14) | Lecture 33.5-34 | MST Kruskal Algorithm | Union -Find Data Structure|

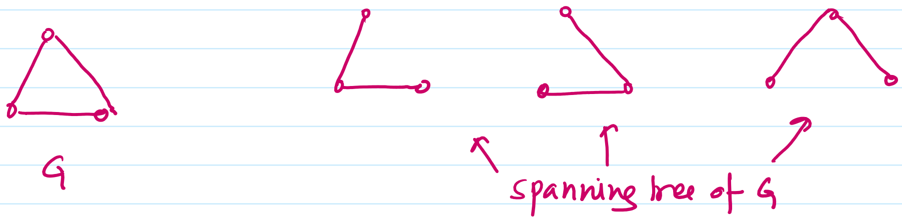
30 October 2024 15:07

$G = (V, E)$ is undirected graph

Spanning tree (i) should include all the vertices in G

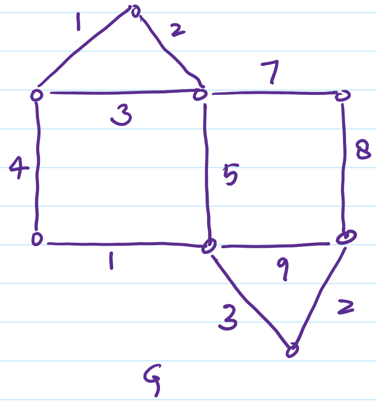
(ii) connected subgraph without cycle.

Spanning tree has $|V|-1$ edges

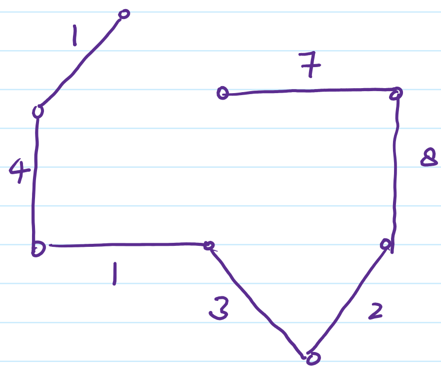


Minimum Spanning Tree

$G = (V, E)$ $\quad w : E \longrightarrow \mathbb{R}^+$



Spanning tree



Sum of wt of edge 26

Weight of spanning tree T = sum of weight of all edges in T

Minimum Spanning tree :: Spanning tree of min weight.

KRUSKAL's ALGO

Input $G = (V, E)$, $|V| = n$, $|E| = m$, $\quad w : E \rightarrow \mathbb{R}^+$

① Sort edges based on the ascending order of weight. $(e_1 \dots e_m)$

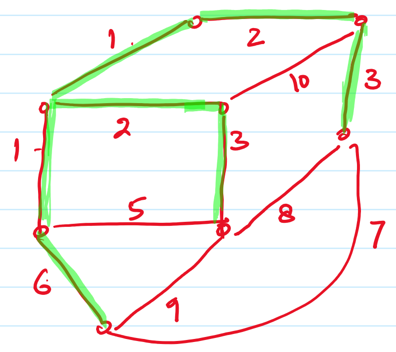
s.t $\quad w(e_i) \leq w(e_{i+1})$

② $T \leftarrow \phi$ /* T stores edges of MST

③ for $i \leftarrow 1$ to m and while (# edges(T) $\leq n-1$) do

if $e_i \cup T$ is a tree then

$T \leftarrow T \cup \{e_i\}$



Proof of correctness: KRUSKAL's output MST.

Assumption: all edge wt are distinct

KRUSKAL $g_1 < g_2 < g_3 < \cdots g_i < g_{n-1} := T_K$

$\quad\quad\quad \| \quad\quad \| \quad\quad \| \quad\quad \neq \;\; \wedge$

OPT $\underbrace{f_1 < f_2 < f_3 <}\; f_i < f_{n-1} := T_{OPT}$

$g_i :=$ wt of i^{th} edge in kruskal

$f_i :=$ wt " " " OPT

Suppose the set differ at first position at i^{th} position

Case I $g_i < f_i$

we take $T_{OPT} \setminus f_i \cup \{g_i\} := T'_{OPT}$

Case I.I T'_{OPT} doesn't contain cycle

$\Rightarrow T'_{OPT}$ is a spanning tree of weight lesser than T_{OPT}

Case (I.I) T'_{OPT} doesn't contain cycle

$\Rightarrow$ T'_{OPT} is a spanning tree of weight lesser than T_{OPT}

$\Rightarrow$ contradiction that T_{OPT} is MST.

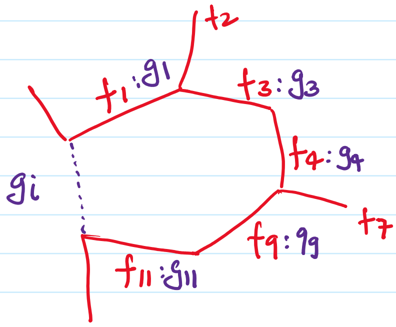
Case (I.II) T'_{OPT} contain cycle

why OPT prefer f_i over g_i — the only possible reason is adding g_i will introduce a cycle

let $f_1, f_3, f_4, f_9, f_{11}$ make cycle with g_i

$\quad\; \| \quad\; \| \quad\; \| \quad\; \| \quad\; \|$

$g_1 \;\; g_3 \;\; g_4 \;\; g_9 \;\; g_{11}$



$g_1, g_3, g_4, g_9, g_{11}, g_i$

will form cycle in Kruskal's algo

contradiction !!

Case II $g_i > f_i$

why did Kruskal not pick f_i

The only reason could be $\{g_1 \dots g_{i-1}\} \cup f_i$ form a cycle

But $\{g_1 \dots g_{i-1}\} \cup f_i \overset{\text{Identical}}{\equiv} \{f_1 \dots f_{i-1}\} \cup f_i$

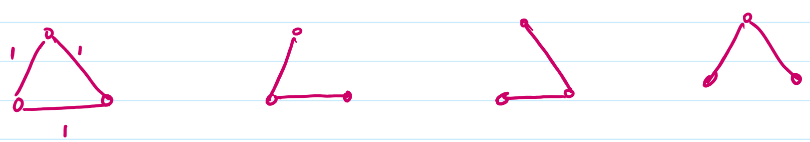
$\Rightarrow \{f_1 \dots f_{i-1}\} \cup f_i$ also form a cycle in OPT

contradiction !!

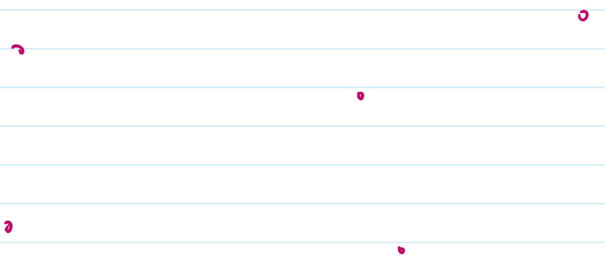
Is MST unique?

Yes if all edges are distinct

No ow.

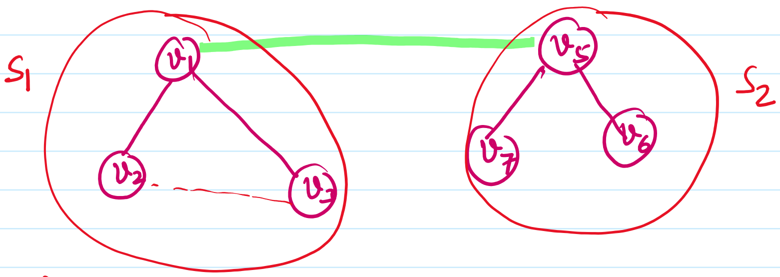


KRUSKAL'S ALGO



n connected component

- Initially we start with n connected components
- In each iteration, we add one edge that reduces # of connected component by 1
- Finally we left with one connected component — MST.



Set Union:

$\Rightarrow$ $S_1 = \{v_1, v_2, v_3\}$ $\quad$ $S_2 = \{v_5, v_6, v_7\}$

after adding edge

$\Rightarrow$ $S_1 \cup S_2 = \{v_1, v_2, v_3, v_5, v_6, v_7\}$ $\quad$ Set Union!

Find:

Suppose add edge $\{v_2, v_3\}$

Find(v_2, v_3) $\quad$ Return true if v_2, v_3 are in same set

$\Rightarrow$ inclusion of $\{v_2, v_3\}$ form a cycle

Union Find data structure

Kruskal's Algo

① $T \leftarrow \phi$

② $S = \{v_1\} \ldots \{v_n\}$

③ Sort edges in increasing order of weights

$e_1 < \ldots < e_m$ | $O(m \log m)$

④ For $i = 1$ to m and while $(\#\ \text{edge}(T) < n-1)$

do {

let $e_i = (u, v)$

if $(\text{Find}(u) \neq \text{Find}(v))$

then $T \leftarrow T \cup \{e_i\}$

Union $(\text{Find}(u), \text{Find}(v))$

}

\# onion : $n-1$ U

\# Find : $O(m)$ F

Total running time $O(m \log m + (n-1)U + mF)$

$= O(m \log m + n + m \cdot \log n)$ - $O(m \log n)$

Naive Soln   Use linked list   Union $O(1)$ time

Find $O(m)$ time   Not acceptable !!

$\{a\}$ $\{b\}$ $\{c\}$ $\{d\}$ $\{e\}$ $\{f\}$

a b c d e f $\boxed{a \mid ptr}$

$\Downarrow$

Union(Find(a), Find(b))   b c d e f $\{a,b\}$ $\{c\}$ $\{d\}$ $\{e\}$ $\{f\}$

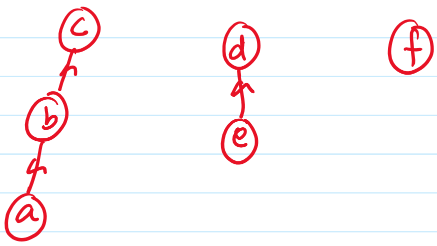
a

Union(Find(a), Find(C))   c d e f $\{a,b,c\}$, $\{d\}$, $\{e\}$ $\{f\}$

Union(Find(a), Find(c))

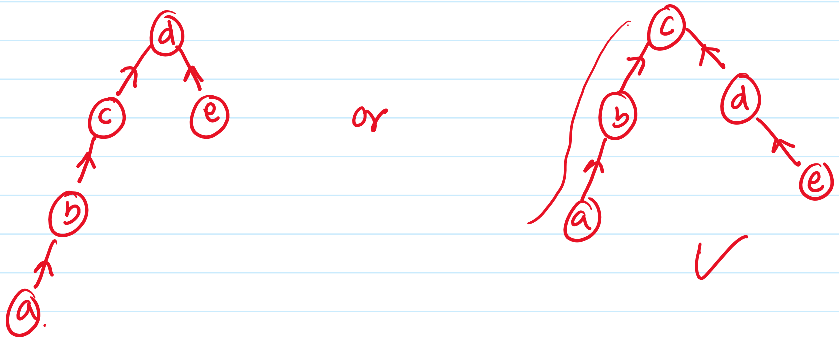
{a,b,c}, {d}, {e} {f}

Union(Find(d), Find(e))

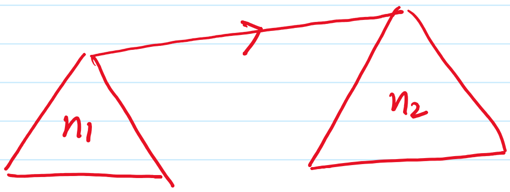


{a,b,c} {d, e} {f}

Union(Find(a), Find(d))



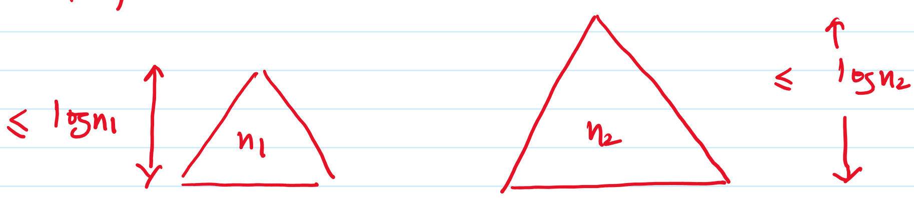
<u>Union by rank</u>

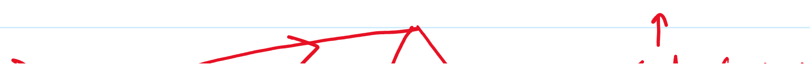


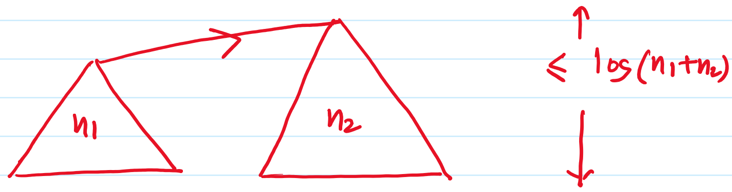
$n_1 < n_2$

<u>Th</u> A tree with n nodes has height $\leq \log n$ (if we follow union by rank procedure)

<u>Proof</u> Proof by induction





$\Rightarrow$ 

Suppose $\underline{n_1 < n_2}$

then height of resultant tree $= \max\{\log n_2 , (\log n_1) + 1\}$

$$\overset{?}{\leq} \log(n_1+n_2) \Leftarrow (1) \; \& \; (2)$$

$$\log n_2 \leq \log(n_1+n_2) \quad \text{—(1)}$$

$$(\log n_1) + 1 \leq \log n_1 + \log\left(\frac{n_1+n_2}{n_1}\right) \qquad \frac{n_1+n_2}{n_1} \geq 2$$

$$= \log(n_1+n_2) \quad \text{—(2)} \qquad \log\left(\frac{n_1+n_2}{n_1}\right) \geq 1$$