UNIT 4: Greedy Technique

Greedy Technique:

Prim's algorithm

Greedy Technique

- constructs a solution through a sequence of steps, each expanding a partially constructed solution obtained so far, until a complete solution to the problem is reached.
- makes locally optimal choice at each stage
- typically fails to produce an optimal solution
- best known as ‘**short sighted**’, and ‘**non-recoverable**’

Greedy algorithm properties

On each step the choice made by the algorithm must be

- **feasible**: it has to satisfy the problem's constraints
- **locally optimal**: it has to be the best local choice among all feasible choices available on that step
- **irrevocable**: once choice is made, it cannot be changed on subsequent steps of the algorithm

Prim’s algorithm to find Minimum Spanning Tree (MST)

Spanning tree

- A spanning tree is a sub-graph of an undirected connected graph, which includes all the vertices of the graph with a minimum possible number of edges
- A graph may have several spanning trees
- A graph that is not connected will not contain a spanning tree
- A spanning tree of a connected graph G can also be defined as a maximal set of edges of G that contains no cycle, or as a minimal set of edges that connect all vertices.

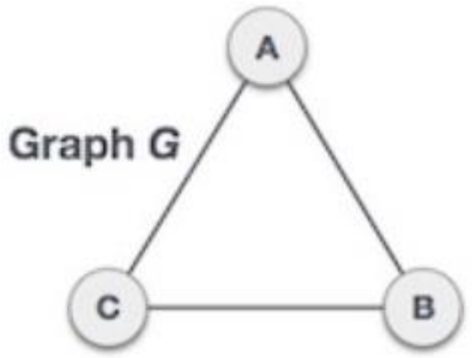
- Spanning tree has n-1 edges, where n is the number of nodes.
- From a complete graph, by removing maximum

  (e – n + 1) edges,

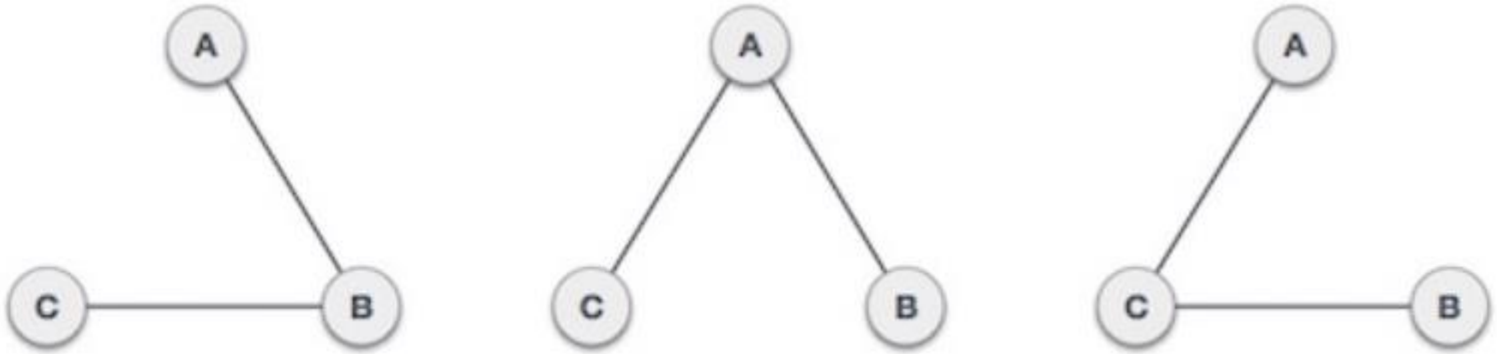
  we can construct a spanning tree.
- The total number of spanning trees with n vertices that can be created from a complete graph is equal to **$n^{(n-2)}$**

  Example:

  if n = 4, then the maximum number of possible spanning trees is equal to 4^{4-2} = 16. Thus, 16 spanning trees can be formed from a complete graph with 4 vertices.



Spanning trees



NOTE:

- A **directed graph** contains a **directed spanning tree** rooted at r if and only if all vertices in G are reachable from r.

- For directed graphs, the minimum spanning tree problem is called the **Arborescence problem** and can be solved in quadratic time using the **Chu–Liu/Edmonds algorithm**. Such a tree can be found with algorithms such as Prim's or Kruskal's after multiplying the edge weights by -1 and solving the MST problem on the new graph

Spanning tree applications:

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

(routing protocols: Spanning Tree Protocol, Open Shortest Path First, Link-state routing protocol, Augmented tree-based routing)

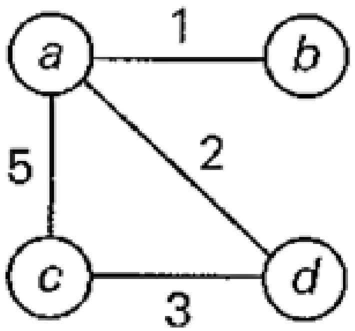
Minimum Spanning Tree (MST)

- The cost of the spanning tree is the sum of the weights of all the edges in the tree
- 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.
- weight can be distance, congestion, traffic load or any arbitrary value denoted to the edges

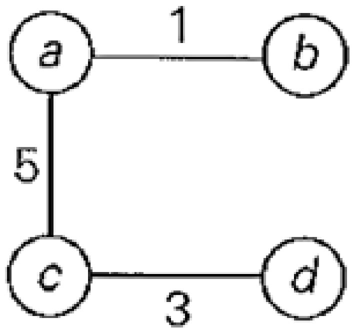
Minimum Spanning Tree (MST) applications:

- To find paths in the map
- To design networks like telecommunication networks, water supply networks, and electrical grids.
- It is also used in algorithms approximating the travelling salesman problem, multi-terminal minimum cut problem and minimum-cost weighted perfect matching.
- Other practical applications are: Cluster Analysis, Handwriting recognition, Image segmentation

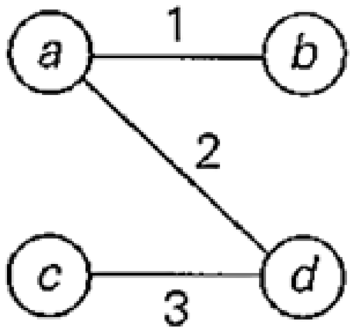
Graph and its spanning trees



graph

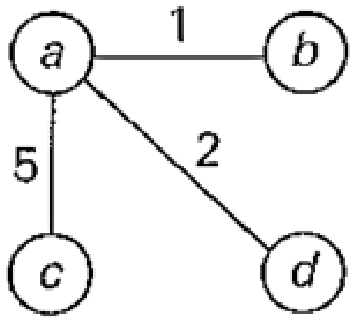


$w(T_2) = 9$



$w(T_1) = 6$

T_1 is the minimum spanning tree



$w(T_3) = 8$

Exhaustive-search approach: limitations

- number of spanning trees grows exponentially with the graph size
- generating all spanning trees for a given graph is not easy

Solution:

- **Prim's algorithm**
- **Kruskal's algorithm**

Prim’s algorithm

- Builds MST starting from any node
- Constructs MST through a sequence of expanding subtrees in a greedy manner.
- Uses adjacency matrix / binary heap / Fibonacci heap
- Run faster on dense graph
- Always stays as a tree

- **Fringe** vertex: Vertex that is not in the partially constructed MST but is adjacent to at least one tree vertex. These vertices are the candidates from which the next tree vertex is selected.
- **Unseen** vertex: are all the other vertices of the graph

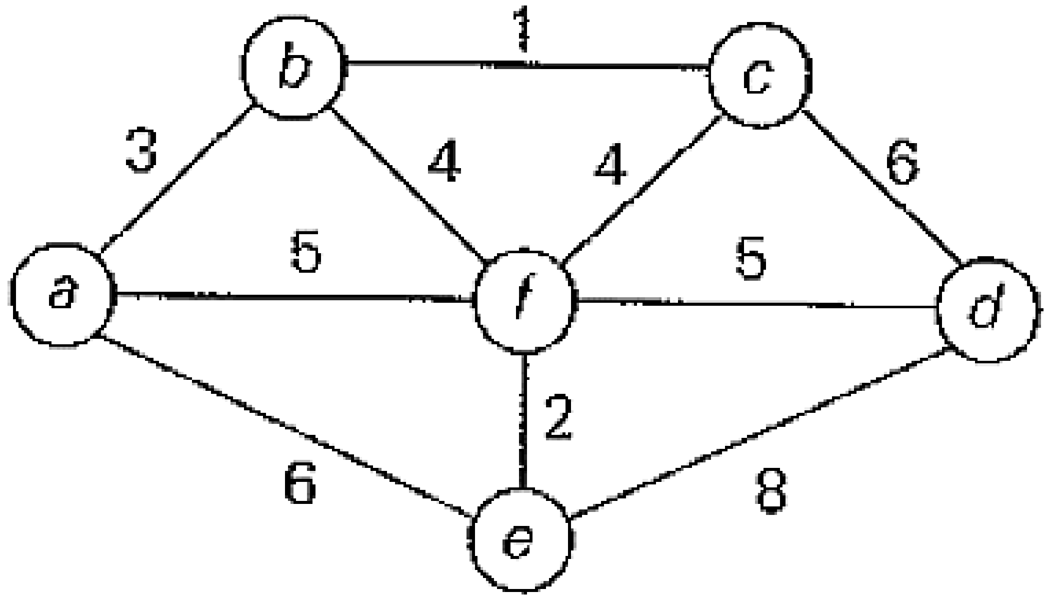
```
ALGORITHM Prim(G)
    //Prim's algorithm for constructing a minimum spanning tree
    //Input: A weighted connected graph $G = \langle V, E \rangle$
    //Output: $E_T$, the set of edges composing a minimum spanning tree of $G$
    $V_T \leftarrow \{v_0\}$      //the set of tree vertices can be initialized with any vertex
    $E_T \leftarrow \emptyset$
    for $i \leftarrow 1$ to $|V| - 1$ do
        find a minimum-weight edge $e^* = (v^*, u^*)$ among all the edges $(v, u)$
        such that $v$ is in $V_T$ and $u$ is in $V - V_T$
        $V_T \leftarrow V_T \cup \{u^*\}$
        $E_T \leftarrow E_T \cup \{e^*\}$
    return $E_T$
```

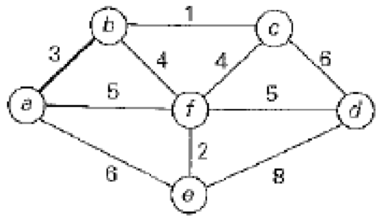
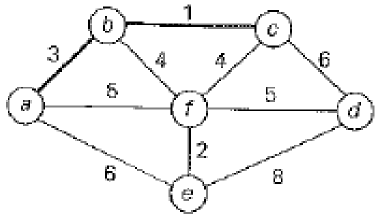
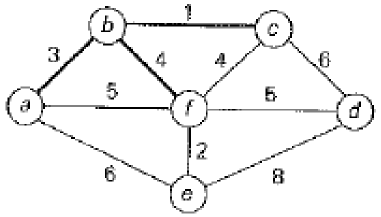
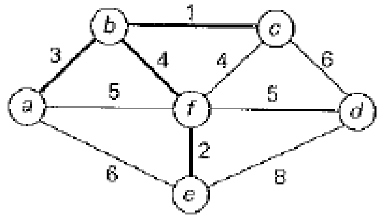
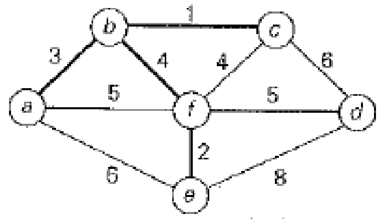
Prim’s algorithm visualization

https://www.cs.usfca.edu/~galles/visualization/Prim.html

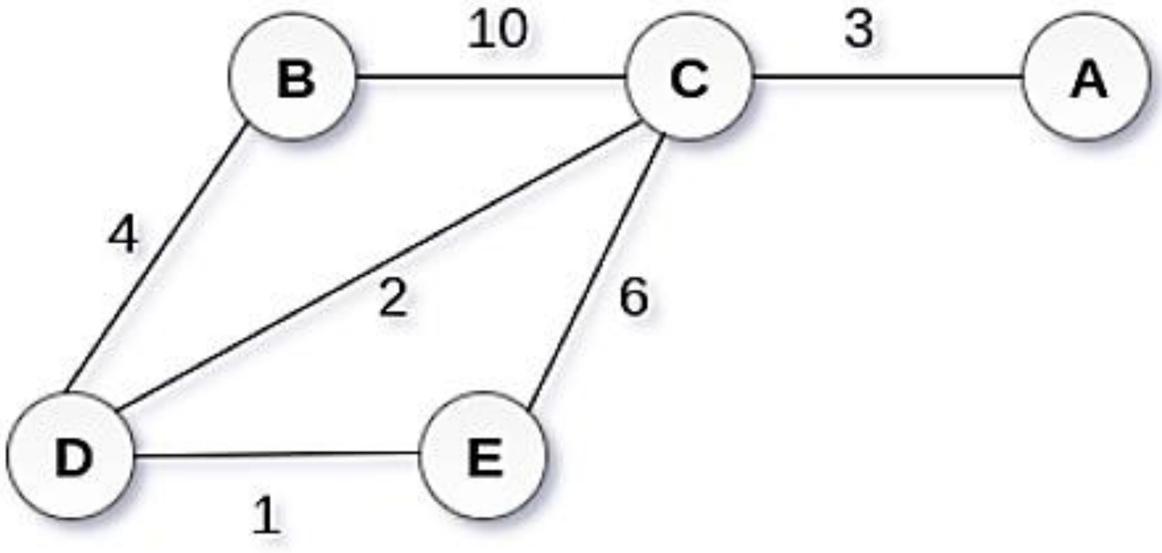
https://visualgo.net/en/mst

Apply Prim’s algorithm to find MST



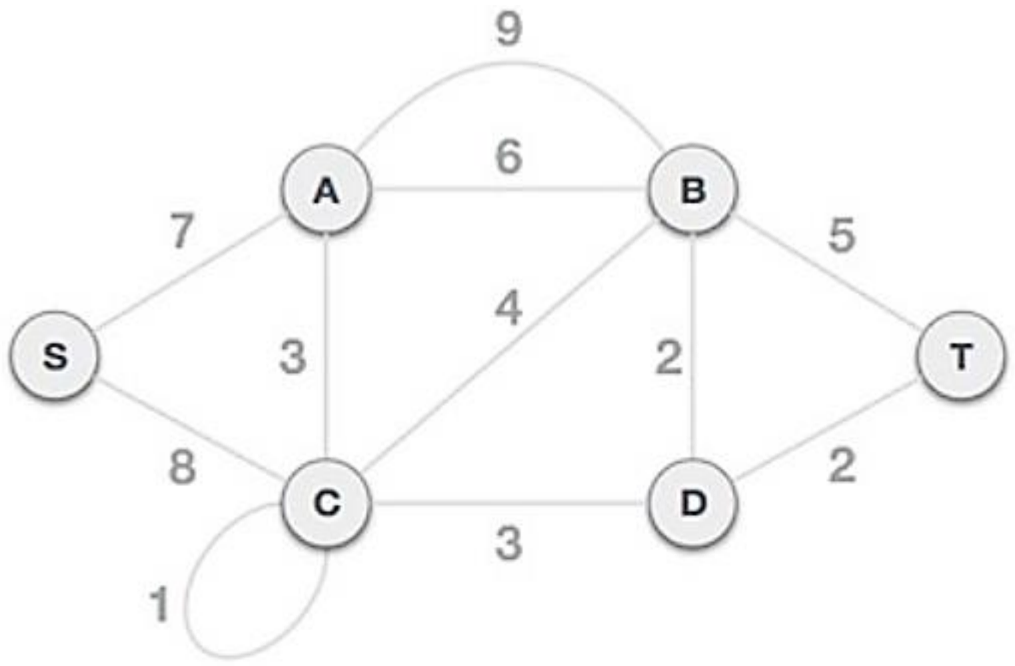
Tree vertices	Remaining vertices	Illustration
a(−, −)	**b(a, 3)** c(−, ∞) d(−, ∞) e(a, 6) f(a, 5)	
b(a, 3)	**c(b, 1)** d(−, ∞) e(a, 6) f(b, 4)	
c(b, 1)	d(c, 6) e(a, 6) **f(b, 4)**	
f(b, 4)	d(f, 5) **e(f, 2)**	
e(f, 2)	**d(f, 5)**	
d(f, 5)		

Let’s check our understanding



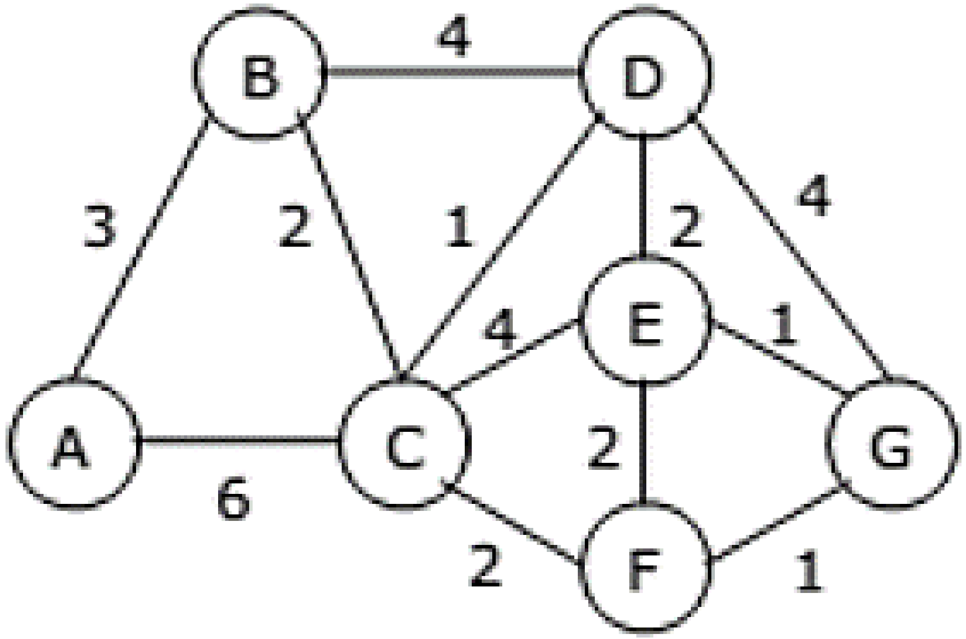
Apply Prim’s algorithm to find MST

Let’s check our understanding



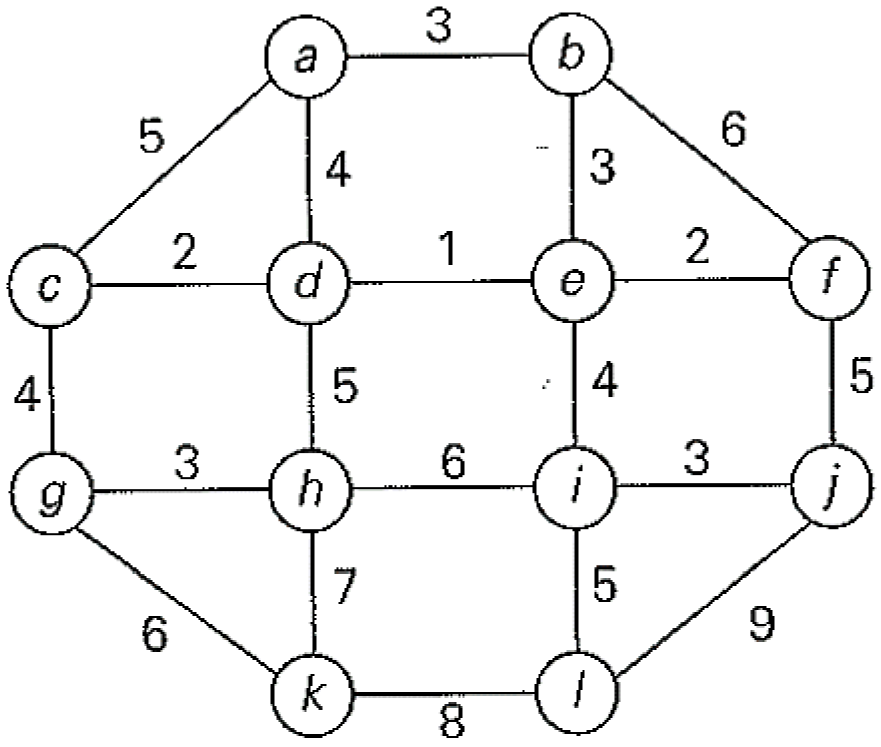
Apply Prim’s algorithm to find MST
(Hint: Remove loops and parallel edges)

Let’s check our understanding…



Apply Prim’s algorithm to find MST

Let’s check our understanding…



Apply Prim’s algorithm to find MST

Extra Miles…

Greedy technique:

Kruskal’s Algorithm

Next session…

Greedy technique:

Dijkstra’s Algorithm