15-745: Optimizing Compilers
Fall 2003
Syllabus

1 Course Details at a Glance

Lectures:	MW 3:00pm-4:20pm, F 10:30am-11:50am, WeH 4615A
Instructor:	Todd C. Mowry, WeH 8123, 268-3725, `tcm@cs.cmu.edu`
TA:	Dave Koes, WeH 3723, 268-1685, `dkoes@cs.cmu.edu`
Class Admin:	Jennifer Landefeld, WeH 8124, 268-4740, `jennsbl@cs.cmu.edu`
Web Page:	`http://www.cs.cmu.edu/afs/cs/academic/class/15745-f03/www/`
Newsgroup:	`cmu.cs.class.cs745`
Handouts:	*Electronic:* `/afs/cs.cmu.edu/academic/class/15745-f03/public`
	Hardcopies: In bins outside WeH 8124.

2 Textbook

Muchnick, Steven S., *Advanced Compiler Design and Implementation.* Morgan Kaufmann, 1997.

3 Course Overview and Objectives

Theoretical and practical aspects of building optimizing compilers that effectively exploit modern architectures. The course will begin with the fundamentals of compiler optimization, and will build upon these fundamentals to address issues in state-of-the-art commercial and research machines. Topics include: intermediate representations, basic blocks and flow graphs, data flow analysis, partial evaluation and redundancy elimination, loop optimizations, register allocation, instruction scheduling, interprocedural analysis, memory hierarchy optimizations, extracting parallelism, and dynamic optimizations. Students will implement significant optimizations within the framework of a modern research compiler.

4 Prerequisites

This course is not intended to be your first compilers course: it is geared toward students who have already had such a course as undergraduates. If you have not taken a compilers course already, it is still possible to take this course provided that you are willing to spend some additional time catching up on your own. It will also be helpful if you have some familiarity with the features of modern processor architectures (e.g., the memory hierarchy, pipelining, branch prediction, and instruction issue mechanisms). If you feel uncertain about whether you are adequately prepared to take this class, please discuss this with the instructor.

5 If You Are Not a CS or ECE PhD Student

If you are not a graduate student in either the CS or ECE PhD program, you need permission to take this class. If you have not already done so, send a message to the instructor stating your status, why you want to take the class, and if you want to take the class for credit or as an auditor.

6 Placing Out

Students who have already taken a *graduate-level* course in compiler optimization may find that some of this course material is familiar. It is likely that the focus and style of this course will be different from what you have experienced before, and that the pace will be fast enough that you will not be bored. However, if you feel strongly that you should be able to "place out" of all or part of this course, please discuss this with the instructor.

7 Course Work

Grades will be based on homeworks, a research project, an exam, and class participation.

Homeworks: There will be roughly three homework assignments. Each assignment involves a non-trivial amount of programming. Please work in groups of two on the assignments. (If you have difficulty finding a partner, please let us know, and we will help you find someone.) Turn in a single writeup per group.

Project: A major focus of this course is the project. We prefer that you work in groups of two on the project, although groups of up to three may be permitted depending on the scale of project (ask the instructor for permission before forming a group of three). The project is intended to be a scaled-down version of a real research project. The project must involve an experimental component—i.e. it is not simply a paper and pencil exercise. We encourage you to come up with your own topic for your project, although we will be posting suggested projects to the class web page at a future date. You will have six weeks to work on the project. You will present your findings in a written report (the collected reports may be published as a technical report at the end of the semester), and also during a poster session during the last day of class. Start thinking about potential project ideas soon!

Exams: There will be one exam covering the earlier (and more fundamental) portion of the course material. The exam will be closed book, closed notes.

Class Participation: In general, we would like everyone to do their part to make this an enjoyable interactive experience (one-way communication is no fun). Hence in addition to attending class, we would like you to actively participate by asking questions, joining in our discussions, etc. Three classes are set aside entirely for student-led in-class discussions on active areas of research and innovation in compiler optimization. All students are expected to lead one of these discussions.

7.1 Grading Policy

To pass this course, you are expected to demonstrate competence in the major topics covered in the course. Your overall grade is determined as follows:

Exam:	35%
Homework:	20%
Project:	35%
Class Participation:	10%

Late assignments will not be accepted without prior arrangement.

8 Schedule

Table 1 shows the tentative schedule. There might be some variations.

Table 1: 15-745, Fall 2003, Tentative Schedule.

Class	Date	Day	Topic	Reading	Assignments	Who
1	9/12	Fri	Overview of Optimizations	1.3-5, 2.1-9	#1 Out	TCM
2	9/15	Mon	Programming in SUIF	4.1-6, 4.9-10		TCM
3	9/17	Wed	Local Optimizations	7,7.1-4,18.1-3		TCM
4	9/19	Fri	Data Flow Analysis: Examples	8.1, 14.1.13		TCM
5	9/22	Mon	Data Flow Analysis: Theory	8.2-5	#1 Due, #2 Out	TCM
6	9/24	Wed	Global Common Subexpr. Elimination	13.1		TCM
7	9/26	Fri	Loop Invariant Code Motion	13.2		TCM
8	9/29	Mon	Induction Variables, Strength Reduction	14.1		TCM
9	10/1	Wed	Partial Redundancy Elimination	13.3		TCM
10	10/3	Fri	Interval Analysis	7.5-7, 8.8		TCM
11	10/6	Mon	Intro to Static Single Assignment (SSA)	8.10-11,12.6	#2 Due, #3 Out	TCM
12	10/8	Wed	SSA-Style Optimizations			TCM
13	10/10	Fri	*Recent Research on Optimization I*	*handouts*		N/A
14	10/13	Mon	*Recent Research on Optimization II*	*handouts*		N/A
15	10/15	Wed	*Recent Research on Optimization III*	*handouts*		N/A
	10/17	Fri	**No Class (Mid-Semester Break)**			
16	10/20	Mon	Register Allocation: Coloring	16.1-3	#3 Due	DK
17	10/22	Wed	Register Allocation: Spilling	16.3.10, 16.3.12		DK
18	10/24	Fri	Instr Scheduling: List Scheduling	17.1, 17.4.3, 17.5	Project Proposal	TCM
19	10/27	Mon	Instr Scheduling: Speculation	17.2.3		TCM
20	10/29	Wed	Instr Scheduling: Software Pipelining	20.3, 17.4		TCM
	10/31	Fri	**Exam**			
21	11/3	Mon	Memory Hierarchy Optimizations	20.1-6		TCM
22	11/5	Wed	Prefetching	20.4.4		TCM
23	11/7	Fri	Automatic Parallelization I	9.3-7		TCM
24	11/10	Mon	Automatic Parallelization II			TCM
25	11/12	Wed	Dynamic Code Optimizations			TCM
26	11/14	Fri	Case Studies, Embedded Systems			DK
	11/19	Wed			Project Milestone	TCM
	12/3	Wed			Project Due	TCM
	12/5	Fri	**Project Poster Session**			