HW 5 (due Tuesday, at noon, March 11, 2014)

CS 473: Fundamental Algorithms, Spring 2014

Make sure that you write the solutions for the problems on separate sheets of paper. Write your name and netid on each sheet.

Collaboration Policy: The homework can be worked in groups of up to 3 students each.

---

1. (30 PTS.) Stock Picking.

You have a group of investor friends who are looking at n consecutive days of a given stock at some point in the past. The days are numbered. $i = 1, 2, \ldots, n$. For each day i, they have a price $p(i)$ per share for the stock on that day.

For certain (possibly large) values of k, they want to study what they call *k-shot strategies*. A k-shot strategy is a collection of m pairs of days $(b_1, s_1), \ldots, (b_m, s_m)$, where $0 \leq m \leq k$ and

$$1 \leq b_1 < s_1 < b_2 < s_2 \cdots < b_m < s_m \leq n.$$

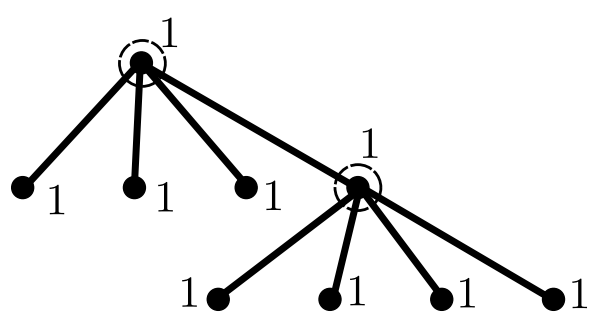
We view these as a set of up to k nonoverlapping intervals, during each of which the investors buy 1,000 shares of the stock (on day b_i) and then sell it (on day s_i). The *return* of a given k-shot strategy is simply the profit obtained from the m buy-sell transactions, namely,

$$1000 \cdot \sum_{i=1}^{m} (p(s_i) - p(b_i)) .$$

Design an efficient algorithm that determines, given the sequence of prices, the k-shot strategy with the maximum possible return. Since k may be relatively large, your running time should be polynomial in both n and k. Full credit for a solution that uses only $O(n)$ space.

2. (35 PTS.) Dominating Set in a Tree

Let $G = (V, E)$ be an undirected graph. A subset $S \subseteq V$ of nodes in G is called a *dominating set* if for all $v \in V$, we have $v \in S$ or there is some node $u \in S$ such that $(u, v) \in E$. In other words every node in $V \setminus S$ is connected by an edge to some node in S. Given non-negative weights $w(v)$ on the nodes of V the goal is to find a minimum-weight dominating set in G. This problem is known to be NP-Hard in general graphs. Describe a polynomial time algorithm for this problem when G is a tree. See the example below to note that two adjacent nodes may be in an optimal dominating set.



3. (30 PTS.) Dinner Planning

A party of n people have come to dine at a fancy restaurant and each person has ordered a different item from the menu. Let $D_1, D_2, \ldots, D_n$ be the items ordered by the diners. Since this is a fancy place, each item is prepared in a two-stage process. First, the head chef (there is only one head chef) spends a few minutes on each item to take care of the essential aspects and then hands it over to one of the many sous-chefs to finish off. Assume that there are essentially an unlimited number of sous-chefs who can work in parallel on the items once the head chef is done. Each item D_i takes h_i units of time for the head chef followed by s_i units of time for the sous-chef (the sous-chefs are all identical). The diners want all their items to be served at the same time which means that the last item to be finished defines the time when they can be served. The goal of the restaurant is to serve the diners as early as possible. Give an efficient algorithm to decide the order in which the head chef should prepare the items so as to minimize the time to serve the diners.