CS 362, Review

Jared Saia
University of New Mexico

Final Info

- You can bring 2 pages of “cheat sheets” to use during the exam. You can also bring a calculator. Otherwise the exam is closed book and closed note.
- Note that the web page contains links to prior classes and their tests. *Many of my questions will be similar in flavor to these past tests!*

Final

- 5 to 6 questions
- There will be some time pressure, so make sure you can solve problems both quickly and correctly.
- I expect a class mean of between 60 :(and 70 :) points

Topics Covered

- Asymptotic Analysis and Recurrence Relations (Chapter 3 and 4 in text) : defns of big-O and friends, recursion trees, master method, annihilators and change of variables
- Dynamic Programming: general concepts, String Alignment, Matrix Multiplication, Longest Common Subsequence (Chapter 15)
- Greedy Algorithms: general concepts, activity selection, fractional knapsack, MST (Chapter 16)
- Amortized Analysis: Aggregate Method, Accounting Method, Potential Method, Dynamic Array (Chapter 17)
- Disjoint-Sets: Disjoint Set Operations, Representation as Forest, Union by Rank and Path Compression, Amortized Costs (Chapter 21)
- Minimum Spanning Trees: Definition, Kruskal's Algorithm, Prim's Algorithm, Safe Edge Theorem and Corollary

- Graph Algorithms: Graph Representations, BFS, DFS, Single-Source Shortest Path, All-Pairs Shortest Paths; Dijkstra's, Bellman-Ford, Floyd-Warshall (Chapters 22 23,24,25)
- NP-Hard Problems: Definitions of P, NP, co-NP, NP-Hard, and NP-Complete; General concepts; Reductions (i.e. how to show that a problem is NP-Hard); Classic NP-Hard problems: Circuit Satisfiability, SAT, 3-SAT, Coloring, Clique, Vertex Cover, Independent Set, Hamiltonian Cycle, TSP. (Chapter 34)
- Approximation Algorithms: Vertex Cover, TSP, etc.

In general should know the resource bounds for all algorithms covered.

Example Problem - Short Answer

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

- T/F questions covering all topics
- Multiple Choice e.g. I give you some "real world" problems and ask you which algorithm we've studied in class that you would use to solve each of them; I give you some problems and ask you how fast they can be solved, etc.

Example Problem - Review

- Possibility 1: Asymptotic Analysis / Recurrence Relations
- Possibility 2: Dynamic Programming (new: *Floyd-Warshall*, Dijkstra's, and Bellman-Ford)
- Possibility 3: Greedy Algorithms (new: Kruskal's and Prim's)
- Possibility 4: Amortized Analysis

Example Problem - Graph Theory

- Possibility 1: BFS or DFS algorithms (and how to use them)
- Possibility 2: Single Source Shortest Paths (Dijkstra's and Bellman-Ford)
- Possibility 3: All Pairs Shortest Paths (Floyd-Warshall)

Example Problem - NP-Hardness

- Possibility 1: Something like Challenge problem 1 from last lecture
- Possibility 2: I give you a problem and ask you to prove it's NP-Hard by a reduction from another NP-Hard Problem.
- Possibility 3: I give you an NP-Hard Problem and ask you give an approximation algorithm for it (the problem would be a variant of something already seen in class)