Abeera Fatima
I180411
E

Question 5

Algorithm:

The algorithm will be a modified version of Fredman and Tarjan's algorithm which has a time complexity of O(m), where m is the number of nodes. Fredman and Tarjan's algorithm makes use of the Boruvka's algorithms, where the main idea is the same as the prims algorithm but instead of selecting one starting edge, multiple trees are made from various edges, which ultimately contract into one MST.

The Fredman and Trajan's algorithm takes this idea and limits the size of the heaps storing the trees to a value k. The value of k is changed according to the number of trees remaining i.e. the more the trees the smaller the value of k will be.

Analysis of the algorithm:

Suppose the value of k for the first phase is:

$K_i = 2^{2m/n_i}$
Then the time for one phase will become:

$O(m + n_i \log(k_i))$
$= O(m + n_i \log(2^{2m/n_i}))$
$= O(m)$.

This multiplied with the number of trees will give us the total time complexity.

We know that each tree touches Ki edges.

So the number of trees x Ki will be less than $\leq 2m$

the number of trees $\leq 2m / Ki$

$n_{i+1} \leq 2m / Ki$ (the number of trees = number of nodes at next level of contraction)
$Ki+1= 2^{2m/n_{i+1}} \geq 2^{k_i}$.

So the Ki s are growing.

$k_i \geq n$

After using the laws of logarithms (excess proof not derived here), the algorithm reduces to a complexity of $O(m \beta(m,n))$. This is a linear time complexity, however the devised algorithm uses loops to compute the minimum weight so it will have a quadratic time complexity overall.

An overall lower time complexity than the prims algorithm.

Finding the Minimum Weight to obtain the minimum spanning tree:

1.Create an array MST of the size equal to the number of vertexes.

2.Create an array structure with all the edges and one bool variable to keep track of the added edges (bool was_added).

3. Assign -1 as key Values of all vertexes except the first/starting vertex. Give it a value of 0.

4. Till all vertexes are not added into the array:

Pick an adjacent vertex u, not already included in the tree (must have minimum key value.)

Add it to the MST array.

Set was_added of the edge used as true in the array of edges.

Update the key values of all adjacent vertexes of u by iterating through all adjacent vertexes v. if the weight of the edge u-v is less than the previous key value of v, update the new weight as the key value.

A minimum spanning tree will be returned.

5. Traverse through the array of edges and find the first edge with a false bool value and save it.

Create a variable min to store minimum weight and initialize it with the largest edge weight, and another variable to store the edge.

6. Repeat the steps above the number of times equal to the number of unadded edges except each time one of the unadded edge is considered.

While adding edges, when it's the turn of that unadded edge, its weight is made to be equal to one less than the key value of the edge that was added instead of it in the minimum spanning tree. So that this edge is considered instead.

The value of the minimum weight is updated if its less than that of the value of the variable and the corresponding edge noted.

7. After step 6. Is completed, the minimum weight to be added to a non-spanning edge to change the MST and the corresponding edge will be known.

A double loop will have the time complexity of $O(n^2)$. Array of edges will be traversed in the outer loop and the Fredman and Trajan's algorithm in the inner loop, and a polynomial time complexity will be achieved.