Prim's Algorithm

Theory

It is a greedy algorithm. It starts with an empty spanning tree. The idea is to maintain two sets of vertices:

Contain vertices already included in MST.

Contain vertices not yet included.

At every step, it considers all the edges and picks the minimum weight edge. After picking the edge, it moves the other endpoint of edge to set containing MST.

Prim's algorithm works

It falls under a class of algorithms called greedy algorithms that find the local optimum in the hopes of finding a global optimum.

We start from one vertex and keep adding edges with the lowest weight until we reach our goal.

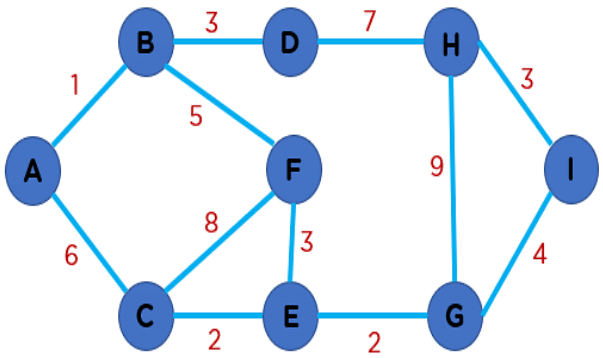
The steps for implementing Prim's algorithm are as follows:

Initialize the minimum spanning tree with a vertex chosen at random.

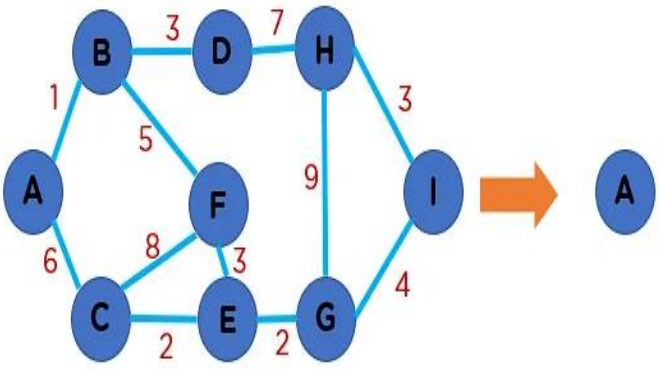
Find all the edges that connect the tree to new vertices, find the minimum and add it to the tree

Keep repeating step 2 until we get a minimum spanning tree

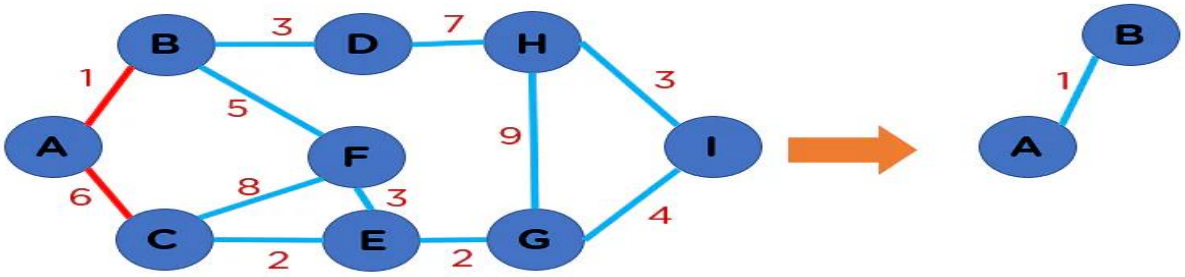
Example



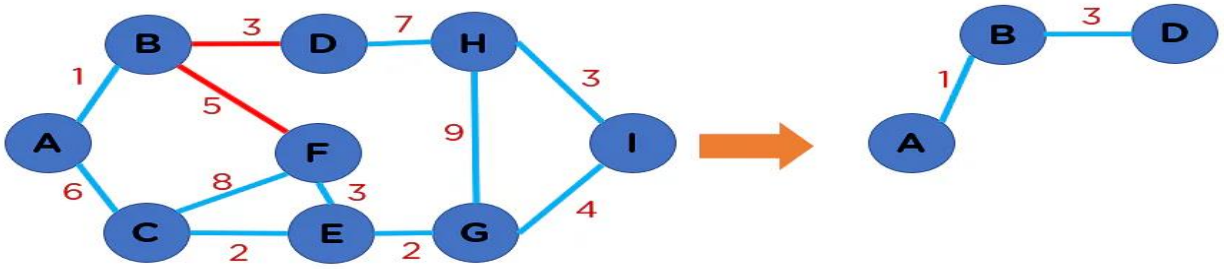
Graph G(V, E)



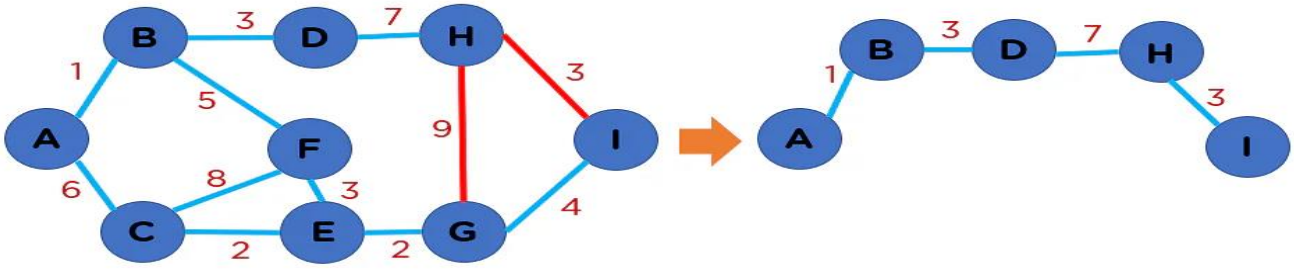
Choose an arbitrary starting vertex.



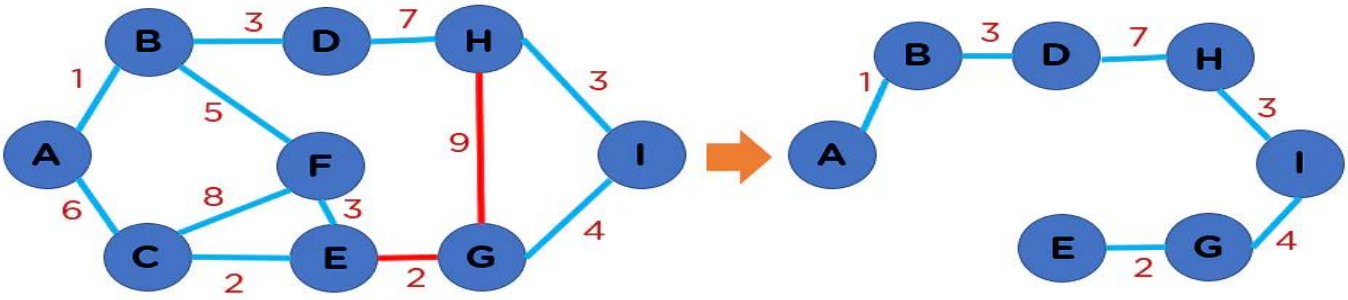
Select the edge with minimal value to include it in MST.



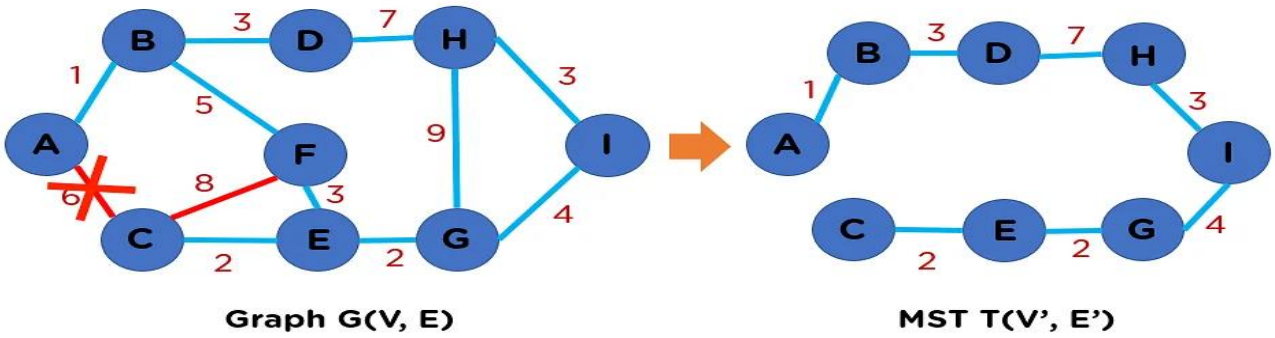
Graph G(V, E) **MST T(V', E')**

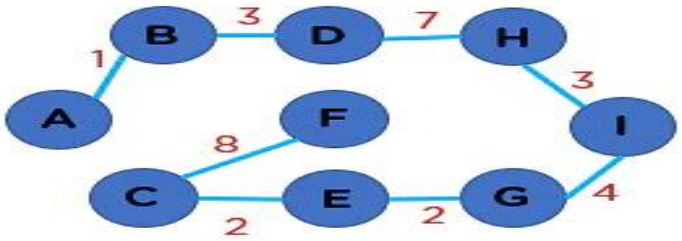


Graph G(V, E) **MST T(V', E')**



Graph G(V, E) **MST T(V', E')**





Condition to check

```
if (graph[u][v] && mstSet[v] == false && graph[u][v] < key[v])
            parent[v] = u, key[v] = graph[u][v];
```

Time Complexity

O(V2), If the input graph is represented using an adjacency list, then the time complexity of Prim's algorithm can be reduced to O(E log V) with the help of a binary heap.

In this implementation, we are always considering the spanning tree to start from the root of the graph

Auxiliary Space: O(V)

Prim's Algorithm Application

- Laying cables of electrical wiring
- In network designed
- To make protocols in network cycles