# Design and Analysis of Algorithms

# L19: Prim's Algorithm

## Minimum Cost Spanning Tree

Dr. Ram P Rustagi
Sem IV (2020-Even)
Dept of CSE, KSIT
rprustagi@ksit.edu.in

# Resources

- Text book 1: Sec 9.1-5.4 - Levitin
- R1: Introduction to Algorithms
  - Cormen et al.
- MIT Open Course Ware
  – https://ocw.mit.edu/courses/civil-and-environmental-engineering/1-204-computer-algorithms-in-systems-engineering-spring-2010/lecture-notes/MIT1_204S10_lec11.pdf

# Spanning Tree

- Consider N number of villages in a district
- Government would like to ensure that these villages are connected by road
  - Reachable from each other, may be via other villages
- The cost of laying road from one village to other villages is known
- Govt would like to incur minimum cost
- Which roads government should lay down
  - How many roads needs to be laid down.
- Answer: **M**inimum **C**ost **S**panning **T**ree
- Q: Provide other examples:

# Application of Spanning Trees

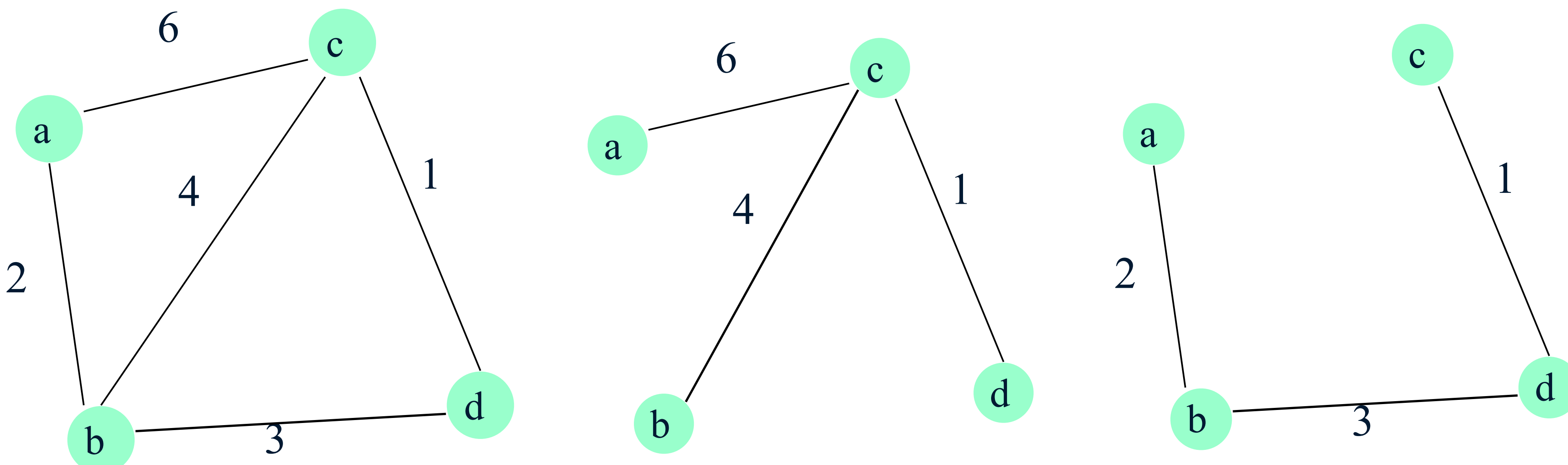
- Laying of utility lines
  – Water,
  – Electrical,
  – Gas,
  – Cable TV lines
  – …
- Road distribution network
- Building floor/room corridors with single entry/exit point.

# Spanning Tree

- Graph: $G=\{V,E\}$
  - A set of nodes $V$
  - A set of edges $E=(u,v)$ connecting node $u$ to node $v$.
- Connected Graph:
  - Each node is reachable from any other node via some path.
  - There may exist multiple paths, (have cycles)
- Spanning tree:
  - A subgraph $T$ of $G$ i.e. $T \subseteq G$ such that
    - It contains all the vertices $V$ of $G$ i.e. if $v \in G \Rightarrow v \in T$
    - Between any two nodes $u$ and $v$, $\exists$ only one path
    - i.e. $T$ is acyclic

# Minimum Spanning Tree

- A minimum spanning tree of weighted connected graph `G` is a spanning tree `T` with minimum total weight.
- Examples:




- Q: Are other spanning trees possible?
- Q: What happens when all edges have same weight?

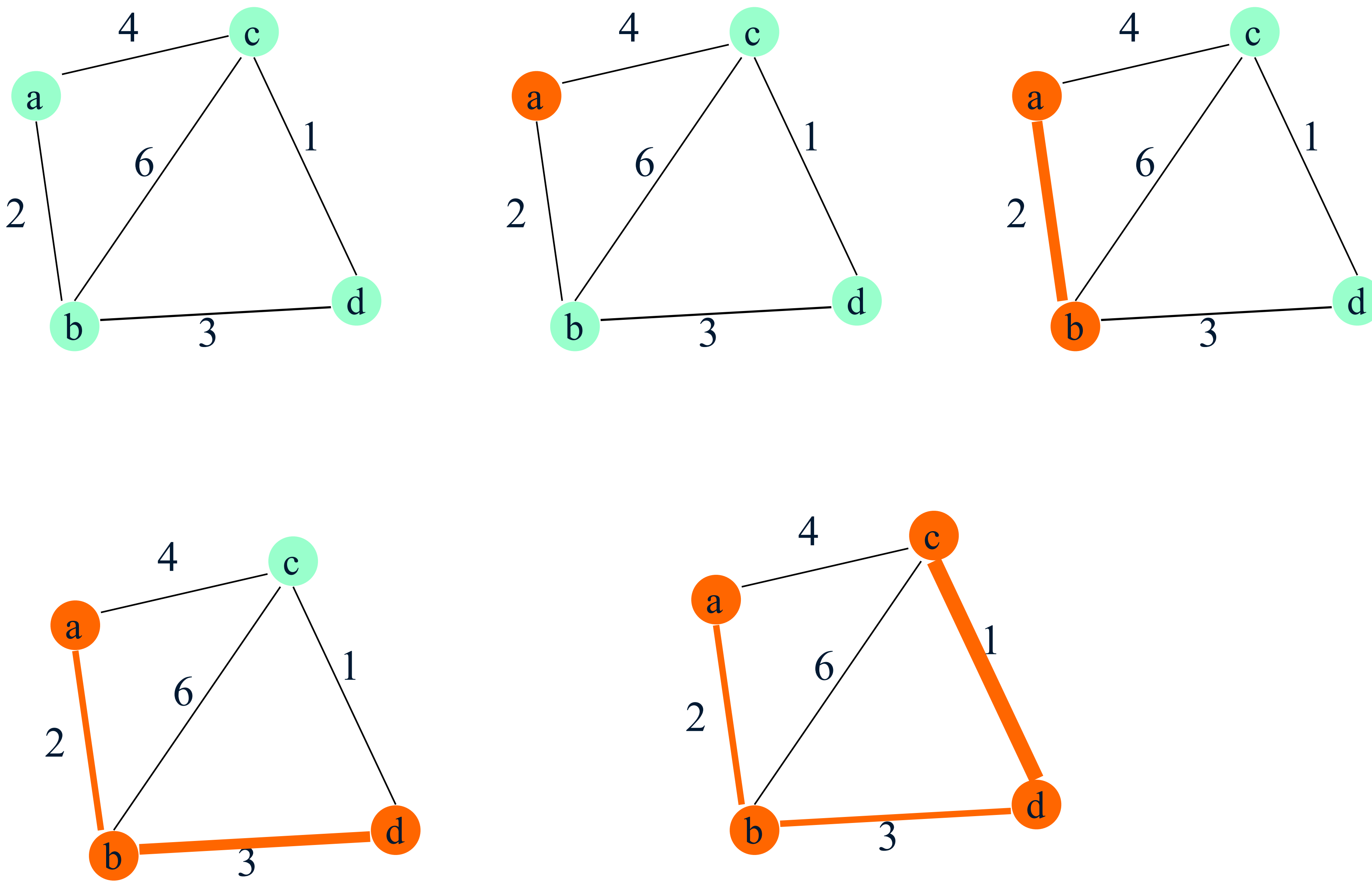
# Spanning Trees

- Property 1:
  - Removing a cycle edge can't disconnect a graph
- Property 2:
  - A tree of `n`-nodes has `n-1` edges
- Property 3:
  - Any connected, undirected graph G=(V, E) with number of edges `|E|=|V-1|` is a tree
- Property 4:
  - An undirected graph is a tree if and only if there is a unique path between any pair of nodes.
  - Corrolary:
    - If a graph has a path between any two nodes, then it is connected. If these paths are unique, then the graph is also acyclic.

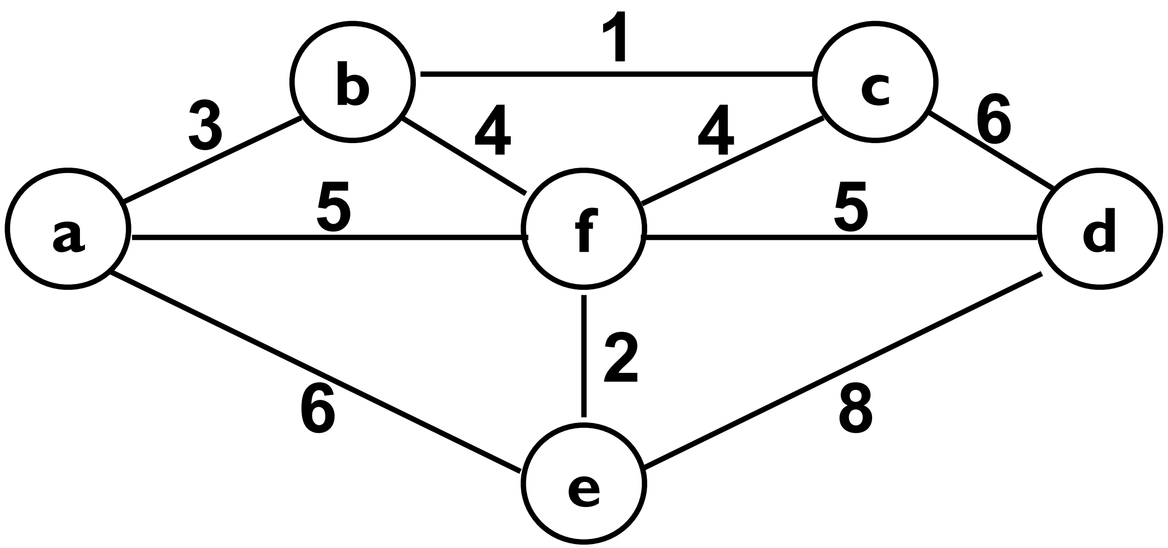
# Prim's MST Algorithm

- Approach:
  - Start with tree $T_1$ consisting of one (any) vertex, and
    - *Grow* tree one vertex at a time to produce MST
    - Through a series of expanding subtrees $T_1, T_2, \ldots, T_n$
- Greedy Appraoch:
  - On each iteration, construct $T_{i+1}$ from $T_i$
    - Add a vertex not in $T_i$ which is
      - Closest to those already in $T_i$
      - This is a **greedy** step!
- Stop when all vertices are included.

# Example 1: Prim's MST

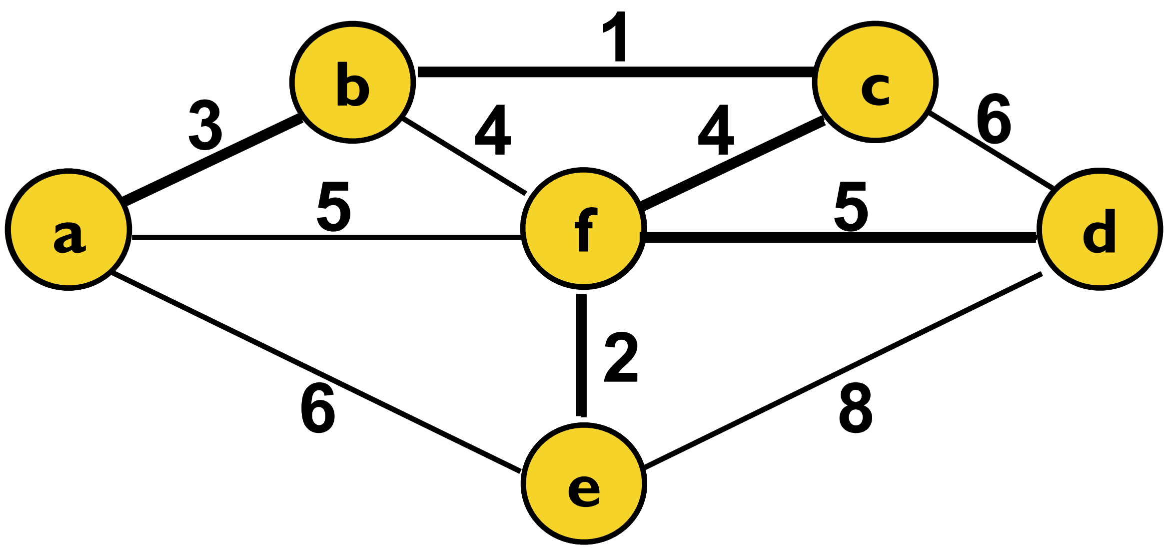


# Example 2: Prim's MST




- Q: Construct an MST starting from vertex `a`

# Example 2: Prim's MST




w(a):0,w(b):∞,w(c):∞,w(d)=∞,w(e)=∞,w(f)=∞

w(**a**):0,w(b):3,w(c):∞,w(d)=∞,w(e)=6,w(f)=5

w(a):0,w(**b**):3,w(c):1,w(d)=∞,w(e)=6,w(f)=4

w(a):0,w(b):3,w(**c**):1,w(d)=6,w(e)=6,w(f)=4

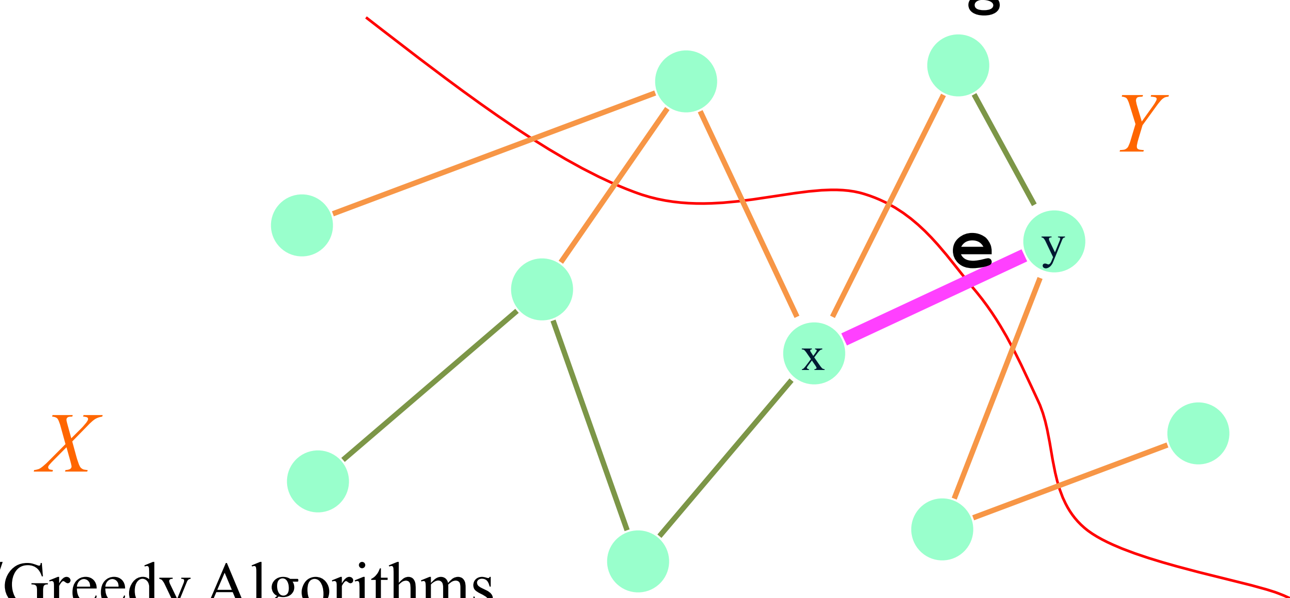
w(a):0,w(b):3,w(c):1,w(d)=5,w(e)=2,w(**f**)=4

w(a):0,w(b):3,w(c):1,w(d)=5,w(**e**)=2,w(f)=4

w(a):0,w(b):3,w(c):1,w(**d**)=5,w(e)=2,w(f)=4

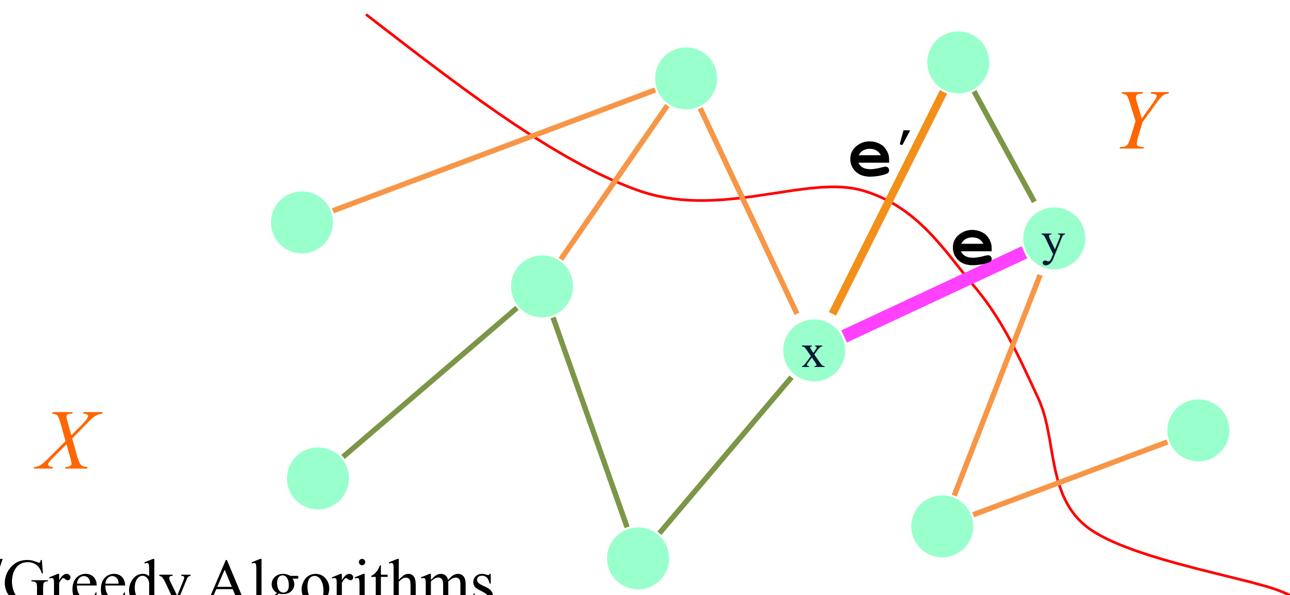
# Prim's Algo: Proof by Induction

- Claim: Let $G = (V,E)$ be a weighted graph and $(X,Y)$ be a partition of V (called a *cut*).
- Cut property:
  - Suppose $e = (x,y)$ is an edge of $E$ across the cut, where
    - $x$ is in $X$, and , $y$ is in $Y$, and
    - $e$ has the minimum weight among all such crossing edges (called a light edge).
  - Then there is an MST containing $e$.



# Proof: Cut Property

- Consider that `e` is not in MST.
- There must be an edge `e′` connecting X to Y.
- Let `T′` be the MST with `e′`
- Add `e` to `T’`, which will make a cycle
- Break the cycle by removing `e′` and let `T` is new tree
  - `weight(T)=weight(T′)-w(e′)+w(e)`
- Since e is lightest between X and Y, `w(e)≤w(e′)`
- Thus, `weight(T)≤weight(T′)`, so `T` must be an MST



# Prim's Algo

- Needs priority queue for implementation

Algo: $\mathtt{Prim(G)}$

// i/p: A weighted connected graph $G=(V,E)$

// o/p: $E_T$, the set of edges composing an MST of $G$

$V_T \leftarrow \{v_0\}$ # initialize with any vertex

$E_T \leftarrow \emptyset$

for $i=1$ to $|V|-1$ do

  Find a min weight edge $e^*=(v^*,u^*)$ among all edges $(v,u)$ such that $v \in V_T$ and $u \in |V|-V_T$

  $V_T \leftarrow V_T \cup \{v^*\}$

  $E_T \leftarrow E_T \cup \{e^*\}$

return $E_T$

# Prim's Algo: Efficiency

- Efficiency depends upon implementation
- Maintain $V-V_T$ in priority queue
- Initially, assign a weight(value) of $\infty$ to each vertex
- Weight of each edge is known (given graph G)
- Using **Adjacency weight matrix**
  - If priority queue is maintained in an unordered array
    - Vertex can be accessed by index in the array
  - Picking min vertex `u` takes `|V|` time.
    - Requires linear search in array
  - For each edge `(u,w)`, update the weight of w
    - `weight(w)=min(weight(w),weight(u,w))`
  - Total time: $O(|V|^2+|E|)=O(|V|^2)$

# Prim's Algo: Efficiency

- Efficiency depends upon implementation
- Maintain $V-V_T$ in priority queue
- Initially, assign a weight(value) of ∞ to each vertex
- Weight of each edge is known (given graph G)
- Using **Adjacency weight List**
  - Maintain priority queue in `BinSearch` Tree
    - Height of the tree is `lg|V|`
  - Find vertex `u` with min weight is `O(1)` time
  - For each edge `(u,w)`, update the weight of w
    - `weight(w)=min(weight(w),weight(u,w))`
    - TIme taken to adjust `BinSearch` Tree is `O(lg|V|)`
  - Total time: `O(E*lg|V|)`

# Summary

- Minimum Spanning Tree
- Prim's algorithm
- Time efficiency
  - Depends upon implementation