Project 3
CISC 471 Compiler Design
Investigation into Program Time and Space Effects of the gcc Optimizing Compiler

Deadline: Friday, 2 pm, May 23, 2008

You may work by yourself or in pairs.

Project Objectives: To learn about the possibilities and impacts of optimizing programs through the flags of an optimizing compiler.

Procedure:

1. Download 2 free, large C programs (over 10,000 lines of code each) and 2 very large C++ programs over 10,000 lines of code each. It might be easiest to download programs that have input test cases with them. Compile and run each program with gcc to make sure that you have 4 programs that will correctly compile and run on orioles.
2. For each program, record the program name, programming language, number of lines of code (use wc or sloccount as an estimate), web address where you got the code, and short 1-2 sentences on what the application does. Create a nicely formatted table that contains this information for the 4 programs, with a clear table caption and clear row and column headings.
3. Look at the gcc manual to determine all the various optimization flags that can be used when compiling a program.
4. For each of the 4 programs, compile the program several times, once for each optimization flag and save the executable for each compilation. You should have many executables clearly labeled.
5. Create a table that contains columns for: program name, size of the executable (1 column for each executable for that program) and one row for each program. The columns should be labeled by the optimization flag used to create that executable. Include a clear table caption.
6. For each executable, run the program with the same input data, and record the running time using the unix time command in /usr/bin/time. If you don't know how to use time, type "man time".
7. Create a table that contains columns for: program name, cpu running time of executable (1 column for each executable for that program) and one row for each program. The columns should be labeled by the optimization flag used to create that executable, and a clear table caption should be included.
8. For each of your tables, create a bar graph. The first bar graph should have the y-axis as the executable program size and the x-axis should have a set of bars for each program grouped together. The set of bars for a given program should have 1 bar for each optimization flag, and be the height of the executable's time (or space depending on which graph). The sets of bars should be marked with the program name while each bar should be marked to indicate the optimization flag. Clearly label the x and y axes.

9. Create your project report by following instructions below carefully.

Project Report Format:

1. On the first page: Project title and your name.
2. **Overview:** Introductory paragraph explaining the optimization flags of gcc in detail enough that a compiler writer would have a sense of what is performed by the compiler for each flag.
3. **Data:** Your tables and graphs. Include any other observations that you have about the executions that you made.
4. **Analysis of Results:** Clearly written description of your assessment of the results. What can you imply from the results on space? What can you imply from the results on time? Anything you learned about optimization of one language or program versus another? Which flags had the most affect over all the programs/languages?
5. **Conclusions:** Any general implications? Any challenges/issues in performing the investigation? Any disappointments in observations? Any surprises in observations?

Submission Procedure:

Please email your completed project report to pollock@cis.udel.edu in pdf format, or print it and put it under my office door at 436 Smith Hall by the deadline.

Deliverables:

1. Your project report with the 3 tables and 2 bar graphs.
2. Be sure that the TA/instructor can access your downloaded programs and executables (basically your work for this assignment!) in your svn for grading.

Grading Criteria:

The project will be graded according to the following criteria:

50 total points

2 points: following directions in getting 2 C and 2 C++ programs of appropriate size
4 points: table showing information about the downloaded programs' characteristics
5 points: table on space measurements
5 points: table on time measurements
5 points: graph for space
5 points: graph for time
5 points: overview section
8 points: analysis of results section
4 points: conclusions
2 points: nicely formatted, clear grammar and spelling of report
5 points: all files accessible on your svn