CSE 421: Midterm Review

Larry Ruzzo

Complexity, I

Asymptotic Analysis

Best/average/**worst** cases

Upper/Lower Bounds

Big-O, Theta, Omega, little-o

- definitions; intuition

Analysis methods

- loops
- recurrence relations
- common data structures, subroutines
- & specialized arguments, e.g. “look at every edge twice”

Graph Algorithms

Graphs

- Representation (edge list/adjacency matrix)
- Breadth/depth first search
- Connected components
- Shortest paths/bipartitness/2-Colorability
- DAGS and topological ordering
- DFS/articulation points/biconnected components

Design Paradigms

Greedy

emphasis on correctness arguments, e.g. stay ahead, structural characterizations, exchange arguments

Divide & Conquer

recursive solution, superlinear work, balanced subproblems, recurrence relations, solutions, Master Theorem

Later:

Dynamic Programming, ⋯

Examples

Greedy

- Interval Scheduling Problems (3)
- Huffman Codes
- Examples where greedy fails (stamps/change, scheduling, knap, RNA,…)

Examples

Divide & Conquer

- Merge sort
- Counting Inversions
- Closest pair of points
- Integer multiplication (Karatsuba)
- Matrix multiplication (Strassen)
- Powering

Midterm Friday, 4/28/2017

Closed book, 1 page of notes

(8.5x 11, 2 sides, handwritten)

(no bluebook needed; scratch paper may be handy; calculators unnecessary)

All up through “Divide & Conquer”

assigned reading up through Ch 5;

slides

homework & solutions

Some Typical Exam Questions

Give O( ) bound on 17n*(n-3+logn)

Give O( ) bound on some code `{for i=1 to n {for j …}}`

True/False: If X is $O(n^2)$, then it's rarely more than n^3 +14 steps.

Explain why a given greedy alg is/isn't correct

Give a run time recurrence for a recursive alg, or

Solve a simple recurrence

For any of the algs we've studied

- Simulate it on given input
- Explain its runtime analysis or analyze a similar alg
- Explain/give counterexample for failure of a modified version

Give an alg for a new problem/analyze it/argue correctness