Ultimate IntBlastingWrapper

Max Barth and Matthias Heizmann

University of Freiburg, Germany

Abstract

This system description presents ULTIMATE INTBLASTINGWRAPPER+SMTINTERPOL which is our participant at the SMT-COMP 2023. This tool is an SMT solver for bitvector logics. It tries to translate bitvector formulas into equisatisfiable integer formulas and applies the SMT solver SMTINTERPOL to the integer formulas.

1 Overview

ULTIMATE INTBLASTINGWRAPPER, or short INTBLASTINGWRAPPER, is an SMT solver for the theory of fixed-sized bitvectors. It is a wrapper tool, i.e., a tool that calls an other SMT solver. Our tool tries to translate bitvector formulas into equisatisfiable integer formulas. The integer formulas are then passed to the wrapped solver which has to be a solver that supports at least the theory of linear integer arithmetic. At the SMT-COMP 2023 the wrapped SMT solver is SMTINTERPOL[2] and hence the full name of our participant is ULTIMATE INTBLASTINGWRAPPER+SMTINTERPOL. The version of the included SMTINTERPOL is 2.5-1252-g82eb3a0.

2 Int-Blasting

The classical approach for reasoning in the theory of fixed-sized bitvectors is called *bit-blasting*. Here, each bit of the bitvector is translated to a propositional logical formula and this formula is passed to a SAT solver. Our tool implements a completely different approach [4, 5, 3, 7, 1, 6] in which bitvectors are considered as the binary encoding of an integer and bitvector formulas are translated to nonlinear integer arithmetic formulas that extensively use modulo operations. In analogy to the term bit-blasting, we call this translation int-blasting. Bit-blasting is effective, every operation from the theory of fixed-sized bitvectors can be translated into a Boolean formula. However, bit-blasting does not scale well for large bitvectors. Independent of the bitvector's width, int-blasting is straightforward for arithmetic operations. However, int-blasting is difficult for bitwise operations (e.g., `bvand`). Our tool implements a novel variation of int-blasting that has not yet been published.

While working on software verification, one application for fixed-sized bitvectors, we observed that bitwise operations often play only a minor role in the SMT reasoning. The same holds to some extend also for the SMT-LIB benchmarks, perhaps because many of these stem from software verification. The aim our submission is to demonstrate that our variation of int-blasting is effective on many SMT benchmarks. We participate only in the Single Query Track.

3 Software Project

Our tool is part of the ULTIMATE program analysis framework[1]. The source code is available in a public repository[2].

[1] https://ultimate-pa.org/

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

References

[1] Peter Backeman, Philipp Rümmer, and Aleksandar Zeljic. Bit-vector interpolation and quantifier elimination by lazy reduction. In Nikolaj S. Bjørner and Arie Gurfinkel, editors, *2018 Formal Methods in Computer Aided Design, FMCAD 2018, Austin, TX, USA, October 30 - November 2, 2018*, pages 1–10. IEEE, 2018.

[2] Jürgen Christ, Jochen Hoenicke, and Alexander Nutz. Smtinterpol: An interpolating SMT solver. In Alastair F. Donaldson and David Parker, editors, *Model Checking Software - 19th International Workshop, SPIN 2012, Oxford, UK, July 23-24, 2012. Proceedings*, volume 7385 of *Lecture Notes in Computer Science*, pages 248–254. Springer, 2012.

[3] Alberto Griggio. Effective word-level interpolation for software verification. In Per Bjesse and Anna Slobodová, editors, *International Conference on Formal Methods in Computer-Aided Design, FMCAD '11, Austin, TX, USA, October 30 - November 02, 2011*, pages 28–36. FMCAD Inc., 2011.

[4] Arie Gurfinkel, Anton Belov, and João Marques-Silva. Synthesizing safe bit-precise invariants. In Erika Ábrahám and Klaus Havelund, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, volume 8413 of *Lecture Notes in Computer Science*, pages 93–108. Springer, 2014.

[5] Yuandong Cyrus Liu, Chengbin Pang, Daniel Dietsch, Eric Koskinen, Ton-Chanh Le, Georgios Portokalidis, and Jun Xu. Source-level bitwise branching for temporal verification of lifted binaries. *CoRR*, abs/2105.05159, 2021.

[6] Aina Niemetz, Mathias Preiner, Andrew Reynolds, Yoni Zohar, Clark W. Barrett, and Cesare Tinelli. Towards bit-width-independent proofs in SMT solvers. In Pascal Fontaine, editor, *Automated Deduction - CADE 27 - 27th International Conference on Automated Deduction, Natal, Brazil, August 27-30, 2019, Proceedings*, volume 11716 of *Lecture Notes in Computer Science*, pages 366–384. Springer, 2019.

[7] Takamasa Okudono and Andy King. Mind the gap: Bit-vector interpolation recast over linear integer arithmetic. In Armin Biere and David Parker, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 26th International Conference, TACAS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25-30, 2020, Proceedings, Part I*, volume 12078 of *Lecture Notes in Computer Science*, pages 79–96. Springer, 2020.