Prim's Algorithm

Introduction

- **Prim's Algorithm** is a **greedy algorithm** used to find the **Minimum Spanning Tree (MST)** of a **weighted, connected, undirected graph.**
- A **Minimum Spanning Tree (MST)** is a subset of the edges that connects all vertices of the graph with the **minimum possible total edge weight** and **without any cycles.**

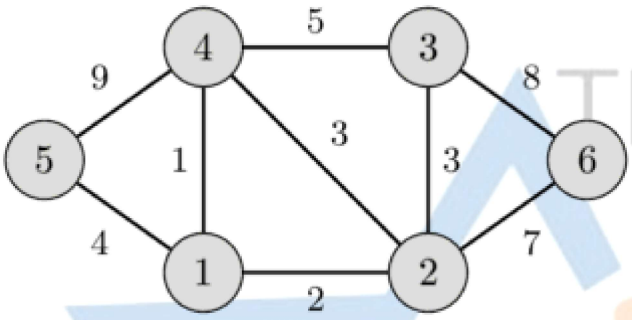
Key Idea

- Start from any vertex.
- At each step, choose the **minimum-weight edge** that connects a vertex in the MST to a vertex **outside** the MST.
- Repeat until all vertices are included in the MST.

Steps of Prim's Algorithm

1. Initialize MST with a starting vertex.
2. Maintain a set of visited vertices.
3. From the visited vertices, pick the edge with the **smallest weight** that leads to an unvisited vertex.
4. Add that edge and vertex to the MST.
5. Repeat until all vertices are included.

Example-



Step 1: Start from vertex 1

Edges from 1 → (1–4 = 1), (1–2 = 2), (1–5 = 4)
Pick the **smallest = 1–4 (1)**

MST so far: {(1–4)}

Step 2: Now vertices {1,4} are inside

Edges → (1–2 = 2), (1–5 = 4)
Pick the **smallest = 1–2 (2)**

MST so far: {(1–4), (1–2)}

Step 3: Now vertices {1,2,4} are inside

Edges → (2–3 = 3), (1–5 = 4), (2–6 = 7)
Pick the **smallest = 2–3 (3)**

MST so far: {(1–4), (1–2), (2–3)}

Step 4: Now vertices {1,2,3,4} are inside

Edges → (1–5 = 4), (2–6 = 7)
Pick the **smallest = 1–5 (4)**

MST so far: {(1–4), (1–2), (2–3), (1–5)}

Step 5: Now vertices {1,2,3,4,5} are inside

Edges → (2–6 = 7)
👉 Only one left = 2–6 (7)

MST so far: {(1–4), (1–2), (2–3), (1–5), (2–6)}

Final MST edges:
(1–4), (1–2), (2–3), (1–5), (2–6)

Total weight = 17

Time Complexity

- Using **Adjacency Matrix**: **O(V^2)**
- Using **Min-Heap / Priority Queue with Adjacency List**: **O(E log V)**
  (where `V = vertices`, `E = edges`)

Applications of Prim's Algorithm

- **Network Design**:
  - Computer networks, telecommunication networks, electrical grids.

- **Cluster Analysis** in Machine Learning.
- **Civil Engineering**: Road and pipeline design with minimum cost.
- **Approximation algorithms** for NP-hard problems (like TSP).