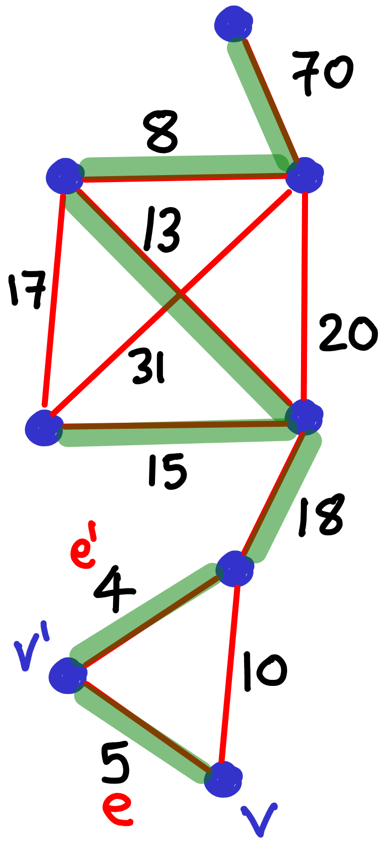
MINIMUM (weight) SPANNING TREES



Input: graph w/ edge weights

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

Observations, that we will generalize:

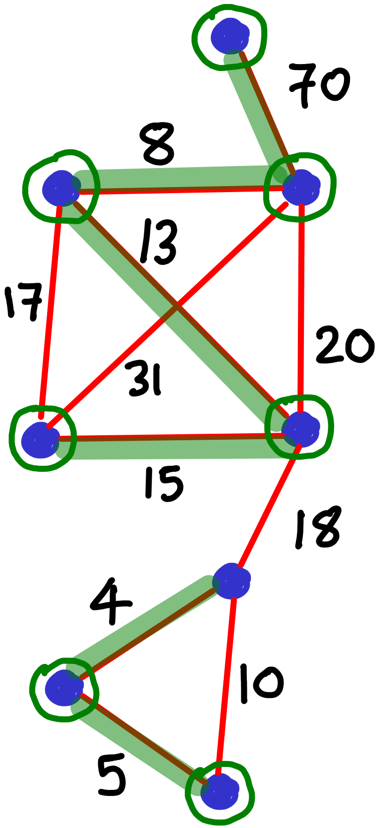
- Any critical edge (in terms of graph connectivity) must be in the MST (e.g. 70, 18)

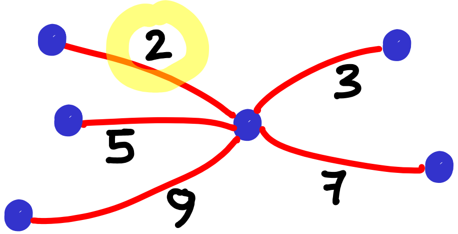
- For any vertex v with 2 incident edges, the smaller edge e must be in the MST

by contradiction: if e not used, v is a leaf in MST. So swap.
↪ get better tree!

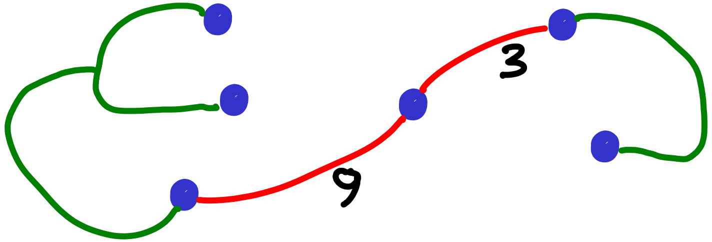
So far, we know that for degree-1 & degree-2 vertices
the lightest incident edge must be in MST

This holds for all vertices

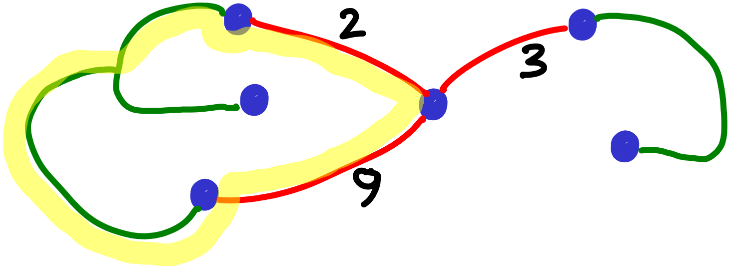




Suppose not true



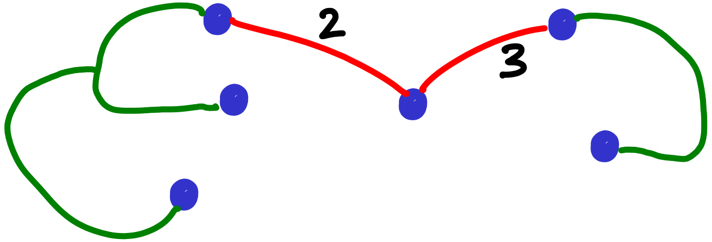
"MST" * without 2



Put 2 in.
Create cycle

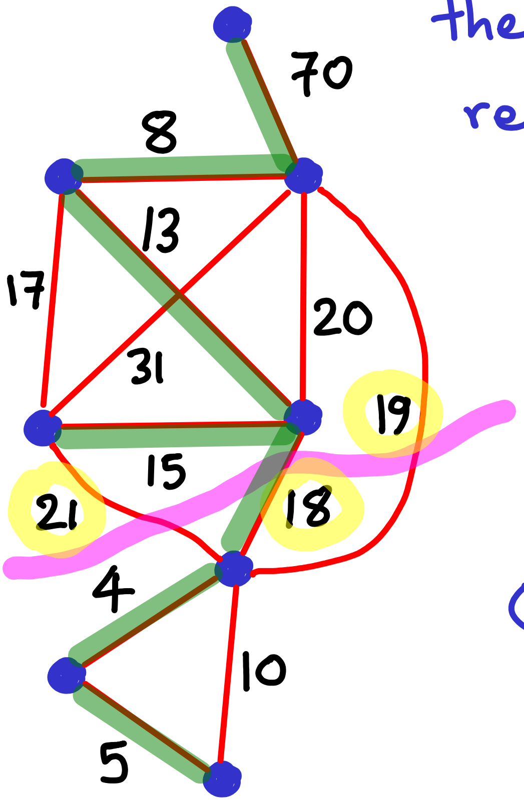
Remove last edge on cycle

* Better spanning tree:
Contradiction

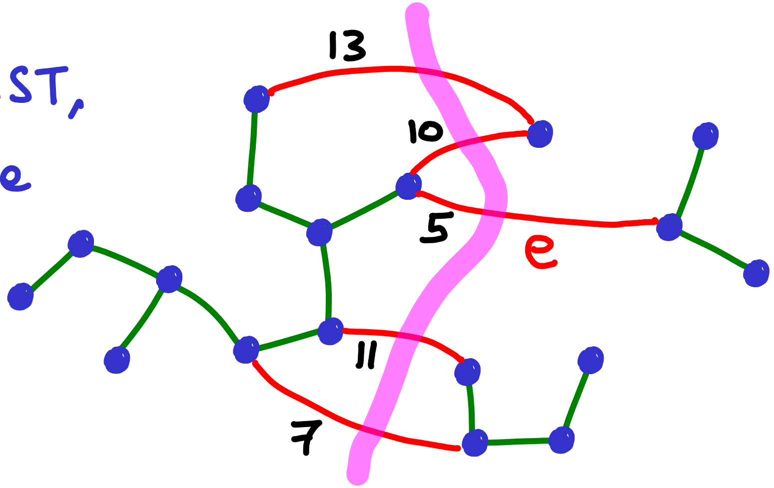


If every vertex votes for one edge, we might not get the entire MST.

What should we do in this example? Best connection: min{21, 18, 19}

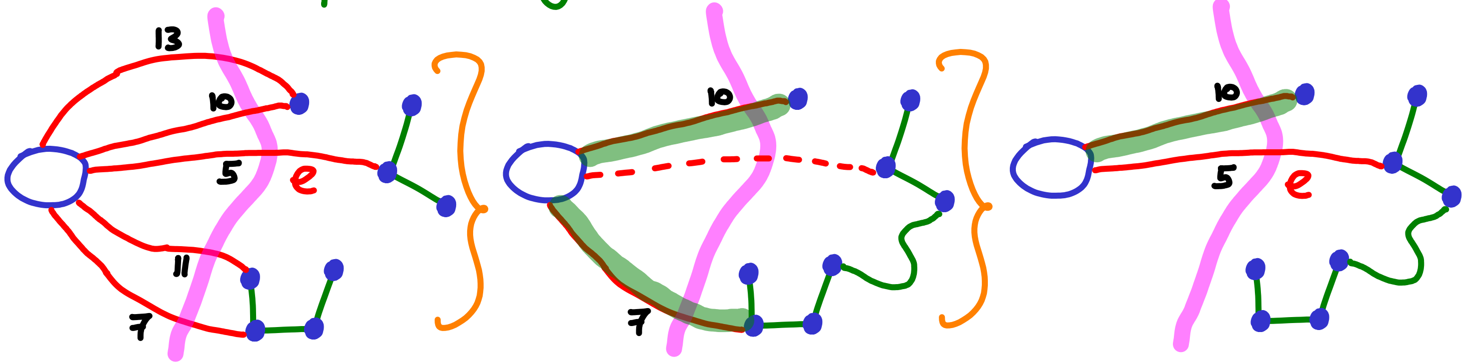


Once you know a component of MST, the lightest edge connecting it to the rest of the graph must be in MST.



WHY?

Same proof by contradiction as before



Whatever MST you get, insert e, get cycle, improve MST, contradiction

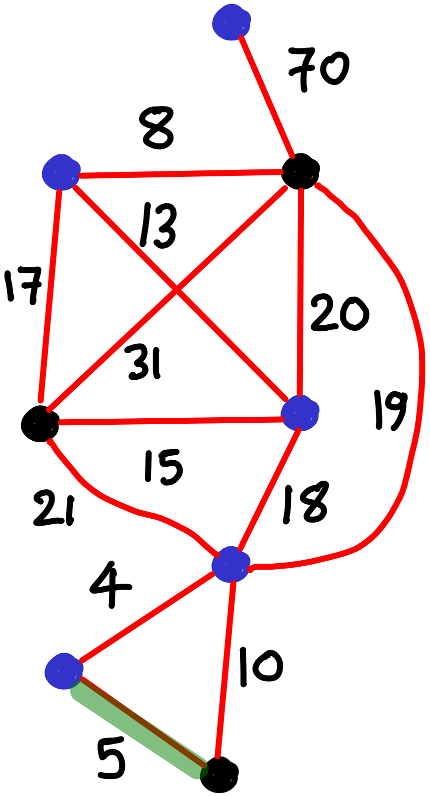
$A \subseteq V$

$B: V - A$

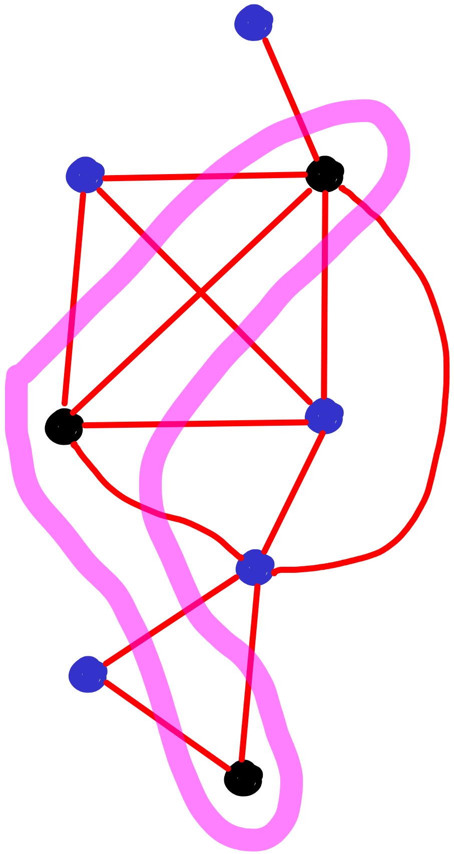
Cut separates A, B

Redraw G

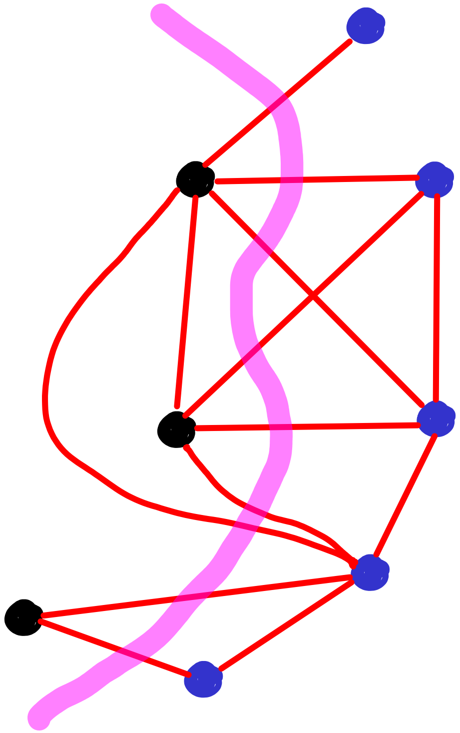
Cut crosses all

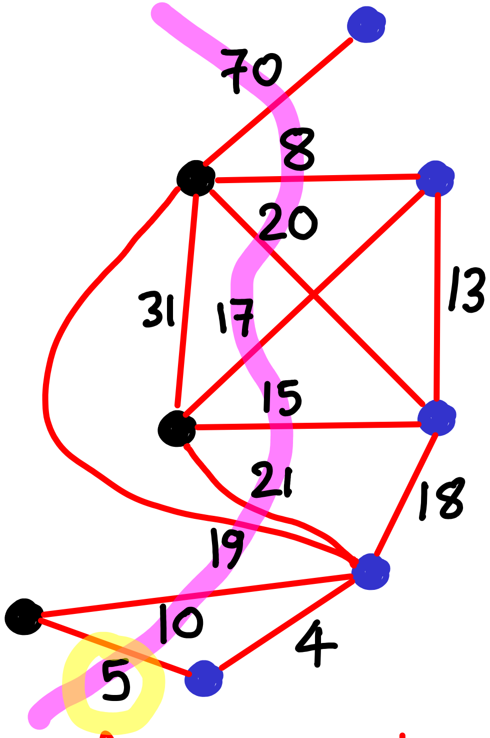


This is an abstract concept.
(independent of drawing)



A cut identifies all edges between A, B

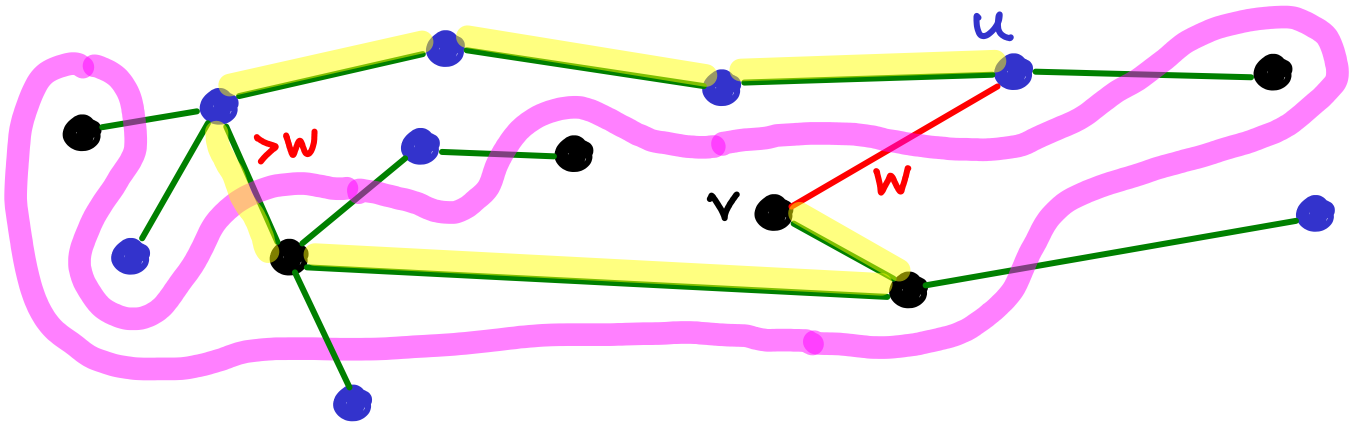




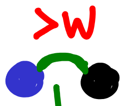
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST

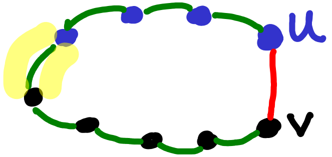
CLAIM: for any cut, the min-weight edge crossing the cut must be in MST

Proof: let u,v be the $\underbrace{\text{min-weight}}_{w}$ 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