EPiC Series in Computing

Volume 40, 2016, Pages 122–126

IWIL-2015. 11th International Workshop on the Implementation of Logics

Experiments with State-of-the-art Automated Provers on Problems in Tarskian Geometry (Position paper)

Josef Urban[1]*and Robert Veroff[2]

[1] Czech Technical University
josef.urban@gmail.com
[2] University of New Mexico
veroff@cs.unm.edu

Abstract

We describe our initial experiments with several state-of-the-art automated theorem provers on the problems in Tarskian Geometry created by Beeson and Wos. In comparison to the manually-guided OTTER proofs by Beeson and Wos, we can solve a large number of problems fully automatically, in particular thanks to the recent large-theory reasoning methods.

1 Introduction

In their 2014 paper [1] Beeson and Wos report on their project which uses OTTER [12] to find proofs of theorems in Tarskian geometry proved in Part I of the book "Metamathematische Methoden in der Geometrie" by Schwabhuser, Szmielew and Tarski [15]. We have become interested in their work for a couple of reasons.

First, we have recently started to look at good ways to apply Veroff's techniques such as *hints* and *proof sketches* [19] on large-theory problems coming from ITPs like Mizar [7], Isabelle [10], and HOL [6]. Under the hints strategy, a generated clause is given special consideration if it *matches* (subsumes) a user-supplied hint clause. A *proof sketch* for a theorem T is a sequence of clauses giving a set of conditions *sufficient* to prove T. Typically, the clauses of a proof sketch identify potentially notable milestones on the way to finding a proof. The hints mechanism provides a natural and effective way to take full advantage of proof sketches in the search for a proof. It can be very effective to include as proof sketches proofs of related theorems in the same area of study. The large number of related problems coming from the ITP libraries seem to be a natural target for such techniques.

This, however, turns out to be nontrivial. Hints currently work on the clause level, and consistency of their symbols (i.e., the same symbol always having the same meaning) across

*Supported by NWO grant *Knowledge-based Automated Reasoning* and by ERC Consolidator grant nr. 649043 *AI4REASON*.

different problems is assumed. This works for the algebraic clausal problems that Prover9 [11] is typically applied to. The large-theory problems, however, are formulated in FOF, and their Skolemization typically produces many symbols that might not be consistently used among the different problems. This, and the large number of problems, formulas and symbols so far confuses the hints technique. The Tarski problems seem to be right in between the two kinds of problems: they are clausal and do not contain too many symbols, yet they at least partially also qualify as large-theory problems. The number of clauses is about 200, and the more advanced problems typically contain all the previously proved lemmas as axioms. Making the hints method work on the Tarski problems would be a useful step towards making hints work on the ITP-generated large-theory problems.

The large-theory aspect of the Tarski problems is another motivation. It turns out that some of the problems really need many axioms (e.g., 59 for an automatically found proof of Satz10.12a by the E.T system [5]). Quite a bit of work has been done recently to develop methods that select the most relevant lemmas for proving a particular conjecture [2] and to develop specialized strategies for solving large-theory problems [18]. Beeson and Wos do not use such methods. Instead, they use OTTER and often revert to what is quite close to interactive theorem proving: writing manually intermediate lemmas that guide the proof search. Looking at the results of recent CASC and its large-theory divisions [17, 16], it would be quite surprising if the performance of unguided OTTER run in an automated mode on the problems would be anywhere near the state-of-the-art systems. That's why we first simply try to run some of those and see how many problems we can today prove without any interactive guidance. Here we report the results and issues found so far.

2 Running Several ATPs on the Tarski Problems

We started by downloading 164 problems in the OTTER format from the web site of the project, and re-proved 163 of them with OTTER (using all the tricks used by Beeson and Wos). The remaining Satz11.15b.in – which was marked as work-in-progress – was eventually proved in 19000 s by OTTER.

We first wrote a script that commented out the main sources of "guiding information" in the files: the special parameter settings, the hints and the demodulators. Then we ran OTTER and Prover9 on such "blank" problems in auto mode for 300 s. OTTER solved 72 problems and Prover9 102. In 20 s Prover9 solved only 92 problems . This indicated that using more CPU time and different strategies would likely help further.

To be able to use other ATPs, we have used the ladr_to_tptp translator, followed by some pre- and post-processing that gets around using some symbols both as functions and as predicates in the OTTER problems. Since the problems are large and may profit from conjecture-oriented clause selection mechanisms in E [14, 13], we also renamed the status of the conjecture-originated axioms to "negated_conjecture". Then we ran E 1.8 (using "–auto-schedule", i.e., its strategy scheduling mode) for 300 s solving 125 problems, and E.T 0.1 also for 300 s, solving 129 problems. E.T uses more strategies than E, and applies stronger axiom pruning. The strongest system is, however, Vampire 3.0 [9], which solves 137 of the problems in 300 s. We have also tried Z3 [3] (to get some more solutions), which solves 92 files in 300 s. All the systems together can prove 141 problems in 300 s at this point – about twice as many as unguided OTTER. The manual coding of the proofs by providing intermediate steps to OTTER could have been largely avoided with quite limited resources.

Finally, we did some experiments with higher time limits. Three more problems could be solved by running Vampire 3.0 in the CASC mode for 1 hour. Running Vampire for longer

in this mode does not work – it seems that the CASC mode is limited to such shorter overall times. Vampire and E.T (run with even higher time limits) together add 6 more problems, making the final count so far at 147 fully automatically solved problems out of the 164. These results are, however, a bit questionable due to the issues described next.

3 Problems with the Problems

During the translation to TPTP and the initial runs with other ATPs we encountered several kinds of problems with the files. Some of them – e.g. problems with naming of clauses – prevent us so far from running strong learning-based large-theory systems such as MALARea [8] on the problems.

These problems arose because the input files were created and edited by hand, and theorems were hand-Skolemized once proved for use in proving subsequent theorems. A number of errors were introduced in this process. Beeson and Wos independently recognized the need for a more systematic approach to ensuring the correctness of the input files, and eventually solved that problem by using a "master list" of theorems and their Skolemizations, with the Skolemizations machine-checked, and the input files mechanically generated from the master list using a PHP script. The files now posted on the project website were produced by this method. They did not have time for this correctness-checking before the original posting (before the IJCAR deadline), and unfortunately the corrected versions came after the experiments reported here. In the interest of emphasizing the point that when working with a large theory, one needs to automate the Skolemization and the generation of input files, we list the problems that we encountered with the hand-generated files:

- In several cases, the statement of a particular lemma or definition is different in different problems. This is really dangerous in a larger formalization and could lead to many issues. Sometimes this only involves naming of the variables, but sometimes the clauses really differ. This could not happen in an ITP-based development, where each lemma/definition is stated only once, and ATP problems are generated automatically by the corresponding hammer systems. Even pervasive use of simple ATP mechanisms such as includes should be enough to prevent this.
- Using manual Skolemization and the clausal form has its dangers: some of the different statements of the same formula from the book are caused by bugs in manual Skolemization, or even just by introducing differently named Skolem functions and constants. This could be prevented by switching to formulas instead of clauses.
- While the OTTER/Prover9 format occasionally allows more mathematician-friendly notation than TPTP, unlike in TPTP the name of a clause is just an unchecked comment that can be anywhere and have typos in it. This already prevents simple duplication/-consistency checking as done by the TPTP tools, and also more advanced checking of the proofs (and formulas uses in them) as done, e.g., by the GDV verifier.
- In some of the clauses there are clear typos, leading, e.g., to the appearance of singleton variables. If a Prolog-friendly language such as TPTP was used, they would be easily detected and reported when loading the formulas into Prolog.

We were quite surprised that relatively many such issues have escaped not only the authors, but also the IJCAR'14 refereeing – at least there is no mention of the debugging processes in the paper. In the conferences focused on interactive theorem proving as well as when reviewing

submissions to some large formal libraries, it is today a common practice to not only run and recheck the formal developments, but also to look at and point out various issues with definitions, obvious naming problems, dangerous use of additional axioms, etc. It would be good if IJCAR – and in general the ATP community – adopted similar refereeing practices in such cases.

4 Future Work

Our plan is to try to prove as many Tarski problems as possible by the standard (large-theory) techniques available in systems like E, Prover9 and Vampire, and then proceed to test Prover9 techniques that transfer hints between the problems. We plan to switch to the new – more automatically produced – version of the problems. It is likely that in this almost-large-theory setting we will need to complement the hints method with methods that select only the most relevant hints for a particular problem. Such methods are similar to the recent methods for premise and lemma selection developed in the context of large theories. We hope that this way we will eventually (thanks to the Tarski problems) also arrive at good techniques for hint guidance in large theories. A useful future extension would be to import and verify the whole development in a safe ITP such as HOL Light [4], using the existing TPTP proof importing capabilities.

5 Acknowledgments

Thanks to Mike Beeson for providing us with the problems at IJCAR'14 and discussing several issues.

References

[1] Michael Beeson and Larry Wos. OTTER proofs in Tarskian geometry. In Stéphane Demri, Deepak Kapur, and Christoph Weidenbach, editors, *Automated Reasoning - 7th International Joint Conference, IJCAR 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 19-22, 2014. Proceedings*, volume 8562 of *Lecture Notes in Computer Science*, pages 495–510. Springer, 2014.

[2] Jasmin Christian Blanchette, Cezary Kaliszyk, Lawrence C. Paulson, and Josef Urban. Hammering towards QED. *J. Formalized Reasoning*, 9(1):101–148, 2016.

[3] Leonardo Mendonça de Moura and Nikolaj Bjørner. Z3: An Efficient SMT Solver. In C. R. Ramakrishnan and Jakob Rehof, editors, *TACAS*, volume 4963 of *LNCS*, pages 337–340. Springer, 2008.

[4] John Harrison. HOL Light: A tutorial introduction. In Mandayam K. Srivas and Albert John Camilleri, editors, *FMCAD*, volume 1166 of *LNCS*, pages 265–269. Springer, 1996.

[5] Cezary Kaliszyk, Stephan Schulz, Josef Urban, and Jirí Vyskocil. System description: E.T. 0.1. In Amy P. Felty and Aart Middeldorp, editors, *Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings*, volume 9195 of *Lecture Notes in Computer Science*, pages 389–398. Springer, 2015.

[6] Cezary Kaliszyk and Josef Urban. Learning-assisted automated reasoning with Flyspeck. *J. Autom. Reasoning*, 53(2):173–213, 2014.

[7] Cezary Kaliszyk and Josef Urban. MizAR 40 for Mizar 40. *J. Autom. Reasoning*, 55(3):245–256, 2015.

[8] Cezary Kaliszyk, Josef Urban, and Jirí Vyskocil. Machine learner for automated reasoning 0.4 and 0.5. In Stephan Schulz, Leonardo de Moura, and Boris Konev, editors, *4th Workshop on Practical*

Aspects of Automated Reasoning, PAAR@IJCAR 2014, Vienna, Austria, 2014, volume 31 of *EPiC Series in Computing*, pages 60–66. EasyChair, 2014.

[9] Laura Kovács and Andrei Voronkov. First-order theorem proving and Vampire. In Natasha Sharygina and Helmut Veith, editors, *CAV*, volume 8044 of *LNCS*, pages 1–35. Springer, 2013.

[10] Daniel Kühlwein, Jasmin Christian Blanchette, Cezary Kaliszyk, and Josef Urban. MaSh: Machine learning for Sledgehammer. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *ITP 2013*, volume 7998 of *LNCS*, pages 35–50. Springer, 2013.

[11] William McCune. Prover9 and Mace4. http://www.cs.unm.edu/~mccune/prover9/, 2005–2010.

[12] W.W. McCune. Otter 3.3 Reference Manual. Technical Report ANL/MSC-TM-263, Argonne National Laboratory, Argonne, USA, 2003.

[13] Stephan Schulz. Simple and efficient clause subsumption with feature vector indexing. In Maria Paola Bonacina and Mark E. Stickel, editors, *Automated Reasoning and Mathematics - Essays in Memory of William W. McCune*, volume 7788 of *Lecture Notes in Computer Science*, pages 45–67. Springer, 2013.

[14] Stephan Schulz. System description: E 1.8. In Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *LPAR*, volume 8312 of *LNCS*, pages 735–743. Springer, 2013.

[15] W. Schwabhauser, W. Szmielew, and A. Tarski. *Metamathematische Methoden in der Geometrie: Teil I: Ein axiomatischer Aufbau der euklidischen Geometrie. Teil II: Metamathematische Betrachtungen (Hochschultext)*. Springer-Verlag (1983), reprinted 2012 by Ishi Press, with a new foreword by Michael Beeson, 2012.

[16] Geoff Sutcliffe. The CADE-24 automated theorem proving system competition - CASC-24. *AI Commun.*, 27(4):405–416, 2014.

[17] Geoff Sutcliffe and Josef Urban. The CADE-25 automated theorem proving system competition - CASC-25. *AI Commun.*, 29(3):423–433, 2015.

[18] Josef Urban. BliStr: The Blind Strategymaker. In Georg Gottlob, Geoff Sutcliffe, and Andrei Voronkov, editors, *Global Conference on Artificial Intelligence, GCAI 2015, Tbilisi, Georgia, October 16-19, 2015*, volume 36 of *EPiC Series in Computing*, pages 312–319. EasyChair, 2015.

[19] Robert Veroff. Using hints to increase the effectiveness of an automated reasoning program: Case studies. *J. Autom. Reasoning*, 16(3):223–239, 1996.