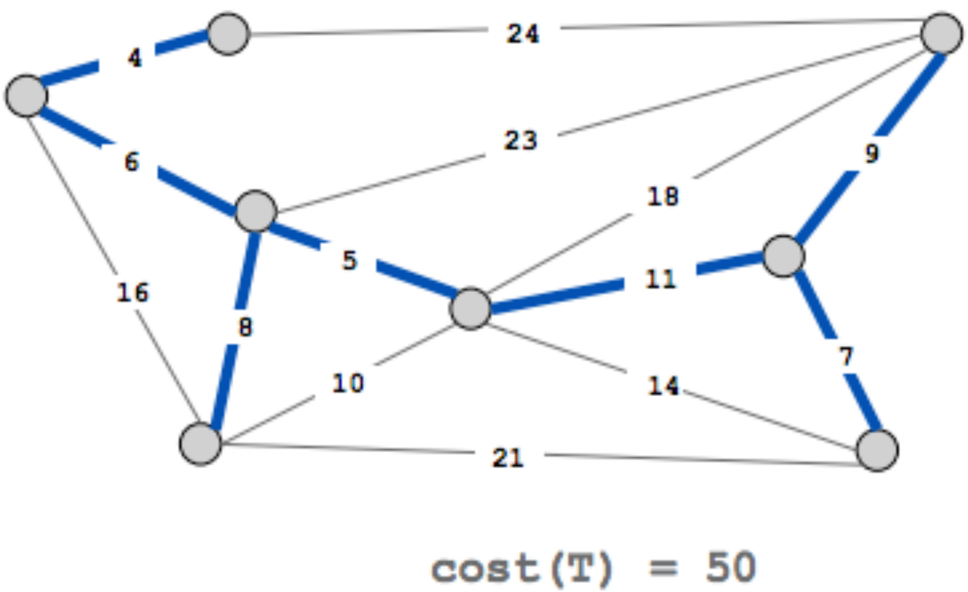
Minimum Spanning Trees

Given a connected graph with weighted edges, find a tree that connects all the vertices and the total weight is minimized.

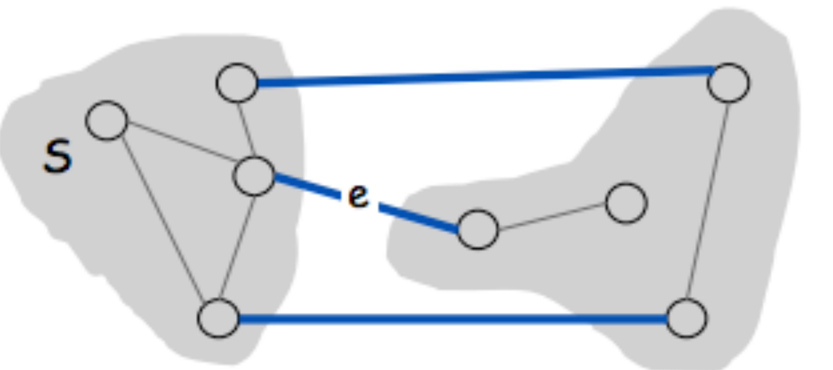


Is this a problem we can solve by brute force? Why or why not?

Minimum Spanning Trees

Cut property: Let S be any subset of vertices, and let e be the min cost edge with exactly one endpoint in S. Then the MST T* contains e.

Proof by Contradiction:
Suppose e does not belong to T*.
Let's see what happens.
Adding e to T* creates a (unique) cycle C in T*
Some other edge in C, say f, has exactly one endpoint in S.
$T = T^* \cup \{ e \} - \{ f \}$ is also a spanning tree.



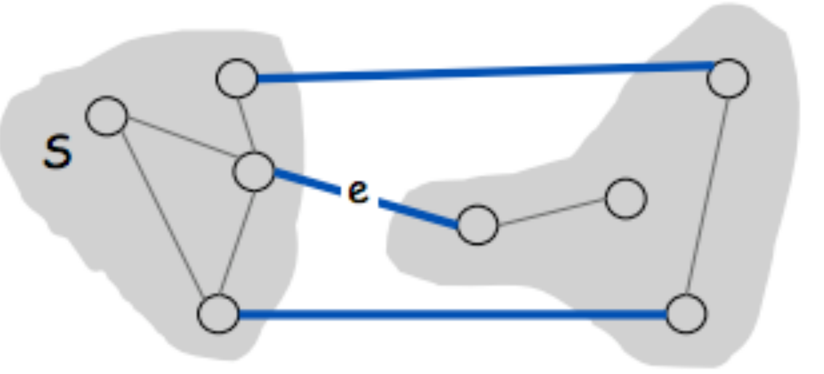
e is in the MST

What do we know about the weight of f compared to the weight of e? Why?

Minimum Spanning Trees

Cut property: Let S be any subset of vertices, and let e be the min cost edge with exactly one endpoint in S. Then the MST T* contains e.

Proof by Contradiction:
Suppose e does not belong to T*.
Let's see what happens.
Adding e to T* creates a (unique) cycle C in T*
Some other edge in C, say f, has exactly one endpoint in S.
$T = T^* \cup \{ e \} - \{ f \}$ is also a spanning tree.



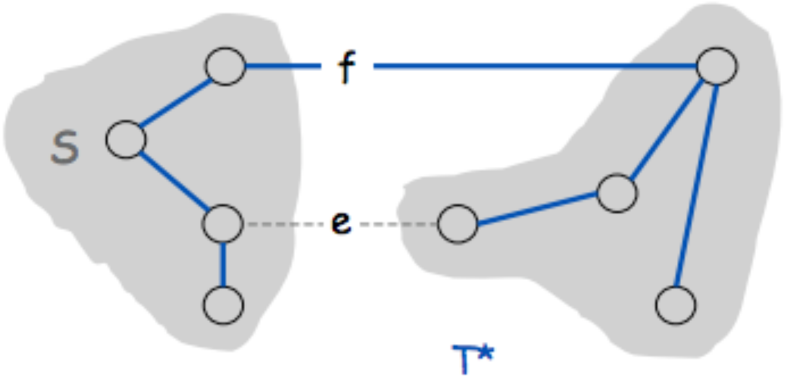
e is in the MST

Why do we get a contradiction?

Minimum Spanning Trees

Cycle property. Let C be any cycle in G, and let f be the max cost edge belonging to C. Then the MST T* does not contain f.

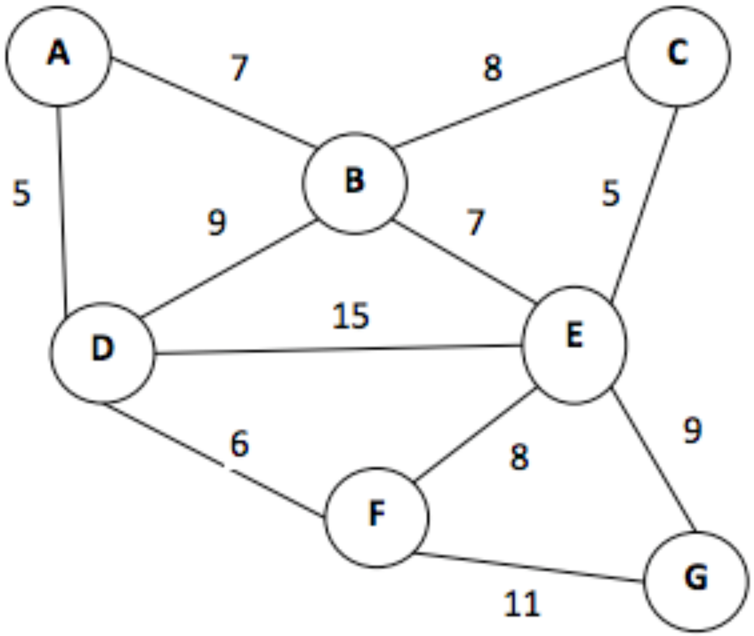
Proof by Contradiction:
Suppose f belongs to T*.
Let's see what happens.
Deleting f from T* disconnects T*.
Let S be one side of the cut.
Some other edge in C, say e, has exactly one endpoint in S.
T = T* ∪{ e } - { f } is also a spanning tree.



Why do we get a contradiction?

Minimum Spanning Trees

Kruskal's algorithm: Consider edges in ascending order of cost. Add the next edge to T unless doing so would create a cycle.



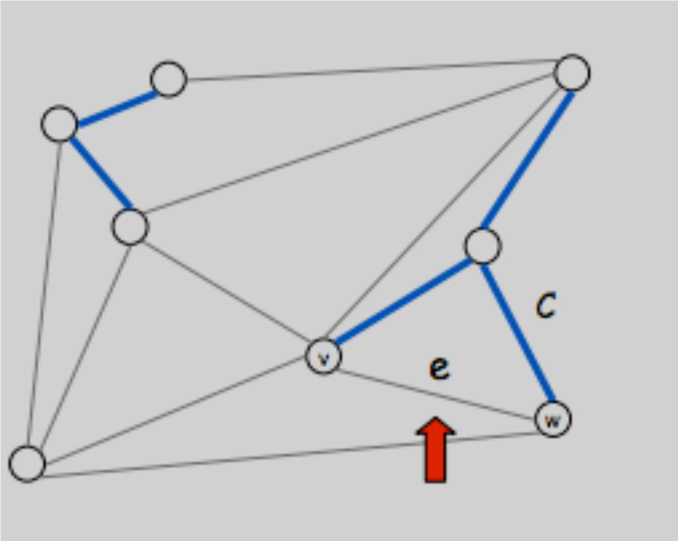
What tree would Kruskal give us?

Minimum Spanning Trees

Kruskal's algorithm: Consider edges in ascending order of cost. Add the next edge to T unless doing so would create a cycle.

Prove that Kruskal's algorithm computes the MST:

Case 1: Adding e to T creates a cycle C

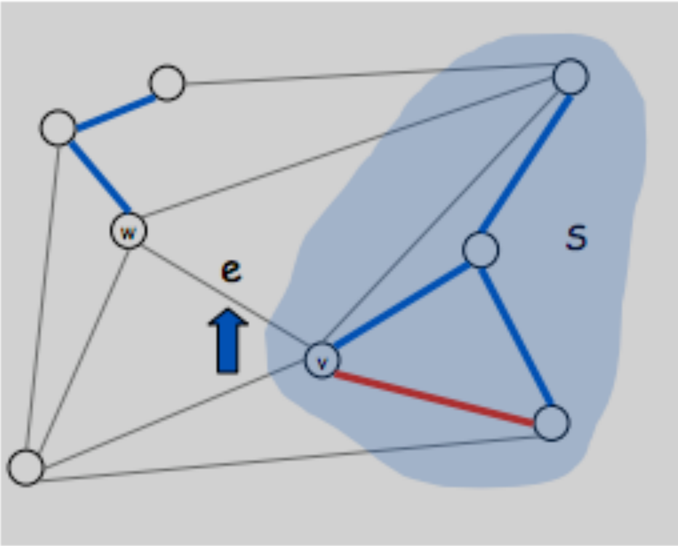


Why must e be the max weight edge in C?
Why must e NOT be in the MST?

Minimum Spanning Trees

Kruskal's algorithm: Consider edges in ascending order of cost. Add the next edge to T unless doing so would create a cycle.

Prove that Kruskal's algorithm computes the MST:
Case 2: Adding e = (v, w) to T does not create a cycle



Why must e be the min weight edge with exactly one endpoint in S?
Why must e be in the MST?

Kruskal Implementation and Cost

Kruskal's algorithm: Consider edges in ascending order of cost. Add the next edge to T unless doing so would create a cycle.

How could we use DFS to check if adding an edge creates a cycle? What would be the cost per cycle check? What would be the overall cost?

Kruskal Implementation and Cost

Kruskal's algorithm: Consider edges in ascending order of cost. Add the next edge to T unless doing so would create a cycle.

Perhaps we can do better.
Proposal: Maintain sets for each connected component of the edges T.

If have edge e=(v,w) and we see that v and w are in the same connected component, what does this tell us? If in different components, what does this tell us?

Union-Find Algorithms

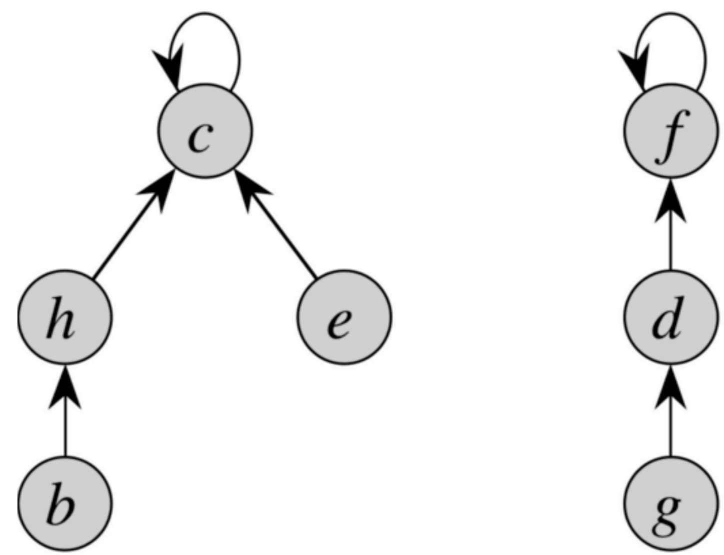
MakeSet(x) creates a new set containing x
Union(x,y) results in the union of the set containing x with the set containing y.
FindSet(x) returns the set containing x

We could store set name of element *i* in array X[i]

Give the order of growth for each of the operations.

Union-Find Algorithms

MakeSet(x) creates a new set containing x
Union(x,y) results in the union of the set containing x with the set containing y.
FindSet(x) returns the set containing x



Give the order of growth for each of the operations.

Kruskal Implementation and Cost

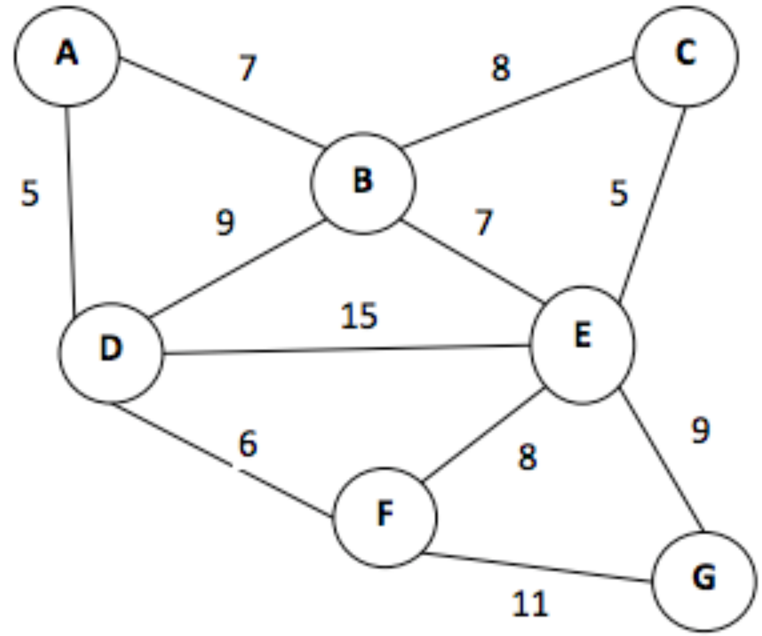
Kruskal's algorithm: Consider edges in ascending order of cost. Add the next edge to T unless doing so would create a cycle.

Perhaps we can do better.
Proposal: Maintain sets for each connected component of the edges T.

What is the order of growth for Kruskal?

Minimum Spanning Trees

Prim's algorithm: Start with a chosen root vertex and greedily grow tree T. At each step, add cheapest edge that has exactly one endpoint in T.

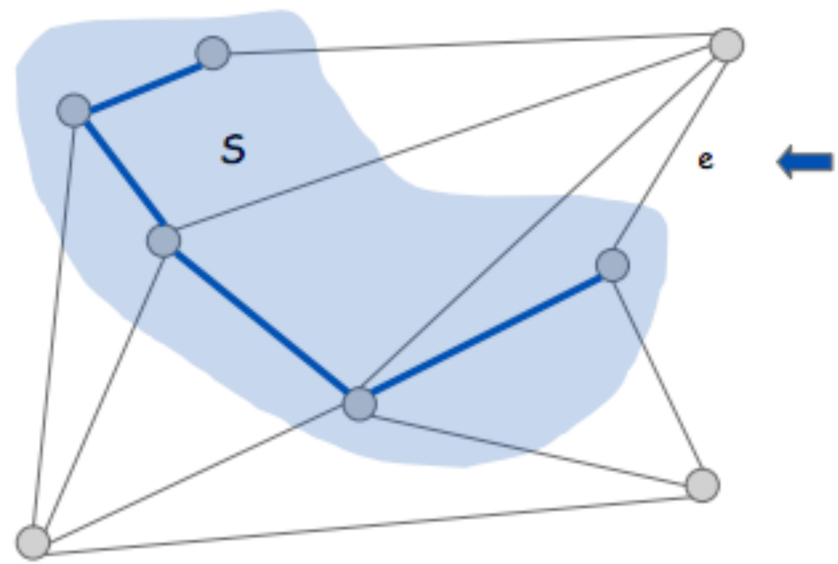


What tree would Prim's give us if we started with vertex A?

Minimum Spanning Trees

Prim's algorithm: Start with a chosen root vertex and greedily grow tree T. At each step, add cheapest edge that has exactly one endpoint in T.

Prove that Prim's algorithm computes the MST:
Let S be the subset of vertices in current tree T.
Prim adds the cheapest edge e with exactly one endpoint in S.



Why must e be in the MST?

Prim Implementation and Cost

Prim's algorithm: Start with a chosen root vertex and greedily grow tree T. At each step, add cheapest edge that has exactly one endpoint in T.

To find the cheapest edge with exactly one endpoint in S:
Maintain edges with (at least) one endpoint in S in a heap (priority queue)
Delete min to determine next edge e to add to T.
Disregard e if both endpoints are in S.
Upon adding e to T, add to PQ the edges incident to one endpoint.

What is the order of growth for delete min?
What is the overall order of growth for Prim's?