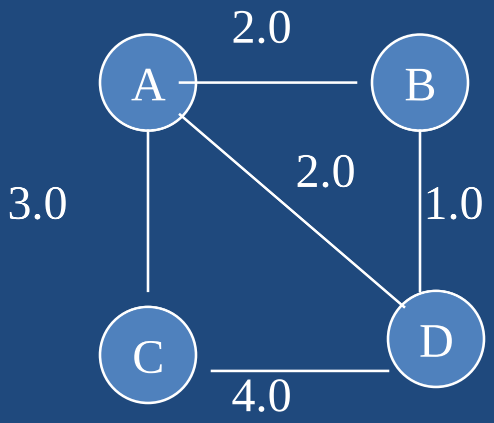
SPANNING TREES

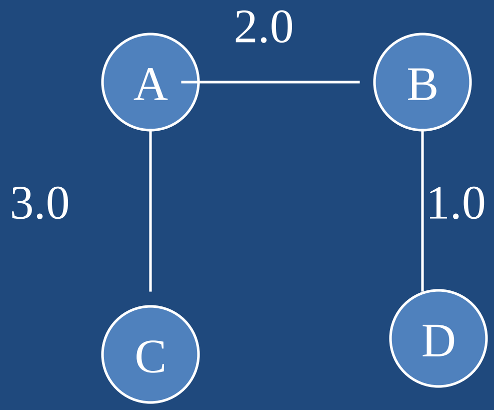
Minimum Spanning Tree

- A **Spanning Tree** for a connected, undirected graph, G = (V, E), is a subgraph of G that is an undirected tree and contains all the vertices of G.
- In a weighted graph G = (V, E, W), the weight of a subgraph is the sum of the weights of the edges in the subgraph.
- A **minimum spanning tree (MST)** for a weighted graph is a spanning tree with minimum weight.

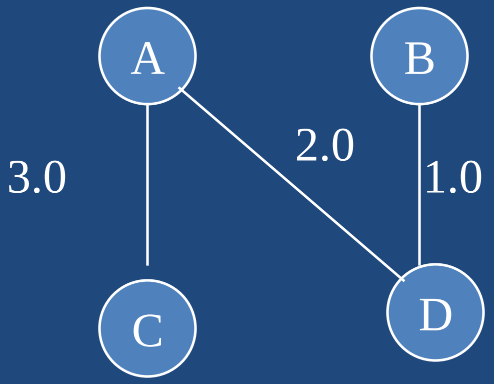
Minimum Spanning Tree

- Consider the following graph
    - The possible spanning trees for this graph are

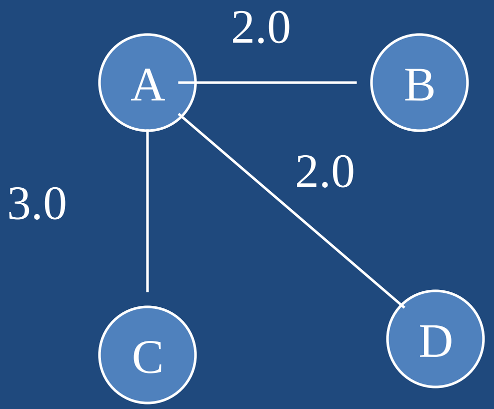




MST Weight is 6



MST Weight is 6



Weight is 7

Minimum Spanning Tree

- Minimum spanning trees are useful when we want to find the cheapest way to connect a
  - Set of cities by roads
  - Set of electrical terminals or computers by wires or telephone lines
  - Etc…

Kruskal's Algorithm

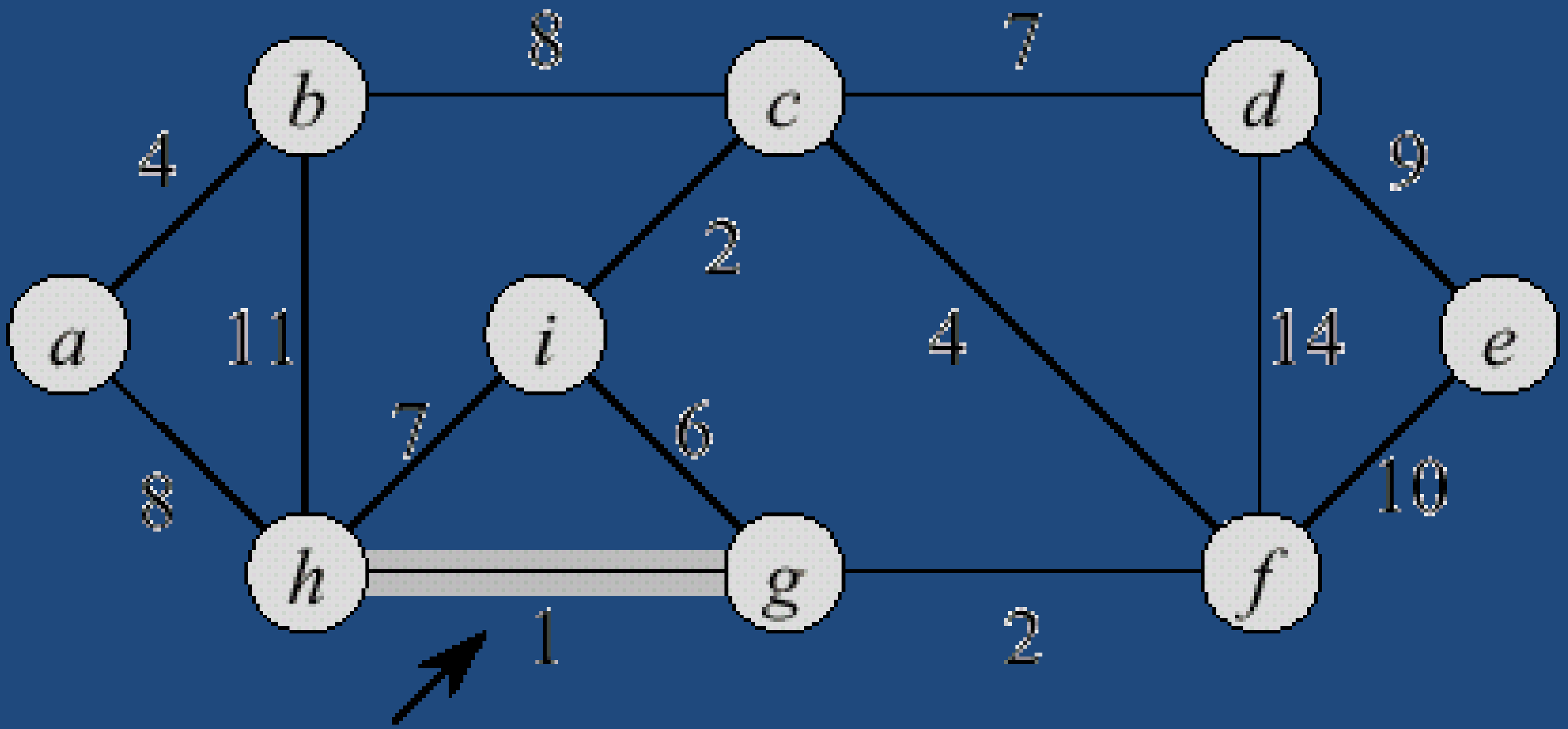
- Edge based algorithm
- Add the edges one at a time, in increasing weight order
- The algorithm maintains a **forest of trees**.
  - An edge is accepted it if connects vertices of distinct trees
- We need a data structure that maintains a partition, i.e.,a collection of disjoint sets
  - MakeSet(S,x): $S \leftarrow S \cup \{x\}$
  - Union(S_i,S_j): $S \leftarrow \{S_i \cup S_j\}$
  - FindSet(S, x): returns unique $S_i \in S$, where $x \in S_i$

Kruskal's Algorithm

- The algorithm adds the cheapest edge that connects two trees of the forest

```
MST-Kruskal(G,w)
 A ← ∅
 for each vertex v ∈ V[G] do
    Make-Set(v)
 sort the edges of E by non-decreasing weight w
 for each edge (u,v) ∈ E, in order by non-
   decreasing weight do
   if Find-Set(u) ≠ Find-Set(v) then
      A ← A ∪ {(u,v)}
      Union(u,v)
 return A
```

Example



8
7
4
b
c
d
9
2
a
11
i
4
14
e
7
6
8
10
h
g
f
1
2