Implementation Report: Program Synthesis using Symbolic Execution

Kartik Jain

231110023

October 5, 2023

1 Introduction

The Program Synthesis using Symbolic Execution tool is designed to find constant assignments to variables in program P1 such that it becomes semantically equivalent to program P2. This report outlines the implementation, assumptions, and limitations of the tool.

2 Implementation Details

The tool is implemented in Python and relies on the Z3 solver for symbolic execution and constraint solving. It comprises several modules, including `sExecutionInterface.py`, `z3solver.py`, `irgen.py`, `interpreter.py`, and `ast.py`, each responsible for specific functionalities like symbolic variable addition, constraint handling, and symbolic encoding.

3 Assumptions

- The tool assumes that programs P1 and P2 are provided in a specific format compatible with the tool's interface, and the functions `example(s)` and `checkEq(args, ir)` are appropriately defined.
- It assumes that the symbolic variables, constraints, and assignments are specified using the correct syntax as demonstrated in the provided code snippet.

- The tool assumes that the input programs are free from syntax errors and conform to the expected format.

4 Limitations

- This implementation works for only few variables(max 6) and for limited number of holes (max 5), This tool can also be modified further to work with many variables,
- The tool may not handle certain complex program structures or specific types of constraints effectively. For example, intricate loop structures may lead to incomplete or incorrect results.
- It may struggle with large-scale programs or those with a high number of variables, as the symbolic execution process can be computationally expensive.
- The tool may not provide meaningful results for programs with non-linear or highly complex arithmetic operations.
- Additionally, if the input programs contain unsupported language features or constructs, the tool may fail to produce accurate results.

5 Bug Reports and Limitations

If any errors or missing features were encountered during evaluation, they are considered bugs. Any known limitations of the tool are listed below:

- The tool may not correctly handle specific types of constraints or expressions, leading to inaccurate results.
- In some cases, the tool may not provide a solution even if one exists due to limitations in the symbolic execution process.