

# 15-451 Algorithms, Fall 2005

Homework # 4

due: Wed-Thurs, October 26-27, 2005

---

## Ground rules:

- This is an oral presentation assignment. You should work in groups of three. At some point before **Sunday, October 23 at 11:59pm** your group should sign up for a 1-hour time slot on the signup sheet on the course web page.
- You are not required to hand anything in at your presentation, but you may if you choose. If you do hand something in, it will be taken into consideration (in a non-negative way) in the grading.

## Problems:

### 1. [Boruvka's MST Algorithm]

Boruvka's MST algorithm (from 1926) is a bit like a distributed version of Kruskal. We begin by having each vertex mark the shortest edge incident to it. (For instance, if the graph were a 4-cycle with edges of lengths 1, 3, 2, and 4 around the cycle, then two vertices would mark the "1" edge and the other two vertices will mark the "2" edge.) For the sake of simplicity, assume that all edge lengths are distinct so we don't have to worry about how to resolve ties. This creates a forest  $F$  of marked edges. (*Convince yourself why there won't be any cycles!*) In the next step, each tree in  $F$  marks the shortest edge incident to it (the shortest edge having one endpoint in the tree and one endpoint not in the tree), creating a new forest  $F'$ . This process repeats until we have only one tree.

- (a) Show correctness of this algorithm by arguing that the set of edges in the current forest is always contained in the MST.
  - (b) Show how you can run each iteration of the algorithm in  $O(m)$  time with just a couple runs of Depth-First-Search and no fancy data structures. (Remember, this algorithm was from 1926!)
  - (c) Prove an upper bound of  $O(m \log n)$  on the running time of this algorithm.
2. [Road trip!] Velma and the gang are going on a road trip to San Francisco immediately after their midterms. To plan the trip, they have laid out a map of the U.S., and marked all the places they think might be interesting to visit along the way. However, the requirements are:
- (a) Each stop on the trip must be closer to SF than the previous stop.
  - (b) The total length of the trip can be no longer than  $D$ .

Velma wants to visit the most places subject to these conditions. As a first step, she creates a DAG with  $n$  nodes (one for each location of interest) and an edge from  $i$  to  $j$  if there is a road from  $i$  to  $j$  and  $j$  is closer to SF than  $i$ . Let  $d_{ij}$  be the length of edge  $(i, j)$  in this graph.

Help out Velma by giving an  $O(mn)$ -time algorithm to solve her problem. Specifically, given a DAG  $G$  with lengths on the edges, a start node  $s$ , a destination node  $t$ , and a distance bound  $D$ , your algorithm should find the path in  $G$  from  $s$  to  $t$  that visits the most intermediate nodes, subject to having total length  $\leq D$ .

(Note that in general graphs, this problem is NP-complete: in particular, a solution to this problem would allow one to solve the *traveling salesman problem*. However, the case that  $G$  is a DAG is much easier.)

3. [Options pricing] An *option* (specifically, an “American call option”) gives the holder the right to purchase some underlying asset (e.g., one share of IBM) at some specified exercise price (e.g., \$100) within some specified time period (e.g., 1 year). The value of an option depends on the current price of the asset, the exercise price of the option, the length of the time period, and one’s beliefs about how the asset’s price is likely to behave in the future.

For example, to take an easy case, suppose we have an option to buy one share of IBM at \$100 that expires right now. If IBM is currently going for \$105, then the value of this option is \$5. If IBM is currently going for \$95, then the value of this option is \$0 (we wouldn’t want to exercise it). However, suppose the option expires tomorrow, and suppose we have some simple model of how IBM shares behave. E.g., perhaps our model says that each day, with probability  $1/4$  the share price goes up by \$10, and with probability  $3/4$  the share price goes down by \$5. In that case, if IBM is currently worth \$95, then the value of this option is \$1.25 because that is our expected gain if we use the optimal strategy “wait until tomorrow, and then exercise the option if IBM went up” (our expected gain is  $\frac{1}{4} \times 5 + \frac{3}{4} \times 0$ ). If IBM is currently worth \$105, then the value of the option is \$5 (and our optimal strategy is to exercise the option right away).

Formally, the value of an option is its expected value under the optimal strategy for using it, given our probabilistic model for the stock. Note that the optimal strategy for using the option need not commit in advance to what day the option will be exercised.

Assume stock prices are integers between 0 and  $B$ . Suppose we are given a probabilistic model  $p_{ij}$  for how the stock behaves: specifically, if the stock has price  $i$  on day  $t$ , then  $p_{ij}$  is the probability that the stock will have price  $j$  on day  $t + 1$ . So, for each  $i$ ,  $\sum_j p_{ij} = 1$ . Give a dynamic-programming algorithm to calculate the value of an option of exercise price  $X$  that expires  $T$  days in the future, given that the current price of the stock is  $S$ . The running time of your algorithm should be  $O(B^2T)$ .