University of Arizona, Department of Computer Science

CSc 553 — Assignment 4 — Due noon, Tue May 3 — 10%

Christian Collberg
January 28, 2011

1 Introduction

In this assignment you will extend the LUCA compiler (`lucac`) with an optimization pass. In particular, the assignment consists of five tasks:

1. Build control-flow graphs for each LUCA procedure. The basic blocks should contain lists of treecode nodes (`tree/*.java`).
2. Perform an *available expressions* dataflow analysis over the basic blocks.
3. Using the result of the dataflow analysis, perform a *global common subexpression elimination* optimization.
4. Perform a final *local optimization* pass to clean up the code: remove jumps-to-jumps, simplify arithmetic identities, etc.
5. Run your compiler on some benchmark programs — does optimization speed up the `lucax` interpreter at all? What if you run `lucax` with jitting turned on?

Note the following:

1. You don't have to worry about the object-oriented features of LUCA nor the garbage collector. I also won't test your jitter on optimized code.
2. Add an option `-O`*`level`* to `lucac` such that `-O0` does no optimization, `-O1` does local optimization, and `-O2` does global optimization.
3. You should work in teams of two students. Teams of three should also implement a *function inliner*.
4. In your documentation, make sure you describe exactly what kind of code you can and cannot handle. Are there any aliasing situations where your optimizer will fail to be conservative? Are there situations when it should be able to find a common subexpression, but doesn't?
5. In your documentation, describe exactly which local optimizations you have implemented.
6. I will be testing your code with the `lucax` interpreter you submit with `lucac`, so make sure that the interpreter can handle any optimized code you throw at it!

2 Submission and Assessment

The deadline for this assignment is noon, Tue May 3. It is worth 10% of your final grade.

You should submit the assignment to `d2l.arizona.edu`.

You should submit *one* file, ass4.zip, containing all the files necessary to build the compiler and interpreter. Modify the `makefile` so that the grader can build the project by simply typing `make`, and nothing else.

Don't show your code to anyone, don't read anyone else's code, don't discuss the details of your code with anyone. If you need help with the assignment see the instructor or the TA.