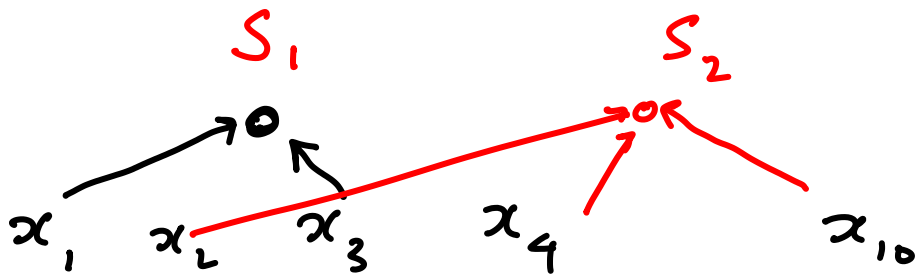
COL 702 Lecture 15 Sept 18

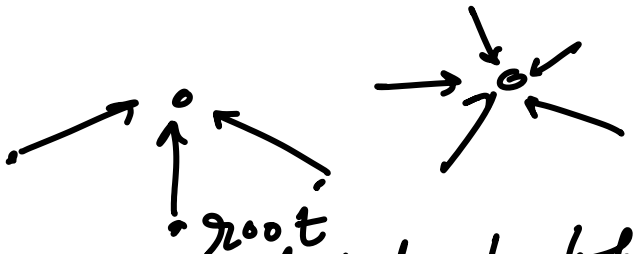
Union Find data structures

m Finds and n unions can be done in $O(m + n\log n)$ time using an array based data structure

Tree based representation of sets



Star like structure



Find(x) reports the label of which x is pointing $O(1)$ time

Union : We can link the root of one tree to the other or create a new root and link both roots to this

Consequence : no longer stars

Distance to root increases

Find becomes more expensive

is the distance to the root

How do we have every bound on the distance of an element to the root?

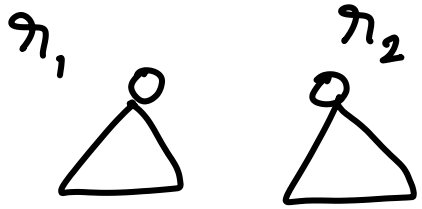
We will maintain some kind of depth/rank information with every tree. The rank function is defined as 0 for a singleton element

When we union two trees with ranks r_1 and r_2 where $r_1 < r_2$ then we make the root with rank r_1 point to the root with rank r_2

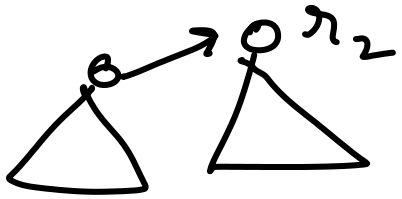
else if $r_1 = r_2$ we can choose to link the roots arbitrarily.

Then rank increases by 1.

rank = 0



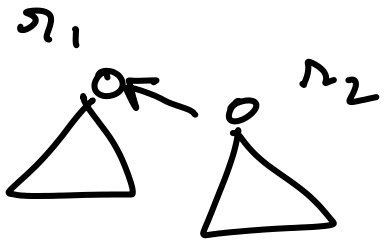
$r_1 < r_2$



rank = r_2

$r_1 = r_2$

Union by rank heuristic



rank = $r_1 + 1$

<u>Claim</u>: The minimum no. of nodes in a tree with rank $r \geq 2^r$

$\Rightarrow$ The max rank $\leq \log_2 n$

$\Rightarrow$ Find will be bound by $O(\log_2 n)$

Proof (by induction on rank)

rank = 0 by defn

Suppose it is true for all ranks $< i$

$\Rightarrow$ when we union two trees, T_1, T_2 with ranks l, k $\quad l, k < i$

then T_1 has at least 2^l nodes

T_2 " " " 2^k nodes

Rank after union

Case 1 $\quad l < k$ $\quad$ Total no of nodes $2^l + 2^k$ and rank is k

Case 2 $\quad l = k$ $\quad$ rank is $k+1$

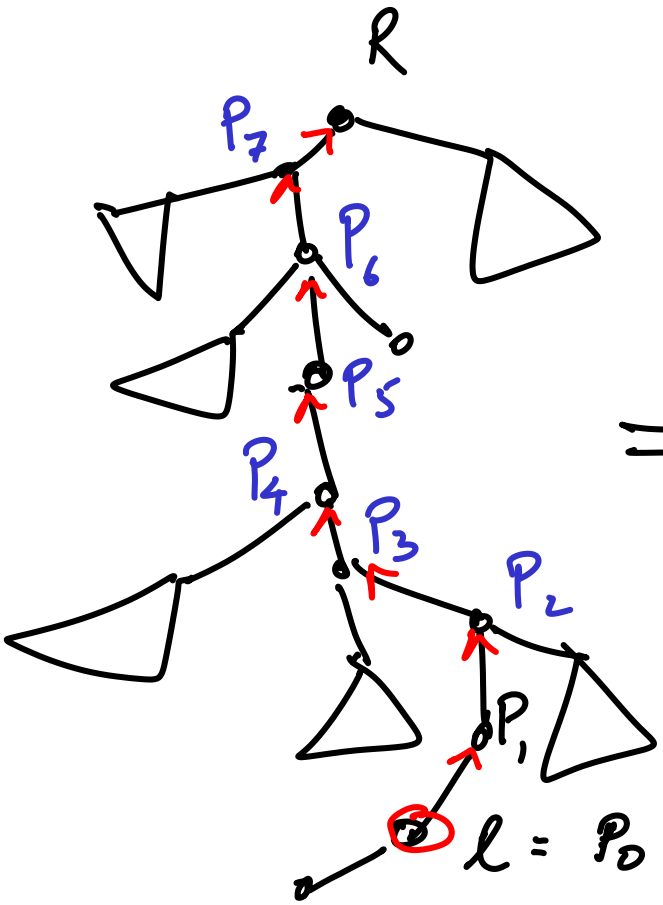
$$\# \text{ nodes} \geq 2^k + 2^k = 2^{k+1}$$

Using tree repr with union heuristic cost of m finds and n union : $O(m \log n + n)$

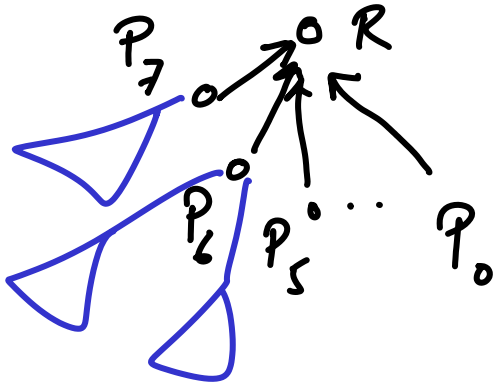
and every individual operation cost is bound by $O(\log n)$

(O(m) for array based)

We can improve the performance of the tree based data structure by using another heuristic called "path compression"



Make every node in path from L to R a child of R



By path compression heuristic we will decrease the distance of many nodes to the root node

The cost of m unions and n finds by using rank and path compression heuristic is $O((m+n)\log^* n)$

$\log^* n$: log star n function

$\log^* 2 = 1$

$\log^* k$ where $k \leq 2^i$

$= \log^* i + 1$

The $\log^*$ function is related to an inverse Ackermann