CSE421: Review
Larry Ruzzo
Summer 2007

© W.L.Ruzzo & UW CSE 1997-2007

Complexity, I

Asymptotic Analysis
Best/average/worst cases
Upper/Lower Bounds
Big O, Theta, Omega
Analysis methods
• loops
• recurrence relations
• common data structures, subroutines
• “progress” arguments and general brute cleverness…

Graph Algorithms

Graphs
- Representation (edge list/adjacency matrix)
- Breadth/depth first search
- Bipartitness/2-Colorability
- DAGs and topological ordering
- Articulation points/Biconnected components

Design Paradigms

Greedy
Dynamic Programming
- recursive solution, redundant subproblems, few
do all in careful order and tabulate
  (usually far superior to “memoization”)
Divide & Conquer
- recursive solution
- superlinear work
- balanced subproblems
- recurrence relations, solutions, Master Theorem
Examples

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

Divide & Conquer
- Merge sort
- Closest pair of points
- Integer multiplication (Karatsuba)
- Matrix Multiplication (Strassen)

Examples

Dynamic programming
- Fibonacci
- Making change/Stamps, Knapsack
- Weighted Interval Scheduling
- RNA
- String Alignment

Flow and matching
- Residual graph, augmenting paths, max-flow/min-cut, Ford-Fulkerson and Edmonds-Karp algorithms, integrality, reducing bipartite matching to flow

Complexity, II

P vs NP
- Big-O and poly vs exponential growth
- Definition of NP - hints and verifiers; nondeterminism
- Example problems from slides, reading & hw
  - SAT, 3-SAT, circuit SAT, vertex cover, quadratic Diophantine equations, clique, independent set, TSP, Hamilton cycle, coloring, max cut, knapsack
- \( P \subseteq \text{NP} \subseteq \text{Exp} \) (and worse)
- Definition(s) of (polynomial time) reduction
- \( \text{SAT} \leq_p \text{VertexCover} \) example (how, why correct, why \( \leq_p \), implications)
- Definition of NP-completeness
- NP-completeness proofs
- 2x, 1.5x approximations to Euclidean TSP

Some Typical Exam Questions

Give \( O() \) bound on \( 17n^n(n-3+\log n) \)
Give \( O() \) bound on some code \( \{ \text{for } i=1 \text{ to } n \{ \text{for } j \ldots \} \} \)
True/False: If \( X \) is \( O(n^2) \), then it’s rarely more than \( n^3 + 14 \) steps.
Give a run time recurrence for a recursive alg, or solve a simple one
Simulate any of the alg we’ve studied
Give an alg for problem \( X \), maybe a variant of one we’ve studied, or prove it’s in \( \text{NP} \)
Understand parts of correctness proof for an algorithm or reduction
Implications of NP-completeness
Reductions
NP-completeness proofs