15-745 Advanced Optimizing Compilers: Task 3
Due 3/27/2006 5pm

March 2, 2006

1 Task Description

In this assignment, you will implement path profiling and trace scheduling.

2 Task Specifics

Please implement efficient path profiling, as described in the Ball/Larus paper. Your compiler should output instrumented code that will print trace information either to the screen or to a trace file (your choice). Your path profiler may ignore any input files which have more than 2^{32} paths (but you must check for this condition). Once you can generate trace files, implement a more intelligent trace scheduler than the one included in the compiler. Your new trace scheduler should attempt to minimize the number of jumps along hot execution paths, using the trace file information to determine which paths are hot.

You may work in groups of one or two (we suggest two). Please use a clean copy of the compiler (ie: clone the reference implementation) for this assignment. Once you have completed the task, please turn in (by email to `mderosa@cs`) the following files:

- the tarball of your modified source code
- a results.txt file (on the Task 0 benchmark suite) comparing the reference compiler against your instrumented code
- a sample trace file, from any of the test cases in the `big` directory (please specify which one)
- a results.txt file comparing the reference compiler against your trace-scheduled code

You will be graded both on the speedup (if any) on your rescheduled code and on the performance hit imposed by your profiling code.

NOTE: The timeframe for this assignment extends through spring break, but you are not expected to work on it during that period.