Learning SMT Enumeration *

Mikoláš Janota[1], Jelle Piepenbrock[1,2], and Bartosz Piotrowski[1,3]

[1] Czech Technical University, Prague,
[2] Radboud University Nijmegen, The Netherlands
[3] University of Warsaw, Poland

Introduction Even though satisfiability-modulo-theories (SMT) solvers were mainly focused on solving quantifier-free first-order problems, many of them currently support quantified formulas. The main technique used in SMTs to handle quantifiers is *quantifier instantiation*. In this approach the solver intersperses generating ground instances of the quantified formulas and attempts to find a model for the ground formulas. Whenever a model is found, new ground instances are generated. This process continues until a contradiction is found at the ground level (which refutes the problem) or computational resources are exhausted.

A large majority of the instantiations produced during the search are not used in the final proof. The challenge is to produce *useful* instantiations, i.e., instantiations which are likely to contribute to the proof. One of the strategies to find these instances is *enumerative instantiation* [5]. Here, quantified variables in formulas are substituted with candidate ground terms. The set of these terms is induced by the ground model found in the last iteration of the main loop of the algorithm. This set is ordered using some predefined term ordering. In CVC5 this order is determined by *age*, i.e., the terms that exist longer in the formula are used first.

In our work we implement a machine-learning (ML) guidance for term-selection for the instantiations. More precisely, we use a machine-learned, formula-dependent term-scoring function in place of the predefined term ordering.

This is closely related to the work presented at AITP 2019 [2]. There, the authors also experimented with machine-learned guidance for term-selection for instantiations. However, their setting was substantially different. In particular, their machine-learned function served as a binary filter on a set of terms, not as a scoring function inducing an ordering. Moreover, their implementation was impractically slow. We show that the ML-guided SMT solver has improved proving performance when compared to the unguided solver with the same time limit.

Implementation As a basis for our experiments we use a well-established and efficient SMT-solver – CVC5 [1]. To model the term-scoring function we use the LightGBM toolkit [3]. It efficiently implements a versatile and powerful ML algorithm – the gradient boosted trees. The scoring function $S\colon \mathcal{F} \to [0,1]$ takes as its argument features $F(\phi, t)$ of a pair of the quantified formula ϕ and the candidate term t which may be used for instantiation. The returned score is intended to reflect how likely it is that ϕ instantiated with t will be used in the final proof.

As features of (ϕ, t) pairs we use information extracted from CVC5. For every symbol appearing in terms and formulas CVC5 determines its *kind*. These kinds include, e.g., *variable*, *skolem*, *not*, *and*, *plus*, *forall*, etc. We use these syntactic *kinds* to define a *bag-of-words*-type featurizer $\mathrm{BOW}(x)$, where x is a term or a formula, and the information returned by BOW consists of counts of kinds of symbols appearing in x[1]. Additionally, we use 6 numerical features describing the candidate terms: *varFrequency*, *age*, *phase*, *relevant*, *depth*, and *tried*.

*The results were supported by the Ministry of Education, Youth and Sports within the dedicated program ERC CZ under the project POSTMAN no. LL1902.

[1] For example, $\mathrm{BOW}(\forall x\, 2 + x = \mathrm{skl}_1 + 3) = \{\mathit{forall} : 1, \mathit{variable:}\ 1, \mathit{const} : 2, \mathit{skolem} : 1, \mathit{plus} : 2\}$.

varFrequency represent the number of times the variable occurs in the quantifier; *age* and *phase* measure how long the term has been in the candidate pool, with *age* being a more fine-grained measure. *depth* indicates the tree depth of the term. These, together with a disjoint union of BOW(ϕ) and BOW(t) constitute the features $F(\phi, t)$ being an input for the scoring function.

Experiments For evaluation we use SMT-LIB problems from the UFNIA/sledgehammer category. Problems solvable by CVC5 without doing any instantiations are filtered out.

First, we grid-search hyper-parameters for training LightGBM model on a random split. The selected ones are: $num\_trees = 50$, $learning\_rate = 0.1$, $num\_leaves = 32$, $max\_depth = 10$.

Then a looping-style evaluation is run (similar to [4, 6]). In the first iteration, an unguided SMT-solver is run on the benchmark and data extracted from the solved problems is used to train the ML model. Then, the model is used to guide the solver in the next iteration of solving the benchmark. The success rate is recorded and examples extracted from the newly solved problems are added to the training set. This solving-training procedure is repeated 20 times. The time limit for each solving attempt is limited to 120 s.

We use three metrics of a success: (1) a number of problems solved in the current iteration, (2) a cumulative number of problems solved so far, (3) an average number of instantiations generated by the solver in the current iteration (it may be seen as an abstract running time).

An ablation study is done by comparing to the standard solver with random perturbations, where the terms ordered by the predefined ordering are randomly swapped with probability 0.1.

The results of the looping evaluation in terms of the metrics (1-3) are presented in Figure 1. The ML-guided solver at the end of the loop is better than the randomized one with respect to all the metrics. Importantly, we see a growing trend in the number of problems solved by the ML-guided solver in individual iterations. However, there is quite high variance in this statistic.

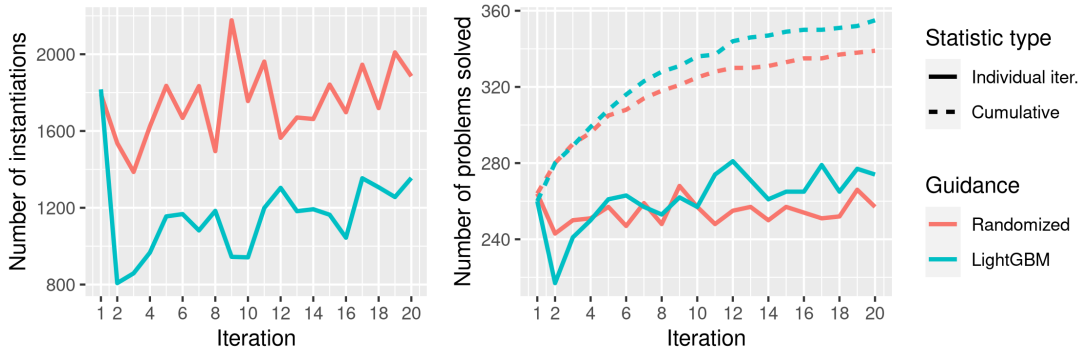

Figure 1: Statictics from the looping evaluation on the UFNIA Sledgehammer problems. The ML-guided solver (blue) *vs* the standard solver with randomly perturbated term orderings (red).

Conclusions and future work We show that ML guidance can be effectively used to guide instantiation within CVC5. There are many possible directions for future work. One of the most important areas for improvement is the treatment of cases where multiple variables in one formula need to be instantiated. Currently we handle each quantifier independently, while a more sophisticated method that takes into account the relations between quantifiers is more natural for the problem. For this, a way to score *tuples* of terms to instantiate with instead of single terms is needed. This will require training a tuple-scoring function and implementing a search procedure guided by this function, like A* search algorithm or a beam-search.

References

[1] Clark W. Barrett, Christopher L. Conway, Morgan Deters, Liana Hadarean, Dejan Jovanovic, Tim King, Andrew Reynolds, and Cesare Tinelli. CVC4. In Ganesh Gopalakrishnan and Shaz Qadeer, editors, *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, volume 6806 of *Lecture Notes in Computer Science*, pages 171–177. Springer, 2011.

[2] Jasmin Christian Blanchette, Daniel El Ouraoui, Pascal Fontaine, and Cezary Kaliszyk. Machine learning for instance selection in smt solving. In Thomas C. Hales, Cezary Kaliszyk, Ramana Kumar, Stephan Schulz, and Josef Urban, editors, *4th Conference on Artificial Intelligence and Theorem Proving, AITP*, 2019.

[3] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[4] Bartosz Piotrowski and Josef Urban. Atpboost: Learning premise selection in binary setting with ATP feedback. In Didier Galmiche, Stephan Schulz, and Roberto Sebastiani, editors, *Automated Reasoning - 9th International Joint Conference, IJCAR 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings*, volume 10900 of *Lecture Notes in Computer Science*, pages 566–574. Springer, 2018.

[5] Andrew Reynolds, Haniel Barbosa, and Pascal Fontaine. Revisiting enumerative instantiation. In Dirk Beyer and Marieke Huisman, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 24th International Conference, TACAS 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings, Part II*, volume 10806 of *Lecture Notes in Computer Science*, pages 112–131. Springer, 2018.

[6] Josef Urban, Geoff Sutcliffe, Petr Pudlák, and Jirí Vyskocil. Malarea SG1- machine learner for automated reasoning with semantic guidance. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *Automated Reasoning, 4th International Joint Conference, IJCAR 2008, Sydney, Australia, August 12-15, 2008, Proceedings*, volume 5195 of *Lecture Notes in Computer Science*, pages 441–456. Springer, 2008.