Compilers

CMPT 432

Semester Project *part two*

This part of the project is meant to be done over the course of our entire semester together. But you can do parts of it in concert with phases of your compiler. This project is also meant to give you the opportunity to really dig in to AI-assisted software development. In the before-times, we could do a few graduate-level courses based on this material. But now, in our brave new world, I want you to practice being 10x productive. So, given the basis of your compiler from the first half of this project . . .

Lexer and Parser

Add advanced error detection, including error fixing and code rewriting. In other words, instead of just reporting an error, fix it too. But be sure that rewriting the code does not change its meaning.

Semantic Analysis

Add an optimization phase where you rewrite the AST to represent a more efficient version of the **same** program. Implement these common AST optimizations:

- constant folding
- constant propagation
- dead code elimination
- loop unrolling

Code Generation

In addition to targeting our subset of 6502 code, write another code generator that takes your AST and generates an LLVM Intermediate Representation. Then, once you have an LLVM IR, use LLVM tools to generate Java byte code. Finally, run your programs on the JVM. Compare their output to that of the same programs running your 6502 code on SvegOS.

Then write another code generation module that takes your AST and generates Java **source** code (not byte code, but actual, readable, Java source code.) Run the Java source code through the *javac* compiler and compare the resulting byte code to your LLVM-generated Java byte code. Then compare the output to that of the same programs running your 6502 code on SvegOS and your generated byte code.

Finally, write one more code generator module that takes your AST and generates TypeScript source code. Then compile the TypeScript code into JavaScript and run it with Node. Compare that output to all other outputs.

Slide Deck

Document all of these enhancements, tests, and their results in a presentation deck, as if you were going to give a talk about it at a conference. Cover what you did, how you did it, setbacks and victories you experienced along the way… and — most of all — **brag** about all this amazing stuff you did!

I can't wait to read it!