



Quiz 3

8th Nov 2017, 8:00 am – 8:20 am

Course Code: CS302	Course Name: Design and Analysis of Algorithm
Instructor Name / Names: Subhash Sagar	
Student Roll No:	Section: B

Time: 20 minutes.

Max Marks: 5 points

Question 1:

Say we are given a weighted directed graph G with strictly positive edge weights $w(u, v)$ and a source node s in G . We would like to modify Dijkstra's shortest-path algorithm to produce a *count* of the number of different minimal-length paths from s to v for every node v .

Recall that Dijkstra's algorithm builds a set X of nodes incrementally, starting with $X = \{s\}$. Normally, the algorithm maintains a value $D(v)$ for each node v giving the length of the shortest path from s to v through only intermediate nodes in X .

We will modify the algorithm to maintain an extra value $N(v)$ giving the number of paths of length $D(v)$ from s to v through only intermediate nodes in X .

Describe how to initialize $D(v)$ and $N(v)$ and how to update the values of $D(v)$ and $N(v)$ when a new node u is added to X . Describe your modifications in words, do not write any code.

Solution:

Initialize $D(v)$ as in Dijkstra's algorithm. Initialize $N(s) = 1$, $N(v) = 1$ for all edges (s, v) , and $N(v) = 0$ for all other nodes.

To update N when a node u is added to X , for each edge (u, v) ,

- if $D(u) + w(u, v) < D(v)$, set $N(v) = N(u)$;
- if $D(u) + w(u, v) = D(v)$, set $N(v) = N(u) + N(v)$;
- if $D(u) + w(u, v) > D(v)$, do nothing.

Update $D(v)$ for all edges (u, v) as in Dijkstra's algorithm.

Quiz 3

8th Nov 2017, 9:00 am – 9:20 am

Course Code: CS302	Course Name: Design and Analysis of Algorithm
Instructor Name / Names: Subhash Sagar	
Student Roll No:	Section: E

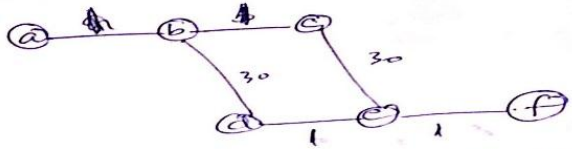
Time: 20 minutes.

Max Marks: 5 points

Professor Borden proposes a new divide-and-conquer algorithm for computing minimum spanning trees, which goes as follows. Given a graph $G = (V, E)$, partition the set V of vertices into two sets V_1 and V_2 such that $|V_1|$ and $|V_2|$ differ by at most 1. Let E_1 be the set of edges that are incident only on vertices in V_1 , and let E_2 be the set of edges that are incident only on vertices in V_2 . Recursively solve a minimum-spanning-tree problem on each of the two sub-graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Finally, select the minimum-weight edge in E that crosses the cut (V_1, V_2) , and use this edge to unite the resulting two minimum spanning trees into a single spanning tree. Either argue that the algorithm correctly computes a minimum spanning tree of G , or provide an example for which the algorithm fails, or both? Also, prove that this Algorithm works for negative weight edges. (**Note:** Prove using graph with at least 6 vertices).


Solution


Suppose for the graph



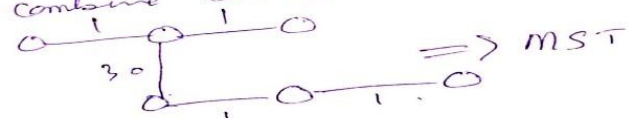
\Rightarrow It depends on the partition of the graph.

Suppose the partitions are G_1 & G_2 .

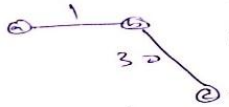
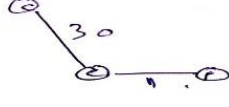
$G_1 =$ 

$G_2 =$ 


And if we combine both G_1 & G_2 with min. weight



\Rightarrow MST

& if $G_1 =$  & $G_2 =$ 

& if we combine, we get



is NOT a MST

So, the answer algo may or may not work properly.



National University of Computer & Emerging Sciences, Karachi
Fall-2017 CS-Department



Quiz 3

8th Nov 2017, 2:00 pm – 2:20 pm

Course Code: CS302	Course Name: Design and Analysis of Algorithm
Instructor Name / Names: Subhash Sagar	
Student Roll No:	Section: D

Time: 20 minutes.

Max Marks: 5 points

Question 1:

Minimum bottleneck spanning tree (MBST) in an undirected graph is a spanning tree in which the most expensive edge is as cheap as possible among all the possible combinations of spanning trees. A bottleneck edge is the highest weighted edge in a spanning tree. A spanning tree is minimum bottleneck spanning trees if the graph contains a spanning tree with a minimum bottleneck edge weight. More clearly, for a graph; list all the possible spanning trees. Among these pick the spanning tree whose maximum edge weight is minimum.

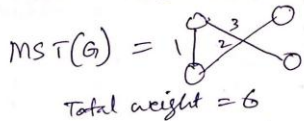
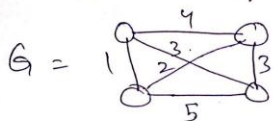
Prove or Disprove with example using graph? (Graph must be of at least 4 vertices).

- 1) Whether a MST is always a MBST? And**
- 2) A MBST is always a MST?**

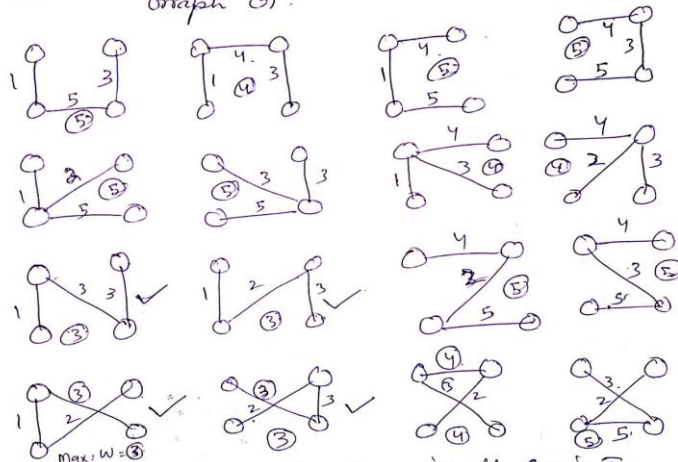
Solution.

Part(a)

Suppose the Graph is



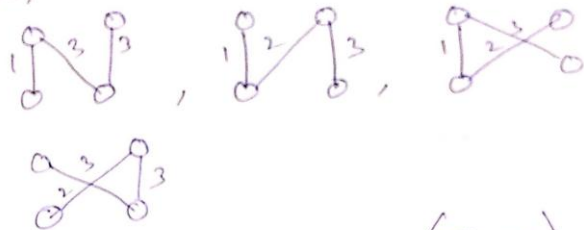
For MBST: Firstly, find all the spanning trees of Graph G.



⇒ Select max: weight edge from all spanning trees
 ⇒ Then select min: from all max: weight edges which is 3. So the ~~graph~~ ^{Tree} with ~~weight~~ ^{edge} with weight = 3 are MBSTs.
 (min)

Part(b)

So, we have Total 4- MBSTs.



① MST is always MBST (TRUE)
 Yes it is always true, as MBSTs contain MST.

② MBST is not always MST, as we can see above, if we select ~~any~~ MBST other than weight = 6 which is not MST as weight for MST is 6. So tree with weight = 6 are MBSTs others are not MSTs but are MBSTs.