Dijkstra's Minimal Spanning Tree Algorithm Write Up:

Theory:

Dijkstra's algorithm is a well-known algorithm primarily used for finding the shortest path between nodes in a graph. However, it can also be adapted to find the minimal spanning tree (MST) of a graph. An MST is a subgraph that connects all nodes of the original graph with the minimum total edge weight.

The algorithm starts by initializing the distance of all nodes to infinity, except for the starting node, which is set to 0. It maintains a set of visited nodes and a priority queue to store the nodes with their corresponding distances. At each iteration, it selects the node with the minimum distance from the priority queue, marks it as visited, and relaxes all its neighboring nodes.

The relaxation step involves examining the neighboring nodes of the current node. If the distance from the starting node to a neighbor, passing through the current node, is smaller than its current distance, the distance is updated. This process continues until all nodes have been visited or the priority queue becomes empty.

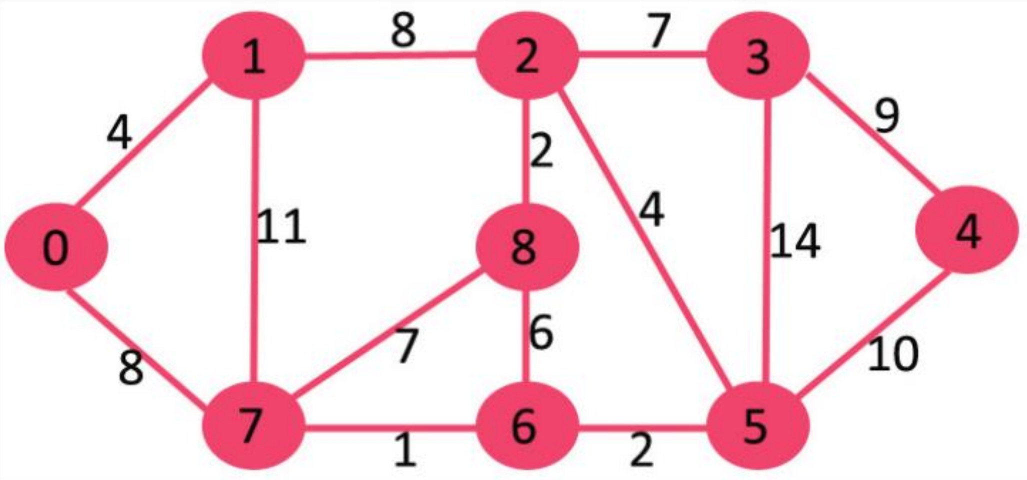
To construct the MST, an additional data structure, such as an array or a heap, is used to store the parent of each node. Initially, all nodes have no parent. During the relaxation step, whenever a node's distance is updated, its parent is set to the current node. This forms a tree-like structure, with the starting node as the root, and the edges between nodes in the MST are the ones connecting a node to its parent.

Once the algorithm finishes visiting all nodes or the priority queue is empty, the MST can be reconstructed using the parent information. It represents the minimum weight connections between nodes in the original graph, ensuring that all nodes are connected and there are no cycles.

Performance Analysis:

The greedy approach involves making locally optimal choices at each step to construct a greedy schedule. The time complexity of the greedy approach is typically O(n log n), where n is the number of greedys. This is because sorting the greedys based on some criteria (e.g., earliest deadline or shortest processing time) often takes O(n log n) time. The space complexity is O(n), as it requires storing the information about each greedy.

Input: src = 0, the graph is shown below.



Output: *0 4 12 19 21 11 9 8 14*

Explanation: *The distance from 0 to 1 = 4.*

The minimum distance from 0 to 2 = 12. 0->1->2

The minimum distance from 0 to 3 = 19. 0->1->2->3

The minimum distance from 0 to 4 = 21. 0->7->6->5->4

The minimum distance from 0 to 5 = 11. 0->7->6->5

The minimum distance from 0 to 6 = 9. 0->7->6

The minimum distance from 0 to 7 = 8. 0->7

The minimum distance from 0 to 8 = 14. 0->1->2->8

Conclusion:

In conclusion, Dijkstra's algorithm is a versatile algorithm that can be used for various graph-related problems, such as finding the shortest path between nodes and constructing a minimal spanning tree. It employs a combination of breadth-first search and greedy techniques to efficiently explore the graph and find the optimal solution.