CS 161
Summer 2018
8/2/18

Tutorial 6: Dynamic Programming Algorithms
Greedy Algorithms

Dynamic Programming Algorithms

1. **(Warmup with DP) (Difficulty: Easy to Medium)**
   For each of the following questions, define a recurrence that you could use to design algorithms.

   (a) Suppose we encode lowercase letters into a numeric string as follows: we encode a as 1, b as 2 ... and z as 26. How many letter strings can a given numeric string S of length n correspond to? For example, the numeric string 126 can correspond to three strings: "abf", "az", or "lf".

   (b) What's the probability that rolling k 6-sided fair dice will result in a sum S?

(c) Given an 8×8 chessboard and a knight that starts at position $A1$, how many ways the knight can end up at position xy after k moves? Knights move +/-1 squares in one direction and +/-2 squares in the other direction.

(d) We have a series of boxes, where box i has dimensions $h_i \times w_i \times d_i$. We want to create the tallest structure possible, where a box a can only be stacked on top of another box if 2D surface of a fits strictly within the surface in contact with b. You are allowed to rotate boxes as you please. You may use each box as many times as you wish.

2. **(Taking stock) (Difficulty: Medium)**

Suppose you are given reliable insider information about the prices for k different stocks over the next n days. That is, for each day $i \in [0, n-1]$ and each stock $j \in [0, k-1]$, you're given $p_{i,j}$, the price at which stock j trades at on the ith day. You start with a budget of P dollars, and on each day, you may purchase or sell as many shares of each type of stock as you want, as long as you can afford them. (Assume that all prices are integer-valued and that you can only purchase whole stocks.) You have an earning goal of Q dollars. Here, we will design an algorithm to determine whether you can meet your goal, and if not, how much money you can earn.

(a) Suppose we are only looking at prices over two days (i.e. $n = 2$). Design an $O(kP)$-time dynamic programming algorithm that computes the amount of money you can make buying stocks on the first day and selling stocks on the second day. (*Hint: Let $M[l]$ be the amount of money you can make after investing l dollars. To start, write an expression for $M[l]$ in terms of $M[l']$ for $l' \leq l$.*)

[We are expecting: An English description of your algorithm and a brief justification of its runtime.]

(b) Now, suppose you are given prices over n days. Using your solution to part (a) as a guide, design an $O(nkQ)$-time algorithm that determines whether you can reach your goal, and if not, reports how much money you can make. (*Hint: Without loss of generality, you can imagine at the start of every day, you sell every stock you own, and purchase stocks with your correct earnings.*)

[We are expecting: An English description of your algorithm and a brief justification of its correctness and runtime.]

Greedy Algorithms

1. **(Restricted MST) (Difficulty: Easy)**

   Given an undirected weighted graph $G = (V, E)$, we have a set $U \subset V$. Design an algorithm to find a minimum spanning tree such that all vertices in U are leaf vertices. (The result may not be a MST of the original graph G.)

   [We are expecting: An English description of the algorithm.]

2. **(Transportation networks) (Difficulty: Medium)**

   Given a set of n cities, we would like to build a transportation system such that there is some path from any city to any other city. There are two ways to travel: by driving or by flying. Initially all of the cities are disconnected. It costs $c_{i,j}$ to build a road between city i and city j. It costs a_i to build an airport in city i. For any two cities i and j, we can fly directly from i to j if there is an airport in both cities.

   Design a $O(m \log(n))$-time algorithm for determining which roads and airports to build to minimize the cost of connecting the cities. Here, "connecting the cities" means that there should be some way to get from any city to any other. Your algorithm should take as input the costs $c_{i,j}$ and a_i, and return a list of roads and airports to build.

   [We are expecting: An English description of your algorithm and a brief justification of its runtime.]

3. **(Fair cake slicing) (Difficulty: Medium)**

You have a cake sliced into n differently sized pieces. You need to split it among $k \leq n$ students in your class and you want to give each student exactly 1 piece of cake. However, you know that since the pieces are differently sized, the students who end up with smaller pieces will most likely complain about unfairness. You want to minimize the complaints by minimizing unfairness. If you select k cake pieces of size $S_1, \ldots, S_k$ then unfairness is defined as

$$\max\{S_1, \ldots S_k\} - \min\{S_1, \ldots S_k\}$$

.

(a) Design a very simple algorithm to maximize unfairness.

(b) Design an algorithm to minimize unfairness.