Last class:

Cut Property:

For $G=(V,E)$, consider $X \subset E$ where $X \subset T$ for a MST T.

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

Let e^* be the edge of G of min weight that crosses $S \leftrightarrow \bar{S}$.

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

Prim's algorithm:

R = explored vertices.

X is part of a MST on R.

Let $S=R$ & take the e^* of min weight out of S.

Say $e^* = (y,z)$ where $y \in R$, $z \notin R$.

Add e^* to X and z to R.

Repeat until $R=V$.

Kruskal's algorithm:

Greedy approach:

For input $G=(V,E)$

Sort E by $\uparrow$ weight

Let $X=\emptyset$

Go thru E in $\uparrow$ order:

For edge $e=(y,z)$

if $X \cup e$ is acyclic

then add e into X

How do we test if $X \cup e$ makes a cycle?

In the graph (V,X), are y & z in the same component?

if so, then $X \cup e$ makes a cycle

if in dif't. components then adding it is ok.

Use union-find datastructure to check if y & z are in the same or different components.

Why is Kruskal's algorithm correct?

Let $G' = (V, X)$.

Let $c(y)$ be the vertices in the component containing y in G'

Let $S = c(y)$.

Note: $e = (y, z)$ is the min weight edge of G crossing $S \Leftrightarrow \overline{S}$.

Why? Suppose $e' = (a, b)$ has $a \in S$, $b \notin S$ & $w(e') < w(e)$.

Then e' is considered by Kruskal's alg. before e & it will get added into X since a & b are in dif't. components. But then $b \in S$ which is a contradiction.

Hence, by the cut property adding e to X we're still on our way to 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
  - the name will be the root vertex.

Operations:

Makeset(x): create a new set just containing x

Find(x): What is the name of the set containing x?

Union(x, y): Merge the sets containing x & y.

$O(\log n)$ time per Find, Union

$O(1)$ time per Makeset

To check if y & z are in the same or different components, we just check: is find(y) = find(z)?

When adding $e = (y, z)$ into X, do Union(y, z) to merge their components.

Kruskal(G,w):

input: connected, undirected $G=(V,E)$ with edge weights $w(e)>0$ for $e\in E$

output: MST defined by $X\subset E$

for all $v\in V$, Makeset(v)

$X=\emptyset$

Sort E by ↑ weight.

Go through edges by ↑ weight:

for $e=(y,z)$,

if $\text{find}(y)\neq\text{find}(z)$

then add e to X

Union(y,z)

Return(X)

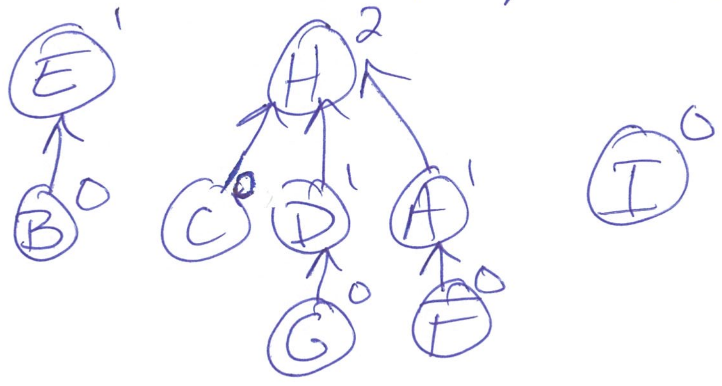
Running time: $O(m\log n)$ time.

Union-find data structure:

Each set is a directed tree:

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

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



Each node x has 2 values:

1) $\pi(x)$ = parent of x
if x is the root
then $\pi(x) = x$

2) rank(x) = height of subtree below x.

Makeset(x):

$\pi(x) = x$

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

Find(x):

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

Return(x)

To merge sets containing x & y, Point root of one to root of other.

Key: to minimize depth, Point root with smaller depth to larger depth.

So root with smaller rank points to larger rank.

Union(x,y):

$a = \text{Find}(x)$
$b = \text{Find}(y)$
if $\text{rank}(a) > \text{rank}(b)$
then $\pi(b) = a$
if $\text{rank}(b) > \text{rank}(a)$
then $\pi(a) = b$
if $\text{rank}(a) = \text{rank}(b)$
then $\pi(b) = a$
$\text{rank}(a)$++

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

Hence, find & union take $O(\log n)$ time.

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

Proof: By induction on k.

Base case: $k=0$: $2^0=1$ & it includes itself.

Assume it's true for rank $<k$.

Consider node of rank k.

It was formed by union of a & b of rank $k-1$.

Both a & b have $\geq 2^{k-1}$ nodes in their subtree's.

So $\geq 2 \times 2^{k-1} = 2^k$ in the union. ∎

Let l = # of nodes of rank k.

By claim 2, $l \times 2^k \leq n$

$$l \leq n/2^k$$

let $l = (\log n) + 1$

Then, $l \leq \frac{1}{2} < 1$ so 0 nodes of rank $> \log n$. ∎