15-745: Optimizing Compilers
Spring 2012
Syllabus

1 Course Details at a Glance

Lectures:	TWR 9:00am-10:20am, GHC 4303
Instructor:	Todd C. Mowry, GHC 9123, 268-3725, `tcm@cs.cmu.edu`
TAs:	Gennady Pekhimenko, `gpekhime@cs.cmu.edu`
	Gabe Weisz, `gweisz@cs.cmu.edu`
Class Admin:	Diana Hyde, GHC 7021, 268-1256, `dhyde@cmu.edu`
Web Page:	`http://www.cs.cmu.edu/afs/cs/academic/class/15745-s12/www/`
Handouts:	`/afs/cs.cmu.edu/academic/class/15745-s12/public`
Discussion:	`http://www.piazza.com/cmu/spring2012/15745`

2 Textbook

Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman, *Compilers: Principles, Techniques, and Tools (2nd Edition)*. Addison Wesley, 2006. ISBN: 978-0321486813. *(NOTE: It is important to use the 2nd Edition, rather than an earlier edition.)*

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 Graduate Student

If you are not a graduate student in either the CS or ECE 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 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!

Exam: 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.

6.1 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.

7 Schedule

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

Table 1: 15-745, Spring 2012. *(Revised 2/27/12.)*

Class	Date	Day	Topic	Reading	Assignments
1	1/17	Tue	Overview of Optimizations	9.1	
2	1/18	Wed	Local Optimizations	8.4-8.5	
3	1/19	Thu	The LLVM Compiler: Getting Started	`llvm.org/docs`	#1 Out
4	1/24	Tue	Data Flow Analysis: Examples	9.2	
5	1/25	Wed	Data Flow Analysis: Theory	9.3	
6	1/26	Thu	The LLVM Compiler: Further Details	`llvm.org/docs`	
7	1/31	Tue	Common Subexpressions, Constant Folding	9.2.6, 9.4	
8	2/1	Wed	Loop Invariant Code Motion	9.6	
9	2/2	Thu	Induction Variables, Strength Reduction	9.1.8	#1 Due, #2 Out
10	2/7	Tue	Partial Redundancy Elimination	9.5-9.5.2	
12	2/8	Wed	Lazy Code Motion	9.5.3-9.5.6	
12	2/9	Thu	Region-Based Analysis	9.7	
13	2/14	Tue	Intro to Static Single Assignment (SSA)	6.2.4	
14	2/15	Wed	SSA-Style Optimizations		
15	2/16	Thu	Register Allocation: Coloring	8.8	#2 Due, #3 Out
16	2/21	Tue	Register Allocation: Spilling		
17	2/22	Wed	Intro to Instruction Scheduling	10.1-10.2	
18	2/23	Thu	List Scheduling, Global Scheduling	10.3-10.4	
19	2/28	Tue	Software Pipelining	10.5	
20	2/29	Wed	Pointer Analysis	12.4, 12.6-12.7	
21	3/1	Thu	Dynamic Code Optimization		
22	3/6	Tue	*Recent Research on Optimization I*	*handouts*	
23	3/7	Wed	*Recent Research on Optimization II*	*handouts*	
24	3/8	Thu	*Recent Research on Optimization III*	*handouts*	#3 Due
			Spring Break		
	3/20	Tue	*Meetings to discuss Project Proposal ideas.*		
25	3/21	Wed	Memory Hierarchy Optimizations	12.1.4-12.1.5, 12.2	
26	3/22	Thu	Locality Analysis	12.3-12.5	Project Proposal
27	3/27	Tue	Prefetching	12.12.4	
28	3/28	Wed	Array Dependence Analysis	12.1.1-3, 12.6-7	
29	3/29	Thu	Thread-Level Speculation		
	4/5	Thu	**Exam**		
	4/19	Thu			Project Milestone
	5/1	Wed			Project Due
	5/3	Thu	**Project Poster Session**		