Introduction to Minimum Spanning Tree (MST)

A **minimum spanning tree** is a subgraph of an undirected weighted graph ***G***, such that

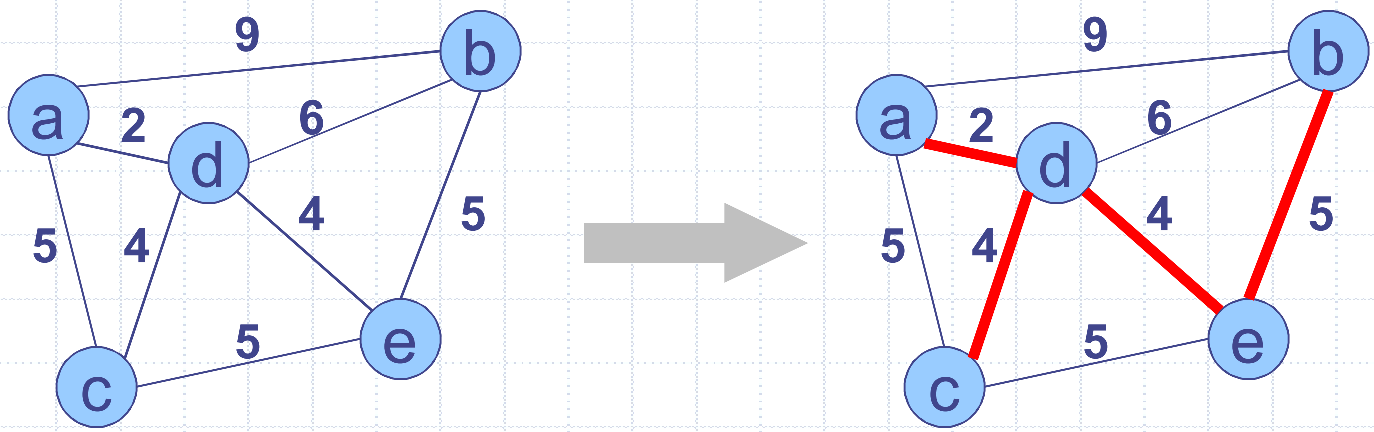
- it is a tree (i.e., it is acyclic)
- it covers all the vertices ***V***
  - contains ***|V| - 1*** edges
- the total cost associated with tree edges is the minimum among all possible spanning trees
- not necessarily unique

Applications & Scope of MST

- Application :Any time you want to visit all vertices in a graph at minimum cost (e.g., wire routing on printed circuit boards, sewer pipe layout, road planning…)

- Scope of research : Internet content distribution
  - Idea: publisher produces web pages, content distribution network replicates web pages to many locations so consumers

How Can We Generate a MST?



Kruskal's Algorithm

- Kruskal's algorithm builds a minimum-cost spanning tree T by adding edges to T one at a time.
- The algorithm selects the edges for inclusion in T in nondecreasing order of their cost.
- An edge is added to T if it does not form a cycle with the edges that are already in T.

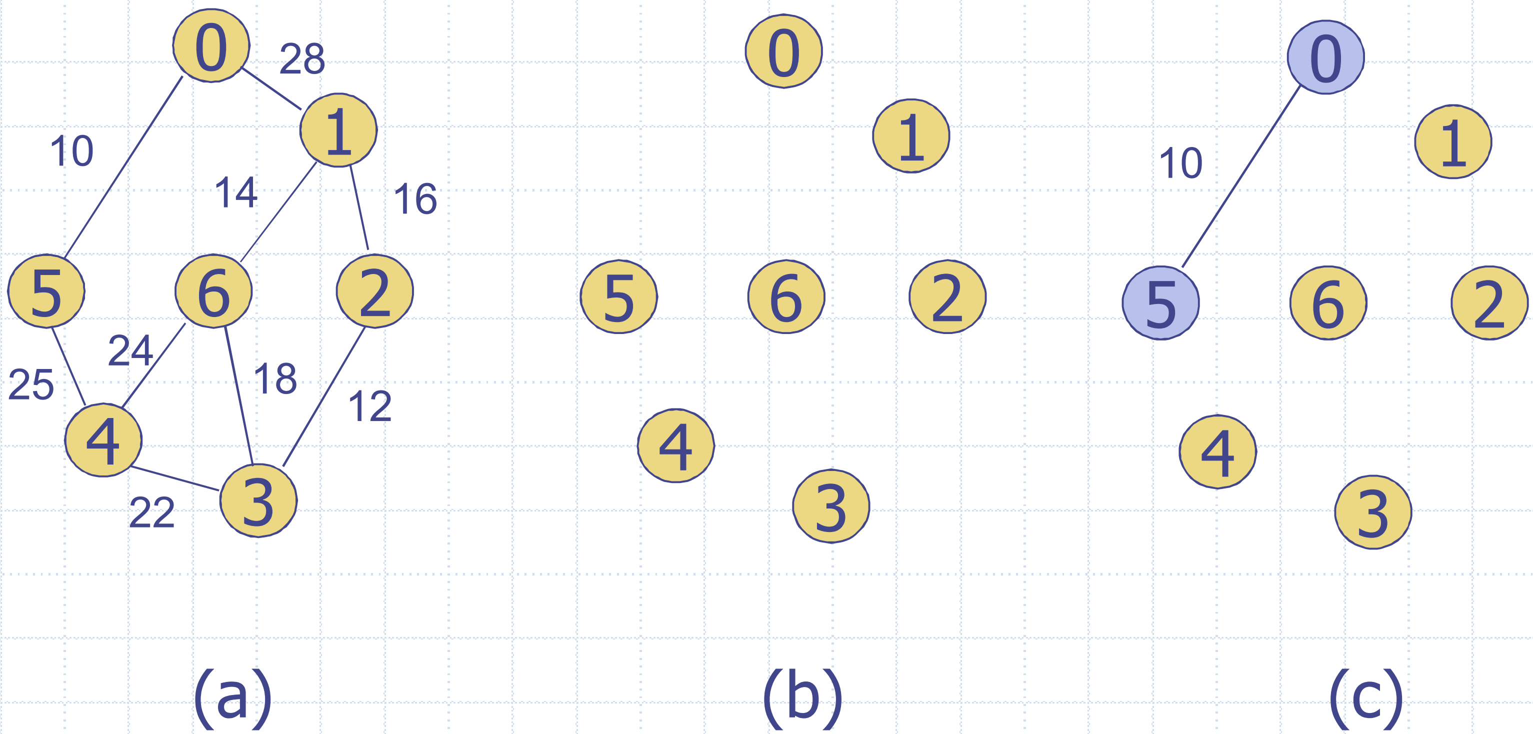
- make_set(v) : create a new set whose only member is V
- find_set(u) : return a representative elements from the set that contain u.
- find_set(v) : return a representative elements from the set that contain v.
- union(u,v):unites the dynamic set that contain u & v into new set that contain u & v

Kruskal's Algorithms

```
MST_Kruskal(G,w)
A:=0
  For each vertex v∈V[G]
      do make_set(v)
  sort the edge of E into nondecreasing order of weight
  For each edge(u,v)∈E taken in nondecresing  order of weight
      do if find_set(u)≠find-set(v)
          then A:=A U {(u,v)}
          union(u,v)
  Return A
```

Stages in Kruskal's Algorithm

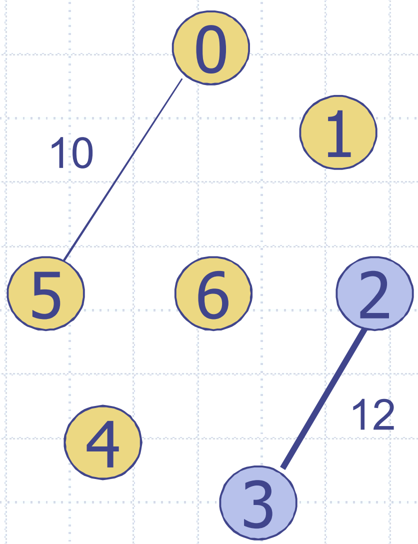
Start from edges of smallest cost which would not cause cycle
Until n-1 edges



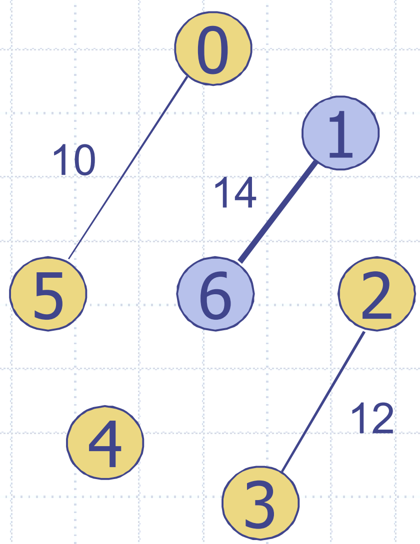
Analysis of Kruskal's algorithm

- Let V be the no. of vertices & E be the no. of edges
- It take Θ(Elog E) time to sort the edges. The for loop iterated E no. of times. Each access takes Θ(V) time, which gives a total of Θ(E log V) times.
- Total running time is Θ(V+ ElogV).Since V is not aymptotically larger than E.Thus running time is Θ(E log V)

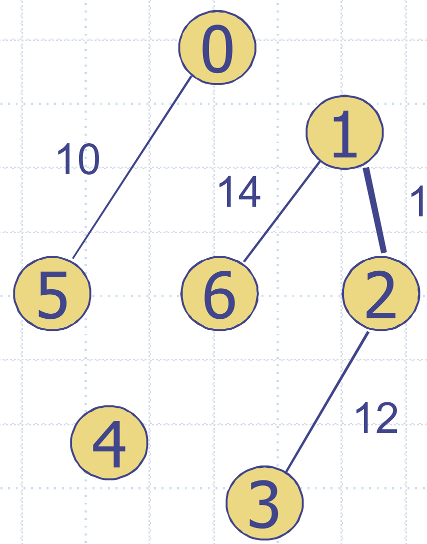
Stages in Kruskal's Algorithm (Cont.)



(d)

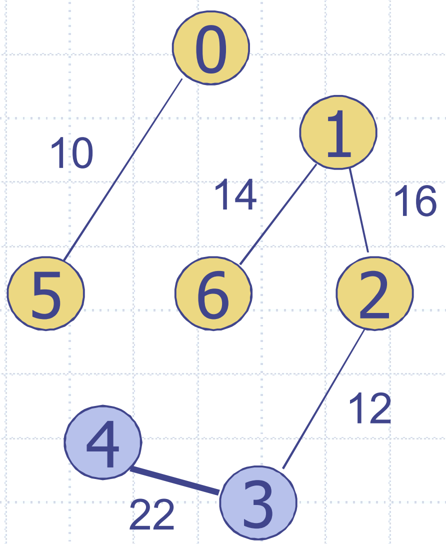


(e)

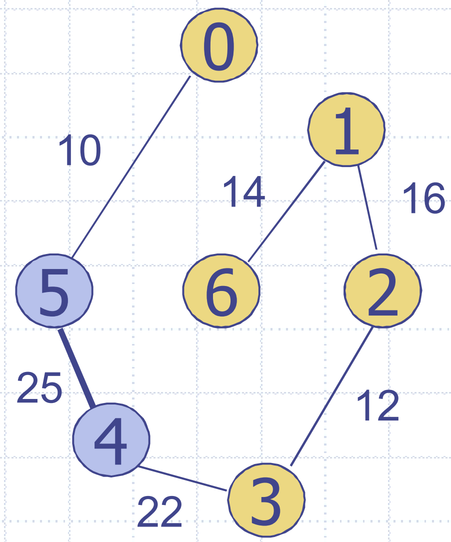


(f)

Stages in Kruskal's Algorithm (Cont.)



(g)



(h)

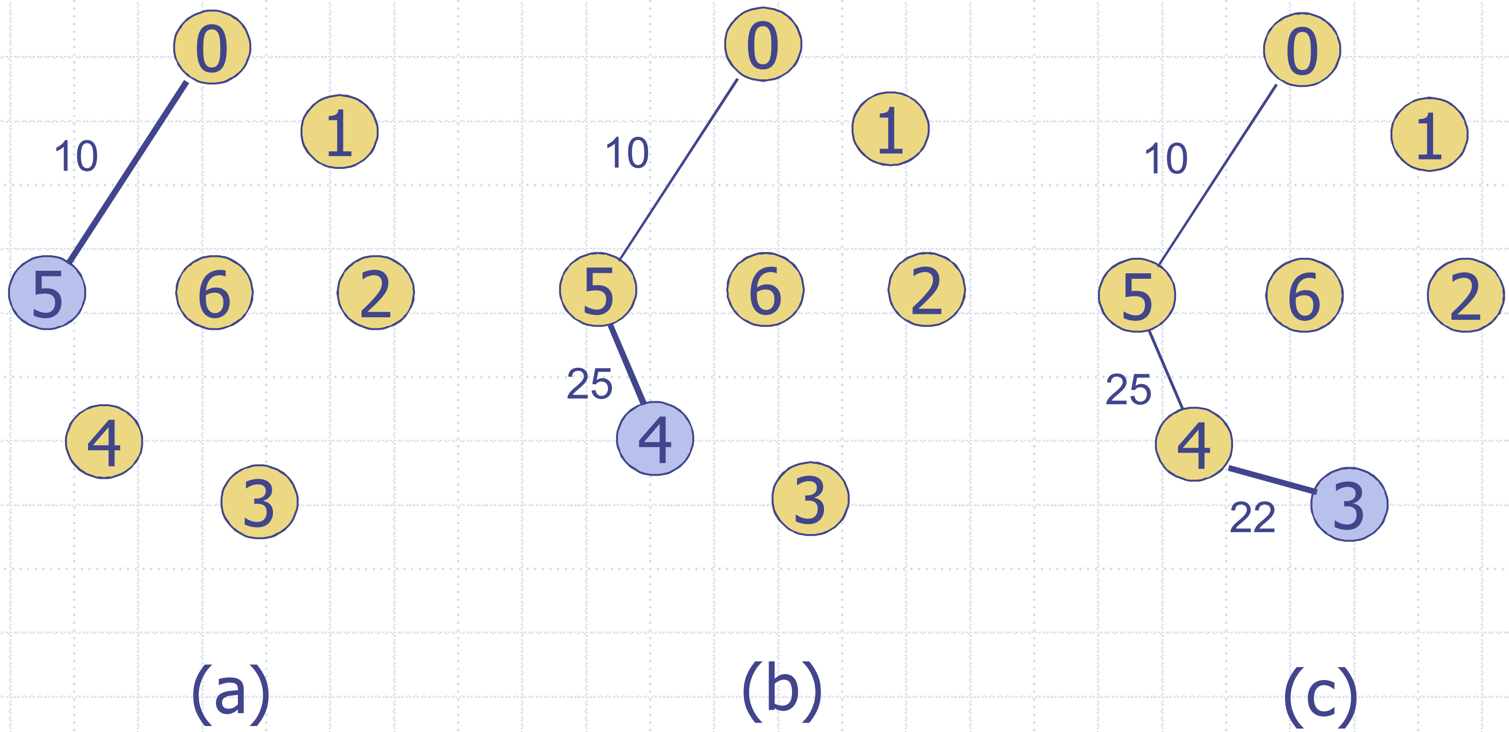
Prim's Algorithm

- Similar to Kruskal's algorithm, Prim's algorithm constructs the minimum-cost spanning tree edge by edge.
- The difference between Prim's algorithm and Kruskal's algorithm is that the set of selected edges forms a tree at all times when using Prim's algorithm while a forest is formed when using Kruskal's algorithm.
- In Prim's algorithm, a least-cost edge (u, v) is added to T such that $T \cup \{(u, v)\}$ is also a tree. This repeats until T contains n-1 edges.
- Prim's algorithm has a time complexity $O(n^2)$.

Prims Algorithm

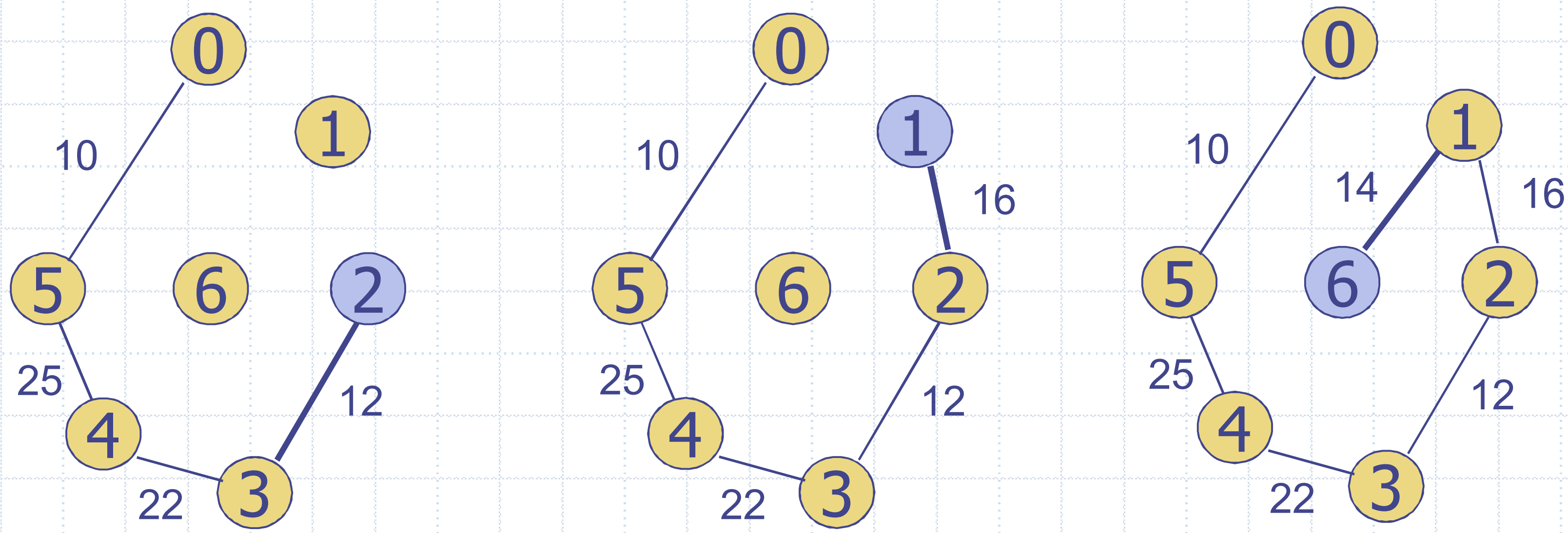
```
MST_Prim(G,w)
  For each vertex u∈V[G]
      do key[u]:= ∞
       π[u]:=NIL
   key[r]:=0
   Q:=V[G]
   While Q≠ф
      Do u:=Extract-Min(Q)
         For each v ∈ adj[u]
           do if v ∈ Q and w(u,v) < key [v]
              then π[v]:=u
                 key[v]:=w(u,v)
```

Stages in Prim's Alogrithm



Include the minimal edges which would form a tree until n-1 edges

Stages in Prim's Alogrithm (Cont.)



(d) (e) (f)

Assignment

Q.1)What is MST?

Q.2)What is difference between Kruskals & Prims algorithm/

Q.3)Find out MST of following graph using Prim's algorithm(root vertex e)