledge, and do not attempt to further accumulate and organize the body of knowledge. In his 1996 review of Quaife’s book, Desmond Fearnley-Sander says:

The real work in proving a deep theorem lies in the development of the theory that it belongs to and its relationships to other theories, the design of definitions and axioms, the selection of good inference rules, and the recognition and proof of more basic theorems. Currently, no resolution-based program, when faced with the stark problem of proving a hard theorem, can do all this. That is not surprising. No person can either. Remarks about standing on the shoulders of giants are not just false modesty...

3. *The benefits for the field of general Artificial Intelligence.* These benefits are perhaps the least mentioned ones, perhaps due to the frequent feeling of too many unfulfilled promises from general AI. However to the authors they appear to be the strongest long-term motivation for this kind of work. In short, the AI fields of *deductive reasoning* and *inductive reasoning* (represented by machine learning, data mining, knowledge discovery in databases, etc.) have so far benefited relatively little from each other’s progress. This is an obvious deficiency in comparison with the human mind, which can both inductively suggest new ideas and problem solutions based on analogy, memory, statistical evidence, etc., and also confirm, adjust, and even significantly modify these ideas and problem

solutions by deductive reasoning and explanation, based on the understanding of the world. Repositories of “human thought” that are both large (and thus allow inductive methods), and have precise and deep semantics (and thus allow deduction) should be a very useful component for cross-fertilization of these two fields. QED-like large formal mathematical libraries are currently the closest approximation to such a computer-understandable repository of “human thought” usable for these purposes. To be really usable, the libraries however again have to be presented in a form that is easy to understand for existing automated reasoning tools. The Fearnley-Sander’s quote started above continues as:

... Great theorems require great theories and theories do not, it seems, emerge from thin air. Their creation requires sweat, knowledge, imagination, genius, collaboration and time. As yet there is not much serious collaboration of machines with one another, and we are only just beginning to see real symbiosis between people and machines in the exercise of rationality.

### 3 Why Mizar?

Mizar is a language for formalization of mathematics and a proof checker for the language, developed since 1974 [19] by the Mizar team led by Andrzej Trybulec. This system was chosen by the first author for experiments with automated reasoning tools because of its focus on building the large formal Mizar Mathematical Library (MML). This formalization effort was begun in 1989 by the Mizar team, and its main purpose is to verify a large body of mainstream mathematics in a way that is close and easily understandable to mathematicians, allowing them to build on this library with proofs from more and more advanced mathematical fields. In 2012 the library consisted of over 1100 formal articles containing more than 50,000 proved theorems and 10,000 definitions. The QED discussions often used Mizar and its library as a prototypical example for the project. The particular formalization goals have influenced:

- the choice of a relatively human-oriented formal language in Mizar
- the choice of the declarative Mizar proof style (Jaskowski-style natural deduction [17])
- the choice of first-order logic and set theory as unified common foundations for the whole library
- the focus on developing and using just one human-obvious first-order justification rule in Mizar
- and the focus on making the large library interconnected, usable for more advanced formalizations, and using consistent notation.

There have always been other systems and projects that are similar to Mizar in some of the above mentioned aspects. Building large and advanced formal libraries seems to be more and more common today in systems like Isabelle, Coq [7], and HOL Light [16]. An example of a recent major formalization project

(in HOL Light) is Flyspeck [15], which required the formal verification of thousands of lemmas in general mathematics. In the work that is described here, Mizar thus should be considered as a suitable particular choice of a system for formalization of mathematics which uses relatively common and accessible foundations, and produces a large formal library written in a relatively simple and easy-to-understand style. Some of the systems described below actually already work also with other than Mizar data: for example, MaLAREa [41,48] has already been successfully used for reasoning over problems from the large formal SUMO ontology [24], and for experiments with Isabelle/Sledgehammer problems [23].

## 4 MPTP: Translating Mizar for Automated Reasoning Tools

The Mizar’s translation (MPTP - Mizar Problems for Theorem Proving) to pure first-order logic is described in detail in [38,39,42]. The translation has to deal with a number of Mizar extensions and practical issues related to the Mizar implementation, implementations of first-order ATP systems, and the most frequent uses of the translation system.

The first version (published in early 2003<sup>3</sup>) has been used for initial exploration of the usability of ATP systems on the Mizar Mathematical Library (MML). The first important number obtained was the 41% success rate of ATP re-proving of about 30000 MML theorems from selected Mizar theorems and definitions (precisely: other Mizar theorems and definitions mentioned in the Mizar proofs) taken from corresponding MML proofs.

No previous evidence about the feasibility and usefulness of ATP methods on a very large library like MML was available prior to the experiments done with MPTP 0.1<sup>4</sup>, sometimes leading to overly pessimistic views on such a project. Therefore the goal of this first version was to relatively quickly achieve a “mostly-correctly” translated version of the whole MML that would allow us to measure and assess the potential of ATP methods for this large library. Many shortcuts and simplifications were therefore taken in this first MPTP version, for example direct encoding in the DFG [14] syntax used by the SPASS system, no proof export, incomplete export of some relatively rare constructs (*structure* types and *abstract terms*), etc.

Many of these simplifications however made further experiments with MPTP difficult or impossible, and also made the 41% success rate uncertain. The lack of proof structure prevented measurements of ATP success rate on all internal proof lemmas, and experiments with unfolding lemmas with their own proofs. Additionally, even if only several abstract terms were translated incorrectly, during such proof unfoldings they could spread much wider. Experiments like finding

<sup>3</sup> <http://mizar.uwb.edu.pl/forum/archive/0303/msg00004.html>

<sup>4</sup> A lot of work on MPTP has been inspired by the previous work done in the ILF project [12] on importing Mizar. However it seemed that the project had stopped before it could finish the export of the whole MML to ATP problems and provide some initial overall statistics of ATP success rate on MML.

new proofs, and cross-verification of Mizar proofs (described below) would suffer from constant doubt about the possible amount of error caused by the incorrectly translated parts of Mizar, and debugging would be very hard.

Therefore, after the encouraging initial experiments, a new version of MPTP started to be developed in 2005, requiring first a substantial re-implementation [40] of Mizar interfaces. This version consists of two layers (Mizar-extended TPTP format processed in Prolog, and a Mizar XML format) that are sufficiently flexible and have allowed a number of gradual additions of various functions over the past years. The experiments described below are typically done on this version (and its extensions).

## 5 Experiments and Projects Based on the MPTP

MPTP has so far been used for

- experiments with re-proving Mizar theorems and simple lemmas by ATPs from the theorems and definitions used in the corresponding Mizar proofs,
- experiments with automated proving of Mizar theorems with the whole Mizar library, i.e. the necessary axioms are selected automatically from MML,
- finding new ATP proofs that are simpler than the original Mizar proofs,
- ATP-based cross-verification of the Mizar proofs,
- ATP-based explanation of Mizar atomic inferences,
- inclusion of Mizar problems in the TPTP [37] problem library, and unified web presentation of Mizar together with the corresponding TPTP problems,
- creation of the MPTP \$100 Challenges<sup>5</sup> for reasoning in large theories in 2006, creation of the MZR category of the CASC Large Theory Batch (LTB) competition [36] in 2008, and creation of the MPTP2078 [2] benchmark<sup>6</sup> in 2011 used for the CASC@Turing<sup>7</sup> competition in 2012,
- testbed for AI systems like MaLAREa and MaLeCoP [49] targeted at reasoning in large theories and combining inductive techniques like machine learning with deductive reasoning.

### 5.1 Re-proving Experiments

As mentioned in Section 4, the initial large-scale experiment done with MPTP 0.1 indicated that 41% of the Mizar proofs can be automatically found by ATPs, if the users provide as axioms to the ATPs the same theorems and definitions which are used in the Mizar proofs, plus the corresponding background formulas (formulas implicitly used by Mizar, for example to implement type hierarchies). As already mentioned, this number was far from certain, e.g., out of the 27449 problems tried, 625 were shown to be counter-satisfiable (i.e., a model of the

<sup>5</sup> [www.tptp.org/MPTPChallenge](http://www.tptp.org/MPTPChallenge)

<sup>6</sup> <http://wiki.mizar.org/twiki/bin/view/Mizar/MpTP2078>

<sup>7</sup> <http://www.cs.miami.edu/~tptp/CASC/J6/Design.html>

negated conjecture and axioms exists) in a relatively low time limit given to SPASS (pointing to various oversimplifications taken in MPTP 0.1). This experiment was therefore repeated with MPTP 0.2, however only with 12529 problems that come from articles that do not use internal arithmetical evaluations done by Mizar. These evaluations were not handled by MPTP 0.2 at the time of conducting these experiments, being the last (known) part of Mizar that could be blamed for possible ATP incompleteness. The E prover version 0.9 and SPASS version 2.1 were used for this experiment, with 20s time limit (due to limited resources). The results (reported in [42]) are given in Table 1. 39% of the 12529 theorems were proved by either SPASS or E, and no counter-satisfiability was found.

**Table 1.** Re-proving MPTP 0.2 theorems from non-numerical articles in 2005

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

These results have thus to a large extent confirmed the optimistic outlook of the first measurement in MPTP 0.1. In later experiments, this ATP performance has been steadily going up, see Table 2 for results from 2007 run with 60s time limit. This is a result of better pruning of redundant axioms in MPTP, and also of ATP development, which obviously was influenced by the inclusion of MPTP problems in the TPTP library, forming a significant part of the FOF problems in the CASC competition since 2006. The newer versions of E and SPASS solved in this increased time limit together 6500 problems, i.e. 52% of them all. With addition of Vampire and its customized Fampire<sup>8</sup> version (which alone solves 51% of the problems), the combined success rate went up to 7694 of these problems, i.e. to 61%. The caveat is that the methods for dealing with arithmetic are becoming stronger and stronger in Mizar. The MPTP problem creation for problems containing arithmetic is thus currently quite crude, and the ATP success rate on such problems will likely be significantly lower than on the nonarithmetical ones. Fortunately, various approaches have been recently started to endow theorem provers with better capabilities for arithmetic [28,18,4,8].

## 5.2 Finding New Proofs and the AI Aspects

MPTP 0.2 was also used to try to prove Mizar theorems without looking at their Mizar proofs, i.e., the choice of premises for each theorem was done automatically, and all previously proved theorems were eligible. Because giving ATPs thousands of axioms was usually hopeless in 2005, the axiom selection was done

<sup>8</sup> Fampire is a combination of Vampire with the FLOTTER [26] clausifier.

**Table 2.** Re-proving MPTP 0.2 theorems from non-numerical articles in 2007

description	proved	counter-satisfiable	timeout or memory out	total
E 0.999	5661	0	6868	12529
SPASS 2.2	5775	0	6754	12529
E+SPASS together	6500	-	-	12529
Vampire 8.1	5110	0	7419	12529
Vampire 9	5330	0	7119	12529
Fampire 9	6411	0	6118	12529
all together	7694	-	-	12529

by symbol-based machine learning from the previously available proofs using the SNoW [10] (Sparse Network of Winnows) system run in the naive Bayes learning mode. The ability of systems to handle many axioms has improved a lot since 2005: the CASC-LTB category and also the related work on the Sledgehammer [23] link between ATPs with Isabelle/HOL have sparked interest in ATP systems dealing efficiently with large numbers of axioms. See [45] for a brief overview of the large theory methods developed so far.

The results (reported in [42]) are given in Table 3. 2408 from the 12529 theorems were proved either by E 0.9 or SPASS 2.1 from the axioms selected by the machine learner, the combined success rate of this whole system was thus 19%.

**Table 3.** Proving MPTP 0.2 theorems with machine learning support in 2005

description	proved	counter-satisfiability	timeout or memory out	total
E 0.9	2167	0	10362	12529
SPASS 2.1	1543	0	10986	12529
together	2408	0	10121	12529

This experiment demonstrates a very real and quite unique benefit of large formal mathematical libraries for conducting novel integration of AI methods. As the machine learner is trained on previous proofs, it recommends relevant premises from the large library that (according to the past experience) should be useful for proving new conjectures. A variety of machine learning methods (neural nets, Bayes nets, decision trees, nearest neighbor, etc.) can be used for this, and their performance evaluated in the standard machine learning way, i.e., by looking at the actual axiom selection done by the human author in the Mizar proof, and comparing it with the selection suggested by the trained learner. However, what if the machine learner is sometimes more clever than the human, and suggests a completely different (and perhaps better) selection of premises, leading to a different proof? In such a case, the standard machine learning evaluation (i.e. comparison of the two sets of premises) will say that the two sets of premises differ too much, and thus the machine learner has failed. This



is considered acceptable for machine learning, as in general, there is no deeper concept of *truth* available, there are just training and testing data. However in our domain we do have a method for showing that the trained learner was right (and possibly smarter than the human): we can run an ATP system on its axiom selection. If a proof is found, it provides a much stronger measure of correctness. Obviously, this is only true if we know that the translation from Mizar to TPTP was correct, i.e., conducting such experiments really requires that we take extra care to ensure that no oversimplifications were made in this translation.

In the above mentioned experiment, 329 of the 2408 (i.e. 14%) proofs found by ATPs were shorter (used less premises) than the original MML proof. An example of such proof shortening is discussed in [42], showing that the newly found proof is really valid. Instead of arguing from the first principles (definitions) like in the human proof, the combined inductive-deductive system was smart enough to find a combination of previously proved lemmas (properties) that justify the conjecture more quickly.

A similar newer evaluation is done on the whole MML in [3], comparing the original MML theory graph with the theory graph for the 9141 automatically found proofs. An illustrative example from there is theorem `COMSEQ_3:40`<sup>9</sup>, proving the relation between the limit of a complex sequence and its real and imaginary parts:

**Theorem 1.** *Let  $(c_n) = (a_n + ib_n)$  be a convergent complex sequence. Then  $(a_n)$  and  $(b_n)$  converge and  $\lim a_n = \text{Re}(\lim c_n)$  and  $\lim b_n = \text{Im}(\lim c_n)$ .*

The convergence of  $(a_n)$  and  $(b_n)$  was done the same way by the human formalizer and the ATP. The human proof of the limit equations proceeds by looking at the definition of a complex limit, expanding the definitions, and proving that a and b satisfy the definition of the real limit (finding a suitable  $n$  for a given  $\epsilon$ ). The AI/ATP just notices that this kind of groundwork was already done in a “similar” case `COMSEQ_3:39`<sup>10</sup>, which says that:

**Theorem 2.** *If  $(a_n)$  and  $(b_n)$  are convergent, then  $\lim c_n = \lim a_n + i \lim b_n$ .*

And it also notices the “similarity” (algebraic simplification) provided by theorem `COMPLEX1:28`<sup>11</sup>:

**Theorem 3.**  $\text{Re}(a + ib) = a \wedge \text{Im}(a + ib) = b$

Such (automatically found) manipulations can be used (if noticed!) to avoid the “hard thinking” about the epsilons in the definitions.

---

<sup>9</sup> [http://mizar.cs.ualberta.ca/mptp/cgi-bin/browserefs.cgi?refs=t40\\_comseq\\_3](http://mizar.cs.ualberta.ca/mptp/cgi-bin/browserefs.cgi?refs=t40_comseq_3)

<sup>10</sup> [http://mizar.cs.ualberta.ca/mptp/cgi-bin/browserefs.cgi?refs=t39\\_comseq\\_3](http://mizar.cs.ualberta.ca/mptp/cgi-bin/browserefs.cgi?refs=t39_comseq_3)

<sup>11</sup> [http://mizar.cs.ualberta.ca/mptp/cgi-bin/browserefs.cgi?refs=t28\\_complex1](http://mizar.cs.ualberta.ca/mptp/cgi-bin/browserefs.cgi?refs=t28_complex1)

### 5.3 ATP-Based Explanation, Presentation, and Cross-verification of Mizar Proofs

While the whole proofs of Mizar theorems can be quite hard for ATP systems, re-proving the Mizar atomic justification steps (called *Simple Justifications* in Mizar; these are the atomic steps discharged by the very limited and fast Mizar refutational checker [51]) turns out to be quite easy for ATPs. The combinations of E and SPASS usually solve more than 95% of such problems, and with smarter automated methods for axiom selection 99.8% success rate (14 unsolved problems from 6765) was achieved in [43]. This makes it practical to use ATPs for explanation and presentation of the (not always easily understandable) Mizar simple justifications, and to construct larger systems for independent ATP-based cross-verification of (possibly very long) Mizar proofs. In [43] such a cross-verification system is presented, using the GDV [34] system which was extended to process Jaskowski-style natural deduction proofs that make frequent use of assumptions (suppositions). MPTP was used to translate Mizar proofs to this format, and GDV together with the E, SPASS, and MaLAREa systems were used to automatically verify the structural correctness of proofs, and 99.8% of the proof steps needed for the 252 Mizar problems selected for the MPTP Challenge (see below). This provides the first practical method for independent verification of Mizar, and opens the possibility of importing Mizar proofs into other proof assistants. A web presentation allowing interaction with ATP systems and GDV verification of Mizar proofs [44] has been set up,<sup>12</sup> and an online service [47] integrating the ATP functionalities has been built.<sup>13</sup>

### 5.4 Use of MPTP for ATP Challenges and Competitions

The first MPTP problems were included in the TPTP library in 2006, and were already used for the 2006 CASC competition. In 2006, the MPTP \$100 Challenges<sup>14</sup> were created and announced. This is a set of 252 related large-theory problems needed for one half (one of two implications) of the Mizar proof of the general topological Bolzano-Weierstrass theorem [5]. Unlike the CASC competition, the challenge had an overall time limit (252 \* 5 minutes = 21 hours) for solving the problems, allowing complementary techniques like machine learning from previous solutions to be experimented with transparently within the overall time limit. The challenge was won a year later by the leanCoP [27] system, having already revealed several interesting approaches to ATP in large theories: goal-directed calculi like connection tableaux (used in leanCoP), model-based axiom selection (used e.g. in SRASS [35]), and machine learning of axiom relevance (used in MaLAREa) from previous proofs and conjecture characterizations (where *axiom relevance* can be defined as the likelihood that an axiom will be needed for proving a particular conjecture). The MPTP Challenge problems

<sup>12</sup> <http://www.tptp.org/MizarTPTP>

<sup>13</sup> <http://mws.cs.ru.nl/~mptp/MizAR.html>, and  
<http://mizar.cs.ualberta.ca/~mptp/MizAR.html>

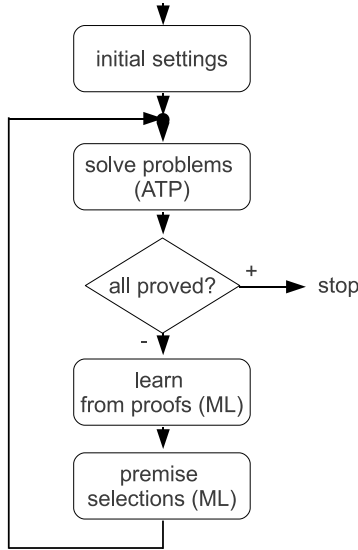
<sup>14</sup> <http://www.tptp.org/MPTPChallenge/>

were again included in TPTP and used for the standard CASC competition in 2007. In 2008, the CASC-LTB (Large Theory Batch) category appeared for the first time with a similar setting like the MPTP Challenges, and additional large-theory problems from the SUMO and Cyc [31] ontologies. A set of 245 relatively hard Mizar problems was included for this purpose in TPTP, coming from the most advanced parts of the Mizar library. The problems come in four versions that contain different numbers of the previously available MML theorems and definitions as axioms. The largest versions thus contain over 50000 axioms. An updated larger version (MPTP2078) of the MPTP Challenge benchmark was developed in 2011 [2], consisting of 2078 interrelated problems in general topology, and making use of precise dependency analysis of the MML for constructing the easy versions of the problems. The MPTP2078 problems were used in the Mizar category of the 2012 CASC@Turing ATP competition. This category had similar rules as the MPTP Challenge, i.e., a large number (400) of related MPTP2078 problems is being solved in an overall time limit. Unlike in the MPTP Challenge, an additional training set of 1000 MPTP2078 problems (disjoint from the competition problems) was made available before the competition together with their Mizar and Vampire proofs. This way, more expensive training and tuning methods could be developed before the competition, however (as common in machine-learning competitions) the problem-solving knowledge extracted from the problems by such methods has to be sufficiently general to be useful on the disjoint set of competition problems.

## 5.5 Development of Larger AI Metasystems Like MaLARea and MaLeCoP on MPTP Data

In Section 5.2, it is explained how the notion of mathematical *truth* implemented through ATPs can improve the evaluation of learning systems working on large semantic knowledge bases like translated MML. This is however only one part of the AI fun made possible by such large libraries being available to ATPs. Another part is that the newly found proofs can be recycled, and again used for learning in such domains. This closed loop (see Figure 1) between using deductive methods to find proofs, and using inductive methods to learn from the existing proofs and suggest new proof directions, is the main idea behind the Machine Learner for Automated Reasoning (MaLARea [41,48]) metasystem, which turns out to have so far the best performance on large theory benchmarks like the MPTP Challenge and MPTP2078. There are many kinds of information that such an autonomous metasystem can try to use and learn. The second version of MaLARea already uses also structural and semantic features of formulas for their characterization and for improving the axiom selection.

MaLARea can work with arbitrary ATP backends (E and SPASS by default), however, the communication between learning and the ATP systems is *high-level*: The learned relevance is used to try to solve problems with varied limited numbers of the most relevant axioms. Successful runs provide additional data for learning (useful for solving related problems), while unsuccessful runs can yield countermodels, which can be re-used for semantic pre-selection and as additional



**Fig. 1.** The basic MaLArea loop

input features for learning. An advantage of such high-level approach is that it gives a generic inductive (learning)/deductive (ATP) metasystem to which any ATP can be easily plugged as a blackbox. Its disadvantage is that it does not attempt to use the learned knowledge for guiding the ATP search process once the axioms are selected.

Hence the logical next step done in the Machine Learning Connection Prover (MaLeCoP) prototype [49]: the learned knowledge is used for guiding the proof search mechanisms in a theorem prover (leanCoP in this case). MaLeCoP follows a general advising design that is as follows (see also Figure 2): The theorem prover (P) has a sufficiently fast communication channel to a general advisor (A) that accepts queries (proof state descriptions) and training data (characterization of the proof state<sup>15</sup> together with solutions<sup>16</sup> and failures) from the prover, processes them, and replies to the prover (advising, e.g., which clauses to choose). The advisor A also talks to external (in our case learning) system(s) (E). A translates the queries and information produced by P to the formalism used by a particular E, and translates E's guidance back to the formalism used by P. At suitable time, A also hands over the (suitably transformed) training data to E, so that E can update its knowledge of the world on which its advice is based.

MaLeCoP is a very recent work, which has so far revealed interesting issues in using detailed smart guidance in large theories. Even though naive Bayes is a

<sup>15</sup> instantiated, e.g., as the set of literals/symbols on the current branch.

<sup>16</sup> instantiated, e.g., as the description of clauses used at particular proof states.

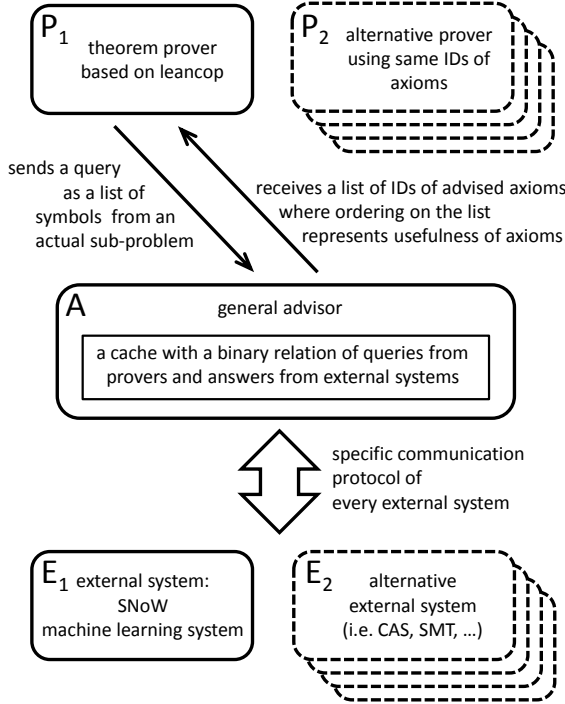


Fig. 2. The General Architecture used for MaLeCoP

comparatively fast learning and advising algorithm, in a large theory it turned out to be about 1000 times slower than a primitive tableaux extension step. So a number of strategies had to be defined that use the smart guidance only at the crucial points of the proof search. Even with such limits, the preliminary evaluation done on the MPTP Challenge already showed an average proof search shortening by a factor of 20 in terms of the number of tableaux inferences.

There are a number of development directions for knowledge-based AI/ATP architectures like MaLAREa and MaLeCoP. Extracting lemmas from proofs and adding them to the set of available premises, creating new interesting conjectures and defining new useful notions, finding optimal strategies for problem classes, faster guiding of the internal ATP search, inventing policies for efficient governing of the overall inductive-deductive loop: all these are interesting AI tasks that become relevant in this large-theory setting, and that also seem to be relevant for the task of *doing mathematics* (and exact sciences) more automatically. A particularly interesting issue is the following.

**Consistency of Knowledge and Its Transfer:** An important research topic in the emerging AI approaches to large-theory automated reasoning is the issue of *consistency of knowledge and its transfer*. In an unpublished

experiment with MaLAREa in 2007 over a set of problems exported by an early Isabelle/Sledgehammer version, MaLAREa quickly solved all of the problems, even though some of them were supposed to be hard. Larry Paulson tracked the problem to an (intentional) simplification in the first-order encoding of Isabelle/HOL types, which typically raises the overall ATP success rate (after checking in Isabelle the imported proofs) in comparison with heavier (but sound) type encodings that are also available in Isabelle/Sledgehammer. Once the inconsistency originating from the simplification was found by the guiding AI system, MaLAREa focused on fully exploiting it even in problems where such inconsistency would be ignored by standard ATP search.

An opposite phenomenon happened recently in experiments with a clausal version of MaLAREa. The CNF form introduces a large number of new skolem symbols that make similar problems and formulas look different after the reduction to clausal form (despite the fact that the skolemization attempts hard to use the same symbol whenever it can), and the AI guidance based on symbols and terms deteriorates. The same happens with the AI guidance based on finite counter-models generated by Mace [21] and Paradox [11]. The `clausefilter` utility from McCune’s Library of Automated Deduction Routines (LADR) [20] is used by MaLAREa to evaluate all available formulas in each newly found finite model, resulting in an additional semantic characterization of all formulas that often improves the premise selection. Disjoint skolem symbols however prevent a straightforward evaluation of many clauses in models that are found for differently named skolem functions. Such inability of the AI guidance to obtain and use the information about the similarity of the clauses results in about 100 fewer problems solved (700 vs. 800) in the first ten MaLAREa iterations over the MPTP2078 benchmark.

Hence a trade-off: smaller pieces of knowledge (like clauses) allow better focus, but techniques like skolemization can destroy some explicit similarities useful for learning. Designing suitable representations and learning methods on top of the knowledge is therefore very significant for the final performance, while inconsistent representations can be fatal. Using CNF and its various alternatives and improvements has been a topic discussed many times in the ATP community (also for example by Quaife in his book). Here we note that it is not just the low-level ATP algorithms that are influenced by such choices of representation, but the problem extends to and significantly influences also the performance of high-level heuristic guidance methods in large theories.

## 6 Future QED-Like Directions

There is large amount of work to be done on practically all the projects mentioned above. The MPTP translation is by no means optimal (and especially proper encoding of arithmetic needs more experiments and work). Import of ATP proofs to Mizar practically does not exist (there is a basic translator taking Otter proof objects to Mizar, however this is very distant from the readable proofs in MML). With sufficiently strong ATP systems, the cross-verification

of the whole MML could be attempted, and work on import of such detailed ATP proofs into other proof assistants could be started. The MPTP handling of second-order constructs is in some sense incomplete, and either a translation to (finitely axiomatized) Neumann-Bernays-Gödel (NBG) set theory (used by Quaipe), or usage of higher-order ATPs like Satallax [9] and LEO-II [6] would be interesting from this point of view.

More challenges and interesting presentation tools can be developed, for example an ATP-enhanced wiki for Mizar is an interesting QED-like project that is now being worked on [46]. The heuristic and machine learning methods, and combined AI metasystems, have a very long way to go, some future directions are mentioned above. This is no longer only about mathematics: all kinds of more or less formal large knowledge bases are becoming available in other sciences, and automated reasoning could become one of the strongest methods for general reasoning in sciences when sufficient amount of formal knowledge exists. Strong ATP methods for large formal mathematics could also provide useful semantic filtering for larger systems for automatic formalization of mathematical papers. This is a field that has been so far deemed to be rather science fiction than a real possibility. In particular, the idea of gradual top-down (semi-)automated formalization of mathematics written in books and papers has been so far considered outlandish in a way that is quite reminiscent of how the idea of ATP in large general mathematics was considered outlandish before the first large-scale experiments in 2003. The proposed AI solution should be similar for the two problems: guide the vast search space by the knowledge extracted from the vast amount of problems already solved. It is interesting that already within the QED project discussions, Feferman suggested<sup>17</sup> that large-scale formalization should have a necessary top-down aspect. Heuristic AI methods used for knowledge search and machine translation are becoming more and more mature, and in conjunction with strong ATP methods they could provide a basis for such large-scale (semi-)automated QED project.

## References

1. The QED Manifesto. In: Bundy, A. (ed.) CADE 1994. LNCS, vol. 814, pp. 238–251. Springer, Heidelberg (1994)
2. Alama, J., Kühlwein, D., Tsvitshivadze, E., Urban, J., Heskes, T.: Premise selection for mathematics by corpus analysis and kernel methods. CoRR, abs/1108.3446 (2011)
3. Alama, J., Kühlwein, D., Urban, J.: Automated and Human Proofs in General Mathematics: An Initial Comparison. In: Bjørner, N., Voronkov, A. (eds.) LPAR-18. LNCS, vol. 7180, pp. 37–45. Springer, Heidelberg (2012)
4. Althaus, E., Kruglov, E., Weidenbach, C.: Superposition Modulo Linear Arithmetic SUP(LA). In: Ghilardi, S., Sebastiani, R. (eds.) FroCoS 2009. LNCS, vol. 5749, pp. 84–99. Springer, Heidelberg (2009)
5. Bancerek, G., Endou, N., Sakai, Y.: On the characterizations of compactness. *Formalized Mathematics* 9(4), 733–738 (2001)

<sup>17</sup> <http://mizar.org/qed/mail-archive/volume-2/0003.html>

6. Benzmüller, C.E., Paulson, L.C., Theiss, F., Fietzke, A.: LEO-II - A Cooperative Automatic Theorem Prover for Classical Higher-Order Logic (System Description). In: Armando, A., Baumgartner, P., Dowek, G. (eds.) *IJCAR 2008*. LNCS (LNAI), vol. 5195, pp. 162–170. Springer, Heidelberg (2008)
7. Bertot, Y., Casteran, P.: *Interactive Theorem Proving and Program Development - Coq'Art: The Calculus of Inductive Constructions*. Texts in Theoretical Computer Science. Springer (2004)
8. Bonacina, M.P., Lynch, C., de Moura, L.M.: On deciding satisfiability by theorem proving with speculative inferences. *J. Autom. Reasoning* 47(2), 161–189 (2011)
9. Brown, C.E.: Satallax: An Automatic Higher-Order Prover. In: Gramlich, B., Miller, D., Sattler, U. (eds.) *IJCAR 2012*. LNCS, vol. 7364, pp. 111–117. Springer, Heidelberg (2012)
10. Carlson, A., Cumby, C., Rosen, J., Roth, D.: *SNoW User's Guide*. Technical Report UIUC-DCS-R-99-210, University of Illinois at Urbana-Champaign (1999)
11. Claessen, K., Sorensson, N.: New Techniques that Improve MACE-style Finite Model Finding. In: Baumgartner, P., Fermueller, C. (eds.) *Proceedings of the CADE-19 Workshop: Model Computation - Principles, Algorithms, Applications* (2003)
12. Dahn, I., Wernhard, C.: First order proof problems extracted from an article in the MIZAR Mathematical Library. In: Bonacina, M.P., Furbach, U. (eds.) *Int. Workshop on First-Order Theorem Proving (FTP 1997)*. RISC-Linz Report Series No. 97-50, vol. (97-50), pp. 58–62. Johannes Kepler Universität, Linz, Austria (1997)
13. Grabowski, A., Kornilowicz, A., Naumowicz, A.: Mizar in a nutshell. *Journal of Formalized Reasoning* 3(2), 153–245 (2010)
14. Hähnle, R., Kerber, M., Weidenbach, C.: Common Syntax of the DFG-Schwerpunktprogramm Deduction. Technical Report TR 10/96, Fakultät für Informatik, Universität Karlsruhe, Karlsruhe, Germany (1996)
15. Hales, T.C.: Introduction to the Flyspeck project. In: Coquand, T., Lombardi, H., Roy, M.-F. (eds.) *Mathematics, Algorithms, Proofs*. Dagstuhl Seminar Proceedings, vol. 05021, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany (2005)
16. Harrison, J.: HOL Light: A Tutorial Introduction. In: Srivas, M., Camilleri, A. (eds.) *FMCAD 1996*. LNCS, vol. 1166, pp. 265–269. Springer, Heidelberg (1996)
17. Jaskowski, S.: On the rules of suppositions. *Studia Logica*, 1 (1934)
18. Korovin, K., Voronkov, A.: Integrating Linear Arithmetic into Superposition Calculus. In: Duparc, J., Henzinger, T.A. (eds.) *CSL 2007*. LNCS, vol. 4646, pp. 223–237. Springer, Heidelberg (2007)
19. Matuszewski, R., Rudnicki, P.: Mizar: the first 30 years. *Mechanized Mathematics and Its Applications* 4, 3–24 (2005)
20. McCune, W.W.: LADR: Library of Automated Deduction Routines, <http://www.mcs.anl.gov/AR/ladr>
21. McCune, W.W.: Mace4 Reference Manual and Guide. Technical Report ANL/MCS-TM-264, Argonne National Laboratory, Argonne, USA (2003)
22. McCune, W.W.: Otter 3.3 Reference Manual. Technical Report ANL/MSC-TM-263, Argonne National Laboratory, Argonne, USA (2003)
23. Meng, J., Paulson, L.C.: Lightweight relevance filtering for machine-generated resolution problems. *J. Applied Logic* 7(1), 41–57 (2009)
24. Niles, I., Pease, A.: Towards a standard upper ontology. In: *FOIS 2001: Proceedings of the International Conference on Formal Ontology in Information Systems*, pp. 2–9. ACM Press, New York (2001)



25. Nipkow, T., Paulson, L.C., Wenzel, M.T.: Isabelle/HOL - A Proof Assistant for Higher-Order Logic. LNCS, vol. 2283. Springer, Heidelberg (2002)
26. Nonnengart, A., Weidenbach, C.: Computing small clause normal forms. In: Handbook of Automated Reasoning, vol. I, pp. 335–367. Elsevier and MIT Press (2001)
27. Otten, J., Bibel, W.: leanCoP: Lean Connection-Based Theorem Proving. Journal of Symbolic Computation 36(1-2), 139–161 (2003)
28. Prevosto, V., Waldmann, U.: SPASS+T. In: Sutcliffe, G., Schmidt, R., Schulz, S. (eds.) ESCoR 2006. CEUR, vol. 192, pp. 18–33 (2006)
29. Quaife, A.: Automated Deduction in von Neumann-Bernays-Godel Set Theory. Journal of Automated Reasoning 8(1), 91–147 (1992)
30. Quaife, A.: Automated Development of Fundamental Mathematical Theories. Kluwer Academic Publishers (1992)
31. Ramachandran, D., Reagan, P., Goolsbey, K.: First-orderized ResearchCyc: Expressiveness and Efficiency in a Common Sense Knowledge Base. In: Shvaiko, P. (ed.) Proceedings of the Workshop on Contexts and Ontologies: Theory, Practice and Applications (2005)
32. Riazanov, A., Voronkov, A.: The Design and Implementation of Vampire. AI Communications 15(2-3), 91–110 (2002)
33. Schulz, S.: E: A Brainiac Theorem Prover. AI Communications 15(2-3), 111–126 (2002)
34. Sutcliffe, G.: Semantic Derivation Verification. International Journal on Artificial Intelligence Tools 15(6), 1053–1070 (2006)
35. 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)
36. Sutcliffe, G.: The 4th IJCAR automated theorem proving system competition - CASC-J4. AI Commun. 22(1), 59–72 (2009)
37. Sutcliffe, G.: The TPTP World – Infrastructure for Automated Reasoning. In: Clarke, E.M., Voronkov, A. (eds.) LPAR-16 2010. LNCS, vol. 6355, pp. 1–12. Springer, Heidelberg (2010)
38. Urban, J.: Translating Mizar for First Order Theorem Provers. In: Asperti, A., Buchberger, B., Davenport, J.H. (eds.) MKM 2003. LNCS, vol. 2594, pp. 203–215. Springer, Heidelberg (2003)
39. Urban, J.: MPTP - Motivation, Implementation, First Experiments. Journal of Automated Reasoning 33(3-4), 319–339 (2004)
40. Urban, J.: XML-izing Mizar: Making Semantic Processing and Presentation of MML Easy. In: Kohlhase, M. (ed.) MKM 2005. LNCS (LNAI), vol. 3863, pp. 346–360. Springer, Heidelberg (2006)
41. Urban, J.: MaLAREa: a Metasystem for Automated Reasoning in Large Theories. In: Urban, J., Sutcliffe, G., Schulz, S. (eds.) Proceedings of the CADE-21 Workshop on Empirically Successful Automated Reasoning in Large Theories, pp. 45–58 (2007)
42. Urban, J.: MPTP 0.2: Design, Implementation, and Initial Experiments. Journal of Automated Reasoning 37(1-2), 21–43 (2007)
43. Urban, J., Sutcliffe, G.: ATP Cross-Verification of the Mizar MPTP Challenge Problems. In: Dershowitz, N., Voronkov, A. (eds.) LPAR 2007. LNCS (LNAI), vol. 4790, pp. 546–560. Springer, Heidelberg (2007)
44. Urban, J., Trac, S., Sutcliffe, G., Puzis, Y.: Combining Mizar and TPTP Semantic Presentation Tools. In: Proceedings of the Mathematical User-Interfaces Workshop 2007 (2007)

45. Urban, J.: An overview of methods for large-theory automated theorem proving (invited paper). In: Höfner, A., McIver, G. (eds.) ATE Workshop. CEUR Workshop Proceedings, vol. 760, pp. 3–8. CEUR-WS.org (2011)
46. Urban, J., Alama, J., Rudnicki, P., Geuvers, H.: A Wiki for Mizar: Motivation, Considerations, and Initial Prototype. In: Autexier, S., Calmet, J., Delahaye, D., Ion, P.D.F., Rideau, L., Rioboo, R., Sexton, A.P. (eds.) AISC 2010. LNCS, vol. 6167, pp. 455–469. Springer, Heidelberg (2010)
47. Urban, J., Rudnicki, P., Sutcliffe, G.: ATP and Presentation Service for Mizar Formalizations. *J. Autom. Reasoning* 50(2), 229–241 (2013)
48. Urban, J., Sutcliffe, G., Pudlák, P., Vyskočil, 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)
49. Urban, J., Vyskočil, J., Štěpánek, P.: **MaLeCoP** Machine Learning Connection Prover. In: Brunnler, K., Metcalfe, G. (eds.) TABLEAUX 2011. LNCS, vol. 6793, pp. 263–277. Springer, Heidelberg (2011)
50. Weidenbach, C., Dimova, D., Fietzke, A., Kumar, R., Suda, M., Wischniewski, P.: SPASS Version 3.5. In: Schmidt, R.A. (ed.) CADE 2009. LNCS, vol. 5663, pp. 140–145. Springer, Heidelberg (2009)
51. Wiedijk, F.: CHECKER - notes on the basic inference step in Mizar (2000), <http://www.cs.kun.nl/~freek/mizar/by.dvi>