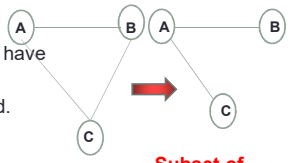
# DATA STRUCTURES

Minimum Spanning Tree & its Algorithm

INSTRUCTOR:
Zainab Malik

## Introduction **Spanning Tree**

- A spanning tree is a subset of Graph G, which has all the vertices covered with minimum possible number of edges.
- If the number of vertices is n,then edges should be n-1.
- A spanning tree does not have cycles.
- It cannot be disconnected.




**Graph** **Subset of Graph**

## Formula

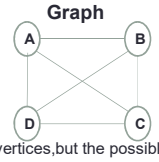
- A complete graph can have maximum $n^{n-2}$ number of spanning trees,

  where n is the number of nodes(Vertices). In the above example, n is

  3, hence $3^{3-2} = 3$ spanning trees are possible.

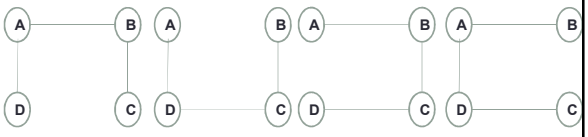
## Example

**Graph**

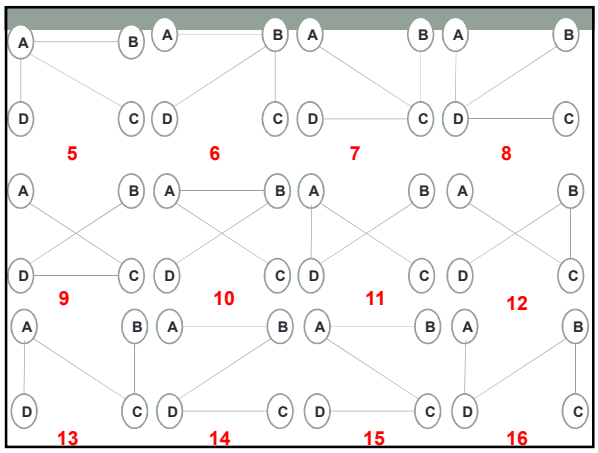



The Graph which have 4 vertices,but the possible Spanning Tree is $n^{n-2} = 4^{4-2} = 16$

and have **edges = n - 1 = 4 - 1 = 3**

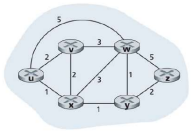


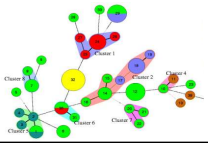

**1** **2** **3** **4**




## Application of Spanning Tree

- Civil Network Planning
- Computer Network Routing Protocol
- Cluster Analysis

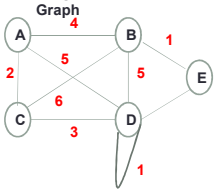




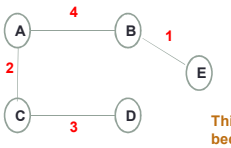
## Minimum Spanning Tree (MST)

- In a weighted graph, a minimum spanning tree is a spanning tree that has minimum weight than all other spanning trees of the same graph.
- For Example:-
- In real-world situations, this weight can be measured as distance, congestion, traffic load or any arbitrary value denoted to the edges.

Weighted Graph




- Here, Number of vertices=**5**.
- Possible spanning trees $n^{n-2} = 5^{5-2} = 125$.
- Number of edges **= n-1 = 4**.




Total minimum weight = 2 + 3 + 4 + 1 = 10

**This is minimum spanning tree because its total weight is minimum for all other graphs.**

## Minimum Spanning Tree-Algorithm

**Two most important Spanning tree Algorithms are :-**

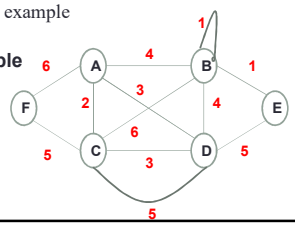
1. Kruskal's Algorithm
2. Prim's Algorithm

## Kruskal's Algorithm

Kruskal's algorithm to find the minimum cost spanning tree uses the greedy approach. A tree connects to another only and only if, it has the least cost among all available options and does not violate MST properties.

To understand Kruskal's algorithm let us consider the following example

**For Example**



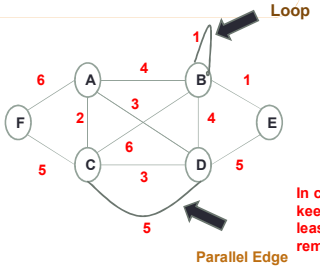



## Kruskal's algorithm

- Remove all loops ( any edge that starts and ends at the same vertex is a loop )
- Remove all parallel edges between two vertices except the one with least weight .
- Choose any arbitrary vertex as root vertex
- Check outgoing edges and select the one with less cost
- And with no cycle .

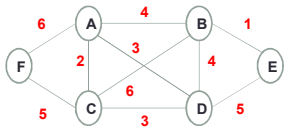
## Steps to construct Kruskal's Algorithm

1 **Remove all loops and Parallel Edges**




**In case of parallel edges, keep the one which has the least cost associated and remove all others.**

**After remove loop and parallel Edge**




**2** **Arrange all edges in their increasing order of weight**

The next step is to create a set of edges and weight, and arrange them in an ascending order of weightage (cost).

| B,E | A,C | C,D | A,D | A,B | B,D | F,C | D,E | F,A | C,B |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 6 |

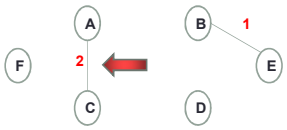
**3** **Add the edge which has the least weightage**

Now we start adding edges to the graph beginning from the one which has the least weight. Throughout, we shall keep checking that the spanning properties remain intact. In case, by adding one edge, the spanning tree property does not hold then we shall consider not to include the edge in the graph.

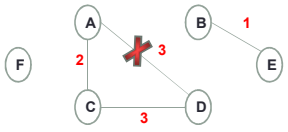
| B,E | A,C | C,D | A,D | A,B | B,D | F,C | D,E | F,A | C,B |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 6 |


| B,E | A,C | C,D | A,D | A,B | B,D | F,C | D,E | F,A | C,B |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 6 |

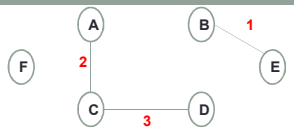
Next cost is 2, and edges isA,C. We add them again –




Next cost is 3, and associated edges are A,D and C,D. We add them again

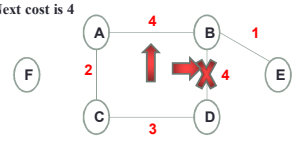


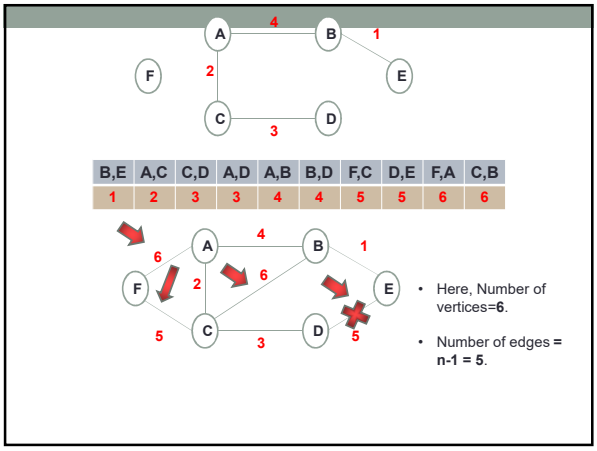

We ignore it beacuse it creates a circuit or Loop.

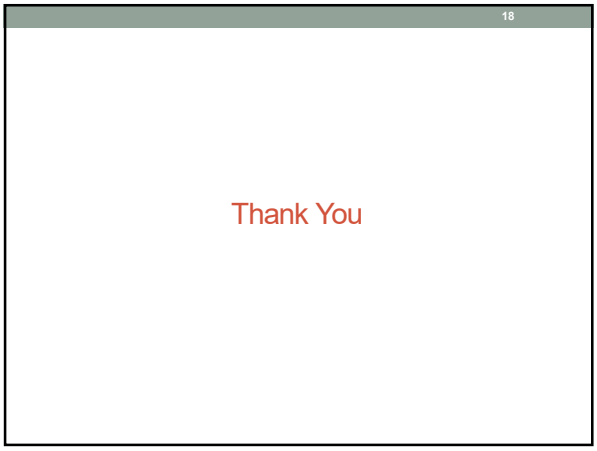



| B,E | A,C | C,D | A,D | A,B | B,D | F,C | D,E | F,A | C,B |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 6 |

Similarly, Next cost is 4




A
B
C
D
E
F
4
1
2
3
B,E A,C C,D A,D A,B B,D F,C D,E F,A C,B
1 2 3 3 4 4 5 5 6 6
6
5
- Here, Number of vertices=**6**.
- Number of edges = **n-1 = 5**.


18
Thank You