CS335 Compiler Design Project
ProofEngine

Project Idea

Build a translator to convert Chiron (Kachhua) IR to SMTLIB for program verification.

Detailed Problem Statement

Our aim is to create a translator which takes a Chiron IR and a satisfiability constraints file as input and generates an SMTLIB code as output. This output can then be given to an SMT solver for verification. The solver's result can be used to prove (or disprove) the correctness of the program. If there is a bug in the program, the solver can also determine an input on which the program produces an incorrect behavior.

For example, suppose we want to check if the turtle does not move outside a particular area for any input. In such a case, we can write suitable constraints, and the solver can verify whether there exists any input for which the constraints are violated.

Work Description

1. Design and define the syntax for the satisfiability constraints file.
2. Implement a parser to interpret the satisfiability constraints file accurately.
3. Research and understand the SMTLIB format to enable precise code generation.
4. Develop a module to generate optimized SMTLIB code from Chiron IR and satisfiability constraints.
5. Integrate an SMT solver (such as Z3 SMT solver) to validate the generated SMTLIB code and check for constraint satisfiability.
6. Write unit tests for each module and ensure coverage for all incremental features of the translator.
7. Document the Chiron IR specifications and define the mapping rules to SMTLIB constructs.
8. Develop detailed user documentation and guides for writing constraints files and interpreting solver outputs.

Milestones

- **5th Feb 2025:** Design and formalize the syntax for writing satisfiability constraints files. Implement a parser to accurately interpret and validate these files. Simultaneously, research and understand the SMTLIB format for translating constraints and Chiron IR into SMTLIB.
- **19th Feb 2025:** SMTLIB code generation for assign statements.
- **20–28 Feb 2025:** Mid-semester exams.
- **10th Mar 2025:** SMTLIB code generation for conditionals, arithmetic, and repeat statements.
- **20th Mar 2025:** SMTLIB code generation for move operations and penup-pendown operations.
- **25th Mar 2025:** Solver integration and overall testing.
- **31st Mar 2025:** Documentation of code.