Kruskal's Algorithm

The Minimum Spanning Tree Problem (MST):

We are given a Graph G = (V,E). with edges with weights w_e for all e belongs to E. We need to find a tree that connects all the vertices of the graph such that the set of all its edges is E' and the sum of weights of all edges in E' is minimum possible.

One of the famous algorithms used to find an MST is the Kruskal's Algorithm.

Kruskal's Algorithm :

The Kruskal Algorithm has a very simple strategy, It states that we can start from any vertex initially and we need to pick the smallest edge at every stage such that on addition of the edge the current tree does not contain a cycle. By following this simple strategy we are always guaranteed that we will end up with a tree with least sum of weights or the MST.

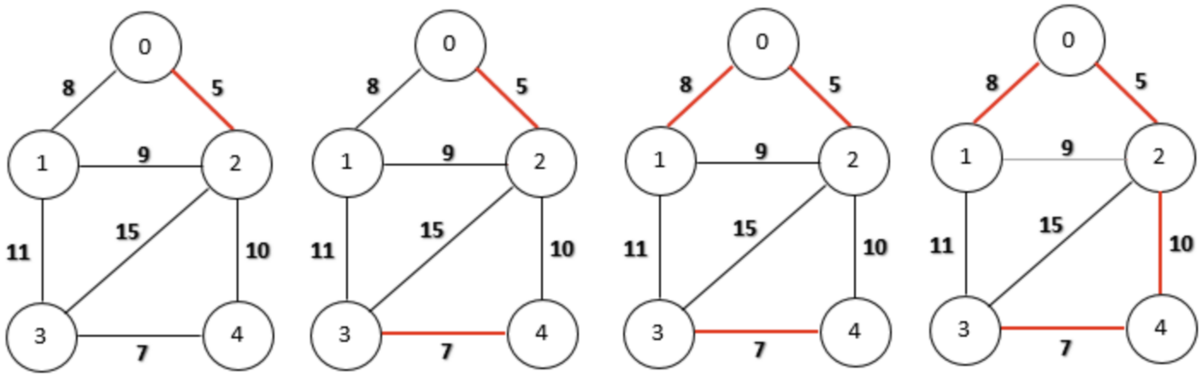
To prove the Kruskal's Algorithm we use the cut property.

Cut Property :

The statement for the Cut property goes as follows :

" Let a set of edges X be a part of an MST of G(V,E). Take any subset of nodes S for which X does not cross between S and V − S, and let e be the lightest edge across this partition. Then $X \cup e$ is part of some MST. "

Visualisation is as follows :



This algorithm can be easily extended to find Maximum spanning trees as well.

Pseudo Code :

```
KRUSKAL(G):
A = {}                   // NULL set
For each vertex v ∈ G.V:
    makeset(v)
For each edge (u, v) ∈ G.E ordered by increasing order by weight(u, v):
    if findset(u) ≠ findset(v):
    A = A ∪ {(u, v)}
    Union(u, v)
return A
```