6.035 Derby Results Show
Outline

- Project Summary
- GCC Results
- Group Discussions
- Derby Results
The 6.035 Compiler Project

- Huge implementation requirement:
  - 10K, 13K, 18K, 25K lines of code

- You built a working optimizing compiler from scratch!
  - You understand design decisions in compiler design.

- However, Decaf was a simple language, imagine an optimizing C/Fortran compiler!
The 6.035 Compiler Project

• You made the decisions regarding what to implement.
• Based on experimentation with architecture.

Examples:
• No Instr Scheduling
• No Software Pipelining
The Derby Program

• A real image processing pipeline
  – Modeled on my normal post-processing workflow

Steps:
1. Levels correction: linear interpolation on the RGB histogram of the image to increase contrast and saturate colors.
2. Conversion from RGB->HSV colorspace
3. 3 passes of sharpening on 3 color components of the image (just like Photoshop’s Unsharp Mask)
4. Convert back HSV->RGB
The Derby Program
Characteristics of derby.dcf

- Loop-based program
  - Doubly-nested loops
  - Small loop bodies
- Lots of array accesses in loops
- Lots of loop invariant code
- Not much register pressure
- All outer loops parallelization
Observations Across Groups

• Optimizations:
  – Register Allocation: 4/4
  – Constant Prop: 4/4
  – Copy Prop: 4/4
  – DCE: 4/4
  – CSE: 3/4 (Phase 4)
  – Loop Inv. Code Motion: 3/4
  – 32-bit: 2/4
  – Strength Reduction: 1/4

Memory Hierarchy

Clean up your code from Phase 3

Opt. Tests were loop-based
Parallelization

• Only 2 groups implemented automatic loop parallelization.

• Big effort needed to get correct, even for Decaf
  – Multicores are here, right?
  – What does this tell us about automatically parallelizing C/Fortran?
  – What is needed for continued performance scaling?
GCC Experiments

**derby.dcf with GCC4**

- **O0**: No optimizations
- **O1**: RA, DCE, ConstProp, CopyProp,
- **O2**: Alignment, GCSE, InstrSched
- **O3**: Inlining, Vectorization

Higher Numbers Better!
Group Discussions

- le01
- le02
- le03
- le05
Derby Results
4\textsuperscript{th} Place: le05

derby.dcf Results

- le05
- le05 -opt all
- gcc -O0
- gcc -O1

Pipelines/Sec
3rd Place: le02

derby.dcf Results

- Pipelines/Sec

le02
le02 -opt all
gcc -O0
gcc -O1
Runner Up: le01

derby.dcf

- Pipelines/Sec

le01 -opt all

gcc -O0

gcc -O1
Derby Winners: le03

- **derby.dcf**

- Pipelines/Sec

- le03
- le03 -opt cse dce cp arrays constant parallel regs
- gcc -O0
- gcc -O1

- 6x!