Proceedings of SAT Competition 2022: Solver and Benchmark Descriptions, volume B-2022-1 of Department of Computer Science Series of Publications B, University of Helsinki 2022.

Verifying Linked List Safety Properties in AWS C99 Package with CBMC

Muhammad Osama and Anton Wijs
Department of Mathematics and Computer Science
Eindhoven University of Technology, Eindhoven, The Netherlands
{o.m.m.muhammad, a.j.wijs}@tue.nl

***Abstract*—In this paper, state-of-the-art proofs are generated with harness using the CBMC bounded model checker for the Amazon Web Services C99 core package. In this submission, we check the safety properties of the *Linked List swap-contents* routine with various *loop unwinding* settings as opposed to the *String compare* submitted last year. The generated proof has proven to be reasonably hard to solve using modern SAT solvers. It has many variable-clause redundancies which are not only challenging for a SAT solver but also useful to assess the performance of different simplification techniques.**

I. Introduction

Bounded Model Checking (BMC) [1]–[3] determines whether a model M satisfies a certain property φ expressed in temporal logic, by translating the model checking problem to a propositional satisfiability (SAT) problem or a Satisfiability Modulo Theories (SMT) problem. The term *bounded* refers to the fact that the BMC procedure searches for a counterexample to the property, i.e., an execution trace, which is bounded in length by an integer k. If no counterexample up to this length exists, k can be increased and BMC can be applied again. This process can continue until a counterexample has been found, a user-defined threshold has been reached, or it can be concluded (via k-induction [2]) that increasing k further will not result in finding a counterexample. CBMC [4], [5] is an example of a successful BMC model checker that uses SAT solving. CBMC can check ANSI-C programs. The verification is performed by *unwinding* the loops in the program under verification a finite number of times, and checking whether the bounded executions of the program satisfy a particular safety property [6]. These properties may address common program errors, such as null-pointer exceptions and array out-of-bound accesses, and user-provided assertions.

II. Benchmarks

In this paper, we are interested in verifying the safety properties of the *swap-contents* routine implemented in the Linked List data structure of the Amazon Web Services (AWS) C99 core package. The proof covers the following:

- Memory allocation failure and access violations
- Pointer/floating-point overflow
- Data types conversion

We generated 30 different formulas using a *loop unwinding* upper-bound in the range $[40, 80]$, with an iincremental step. These bounds produce SAT formulas with 100% coverage of all functionalities. All problems are written in this format: `linked_list_swap_contents_safety_unwind<x>` where `x` denotes the unwinding value. The first and the last formulas are solved via MiniSat [7] within 90 and 540 seconds respectively on a machine with AMD EPYC 7H12 64-Core processor operating at base clock of 2.6 GHz. The solving time of the rest of the benchmarks are expected to be monotonically increasing.

References

[1] A. Biere, A. Cimatti, E. M. Clarke, and Y. Zhu, "Symbolic Model Checking without BDDs," in *Proc. of TACAS (Mar. 1999), Amsterdam, The Netherlands*, ser. LNCS, vol. 1579. Springer, 1999, pp. 193–207.

[2] A. Biere, A. Cimatti, E. M. Clarke, O. Strichman, and Y. Zhu, "Bounded model checking," *Advances in Computers*, vol. 58, pp. 117–148, 2003.

[3] M. Osama and A. Wijs, "GPU Acceleration of Bounded Model Checking with ParaFROST," in *Proc. of CAV (Jul. 2021), USA*, ser. LNCS, vol. 12760. Springer, 2021, pp. 447–460.

[4] E. M. Clarke, D. Kroening, and F. Lerda, "A Tool for Checking ANSI-C Programs," in *Proc. of TACAS (Mar. 2004), Barcelona, Spain*, ser. LNCS, vol. 2988. Springer, 2004, pp. 168–176.

[5] D. Kroening and M. Tautschnig, "CBMC - C Bounded Model Checker - (Competition Contribution)," in *Proc. of TACAS (Apr. 2014), Grenoble, France*, ser. LNCS, vol. 8413. Springer, 2014, pp. 389–391.

[6] D. Kroening and O. Strichman, *Decision Procedures - An Algorithmic Point of View, Second Edition*, ser. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2016.

[7] N. Eén and N. Sörensson, "An Extensible SAT-solver," in *Proc. of SAT (May 2003) Santa Margherita Ligure, Italy*, ser. LNCS, vol. 2919. Springer, 2003, pp. 502–518.