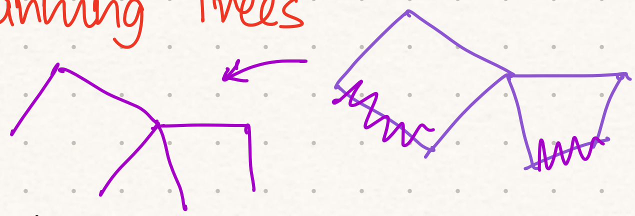
Minimum Spanning Trees



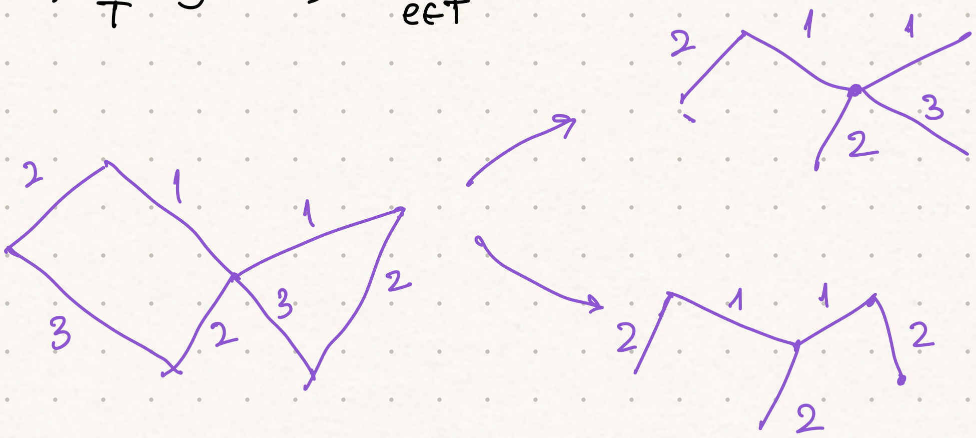
$G = (V, E)$

Spanning Tree: A spanning tree is a subgraph of G s.t. • it is a tree
• it contains all vertices.
covers

If G is connected then so must the spanning tree.

→ Minimum Spanning tree given $wt : E \to \mathbb{R}$

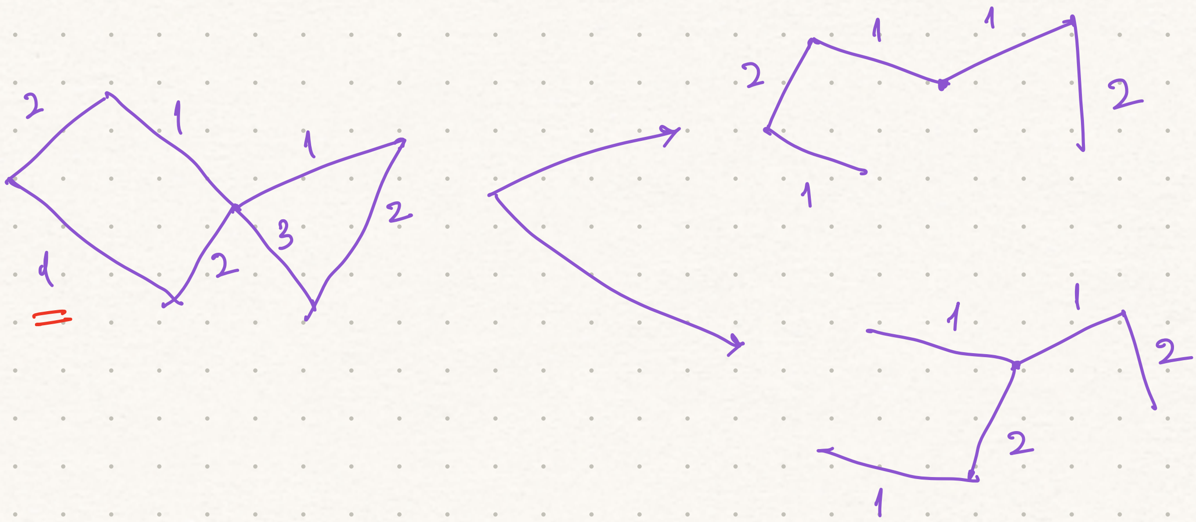
↳ $wt(\text{spanning tree } T) = \sum_{e \in T} wt(e)$.



wt = 9

wt = 8

Remark: It is possible to have multiple MSTs.



wt 7

wt = 7

Algorithmic task: Find a min Spanning Tree.
(Connected)

If all the edge weights are distinct, then we have a unique MST.

For the sake of contradiction, let us assume that $\underbrace{T_1 \text{ and } T_2}_{\text{spanning trees.}}$ both attain min wt of C^*.

T_1	T_2
$E(T_1)$	$E(T_2)$
$e_1 \in T_1$ but $\notin T_2$	$e_2 \in T_2$ $\notin T_1$
(u_1, v_1)	(u_2, v_2).

"Swapping Arguments"

"Exchange Arguments"

Say $wt(e_1) < wt(e_2)$

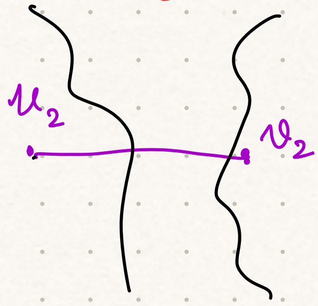
→ ~~Remove e_2 from T_2~~

→ ~~Add e_1 to $T_2 \setminus \{e_2\}$.~~

Not precise.

$e_1 \in T_1 \setminus T_2$ ← min wt edge in $T_1 \setminus T_2$

$e_2 \in T_2 \setminus T_1$. ← min wt edge in $T_2 \setminus T_1$.



$\boxed{e_3 \geq e_2}$

$\boxed{e_3 = e_2}$

$e_3 \in T_1 \cap T_2$

$wt(T_2) = wt(T_1)$

$\boxed{T_2 \cup \{e_1\}}$ ← This has a cycle.

↳ Remove an edge from the cycle. e_3 ↗ $e_3 \in T_2$

$\Rightarrow e_3 \notin T_1 \setminus T_2$

$T_3 = T_2 \cup \{e_1\} - \{e_3\}$ $\qquad wt(T_3) = wt(T_2) + wt(e_1) - wt(e_3)$

if $e_3 = e_2$ then $wt(T_3) < wt(T_2)$ $\qquad \leq w(T_2)$.

↑ they are the same edge.

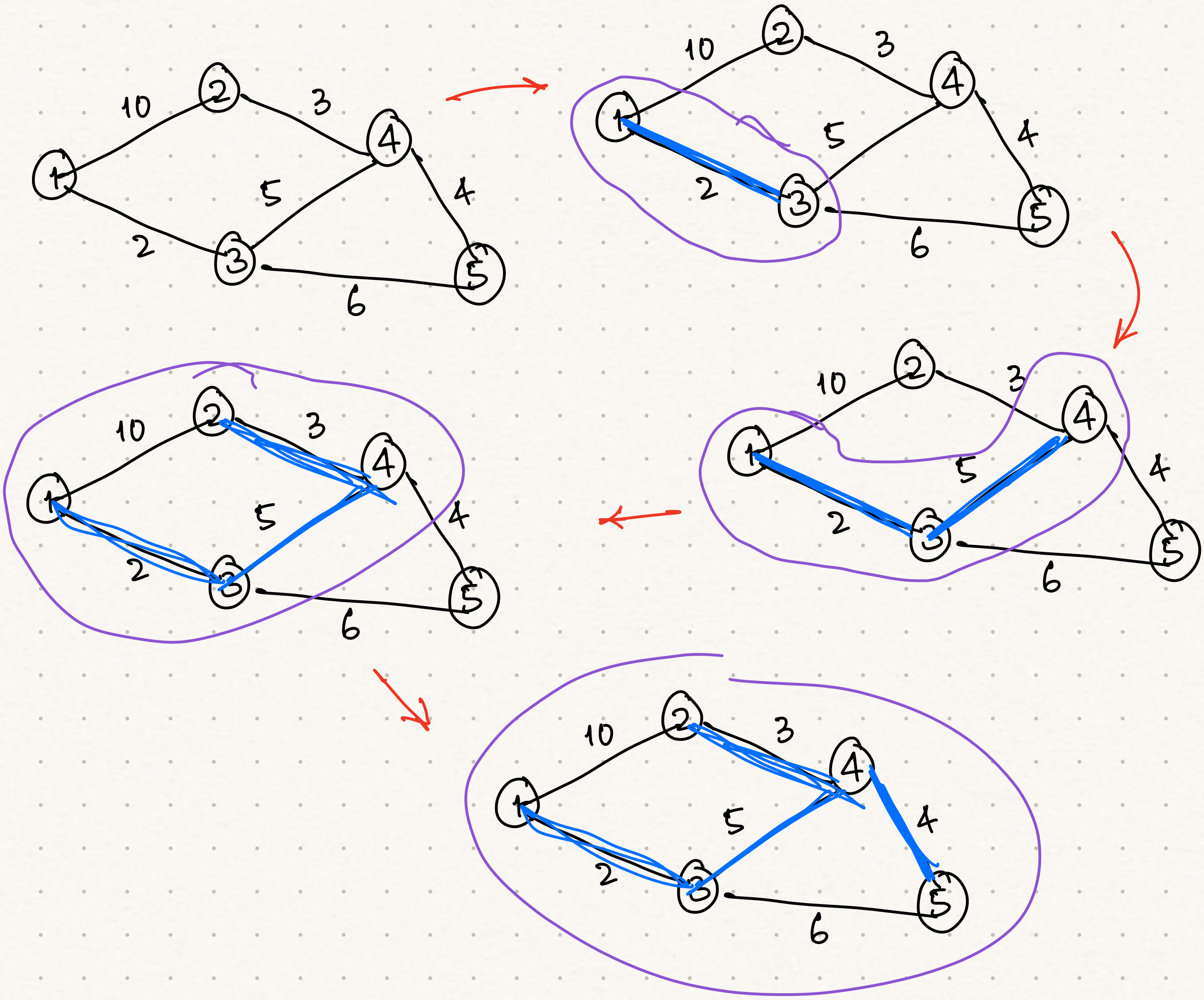
↳ $e_3 \in T_2 \setminus T_1$ (by enforcing that we pick an edge that does not belong to T_1 from the cycle).

$wt(e_3) > wt(e_2) > wt(e_1)$.

$\Rightarrow wt(T_3) < wt(T_2)$.

Ref: Jeff Erikson's MST chapter.

Prim's algorithm:



$T \leftarrow \{\}$

$S = \{s\}$

While $S \neq V$:

Pick v and edge (u,v) s.t

- $v \in N(S) \cap (V \setminus S)$ and $u \in S$
- $wt(u,v) \leq wt(u',v')$ $\forall$ $u' \in S$ and $v' \in N(S) \cap V \setminus S$

$T \leftarrow T \cup \{(u,v)\}$

$S \leftarrow S \cup \{v\}$.

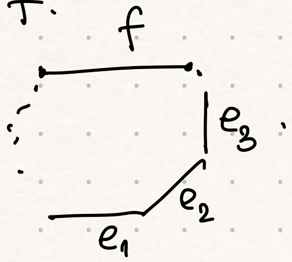
Return T.

n Extract Mins + O(n) overhead.

+O(m)

Cycle Property: Let C be any cycle in the given graph. Let edge f be the max wt edge in C. Then f cannot be part of a minimum spanning tree.

Pf: Suppose T^* be the MST s.t it contains f.



$\rightarrow T^* - \{f\}$ is a disconnected graph.
(it is a forest w/ 2 trees)

$e_1, e_2, \dots, e_k$ be edges in the cycle apart from f.

$(u_1, v_1), \dots, (u_k, v_k)$. $\forall i \in [k]$ there was a path from u_i to v_i

There must be a pair (u_{i^*}, v_{i^*}) that are disconnected. adding that edge e_{i^*} would give us another spanning tree.

$T \leftarrow (T^* - \{f\}) \cup \{e_{i^*}\}$

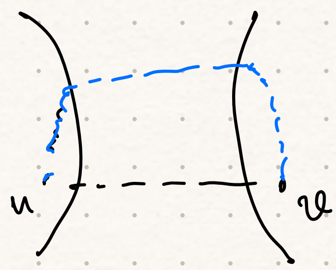
$wt(T) = wt(T^*) - wt(f) + wt(e_{i^*}) < wt(T^*)$

This contradicts the minimality of T^*.

Cut Property: Let U be a subset of V. Let e be the min wt edge in $E(U, V \setminus U)$. Then ~~a~~ the minimum spanning tree contains e.

Pf: T^* be the MST that does not contain $e = (u,v)$.

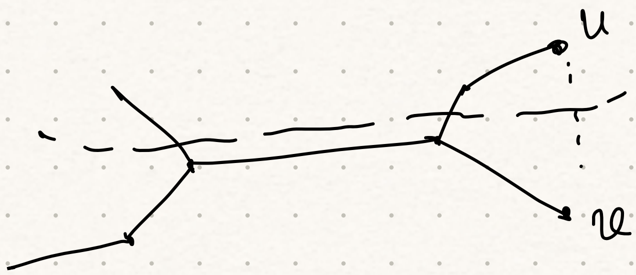
$u \in U$ $v \in V \setminus U$.



$T^* \cup \{e\}$ would have a cycle.

Since T^* is a spanning tree, u and v must be connected through a path which also must have an edge that crosses the cut.

Let f be that edge.



$T \leftarrow T^* - \{f\} \cup \{e\}$.

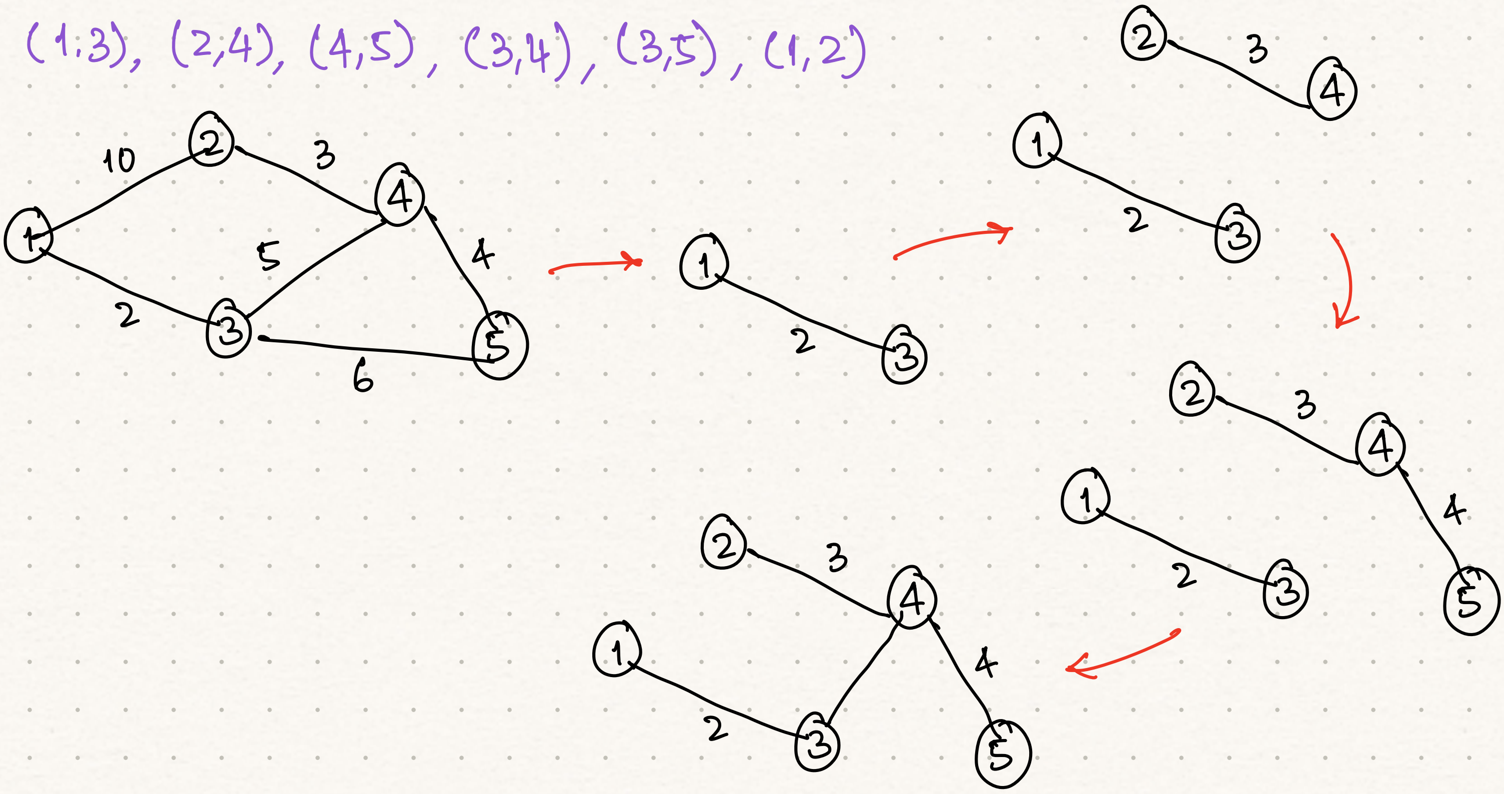
$wt(T) < wt(T^*)$ This contradicts the minimality of T^*.

Correctness of each step of Prim's algorithm is guaranteed by Cut property.

Kruskal's algo:

- Sort the edges in increasing order of their weights.
- Add edges to a tree as long as they do not create cycles.

(1,3), (2,4), (4,5), (3,4), (3,5), (1,2)



Cycle Property ~~and Cut property~~ guarantee the correctness. (by considering edges in the sorted order)