YicesQS 2025,
an extension of Yices for quantified satisfiability

Stéphane Graham-Lengrand

SRI International, USA

YicesQS is a solver derived from Yices 2, an open-source SMT solver developed and distributed by SRI International. It extends Yices to supports quantifiers for complete theories, and is unrelated to the support of quantifiers in Yices 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. YicesQS submits quantifier-free queries to one of Yices's core solvers CDCL(T) or MCSAT [dMJ13]. It is written in OCaml and uses the OCaml bindings for the Yices C API; it indirectly relies on the libpoly library for arithmetic. In the 2024 SMT competition, YicesQS entered logics BV, NRA, NIA, LRA, and LIA (single-query, non-incremental tracks), as in 2023 and 2022. The exact commits are as follows:

`https://github.com/disteph/yicesQS`	`commit 56ab860`
`https://github.com/SRI-CSL/yices2_ocaml_bindings`	`commit 4af4bee`
`https://github.com/SRI-CSL/yices2`	`commit 98a0d52`
`https://github.com/SRI-CSL/libpoly_ocaml_bindings`	`commit 8e9dc78`
`https://github.com/SRI-CSL/libpoly`.	`commit 331cdff`

and the exact command is

```
./main.exe -under 20 -cdclT-mcsat 700 $1
```

Algorithmic approach. The main algorithm in YicesQS is QSMA [BGLV23]. It 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. This 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, YicesQS 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. YicesQS'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
- 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
- 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, YicesQS 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 CEGQI one used in Yices' $\exists\forall$ solver [Dut15] using quantifier-free solving and *model generalization*. *Model* *interpolation*, a form of which is used within Yices's MCSAT solver to solve quantifier-free problems [JD21], also becomes useful with more quantifier alternations than ∃∀. It generalizes to non-Boolean inputs the notion of UNSAT cores, which has been used in the *quantified-problems-as-games* approach [BJ15].

Arithmetic (NRA, NIA, LRA, LIA). *Solving Modulo a Model*: We use Yices's MCSAT solver; the MCSAT approach is inspired by CDCL, building a tentative solution model on a trail where Boolean and theory assignments are decided, propagated, and backtracked upon. MCSAT can natively take a partial model as input [JD21]. *Model interpolation*: We use the model interpolants natively produced by Yices's MCSAT solver. A lemma that is learnt at decision level 0, defeating the input model, constitutes a model interpolant. Learnt lemmas arise from theory-specific mechanisms for explaining conflicts, which in the case of arithmetic is leveraging Cylindrical Algebraic Decomposition (CAD) [JdM12, Jov17]. *Model generalization*: We mainly use model-projection (based on CAD once again). For NRA, the presence of division in benchmarks departs from the theoretic applicability of YicesQS'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. When YicesQS needs to perform model generalization with a formula involving division, it cannot use CAD model-projection and resorts to generalization-by-substitution, which is a generic mechanism already used in Yices's ∃∀ solver [Dut15]. Resorting to generalization-by-substitution for NRA also means that YicesQS's algorithm may not terminate.

Arithmetic (LIA). In 2025, we make two runs in a sequential portfolio; the second one leverages MCSAT as for the other arithmetic logics. The first one leverages Yices's CDCL(T) solver as follows. *Solving Modulo a Model*: We represent the partial input model as equalities between integer variables and their values, and given to Yices as assumptions. *Model interpolation*: We simply use UNSAT cores produced by Yices's CDCL(T) solver. *Model generalization*: as for the other arithmetic logics.

Bitvectors (BV). As for LIA, we make two runs in a sequential portfolio, starting with the version relying on CDCL(T) as the quantifier-free solver, which in this logic relies on bitblasting and, ultimately, Yices's internal SAT-solver. The second run uses Yices's MCSAT as the quantifier-free solver, which support quantifier-free bitvectors [GLJD20]. In both cases, *Model generalization* uses 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 YicesQS.

Changes since 2024. There are no changes in YicesQS since the 2024 SMT-competition, except for the use of the CDCL(T) solver for LIA. There are some changes in the underlying Yices 2 quantifier-free solver that may affect performance of YicesQS.

References

BGLV23. M. P. Bonacina, S. Graham-Lengrand, and C. Vauthier. QSMA: A new algorithm for quantified satisfiability modulo theory and assignment. In B. Pientka and C. Tinelli, editors, *Proc. of the 29th Int. Conf. on Automated Deduction (CADE'23)*, volume 14132 of *LNAI*. Springer-Verlag, 2023. 1

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

JD21. D. Jovanovic and B. Dutertre. Interpolation and model checking for nonlinear arithmetic. In A. Silva and K. R. M. Leino, editors, *Computer Aided Verification - 33rd International Conference, CAV 2021, Virtual Event, July 20-23, 2021, Proceedings, Part II*, volume 12760 of *Lecture Notes in Computer Science*, pages 266–288. Springer, 2021. https://doi.org/10.1007/978-3-030-81688-9_13 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 2

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