MINIMUM (weight) SPANNING TREES
MINIMUM (weight) SPANNING TREES

Input: graph w/ edge weights
MINIMUM (weight) SPANNING TREES

Input: graph w/ edge weights

Output:
- tree
- span (reach) all vertices
- minimize sum of weights
MINIMUM (weight) SPANNING TREES

Input: graph w/ edge weights

Observations:
- Any critical edge (in terms of graph connectivity) must be in the MST (e.g. 70, 18)

Output:
- tree
- span (reach) all vertices
- minimize sum of weights
**Minimum (Weight) Spanning Trees**

**Input:** graph w/ edge weights

**Output:**
- ✓ tree
- ✓ span (reach) all vertices
- ✓ minimize sum of weights

**Observations:**
- Any critical edge (in terms of graph connectivity) must be in the MST (e.g., 70, 18)
- For any vertex \( v \) with 2 incident edges, the smaller edge \( e \) must be in the MST

**WHY?**
MINIMUM (weight) SPANNING TREES

Input: graph w/ edge weights

Output:
✓ tree
✓ span (reach) all vertices
✓ minimize sum of weights

Observations:
- Any critical edge (in terms of graph connectivity) must be in the MST (e.g., 70, 18)
- For any vertex v with 2 incident edges, the smaller edge e must be in the MST by contradiction: if e not used, v is a leaf in MST. So swap, get better tree!
MINIMUM (weight) SPANNING TREES

Input: graph w/ edge weights

Output:
- tree
- span (reach) all vertices
- minimize sum of weights

Observations:
- Any critical edge (in terms of graph connectivity) must be in the MST (e.g. 70, 18)
- For any vertex \( v \) with 2 incident edges, the smaller edge \( e \) must be in the MST

\( \{ \) by contradiction: if \( e \) not used, \( v \) is a leaf in MST. So swap, \( \} \) get better tree!
MINIMUM (weight) SPANNING TREES

Input: graph w/ edge weights
Output:
- tree
- span (reach) all vertices
- minimize sum of weights

Observations, that we will generalize:
- Any critical edge (in terms of graph connectivity) must be in the MST (e.g. 70, 18)
- For any vertex \( v \) with 2 incident edges, the smaller edge \( e \) must be in the MST

\( v \)
\( e \)
\( e' \)
\( v' \)

\( 17 \)
\( 13 \)
\( 15 \)
\( 31 \)
\( 20 \)

\( 70 \)

\( 4 \)
\( 10 \)
\( 8 \)
So far, we know that for degree-1 & degree-2 vertices, the lightest incident edge must be in MST.
So far, we know that for degree-1 & degree-2 vertices the lightest incident edge must be in MST

This holds for all vertices
So far, we know that for degree-1 & degree-2 vertices the lightest incident edge must be in MST

This holds for all vertices
So far, we know that for degree-1 & degree-2 vertices, the lightest incident edge must be in MST. This holds for all vertices.
So far, we know that for degree-1 & degree-2 vertices the lightest incident edge must be in MST

This holds for all vertices

WHY?
So far, we know that for degree-1 & degree-2 vertices, the lightest incident edge must be in MST.

This holds for all vertices.

Suppose not true.
So far, we know that for degree-1 & degree-2 vertices, the lightest incident edge must be in MST.

This holds for all vertices.

Suppose not true.

"MST" without 2.
So far, we know that for degree-1 \& degree-2 vertices the lightest incident edge must be in MST.

This holds for all vertices.

Suppose not true.

"MST" without 2

Put 2 in.
So far, we know that for degree-1 & degree-2 vertices, the lightest incident edge must be in MST.

This holds for all vertices.

Suppose not true.

"MST" without 2

Put 2 in. Create cycle.
So far, we know that for degree-1 & degree-2 vertices the lightest incident edge must be in MST.

This holds for all vertices.

Suppose not true.

"MST" without 2
Put 2 in.
Create cycle
Remove last edge on cycle.
So far, we know that for degree-1 & degree-2 vertices, the lightest incident edge must be in MST.

This holds for all vertices.

Suppose not true.

Better spanning tree:

"MST" without 2.

Put 2 in.
Create cycle.
Remove last edge on cycle.
So far, we know that for degree-1 & degree-2 vertices, the lightest incident edge must be in MST.

This holds for all vertices.

Suppose not true.

"MST" without 2
Put 2 in.
Create cycle
Remove last edge on cycle.

Better spanning tree: Contradiction.
If every vertex votes for one edge, we might not get the entire MST.
If every vertex votes for one edge, we might not get the entire MST. What should we do in this example?
If every vertex votes for one edge, we might not get the entire MST. What should we do in this example? Best connection: $\min\{21, 18, 19\}$
If every vertex votes for one edge, we might not get the entire MST. What should we do in this example? Best connection: \(\min\{21, 18, 19\}\)

Once you know a component of MST, the lightest edge connecting it to the rest of the graph must be in MST.

\[\text{WHY?}\]
If every vertex votes for one edge, we might not get the entire MST. What should we do in this example? Best connection: \( \min \{21, 18, 19\} \)

Once you know a component of MST, the lightest edge connecting it to the rest of the graph must be in MST.

\[ \text{WHY?} \]

Same proof by contradiction as before.
If every vertex votes for one edge, we might not get the entire MST. What should we do in this example? Best connection: min\{21, 18, 19\}

Once you know a component of MST, the lightest edge connecting it to the rest of the graph must be in MST.

WHY?

Same proof by contradiction as before

Whatever MST you get
If every vertex votes for one edge, we might not get the entire MST. What should we do in this example? Best connection: \( \min \{21, 18, 19\} \)

Once you know a component of MST, the lightest edge connecting it to the rest of the graph must be in MST.

\[ \downarrow \text{WHY?} \]

Same proof by contradiction as before

Whatever MST you get, insert \( e \), get cycle
If every vertex votes for one edge, we might not get the entire MST. What should we do in this example? Best connection: \( \min\{21, 18, 19\} \)

Once you know a component of MST, the lightest edge connecting it to the rest of the graph must be in MST.

**WHY?**

Same proof by contradiction as before.

Whatever MST you get, insert \( e \), get cycle, improve MST, contradiction.
$A \subseteq V$

$B : V - A$
A ⊆ V
B: V - A
\text{Cut} \text{ separates } A, B
Cut separates $A, B$

Redraw $G$

Cut crosses all

This is an abstract concept. (independent of drawing)

A cut identifies all edges between $A, B$
This is an abstract concept. (independent of drawing)

A cut identifies all edges between $A, B$

$A \subseteq V$
$B : V - A$

Cut separates $A, B$

Cut crosses all

Redraw $G$
A \subseteq V
B : V - A

Cut separates A, B

Redraw G
Cut crosses all

This is an abstract concept. (independent of drawing)
A cut identifies all edges between A, B

CLAIM: for any cut, the min-weight edge crossing the cut must be in MST
This is an abstract concept.
(independent of drawing)

A cut identifies all edges between $A, B$

CLAIM: for any cut, the min-weight edge crossing the cut must be in MST
This is an abstract concept. (independent of drawing)

A cut identifies all edges between $A, B$

CLAIM: for any cut, the min-weight edge crossing the cut must be in MST

Redraw $G$

Cut crosses all
This is an abstract concept. (independent of drawing)

A cut identifies all edges between $A, B$

CLAIM: for any cut, the min-weight edge crossing the cut must be in MST
A ∈ V  
B: V - A

**Cut** separates A, B

Redraw G

Cut crosses all

---

This is an abstract concept. (independent of drawing)

A cut identifies all edges between A, B

CLAIM: for any cut, the min-weight edge crossing the cut must be in MST
A \subseteq V
B: V - A

Cut separates A, B

Redraw G

Cut crosses all

This is an abstract concept.
(independent of drawing)

A cut identifies all edges between A, B

CLAIM: for any cut, the min-weight edge crossing the cut must be in MST
This is an abstract concept. (independent of drawing) A cut identifies all edges between \( A, B \).


CLAIM: for any cut, the min-weight edge crossing the cut must be in MST.
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST

Proof: let $u,v$ be the min-weight edge. Suppose it is not in MST.
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST.

Proof: let $u, v$ be the min-weight edge. Suppose it is not in MST.

Focus on MST and the given cut.
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST

Proof: let $u,v$ be the min-weight edge. Suppose it is not in MST.

- Focus on MST and the given cut
- Insert $u,v$
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST

Proof: let \( u, v \) be the min-weight edge. Suppose it is not in MST.

- Focus on MST and the given cut
- Insert \( u, v \): create cycle
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST

Proof: let $u, v$ be the min-weight edge. Suppose it is not in MST.

- Focus on MST and the given cut
- Insert $u, v$: create cycle

$\downarrow$ must contain another edge that crosses cut
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST.

Proof: let $u, v$ be the min-weight edge. Suppose it is not in MST.

- Focus on MST and the given cut
- Insert $u, v$: create cycle
  - must contain another edge that crosses cut
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST

Proof: let $u,v$ be the min-weight edge. Suppose it is not in MST.

- Focus on MST and the given cut
- Insert $u,v$: create cycle
  - must contain another edge that crosses cut
- Remove that edge: improve tree: CONTRADICTION