C.S. 222, Fall 2007, Computer Algorithms Course Outline

I assume that everyone has had an undergraduate course in this area, or equivalent background. In some ways, this course examines the same issues as the undergraduate course (cs 122A here), but chooses more advanced and complex algorithms, examines some issues more deeply, and requires more difficult and interesting homework.

The purpose of the course is a) to raise your level of sophistication in thinking about the design and analysis of algorithms; b) learn some of the classic results and recent improvements; c) exercise your creativity in designing efficient algorithms.

Course website www.cs.ucdavis.edu~gusfield/cs222f07

Lecture Topics F 2007

Numbers in parentheses give section numbers in the text, but other materials will also be used, mostly posted on the class website.

1. Introduction (Types of analysis) Divide and Conquer algorithms and analysis by recurrence relation: mergesort, counting crossings, closest point problem, Methods for solving recurrence relations, Master method. Read the web postings on complexity game plan, and on the Master method.

2. Dynamic Programming: Points to lines (6.3), RNA folding (6.5)

3. Sequence alignment (6.6) linear space (6.7), shortest paths intro

4. Shortest paths (6.8-6.9, 6.10)

5. Network Flows (7.1, 7.2)

5. Network Flows (7.3, 7.5,7.6)

6. 10/17 Network Flow applications, 7.5 (max-matching = min-cover), Baseball (7.12), advanced graph algorithms (summary 7.13)

7. Hard Problems: P, NP, reductions, NP-hard problems (8.1,2)

8. Hard Problems, subset sum reduction (8.8), Pspace (9.1,9.2)

9. Dealing with hard problems, special cases: 10.1 Vertex Cover, 10.2 Independent Set, Approximations, Scheduling 11.1, Multiple Alignment

10. Approximations, k-center 11.2, vertex cover 11.4, Disjoint paths 11.5

14. VC approx (11.6), Branch-bound, Local Search (12.1)

15. Randomized algoriths: contention resolution (13.1)

16. Randomized Min-cut (13.2) Randomized Max-SAT (13.4), Amortized analysis: deterministic global Min-cut

17. Linear-time pattern matching, constant-time least common ancestor algorithm, suffix arrays.

18. Hashing (13.6)

19. Other topics as time allows.

Course Instructor: Dan Gusfield gusfield@cs.ucdavis.edu

Office: 2125 EUII

Office Hours: To be posted next week.

T.A': Till Stegers - office hours to be posted next week

Discussion sections begin next week.

Grades: There will be written homeworks that might only be partially graded due to limitations in available graders. There be one midterm and a final exam.

Also, note the date of the final now - no early or late exams can be given.

Generally, the homeworks will count for 25% of the grade; the midterm will count for 25% and the final for 50%. However, these might be slightly altered depending on our ability to grade homeworks.

CLASSROOM PARTICIPATION WILL ALSO BE TAKEN INTO CONSIDERATION IN DETERMINING GRADES.

The text is by J. Kleinberg and E. Tardos, Algorithm Design, Addison-Wesley 2006, but will be supplemented by other postings and links. The classic text by Coreman, Leiserson and Rivest has often been used in the class and is also a good book to consult.

About Homeworks

0) Homeworks are intended to be done individually, unless noted otherwise. Doing homeworks is the most important part of learning in this course. It is assumed you can master the material of the class; homeworks push you to actively discover, think, prove and write. Those are the most important things to learn in this class. NEVER search for homework solutions on the internet. These are generally easy to spot because they don't fully match the problem as stated or use techniques we haven't discussed.

1) Homeworks must be written clearly and cleanly. Typed homework is much appreciated if possible. Or type the English part and write by hand, very clearly, any mathematical parts. View this as an opportunity to learn to use Latex to handle the mathematical symbols. Homeworks that are grossly difficult to read (for example, light pencil or really bad handwritting - I have bad handwritting, so I type) or just contain sentence fragments or fragments of ideas will mostly not be graded. Try NOT to write code (generally it will not be read), and try to limit the amount of pseudo-code you write. Explain ideas and algorithms and write proofs, not code.

2) Each homework has a due date, and the homeworks are designed so that that due date is realistic. Homeworks should be put in the homework box (room 2131) on the day they are due. However, there is an automatic extension for everyone on every homework. A homework due on a class day (MWF) will be accepted the next class day (in the homework box, room 2131) before 4:30pm. NO homeworks will be accepted after the extension period. It is not a good idea to consider the extension period as the de facto due date, since if you get into the habit of turning in each homework on the extension date, you will then not have any time cushion when you really need it.

3) **Please read your homeworks soon after getting them. I sometimes make mistakes and the sooner you ask me about anything questionable, the better for all the students (and for me).** Thanks.

4) Most important notices (including this one) will be posted to the class website. Sometimes someone will send me email asking about something and I will post the question and answer on the website so everyone can see it. So if you find something odd in the homework, for example, please look on the website first for a possible answer or update.