Performance Analysis of Undefined Behavior Optimizations

Wednesday 1st March, 2023

Abstract

State-of-the-art compilers, such as Clang/LLVM, use undefined behavior (UB) to issue optimizations. In this poster session, we present the impact of UB optimizations for a diverse set of application categories. We use Phoronix to test this impact in the form of total execution time, FPS, transactions per second, etc. With the recent proposal in the C++ Comittee of introducing errnoeous behavior (EB) as an alternative for UB, the results can be used to better understand what behaviors should be kept UB and what behaviors can be safely moved to EB based on performance effects.

1 Context and Motivation

The C and C++ Standards [2, 3] provide a definition of undefined behavior that gives absolute freedom to compiler implementation when erroneous program constructs, erroneous data or indeterminately-valued objects are encountered. This allows Clang/LLVM to treat undefined behavior in various ways while still being standard conformant.

This allows the compiler implementation to use the definition of undefined behavior to issue optimizations. However, the performance impact of such optimizations is not clear. With the recent proposal [4] in the C++ Comittee of introducing erroneous behavior in the Standard this problem gathers more attention. There is no data that shows what are the most critical undefined behaviors for performance.

In this context we take a set of application categories and assess the performance impact of undefined behavior optimizations. The categories are the following: webservers, circuit simulators, telephony, finance, GUI, software defined radio, speech, compression, texture compression, audio encoding, databases, chess, password cracking, cryptography, security, parallel processing, image processing, bioinformatics, simulation, video encoding, neural networks, HPC and compiler build speed.

These categories are the result of a fine grained analysis of the benchmarks provided by Phoronix Test Suite [1]. We use this benchmark framework as it provides support for a wide range of applications and a fast and mature interface for running the benchmarks.

We test the appplication categories against different undefined behavior optimization flags such as: -fwrapv, -fno-strict-aliasing, -fstrict-enums, etc.

To display the results, we take each application category and present a comparison between the performance of compiling the category with various undefined behavior optimization flags. The performance may be measured in execution time, FPS, transictions per second, etc.

References

[1] Phoronix test suite. `https://www.phoronix-test-suite.com/`, last visited Wednesday 1st March, 2023.

[2] ISO. *ISO/IEC 14882:2017 Information technology — Programming languages — C++*. Fifth edition, Dec. 2017.

[3] ISO. *ISO/IEC 9899:2018 Information technology — Programming languages — C*. Fourth edition, June 2018.

[4] T. Köppe. Correct and incorrect code, and "erroneous behaviour". In *ISO/IEC JTC1 SC22 WG21 D2795R0*, 2023. `https://isocpp.org/files/papers/D2795R0.html`, last visited Wednesday 1st March, 2023.