Minimum Spanning Tree

Outline

- What is Minimum Spanning Tree(MST)
- Applications
- Algorithm
  - Kruskal’s Minimum Spanning Tree
    - Disjoint Set Data Structure
    - Union-Find Algorithm
  - Prim’s Minimum Spanning Tree

What is
Minimum Spanning Tree

What is Minimum Spanning Tree

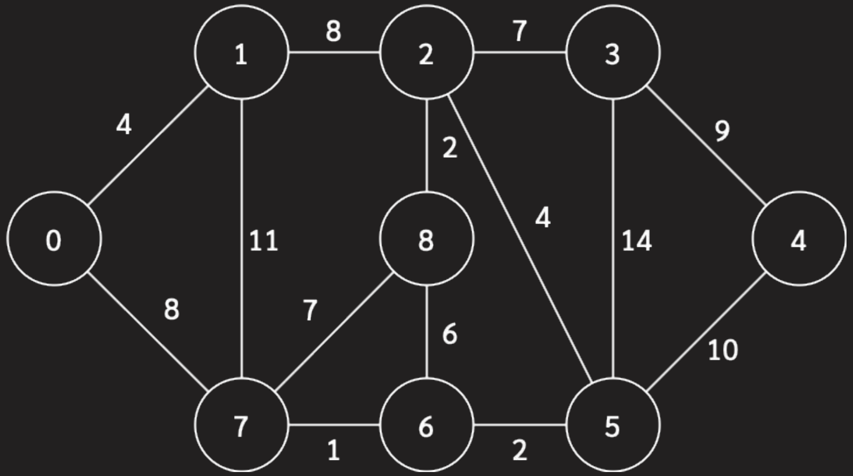
Given:

- Connected graph G with positive edge weights
- V is a number of vertices

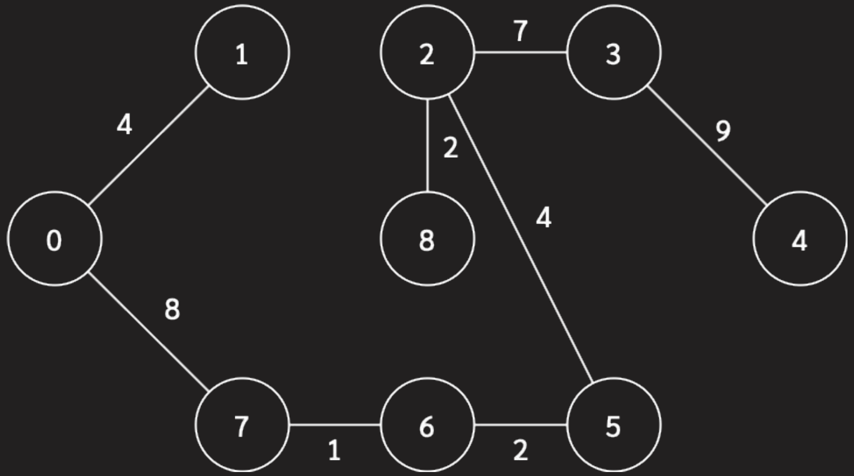
Minimum Spanning Tree is a set of (V-1) edges the connect all of vertices without loop/cycle with minimum total weight.

Example

Given:



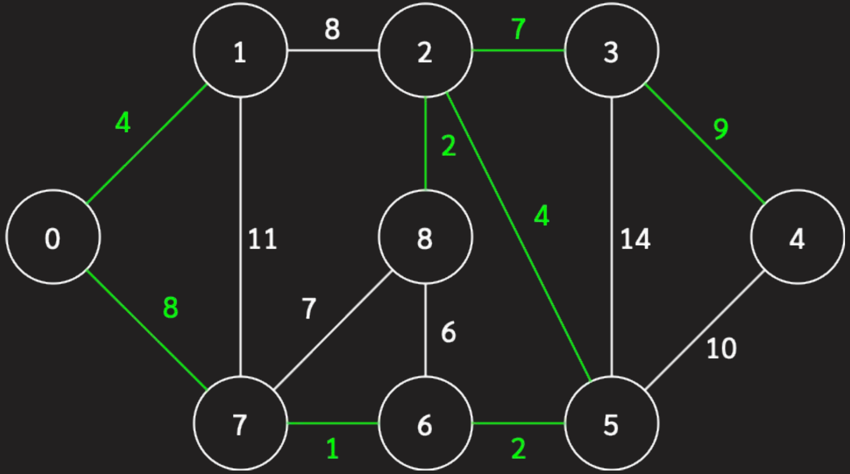
Output: With sum = 37



Applications

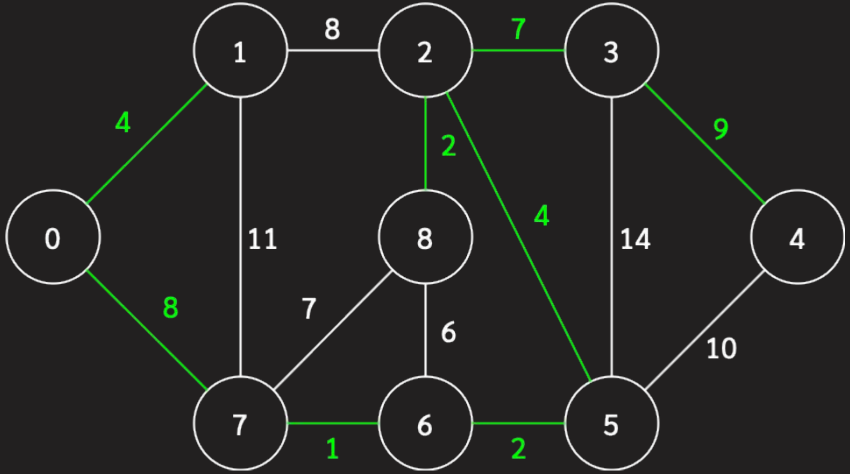
Applications

- Network Design
  - Telephone
  - Electric
  - Road
- Approximation Algorithm for NP-hard Problems
- Cluster Analysis



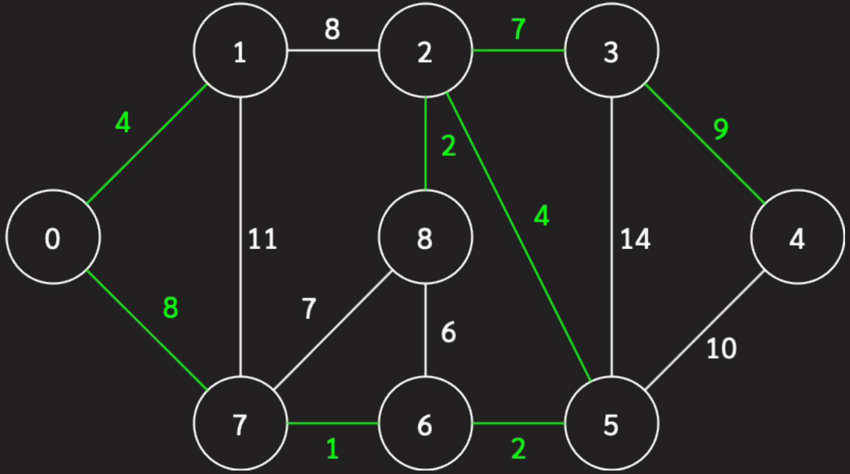
Applications

- **Network Design**
  - Telephone
  - Electric
  - Road
- Approximation Algorithm for NP-hard Problems
- Cluster Analysis



Applications

- Network Design
  - Telephone
  - Electric
  - Road
- Approximation Algorithm for NP-hard Problems



Algorithms

Kruskal’s MST

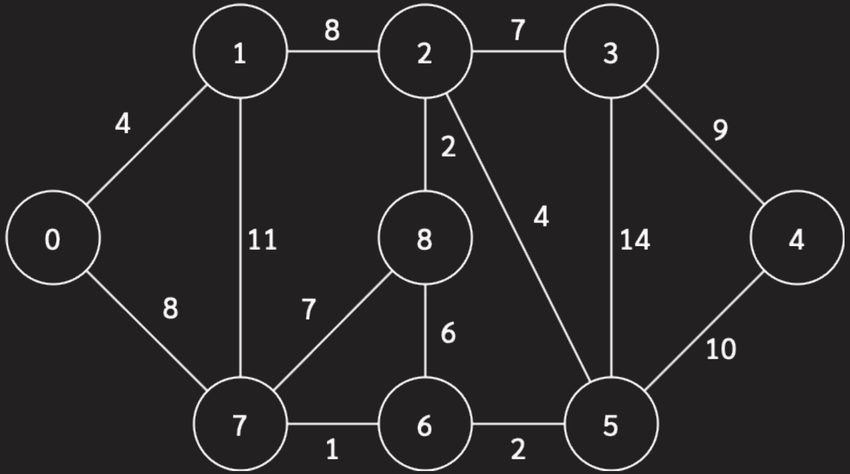
Kruskal’s MST

- We try to insert the least weight edge which not from the cycle edge by edge until the graph has V-1 edges

Kruskal’s MST

Algorithm

1. Sort all the edges in non-decreasing order of their weight.
2. Pick the smallest edge. Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it.
3. Repeat step#2 until there are (V-1) edges in the spanning tree.


0
1
2
3
4
5
6
7
8
4
8
7
8
11
2
4
14
9
7
6
10
1
2

How to check-loop

- We need a subtle data structure call Disjoint Set and algorithm call Union-Find algorithm
    - Disjoint Set keeps track a set of elements divided into a number of disjoint subsets
    - Union-Find algorithm composed of two operation
        - Find: Determine which subset elements is in
        - Union: Join two subsets into a one

Disjoint Set

- Disjoint set is an array to store subset which each element is a member
- Index represents node number
- Value shows which subset is node belongs to
- Initial values are -1

Node	0	1	2	3	4	5	6	7
Belong to subset	-1	-1	-1	-1	-1	-1	-1	-1

How does Disjoint Set check loop?

- For each edge, if both the vertices are in the same subset, a cycle is found

Union-Find Algorithm

We define Disjoint set call *parent* to track the subset of each element

find(parent, node) #find the parent of each node

```
a) If node doesn't has parent, it is a parent
b) else find its parent's parent
```

Union-Find Algorithm

Union(parent, node_a, node_b) #merge two subset to one

```
1) find parent of node_a
2) find parent of node_b
3) if node_a and node_b have different parents, set parent of node_b is the parent of
node_a or vice versa
```

Now we can check loop by

is_loop(graph G)

```
1) create disjoint set call parent with size of V
2) for each edge(u,v)
    a) find parent of u and v, if u, v have the same parent, this graph has a loop and
return 1
    b) union subset of u and v
3) return 0
```

Prim’s MST

Prim’s MST

- Choose one edge as a starting edge
- Choose the least weight adjacent which is not included in MST until we reach all vertices.

Prim’s MST

1) Create a set mstSet that keeps track of vertices already included in MST.
2) Assign a key value to all vertices in the input graph. Initialize all key values as INFINITE. Assign key value as 0 for the first vertex so that it is picked first.
3) While mstSet doesn’t include all vertices
    a) Pick a vertex u which is not there in mstSet and has minimum key value.
    b) Include u to mstSet.
    c) Update key value of all adjacent vertices of u. To update the key values, iterate through all adjacent vertices. For every adjacent vertex v, if weight of edge u-v is less than the previous key value of v, update the key value as weight of u-v