Yices 2 in SMT-COMP 2019

Bruno Dutertre, Dejan Jovanović,
Ian A. Mason, Stéphane Graham-Lengrand

Computer Science Laboratory, SRI International

Introduction

Yices 2 is an open-source SMT solver developed and distributed by SRI International. It is available for download at `http://yices.csl.sri.com` and on our GitHub repository at `https://github.com/SRI-CSL/yices2`. Yices 2 supports linear and non-linear arithmetic, bit-vectors, uninterpreted functions, and arrays.

Yices 2 relies on the standard CDCL(T) architecture and uses a variant of the Nelson-Oppen method for combining decision procedures. Details are presented in [1]. Yices 2 also includes a solver that implements the Model-Construction Satisfiability Calculus (MC-SAT) [5, 6]. By default, MC-SAT is used for all theories that require non-linear arithmetic and CDCL(T) is used for everything else. Yices 2 is mostly focused on quantifier-free theories, but it supports a limited form of quantifier reasoning known as exists/forall solving [2].

In the 2019 SMT competition, we are entering the latest development version of Yices 2 and several new variants that are specialized for quantifier-free bit-vector problems. Three of these variants employ the traditional "bit-blasting" method but use different backend SAT solvers. Another variant does not use bit-blasting but relies on MC-SAT.

Basic Version

The main version of Yices 2 in the competition is an evolution of the SMT solver that we entered in 2018. We plan to release the 2019 entrant as Yices-2.6.2 after the competition. This new version can display models in the SMT-LIB 2.6 format, which allows Yices 2 to enter the model-validation track, and it includes new rewriting and simplification rules (related to bit-vector problems). We have also added support for multi-threaded operations (which is not used in the competition) and fixed bugs.

We are entering this version of Yices 2 in all tracks and logics it currently supports, including the model-validation, unsat-core, and industrial-challenge tracks.

New Bit-blasting Solver

Yices 2 has supported quantifier-free bit-vectors for many years using a classic "bit-blasting" method, which converts bit-vector problems into purely Boolean problems. Until this year, Yices 2 used its own internal SAT solver as backend. This SAT solver is old (close to Minisat 1.4) and it is not competitive with modern CDCL SAT solvers. We have recently modified Yices 2 to support other SAT solvers. The implementation currently supports three backends:

Armin Biere's CaDiCal We use version `sc18` from the master branch of CaDiCal's git repository `https://github.com/arminbiere/cadical`

Mate Soos's Cryptominisat [7] We use a fork of Cryptominisat 5 that provides a new C API. The main Cryprominisat repository is at `https://github.com/msoos/cryptominisat` and our fork is at `https://github.com/BrunoDutertre/cryptominisat`.

Our own improved CDCL-based solver. This SAT solver implements known techniques from the literature (e.g., variable elimination and other forms of preprocessing, modern restart heuristics, and LBD-based estimates of clause quality).

All three variants use the same bit-blasting engine and do not support incremental solving. We are entering them in the single-query track, the model-validation track, and the industrial-challenge track.

MC-SAT For Bit Vectors

Our last entrant is based on on-going work on extending MC-SAT to bit-vector problems. We have added support for quantifier-free bitvector solving to Yices 2's MC-SAT engine. This engine works at the word-level rather than the bit level. It employs BDDs to represent sets of possible values for bit-vector variables and it uses high-level forms of reasoning for propagation and conflict explanation. The MC-SAT solver currently includes two reasoning components for fragments of the QF_BV logic. One component is specialized to reasoning about concatenation, extraction, and equalities. A second component handles a subset of linear-arithmetic constraints. When a conflict requires reasoning outside these two fragments, we use Yices 2's bit-blasting engine and unsat cores as a last resort [3, 4].

Acknowledgment

Recent Yices developments were supported by the Defense Advance Research Projects Agency (DARPA) and Space and Naval Warfare Systems Center Pacific (SSC Pacific) under Contract No. N66001-18-C-4011. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA or SSC Pacific

References

[1] Bruno Dutertre. Yices 2.2. In Armin Biere and Roderick Bloem, editors, *Computer-Aided Verification (CAV'2014)*, volume 8559 of *Lecture Notes in Computer Science*, pages 737–744. Springer, July 2014.

[2] Bruno Dutertre. Solving exists/forall problems with yices. In *13th International Workshop on Satisfiability Modulo Theories (SMT 2015)*, July 2015.

[3] Stéphane Graham-Lengrand and Dejan Jovanović. An MCSAT treatment of bit-vectors. In Martin Brain and Liana Hadarean, editors, *15th International Workshop on Satisfiability Modulo Theories (SMT 2017)*, July 2017.

[4] Stéphane Graham-Lengrand and Dejan Jovanović. Interpolating bit-vector arithmetic constraints in MCSAT. Under submission, 2019.

[5] Dejan Jovanović, Clark Barrett, and Leonardo de Moura. The design and implementation of the model model constructing satisfiability calculus. In *Formal Methods in Computer-Aided Design (FMCAD)*, pages 173–180. IEEE, October 2013.

[6] Dejan Jovanović and Leonardo de Moura. Solving non-linear arithmetic. In *International Joint Conference on Automated Reasoning*, pages 339–354. Springer Berlin Heidelberg, 2012.

[7] Mate Soos, Karsten Nohl, and Claude Castelluccia. Extending SAT solvers to cryptographic problems. In *Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings*, pages 244–257, 2009.