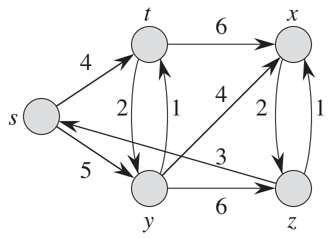
Problem Set 11

Data Structures and Algorithms, Fall 2021

Due: December 16, in class.

Problem 1

(a) Run Dijkstra's algorithm on the following directed graph, first using vertex s as the source and then using vertex z as the source. Show the $dist$ and $parent$ values and the vertices in "the known region R" after each iteration of the $while$ loop.



(b) Give a simple example of a directed graph with negative-weight edges for which Dijkstra's algorithm produces incorrect answers.

(c) Give a simple example of an undirected graph containing a vertex s such that the minimum spanning tree and the shortest-path tree rooted at s are different.

Problem 2

We are given a directed graph $G = (V, E)$ on which each edge $(u, v) \in E$ has an associated value $r(u, v)$, which is a real number in the range $0 \le r(u, v) \le 1$ that represents the reliability of a communication channel from vertex u to vertex v. We interpret $r(u, v)$ as the probability that the channel from u to v will not fail, and we assume that these probabilities are independent. Devise an efficient algorithm to find the most reliable path between two given vertices s and t. You need to give the pseudocode of your algorithm and analyze its running time.

Problem 3

For any edge e in any graph $G = (V, E)$, let $G \setminus e$ denote the graph obtained by deleting e from E. Suppose we are given a graph G and two vertices $s \in V$ and $t \in V$. The *replacement paths* problem asks us to compute the shortest-path distance from s to t in $G \setminus e$, for every edge e of E. The output is an array of $|E|$ distances, one for each edge of G.

(a) Suppose G is a directed graph, and there is a unique shortest path from vertex s to vertex t that passes through every vertex of G. Devise an algorithm to solve this special case of the replacement paths problem in $O((|V| + |E|) \log |V|)$ time.

(b) Devise an algorithm to solve the replacement paths problem for arbitrary undirected graphs in $O((|V| + |E|) \log |V|)$ time.

In both subproblems, you may assume that all edge weights are non-negative. You need to briefly argue the correctness of your algorithms.

Problem 4

The greedy algorithm we described in class for the activity selection problem is not the only greedy strategy we could have tried. For each of the following alternative greedy strategies, either claim that the resulting algorithm always constructs an optimal solution, or describe a small input example for which the algorithm does not produce an optimal solution (that is, either answer "correct strategy" *without* proof, or answer "incorrect strategy" and give a counterexample). Assume that all algorithms break ties arbitrarily (that is, in a manner that is completely out of your control).

(a) Choose the activity x that ends last, discard all activities that conflict with x, and recurse.

(b) Choose the activity x that starts first, discard all activities that conflict with x, and recurse.

(c) Choose the activity x that starts last, discard all activities that conflict with x, and recurse.

(d) Choose the activity x with shortest duration, discard all activities that conflict with x, and recurse.

(e) Choose an activity x that conflicts with the fewest other activities, discard all activities that conflict with x, and recurse.

(f) If no activities conflict, choose them all. Otherwise, discard the activity with longest duration and recurse.

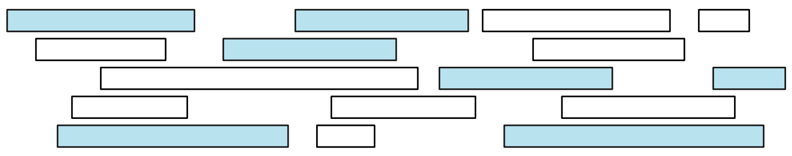
(g) If no activities conflict, choose them all. Otherwise, discard an activity that conflicts with the most other activities and recurse.

(h) If any activity x completely contains another activity, discard x and recurse. Otherwise, choose the activity y that ends last, discard all activities that conflict with y, and recurse.

Problem 5

Let X be a set of n intervals on the real line. We say that a subset of intervals $Y \subseteq X$ *covers* X if the union of all intervals in Y is equal to the union of all intervals in X. The size of a cover is just the number of intervals in the cover. See following figure for an example.

Devise an efficient algorithm to compute the smallest cover of X. Assume that your input consists of two arrays $L[1 \cdots n]$ and $R[1 \cdots n]$, representing the left and right endpoints of the intervals in X. (For simplicity, you may assume all $2n$ endpoints are distinct.) You need to give the pseudocode of your algorithm and analyze its runtime. You also need to prove that it is correct.


A set of intervals, with a cover (shaded) of size 7.

Problem 6

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.