Yices-QS 2022,
an extension of Yices for quantified satisfiability

Stéphane Graham-Lengrand

SRI International, USA

1 Introduction

Yices-QS is a solver derived from Yices 2, an open-source SMT solver developed and distributed by SRI International. It extends Yices 2 to supports quantifiers for complete theories, and is unrelated to the support of quantifiers in Yices 2 for UF. Its core algorithm is a generalization of counterexample-guided quantifier instantiation (CEGQI) [Dut15] that can be seen as a form of lazy quantifier elimination. Yices-QS submits quantifier-free queries to Yices 2, leveraging some unique features pertaining to Yices' MCSAT solver [dMJ13, Jov17]. Yices-QS is written in OCaml and uses the OCaml bindings for the Yices 2 C API:
https://github.com/disteph/yicesQS
https://github.com/SRI-CSL/yices2_ocaml_bindings
https://github.com/SRI-CSL/yices2.

In the 2022 SMT competition, Yices-QS entered logics BV, NRA, NIA, LRA, and LIA (single-query, non-incremental tracks), extending the scope of the 2021 submission (BV and NRA). The 2022 version is commit aa67ec2 of Yices-QS, with commit 989b4ad of the OCaml bindings and commit 09f1621 of Yices 2.

2 Algorithm

Yices-QS does not modify the structure of quantifiers in formulas: it does not prenexify formulas and, more importantly, it does not skolemize them to avoid introducing uninterpreted function symbols. In that, Yices-QS departs from the standard architecture for quantifier support consisting of keeping a set of universally quantified clauses, to be grounded and sent to a core SMT-solver for ground clauses. Instead, it sees a formula as a 2-player game, in the tradition of Bjørner & Janota's *Playing with Quantified Satisfaction* [BJ15] and earlier work on QBF. Yices-QS's algorithm is designed to answer queries of the following form:

"Given a formula $A(\overline{z},\overline{x})$ and a model $\mathfrak{M}_{\overline{z}}$ on $\overline{z}$, produce either
– $\mathrm{SAT}(U(\overline{z}))$, with $U(\overline{z})$ under-approx. of $\exists\overline{x}\ A(\overline{z},\overline{x})$ satisfied by $\mathfrak{M}_{\overline{z}}$; or
– $\mathrm{UNSAT}(O(\overline{z}))$, with $O(\overline{z})$ over-approx. of $\exists\overline{x}\ A(\overline{z},\overline{x})$ not satisfied by $\mathfrak{M}_{\overline{z}}$;
where under-approximations and over-approximations are quantifier-free."

To answer such queries, Yices-QS calls Yices's feature *satisfiability modulo a model*, while the production of under- and over-approximations leverages *model interpolation* and *model generalization*.

When the input formula is in the exists-forall fragment, the algorithm degenerates to the one used in Yices' $\exists\forall$ solver, using quantifier-free solving and *model generalization*, as described in [Dut15]. *Model interpolation*, a form of which is used within MCSAT to solve quantifier-free problems, also becomes useful with more quantifier alternations than $\exists\forall$. It generalizes to non-Boolean inputs the notion of UNSAT cores, which has been used in the *quantified-problems-as-games* approach [BJ15].

3 Theory-specific aspects

- *Model interpolation* is available in Yices's MCSAT solver for quantifier-free formulas. In particular, it has theory-specific techniques for arithmetic, based on NLSAT [JdM12] (and ultimately, Cylindrical Algebraic Decomposition–CAD), and bitvectors [GLJD20].
- *Model generalization* for quantifier-free formulas can be done generically by substitutions [Dut15], but this can be complemented by theory-specific techniques that can provide better generalizations. For arithmetic, we use model-projection (based on CAD once again), in combination with generalization-by-substitution. For bitvectors, we use invertibility conditions from Niemetz et al. [NPR+18], including ϵ-terms, in combination with generalization-by-substitution. For the BV theory, the cegqi solver from [NPR+18] is probably the closest to Yices-QS, which approaches BV as an instance of the theory-generic algorithm from Section 2.

Note: For NRA, the presence of division in benchmarks departs from the theoretic applicability of Yices-QS's algorithm for complete theories, because of division-by-zero (which also makes the theory undecidable). Yices's CAD-based model-projection in NRA does not support division. In practice, when Yices-QS needs to perform model generalization with a formula involving division, it cannot use CAD model-projection and resorts to generalization-by-substitution. Resorting to generalization-by-substitution for NRA also means that Yices-QS's algorithm may not terminate.

2022 vs 2021: Beyond a few bug fixes and optimizations, the 2022 version differs from the 2021 one in that it supports algebraic numbers in the generalization-by-substitution approach. The 2022 version can substitute a variable by its algebraic value in the model (represented with an ϵ-term), while the 2021 version would fail and give up.

References

BJ15. N. Bjørner and M. Janota. Playing with quantified satisfaction. In M. Davis, A. Fehnker, A. McIver, and A. Voronkov, editors, *Proc. of the the 20th Int. Conf. on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'15)*, volume 9450 of *LNCS*. Springer-Verlag, 2015. https://doi.org/10.1007/978-3-662-48899-7 1, 2

dMJ13. L. M. de Moura and D. Jovanovic. A model-constructing satisfiability calculus. In R. Giacobazzi, J. Berdine, and I. Mastroeni, editors, *Proc. of the 14th Int. Conf. on Verification, Model Checking, and Abstract Interpretation (VMCAI'13)*, volume 7737 of *LNCS*, pages 1–12. Springer-Verlag, 2013. https://doi.org/10.1007/978-3-642-35873-9_1 1

Dut15. B. Dutertre. Solving exists/forall problems with yices. In *13th International Workshop on Satisfiability Modulo Theories (SMT 2015)*, 2015. `https://yices.csl.sri.com/papers/smt2015.pdf`. 1, 2

GLJD20. S. Graham-Lengrand, D. Jovanović, and B. Dutertre. Solving bitvectors with MCSAT: explanations from bits and pieces. In N. Peltier and V. Sofronie-Stokkermans, editors, *Proceedings of the 10th International Joint Conference on Automated Reasoning (IJCAR'20)*, volume 12166(1) of *Lecture Notes in Computer Science*, pages 103–121. Springer-Verlag, 2020. 2

JdM12. D. Jovanović and L. de Moura. Solving non-linear arithmetic. In B. Gramlich, D. Miller, and U. Sattler, editors, *Proc. of the 6th Int. Joint Conf. on Automated Reasoning (IJCAR'12)*, volume 7364 of *LNCS*, pages 339–354. Springer-Verlag, 2012. 2

Jov17. D. Jovanović. Solving nonlinear integer arithmetic with MCSAT. In A. Bouajjani and D. Monniaux, editors, *Proc. of the 18th Int. Conf. on Verification, Model Checking, and Abstract Interpretation (VMCAI'17)*, volume 10145 of *LNCS*, pages 330–346. Springer-Verlag, 2017. https://doi.org/10.1007/978-3-319-52234-0_18 1

NPR+18. A. Niemetz, M. Preiner, A. Reynolds, C. W. Barrett, and C. Tinelli. Solving quantified bit-vectors using invertibility conditions. In H. Chockler and G. Weissenbacher, editors, *Proc. of the 30th Int. Conf. on Computer Aided Verification (CAV'18)*, volume 10982 of *LNCS*, pages 236–255. Springer-Verlag, 2018. https://doi.org/10.1007/978-3-319-96142-2_16 2