Summary of Model Checking C++ Programs

Felipe R. Monteiro
Federal University of Amazonas
Manaus, Amazonas, Brazil
felipemonteiro@ufam.edu.br

Mikhail R. Gadelha
Igalia
A Coruña, Spain
mikhail.ramalho@gmail.com

Lucas C. Cordeiro
University of Manchester
Manchester, UK
lucas.cordeiro@manchester.ac.uk

***Abstract*—This is an extended abstract of the article "Model Checking C++ Programs" by Felipe R. Monteiro, Mikhail R. Gadelha, and Lucas C. Cordeiro. We describe and evaluate a novel verification approach based on bounded model checking (BMC) and satisfiability modulo theories (SMT) to verify C++ programs. Our verification approach analyzes bounded C++ programs by encoding into SMT various sophisticated features that the C++ programming language offers, such as templates, inheritance, polymorphism, exception handling, and the Standard Template Libraries. We implemented our verification approach on top of ESBMC. We compare ESBMC to LLBMC and DIVINE, which are state-of-the-art verifiers to check C++ programs directly from the LLVM bitcode. Experimental results show that ESBMC can handle a wide range of C++ programs, presenting a higher number of correct verification results. Additionally, ESBMC has been applied to a commercial C++ application in the telecommunication domain and successfully detected arithmetic-overflow errors, which could lead to security vulnerabilities.**

***Index Terms*—C++, memory safety, model checking, SMT, software verification**

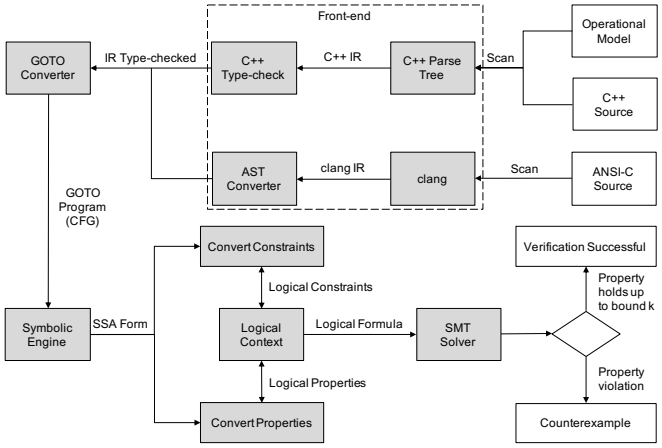

Fig. 1. ESBMC architectural overview. White rectangles represent input and output; gray rectangles represent the steps of the verification.

I. Overview

Over the last 15 years, formal techniques dramatically evolved, its adoption in industry has been growing, and several tools to formally verify C programs have been proposed. However, there exist only a few attempts with limited success to cope with the complexity of C++ program verification. The main challenge here is to support sophisticated features that the C++ programming language offers, such as templates, sequential and associative template-based containers, strings & streams, inheritance, polymorphism, and exception handling.

In "*Model Checking C++ Programs*" [1], we describe and evaluate a novel SMT-based BMC approach to verify C++ programs integrated into ESBMC [2], a state-of-the-art context-bounded model checker. ESBMC can check for undefined behaviors and memory safety issues such as under- and overflow arithmetic, division-by-zero, pointer safety, array out-of-bounds violations, and user-defined assertions.

The article starts with an overview of ESBMC's type-checking engine, which includes our approach to support templates (similar to conventional compilers) that replaces the instantiated templates before the encoding phase. It also describes the type-checking mechanism to handle single and multiple inheritance and polymorphism in C++ programs. It then presents the significant contributions of this work: the C++ operational models (COM) and the support for exception handling. It describes an abstraction of the Standard Template Libraries (STL), which replaces them during the verification process to reduce complexity while checking whether a given program uses the STL correctly. Finally, it presents novel approaches to handle critical features of exception handling in C++ (e.g., unexpected and termination function handlers). We also compare ESBMC against LLBMC, a state-of-the-art bounded model checker based on SMT solvers, and DIVINE, a state-of-the-art explicit-state model checker, both for C and C++ programs. Our experimental evaluation contains a broad set of benchmarks with over $1,500$ instances, where ESBMC reaches a success rate of 84.27% (in approximately 4 hours), outperforming LLBMC and DIVINE.

For future work, we intend to extend ESBMC coverage in order to verify C++11 programs, rewrite our front-end using clang to generate the program abstract syntax tree (AST), and develop a conformance testing procedure to ensure that our COM conservatively approximates the STL semantics.

This research was partially funded by the EPSRC grants EP/T026995/1, EP/V000497/1, EU H2020 ELEGANT 957286, Nokia Institute of Technology (INdT), and Soteria project awarded by the UK Research and Innovation for the Digital Security by Design (DSbD) Programme.

References

[1] F. R. Monteiro, M. R. Gadelha, and L. C. Cordeiro, "Model checking c++ programs," *Software Testing, Verification and Reliability*, vol. 32, no. 1, p. e1793, 2022.

[2] M. R. Gadelha, F. R. Monteiro, J. Morse, L. C. Cordeiro, B. Fischer, and D. A. Nicole, "ESBMC 5.0: An industrial-strength C model checker," in *Automated Software Engineering*, 2018, pp. 888–891.