Qeios

Peer Review

Review of: "Finding Missed Code Size Optimizations in Compilers using LLMs"

Hyungjoon Koo[1]

1. Computer Science and Engineering, Sungkyunkwan University, Seoul, South Korea

This paper introduces a novel technique combining large language models (LLMs) with differential testing strategies for finding bugs. The proposed approach is simple (e.g., fewer than 150 LoC), but the results look promising (e.g., 24 bugs were discovered and confirmed). Besides, the approach is extensible to languages such as Rust and Swift.

However, I have several concerns as follows.

First, the main contribution of this paper is to find bugs in the existing C/C++ compilers (e.g., gcc, clang) with LLMs, as supported by Section 3. However, the title (and the abstract) emphasizes finding missed code size optimizations in compilers using LLMs. I am not confident about what points the authors attempt to claim regarding this point. Please clarify it.

Second, in Figure 1, the steps of mutating a program are inherently not deterministic. Is it possible to discover a similar bug with different orders of each step (e.g., step 1 after step 2)? Also, Step 5 has many different ways of generating samples when adding a non-trivial condition. Does this mean the proposed technique to discover a bug may rely mainly on the sample generation?

Third, some mutations might not be compilable due to syntactic errors (e.g., incorrect syntax). How can this approach check if the mutations are accurate before performing differential testing?

doi.org/10.32388/XI0BSG

Declarations

Potential competing interests: No potential competing interests to declare.