University of Michigan EECS 586, Winter 2005, Algorithms

January 4, 2006

1 Infomation

Title:	EECS 586, Design and Analysis of Algorithms
Lecture Time:	TTh, 1:30-3
Lecture Place:	1001 EECS
Discussion Time:	F, 1:30-2:30
Discussion Place:	1003 EECS
Prerequisites:	EECS 281 or permission
Texts:	1. Introduction to Algorithms, 2e Cormen, Leiserson, Rivest, and Stein `http://mitpress.mit.edu/algorithms/` (Follow links for errata) 2. Computers and Intractability Garey and Johnson
Course Webpage:	`http://www.eecs.umich.edu/~martinjs/eecs586/`
Instructor:	Martin Strauss
Office, central campus:	3063 East Hall
Office, north campus:	3611 CSE
Email:	`martinjs@eecs.umich.edu`
Office hours:	After class and by appointment
GSI:	Xiaolin Shi
Email:	`shixl@eecs.umich.edu`

2 Goals

We will learn techniques for analysis of algorithms. This includes language for discussing algorithms separately from their implementation and particular inputs. Is the algorithm correct on all or most inputs? Is it approximately correct? Is the algorithm efficient in terms of run time, memory space, or other resource? Is it efficient on all or most inputs? If the algorithm behaves randomly (often a good thing!), is the algorithm correct or efficient for most of its randomly chosen runs? Will the algorithm be run once, or run many times? In the latter case, do we know in advance the sequence of inputs? After developing the appropriate vocabulary, we study tools for answering these questions with the rigor of mathematical proof. The answer may be negative—there is no algorithm that efficiently solves our problem.

We also design *simple* algorithms and perform *simple* modifications to existing algorithms. Typically the analysis is difficult enough even if the algorithms are simple!

As a secondary goal, to illustrate our techniques, we will study particular algorithms that are fundamental and/or of particular interest to the students. These include sorting and graph algorithms.

3 Required Work

The grade will be based on performance on homework assignments. Homework will be done either by individuals or in small groups—instructions will be provided for each assignment. You may read the literature and consult with the professor, GSI, classmates, or others about general questions on the material, but you should not receive specific help on the assignments. (Naturally, for group homework, each group is encouraged to work together interactively.) If you have a question about material and are unsure as to whether the answer would be general or specific to an assigned problem, ask the instructor. If you do use an outside source (other than your homework groupmates, when appropriate), give credit in writing on your homework.

When group work is allowed, each student participating in group must have full understanding of all the work submitted by the group. All homework is subject to oral review with the instructor.

Each homework problem will be graded on the following scale:

5	Outstanding in some way
4	Basically correct
3	More right than wrong
2	More wrong than right, but some good stuff
1	Very little right; *or* no attempt
0	Nothing right; lots of irrelevant detail

Note that handing in a blank answer earns one point. In general, one point may be deducted from any score for adding irrelevant details.

Historically, most grades have been B+ or A- and most of the remaining grades have been B or A.

Please advise me as soon as possible about any unusual bureaucratic circumstances *e.g.*, expected long-term absences.

4 Course Outline

- Basics—non-Randomized
  - CLRS 1-4, Appendices A. (Preliminaries, asymptotic growth, recursive algorithms.)
  - Additional notes (generating functions, sums of binomial symbols)
  - CLRS 6 (Heapsort algorithm, data structures, loop invariants)
- Basics—Probability
  - CLRS 5, Appendix C.1, C.2, C.3, C.5
  - Additional notes (Markov, Chebychev, and Chernoff bounds)
  - CLRS 7, 9 (More sorting and selection algorithms, randomized algorithms, average case analysis)
  - CLRS 8.1 (lower bound)
  - CLRS 11.1, 11.2, 11.3, 11.5 (Hashing)
  - Additional notes (random variables of limited independence)
- Meta-algorithmic techniques
  - CLRS 15.2, 15.3 (Matrix-chain multiplication algorithms, dynamic programming)
  - CLRS 16.2, 16.3 (Huffman code construction algorithm, Greedy Algorithms)
  - CLRS 17.1–17.3 (Amortized analysis)
- Specific Algorithms
  - CLRS 12 (Binary Search Trees)
  - CLRS 22–26, as time permits (Graph Algorithms)
- NP-Completeness
  - GJ and/or CLRS 34
- Approximation Algorithms
  - CLRS 35