

Extra Credit Programming Assignment

Due Saturday, March 19, 2005 at 11:59pm

1 Introduction

This extra credit programming assignment is intended to be a fun project with which to end the semester. You will need to implement some optimizations in your compiler such that the generated code has the same semantic meaning, but executes in fewer cycles.

You may work in a group of one or two people. The submit program will ask you to specify group members when you turn in your assignment.

2 Optimizer Project

Extra credit will be awarded for projects that, in addition to code generation, perform some significant optimization of the code. The amount of extra credit depends on how well the optimization is written, documented, and demonstrated. Two critical factors are

1. correctness (the optimizations don't result in incorrect programs); and
2. the percentage speed-up your optimized code achieves over `coolc`, as measured by a weighted sum of the instructions executed on `spim` over a suite of benchmarks of our choosing.

To find out how many instructions a Cool program executes, run `spim` with the `-keepstats` option.

There are many possible optimizations to implement; see the ASU chapters 9 and 10 for ideas. Assuming your initial code generator is straightforward (like `coolc`'s), then two directions that may yield significant improvement are (1) improving register usage and (2) specializing the implementation of the basic classes `Int` and `String`.

We have not implemented an optimization phase in `coolc`, so we have no skeleton code to give you—you are on your own. If you want to do an optimization phase, you are encouraged to talk it over with one of the course staff first. *Under absolutely no circumstances should you try optimization before your code generator is finished!!*

There is a `-O` flag that controls the global variable `cgen_optimize` (C++) and `Flags.cgen_optimize` (Java). You may use this flag to switch between generating normal code and optimized code. For this project, we will always run your compiler with the `-O` flag on.

The total extra credit for doing optimization will not exceed 5% of the total grade for the course. Roughly speaking, the extra credit is worth up to about one-third of one of the two large programming assignments.

3 Grading

The final curve for the course will be determined *before* including the extra credit. In other words, if you elect not to do extra credit work, you will not be at a disadvantage in the final grading with respect to those who do.

This extra-credit option is open-ended; you can do as much as you like. We will award credit for results. For example, a project that merely attempts, but does not complete, an optimization phase may receive as little as no extra credit.

4 Submission

4.1 Optimizer Project

Submit your optimizer using “`submit PAX1`”, which will collect the same files as for the code generator assignment (PA5).

- Include a write-up of the optimizations you performed in the README.
- Make sure all your code for the optimizer and code generator is in
 - `cool-tree.h`, `cgen.h`, `cgen.cc`, `cgen_supp.cc`, and `emit.h` for the C++ version; or
 - `cool-tree.java`, `CgenClassTable.java`, `CgenNode.java`, `CgenSupport.java`, `BoolConst.java`, `IntSymbol.java`, `StringSymbol.java`, `TreeConstants.java`, and additional `.java` files you may have added for the Java version.
- Be sure to answer ‘yes’ to the submission prompt for files that contain your code.