ShanghaiTech University

CS101 Algorithms and Data Structures

Fall 2021

Homework 8

Due date: 23:59, November 28, 2021

1. Please write your solutions in English.
2. Submit your solutions to gradescope.com.
3. Set your FULL NAME to your Chinese name and your STUDENT ID correctly in Account Settings.
4. If you want to submit a handwritten version, scan it clearly. Camscanner is recommended.
5. When submitting, match your solutions to the according problem numbers correctly.
6. No late submission will be accepted.
7. Violations to any of the above may result in zero grade.

1: (4×2') Choice

The following questions are choice questions, each question may have **one** or **multiple** correct answers. Select all the correct answer, you will get half points if you choose a strict subset(excluding empty set) of the right answer.

Note: You should write those answers ***in the box*** *below.*

Question 1	Question 2	Question 3	Question 4

Question 1. *Which of the following statements are/is true?*

(A) *Prim's algorithm for computing a minimum spanning tree of a graph only works if the weights of the edges are non-negative.*

(B) *A minimum spanning tree for a connected graph has exactly $N - 1$ edges, where N is the number of vertices in the graph.*

(C) *A connected graph that has unique edge weights (there are* ***no*** *two edges with the same weight) has a unique minimum spanning tree.*

(D) *None of above is True*

Question 2. *Which of the following statements are/is true?*

(A) *For Prim's algorithm, it is possible to calculate the minimum spanning tree of some graph in $\Theta(E)$ time without using priority queue and adjacent list. E represents the number of edges in the graph.*

(B) *Kruskal's algorithm adds the least-weighted edge each time to the MST.*

(C) *The removal of any edge from an MST will not disconnect this MST.*

(D) *None of above is True.*

Question 3. *You are given a connected undirected graph G with m distinct edges (distinct costs), in adjacency list representation. You are also given the edges of a minimum spanning tree T of G. This question asks how quickly you can recompute the MST if we change the cost of a single edge. Which of the following are/is true? (RECALL: The disjoint set data structure has run-time $O(\alpha(n))$, which is effectively a constant)*

(A) *Suppose $e \in T$ and we increase the cost of e. Then, the new MST can be recomputed in $O(m)$ deterministic time.*

(B) *Suppose $e \notin T$ and we increase the cost of e. Then, the new MST can be recomputed in $O(m)$ deterministic time.*

(C) *Suppose $e \in T$ and we decrease the cost of e. Then, the new MST can be recomputed in $O(m)$ deterministic time.*

(D) *Suppose $e \notin T$ and we decrease the cost of e. Then, the new MST can be recomputed in $O(m)$ deterministic time.*

Question 4. *Consider the following algorithm that attempts to compute a minimum spanning tree of a connected undirected graph G with distinct edge costs. First, sort the edges in decreasing cost order (i.e., the opposite of Kruskal's algorithm). Initialize T to be all edges of G. Scan through the edges (in the sorted order), and remove the current edge from T if and only if it lies on a cycle of T.*
Which of the following statements is true?

(A) The output of the algorithm will never have a cycle, but it might not be connected.

(B) The algorithm always outputs a spanning tree, but it might not be a minimum cost spanning tree.

(C) The output of the algorithm will always be connected, but it might have cycles.

(D) The algorithm always outputs a minimum spanning tree.

2: (8') Play with Prim and Kruskal

Here we give the adjacent list of a DAG(Directed Acyclic Graph). The letters on the left of the arrows are the source vertices of the edges. The destination vertex v and weights of the edge w is given in the form of $\boxed{v \mid w}$ on the right of the arrows.

$$\boxed{A} \rightarrow \boxed{B \mid 3} \rightarrow \boxed{C \mid 5} \rightarrow \boxed{D \mid 2}$$
$$\boxed{B} \rightarrow \boxed{C \mid 4} \rightarrow \boxed{D \mid 6} \rightarrow \boxed{E \mid 4}$$
$$\boxed{C} \rightarrow \boxed{D \mid 2}$$
$$\boxed{D} \rightarrow \boxed{E \mid 10} \rightarrow \boxed{F \mid 7}$$
$$\boxed{E} \rightarrow \boxed{F \mid 3}$$
$$\boxed{F}$$

Suppose we start from vertex "A".

(1)(2') Draw the graph of the DAG with the adjacent list given above. Annotate the edges with their weight.

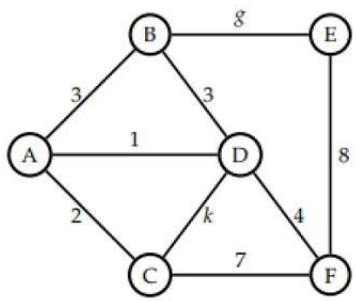
(2)(2') View the DAG above as an undirected graph and draw its **maximum** spanning tree.

(3)(2') Write down the sequence of edges added to the **maximum** spanning tree using Kruskal's algorithm.

(4)(2') Write down the sequence of edges added to the **maximum** spanning tree using Prim's algorithm.

3: (12') Minimum Spanning Tree

We are given the following graph G:



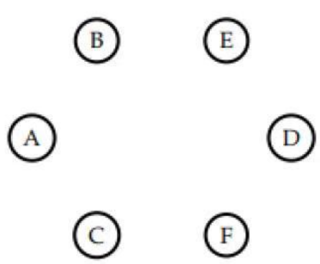
For each question below, please give your answer firstly and then explain it briefly.
Note that questions below are independent to each other.
(1)(2') For what range of values of g are you **guaranteed** to include edge BE in the MST?

(2)(2') For what range of values of k are you **guaranteed** to include edge CD in the MST?.

(3)(2') For what range of values of k are you **guaranteed to NOT** include edge CD in the MST?.

(4)(3') Suppose an adversary can set g and k arbitrarily. What is the maximum cost of a MST that the adversary can force?

(5)(3') Draw seven edges on the following graph with six vertices, and then mark six edges with cost=1 and one edge with cost=x such that an adversary can choose x such that the cost of the MST is arbitrarily large.



4: (8') Building A Tower

We build up a tower with a pile of stones. Each layer of the tower is exactly one piece of stone. We index the layers from top to bottom, so layer 1 is the top layer. The stone weight of the i-th layer is s_i. The tower will fall if there exists a layer i s.t. $\Sigma_{j=1}^{i-1} s_j > s_i$. For example, if the tower is built with 5 stones and the stone weight from top to bottom is 2, 3, 5, 9, 100, then the tower would fall because $9 < 5 + 3 + 2 = 10$.

You are the designer of the tower and you have n stones. Each stone has different weight and the same height. Can you build the tower as high as possible? Give an $O(nlogn)$ algorithm and prove that the algorithm produces the highest tower. Show the time complexity of your algorithm. Your answer should contain at least the following parts:

1. Main idea of your algorithm (which can also include pseudo-code here). (2')
2. Proof of the correctness of your algorithm. (4')
3. Time complexity analysis. (2')

5: (9') To be A Grandparent

Assume you are a grandparent and going to give your grandchildren some pieces of cake. However you cannot satisfy a child unless the size of cake piece he receives is no less than his expected size. Different children may have different expected sizes. Meanwhile, you can't give each child more than one piece. For example, if three children's expected sizes are 1, 3, 4 and the sizes of pieces of cake are 1, 2, then you could only satisfy the first child. Given the expected sizes of the children and the sizes of cake pieces that you have, how can you make the most children satisfied? Show that your algorithm is correct and analyze the time complexity. Same as Q4, your answer should contain at least the following parts:

1. Main idea of your algorithm (which can also include pseudo-code here). (3')
2. Proof of the correctness of your algorithm. (4')
3. Time complexity analysis. (2')