In class Lab 11 – MST

Name: Meddepola M.A.C.T

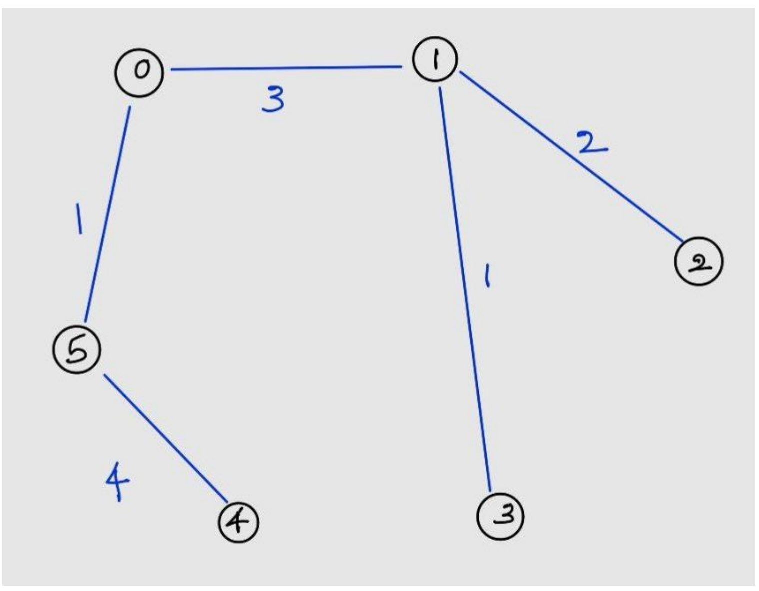
Index No: 210383L

GitHub Repository:
https://github.com/Thejas0604/CSE-labs

(01) Adjacency matrix representation:

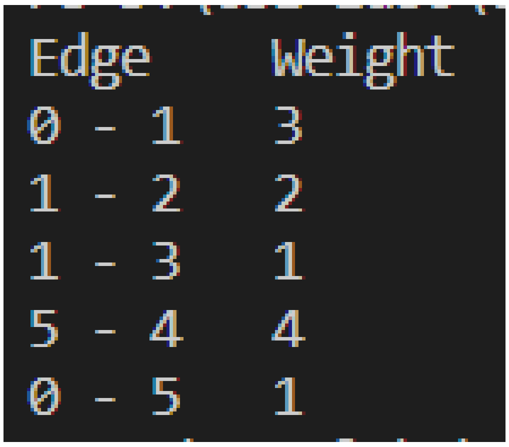
$$\begin{pmatrix} 0 & 3 & 0 & 0 & 0 & 1 \\ 3 & 0 & 2 & 1 & 10 & 0 \\ 0 & 2 & 0 & 3 & 0 & 5 \\ 0 & 1 & 3 & 0 & 5 & 0 \\ 0 & 10 & 0 & 5 & 0 & 4 \\ 1 & 0 & 0 & 5 & 4 & 0 \end{pmatrix}$$

(02) MST for the given scenario:



Cost = 11

(03) MST, that will form when starting vertex is 0.



```
Edge    Weight
0 - 1   3
1 - 2   2
1 - 3   1
5 - 4   4
0 - 5   1
```

(04)

- As we can this is same as the previous one. So, in this case MST is unique.
- **If each edge has a distinct weight, then there will be only one, unique minimum spanning tree.**
- Since this graph satisfies the above statement, this has a unique MST.

(05)

- Let’s consider Prim's Algorithm.
    - The algorithm starts with an empty spanning tree. The idea is to maintain two sets of vertices. The first set contains the vertices already included in the MST, and the other set contains the vertices not yet included. At every step, it considers all the edges that connect the two sets and picks the minimum weight edge from these edges. After picking the edge, it moves the other endpoint of the edge to the set containing MST.
    - Therefore, this algorithm’s main loop runs V times. (V = Number of vertices). Then in each vertex it will search for the least weighted edge. That will take O(log(V)) time. (Since this will most likely be extracted from a priority queue) It also updates the key values and parent pointers for adjacent vertices, which takes O(E) time in total. (E = number of edges.)
    - Then the total time will be O (V (E + log(V)))

  - For a connected graph this can be simplified as O (E log(V)) or O (V + E) log V).

- Let’s consider Kruskal's Algorithm.
  - Kruskal’s algorithm starts with a forest of vertices. It will sort all edges of the given graph in increasing order. Then it keeps on adding new edges and nodes in the MST if the newly added edge does not form a cycle. It picks the minimum weighted edge at first at the maximum weighted edge at last.
  - So the sorting takes O (E log E) or O(E log V) time. This can be done using an efficient sorting technique like mergesort or quicksort.
  - Then for each edge it will run disjoint set operation which takes O (log V) time.
  - So, the total is O(E log E) or O(E log V). (Simplified)

- While the graph is not much dense Prims’s algorithm can be useful in terms of the number of edges. However, in dense graphs where the number of edges is close to V^2, Kruskal's algorithm may have a slightly better practical performance due to its simpler edge-based processing and efficient disjoint set data structure usage.