Project Abstract

Wesley Nuzzo

1 Description

In a previous class (Compiler Construction) I created a compiler in Python from a very simple language with LISP-like syntax into working x86 assembly. The compiler contained 5 passes, each with unit tests and with interpreters for the intermediate languages.

One of the weakness of the testing approach I used in that compiler is that it only tests the specific test cases I came up with; there may still be blind spots and edge cases I'm not aware of.

A more robust approach is called "Property-Based Testing", where a programmer specifies the properties they want a function to have, and provide a way to generate random inputs to the function, and the test suite then tries to find counterexamples to that property. Once a counterexample is found, a number of strategies are used to simplify it down to the most simple/readable counterexample it can find.

Python has a library called Hypothesis which allows for Property-Based Testing.

2 Summary

My goal for this project is to use Hypothesis to create a property-based test suite for each pass of the compiler I built before.

My main idea is to test correctness, i.e. that the result after each pass has the same behavior as before the pass. I may try to come up with other properties, such as that the output is always in the correct target language.

I'll also try adding a test for the entire compilation relation, from start to finish, just make sure everything holds up.

After this is done, I would like to try to add a few other features. The language as it stands is very simple, and does not have function calls or loops. I would like to try to add these, as well as tests for them, if I can.

The tests for recursive functions and loops will be a little more difficult because of the possibility of infinite loops/recursion, so I cannot just run the programs to completion and then compare the results. There are a couple of ways to approach this; I may have to experiment with a few of them.

3 Deliverables

My deliverables will be my updated compiler with any features added, and the property based test suite I used to test its correctness.

4 Milestones

Milestones 1.1–1.5: property-based tests for each step in the compiler
Milestone 1.6: at least one test for entire compilation
Milestone 2: adding function calls
Milestone 3: adding tests for function calls
Milestone 4: additional features