Name: Georges Hatem

CS590 Minimum Spanning Tree Application Programming Assignment

Due Date: May 4, 2022

Give an algorithm that, given a diagram and two switching centers a and b, will output the maximum bandwidth of a path between a and b. What is the running time of your algorithm?

Describe an efficient algorithm for finding a maximum spanning tree in G, which would maximize the bandwidth between two switching centers.

Describe the problem in terms of input and expected output clearly.

This can be accomplished by modifying Dijkstra's algorithm. Instead of representing the shortest path from a to u, the label D[u] represents the maximum bandwidth of any path from a to u. The maximum bandwidth for path from a through u to a vertex z adjacent to u is min{D[u], w((u, z))} so that the relaxation step updates D[z] to max{D[z], min{D[u], w((u, z))}}.

The algorithm is as follows:

MaxBandwidth (G, a, b)

Input: Weighted graph G and two distinguished vertices a and b
Output: Maximum bandwidth over all paths between a and b

Initialize the labels D[a] to infinity and D[u] to 0 for each vertex u! = a in G

Let there be a priority queue Q that contains all the vertices of G using the D labels as keys

```
while (Q != NULL)

    u ← Q.removeMaxElement()

        if (u == y)

            return D[u]

        else

            for each vertex z, adjacent to u such
            that z E Q do
```

bandwidth ← min {D[u], w(u,z)}

if (bandwidth > D[z])

D[z] <-bandwidth

change the key of z in Q to D[z]

Explanation:

1) No value is returned in the algorithm because the algorithm will find the vertex.
2) The label D[a] is initialized to infinity because the minimum value is found for D[a].
3) The label D[u] is initialized to zero as it is required to find the maximum value.

The running time of the algorithm is the same as Dijkstra's Algorithm. In this scenario, the adjacency list is being used. So, the running

time complexity is O((n+m)log(n)) if the priority queue is being implemented as a heap, and O(n^2) if the priority queue is being implemented as an unsorted sequence. (Or even O(n log n + m) using a fancier data structure to implement the priority queue.)

This above describes an efficient algorithm for finding a maximum spanning tree in G, which would maximize the bandwidth between two switching centers, and it describes the problem in terms of input and expected output clearly (Input being Graph G with 2 switching center vertices a and b and output being the maximum bandwidth over all paths between a and b). The algorithm and running time of the algorithm is provided above as well.

<u>Develop a program that accepts a network G of switching centers and the bandwidth between them (not all are connected directly with each other) and two switching centers a and b; it will output the maximum bandwidth between any two switches a and b.</u>

The program is provided in the zip file named Telephone_Network_Hatem.zip

Here is result of my testing (you should be able to get it by running the program as well):

```
Covered Nodes = 0
Next selected edge: (0,4) Bandwidth = 5
Covered Nodes = 0 4

Next selected edge: (4,3) Bandwidth = 7
Covered Nodes = 0 3 4

Next selected edge: (3,1) Bandwidth = 6
Covered Nodes = 0 1 3 4

Next selected edge: (3,2) Bandwidth = 6
Covered Nodes = 0 1 2 3 4

Edges in Maximum Spanning Tree:
0 --> 0
3 --> 1
3 --> 2
4 --> 3
0 --> 4
```