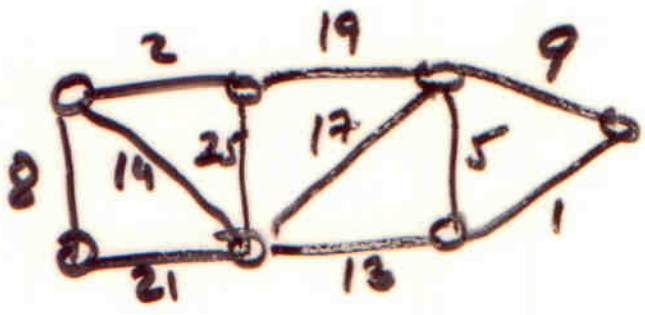
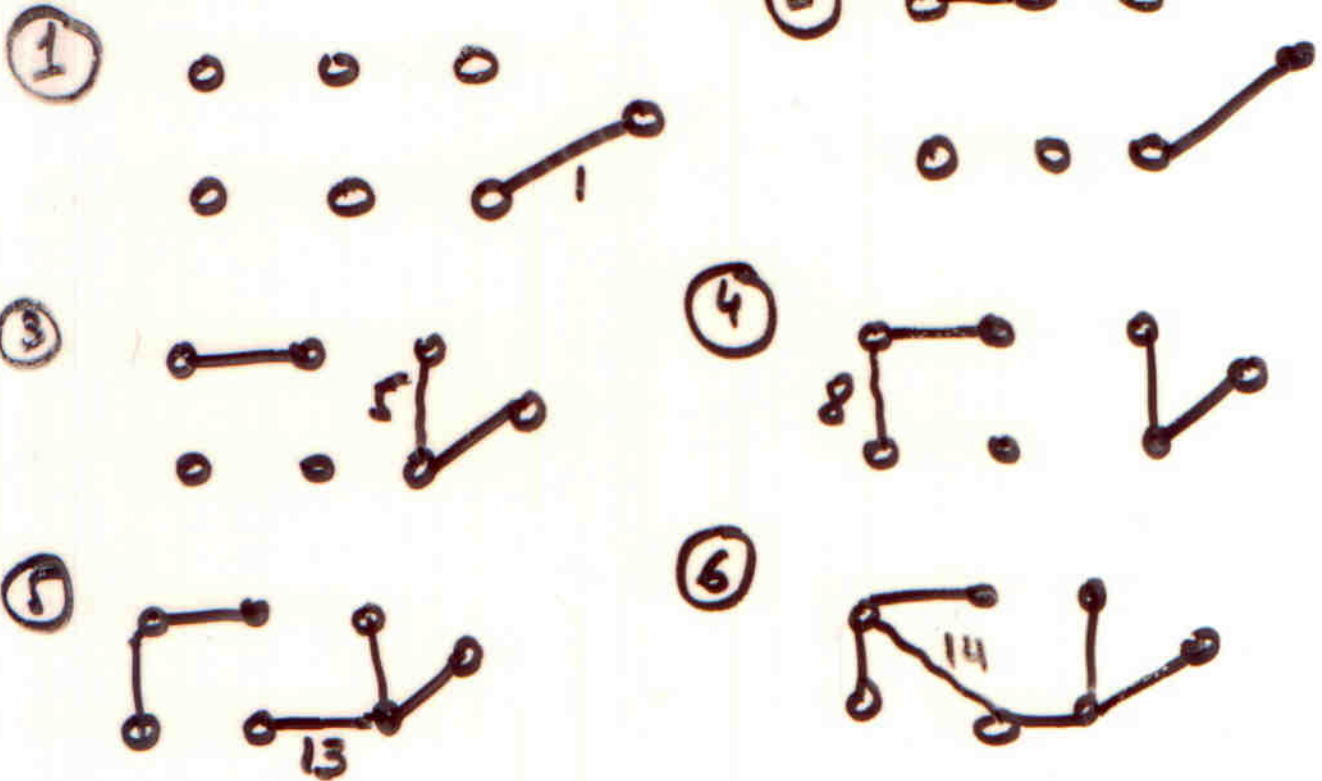


Kruskal's algorithm ^{for finding MST} - start with n trees each of 1 node (forest) at each step merge 2 trees by adding min. weight edge that connects 2 separate trees.

Ex.



Kruskal's:



We need to keep track of which nodes are in which tree.

Disjoint-Set Union Problem

(Chpt. 21.1-21.2 of CLRS book)

We have disjoint sets S_i , i.e., $S_i \cap S_j = \emptyset$.
So we have a set

$$\mathcal{S} = \{S_i\}.$$

Operations we need to be able to do:

make-set(x) - creates a new set whose only element is x :

$$\mathcal{S} \leftarrow \mathcal{S} \cup \{x\}$$

union(S_i, S_j) -

$$\mathcal{S} \leftarrow \mathcal{S} \setminus \{S_i, S_j\} \cup \{S_i \cup S_j\}$$

find-set(x) - returns $S_i \in \mathcal{S}$ such that $x \in S_i$.

Kruskal's Alg.

$T \leftarrow \emptyset$ (tree being built)

for each $v \in V$
do make-set(v)

sort E by increasing edge weight w

for each $(u, v) \in E$ (in sorted order)

do if find-set(u) \neq find-set(v)

then $T \leftarrow T \cup \{(u, v)\}$

union(find-set(u), find-set(v))

Sorting takes $\Theta(E \log E)$ time.

$\Rightarrow \Theta(E \log E) = \Theta(E \log V)$ because

$$\log V \leq \log E \leq \log V^2$$

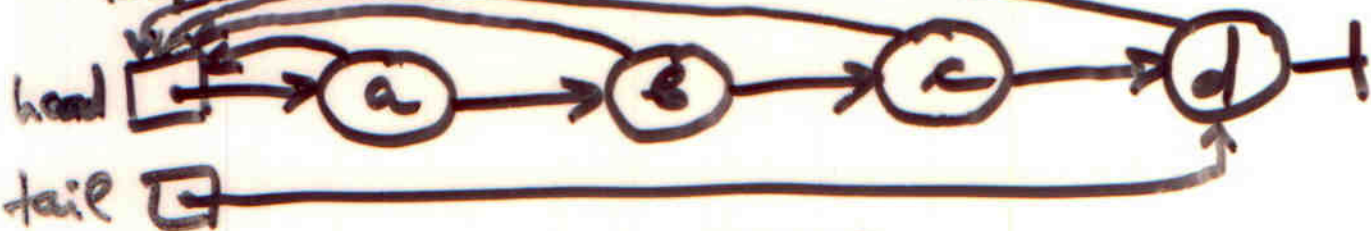
$$\log V \leq \log E \leq 2 \log V.$$

We do:

- $O(V)$ make-sets
- $O(E)$ find-sets
- $O(V)$ unions

How to implement this data structure?

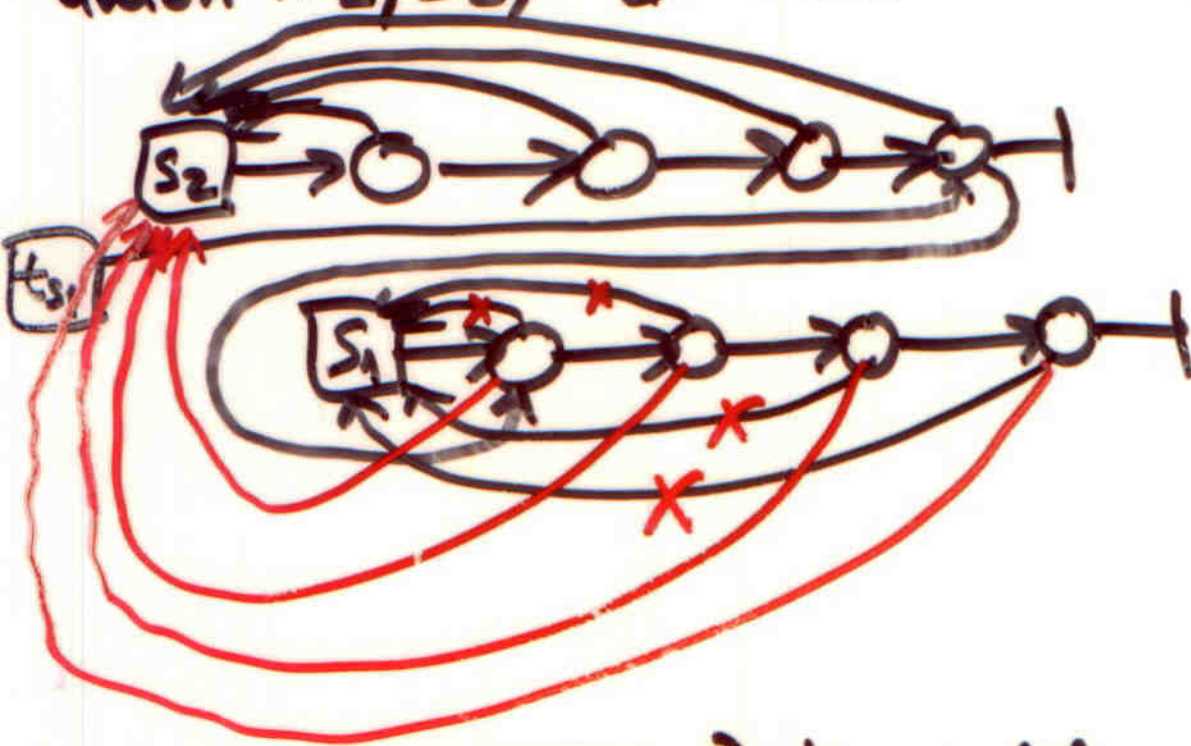
- Represent each set by a linked list.



Make-set takes $O(1)$.

Find-set takes $O(1)$.

Union (S_1, S_2) & result is new set S_2 .



This takes $O(|S_1|)$ time (if you keep "tail" pointers).