CS2023 - Data Structures and Algorithms

In-class Lab Exercise

Week 11

Name: A.A.H. Pramuditha

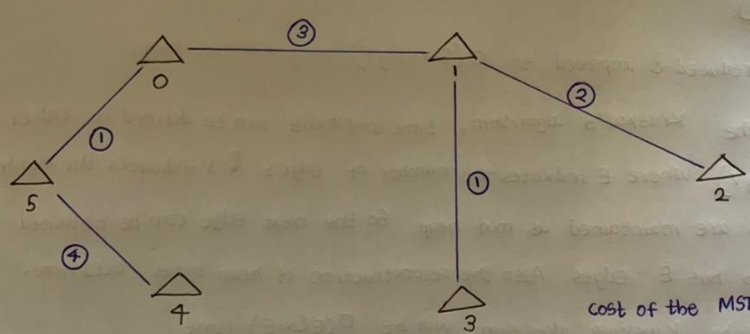
Index: 200476P

Question 1

Q1 Adjacency matrix for the given graph.

$$\begin{array}{c|cccccc} & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline 0 & 0 & 3 & 0 & 0 & 0 & 1 \\ 1 & 3 & 0 & 2 & 1 & 10 & 0 \\ 2 & 0 & 2 & 0 & 3 & 0 & 5 \\ 3 & 0 & 1 & 3 & 0 & 5 & 0 \\ 4 & 0 & 10 & 0 & 5 & 0 & 4 \\ 5 & 1 & 0 & 5 & 0 & 4 & 0 \end{array}$$

Q2 Starting from Node 3



cost of the MST = 1+2+ 3+1+4

C = 11 //

Q4 Yes, both MSTs have the same edges with same cost. If each edge of the tree has a distinct weight then there will only be one, unique minimum spanning tree.

3)

```
"D:\Semester 4\CS2023 - Data Structures and Algorithms\In Class labs\In-class Lab 11\MST_with_Prim's_Algorithm\bin\Debug\MST_with_Prim's_Algorithm.exe"
Edge   Weight
0 - 1    3
1 - 2    2
1 - 3    1
5 - 4    4
0 - 5    1

Process returned 0 (0x0)   execution time : 0.014 s
Press any key to continue.
```

(Q5) When it comes to the prim's Algorithm, if adjacancy list is used to represent the graph, using BFS all the vertices can be traversed in $O(V+E)$ time. Then we can use a min heap for storing the vertices not yet included in the MST.

This min heap can be used ~~tg~~ a priority queue to get the minimum weight edge. In min heap, there are operations like extracting minimum element & decreasing the key values which takes $O(\log V)$ time.

So overall time complexity can be given as,

$$= O(E+V) \cdot O(\log V)$$

$$= O[(E+V)\log V]$$

$$= O[E \log V]$$

This can be further reduced & improved to $O(E+V\log V)$ using Fibonacci heap.

But when it comes to the Kruskal's algorithm, time complexity can be defined as either $O(E \log E)$ or $O(V \log V)$ where E indicates the number of edges & V indicates the number of vertices. The edges are maintained as min heap. So the next edge can be obtained in $O(\log E)$ time if graph has E edges. Also the reconstruction of heap takes $O(E)$ time.

So overall time taken by Kruskal's algorithm will be $O(E \log E)$ time.

Link for GitHub repository: [hashirupramuditha/CS2023---Data-Structures-Algorithms (github.com)]()