CMSC 451: Design and Analysis of Algorithms

Fall 2001

http://www.cs.umd.edu/~samir/451/

Instructor: Samir Khuller - Office: AVW 3217. Office phone: 405–6765.
E-mail: samir@cs.umd.edu.
Office hours: Monday 3:00 - 4:00 and Thursday: 11:00am - 12:00am.

Class Time: TuTh 9:30 - 10:45, Room: CSS 1113

Teaching Assistant: Eiman Elnahrawy. Office hours will be held in 1151 AVW.
Email: eiman@cs.umd.edu.
Office Hours: Will be posted on the class page and by appointment.
The BEST way to contact the TA is to send her email.

Course Overview: This course presents the fundamental techniques for designing efficient computer algorithms, proving their correctness, and analyzing their complexity. General topics include graph algorithms, and basic algorithm design paradigms (such as divide-and-conquer, dynamic programming and greedy algorithms), lower bounds and NP-completeness.

Text: Thomas Cormen, Charles Leiserson, Ron Rivest, *Introduction to Algorithms*, McGraw Hill and MIT Press, 1990. There is a second edition out now (2001). You may use either edition this semester. The reason we are not being particular is that a lot of students have copies of the first edition if they took CMSC 251 here. Now only the second edition is available. If something particular is needed from either section, you can make a copy of the relevant pages after borrowing the book from the library (or from me, I will have both editions). However, the newer edition has some useful new material. So if you are interested in this area, it will be wise to invest the money to get the new edition.

Prerequisites: CMSC 112, CMSC 150/250, and CMSC 251. Each student is expected to know the basic concepts of programming (e.g. loops, pointers, recursion), discrete mathematics (proof by induction, sets), simple data structures (lists, stacks, queues, trees, heaps), and calculus (logarithms, differentiation, integration). I will assume knowledge of algorithm analyses techniques (material normally covered in CMSC 251).

Course Work: Course work will consist of 5-6 homework assignments, a small project and two exams (one midterm and a comprehensive final). Homework problems will be mathematically oriented.

The midterm will be in class in October (date will be posted on the web page). Homeworks are to be turned in at the start of class on the due date. Since homework solutions will be handed out on the day the homework is due NO LATE HOMEWORKS WILL BE ACCEPTED. (In other words, hand in whatever you have finished. You are also welcome to turn in homeworks *before* the due date if you cannot come to class on the due date.) If you cannot come to class for some reason, please mail the homework to me (should be postmarked a day **before** the due date).

All homeworks are to be done independently, with no help from the web, or other sources. If you have questions, please talk to the TA or the Instructor. Assignments are to be written up NEATLY. Badly written assignments WILL NOT be graded. Please staple your homework (every semester students lose parts of homeworks due to them not being stapled). **It is your responsibility to make sure that you pick up all homeworks and handouts. All course information and handouts will be available on the web page.**

Grading: Final grades will be based on homework assignments, the project, the midterm exam, and the comprehensive final exam. The relative weights of these will be 15% for the homework total, 10% for the project, 35% for the midterm, and 40% for the final exam. The Instructor will make the use of the new grading system that allows for + and − grades. Graduate students in this class will be given extra work on homeworks and exams.

Syllabus: The topics and order listed below are tentative and subject to change.

1. Graph exploration: connected components, topological sorting, strongly connected components (6 lectures).
2. Greedy algorithms: minimal spanning trees, shortest paths, scheduling (4 lectures).
3. Divide and Conquer algorithms: geometric algorithms, selection, lower bounds for minimum and sorting, Strassen's matrix multiplication (4 lectures).
4. Dynamic programming: shortest paths, Warshall's algorithm, optimal search trees (5 lectures).
5. String matching, other string type algorithms (2 lectures).
6. NP-completeness: introduction to reductions, the classes P and NP, NP-complete problems, approximation algorithms (6 lectures).