Md. Mehedi Hassan

Lecturer, Department of CIS, DIU

Prim's and Kruskal Algorithm

Graph

Outlines

- ❑ MST
- ❑ Prim’s Algorithm
- ❑ Kruskal Algorithm

Minimum Spanning Tree (MST)

Spanning tree

A tree that connects all the vertices.

Minimum spanning tree

A tree that connects all the vertices with minimum cost.

Basic feature of MST

- ➢ It is a tree
- ➢ It covers all the vertices V and edge E=V-1
- ➢ Total cost associated with tree edges is the minimum among all possible spanning tree.
- ➢ Not necessarily unique.

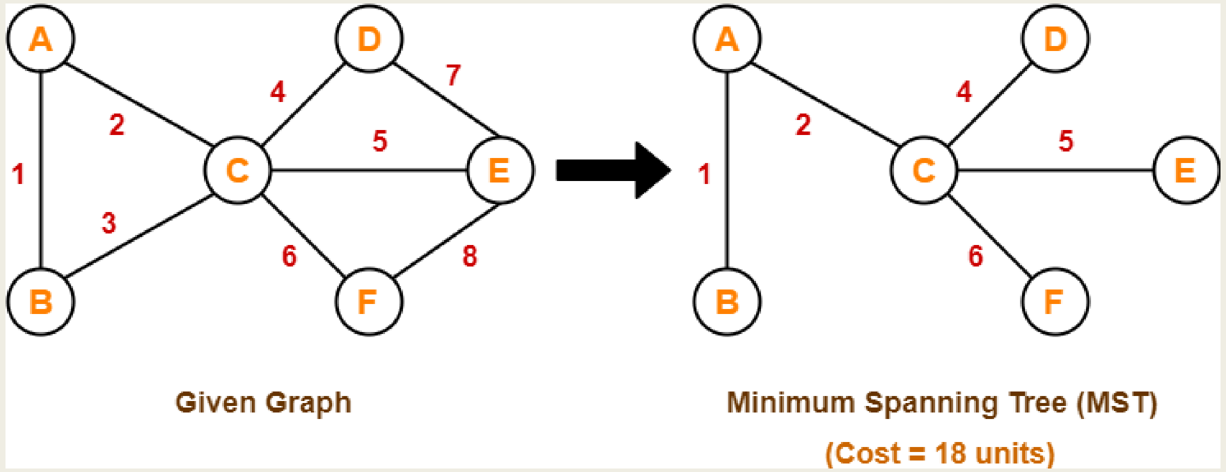
Basic of Prim’s and Kruskal Algorithm

- Prim’s and Kruskal’s Algorithm are the famous greedy algorithms.
- They are used for finding the Minimum Spanning Tree (MST) of a given graph.
- To apply these algorithms, the given graph must be weighted, connected and undirected.

Concept-01:

If all the edge weights are distinct, then both the algorithms are guaranteed to find the same MST.

Example-



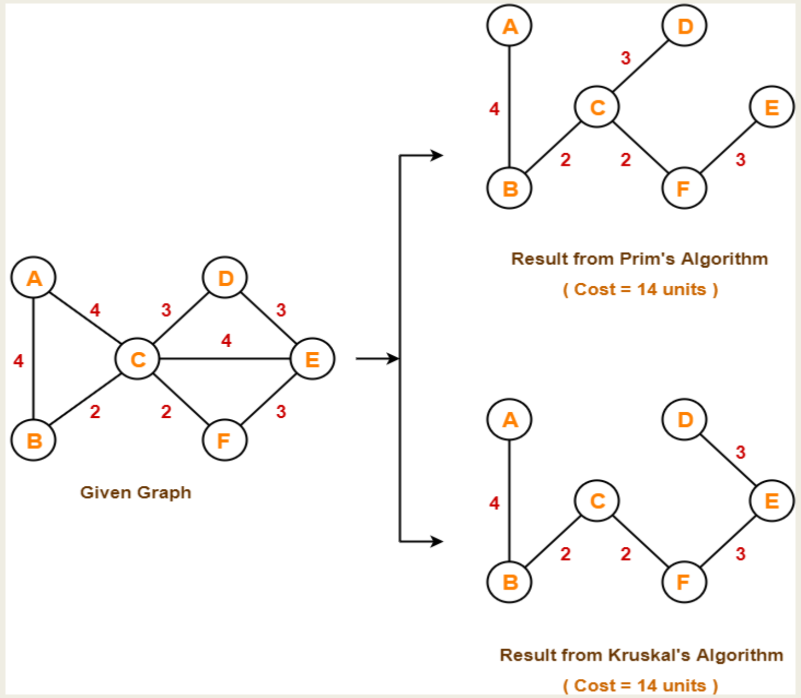
Here, both the algorithms on the above given graph produces the same MST as shown.

Basic of Prim’s and Kruskal Algorithm

Concept-02:

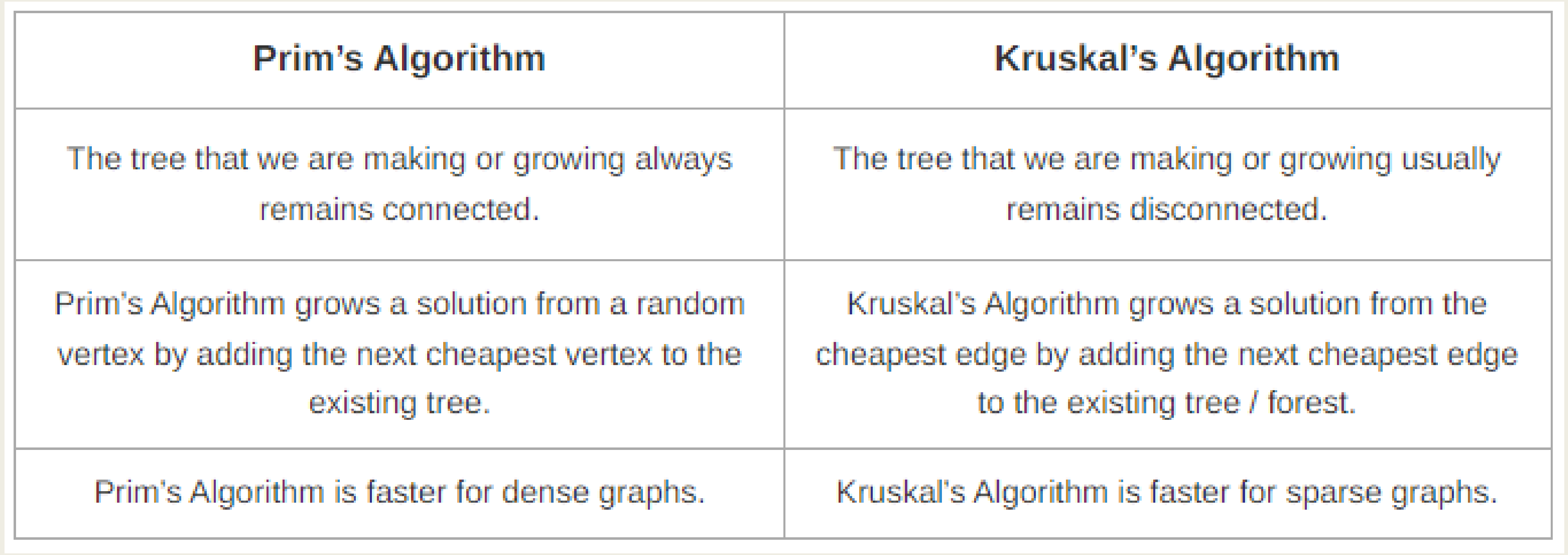
- If all the edge weights are not distinct, then both the algorithms may not always produce the same MST.
- However, cost of both the MST_s would always be same in both the cases.

Example-



Here, both the algorithms on the above given graph produces different MSTs as shown but the cost is same in both the cases.

Difference between Prim’s and Kruskal Algorithm



Prim’s Algorithm	Kruskal’s Algorithm
The tree that we are making or growing always remains connected.	The tree that we are making or growing usually remains disconnected.
Prim’s Algorithm grows a solution from a random vertex by adding the next cheapest vertex to the existing tree.	Kruskal’s Algorithm grows a solution from the cheapest edge by adding the next cheapest edge to the existing tree / forest.
Prim’s Algorithm is faster for dense graphs.	Kruskal’s Algorithm is faster for sparse graphs.

Prim’s Algorithm

Prim's Algorithm is a greedy algorithm that is used to find the minimum spanning tree from a graph. Prim's algorithm finds the subset of edges that includes every vertex of the graph such that the sum of the weights of the edges can be minimized.

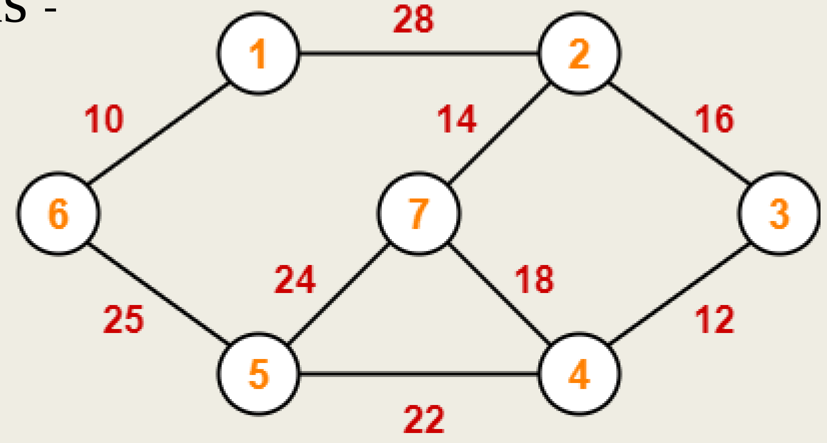
Prim's algorithm starts with the single node and explores all the adjacent nodes with all the connecting edges at every step. The edges with the minimal weights causing no cycles in the graph got selected.

The steps to implement the prim's algorithm are given as follows –

- First, we have to initialize an MST with the randomly chosen vertex.
- Now, we have to find all the edges that connect the tree in the above step with the new vertices. From the edges found, select the minimum edge and add it to the tree.
- Repeat step 2 until the minimum spanning tree is formed.

Prim’s Algorithm: Example

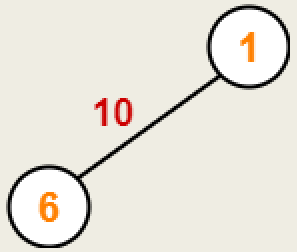
Suppose a weighted graph is -



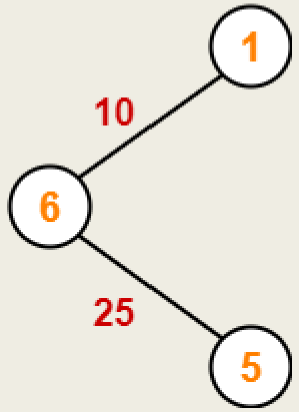
Solution:

Now, let's start constructing the minimum spanning tree.

Step 1:

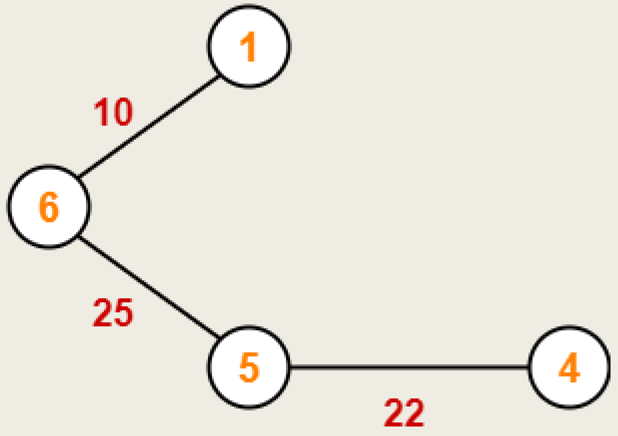


Step 2:

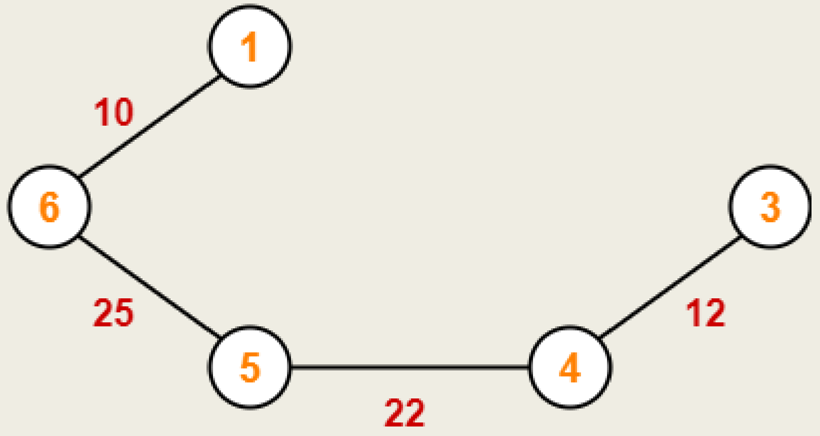


Prim’s Algorithm: Example

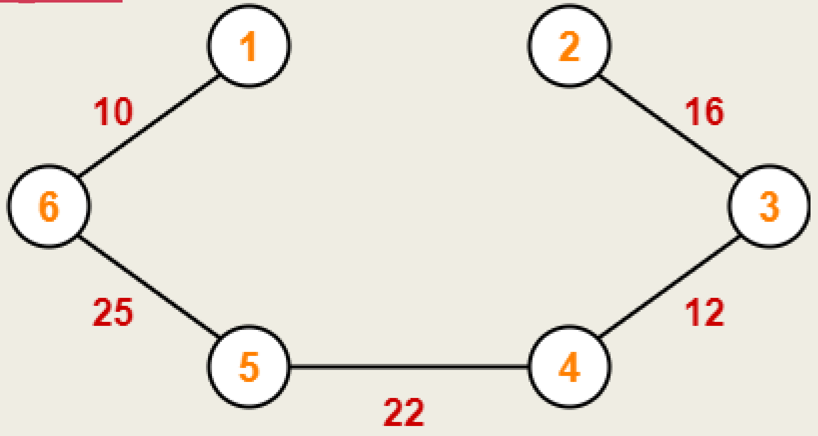
Step 3:



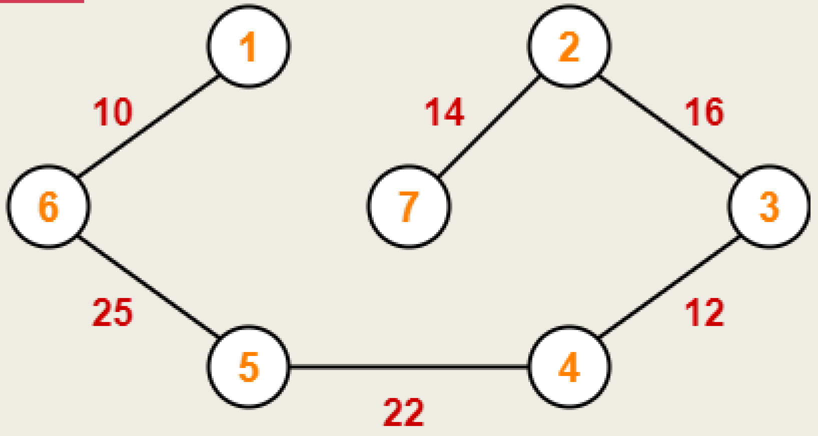
Step 4:



Step 5:



Step 6:



Prim’s Algorithm: Example

Since all the vertices have been included in the MST, so we stop.

Now, Cost of Minimum Spanning Tree = Sum of all edge weights

$= 10 + 25 + 22 + 12 + 16 + 14$

$= 99$ units

Kruskal Algorithm

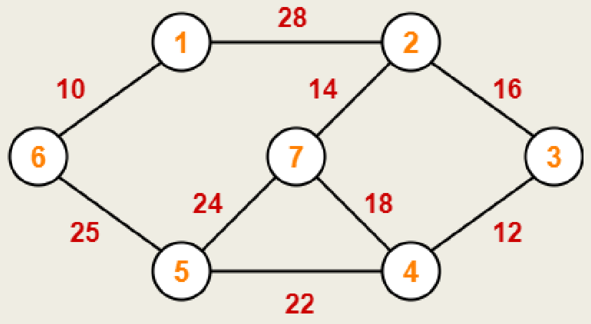
Kruskal's Algorithm is used to find the minimum spanning tree for a connected weighted graph. The main target of the algorithm is to find the subset of edges by using which we can traverse every vertex of the graph. It follows the greedy approach that finds an optimum solution at every stage instead of focusing on a global optimum.

The steps to implement the kruskal algorithm are given as follows –

- First, sort all the edges from low weight to high.
- Now, take the edge with the lowest weight and add it to the spanning tree. If the edge to be added creates a cycle, then reject the edge.
- Continue to add the edges until we reach all vertices, and a minimum spanning tree is created.

Kruskal Algorithm: Example

Suppose a weighted graph is -

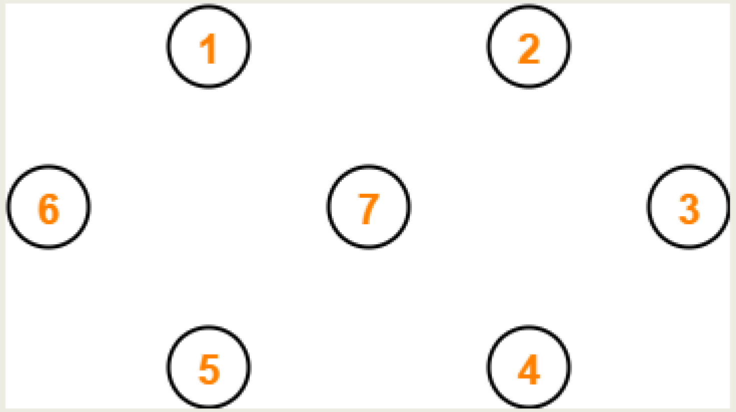


Solution:

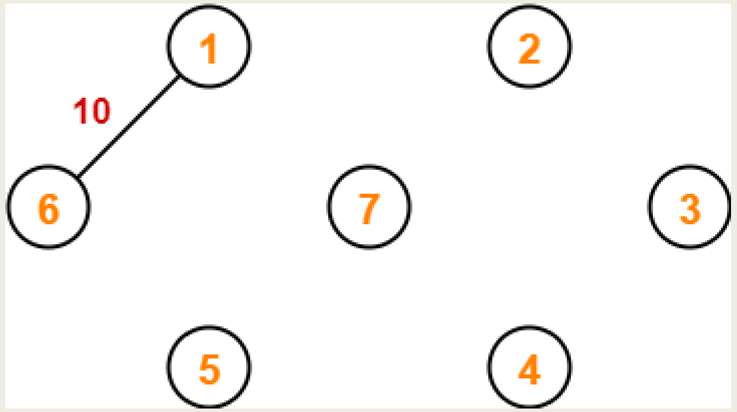
To construct MST using Kruskal's Algorithm,

- Simply draw all the vertices on the paper.
- Connect these vertices using edges with minimum weights such that no cycle gets formed.

Step 1:

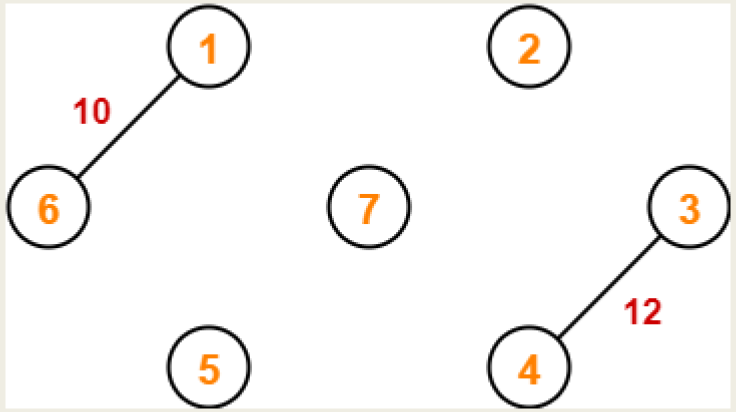


Step 2:

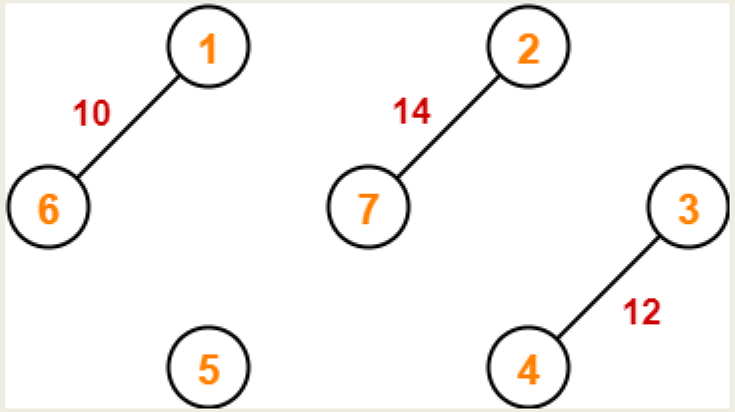


Kruskal Algorithm: Example

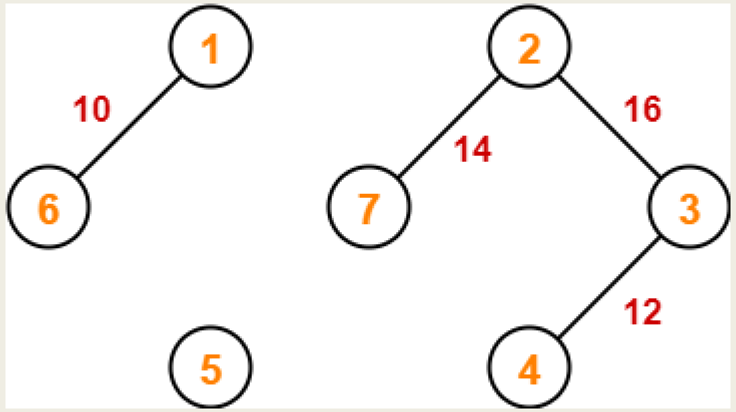
Step 3:



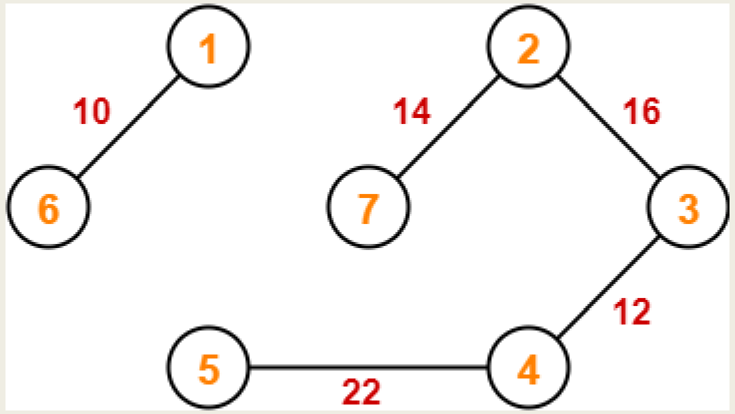
Step 4:



Step 5:

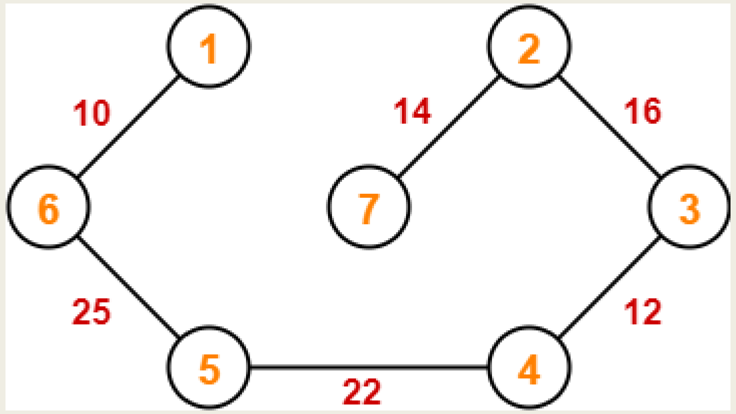


Step 6:



Kruskal Algorithm: Example

Step 7:

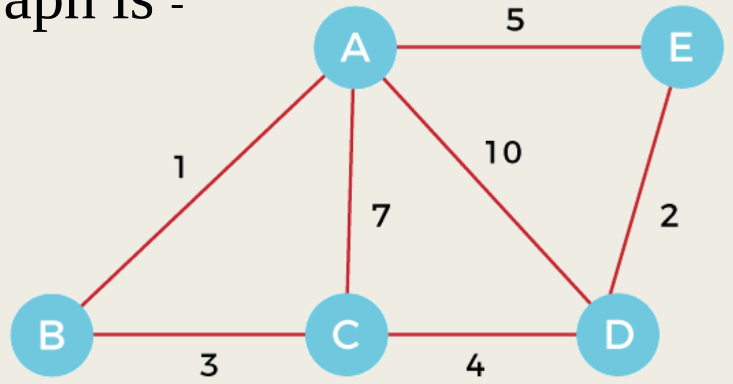


Since all the vertices have been connected / included in the MST, so we stop.

Weight of the MST = Sum of all edge weights
= 10 + 25 + 22 + 12 + 16 + 14
= 99 units

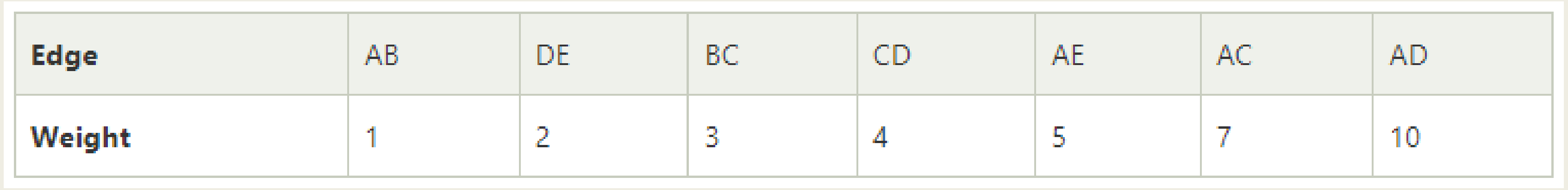
Kruskal Algorithm: Example

Suppose a weighted graph is -



Solution:

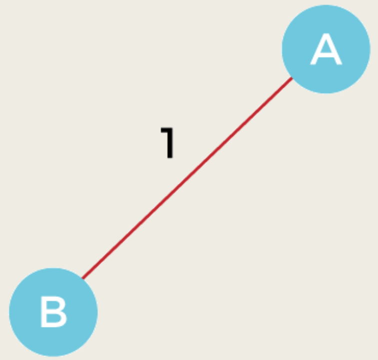
Now, sort the edges given above in the ascending order of their weights.



Edge	AB	DE	BC	CD	AE	AC	AD
Weight	1	2	3	4	5	7	10

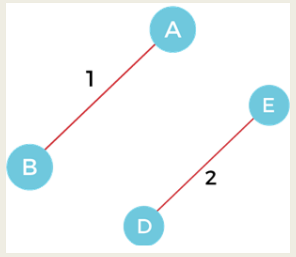
Now, let's start constructing the minimum spanning tree.

Step 1 - First, add the edge AB with weight 1 to the MST.

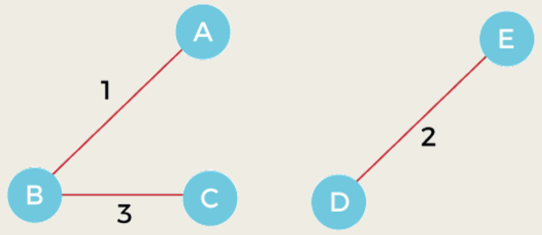


Kruskal Algorithm: Example

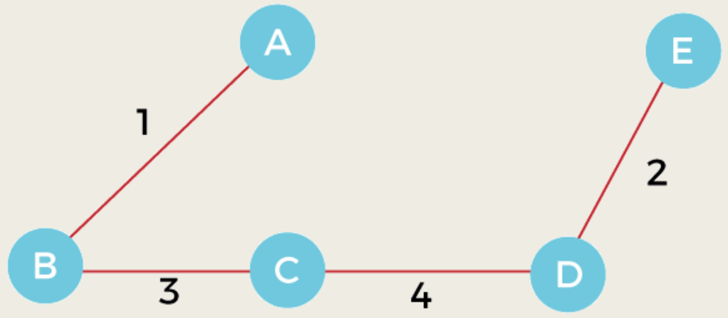
Step 2 - Add the edge DE with weight 2 to the MST as it is not creating the cycle.



Step 3 - Add the edge BC with weight 3 to the MST, as it is not creating any cycle or loop.



Step 4 - Now, pick the edge CD with weight 4 to the MST, as it is not forming the cycle.



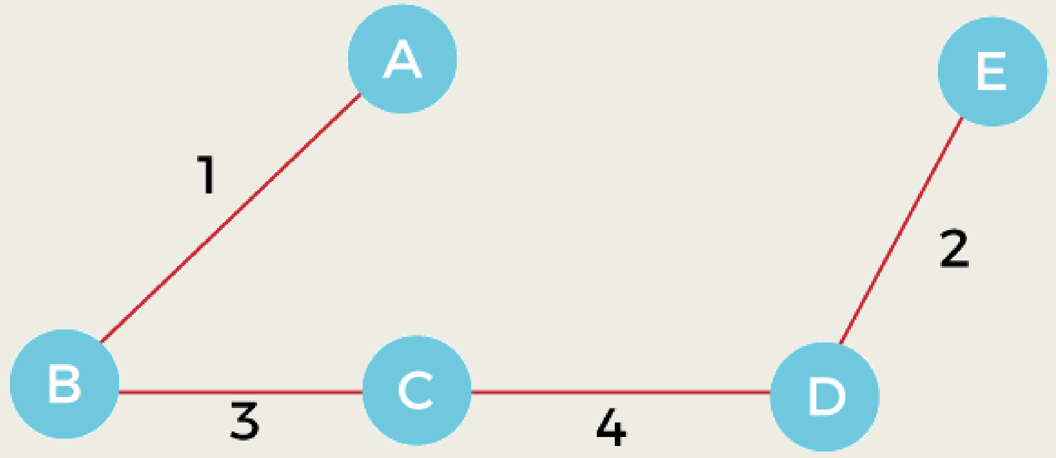
Kruskal Algorithm: Example

Step 5 - After that, pick the edge AE with weight 5. Including this edge will create the cycle, so discard it.

Step 6 - Pick the edge AC with weight 7. Including this edge will create the cycle, so discard it.

Step 7 - Pick the edge AD with weight 10. Including this edge will also create the cycle, so discard it.

So, the final minimum spanning tree obtained from the given weighted graph by using Kruskal's algorithm is -



The cost of the MST is = AB + DE + BC + CD = 1 + 2 + 3 + 4 = 10.

Now, the number of edges in the above tree equals the number of vertices minus 1. So, the algorithm stops here.

Thank You

Any Question ?