ISMT for SMT COMP 2022

Fuqi Jia[1,2], Rui Han[1,2], Minghao Liu[1,2], Cunjing Ge[4],
Pei Huang[1,2], Feifei Ma[1,3], and Jian Zhang[1,2]

[1] University of Chinese Academy of Sciences, Beijing, 100049, China
[2] State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences (ISCAS), Beijing, 100190, China
[3] Laboratory of Parallel Software and Computational Science, ISCAS, Beijing, 100190, China
[4] School of Artificial Intelligence, Nanjing University, Nanjing, 210023, China
{jiafq, maff, zj}@ios.ac.cn

1 Overview

We propose a solver *ismt* for the model validation and a wrapper solver *yices-ismt* for the single query on QF_NIA that combines *yices2* [1] and *ismt*, where *yices* is the winner solver of QF_NIA of UNSAT performance in SMT-COMP 2021. The solver *ismt* is our experimental SMT solver, composed of a parser, a preprocessor, and a theory solver. The preprocessor performs a sequence of calls to formula rewriters. The theory solver consists of four parts: collector, decider, searcher, and resolver. In addition, *ismt* implements optional algorithm plugins invoked in the theory solver, such as bit-blasting, interval arithmetic, etc.

2 Dependencies

For completeness, *yices-ismt* invokes the SMT solver *yices2* 2.6.2 as a sub-solver. Rewriters and algorithms related to polynomial and interval are implemented on the libpoly library v0.1.11 [2]. The back-end SAT solver of the bit-blasting algorithm is from CaDiCal 1.5.2 [3].

3 Implementation

Ismt. Currently, *ismt* only supports QF_NIA. We propose a series of bit-width decision heuristics as a decider. After intervals for all variables are (partially or) completely decided, the searcher attempts to find a model in the space via an improved bit-blasting method from [4].

yices-ismt. Assume that the original SMT formula is ϕ, ismt searches in a finite space $F = \bigotimes_{x \in V} I(x)$, where V is the set of variables and $I(x) = l_x \leq x \leq u_x$ is the interval constraint for variable x. If *ismt* has not found a model in F, the combined solver *yices-ismt* will generate a lemma ψ to rule out the space where $\psi = \bigvee_{x \in V} \neg I(x)$. Otherwise, if *ismt* runs out of resources during searching, $\psi = \top$. We implement the solver *yices-ismt* with a portion of time-bound allocation

$\{0.1, 0.8, 0.1\}$ for sequential execution of $\{yices2(\phi)$, $ismt(\phi)$, $yices2(\phi \wedge \psi)\}$. If the first two end earlier, $yices2(\phi \wedge \psi)$ runs within the remaining time. *yices-ismt* returns $\{sat, unsat\}$ once *yices* or *ismt* results $\{sat, unsat\}$, and *unknown* when the time budget is exhausted.

References

1. B. Dutertre, "Yices 2.2," in *Computer Aided Verification - 26th International Conference, CAV 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 18-22, 2014. Proceedings* (A. Biere and R. Bloem, eds.), vol. 8559 of *Lecture Notes in Computer Science*, pp. 737–744, Springer, 2014.
2. D. Jovanovic and B. Dutertre, "Libpoly: A library for reasoning about polynomials," in *Proceedings of the 15th International Workshop on Satisfiability Modulo Theories affiliated with the International Conference on Computer-Aided Verification (CAV 2017), Heidelberg, Germany, July 22 - 23, 2017* (M. Brain and L. Hadarean, eds.), vol. 1889 of *CEUR Workshop Proceedings*, pp. 28–39, CEUR-WS.org, 2017.
3. A. Biere, K. Fazekas, M. Fleury, and M. Heisinger, "CaDiCaL, Kissat, Paracooba, Plingeling and Treengeling entering the SAT Competition 2020," in *Proc. of SAT Competition 2020 – Solver and Benchmark Descriptions* (T. Balyo, N. Froleyks, M. Heule, M. Iser, M. Järvisalo, and M. Suda, eds.), vol. B-2020-1 of *Department of Computer Science Report Series B*, pp. 51–53, University of Helsinki, 2020.
4. C. Fuhs, J. Giesl, A. Middeldorp, P. Schneider-Kamp, R. Thiemann, and H. Zankl, "SAT solving for termination analysis with polynomial interpretations," in *Theory and Applications of Satisfiability Testing - SAT 2007, 10th International Conference, Lisbon, Portugal, May 28-31, 2007, Proceedings* (J. Marques-Silva and K. A. Sakallah, eds.), vol. 4501 of *Lecture Notes in Computer Science*, pp. 340–354, Springer, 2007.