CSE 421: Review

Larry Ruzzo

Winter 2012

Complexity, I

Asymptotic Analysis

Best/average/**worst** cases

Upper/Lower Bounds

Big O, Theta, Omega

Analysis methods

  loops

    recurrence relations

    common data structures, subroutines

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. <u>exchange</u>

Divide & Conquer

recursive solution, superlinear work, balanced sub-problems, recurrence relations, solutions, Master Thm

Dynamic Programming

recursive solution, redundant subproblems, few

do all in careful order and tabulate; <u>OPT function</u>

(usually far superior to "memoization")

Powerful Subproblems

Flow, Matching, Linear Programming

Examples

Greedy

    Interval Scheduling Problems (3)

    Huffman Codes

    Examples where greedy fails (stamps/change, scheduling, knap, RNA,…)

Examples

Divide & Conquer

> Merge sort
>
> Closest pair of points
>
> Integer multiplication (Karatsuba)
>
> Powering

Examples

Dynamic programming

    Fibonacci

    Making change/Stamps, Knapsack

    Weighted Interval Scheduling

    RNA

    String Alignment

OPT function

Examples & Concepts

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/certificates and verifiers

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 $\subseteq$ NP $\subseteq$ Exp (and worse)

Definition(s) of (polynomial time) reduction

SAT $\leq_p$ IndpSet, Knap examples (how, why correct, why $\leq_p$, implications)

Definition of NP-completeness

NP-completeness proofs

2x, 1.5x approximations to Euclidean TSP

And see how relevant it is to your daily life!

Classic Nintendo Games are (NP-)Hard

Greg Aloupis[*]   Erik D. Demaine[†]   Alan Guo[††‡]

March 9, 2012

Abstract

We prove NP-hardness results for five of Nintendo's largest video game franchises: Mario, Donkey Kong, Legend of Zelda, Metroid, and Pokémon. Our results apply to Super Mario Bros. 1, 3, Lost Levels, and Super Mario World; Donkey Kong Country 1–3; all Legend of Zelda games except Zelda II: The Adventure of Link; all Metroid games; and all Pokémon role-playing games. For Mario and Donkey Kong, we show NP-completeness. In addition, we observe that several games in the Zelda series are PSPACE-complete.

Final Exam Mechanics

Closed book, 1 pg notes (8.5x11, 2 sides, handwritten)

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

Comprehensive, w/ post-midterm bias

assigned reading

slides

homework & solutions

Some Typical Exam Questions

Give O( ) bound on 17n*(n-3+logn), or on code {for i=1 …}}

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 one

Simulate any of the algs we've studied

Give an alg for problem X, maybe a variant of one we've studied, or prove it's in NP

Understand parts of correctness proof for an algorithm or reduction

Implications of NP-completeness

Reductions

NP-completeness proofs