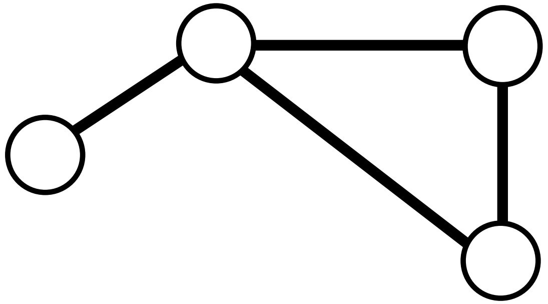
Minimum Spanning Trees

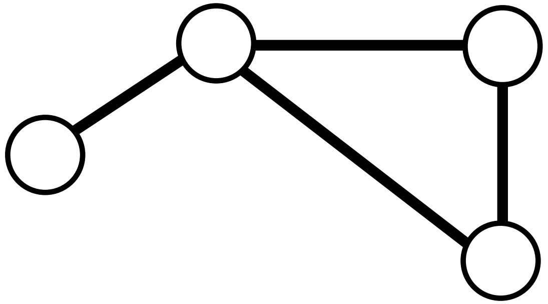
CSCI 432

Minimum Spanning Tree (MST)



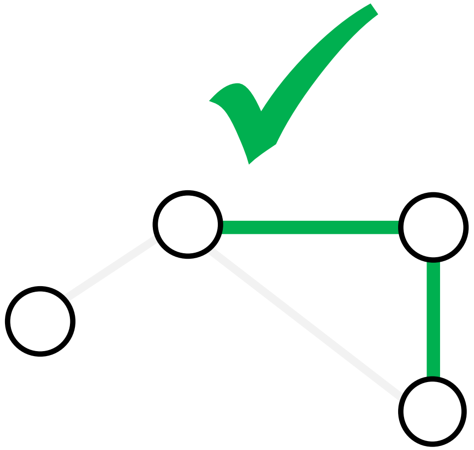
Given a connected graph, a subset of edges is a…

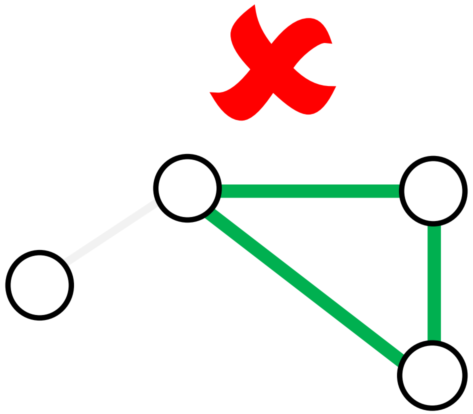
Minimum Spanning Tree (MST)



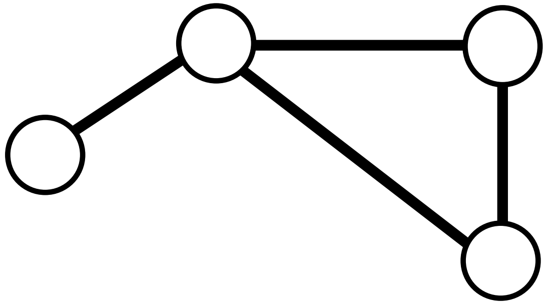
Given a connected graph, a subset of edges is a…

Tree if it is connected and acyclic.

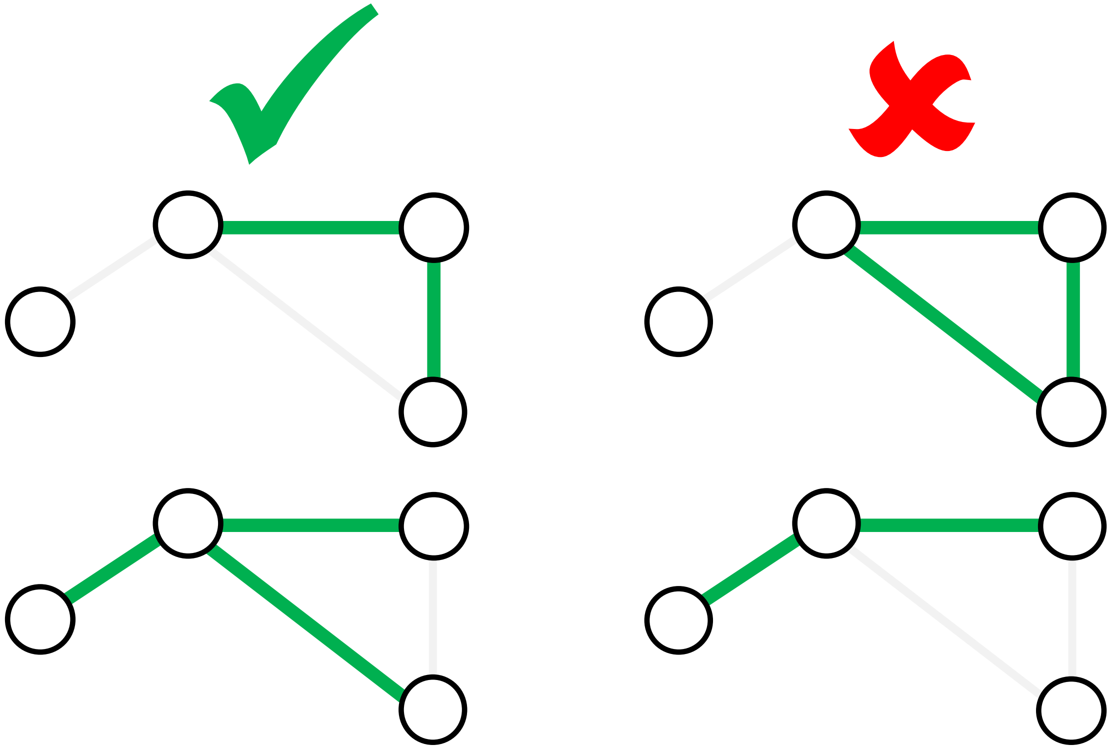




Minimum Spanning Tree (MST)



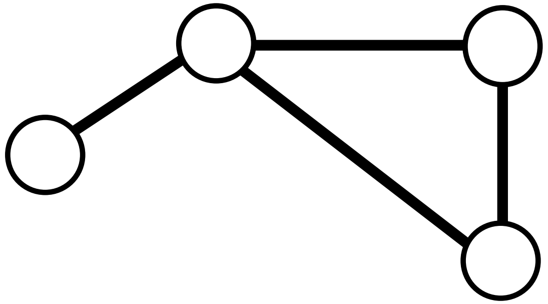
Given a connected graph, a subset of edges is a…



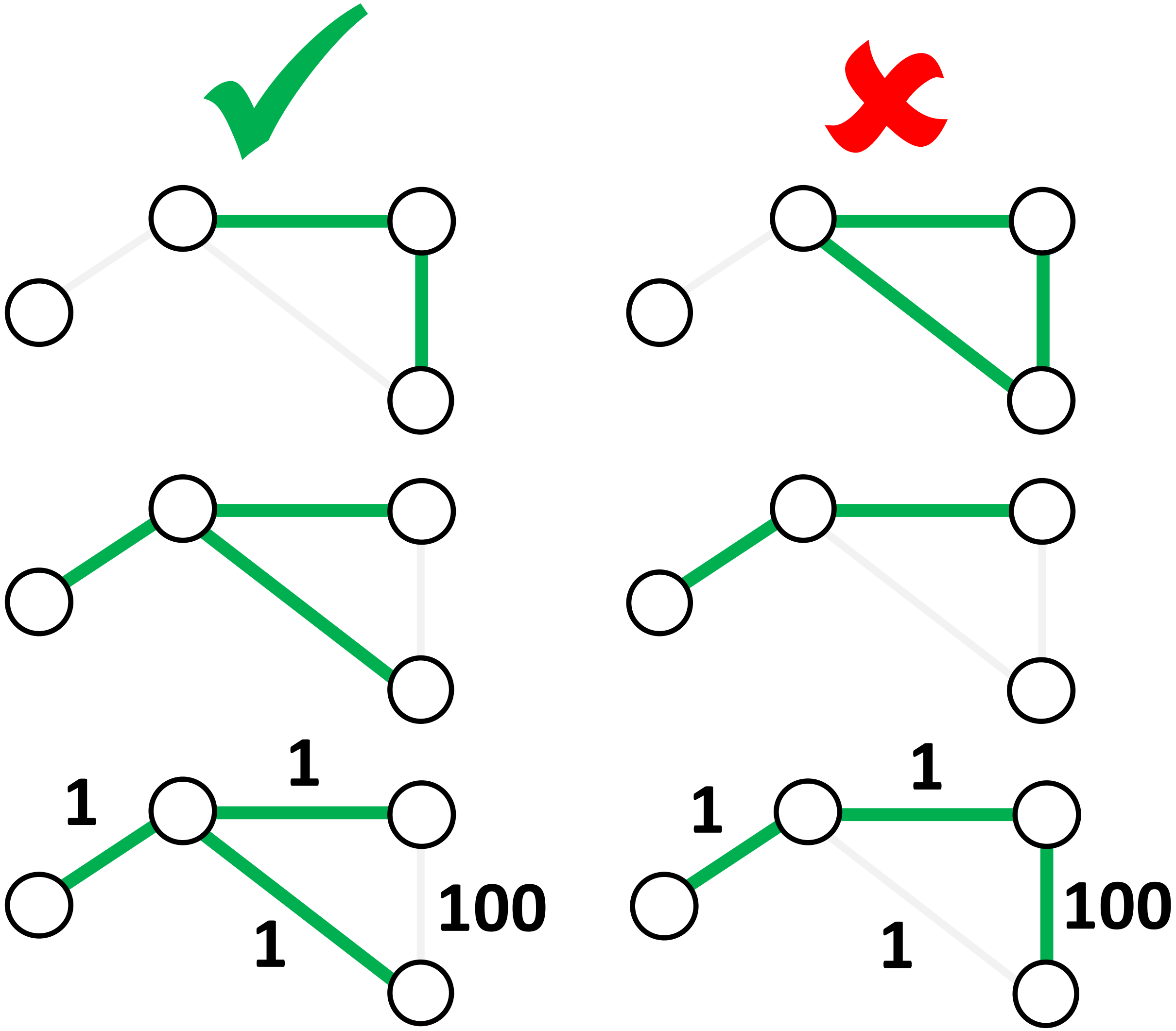
Tree if it is connected and acyclic.

Spanning tree if it is a tree and includes all vertices in the graph.

Minimum Spanning Tree (MST)



Given a connected graph, a subset of edges is a…

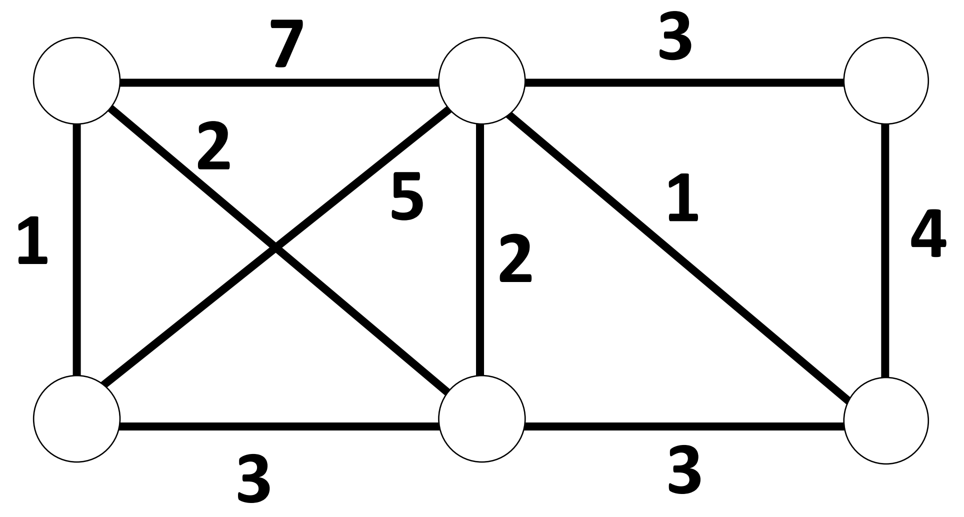


Tree if it is connected and acyclic.

Spanning tree if it is a tree and includes all vertices in the graph.

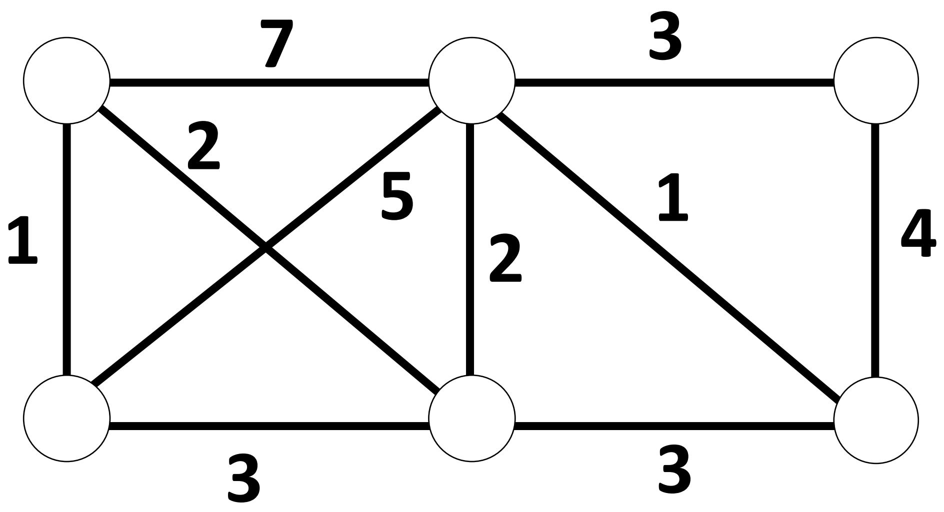
Minimum spanning tree if it is a spanning tree whose sum of edge costs is the minimum possible value.

MST Problem



<u>Goal:</u> Given a connected, edge weighted graph, find its Minimum Spanning Tree.

MST Problem

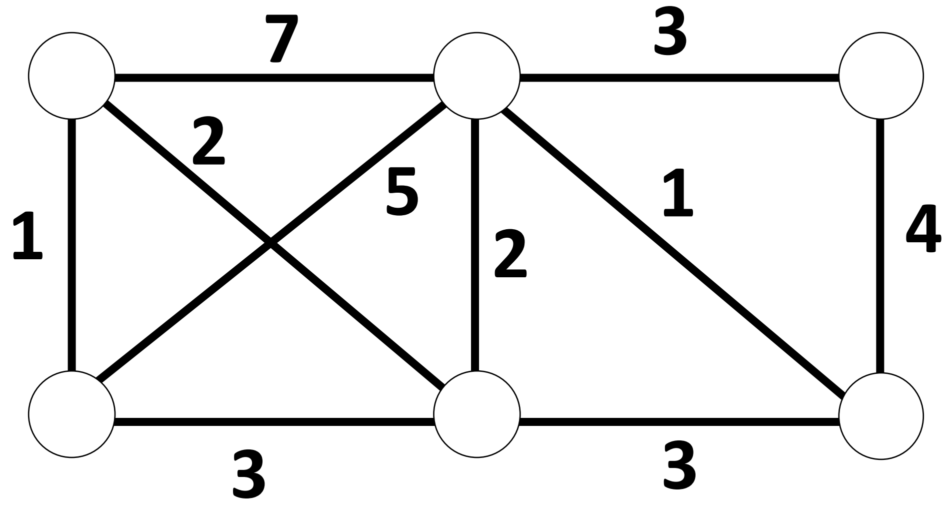


Greedy Algorithms:

- Make the choice that best helps some objective.
- Do not look ahead, plan, or revisit past decisions.
- Hope that optimal local choices lead to optimal global solutions.

Kruskal’s MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

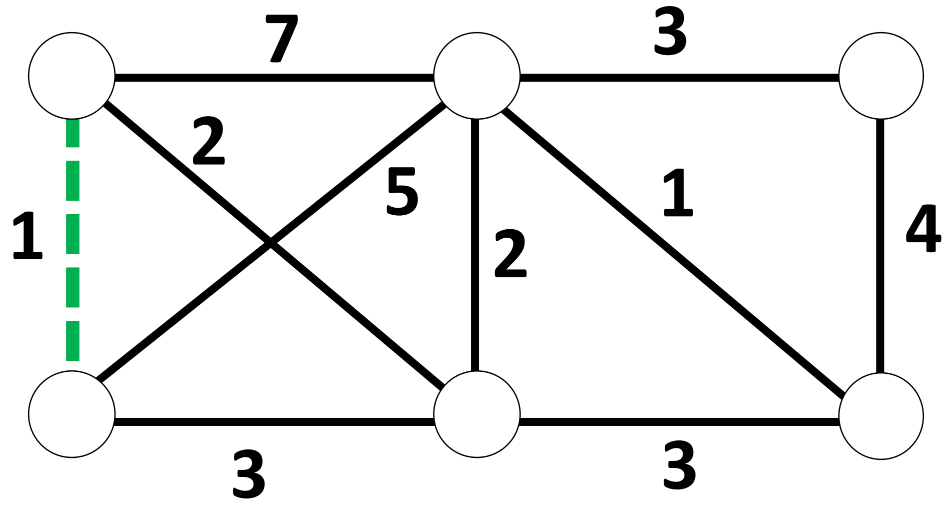


Greedy Algorithms:

- Make the choice that best helps some objective.
- Do not look ahead, plan, or revisit past decisions.
- Hope that optimal local choices lead to optimal global solutions.

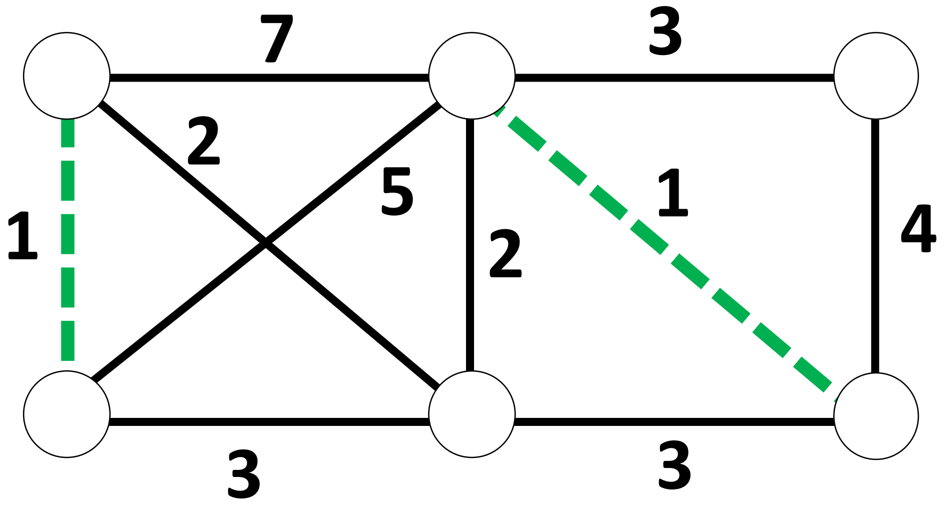
Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.



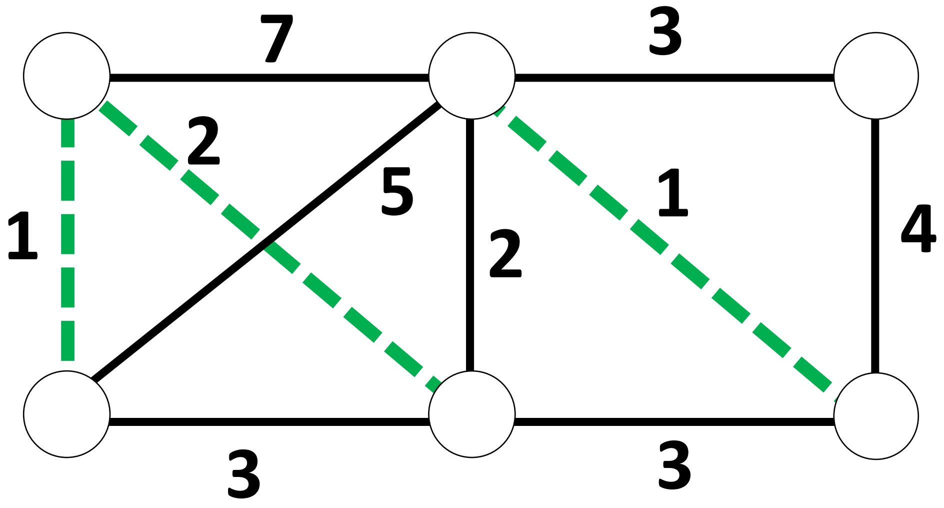
Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.



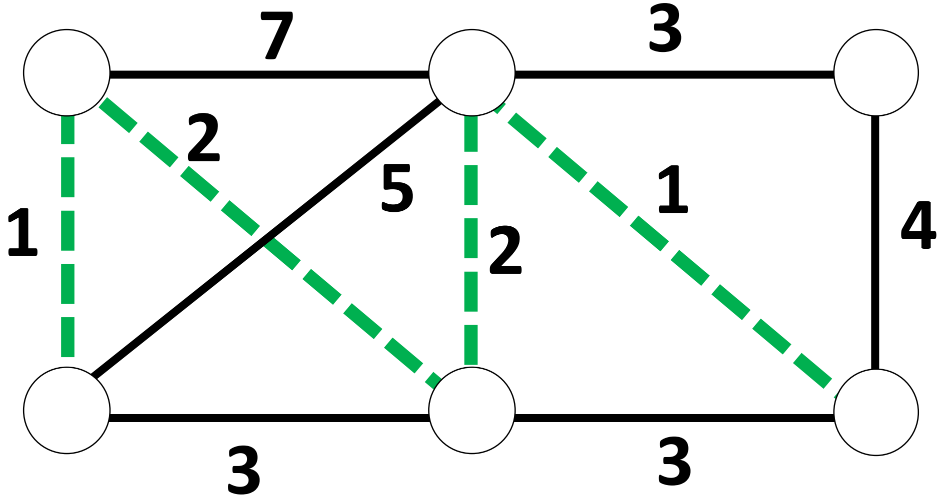
Kruskal’s MST Algorithm

<u>Algorithm:</u> Add the edge with smallest weight, that does not create a cycle.



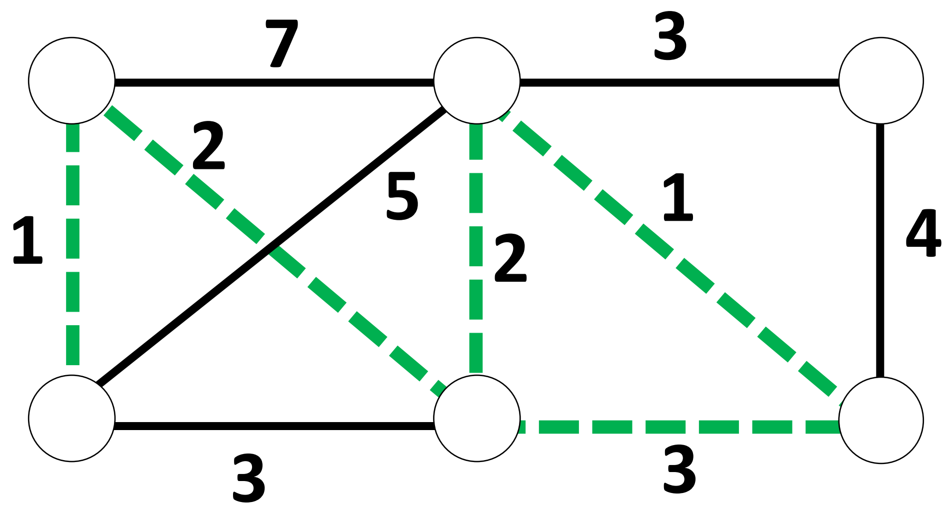
Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.



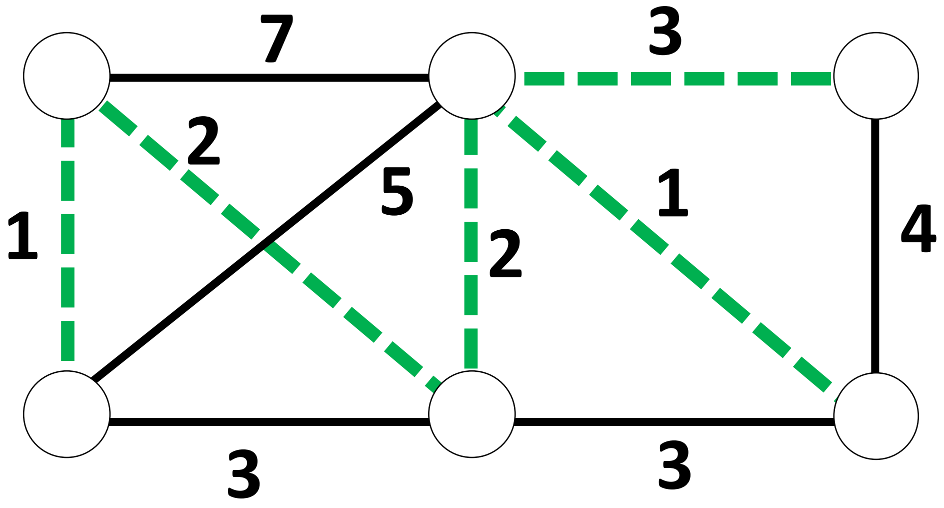
Kruskal’s MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.



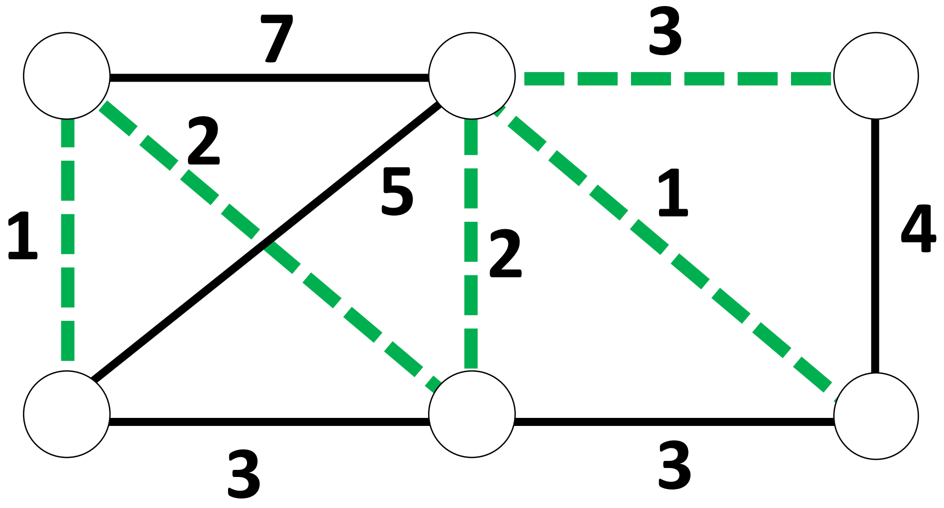
Kruskal's MST Algorithm

<u>Algorithm:</u> Add the edge with smallest weight, that does not create a cycle.



Kruskal's MST Algorithm

<u>Algorithm:</u> Add the edge with smallest weight, that does not create a cycle.



Basic algorithm analysis questions:

1. Is the solution valid? (Does it actually find a spanning tree?)
2. Is the solution optimal? (Does it find the best spanning tree?)
3. What is the running time?

Kruskal’s MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of validity: ?

Kruskal’s MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of validity: Let $G = (V, E)$ be the connected graph, and $T \subseteq E$ be the set of edges resulting from Kruskal’s algorithm.

Kruskal’s MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of validity: Let $G = (V, E)$ be the connected graph, and $T \subseteq E$ be the set of edges resulting from Kruskal’s algorithm.

What do we need to show?

Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of validity: Let $G = (V, E)$ be the connected graph, and $T \subseteq E$ be the set of edges resulting from Kruskal's algorithm.

What do we need to show?

1. T is a tree.

2. T spans G.

Kruskal's MST Algorithm

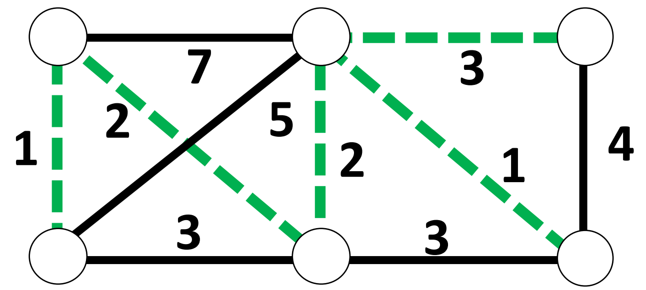
<u>Algorithm:</u> Add the edge with smallest weight, that does not create a cycle.

<u>Proof of validity:</u> Let $G = (V, E)$ be the connected graph, and $T \subseteq E$ be the set of edges resulting from Kruskal's algorithm.

What do we need to show?

1. T is a tree. T is connected (otherwise we could add more edges without creating cycles) and has no cycles (can't add edges that create cycles).

2. T spans G.



Kruskal’s MST Algorithm

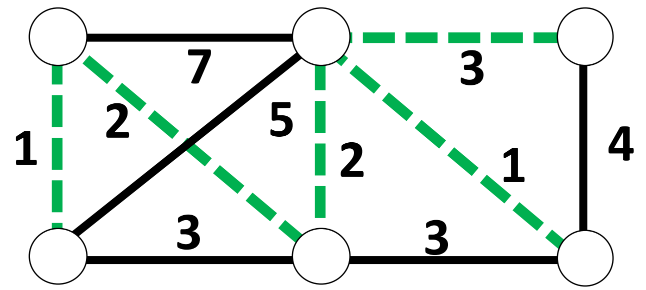
Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of validity: Let $G = (V, E)$ be the connected graph, and $T \subseteq E$ be the set of edges resulting from Kruskal’s algorithm.

What do we need to show?

1. T is a tree. T is connected (otherwise we could add more edges without creating cycles) and has no cycles (can’t add edges that create cycles).

2. T spans G. If it did not, we could have added more edges to connected unreached nodes without creating cycles.



Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

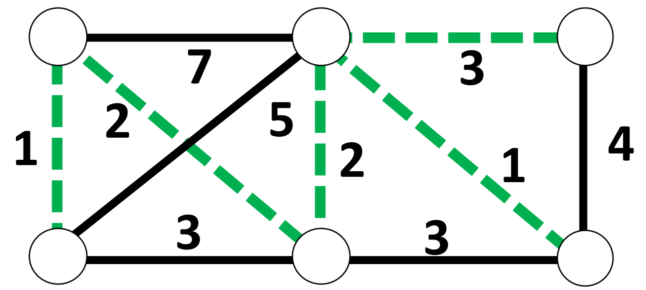
Proof of validity: Let $G = (V, E)$ be the connected graph, and $T \subseteq E$ be the set of edges resulting from Kruskal's algorithm.

What do we need to show?

1. T is a tree. T is connected (otherwise we could add more edges without creating cycles) and has no cycles (can't add edges that create cycles).

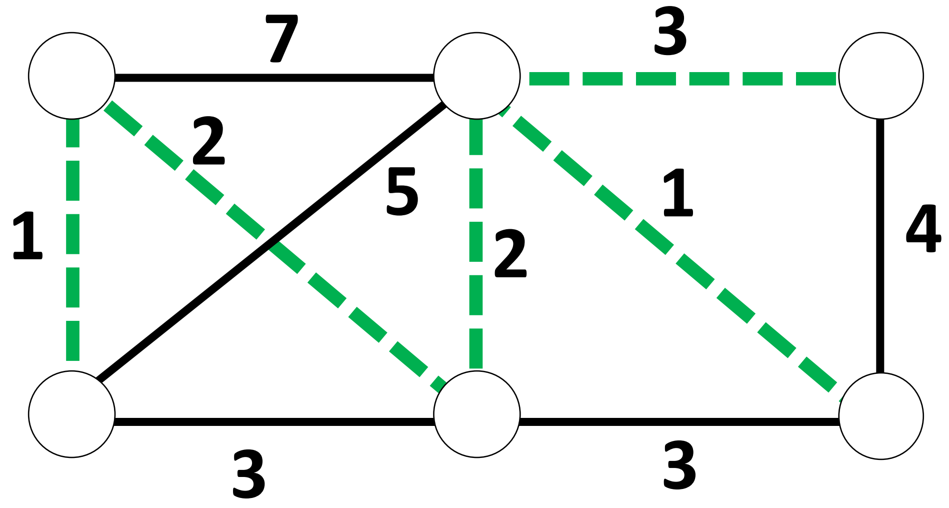
2. T spans G. If it did not, we could have added more edges to connected unreached nodes without creating cycles.

$\therefore T$ is a spanning tree of G



Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.



Basic algorithm analysis questions:

1. ~~Is the solution valid? (Does it actually find a spanning tree?)~~
2. Is the solution optimal? (Does it find the best spanning tree?)
3. What is the running time?

Kruskal's MST Algorithm

<u>Algorithm:</u> Add the edge with smallest weight, that does not create a cycle.

<u>Proof of optimality:</u> T is an MST, because???

Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of optimality: T is an MST, because???

We need a plan:

Kruskal's MST Algorithm

<u>Algorithm:</u> Add the edge with smallest weight, that does not create a cycle.

<u>Proof of optimality:</u> T is an MST, because???

We need a plan:

Suppose we show that every edge selected by the algorithm is part of the optimal MST.

Kruskal’s MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of optimality: T is an MST, because???

We need a plan:

Suppose we show that every edge selected by the algorithm is part of the optimal MST. Can we conclude that the selected edges form the MST?

Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of optimality: T is an MST, because???

We need a plan:

Suppose we show that every edge selected by the algorithm is part of the optimal MST. Can we conclude that the selected edges form the MST?

How could a set of edges not be the MST?

Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of optimality: T is an MST, because???

We need a plan:

Suppose we show that every edge selected by the algorithm is part of the optimal MST. Can we conclude that the selected edges form the MST?

How could a set of edges not be the MST?

It could have edges not in the MST.
It could be missing edges from the MST.

Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of optimality: T is an MST, because???

We need a plan:

Suppose we show that every edge selected by the algorithm is part of the optimal MST. Can we conclude that the selected edges form the MST?

How could a set of edges not be the MST?

It could have edges not in the MST. ✗

It could be missing edges from the MST.

Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of optimality: T is an MST, because???

We need a plan:

Suppose we show that every edge selected by the algorithm is part of the optimal MST. Can we conclude that the selected edges form the MST?

How could a set of edges not be the MST?

It could have edges not in the MST. ✗

It could be missing edges from the MST. ✗

Spanning Trees all have $|V| - 1$ edges.

Kruskal's MST Algorithm

Algorithm: Add the edge with smallest weight, that does not create a cycle.

Proof of optimality: T is an MST, because???

We need a plan:

Suppose we show that every edge selected by the algorithm is part of the optimal MST. Can we conclude that the selected edges form the MST?

How could a set of edges not be the MST?

It could have edges not in the MST. ✗

It could be missing edges from the MST. ✗

Thus, the spanning tree constructed *is* the MST.

Spanning Trees all have $|V| -$ 1 edges.

MST Cut Property

Assume unique edge costs.

<u>Lemma:</u> Suppose that S is a subset of nodes from $G = (V, E)$. Then, the cheapest edge e between S and $V \setminus S$ is part of the MST.

<u>Proof:</u>

We need a plan:

Suppose we show that every edge selected by the algorithm is part of the optimal MST. Can we conclude that the selected edges form the MST?

How could a set of edges not be the MST?

It could have edges not in the MST. ✗

It could be missing edges from the MST. ✗

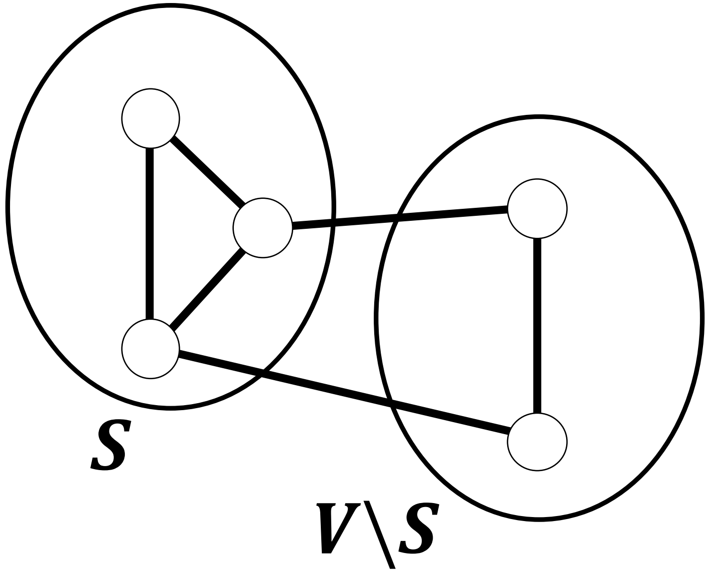
Thus, the spanning tree constructed *is* the MST.

Spanning Trees all have $|V| - 1$ edges.

MST Cut Property

Lemma: Suppose that S is a subset of nodes from $G = (V, E)$. Then, the cheapest edge e between S and $V \backslash S$ is part of the MST.

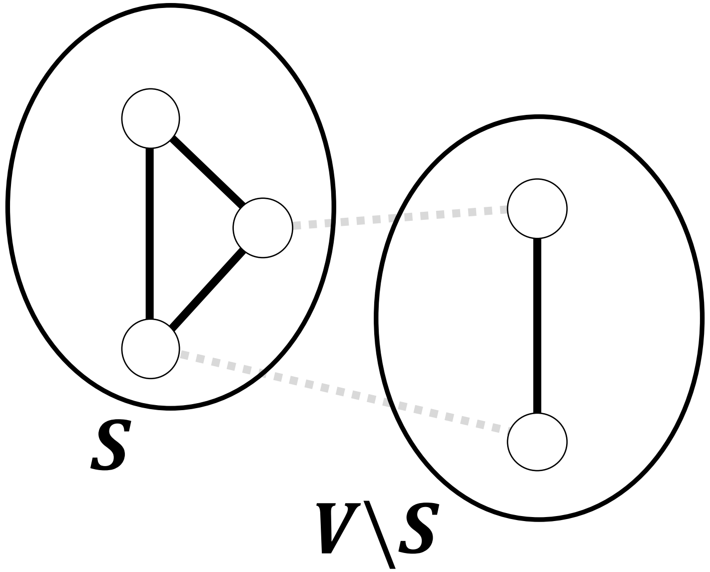
Proof:



MST Cut Property

<u>Lemma:</u> Suppose that S is a subset of nodes from $G = (V, E)$. Then, the cheapest edge e between S and $V \backslash S$ is part of the MST.

<u>Proof:</u> Any MST of G must include some edge between S and $V \backslash S$ (otherwise it would not be a spanning tree).

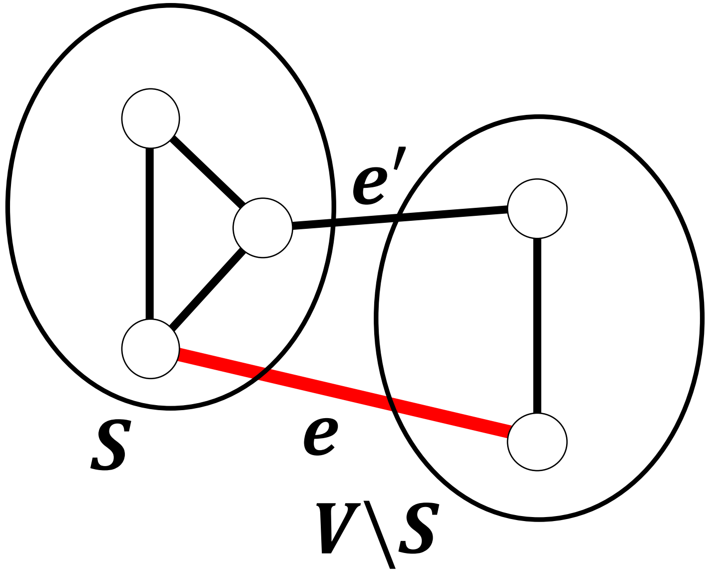


MST Cut Property

<u>Lemma:</u> Suppose that S is a subset of nodes from $G = (V, E)$. Then, the cheapest edge e between S and $V \backslash S$ is part of the MST.

<u>Proof:</u> Any MST of G must include some edge between S and $V \backslash S$ (otherwise it would not be a spanning tree).

Let $\boldsymbol{e}$ be the cheapest edge between S and $V \backslash S$.



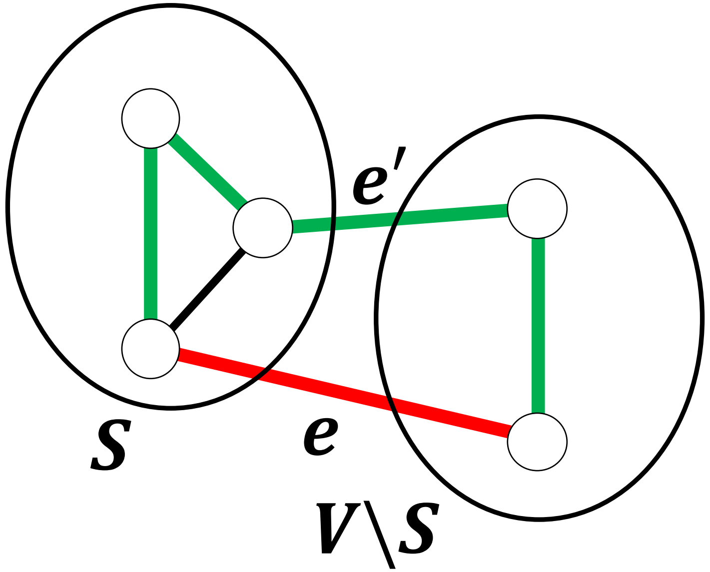
MST Cut Property

Lemma: Suppose that S is a subset of nodes from $G = (V, E)$. Then, the cheapest edge e between S and $V \setminus S$ is part of the MST.

Proof: Any MST of G must include some edge between S and $V \setminus S$ (otherwise it would not be a spanning tree).

Let $\boldsymbol{e}$ be the cheapest edge between S and $V \setminus S$.

Suppose $\boldsymbol{T}$ is the MST that does not include $\boldsymbol{e}$.



MST Cut Property

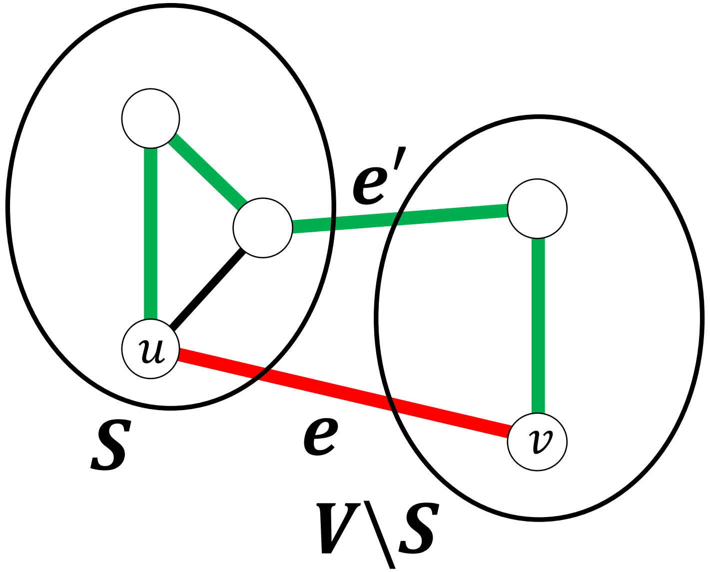
Lemma: Suppose that S is a subset of nodes from $G = (V, E)$. Then, the cheapest edge e between S and $V \backslash S$ is part of the MST.

Proof: Any MST of G must include some edge between S and $V \backslash S$ (otherwise it would not be a spanning tree).

Let $\boldsymbol{e}$ be the cheapest edge between S and $V \backslash S$.

Suppose $\boldsymbol{T}$ is the MST that does not include $\boldsymbol{e}$. Then:

1. $\boldsymbol{T} \cup \{\boldsymbol{e}\}$ must have a cycle. Because?



MST Cut Property

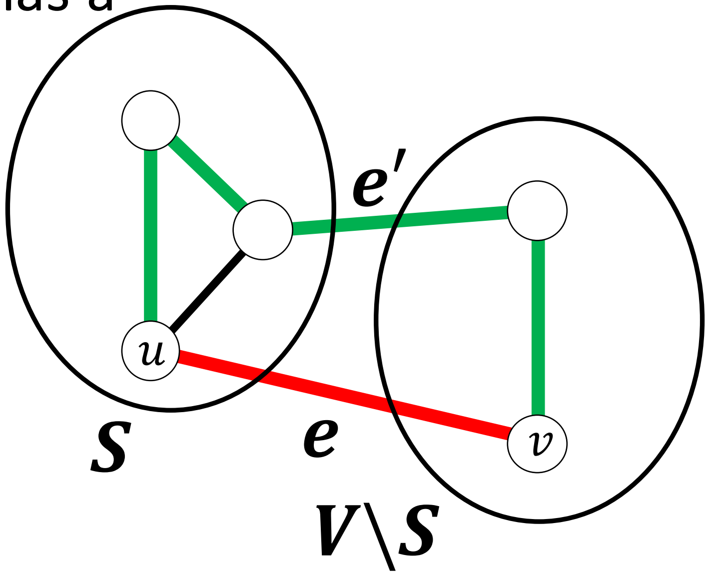
Lemma: Suppose that S is a subset of nodes from $G = (V, E)$. Then, the cheapest edge e between S and $V\backslash S$ is part of the MST.

Proof: Any MST of G must include some edge between S and $V\backslash S$ (otherwise it would not be a spanning tree).

Let $\boldsymbol{e}$ be the cheapest edge between S and $V\backslash S$.

Suppose $\boldsymbol{T}$ is the MST that does not include $\boldsymbol{e}$. Then:

1. $\boldsymbol{T} \cup \{\boldsymbol{e}\}$ must have a cycle. (Since spanning tree $\boldsymbol{T}$ already has a path between u and v, adding $\boldsymbol{e}$ will create a cycle.)



MST Cut Property

Lemma: Suppose that S is a subset of nodes from $G = (V, E)$. Then, the cheapest edge e between S and $V \backslash S$ is part of the MST.

Proof: Any MST of G must include some edge between S and $V \backslash S$ (otherwise it would not be a spanning tree).

Let $\boldsymbol{e}$ be the cheapest edge between S and $V \backslash S$.

Suppose $\boldsymbol{T}$ is the MST that does not include $\boldsymbol{e}$. Then:

1. $\boldsymbol{T} \cup \{\boldsymbol{e}\}$ must have a cycle. (Since spanning tree $\boldsymbol{T}$ already has a path between u and v, adding $\boldsymbol{e}$ will create a cycle.)
2. That cycle must have another edge e' between S and $V \backslash S$. (Since there must be a path from $u \in S$ to $v \in V \backslash S$ in $\boldsymbol{T}$)