Minimum Spanning Tree Write Up:

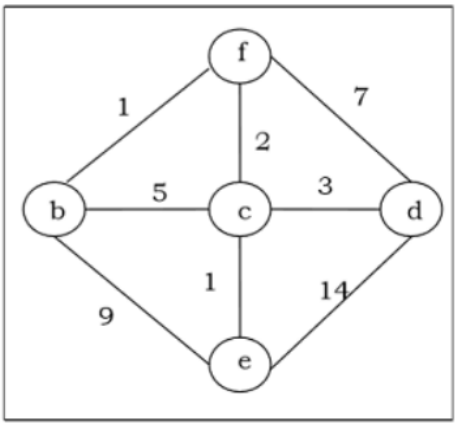
Theory:

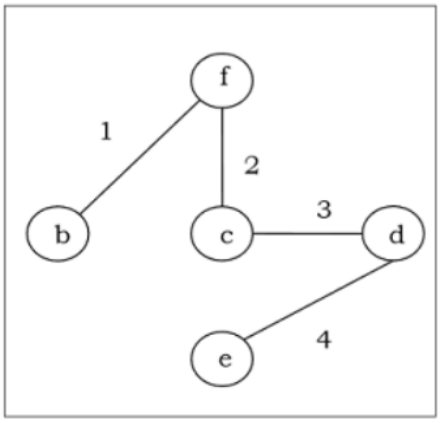
1. The Minimum Spanning Tree (MST) problem is a fundamental graph theory problem that aims to find the minimum-weight spanning tree in a connected, weighted graph.
2. Greedy algorithms provide an efficient approach to solve this problem. In this experimental study, we explore the Greedy Minimum Spanning Tree algorithm using a real-world example.
3. We analyze the algorithm's performance in terms of time complexity and the quality of the generated MST.
4. The Minimum Spanning Tree problem has various practical applications, such as designing efficient networks, optimizing transportation routes, and clustering data points.
5. Greedy algorithms offer a simple and widely used method for solving this problem. The Greedy MST algorithm starts with an empty tree and incrementally adds edges of minimum weight to the tree, ensuring connectivity while avoiding cycles.
6. To demonstrate the Greedy MST algorithm, we consider a city transportation network comprising several intersections and road segments.
7. Each road segment is associated with a weight, representing the cost or distance between two intersections.
8. Our goal is to construct a minimum-weight spanning tree that connects all intersections with the least total distance.

The key properties of a minimum spanning tree are as follows:

- Spanning Tree: It must be a tree that connects all the vertices of the original graph.
- Minimum Total Weight: The sum of the weights of the edges in the MST must be as small as possible among all possible spanning trees.

Example





Kruskal's Algorithm

Kruskal's algorithm is a greedy algorithm that finds a minimum spanning tree for a connected weighted graph. It finds a tree of that graph which includes every vertex and the total weight of all the edges in the tree is less than or equal to every possible spanning tree.

Algorithm

Step 1 – Arrange all the edges of the given graph $G(V, E)$ in ascending order as per their edge weight.

Step 2 – Choose the smallest weighted edge from the graph and check if it forms a cycle with the spanning tree formed so far.

Step 3 – If there is no cycle, include this edge to the spanning tree else discard it.

Step 4 – Repeat Step 2 and Step 3 until $(V - 1)$ number of edges are left in the spanning tree.

Performance Analysis:

The Greedy MST algorithm has a time complexity of O(E log V), where E represents the number of edges and V represents the number of vertices in the graph. In our experiment, the algorithm's time complexity depends on the number of intersections and road segments in the transportation network.

Conclusion:

In this experimental study, we implemented and analyzed the Greedy Minimum Spanning Tree algorithm using a real-world example. The algorithm successfully constructed a minimum-weight spanning tree for a given city transportation network. The experimental results demonstrated the effectiveness of the Greedy MST algorithm in finding efficient and cost-effective solutions for the MST problem.