Algorithms

Lecture Topic: Minimum Spanning Tree

Anxiao (Andrew) Jiang

Roadmap of this lecture:

1. Minimum Spanning Tree.

    1.1 Define "Minimum Spanning Tree (MST)".

    1.2 Kruskal's Algorithm for MST.

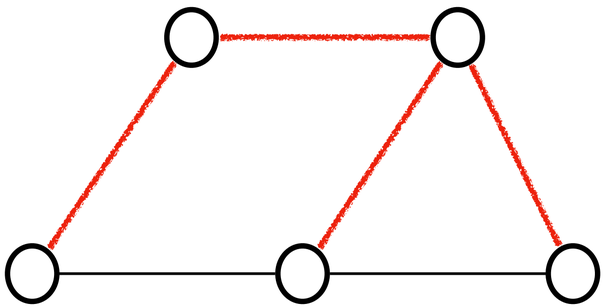
    1.3 Prim's Algorithm for MST.

    1.4 Correctness of Kruskal's Algorithm and Prim's Algorithm.

Minimum Spanning Tree

Spanning Tree of graph G=(V,E)
is a sub-graph of G that:

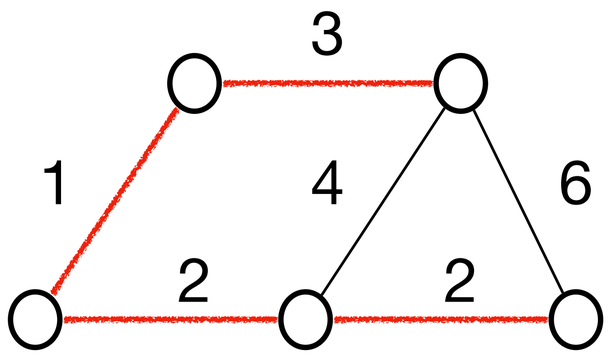
1) is a tree
2) contains all nodes of G



Minimum Spanning Tree

Minimum Spanning Tree (MST) of graph G=(V,E)
is a sub-graph of G that:

1) is a tree
2) contains all nodes of G
3) Minimizes its total edge weight



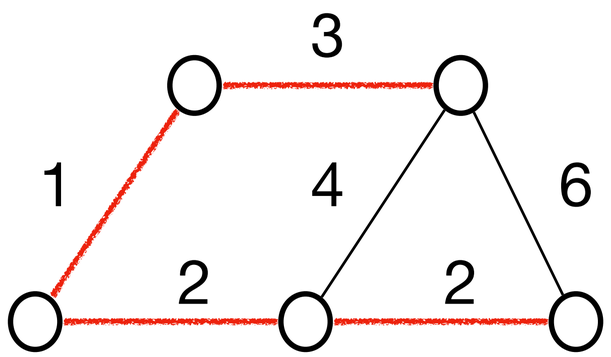
MST

Total weight = 1 + 2 + 3 + 2 = 8

Minimum Spanning Tree

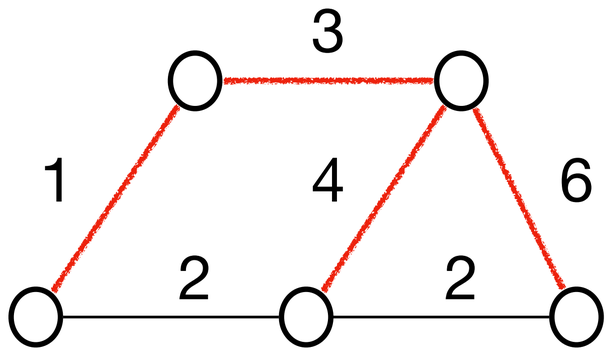
Minimum Spanning Tree (MST) of graph G=(V,E)
is a sub-graph of G that:

1) is a tree
2) contains all nodes of G
3) Minimizes its total edge weight



MST

Total weight = 1 + 2 + 3 + 2 = 8



Not MST

Total weight = 1 + 3 + 4 + 6 = 14

Minimum Spanning Tree

Input: A connected, undirected graph G=(V,E),

where every edge $e \in E$ has a weight $w(e)$.

Output: Find an MST of G.

Quiz questions:

1. Given a graph, is its MST uniquely determined?

2. What are the applications of finding an MST in a graph?

Roadmap of this lecture:

1. Minimum Spanning Tree.

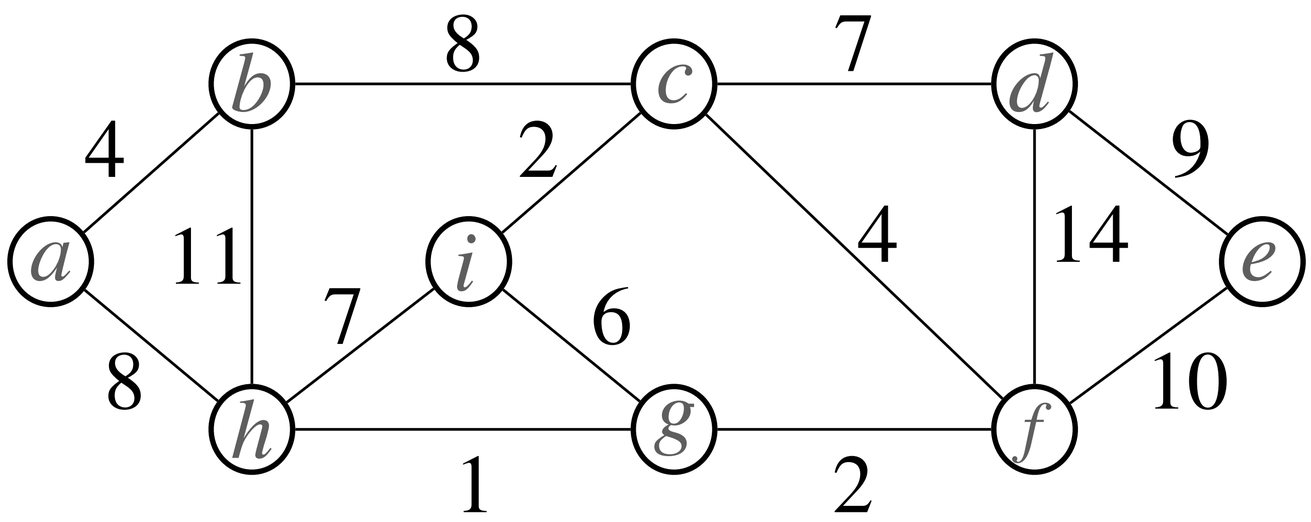
    1.1 Define "Minimum Spanning Tree (MST)".

    1.2 Kruskal's Algorithm for MST.

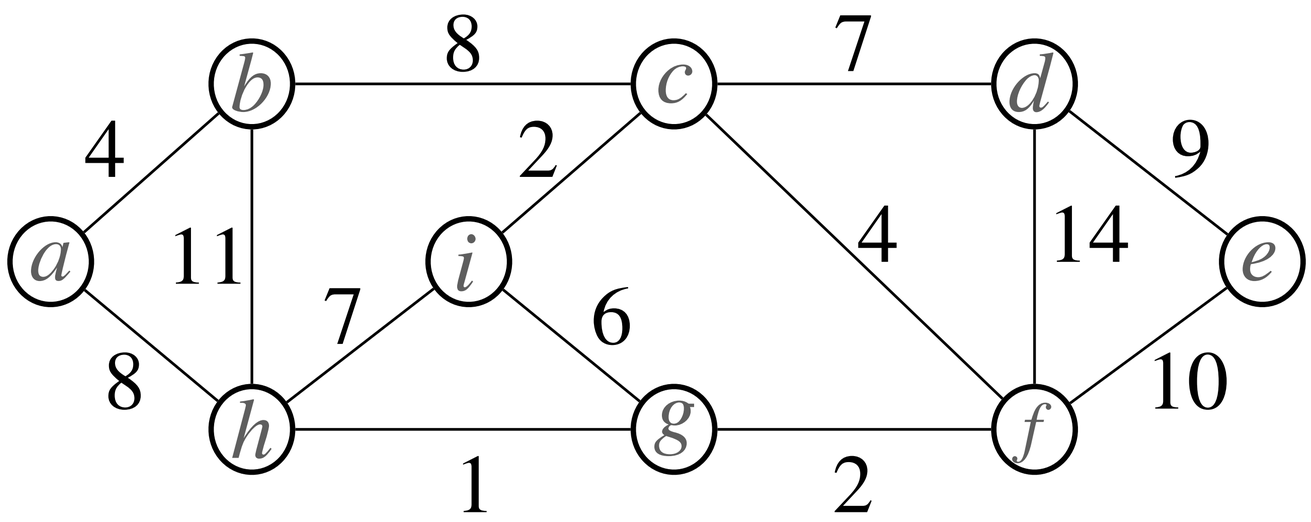
    1.3 Prim's Algorithm for MST.

    1.4 Correctness of Kruskal's Algorithm and Prim's Algorithm.

Minimum Spanning Tree



Minimum Spanning Tree

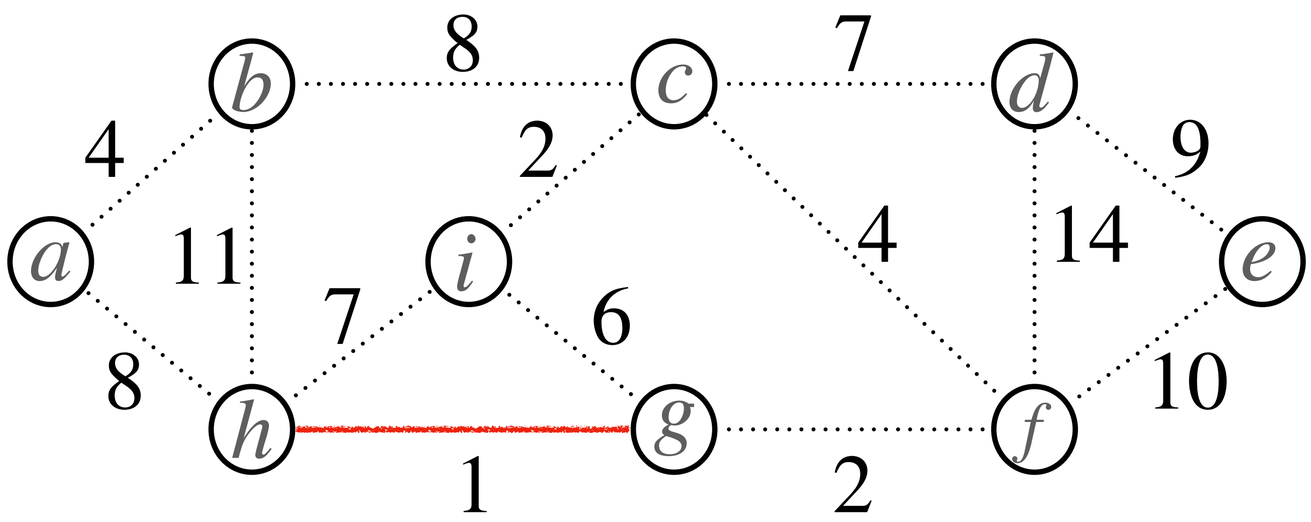


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree

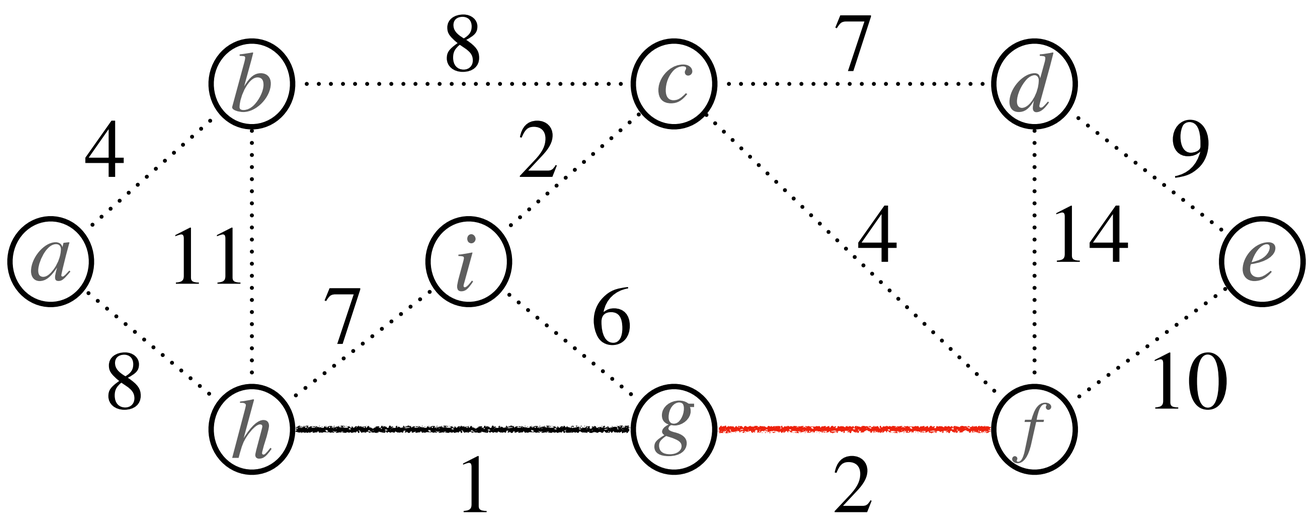


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree

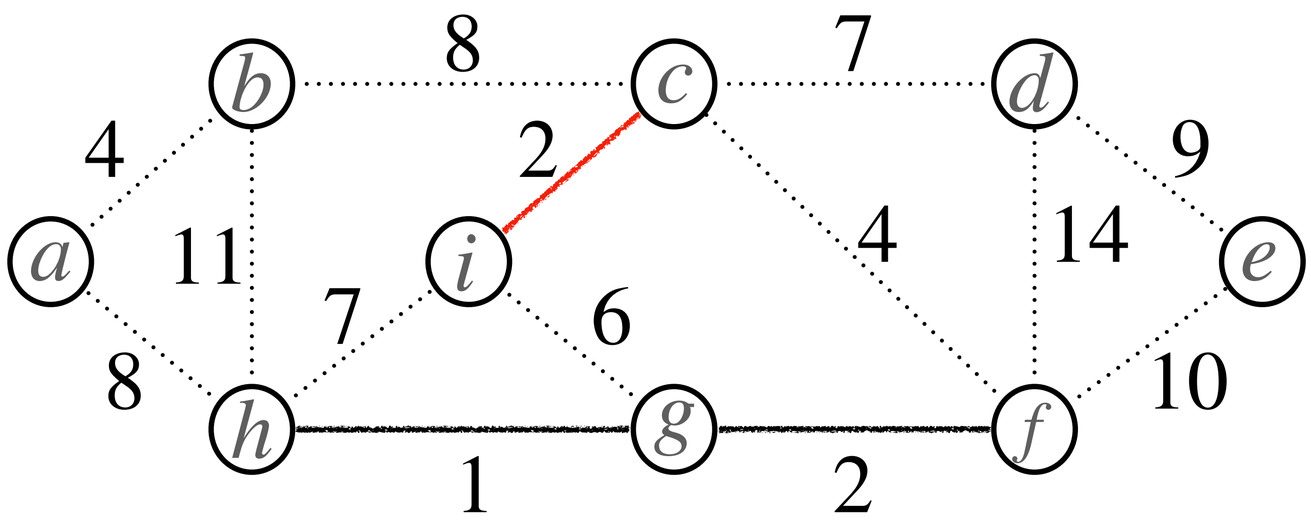


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree

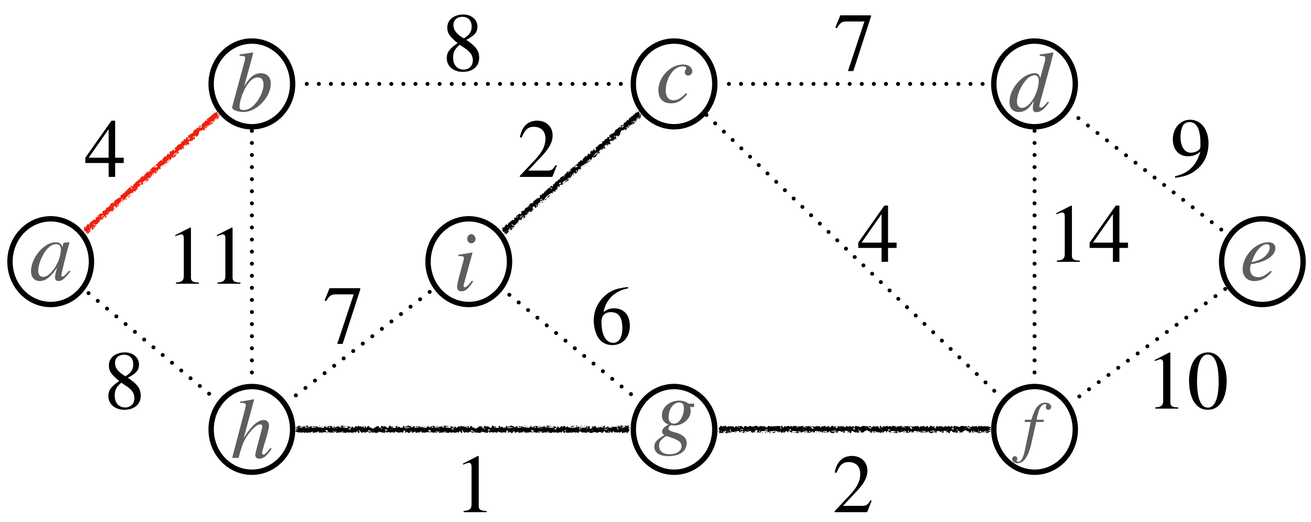


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree

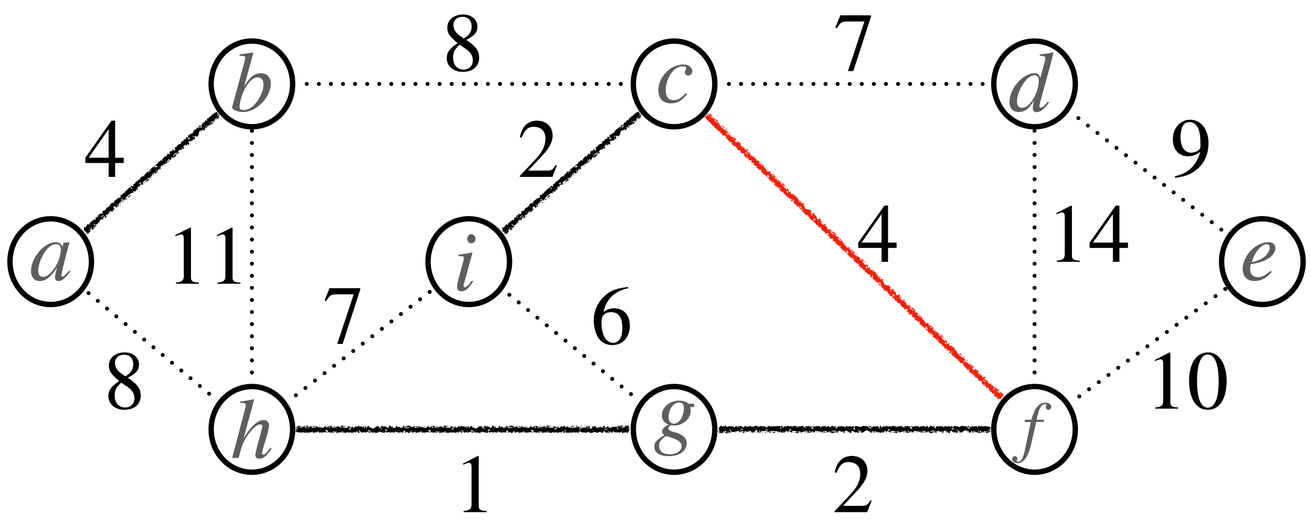


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree

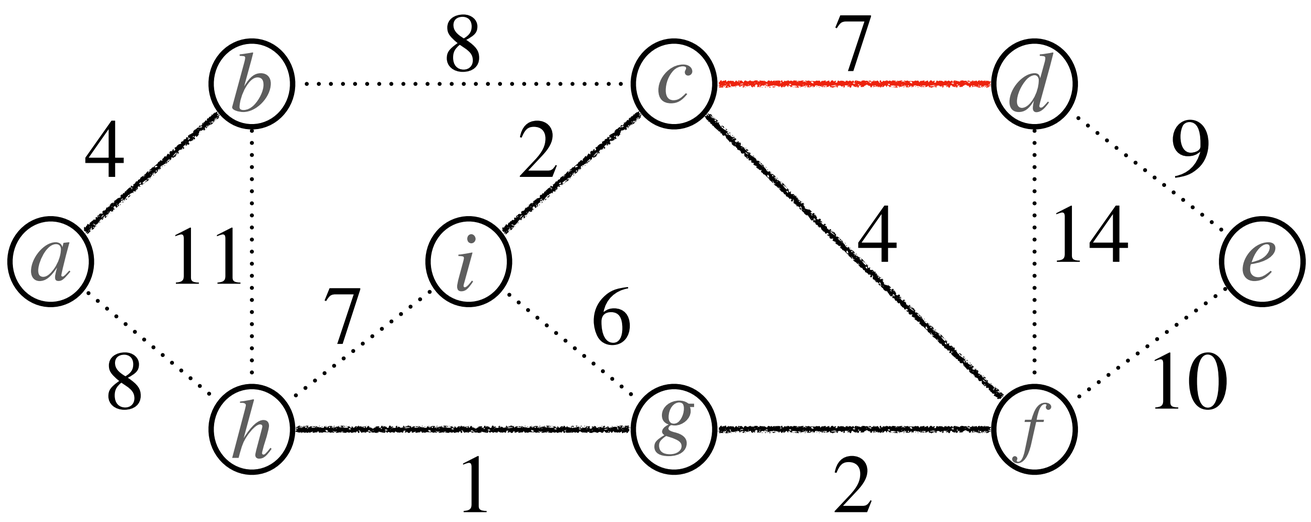


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree

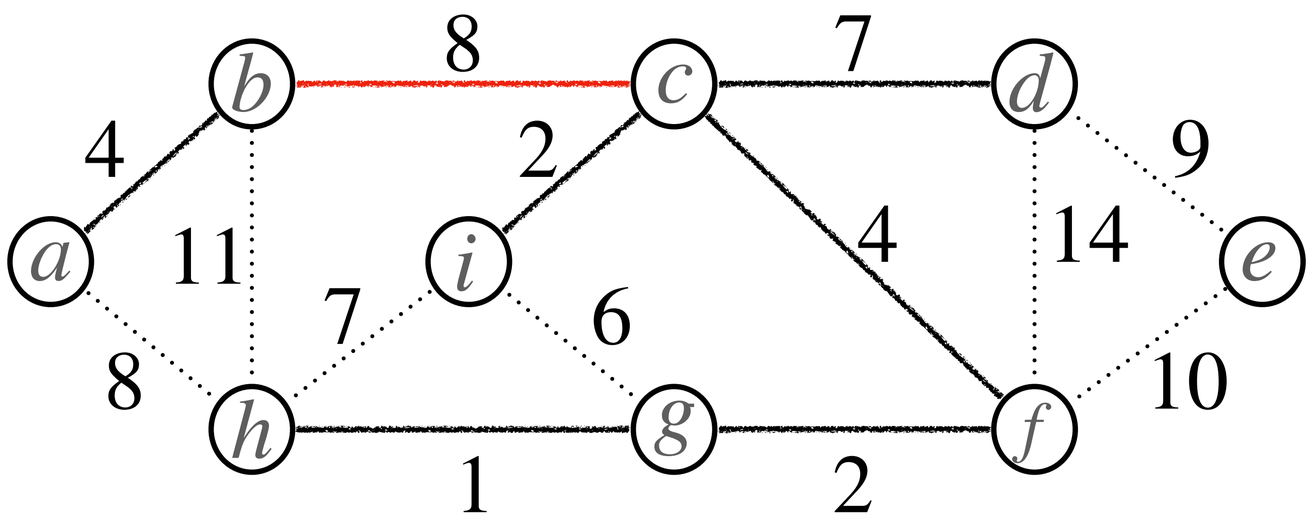


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree

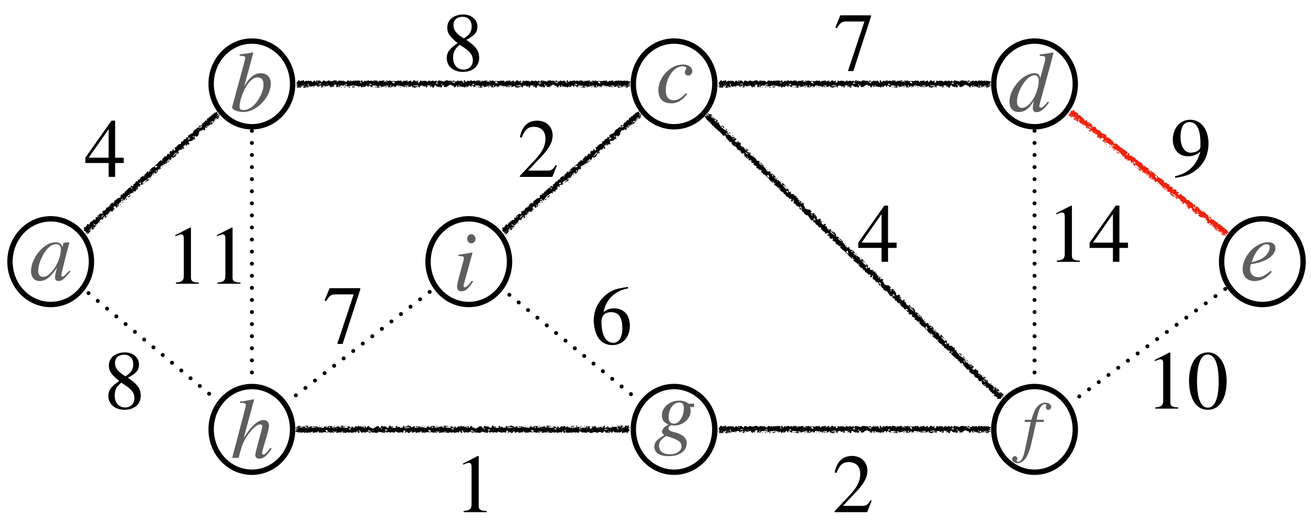


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree

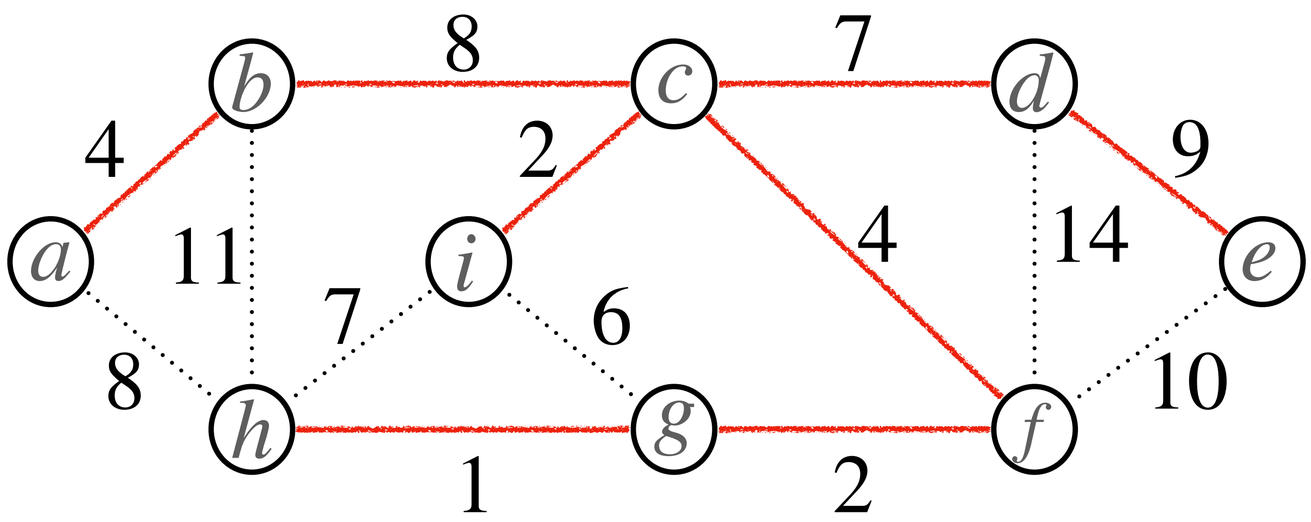


Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Minimum Spanning Tree



Weight = 1+2+2+4+4+7+8+9
= 37

Idea of Algorithm (Kruskal's Algorithm):

1. Pick edges one by one until they form a spanning tree.

2. Each time, we pick an edge of minimum weight that connects two nodes that have not been connected yet.

Quiz questions:

1. What is the main idea of Kruskal's algorithm?

2. Is Kruskal's algorithm a greedy algorithm?

Roadmap of this lecture:

1. Minimum Spanning Tree.

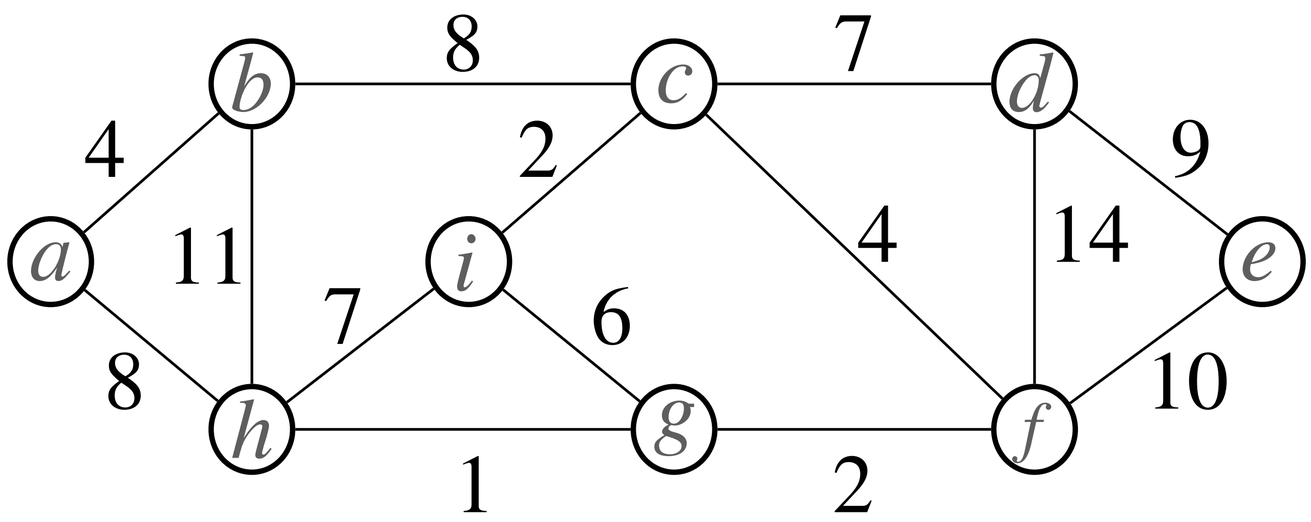
    1.1 Define "Minimum Spanning Tree (MST)".

    1.2 Kruskal's Algorithm for MST.

    1.3 Prim's Algorithm for MST.

    1.4 Correctness of Kruskal's Algorithm and Prim's Algorithm.

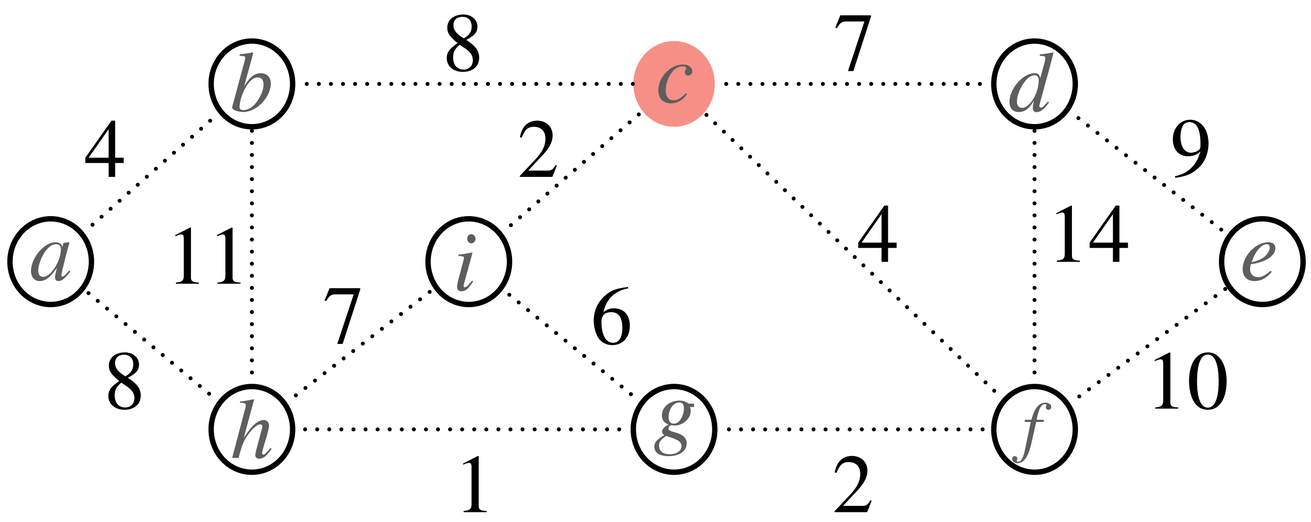
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

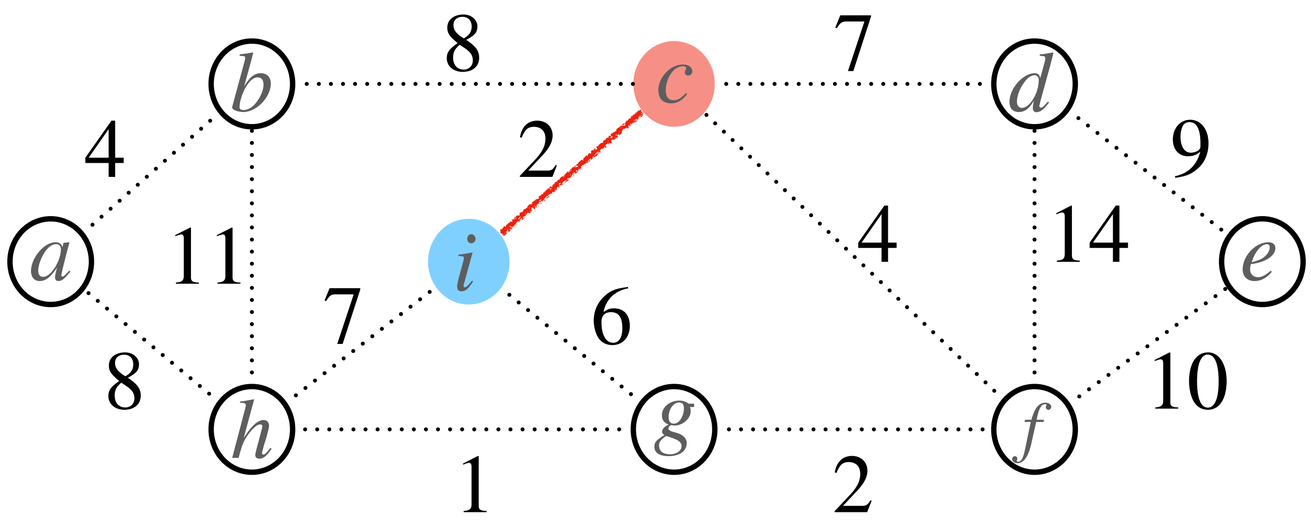
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

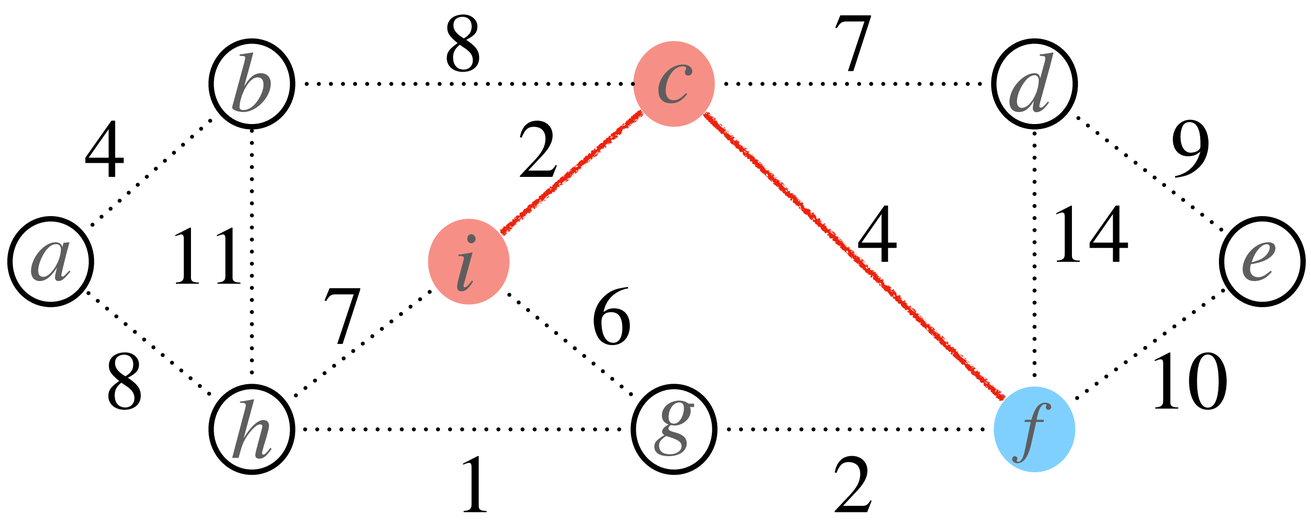
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

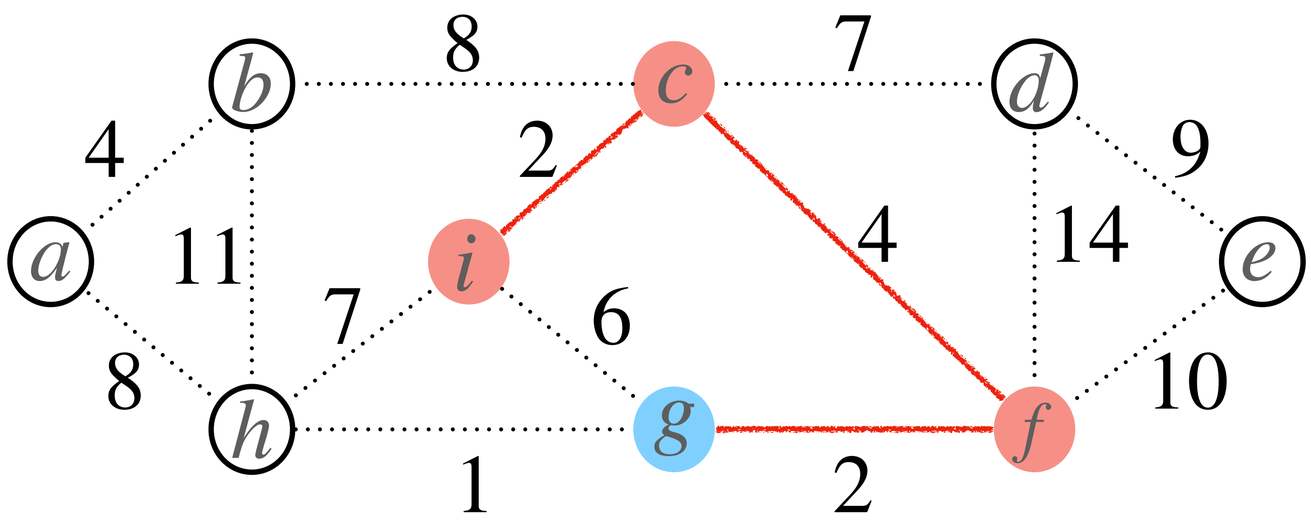
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

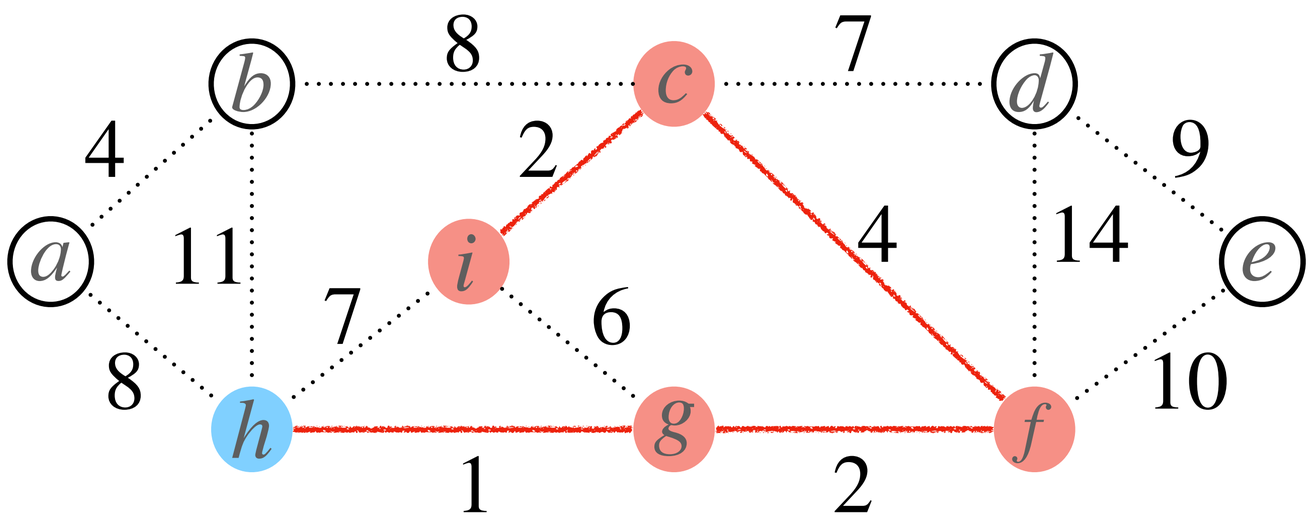
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

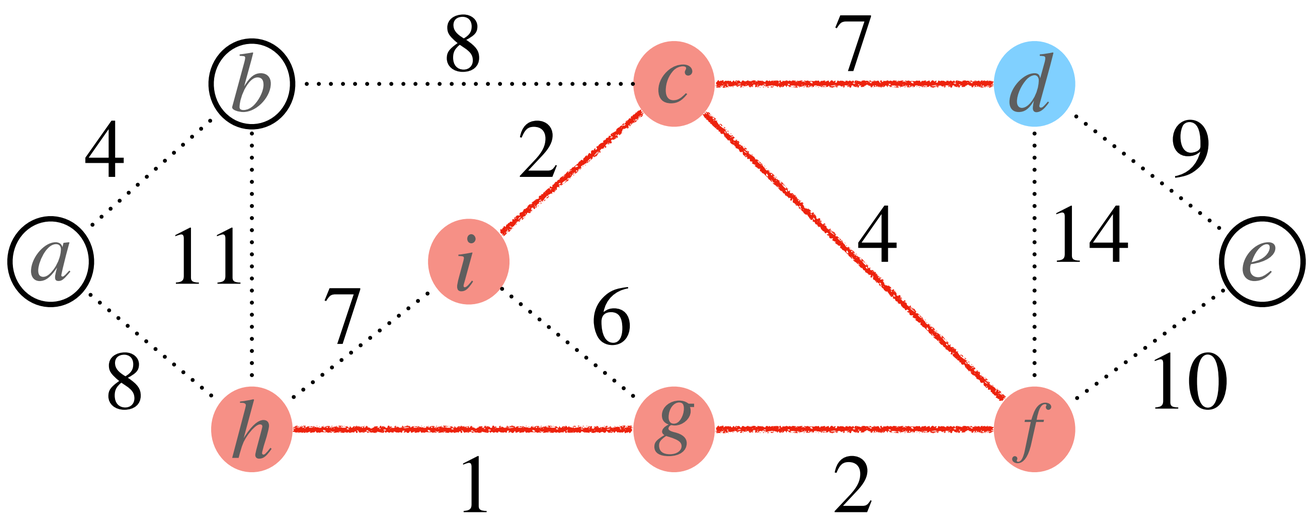
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

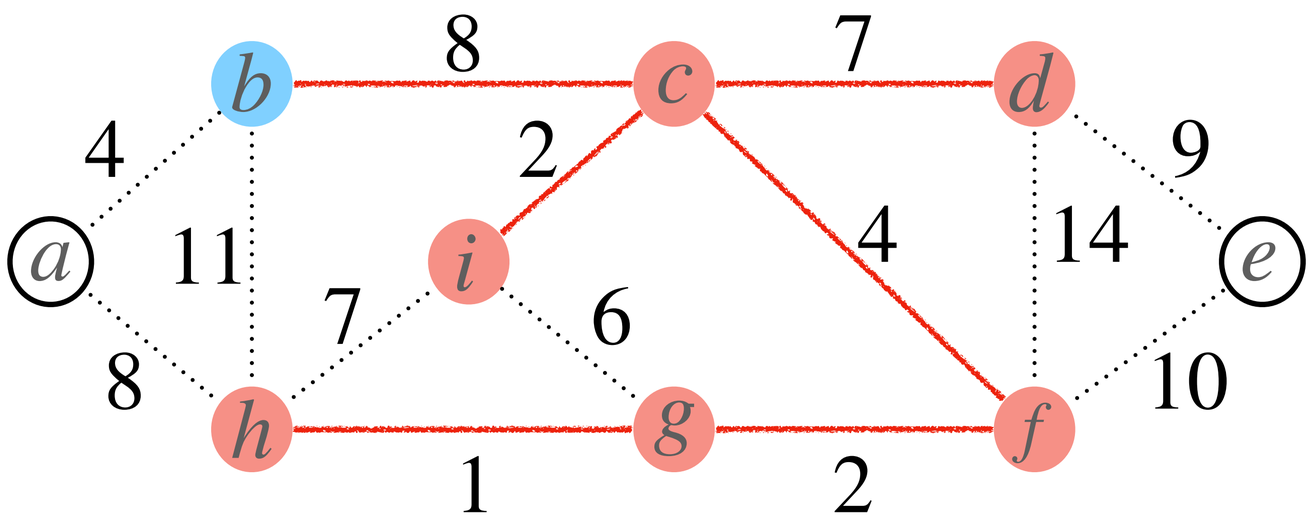
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

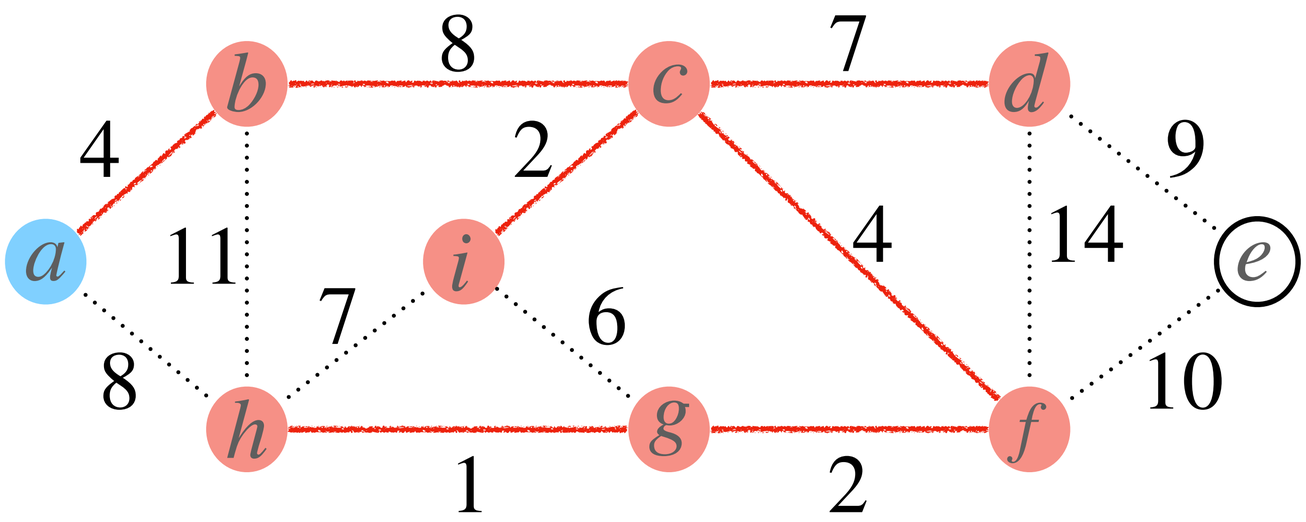
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

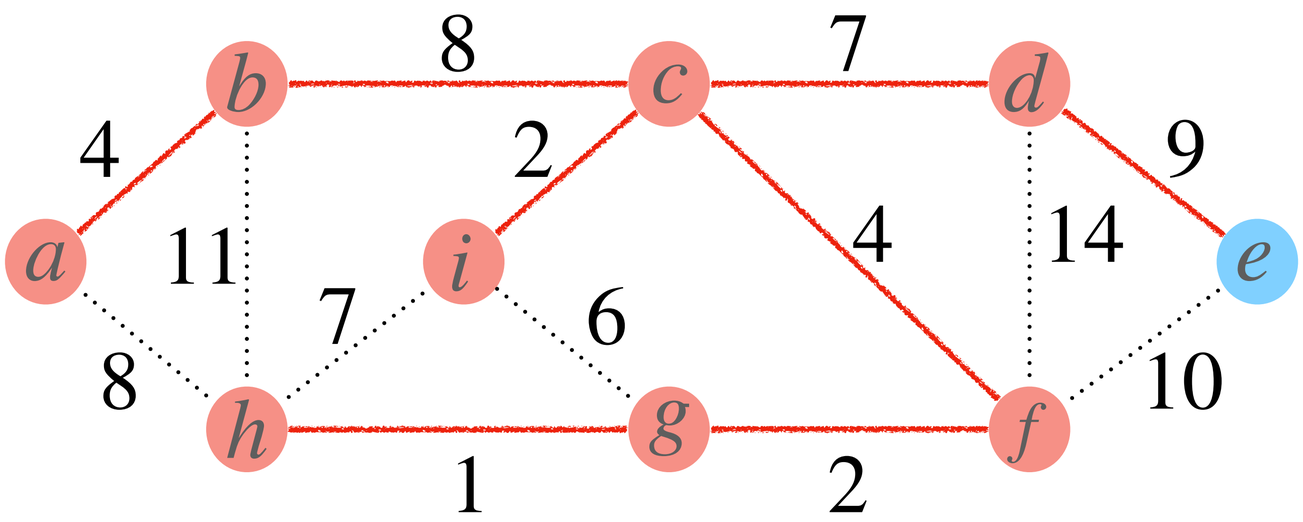
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

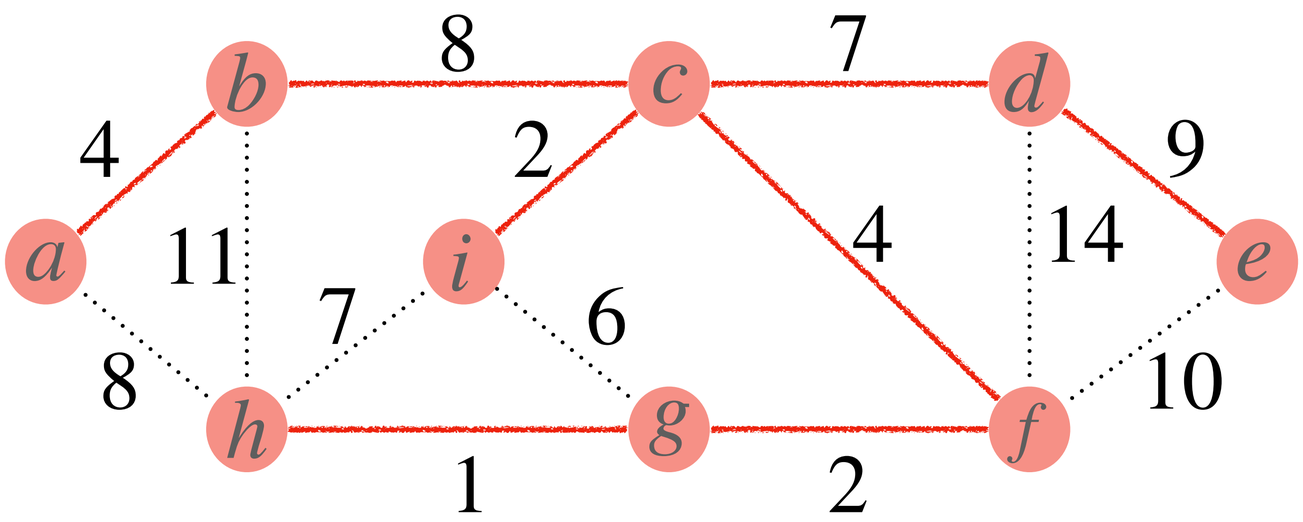
Minimum Spanning Tree



Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

Minimum Spanning Tree



weight = 4+8+2+7+4+2+1+9
=37

Idea of Algorithm (Prim's Algorithm):

1. Start with a root node, pick edges one by one until them form a spanning tree.
2. Each time, we pick an edge of minimum weight that connects the rooted tree to a node outside the tree.

Quiz questions:

1. What is the main idea of Prim's algorithm?

2. Is Prim's algorithm a greedy algorithm?

Roadmap of this lecture:

1. Minimum Spanning Tree.

    1.1 Define "Minimum Spanning Tree (MST)".

    1.2 Kruskal's Algorithm for MST.

    1.3 Prim's Algorithm for MST.

    1.4 Correctness of Kruskal's Algorithm and Prim's Algorithm.

Minimum Spanning Tree

Both Kruskal's Algorithm and Prim's Algorithm give optimal solutions (i.e, MST).
Why?

Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we pick an edge such that for all the edges we have chosen so far, there exists an MST that contains them.

Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we pick an edge such that for all the edges we have chosen so far, there exists an MST that contains them.

This is not saying much …

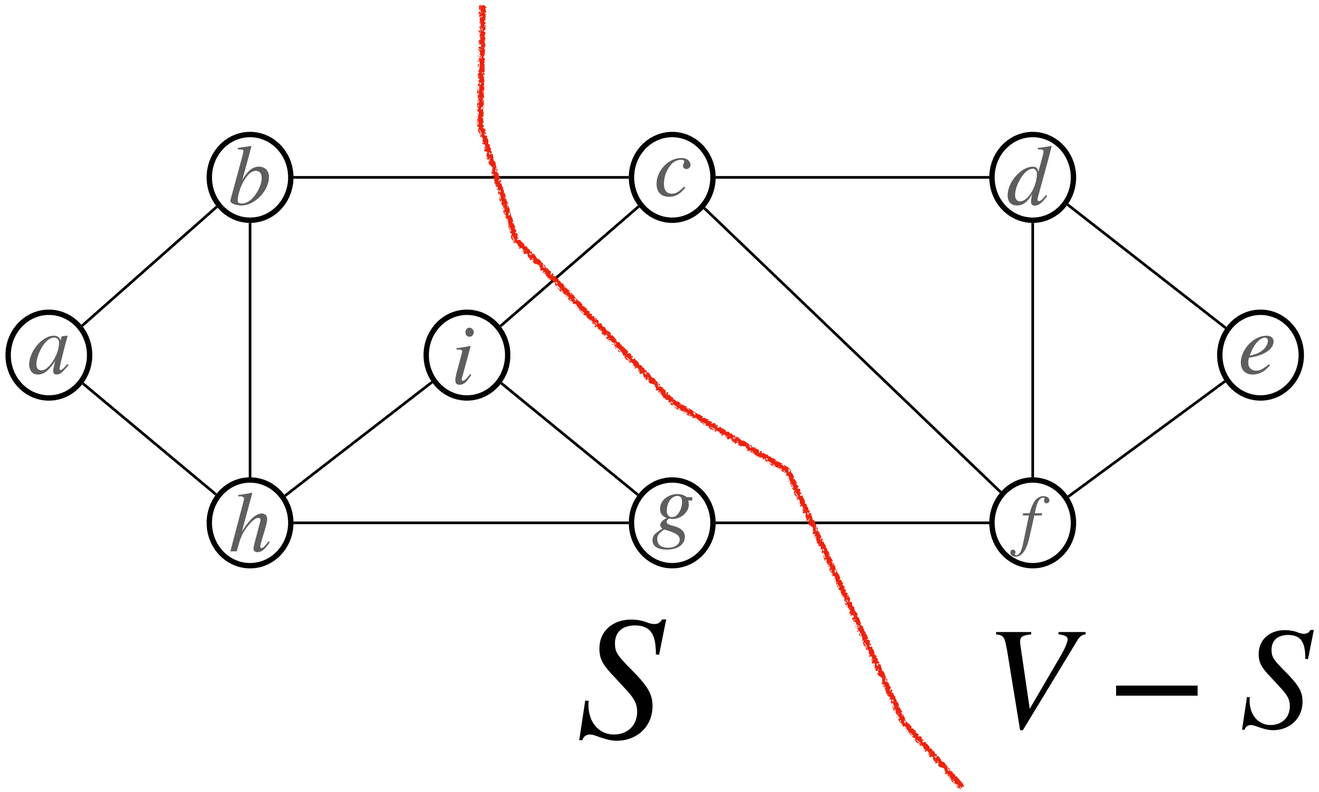
Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we pick an edge such that for all the edges we have chosen so far, there exists an MST that contains them.

Cut: A cut of an undirected graph G=(V,E) is a partition of its nodes V into two subsets S and V-S.



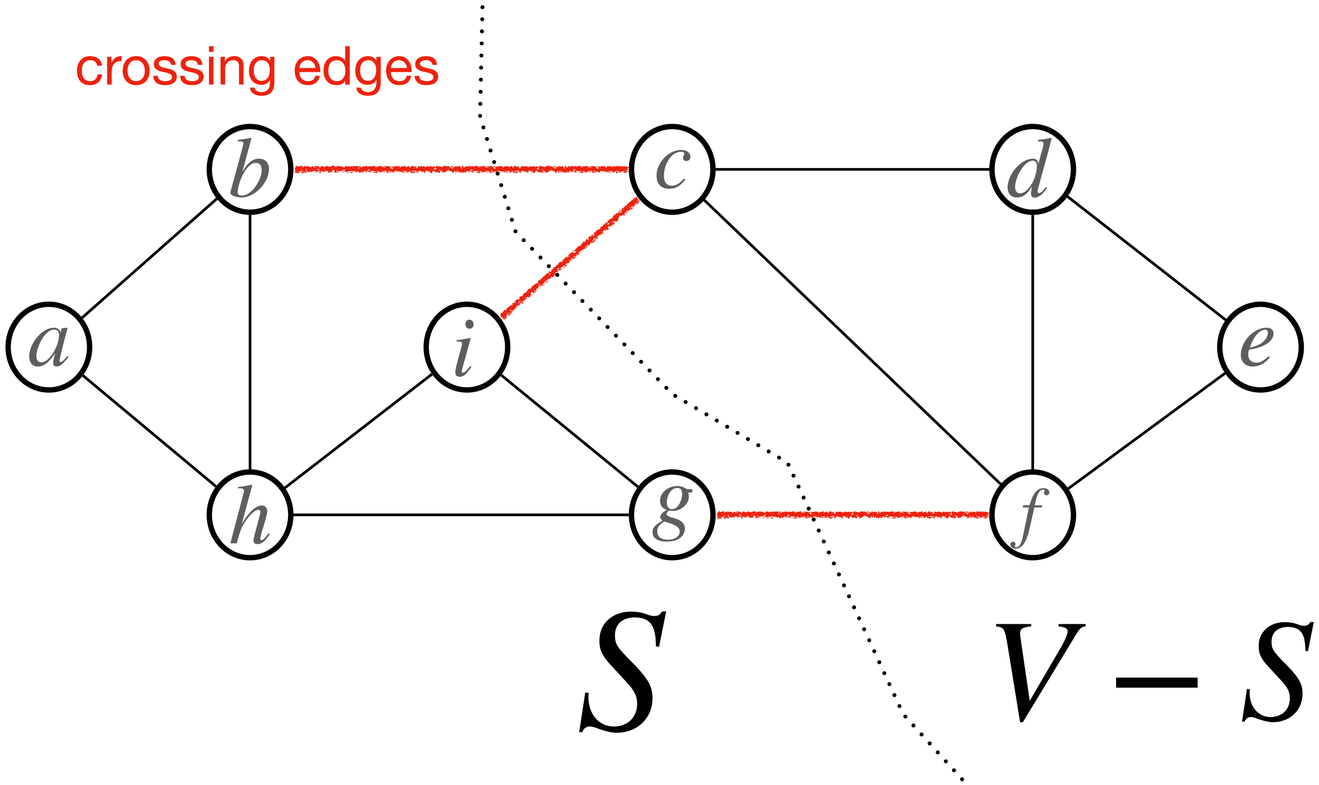
Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we pick an edge such that for all the edges we have chosen so far, there exists an MST that contains them.

Cut: A cut of an undirected graph G=(V,E) is a partition of its nodes V into two subsets S and V-S.



Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we pick an edge such that for all the edges we have chosen so far, there exists an MST that contains them.

Cut: A cut of an undirected graph G=(V,E) is a partition of its nodes V into two subsets S and V-S.

Respect: Let T be a subset of edges in G.

Let $(S, V - S)$ be a cut.

We say the cut respects T if no edge in T is a crossing edge.

Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

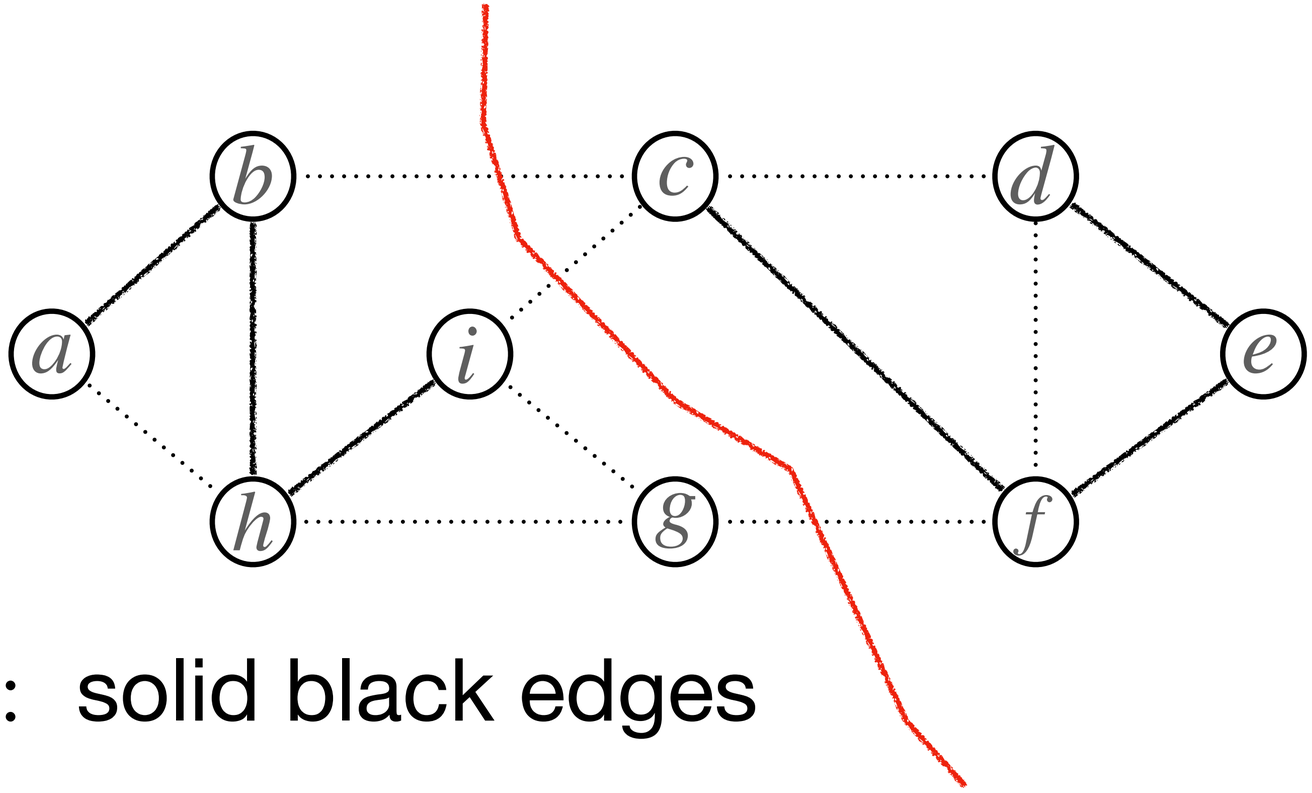
2) Each time, we pick an edge such that for all the edges we have chosen so far, there exists an MST that contains them.

Cut: A cut of an undirected graph G=(V,E) is a partition of its nodes V into two subsets S and V-S.

Respect: Let T be a subset of edges in G.

Let $(S, V-S)$ be a cut.

We say the cut respects T if no edge in T is a crossing edge.



T : solid black edges

Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

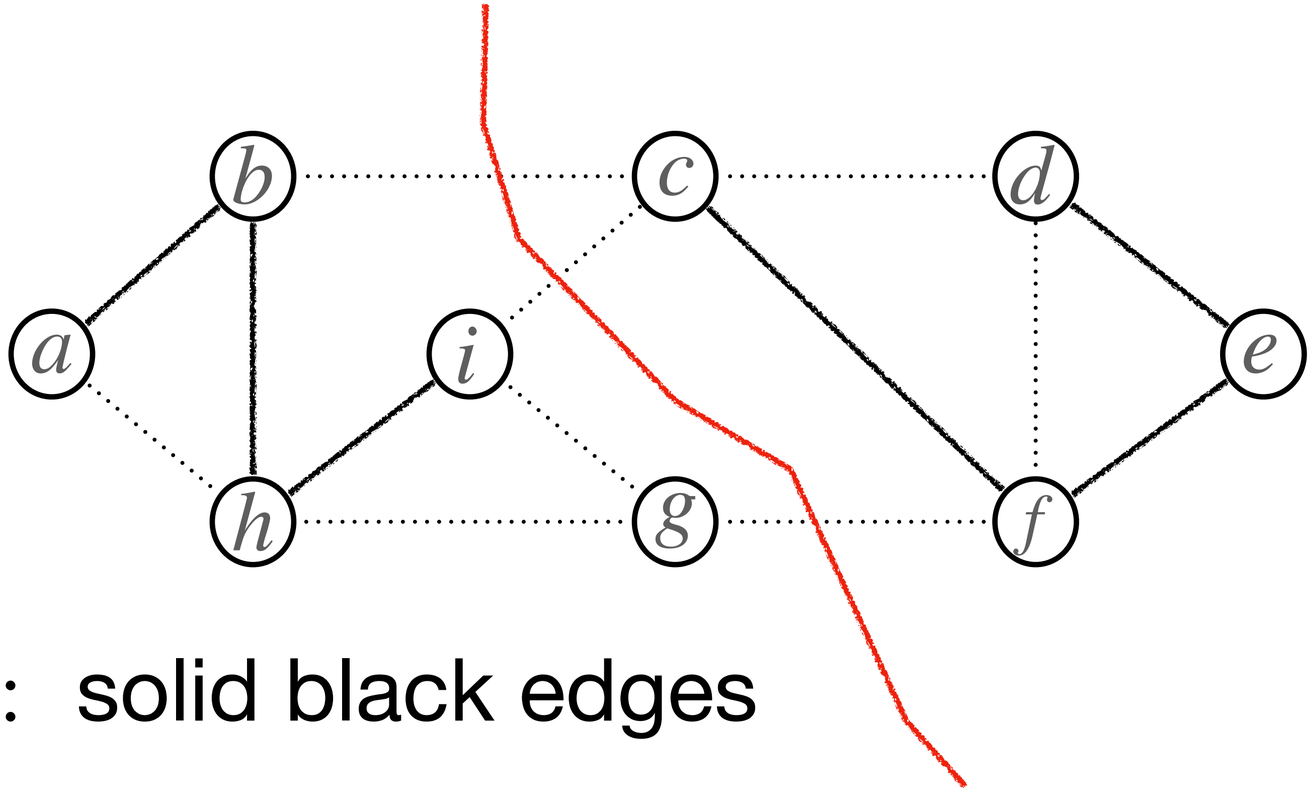
2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Cut: A cut of an undirected graph G=(V,E) is a partition of its nodes V into two subsets S and V-S.

Respect: Let T be a subset of edges in G.

Let $(S, V-S)$ be a cut.

We say the cut respects T if no edge in T is a crossing edge.



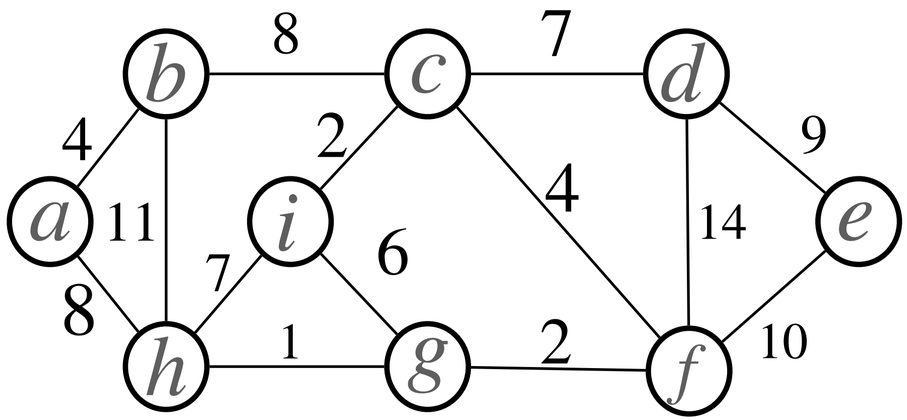
T : solid black edges

Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.



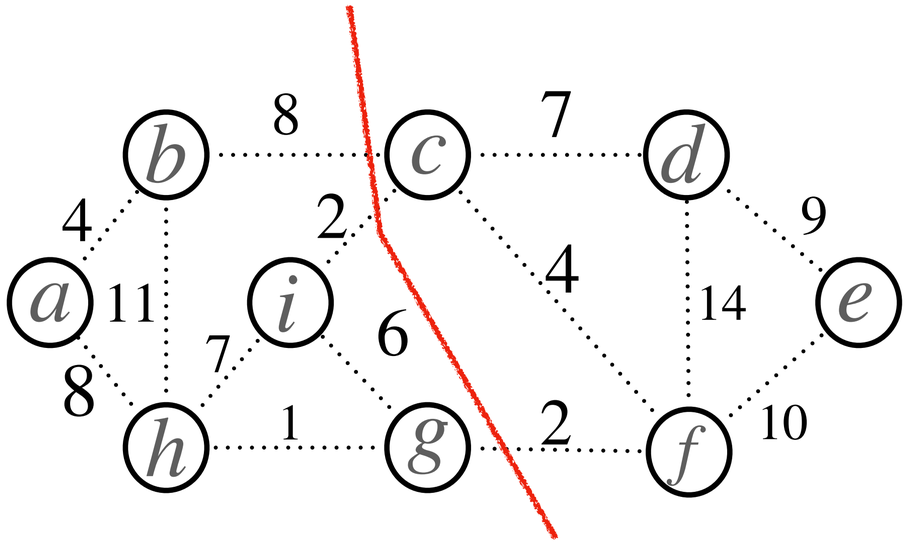
This is better…

Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

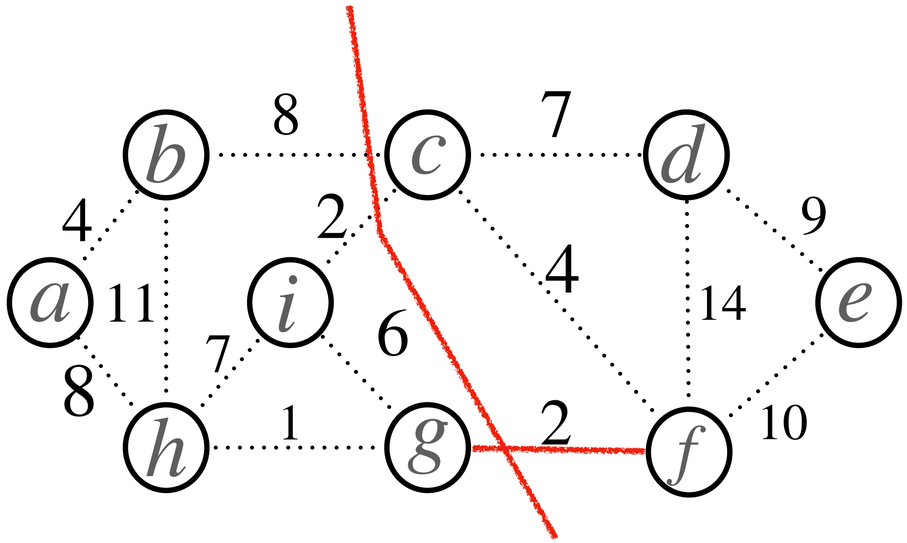


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

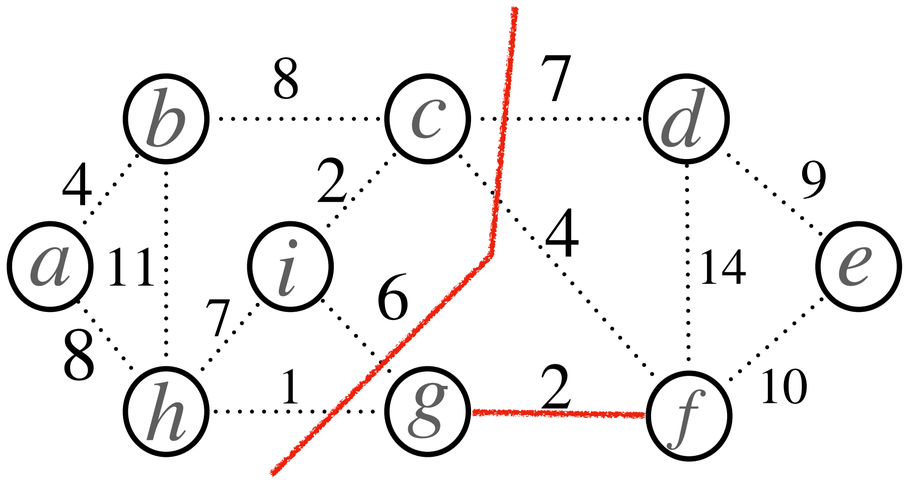


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

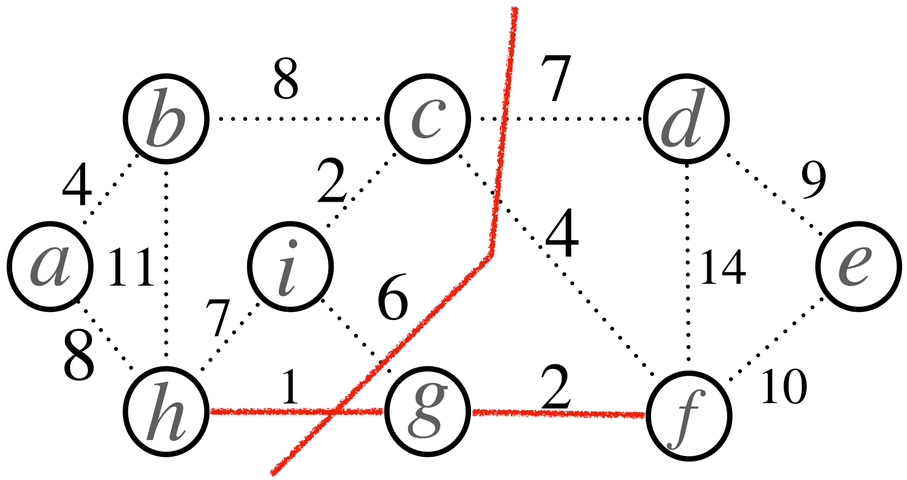


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

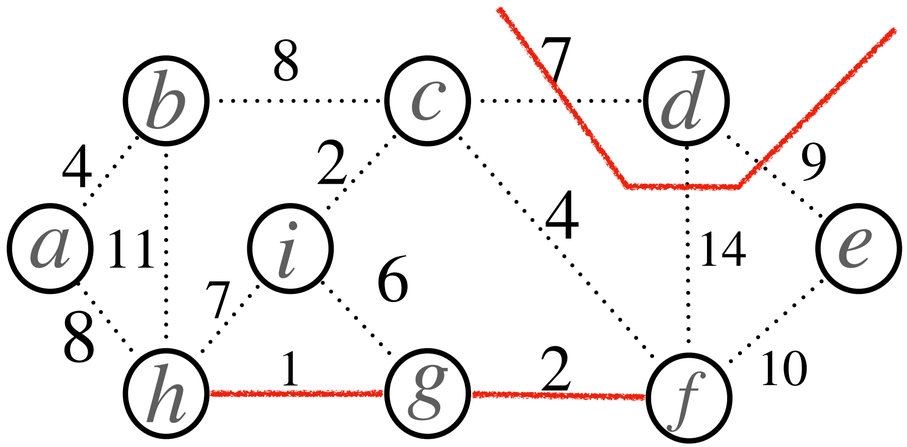


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

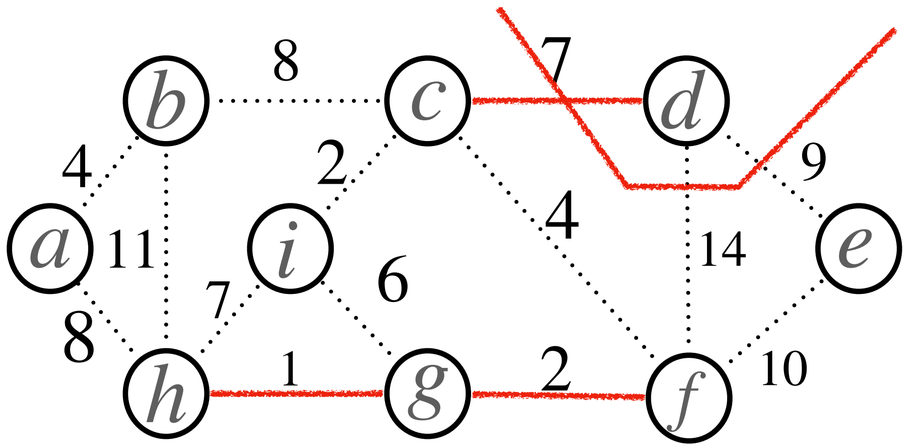


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

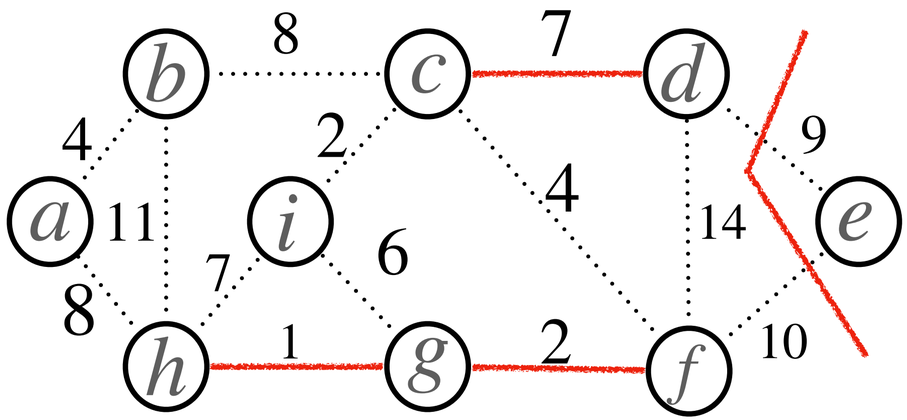


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

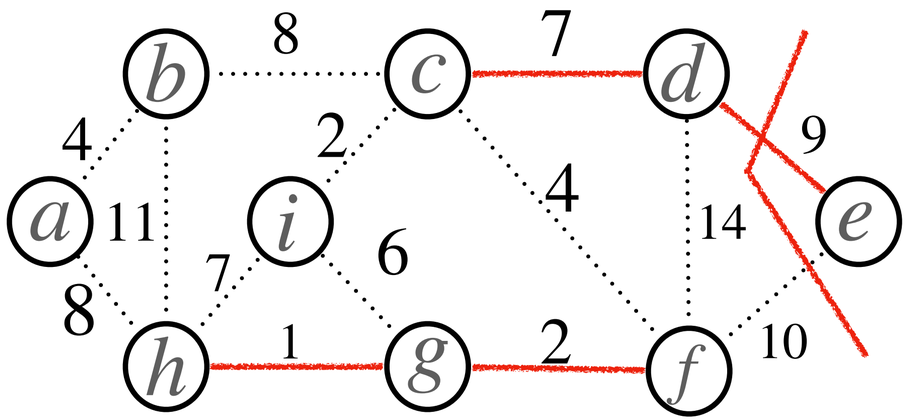


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

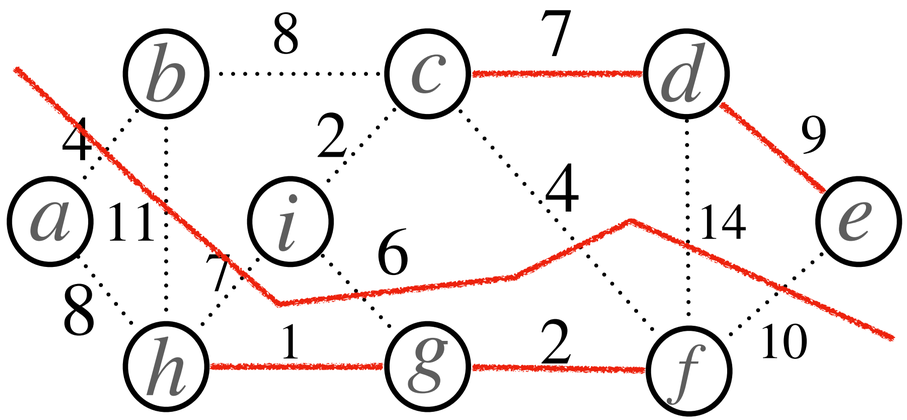


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

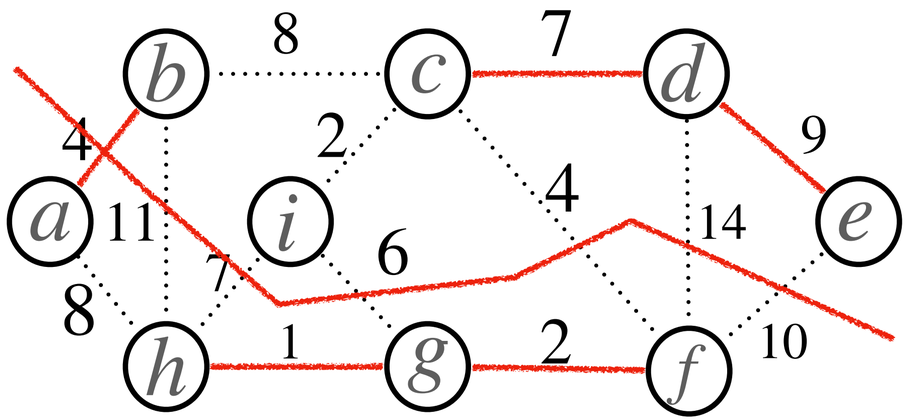


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

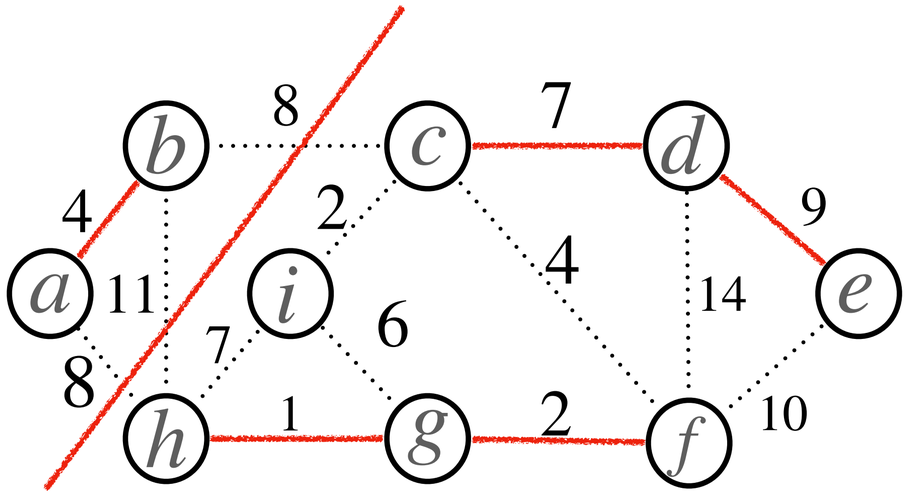


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

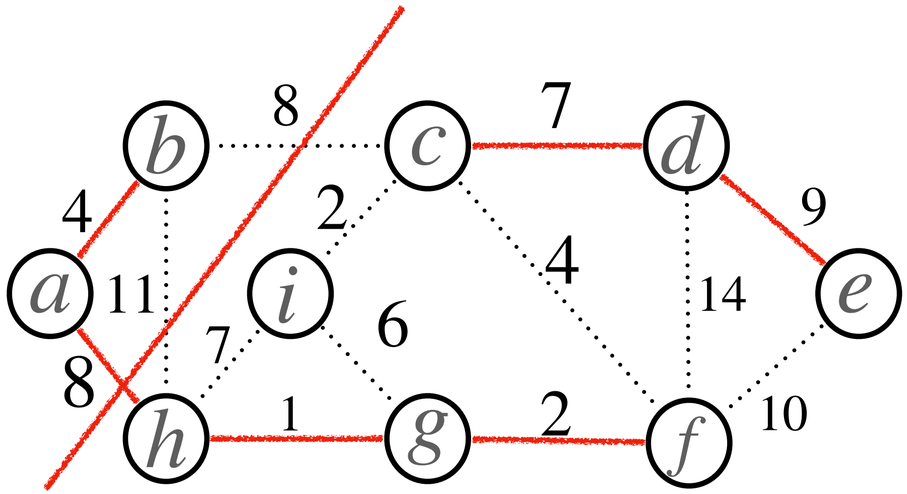


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

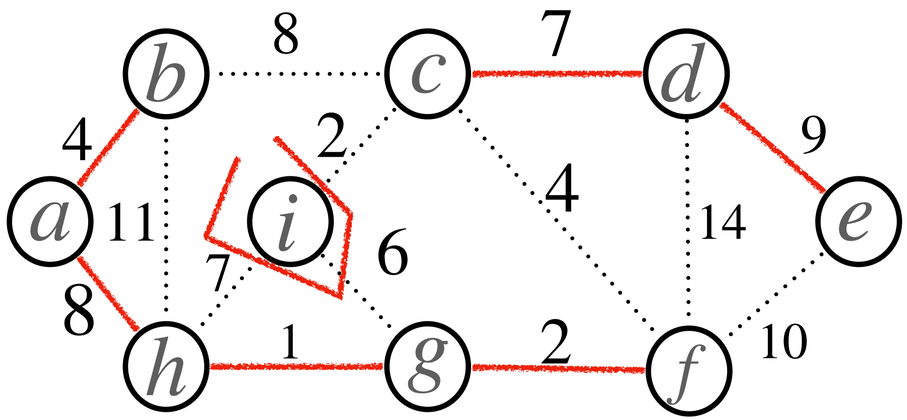


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

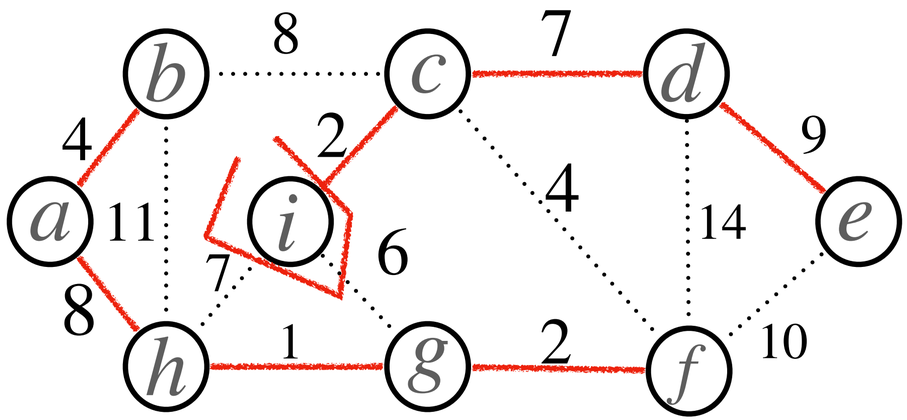


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

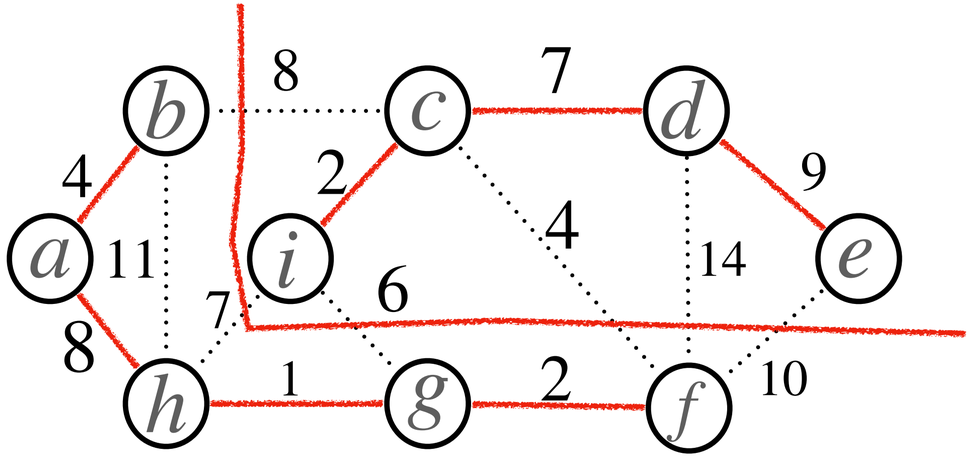


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

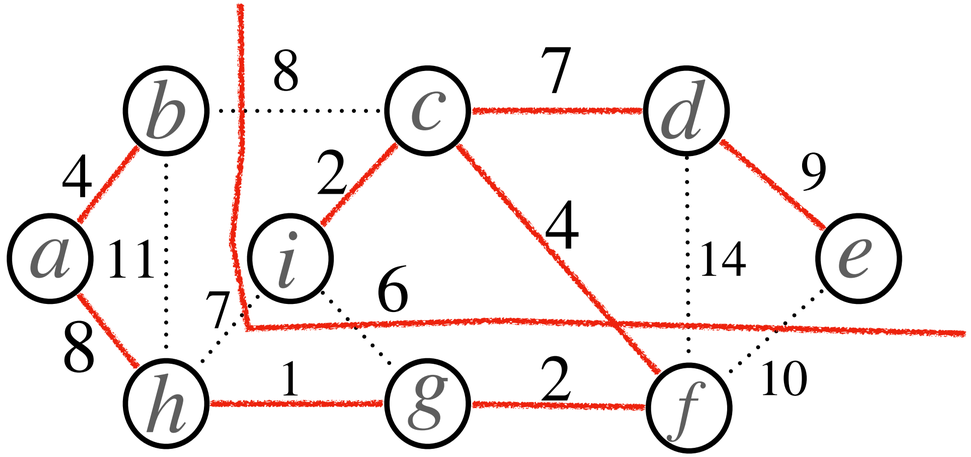


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

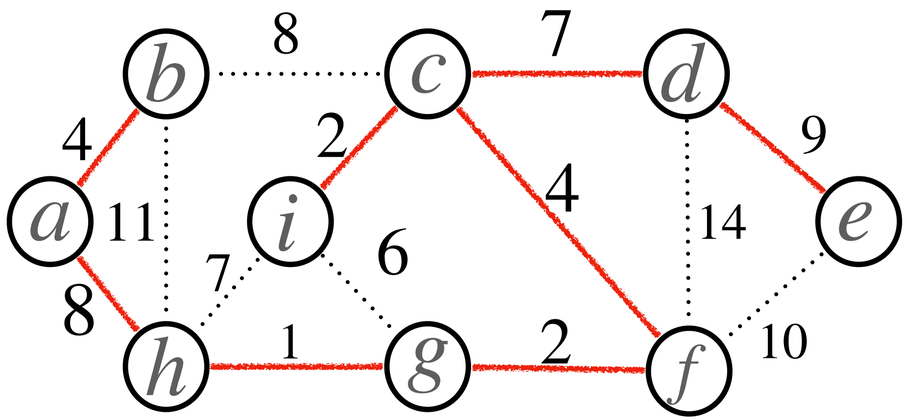


Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.



Weight = 4+8+1+2+4+2+7+9 = 37

Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

Minimum Spanning Tree

A more general algorithm:

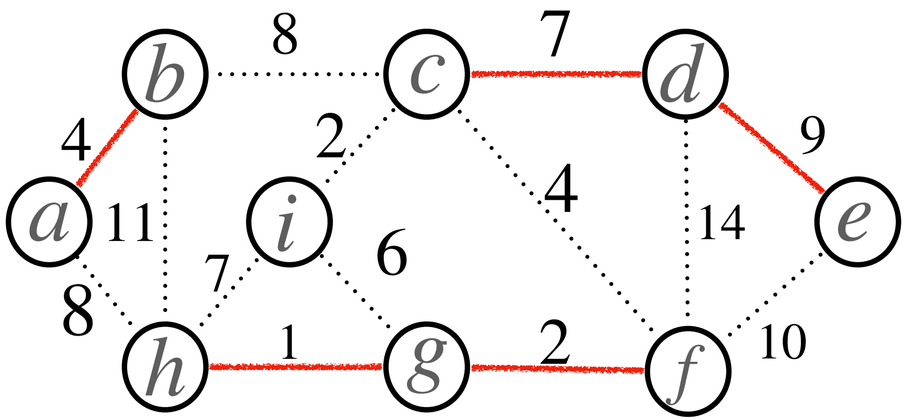
1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

Proof:

Red edges: edges chosen so far.



Minimum Spanning Tree

A more general algorithm:

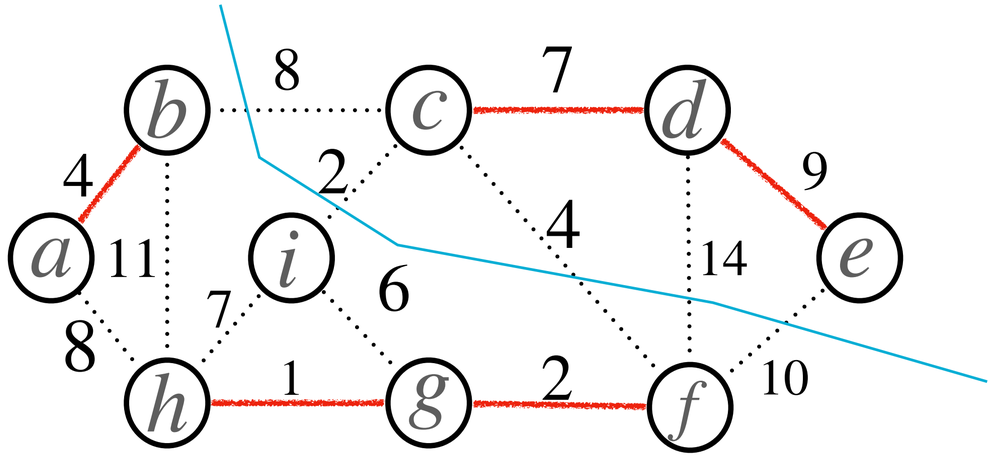
1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

Proof:

A cut that respects the chosen edges.



Minimum Spanning Tree

A more general algorithm:

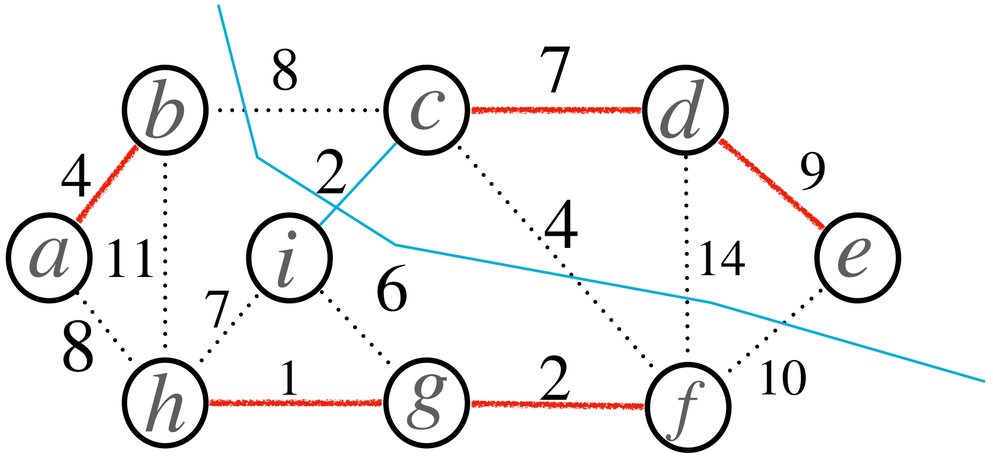
1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

Proof:

The minimum-weight crossing edge



Minimum Spanning Tree

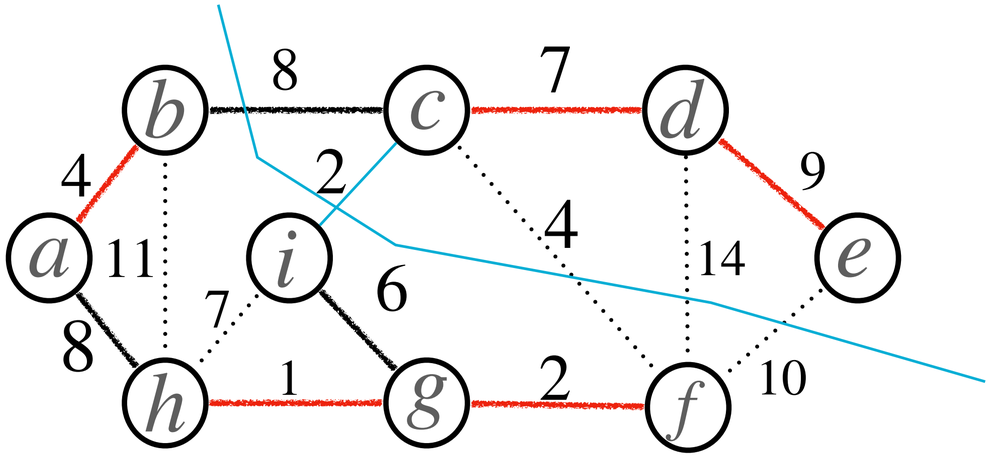
A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

Proof: Consider an MST that contains the chosen edges. Assume it does not contain the blue edge.



Minimum Spanning Tree

A more general algorithm:

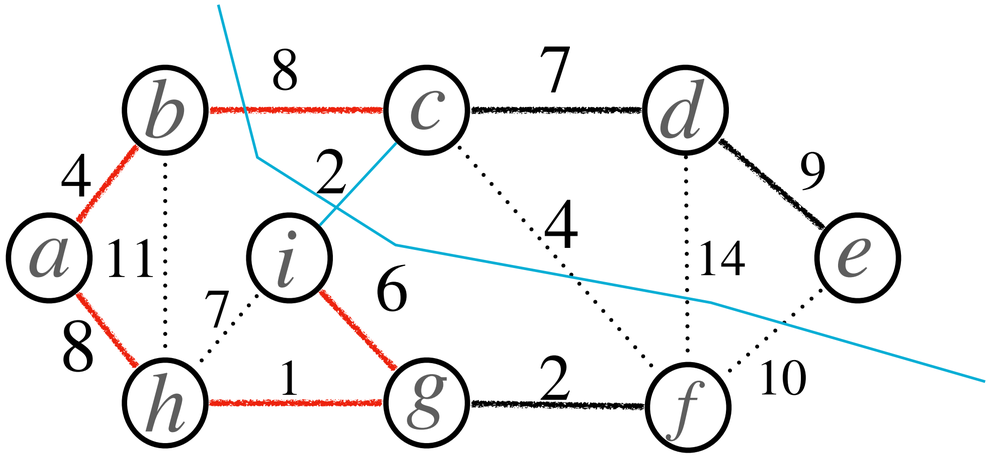
1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

Proof:

There is a path in MST connecting i and c



Minimum Spanning Tree

A more general algorithm:

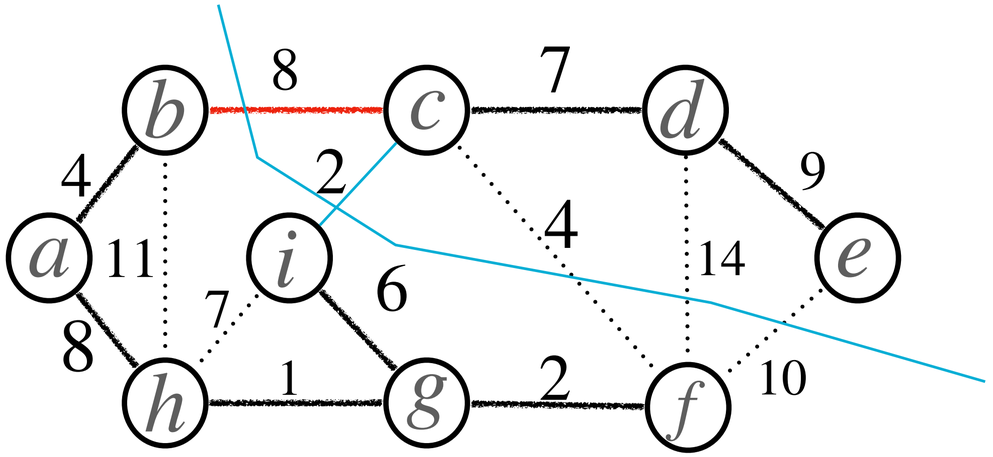
1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

Proof:

This path must contain a crossing edge



Minimum Spanning Tree

A more general algorithm:

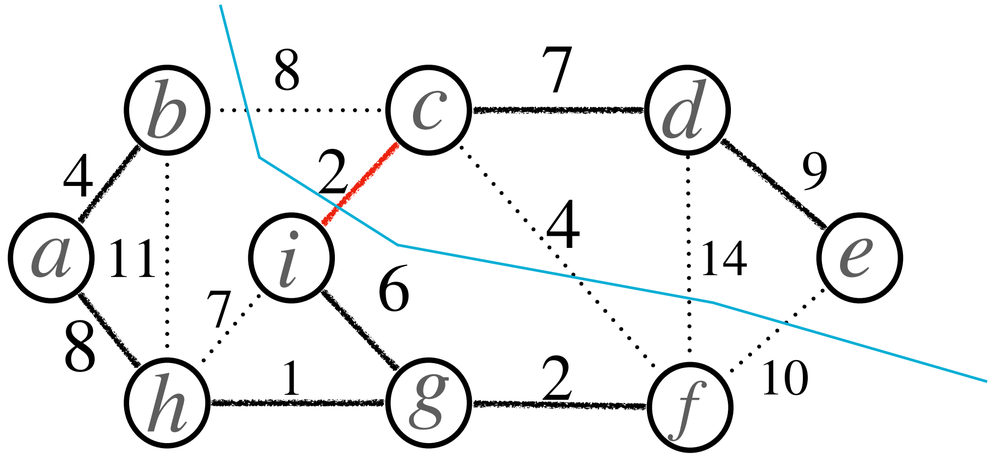
1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

Proof:

Replace it by the minimum-weight crossing edge, we get a better MST (but that's a contradiction).



Minimum Spanning Tree

A more general algorithm:

1) Pick edges one by one until they form a spanning tree.

2) Each time, we select a cut that respects the edges chosen so far. Then, among the crossing edges, we pick the edge of the minimum weight.

This is GREAT…

Lemma: Assume there exists an MST that contains all the edges chosen so far. Then, given a cut that respects those chosen edges, there also exists an MST that contains not only the chosen edges, but also the minimum-weight crossing edge for the cut.

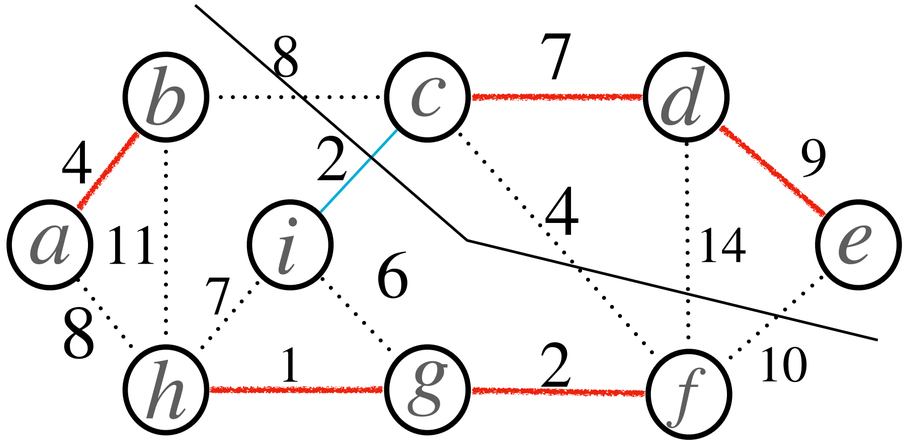
This "more general algorithm" must produce an MST.

But why is it more general than Kruskal's algorithm and Prim's algorithm?
(If that is the case, then Kruskal's algorithm and Prim's algorithm also produce MST.)

Minimum Spanning Tree

Kruskal's Algorithm:

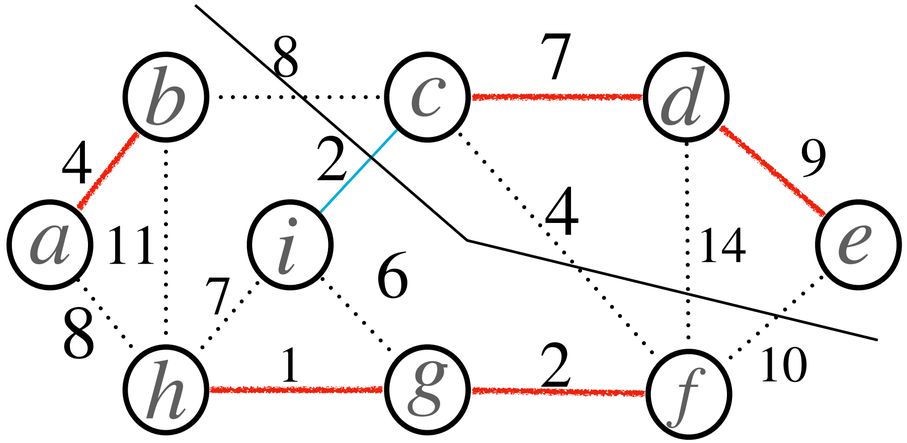
Select the cut that minimizes the weight of the minimum-weight crossing edge.



Minimum Spanning Tree

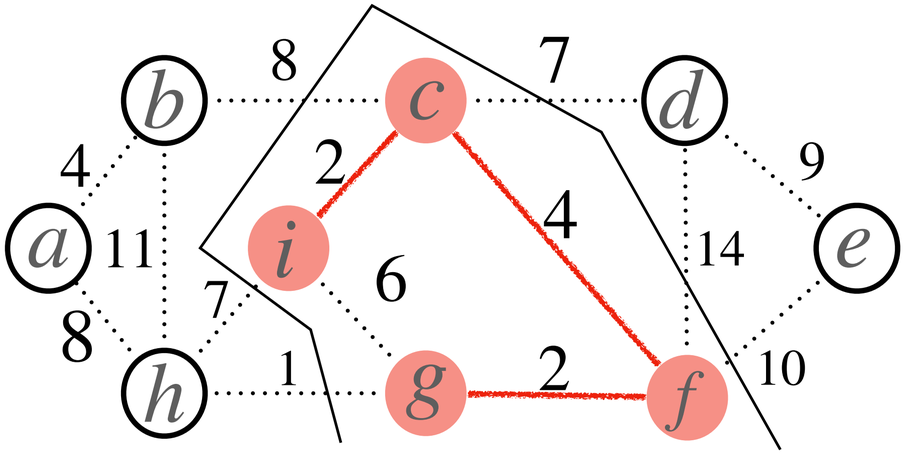
Kruskal's Algorithm:

Select the cut that minimizes the weight of the minimum-weight crossing edge.



Prim's Algorithm:

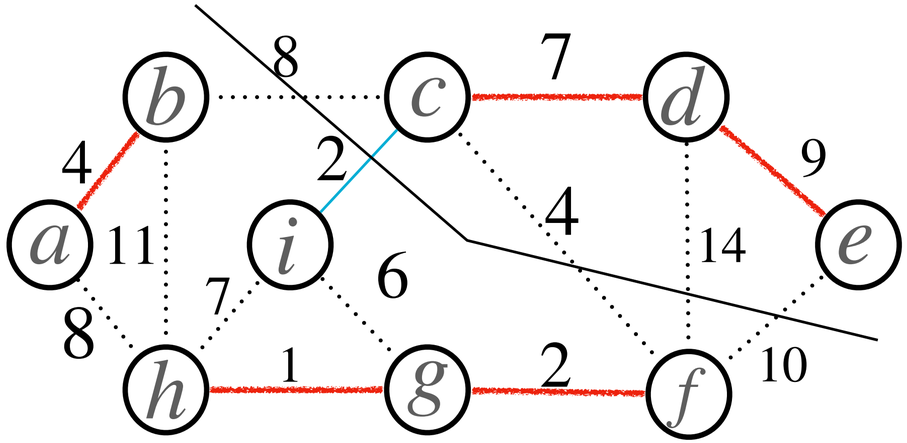
Select the cut that separates the connected rooted tree from the remaining nodes.



Minimum Spanning Tree

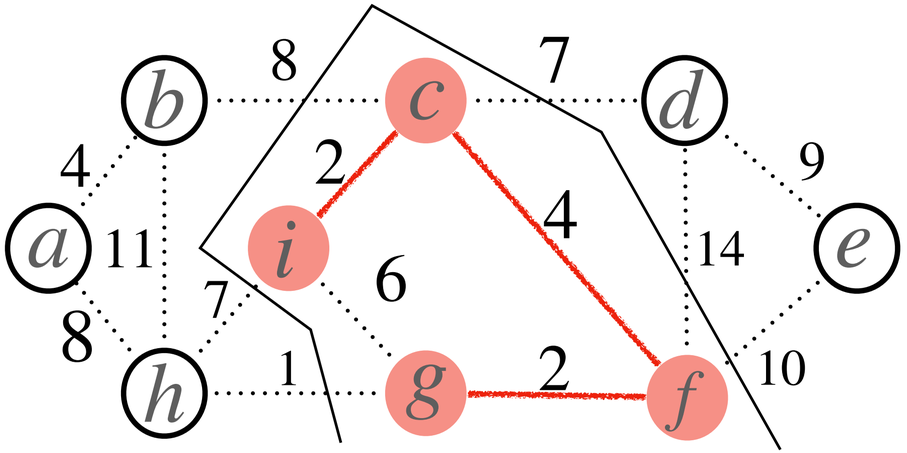
Kruskal's Algorithm:

Select the cut that minimizes the weight of the minimum-weight crossing edge.



Prim's Algorithm:

Select the cut that separates the connected rooted tree from the remaining nodes.



Then why use the above two algorithms instead of the "more general algorithm"?
Lower time complexity can be achieved when the algorithm is more specific.

Quiz questions:

1. What is the main idea of the more general algorithm for finding an MST?

2. Why are Kruskal's Algorithm and Prim's Algorithm both special cases of the more general algorithm?