CS 561, Midterm Review

Jared Saia
University of New Mexico

Midterm

- 5 questions
- There will be some time pressure, so make sure you can solve problems both quickly and correctly.
- I expect a class mean between 50 and 60 points

Topics Covered

- Probability and Randomized Algorithms: Linearity of Expectation, Union Bounds, Markov's inequality. Randomized Quicksort, Bucket Sort, Skip Lists, Count-Min sketch
- Recurrence Relations and Induction: Definitions of big-O and friends, recursion trees, Master method, annihilators and change of variables; Proof by induction! (Chapters 3 and 4)
- Dynamic Programming: String Alignment, Matrix Multiplication, Longest Common Subsequence (Chapter 15)
- Greedy Algorithms: Activity selection, fractional knapsack, MST, proof via exchange property (Chapter 16)
- Amortized Analysis: Aggregate Method, Accounting Method, Potential Method (Chapter 17)

Problem: Short Answer

Collection of true/false questions, matching and short answer questions. Some examples:

- Short Answer, Θ notation. May cover any of the topics we've worked on in class.
- Know the resource bounds for all algorithms covered. Know when you might use them.

Problem: Induction/Recursion

Possibilities:

- Proof by Induction
- Remember: Solve big problems by piecing together solutions to smaller sub-problems.
- Recursion/Recurrences

Problem: Dynamic Programming/Greedy

- Key focus will be on getting the correct recurrence
- Probably related to some problem we did in class and/or homework
- Practice solving a big problem by using solutions to sub-problems
- Greedy: Show greedy algorithm for the problem fails and/or give correct greedy algorithm for a variant of the problem

Problem: Probability

- Use Linearity of Expectation, Union Bounds, Markov's inequality to solve problems
- Remember: LOE and Union Bounds work even without independence or random variables/events.

Problem: A Harder Problem

- Uses tools from class
- May need to apply them in a new/clever way
- Requires lots of thinking, little writing.

How to Study?

A: Solve Problems! Start with worked examples from lecture

1. Cover up the answer
2. Try to re-derive
3. If you get stuck, uncover a couple lines of the worked example
4. Repeat

More Problems

Hungry for more problems? Good!

1. Redo HW problems!
2. Do worked examples from our textbook
3. Continue with Jeff Erickson's book *Algorithms* (free online)
4. Website leetcode.com is a great resource (click on the tag "dynamic programming" or "greedy algorithm" for job interview type questions in that area).
5. Do problems from my past midterms.