EE663 Compiler Code Generation, Optimization, and Parallelization

Tentative Syllabus

Spring 2002 Mo/We/Fr 10:30-11:25 AM, MSEE 140

Instructor	Prof. R. Eigenmann
Tel	49-41741
Email	eigenman@ecn
Office	EE334C
Office Hours	Tu/Th 8:00-10:00
Secretary:	Connie Boss, EE339
Course Web Page:	http://www.ece.purdue.edu/~eigenman/EE663/

Prerequisites: EE573 or EE468 or equivalent. Substantial programming experience.

Text: (Optional) Fischer and LeBlanc, Crafting a Compiler with C, Benjamin/Cummings, 1991, ISBN 0-8053-2166-7
Course notes and research papers will be used.
Background texts:
Michale Wolfe, High Performance Compilers for Parallel Computing, Addison-Wesley, ISBN 0-8053-2730-4.
Utpal Banerjee, Dependence Analysis, Kluwer, ISBN 0-7923-9809-2.
Ken Kennedy and John R. Allen, Optimizing Compilers for Modern Architectures: A Dependence-based Approach, Morgan Kaufmann Publishers, ISBN 1558602860.

Description: This course presents the concepts needed to design and implement advanced optimizing preprocessors and code generators. Specific emphasis will be put on techniques for exploiting parallelism in multiprocessors and in microarchitectures. Each student will conduct a compiler implementation project. Each student will also present a research paper from a list of selected articles.

Objective: Students who successfully complete this course will have demonstrated that they can

- explain the various passes of an optimizing compiler; including program analysis, dependence analysis, enabling transformations, loop restructuring, and instruction level

parallelization,

- describe implementation methods and performance characteristics of these passes and tasks.
- explain program analysis techniques used to decide on correctness and profitability of the various program transformations.
- explain research issues related to these techniques, known solutions, differences between alternative solutions, and open issues.
- implement an optimizing compiler pass in the context of a realistic compiler.

Course Grading:

Tests	50%.	(20% midterm, 30% final exam)
Participation in class	10%.	
Class Presentation	10%.	
Projects	30%	

Regrading policy: any information that you feel will affect your grade, including grade disputes and emergencies that prevent you from attending class, must be sent by email to the instructor within a week from the event. In general, the instructor will not respond to these notes or change your intermediate grades. However, the information will be factored into your final class grade.

Course schedule:

```
45 lectures (eff. 15 weeks) plus final exam
                                      week
Monday  Jan  7 Sem starts    M T R    1
            14               M T -    2
            21               . T R    3       Jan 21: M.L. King Day
            28               M T R    4
        Feb  4               M T R    5
            11               M T R    6
            18               M T R    7
            25               M T R    8
        Mar  4               M T R    9       March 6: Midterm exam
            11              Spring Break
            18               M T -    10
            25               M T R    11
        Apr  1               M T R    12
             8               M T R    13
            15               - T R    14
            22               M T R    15
        Apr 29             final exams week     Final exam date: TBD

( no classes at dates marked ‘‘-’’. Time is made up by 55-minute lectures)
```

Tentative Course Outline:

	Topic	#weeks
1.	Introduction and Motivation	1
2.	Effectiveness of parallelizing compilers	1
3.	Basic Transformations	1
4.	Program Analysis I	1
5.	Advanced Loop Optimizations	1
7.	Program Analysis II	2
8.	Performance of Compiler Techniques	1
9.	Instruction Level Parallelization	2
10.	Class Presentations	3

Class Projects: Each student will implement an optimizing compiler pass within either the Gnu C 3.0 compiler infrastructure or a new research infrastructure. Project details will be presented in the first week of class. Much of the implementation work is done individually. However, the design and the final overall evaluation will be done collaboratively with the other groups.

Class Presentations: Each student will present a paper from the selected list of papers on the class web page. The presentation will be 35 minutes, followed by questions. Timing of the class presentation will factor into the grade.