CME 2001
Data Structures and Algorithms

Zerrin Işık
zerrin@cs.deu.edu.tr

Minimum Spanning Trees

Minimum Spanning Tree (MST)

- A town has a set of houses and a set of roads.
- A road connects only 2 houses.
- A road connecting houses $\boldsymbol{u}$ and $\boldsymbol{v}$ has a repair cost $\boldsymbol{w(u,v)}$.

Goal: Repair enough (not more) roads such that :

1. everyone stays connected: can reach every house from all other houses, and
2. total repair cost will be minimum.

Minimum Spanning Tree (MST) …

- Undirected graph $G = (V, E)$
- Weight $\boldsymbol{w(u,v)}$ on each edge $\boldsymbol{(u,v)} \in E$.

- Find $\boldsymbol{T \subseteq E}$ such that:
  1. T connects all vertices (T is spanning tree), and
  2. $w(T) = \sum_{(u,v) \in T} \boldsymbol{w(u,v)}$ is minimized

- A spanning tree whose weight is minimum over all spanning trees is called a ***minimum spanning tree*** or ***MST***.

Properties of an MST

- It has $|V| - 1$ edges.
- It has no cycles.
- It might not be unique.

Definitions

Let $\boldsymbol{S}$ is s subset of $\boldsymbol{V}$, and $\boldsymbol{A} \subseteq \boldsymbol{E}$

- A ***cut*** $(S, V - S)$ is a partition of vertices into disjoint sets V and S - V.
- Edge (u,v) $\in$ E ***crosses*** cut $(S, V - S)$ if one endpoint is in S and the other is in $V - S$.
- An edge is a ***light edge*** crossing a cut, if and only if its weight is minimum over all edges crossing the cut. For a given cut, there can be > 1 light edge crossing it.

Kruskal's Algorithm

$G = (V, E)$ is a connected, undirected, weighted graph. $w : E \to \mathbf{R}$.

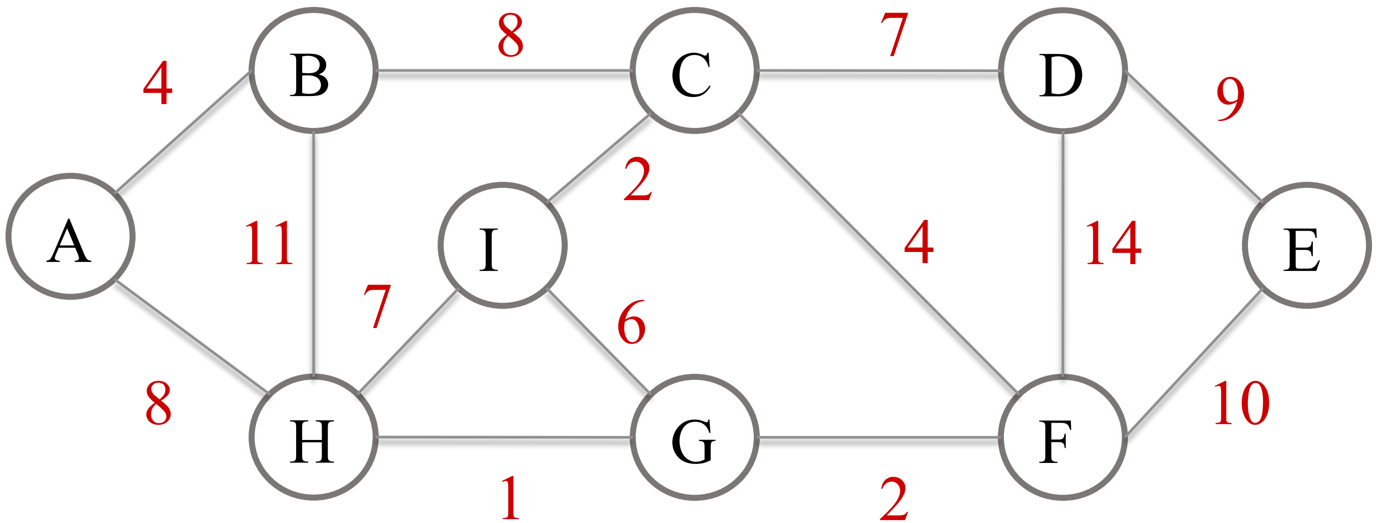
- Starts with each vertex being its own component.
- Repeatedly merges two components into one by choosing the light edge that connects them (i.e., the light edge crossing the cut between them).
- Scans the set of edges in monotonically increasing order by weight.
- Uses a disjoint-set data structure to determine whether an edge connects vertices in different components.

Kruskal's Algorithm

KRUSKAL(G, w)

```
A = ∅
for each vertex v ∈ G.V
    MAKE-SET(v)
sort the edges of G.E into nondecreasing order by weight w
for each (u, v) taken from the sorted list
    if FIND-SET(u) ≠ FIND-SET(v)
        A = A ∪ {(u, v)}
        UNION(u, v)
return A
```

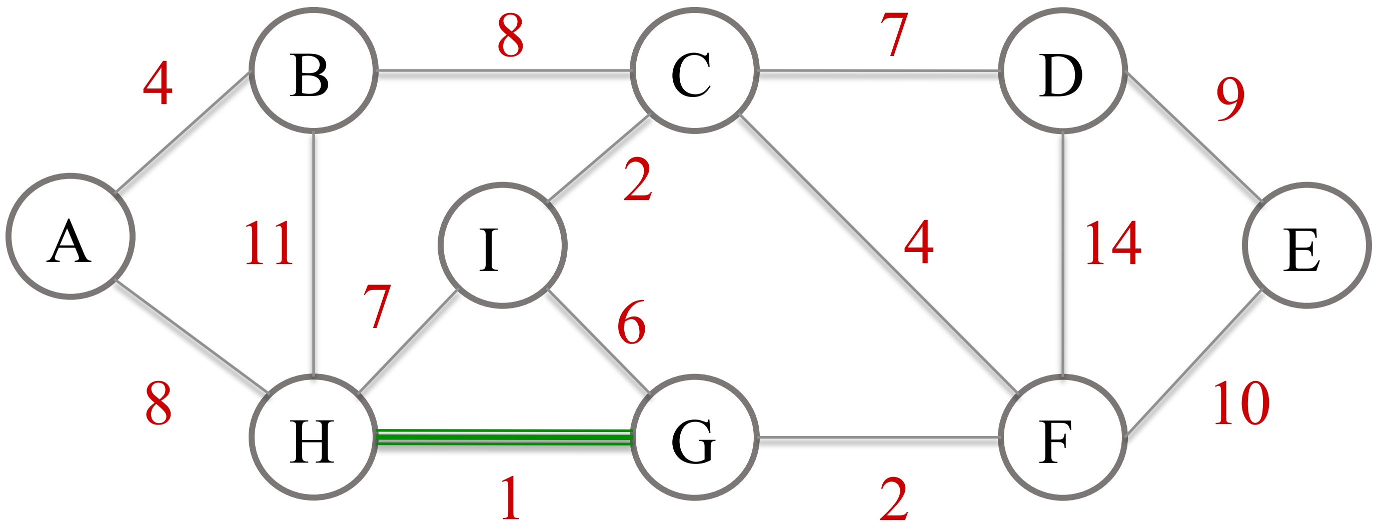
Example (MST)



A: {}

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

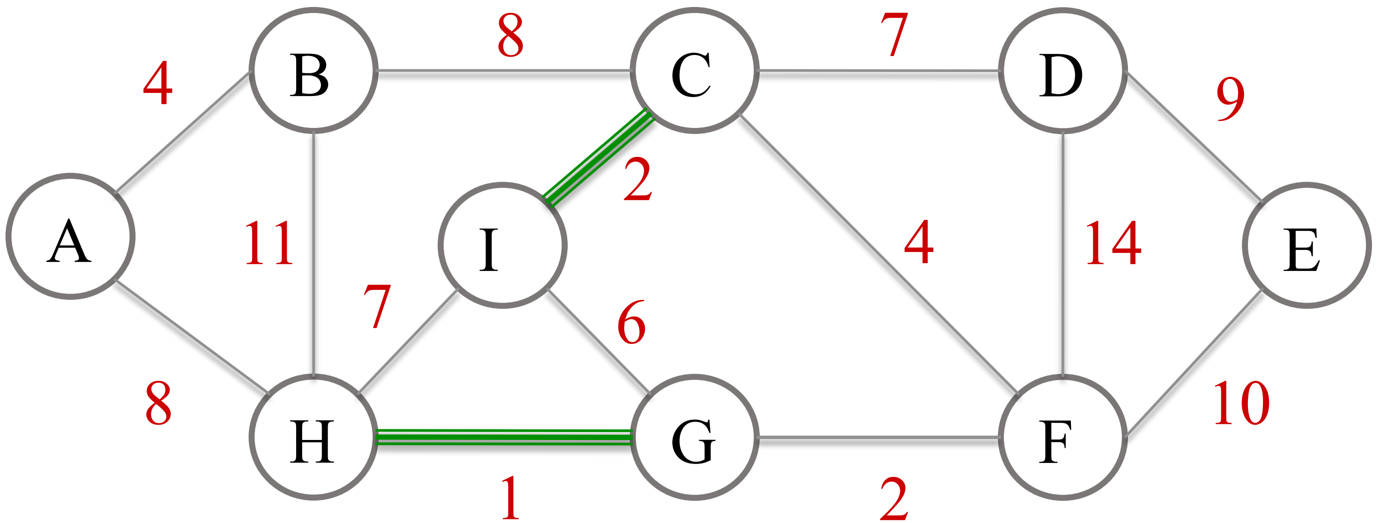
Example (MST)



A:{ (h,g) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

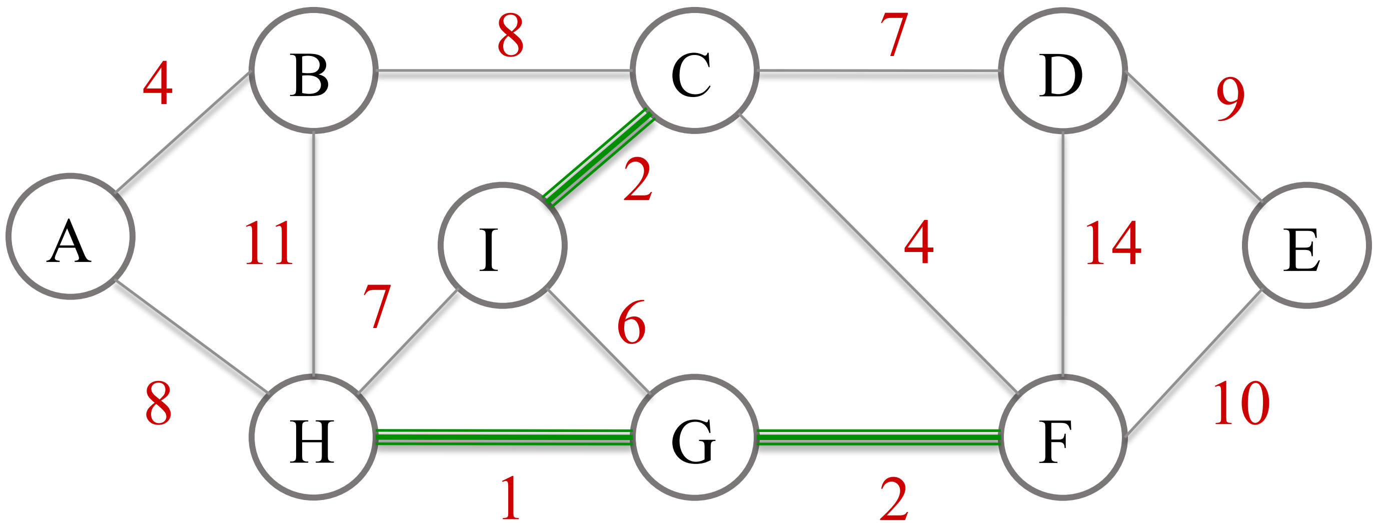
Example (MST)



A:{ (h,g),(i,c) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

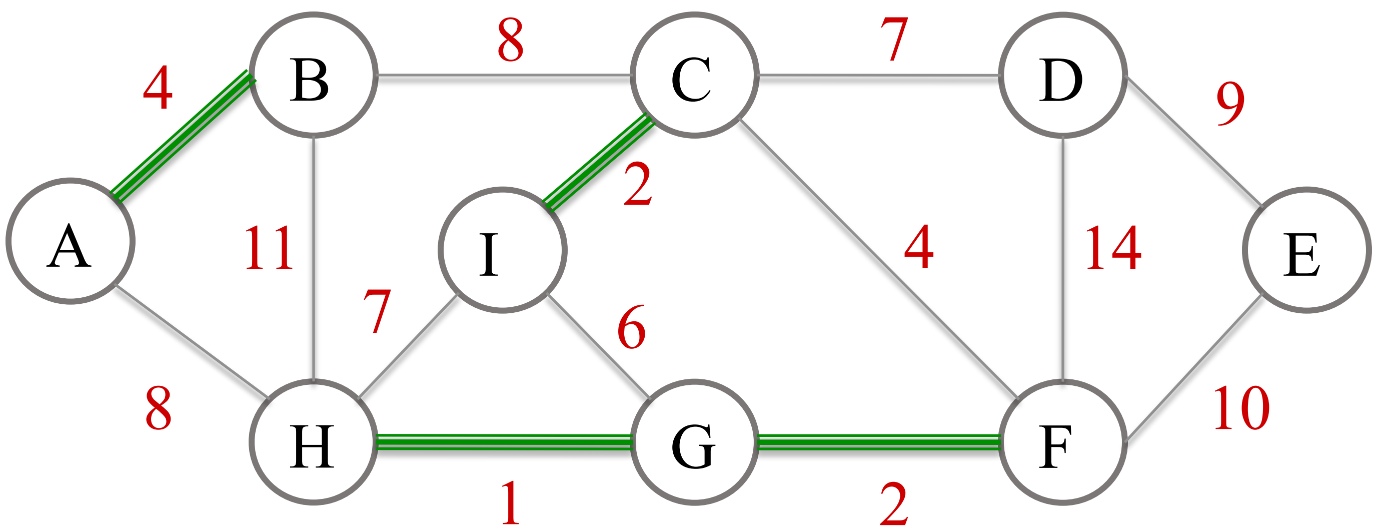
Example (MST)



A:{ (h,g),(i,c),(g,f) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

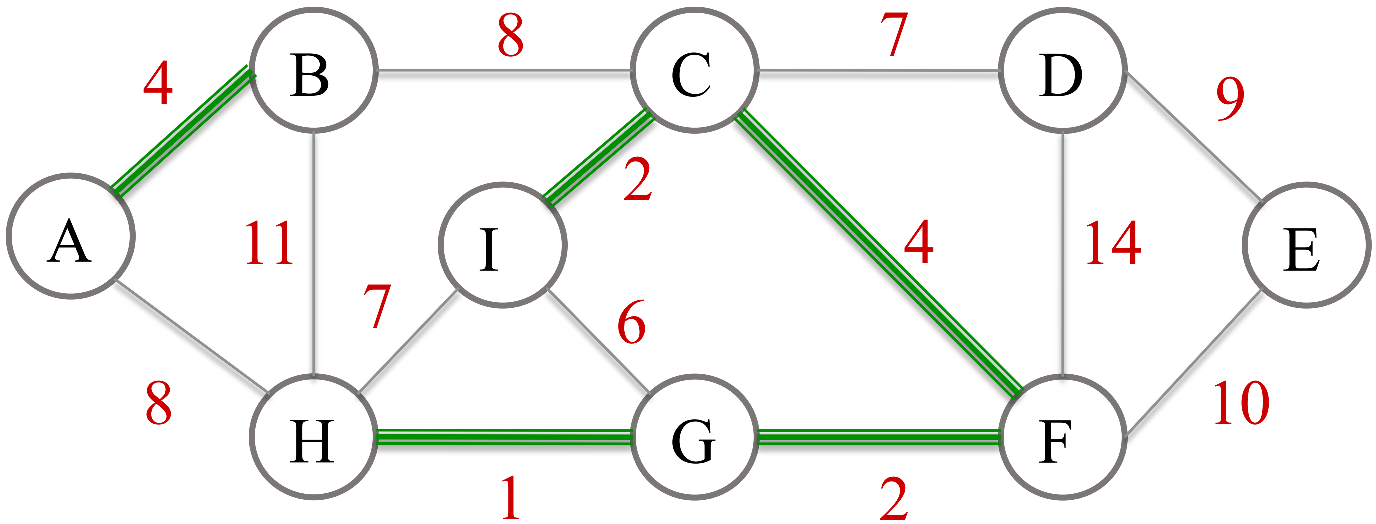
Example (MST)



A:{ (h,g),(i,c),(g,f),(a,b) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

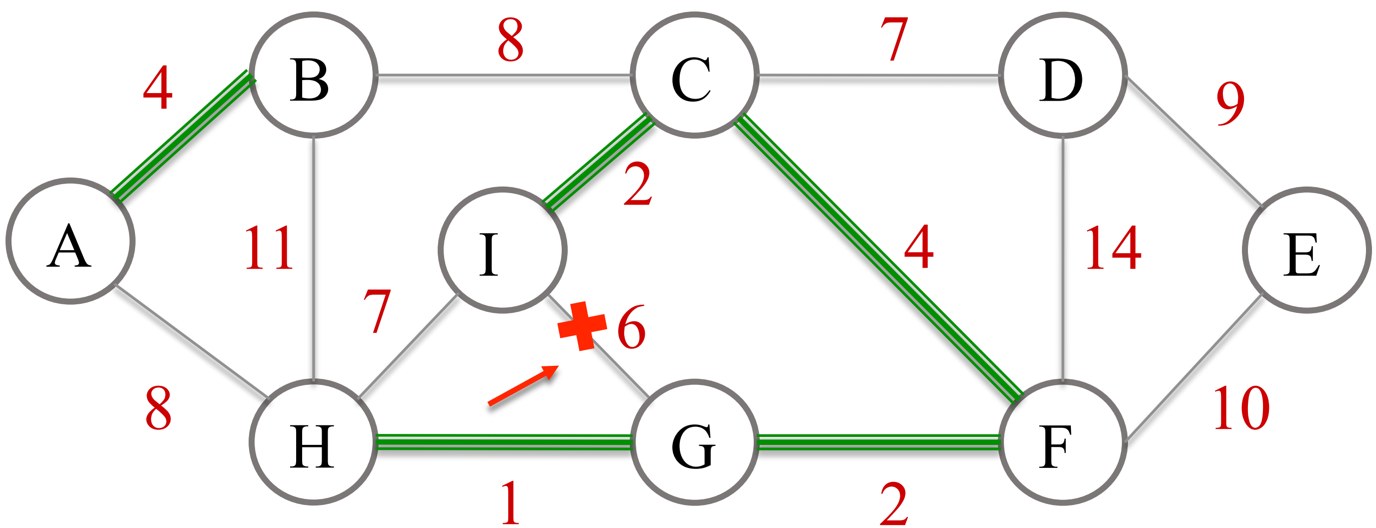
Example (MST)



A:{ (h,g),(i,c),(g,f),(a,b),(c,f) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

Example (MST)



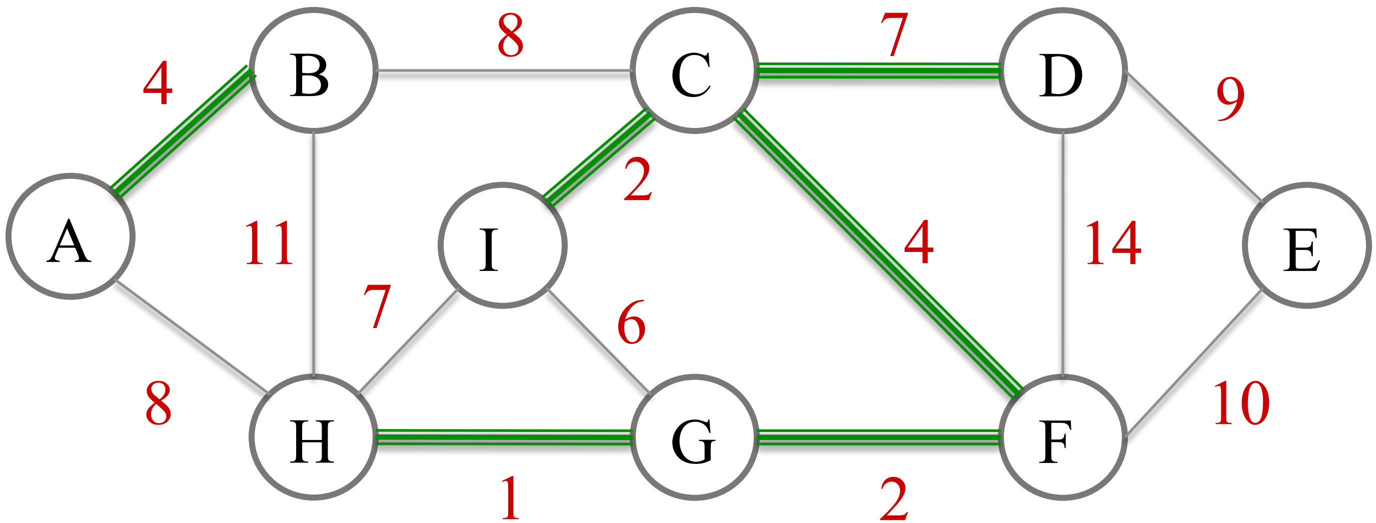
Otherwise it would create a cycle in MST!

FIND-SET($\boldsymbol{u}$) == FIND-SET($\boldsymbol{v}$) ➡ not add (u,v) to $\boldsymbol{A}$

A: { (h,g),(i,c),(g,f),(a,b),(c,f) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

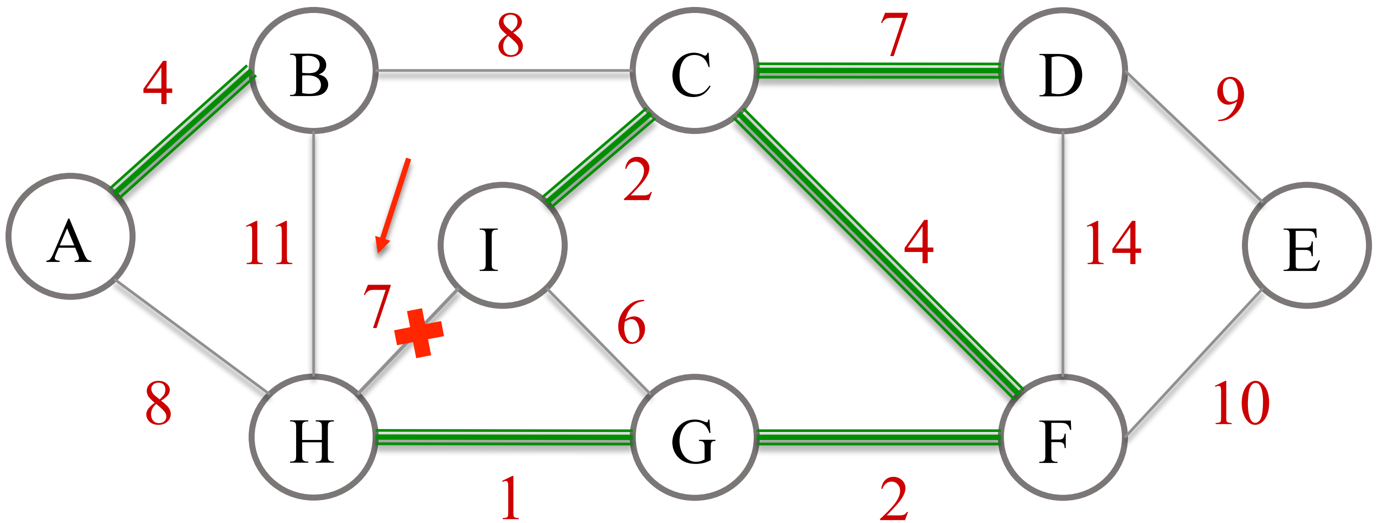
Example (MST)



A: { (h,g),(i,c),(g,f),(a,b),(c,f),(c,d) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

Example (MST)

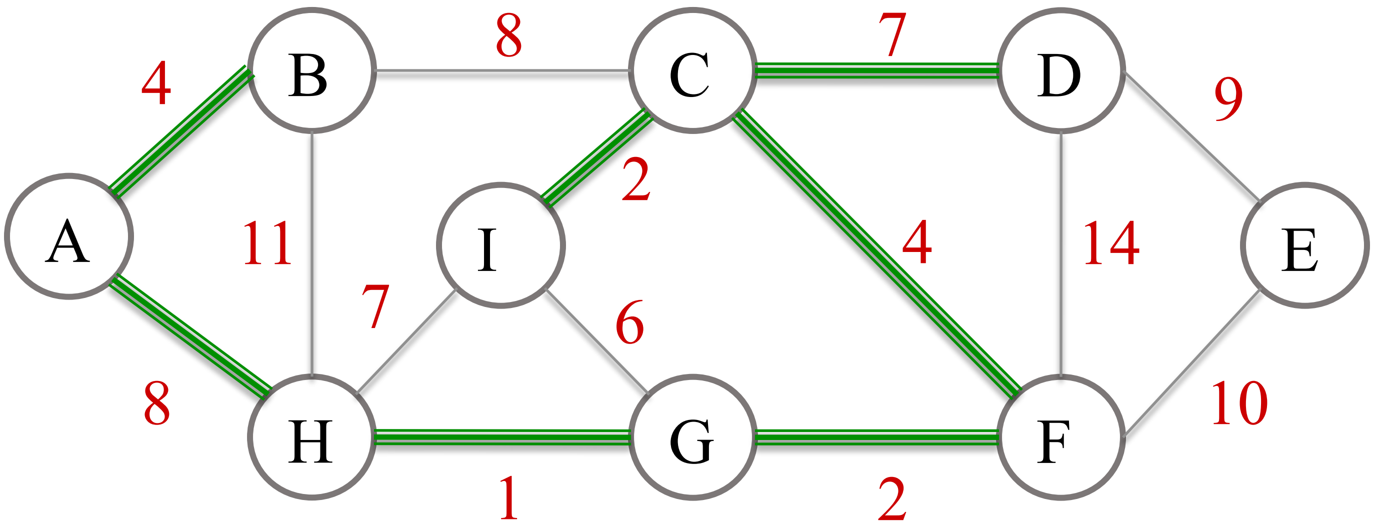


FIND-SET($\boldsymbol{u}$) == FIND-SET($\boldsymbol{v}$) ➟ not add (u,v) to $\boldsymbol{A}$

A:{ (h,g),(i,c),(g,f),(a,b),(c,f),(c,d) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

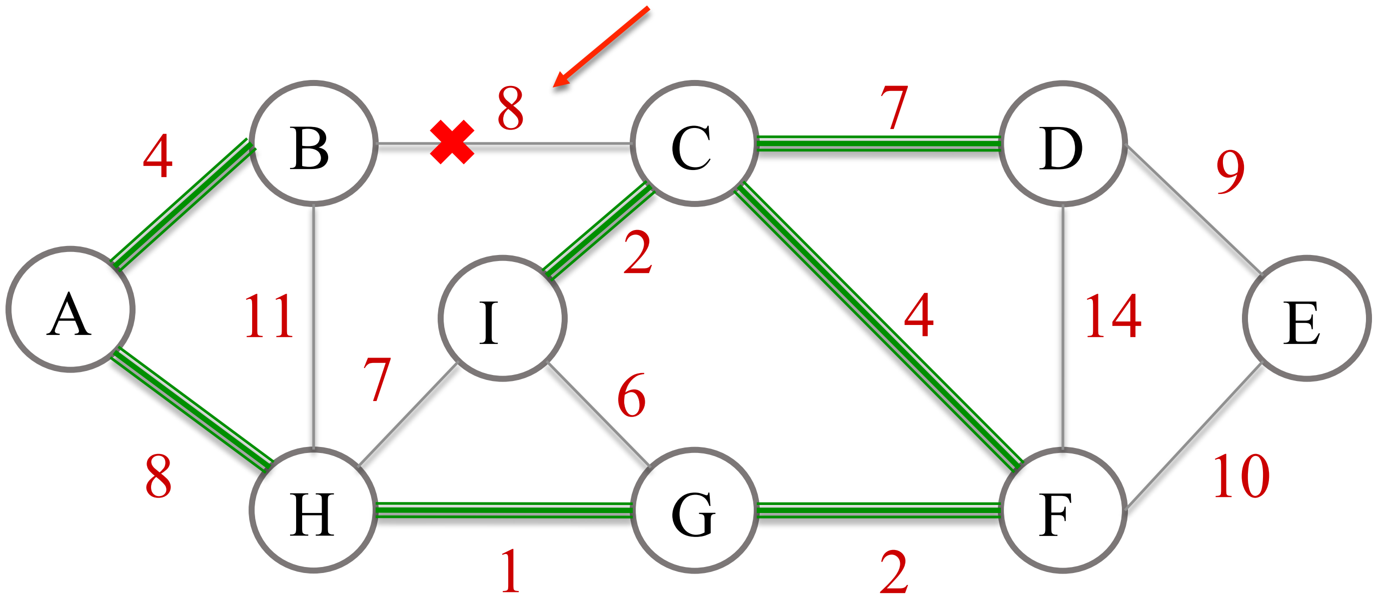
Example (MST)



A: { (h,g),(i,c),(g,f),(a,b),(c,f),(c,d),(a,h) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

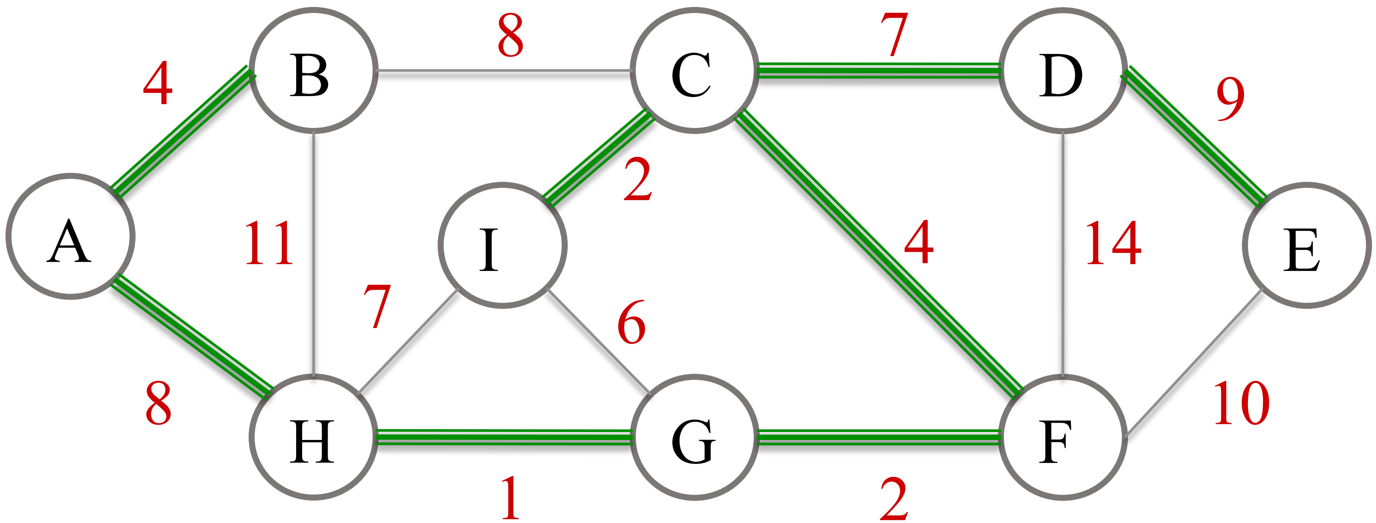
Example (MST)



FIND-SET(u) == FIND-SET(v) ➡ not add (u,v) to A

A: { (h,g),(i,c),(g,f),(a,b),(c,f),(c,d),(a,h) }
E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

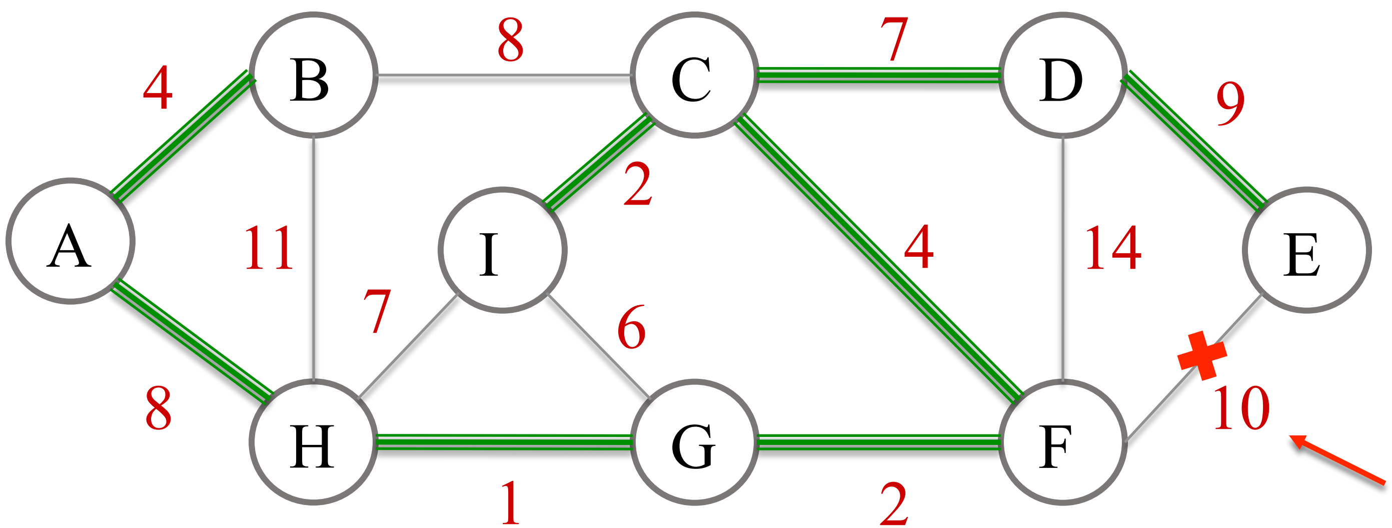
Example (MST)



A:{ (h,g),(i,c),(g,f),(a,b),(c,f),(c,d),(a,h),(d,e) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

Example (MST)

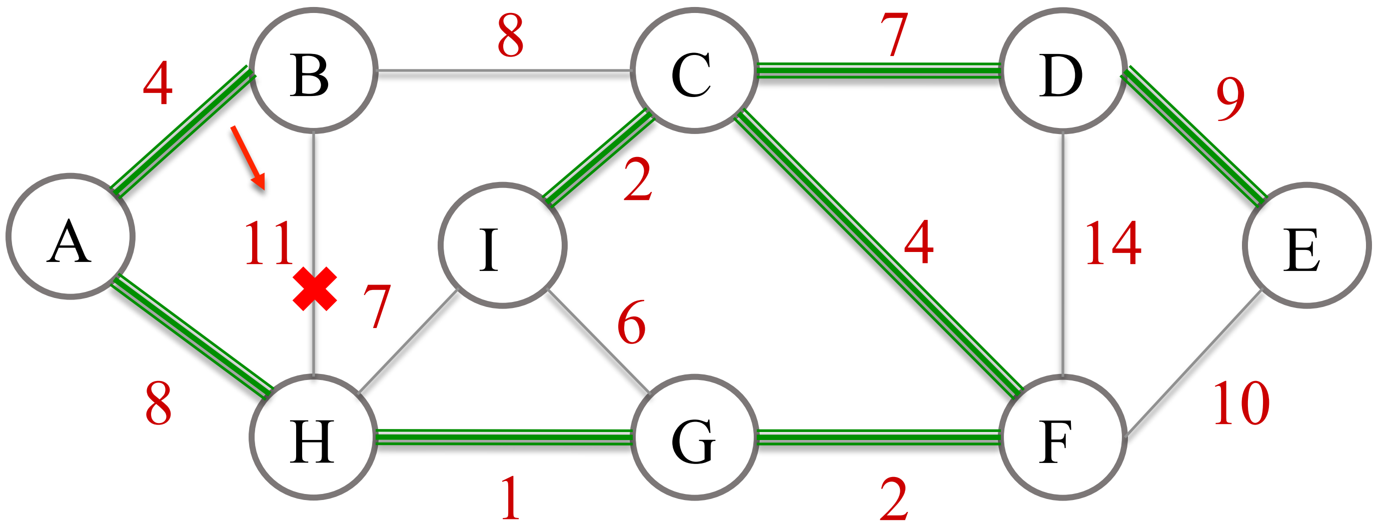


FIND-SET(u) == FIND-SET(v) ➡ not add (u, v) to A

A: { (h,g),(i,c),(g,f),(a,b),(c,f),(c,d),(a,h),(d,e) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

Example (MST)

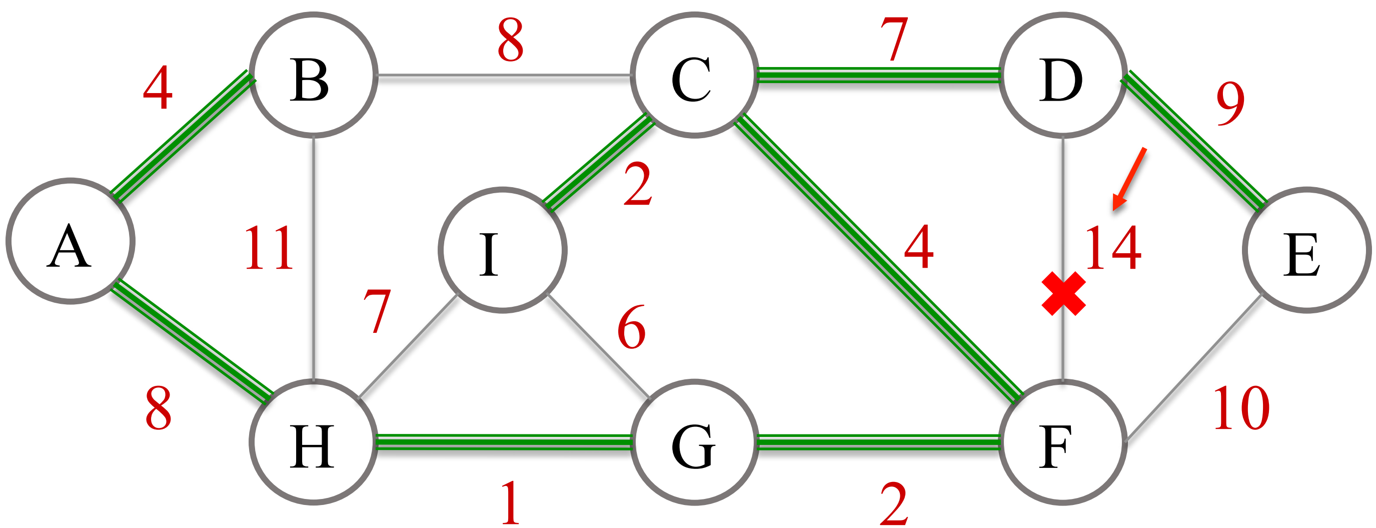


FIND-SET($\boldsymbol{u}$) == FIND-SET($\boldsymbol{v}$) ➠ not add (u,v) to $\boldsymbol{A}$

A: { (h,g),(i,c),(g,f),(a,b),(c,f),(c,d),(a,h),(d,e) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

Example (MST)

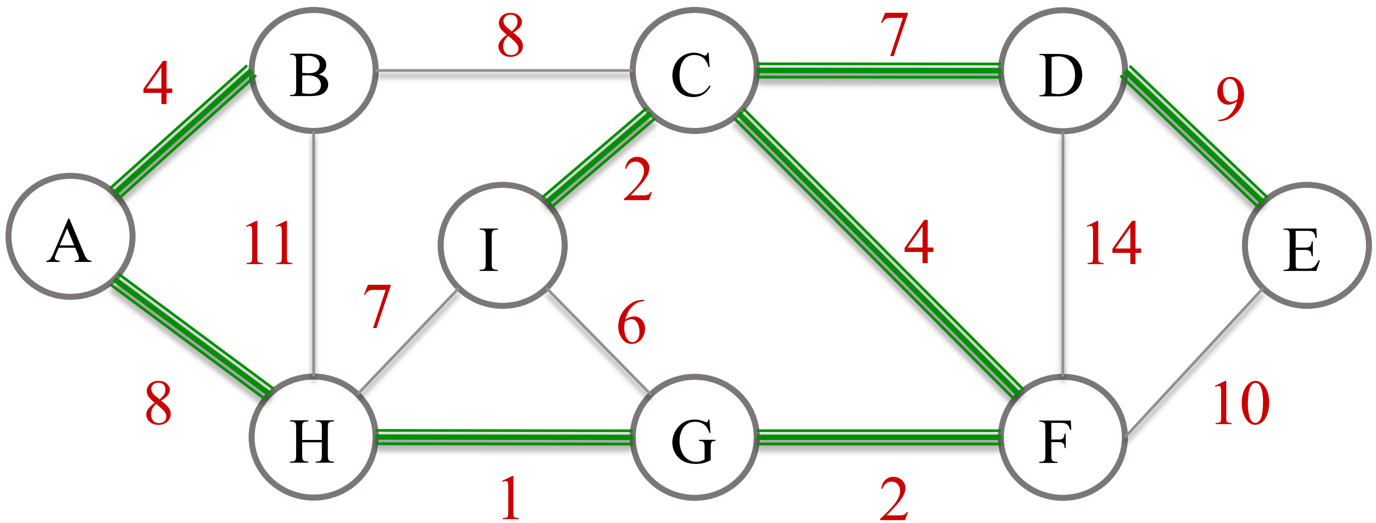


FIND-SET(***u***) == FIND-SET(***v***) ➟ not add *(u,v)* to ***A***

A:{ (h,g),(i,c),(g,f),(a,b),(c,f),(c,d),(a,h),(d,e) }

E: 1 2 2 4 4 6 7 7 8 8 9 10 11 14

Final MST



Analysis of Kruskal's Algorithm

KRUSKAL(G, w)

$A = \emptyset$

for each vertex $v \in G.V$ $\left.\right\}$ O(V)

MAKE-SET(v)

sort the edges of $G.E$ into nondecreasing order by weight w $\longrightarrow$ O(E.lg$E$)

for each (u, v) taken from the sorted list

if FIND-SET$(u) \neq$ FIND-SET(v)

$A = A \cup \{(u, v)\}$

UNION(u, v) $\left.\right\}$ O((V+E) α(V))

return A

In total : O((V+E) α(V)) + O(E.lg$E$)

$|E| \geq V-1$ ➟ O(E α(V)) + O(E.lg$E$)

$\alpha(V)$ = O(lgV) = O(lgE) ➟ O(E lgE) + O(E lgE) = **O(E.lg$E$)**

(Note: if $|E| \leq |V|^2$ ➟ lgE = O(2 lgV) = O(lgV) ➟ Runtime: O(E.lg$V$))

Applications of MST

- Phone network design. Connect all offices of your company via a landline phone system by paying minimum cost for telephone leasing.
- Travelling salesman
- Clustering
- Construct trees to broadcast in computer networks
- Image segmentation
- Circuit design
- …

Next Week Topic

- Shortest Path Algorithms (Chapter 24)