23 Feb. 2018 PRELIM 1 REVIEW

Announcements.

[1] See Piazza for review sheet and dyn prog practice problems.

[2] Some office hrs next week are shifting to Mon. Watch Piazza for more news on this.

Prelim time, date, location: Tues 2/27 7:30–9:00pm Statler Aud. lower level.

Make-up prelim 3/1 6pm. (write to me and Xiaotong, xg259)
Closed book, closed notes.

Topics.

1. Stable Matching
2. Greedy Algorithms
3. Minimum Spanning Trees
4. Dynamic Programming

Specific Algorithms: Gale Shapley "Proposal" Algorithm, Kruskal, Prim, Bellman-Ford

General Problem Solving Principles. (Solving algo problems resembling the homework. Find example where an incorrect algo fails.)

Properties / Mathematical Structures having to do with stable matchings, minimum spanning trees.
E.g. cut property, cycle property.

How to identify the correct method for solving a problem, X?

(1) Is X a "disguised version" of some problem, Y, whose solution you already know?
If so, solve by reducing to Y.

E.g. Problems about matching objects of 2 different types might reduce to stable matching.

E.g. Problems about connecting a bunch of things together or selecting a subgraph of graph might reduce to MST.

E.g. An unbelievably wide variety of problems might reduce to shortest path.

(2) What's the natural greedy algorithm?
Can you find a counterexample?
If not, try proving it works.

(3) If the natural greedy idea fails, try dyn prog.
Or if part a of the question by filling in a table.

3a. What would be the last choice you need to make when assembling a solution to the problem?
(E.g. include item #n in the knapsack or leave it out.)

3b. What does my alg need to know at the time of making that choice?
(E.g. what is the best knapsack solution I can assemble from items $1,\ldots,n-1$)

3c. How can I format a table so that the relevant info is already stored before I have to make my last choice?

3d. How to initialize table? In what order to fill in its entries? What formula computes a table entry using previously computed values?

Proof of correctness: Induction over table entries in the order the alg. fills them in.

IND HYP Defines what value each entry of the table is intended to store.

Most common mistake: not actually defining what you mean.

E.g. "My induction hypothesis is that A[i] is the correct value for i."

BASE CASE Initializations are correct.

INDUCTION STEP Justify recurrence relation.

Correctness Proofs for Greedy Algorithms.

(1) "Greedy Stays Ahead".

IND HYP Assert some type of optimality property of the partial solution constructed after k iterations.

E.g. "After k iterations, the edge set selected by Kruskal's Alg is a minimum cost acyclic subgraph of the graph formed by the k least-cost edges."

BASE CASE Usually trivial.

"The only subset of empty set is empty set."

INDUCTION STEP Justify the greedy rule.

Less formulaic.

(2) "Exchange Argument"

2a. Define a "local improvement" operation.

2b. Show that when local improvement is applied to a solution other than the greedy one, either it improves that solution (implying the alternative sol'n was suboptimal) or it produces an equally good solution that is "more similar to" the greedy one. (Defined by sets having more elements in common or permutations having more pairs on which they agree about the relative ordering.)

(3) Direct evidence of optimality.

3a. Supplement your algorithm to output "piece of evidence" in addition to solution.

3b. Argue that the piece of evidence furnishes an upper/lower bound on the quality of any solution and that your algo produces a solution that matches this bound.