5112

Minimal Spanning Trees

Today

Finish EFT analysis (Interval Scheduling)

Minimal Spanning Trees

Does it produce a largest compatible subset?

Lemma: Let S be the output of EFT.
Let T be any compatible subset.
Order all jobs in S and T by EFT.

$K_1, \ldots, K_{|S|}$ $\qquad$ $L_1, \ldots, L_{|T|}$

Then $f(K_i) \leq f(L_i)$ for all $i \leq \min(|S|, |T|)$

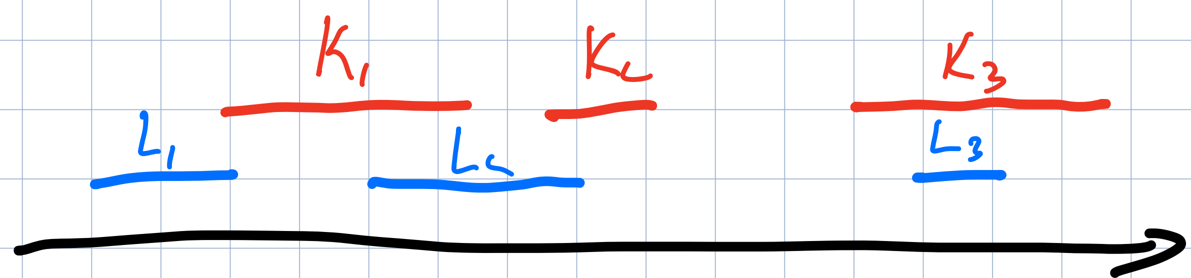
Then $|S| \geq |T|$ for any ... T.

WTS using the lemma.

Lemma: Let S be the output of EFT.
Let T be any compatible subset.
Order all jobs in S and T by EFT.
$K_1, \dots, K_{|S|}$ $\qquad$ $L_1, \dots, L_{|T|}$

Then $f(K_i) \le f(L_i)$ for all $i \le \min(|S|, |T|)$

$$f(L_1) \le f(K_1) \xRightarrow{IS} f(L_2) \le f(K_2) \xRightarrow{IS} f(L_3) \le f(K_3)$$



Proof. By induction on i.

When $i=1$, $f(K_1) \le f(L_1)$ by def. of EFT.

Now suppose $f(K_{i-1}) \le f(L_{i-1})$
then $f(K_{i-1}) \le f(L_{i-1}) \le s(L_i)$,
So L_i is compatible with $K_1, \dots, K_{i-1}$
$\Rightarrow f(K_i) \le f(L_i)$, by def. of EFT. $\square$

Theorem

Let S be the output of EFT, and M be any largest compatible subset. The $|S| = |M|$

Proof. Write $S = K_1, \ldots, K_{|S|}$ and $M = L_1, \ldots, L_{|M|}$. Suppose $|S| < |M|$.

By the lemma, $f(K_{|S|}) \leq f(L_{|S|})$. But then

$$f(K_{|S|}) \leq f(L_{|S|}) \leq s(L_{|S|+1}).$$

But then $L_{|S|+1}$ is compatible with S, a contradiction

$\square$

Theorem

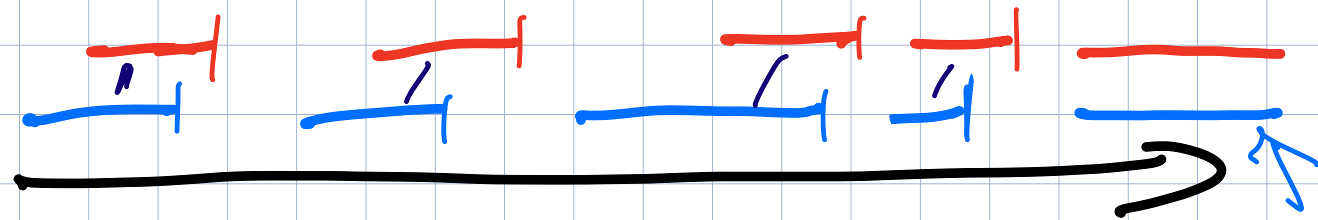
Let S be the output of EFT, and M be any largest compatible subset. The $|S| = |M|$

Proof. Write $S = K_1, \ldots, K_{|S|}$ and $M = L_1, \ldots, L_{|M|}$. Suppose $|S| < |M|$.

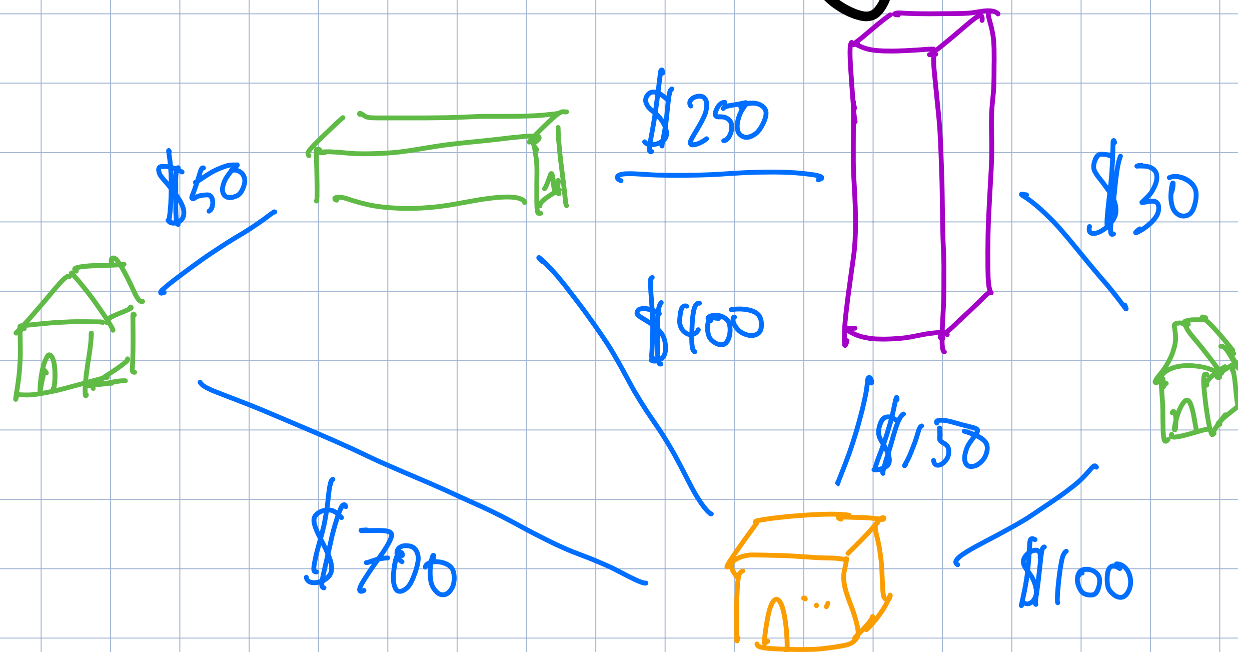
By the lemma, $f(K_{|S|}) \leq f(L_{|S|})$. But then

$$f(K_{|S|}) \leq f(L_{|S|}) \leq s(L_{|S|+1}).$$

But then $L_{|S|+1}$ is compatible with S, a contradiction $\square$



Minimum Spanning Tree



What is the minimum cost to connect all the buildings?

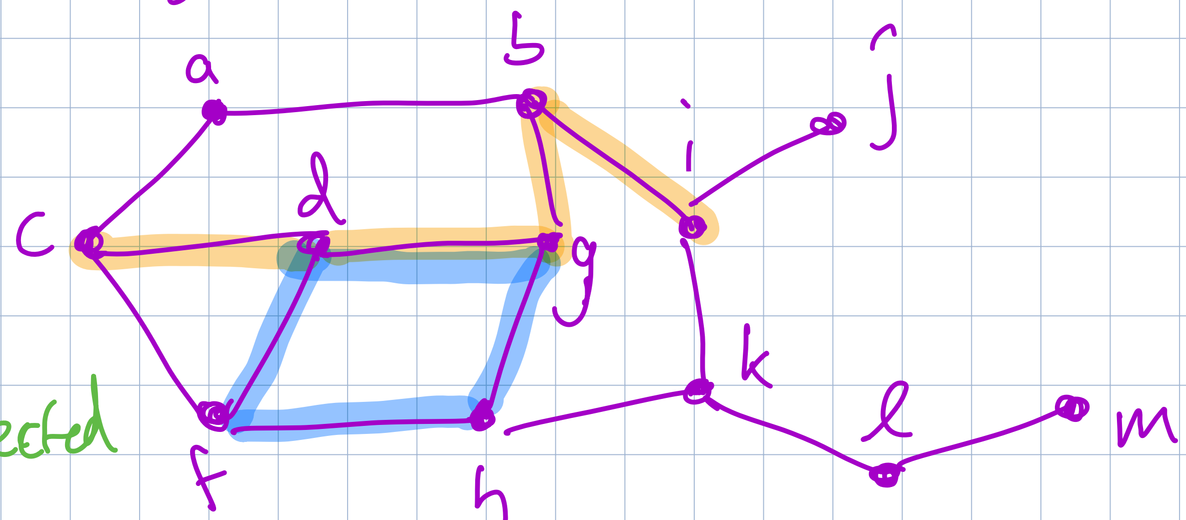
Weighted graph G.

Finding the MST for G.

Graphs

A graph $G = (V, E)$ is a vertex set V together with an edge set $E \subseteq V^2$.

A cycle is a path from a vertex back to itself.



(g, h) is an edge
$(g, h) \in E$

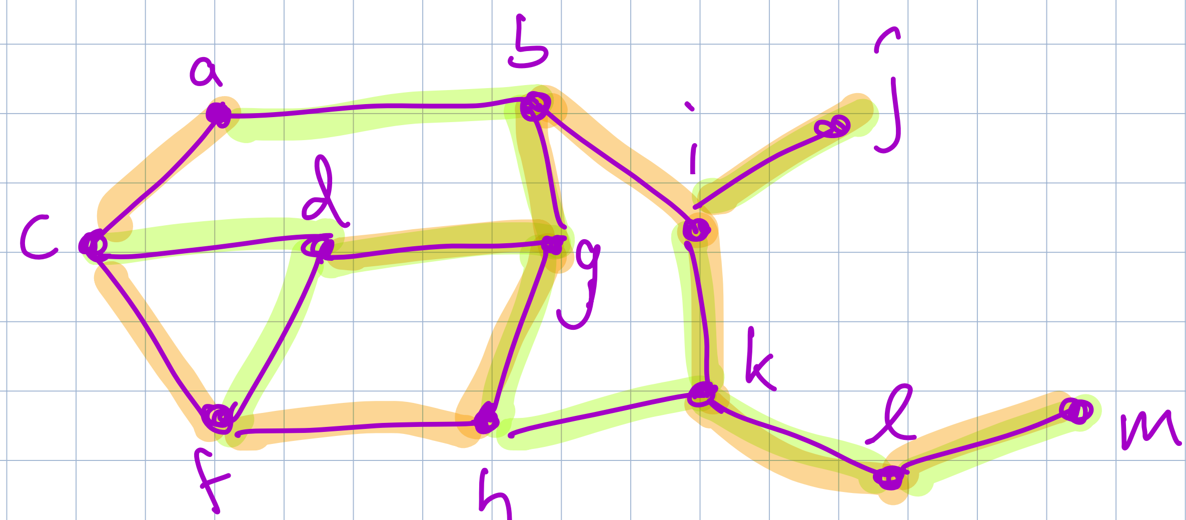
A tree is a connected graph w/o cycles.

Two vertices are connected if there is a path between them.

E.g. c is connected to i.

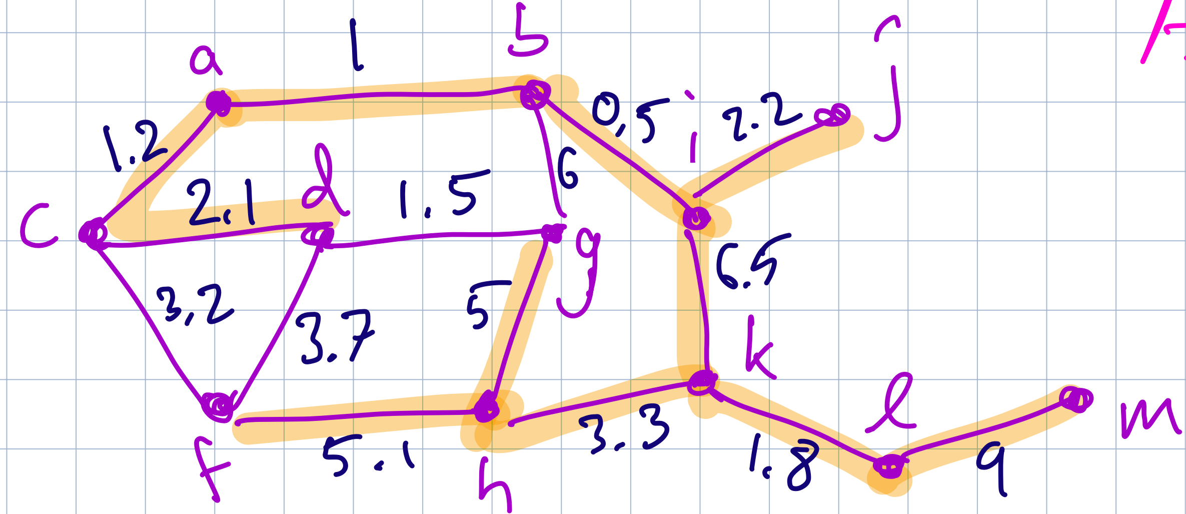
Spanning Trees

Suppose we have a connected graph G.
A <u>spanning tree</u> is a subgraph T which is a tree and contains all the vertices of G.



Minimum Spanning Trees

Start with a weighted graph G. A MST is a spanning tree with minimum weight.



Assumption: Distinct edge weights.

Greedy Algorithms for MST.

Kruskal's Algorithm.

Start with no edges.

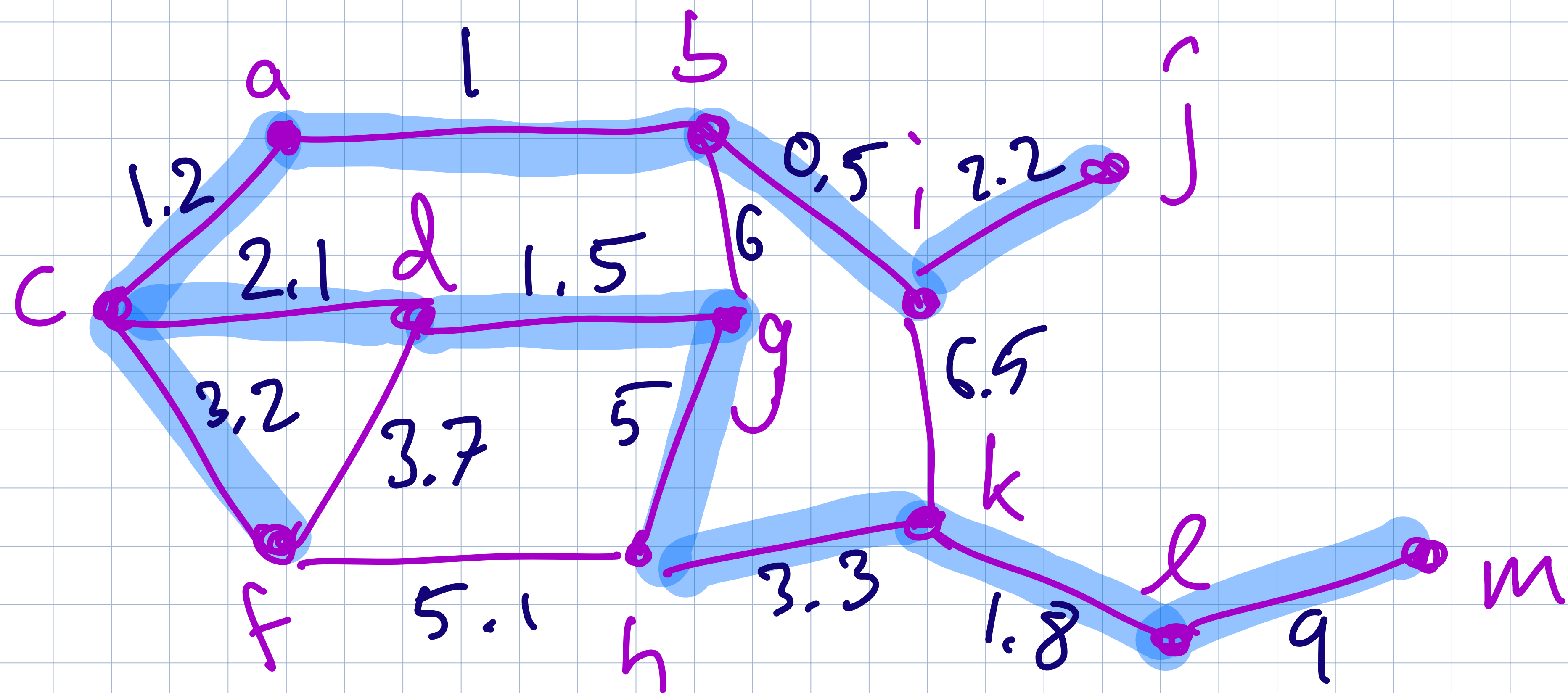
At each step add the lowest weight edge that doesn't create a cycle.

Prim's Algorithm.

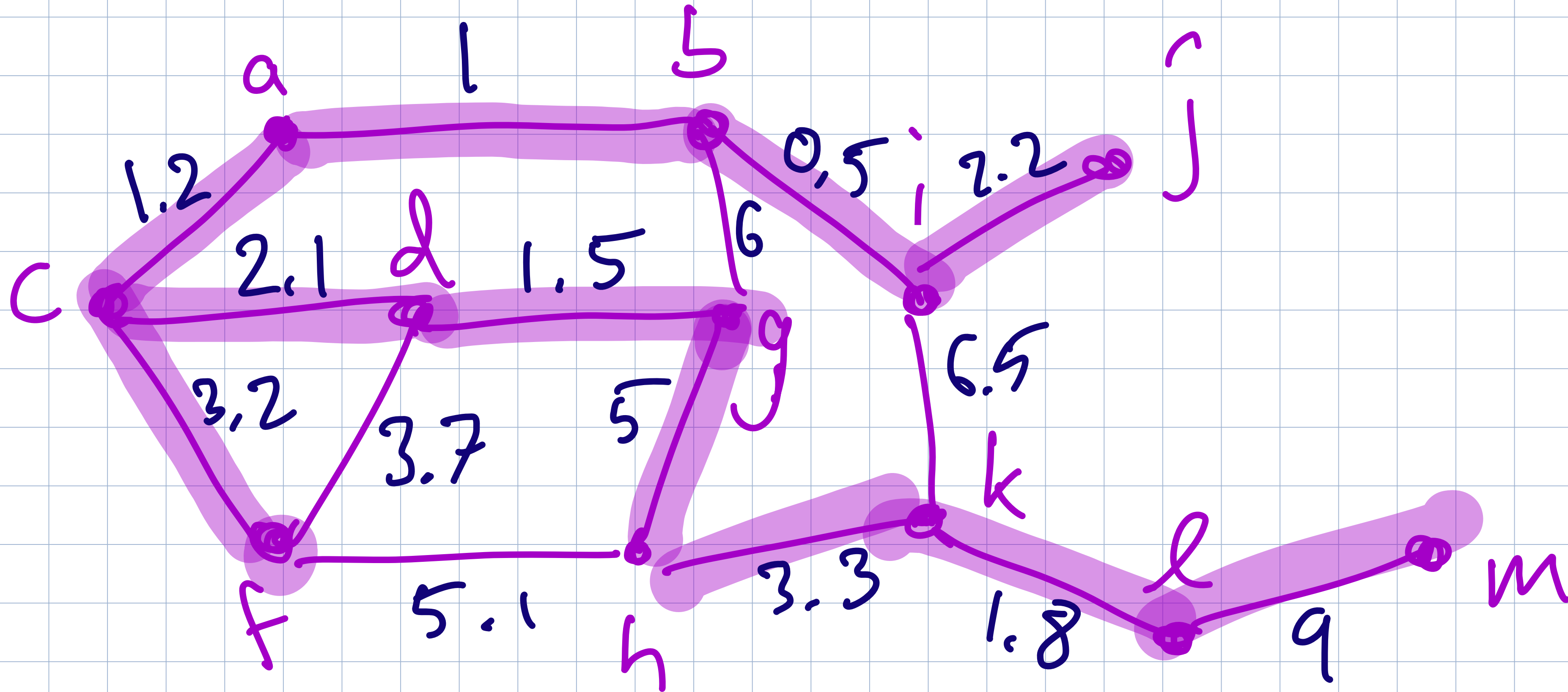
Start with an arbitrary root and no edges.

At each step, extend the ~~tree~~ tree by adding the lowest weight edge on the boundary.

Example of Kruskal's Algorithm



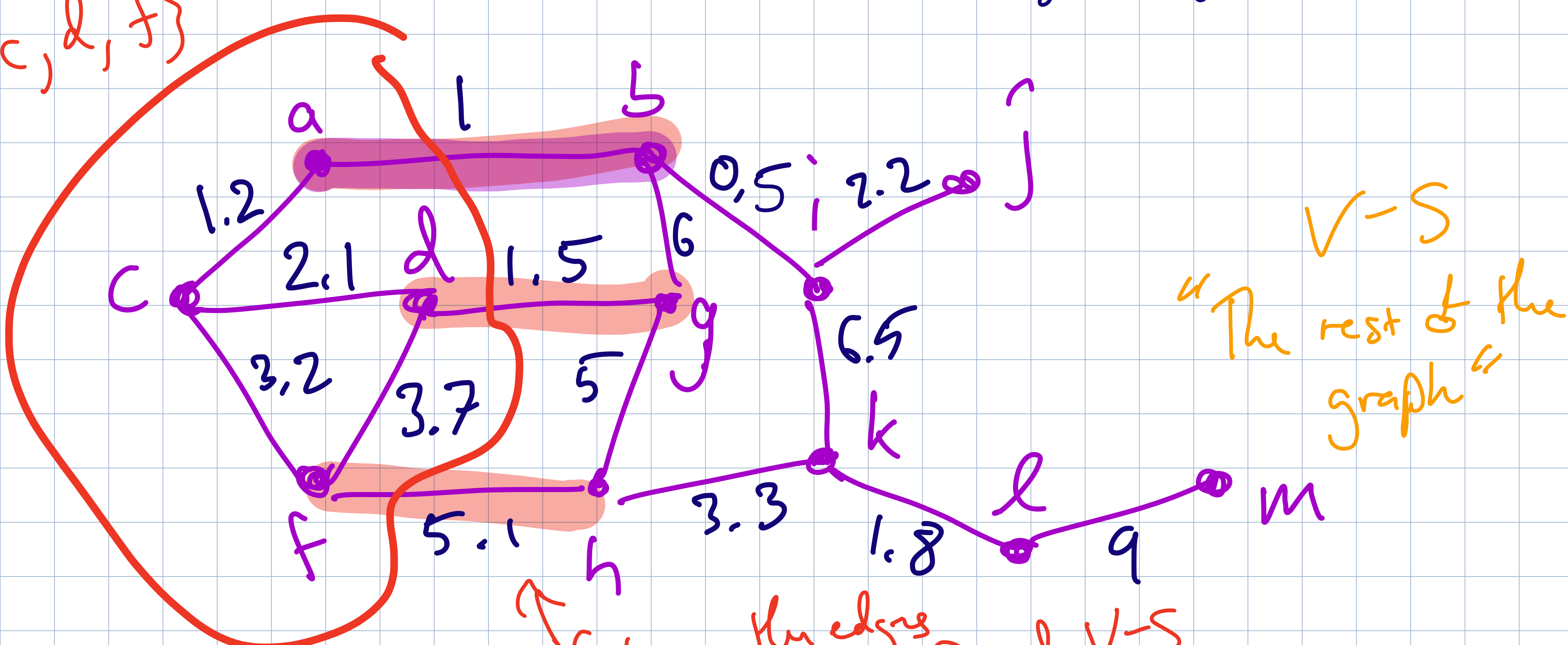
Prim's Algorithm



Why do Kruskal's and Prim's work?

Cut Lemma. For any $S \subset V$, every MST contains the lowest weight edge in the cut of S.

$S = \{a, c, d, f\}$

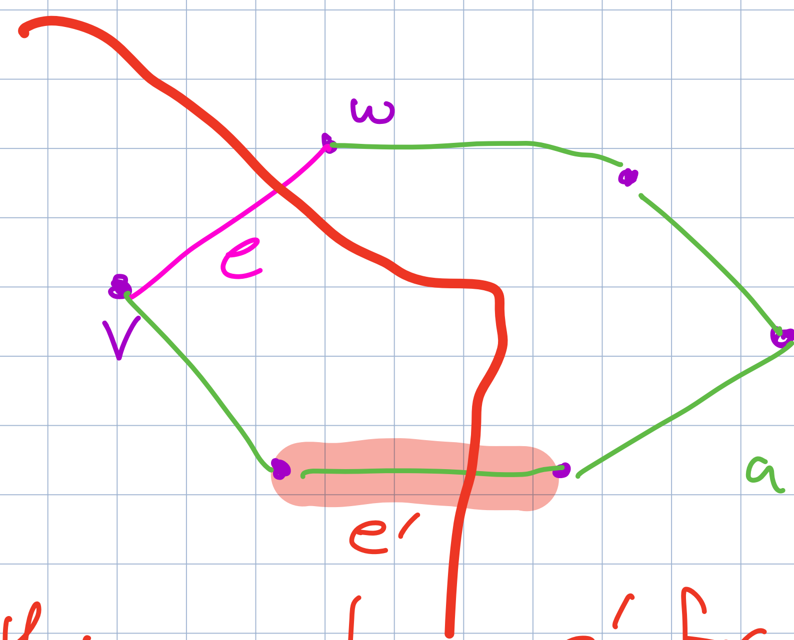


Cut are the edges between S and V-S.

Cut Lemma. For any $S \subset V$, every MST contains the lowest weight edge in the cut δS.

Proof. Let e be the lowest weight edge in the cut δS.
Let T be any spanning tree.

Suppose T doesn't contain e, WTS is that T is not a MST.



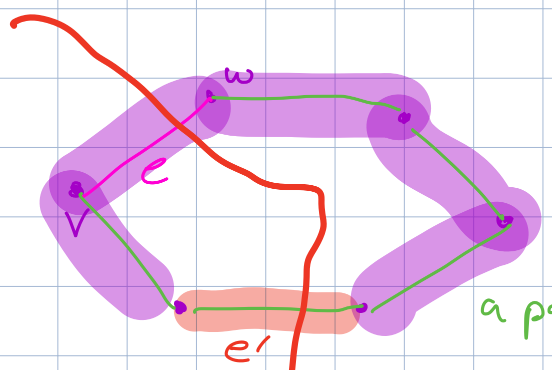
a path in T from v to w.

Idea is that we exchange e' for e. $T' = T \setminus \{e'\} \cup \{e\}$

Cut Lemma. For any $S \subset V$, every MST contains the lowest weight edge in the cut δS.

Proof. Let e be the lowest weight edge in the cut δS.
Let T be any spanning tree.

Suppose T doesn't contain e, WTS is that T is not a MST.



Idea is that we exchange e' for e. $T' = T \setminus \{e'\} \cup \{e\}$

$$w(T') = w(T) - w(e') + w(e) < w(T)$$

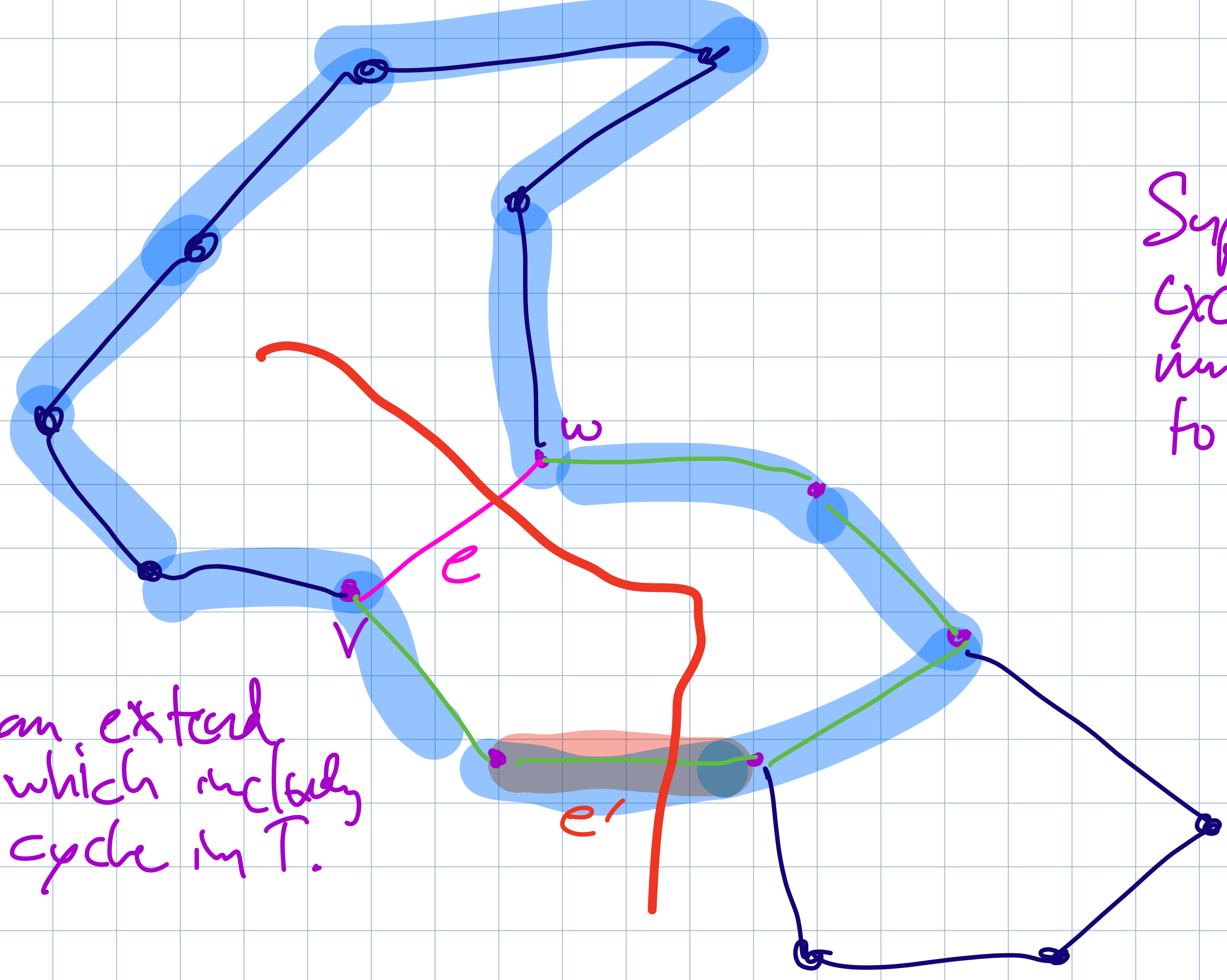
$\Rightarrow T'$ has lower weight than T

Need to show T' is a tree.

Connected because we can replace e' in any path by the rest of the cycle

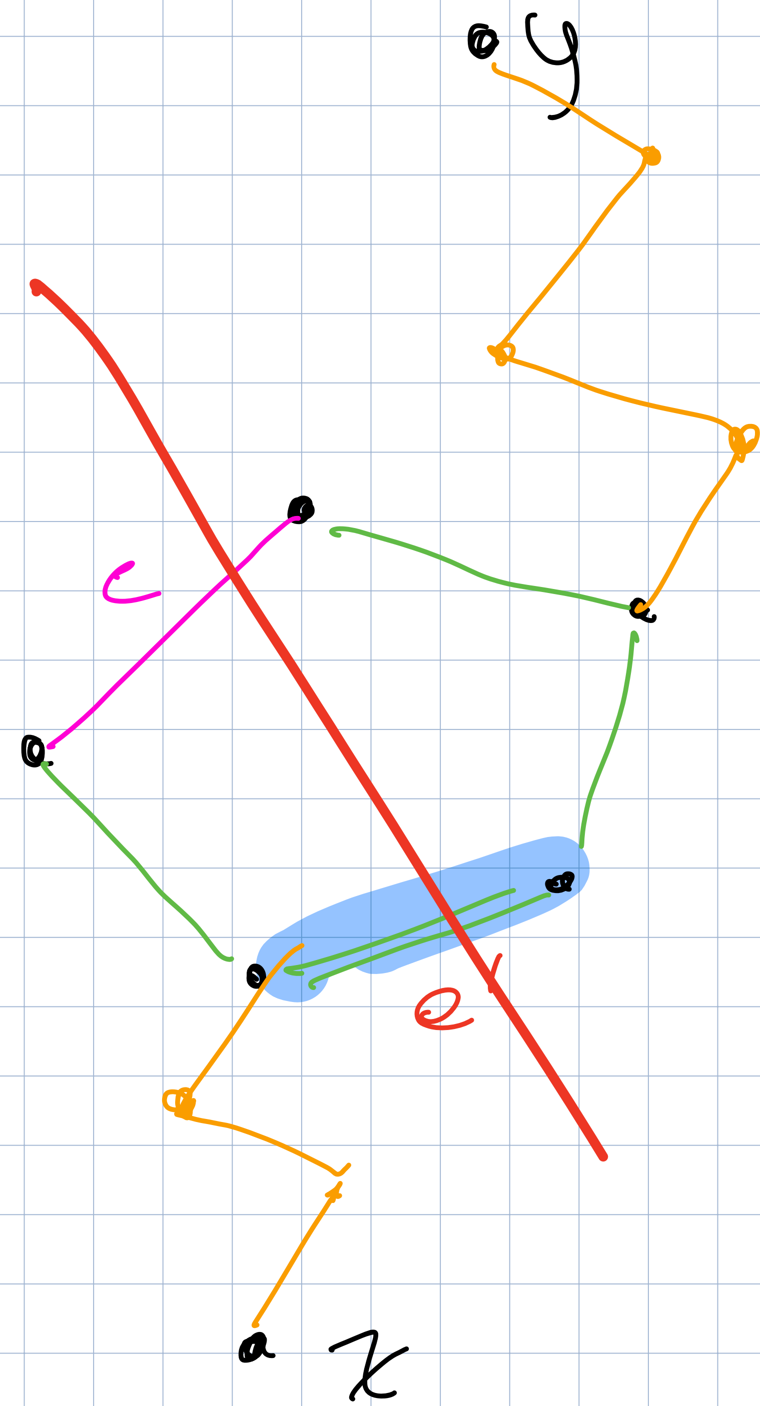
Suppose we have a cycle in T'. This cycle must include e. Can extend to a cycle in T using the same argument

Suppose we have a cycle in T'. This cycle must include e. Can extend to a cycle in T using the same arguments



We can extend a cycle which includes e to a cycle in T.

Can't happen b/c T is a tree and all these edges are in T.



WTS T' is connected

$x, y \in V$

$\exists$ a path in T from x to y