Problem Set 12

Data Structures and Algorithms, Fall 2019

Due: December 12, in class.

From CLRS

Exercise 25.1-5, 25.1-9, 25.2-4, 25.2-7, 25.3-1, 25.3-6, 15.1-5, 15.2-4, 15.3-2, 15.3-5.

Additional Problem One

All of the all-pairs shortest paths algorithms discussed in class fail if the graph contains a negative cycle. Johnson's algorithm detects the negative cycle in the initialization phase (via Bellman-Ford) and aborts; the dynamic programming algorithms just return incorrect results. However, all of these algorithms can be modified to return correct shortest-path distances, even in the presence of negative cycles. Specifically, for all vertices u and v:

- If u cannot reach v, the algorithm should return $dist[u, v] = \infty$.
- If u can reach a negative cycle that can reach v, the algorithm should return $dist[u, v] = -\infty$.
- Otherwise, there is a shortest path from u to v, so the algorithm should return its length.

(a) Describe how to modify Johnson's algorithm to return the correct shortest-path distances, even if the graph has negative cycles.

(b) Describe how to modify the recursive algorithm (on length of paths) to return the correct shortest-path distances, even if the graph has negative cycles.

(c) Describe how to modify Floyd-Warshall algorithm to return the correct shortest-path distances, even if the graph has negative cycles.

Additional Problem Two

Suppose you are a simple shopkeeper living in a country with n different types of coins, with values $1 = c[1] < c[2] < \cdots < c[n]$. (In the U.S., for example, $n = 6$ and the values are 1, 5, 10, 25, 50 and 100 cents.) Your beloved and benevolent dictator, El Generalissimo, has decreed that whenever you give a customer change, you must use the smallest possible number of coins, so as not to wear out the image of El Generalissimo lovingly engraved on each coin by servants of the Royal Treasury.

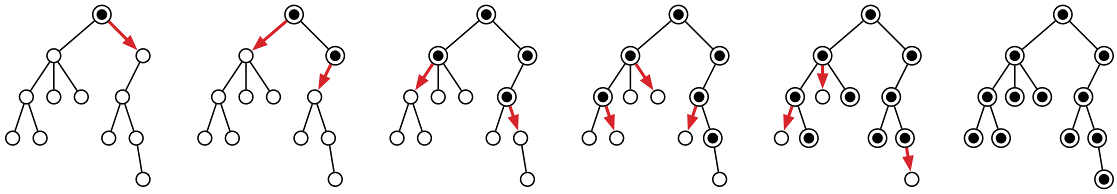
(a) In the United States, there is a simple greedy algorithm that always results in the smallest number of coins: subtract the largest coin and recursively give change for the remainder. El Generalissimo does not approve of American capitalist greed. Show that there is a set of coin values for which the greedy algorithm does *not* always give the smallest possible of coins.

(b) Suppose El Generalissimo decides to impose a currency system where the coin denominations are consecutive powers $b^0, b^1, b^2, \cdots, b^k$ of some integer $b \geq 2$. Prove that despite El Generalissimo's disapproval, the greedy algorithm described in part (a) does make optimal change in this currency system.

(c) Describe and analyze an efficient algorithm to determine, given a target amount T and a sorted array $c[1 \cdots n]$ of coin denominations, the smallest number of coins needed to make T cents in change. Assume that $c[1] = 1$, so that it is possible to make change for any amount T.

Additional Problem Three

Suppose we need to broadcast a message to all the nodes in a rooted tree. Initially, only the root knows the message. In a single round, any node that knows the message can forward it to at most one of its children. See the following figure for an example, in which a message is distributed in five rounds.



(a) Design and analyze an efficient algorithm to compute the minimum number of rounds required to broadcast the message to all nodes in a *binary* tree.

(b) Design and analyze an efficient algorithm to compute the minimum number of rounds required to broadcast the message to all nodes in an *arbitrary* rooted tree.

Additional Problem Four

Describe and analyze an efficient algorithm to find the length of the longest *contiguous* substring that appears both forward and backward in an input string $T[1 \cdots n]$. The forward and backward substrings must *not* overlap. Here are several examples:

- Given the input string `ALGORITHM`, your algorithm should return 0.
- Given the input string `RECURSION`, your algorithm should return 1, for the substring `R`.
- Given the input string `REDIVIDE`, your algorithm should return 3, for the substring `EDI`. (The forward and backward substrings must not overlap!)
- Given the input string `DYNAMICPROGRAMMINGMANYTIMES`, your algorithm should return 4, for the substring `YNAM`. (In particular, it should not return 6, for the subsequence `YNAMIR`).