**Disjoint Set Data Structure**

**Items:** $X$

**Maintain Sets** $S_1, \ldots, S_k$

**Disjoint** $S_i \cap S_j = \emptyset \quad \forall i \neq j$

**Make Set** $(x) -$ create a one element set

**Find** $(x) -$ return the "name" of the set containing $x$

**Union** $(x,y) -$ put together $x$'s and $y$'s sets
A set as a tree w/ nodes point to points,
name of set = root
\{a, b, c, d\} \{e, f, g, h\} \{i, j\}

Time for Find \(= O(\text{ht of tree})\)

Unions make trees grow.
Union by rank

Make the tree w/ fewer nodes a child of the tree w/ more nodes
V makesets
V unions
E finds

How big does the height of the tree get?

In order to increase the height of a tree, the nodes have to at least double.

Look at node $u$, how many times can it be in the tree that is on the small side of a union?

Between two instances of being on the small side of a union, your tree size must at least double. \( \Rightarrow \) only on small side < $\lg U$ times

$\Rightarrow \text{height} = O(\lg U)$
Kruskal

\[ \text{Sort } O(E \lg V) \]

\[ \rightarrow O(E \lg V) \text{ rest of alg.} \]

Path compression

Wherever you touch a node, make it a child of the root.
Example

![Graph Image]

Running time = \( O(1g^xV) \)

of one op. (height tree)