

Homework 1

Due: Friday, Sept 26, 11:59 PM PT

1. Design a data structure that has the following properties (assume n elements in the data structure, and that the data structure properties need to be preserved at the end of each operation):

- a. Find-median takes $O(1)$ time
- b. Extract-Median takes $O(\log n)$ time
- c. Insert takes $O(\log n)$ time

Do the following:

- a) Describe how your data structure is designed. (5 points)
 - b) Give algorithms that implement the Extract-Median() and Insert() functions. (8 points)
2. Given an undirected graph $G = (V, E)$ where a set of data centers V is connected by a network of optical links E . Each link (u, v) has a positive latency cost $l(u, v)$. There is a proposal to add a new optical link to the network. The proposal specifies a list C of candidate pairs of data centers between which the new link may be established. Your task is to choose the link that yields the greatest reduction in end-to-end latency between a given source data center s and target data center t . Design an efficient algorithm for this problem. No proof is required. Give the runtime complexity of your algorithm. (Note that your algorithm's time complexity should not be worse than Dijkstra's shortest path algorithm.) (12 points)
 3. A network of n servers under your supervision is modeled as an undirected graph $G = (V, E)$ where a vertex in the graph corresponds to a server in the network and an edge models a link between the two servers corresponding to its incident vertices. Assume G is connected. Each edge is labeled with a positive integer that represents the cost of maintaining the link it models. Further, there is one server (call its corresponding vertex as S) that is not reliable and likely to fail. Due to a budget cut, you decide to remove a subset of the links while still ensuring connectivity. That is, you decide to remove a subset of E so that the remaining graph is a spanning tree. Further, to ensure that the failure of S does not affect the rest of the network, you also require that S is connected to exactly one other vertex in the remaining graph. Design an algorithm that, given G and the edge costs, efficiently decides if it is possible to remove a subset of E , such that the remaining graph is a spanning tree where S is connected to exactly one other vertex and (if possible) finds a solution that minimizes the sum of maintenance costs of the remaining edges. (10 points)

Ungraded Problems:

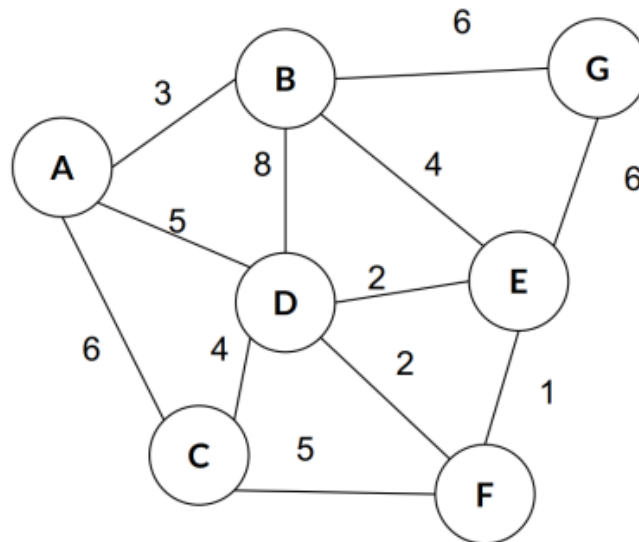
- Imagine a dynamically resizing array that supports the following operations:
 - Insert(X): Adds X to the end of the array if it is not full. If full, first double the allocated memory of the array and then insert. The doubling operation takes time proportional to the current size of the array.
 - Delete(): Deletes the most recently inserted element, and halves the allocated memory of the array if the array becomes at most half filled as a result. The halving operation takes time proportional to the current size of the array. (If the current allocated memory is an odd number, the new memory is the floor of its half)

Suppose the array begins at size = 1.

- For a sequence of n insert operations, show that the amortized cost of each operation is $\Theta(1)$. Use accounting methods.
- For a sequence of n insertions followed by n deletions, show that the amortized cost of each operation is $\Theta(1)$. Use an aggregating method.
- For arbitrary n , show a sequence of n Insert/Delete operations (in a suitable order) which have total running time $\Theta(n^2)$, making the amortized cost of the involved operations $\Theta(n)$ for this sequence.

In all the parts, you can assume n to be a power of 2 to simplify your analysis if needed.
(15 points)

- Considering the following graph G:



1. In graph G, if we use Kruskal's Algorithm to find the MST, what is the third edge added to the solution? Select all correct answers (3 points)

- E-F
- D-E
- A-B
- C-F
- D-F

2. In graph G , if we use Prim's Algorithm to find MST starting at A , what is the second edge added to the solution? (3 points)

- a. B-G b. B-E c. D-E d. A-D e. E-F

3. What is the cost of the MST in the Graph? (3 points)

- a. 18 b. 19 c. 20 d. 21 e. 22

3. Prove or disprove the following:

T is a spanning tree on an undirected graph $G = (V, E)$. Edge costs in G are NOT guaranteed to be unique. If every edge in T belongs to SOME minimum cost spanning trees in G , then T is itself a minimum cost spanning tree. (5 points)

4. You are given a minimum spanning tree T in a graph $G = (V, E)$. Suppose we remove an edge from G to create a new graph G_1 . Assume that G_1 is still connected, devise a linear time algorithm to find an MST in G_1 . (8 points)

5. Imagine you are in a car and you want to travel on a road. You need to go up and down several hills on this road. Fortunately, your car is specially equipped with a nitrogen booster in addition to a regular gasoline burning engine. You can only use the booster for a limited number of times (k), and you have m gallons of gas in your tank at the start. The car does not need to burn any gas or use the booster when you are coming down the hills or going straight on a flat road. But for each hill (with elevation gain of " h ") that you want to go up, you need to use one of these two options:

- 1) You can use the engine and burn ' h ' gallons of gas, or
- 2) You can spend one of your k special boosters (if you have any left). The booster can take the car from any height to any height.

You are given an array ' $\text{hills}[1..n]$ ' of heights for the hills you need to pass, in the order in which you encounter the hills. Your job is to find the maximum number of hills that you can pass with the given amount of gas (m gallons) and the given number of boosters (k). Thus, you need to design an algorithm so as to reach the furthest on this road by optimally using either your boosters or gas to climb up each hill. Your algorithm should run in $O(n \log k)$. Explain your algorithm. How much space does it take? (10 points)

6. Given an array of elements $A[1..n]$, what is the tight bound on the performance of the algorithm below, which places the elements of array A into a binomial heap H ? (8 points)

Initialize Heap H

For $i=1$ to n

 Insert $A[i]$ into H

Endfor