Towards x86 Instruction Set Emulation in Java via Project-based Text-to-Code Generation using Reinforcement Learning

2210422 TRAN, Thu Thi Anh

Malware analysis by formal methods using Control Flow Graphs (CFGs) has been proved to be more effective than conventional signature-based strategies. To reconstruct a CFG of a given program, Dynamic Symbolic Execution (DSE) techniques are often used. The implementation of a DSE tool must strictly comply with the specifications of its designated architecture - the Instruction Set Architecture (ISA) manual. As there is a number of computer processor families with each has several variations and editions, fully manual DSE tools construction certainly demands extensive engineering work. To help reduce human effort, tasks such as environment emulation and instruction set emulation can be semi/fully automated with the help of natural language processing techniques.

The semi-automated approach of such tasks includes two steps: extracting semantics from natural language text of the ISA manual and mapping them into a prepared code template tailored to the platform that constructs the DSE tool. Two notable DSE tools which are BEPUM (Binary Emulation for PUshdown Model) for x86 architecture and CORANA for ARM architecture employs semi-automatic instruction set emulation. It is reported that BE-PUM successfully generates Java code implementation for 56.41% of 530 selected x86 instructions and CORANA scores at 63.72% of 1039 ARM - Cortex M instructions in 5 variations. While achieving promising emulation results, this approach still requires manual preparation of both interpretation rules for semantic extraction and project-based code templates. Although the current progress of BE-PUM and CORANA shows that the amount of human effort spent on the manual preparation is minimal compared to the traditional workload, it is evidence that to yield higher results than those does demand greater human labor.

The fully automated approach eliminates the need for rule preparation, concentrating instead on end-to-end text-to-code generation. In this study, we explore the feasibility of this approach by developing CoDeb system which aims at applying reinforcement learning to large language models for fully-automatic emulation of x86 instruction set based on its description in natural language, utilizing feedback from compiler and the existing Java codebase of BE-PUM project. As a result, the performance of this method would not be bounded by human effort. However, the quality of the automatically generated codes must meet standard requirements, including syntactical and semantic correctness.

The scope of our study focuses on ensuring project-level syntactical correctness via successful compilation. This requires that the generated code is valid within the project-level context of BE-PUM, meaning it must correctly utilize the existing code base, including function calls, variable names, and data types. In our work, we adopt two generative models, one acts as a code writer (Coder) and the other as a code debugger (Debugger), hence the name CoDeb. Additionally, the code base knowledge of BE-PUM project is built into separate vector database which serves as syntax references for the generative models. To eliminate the need for manual work spent on preparing labelled dataset or coding examples, we approach via almost-zero-shot generation by preparing a small code template and employing a set of rule-based feedback and compiler feedback to help iteratively improve the generation through reinforcement learning with Proximal Policy Optimization. Out of 200 selected x86 instructions, CoDeb's best attempt successfully generates project-level compilable code for 20 instructions, achieving a 10% success rate. Due to time and computing resource constraints, only this attempt (among other experimental trials) completed a total of 1,147 instructions, achieving a 14.39% success rate with 165 successfully compiled instructions. Compared to the baseline of semi-automatic approaches, our work, though with modest results, shows promising potential for application and adaptability.