Ultimate Eliminator: a Quantifier Upgrade for SMT Solvers at SMT-COMP 2019

Max Barth, Daniel Dietsch, Leonard Fichtner, and Matthias Heizmann

University of Freiburg, Germany

Abstract

Ultimate Eliminator is a tool for eliminating quantifiers in SMT formulas. For the SMT-COMP 2019 we developed a user interface that takes SMT-LIB 2.6 compatible input tries to eliminate quantifiers in this input and passes the modified input to an SMT-LIB 2.6 compatible solver. This paper presents three solvers that we submitted to the competition, each of them consists of an SMT solver that does not support quantifiers and that is wrapped with Ultimate Eliminator.

1 Overview

Quantified formulas are notoriously difficult to solve and several state-of-the-art SMT solvers support only quantifier-free formulas. We found out that, perhaps surprisingly, the quantifier elimination algorithms that are implemented in the Ultimate software verification framework can often find quantifier-free formulas that are logically equivalent to formulas in SMT-LIB benchmarks. Hence, these quantifier elimination algorithms empower existing SMT solvers without support for quantifiers to solve quantified formulas. E.g., a formula of the form $\exists x.\varphi(x) \wedge x = t$ is transformed to $\varphi(t)$ and the formula $\forall a.\mathit{select}(a,k) = \mathit{select}(a,i)$ is transformed to $k = i$.

2 Quantifier Elimination in Ultimate

A key algorithm [2] of several software verifiers [4] in the Ultimate framework [1] does an iterative application of the strongest post predicate transformer to a sequence of statements resp. an iterative application of the weakest precondition predicate transfomer. Both variants of the algorithm produce quantified formulas and the handling of these quantified formulas often was a bottleneck for the overall software verification approach. The performance of the tool improved significantly when the developers started to apply quantifier elimination techniques to every intermediate result. As a consequence, in the last years an increasing number of quantifier elimination techniques was implemented into the Ultimate framework.

3 Ultimate Eliminator

Ultimate Eliminator[2] is a user interface that takes SMT-LIB 2.6 compatible input tries to eliminate quantifiers in this input and passes the modified input to a user defined SMT-LIB 2.6 compatible solver. Ultimate Eliminator is implemented in Java as a plug-in of the Ultimate framework. The source code is available in a public repository[3].

[1] https://ultimate.informatik.uni-freiburg.de/

[2] https://ultimate.informatik.uni-freiburg.de/eliminator/

[3] https://github.com/ultimate-pa/ultimate/

4 SV-COMP 2019 Submissions

We submitted three SMT solvers to the competition. Each of them wraps an SMT solver that does not support quantifiers. For each logic that the wrapped solver (to the best of our knowlege) supports, we participated in the corresponding logic with quantifiers.

- Our UltimateEliminator+MathSAT-5.5.4 submission wraps the version 5.5.4 of the MathSAT SMT solver[4][1]. We call MathSAT without any additonal arguments.
- Our UltimateEliminator+SMTInterpol submission wraps version 2.5-102-g6cab3187 of the SMTInterpol SMT solver[5][5]. This version of SMTInterpol is integrated in the Ultimate Framework and called via the Java interface without any additional options.
- Our UltimateEliminator+Yices-2.6.1 submission wraps the version 2.6.1 of the Yices SMT solver[6][3]. We use the command `yices-smt2 --incremental` to call Yices.

In all three solvers the version of Ultimate was 0.1.24-45d5f27-m. All three solvers participate in the Single Query Track, in the Incremental Track, and in the Unsat Core Track. We did not participate in other tracks because the other tracks use only quantifier-free benchmarks.

References

[1] Alessandro Cimatti, Alberto Griggio, Bastiaan Joost Schaafsma, and Roberto Sebastiani. The mathsat5 SMT solver. In *TACAS*, volume 7795 of *Lecture Notes in Computer Science*, pages 93–107. Springer, 2013.

[2] Daniel Dietsch, Matthias Heizmann, Betim Musa, Alexander Nutz, and Andreas Podelski. Craig vs. newton in software model checking. In *ESEC/SIGSOFT FSE*, pages 487–497. ACM, 2017.

[3] Bruno Dutertre. Yices 2.2. In *CAV*, volume 8559 of *Lecture Notes in Computer Science*, pages 737–744. Springer, 2014.

[4] Matthias Heizmann, Yu-Fang Chen, Daniel Dietsch, Marius Greitschus, Jochen Hoenicke, Yong Li, Alexander Nutz, Betim Musa, Christian Schilling, Tanja Schindler, and Andreas Podelski. Ultimate automizer and the search for perfect interpolants - (competition contribution). In *TACAS (2)*, volume 10806 of *Lecture Notes in Computer Science*, pages 447–451. Springer, 2018.

[5] Jochen Hoenicke and Tanja Schindler. Solving and interpolating constant arrays based on weak equivalences. In *VMCAI*, volume 11388 of *Lecture Notes in Computer Science*, pages 297–317. Springer, 2019.

[4] http://mathsat.fbk.eu/

[5] https://ultimate.informatik.uni-freiburg.de/smtinterpol/

[6] http://yices.csl.sri.com/