Kruskal's MST algorithm:

For input $G=(V,E)$

Sort E by $\uparrow$ weight

Let $X=\emptyset$

Go through E in $\uparrow$ order:

for edge $e=(y,z)$

if $X \cup e$ is acyclic

then add e into X

Why is it correct?

Cut property: For $G=(V,E)$,

Consider $X \subset E$ where $X \subset T$ for a MST T.

Take any $S \subset V$ where no edge of X crosses $S \leftrightarrow \bar{S}$.

Let e^* be the min wt. edge in E crossing $S \leftrightarrow \bar{S}$.

Then, $X \cup e^* \subset T'$ for a MST T'.

For Kruskal's, assume current $X \subset T$ for a MST.

Suppose we're adding edge $e = (y, z)$.

That means in X, y & z are disconnected.

Let $c(y)$ be y's component,

& $c(z)$ be z's component.

Let $S = c(y)$.

Note, no edge of X crosses $S \leftrightarrow \bar{S}$,

otherwise $c(y)$ would be bigger.

And e is the min wt. edge of E

crossing $S \leftrightarrow \bar{S}$, otherwise we would have considered that lighter edge earlier & added it to X and expanded $c(y)$.

Thus by the cut property,

$X \cup e \subset T'$ for a MST. ∎

Union-find data structure:

- collection of sets, each set corresponds to a component in the graph (V, X).
- each set has a unique ~~name~~ name which is the root of its "tree"

Operations:

- MakeSet(x): create a new set just containing x
- Find(x): return name of set containing x
- Union(x,y): Merge sets containing x & y.

O(1) time

O(log n) time

Kruskal(G, w)

for all $z \in V$, MakeSet(z)

$X = \emptyset$

Sort E by ↑ weight.

For edge $e = (y, z)$: (go through E in ↑ order)

if Find(y) $\neq$ Find(z)

then [add e to X

Union(y, z)

Return(X)

Run time:

Sorting $E \Rightarrow O(m \log n)$ time.

n Makesets $\Rightarrow O(n)$

$O(m)$ finds & unions $\Rightarrow O(m \log n)$

Total: $O(m \log n)$ time.

Union-find data structure:

Each set is a directed tree:

- edges point up to the root
- name of the set is the root

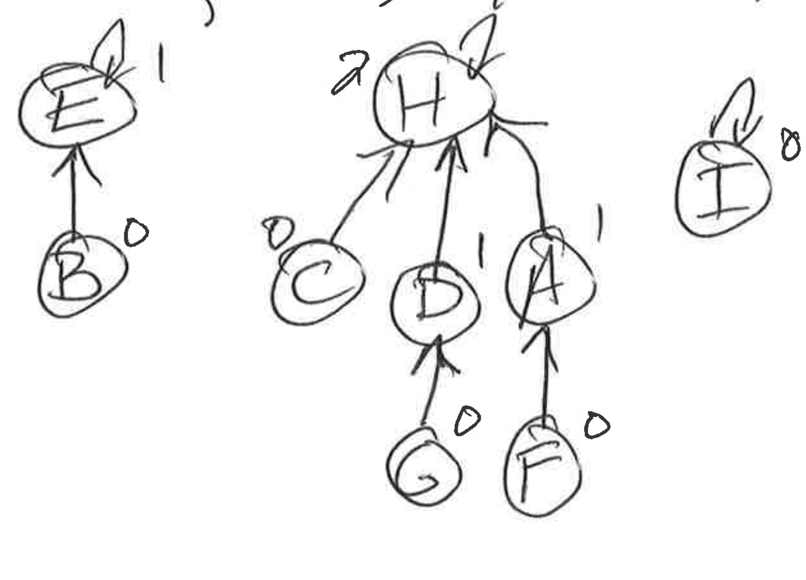
Every node also has its rank

rank(x) = height of subtree below x

$\pi(x)$ = parent of x

if $\pi(x) = x$ then x is the root.

Example: {B, E}, {A, C, D, F, G, H}, {I}



MakeSet(x):

$\pi(x) = x$
$\text{rank}(x) = 0$

Find(x):

While $x \neq \pi(x)$ do:
$x = \pi(x)$
Return(x)

To merge 2 sets, point root with smaller depth to other root

$\Rightarrow$ minimizes ~~dep~~ max depth.

Union(x, y)

$r_x = \text{find}(x)$
$r_y = \text{find}(y)$
if $\text{rank}(r_x) > \text{rank}(r_y)$ then $\pi(r_y) = r_x$
if $\text{rank}(r_y) > \text{rank}(r_x)$ then $\pi(r_x) = r_y$
if $\text{rank}(r_x) = \text{rank}(r_y)$ then
$\pi(r_x) = r_y$
$\text{rank}(r_y)$++

Key claim: max depth is $\leq \log n$.

Claim 2: root of rank k has $\geq 2^k$ nodes
in its subtree (including itself).

Proof of claim 2:

~~Let~~ induct on k.

Base case: $k=0$: count node itself
so $2^0 = 1$ ✓

assume true for nodes of rank $\leq k-1$.

to get node of rank k,
merge 2 nodes of rank $= k-1$,
each has $\geq 2^{k-1}$ nodes by induction
So new subtree has $\geq 2 \times 2^{k-1} = 2^k$. ✓

Proof of key claim:

Let l be # of nodes of rank $= k$.

Then, $(l)(2^k) \leq n$

so $l \leq n/2^k$

let $k = \log_2 n + 1$, Then $l \leq \frac{1}{2} < 1$ so no nodes of rank $\log n + 1$. ∎