Lecture 19: 19th March

List Ranking

- given LL find distance of elements from one end of list

→ slow in parallel

Storing LL in parallel program

- using successor pointers
- using an array of successors S.

$|S| = |LL|$

S[i] = next of i in LL

S[last element] = -1

eg.

LL: 1 → 5 → 2 → 6 → 4 → 3

	1	2	3	4	5	6
S:	5	6	-1	3	2	4

head = value missing in S

→ Start loop : S[head] * fill S

Ranking

Rank [i] = j if i is jth element of LL

R:

1	3	6	5	2	4
1	2	3	4	5	6

i.e. Rank is # elements incl. me to my left.

Pointer Jumping (Wyllie 1980)

head → 2 → 5 → 6 → 8 → 3 → 1 → 7 → 4 ↻

Find who is end node

for every u ← in parallel
while S[u] ≠ S[S[u]]:
S[u] = S[S[u]]

2 → 6 → 3 → 7 }
5 → 8 → 1 → 4 } after one iteration

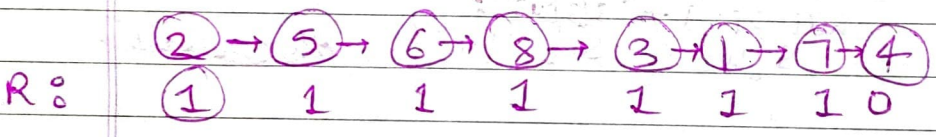
2 → 3 → 4
6 → 7 → 4
8 → 4
5 → 1 → 4 } second

5 → 4
1 → 4
8 → 4
2 → 4
7 → 4
3 → 4
6 → 4 } third!

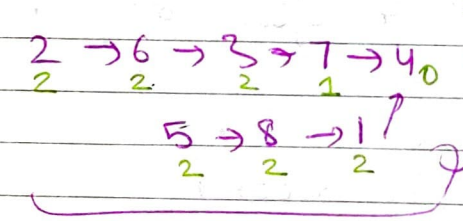
Extend this algo to find Ranks from end.

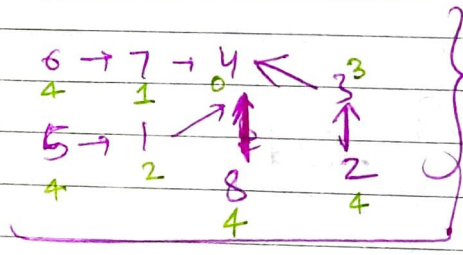
Algo FindRoot
for $1 \le i \le n$ do in parallel
$R(i) = 1$
$R(i) = 0$ if node i is last node
while $S[i] \neq S[S[i]]$ do
$R[i] = R[i] + R[S[i]]$
$S[i] = S[S[i]]$
end
end

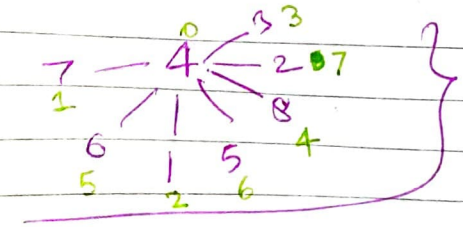
my rank + my parents rank



Rank







→ In PRAM → sync execution – all n processors execute each step in while loop at same time

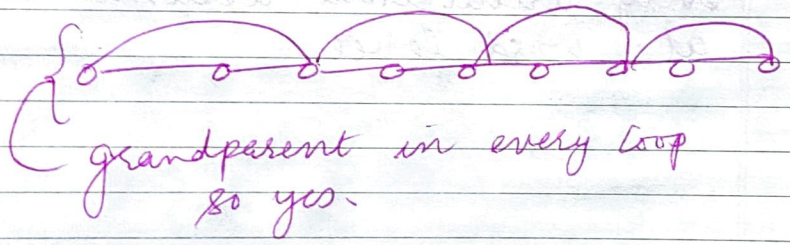
Trouble 1

→ Use synchronisation primitives to avoid sync issues

Claim: Algo Find root finishes in $O(\log n)$ time

Proof: Show that the distance b/w a node and root reduces by a factor of 2 in every iteration of while loop.

↳ maximum distance is n



grandparent in every loop so yes.

Claim = Work = $O(n \log n)$

In sequential time comp = $O(n)$

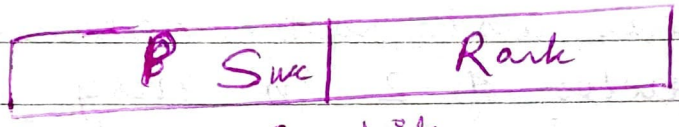
so, sub-optimal algo

Trouble 2 = Inconsistent Update

↳ so, in while loop either both statements should get executed for a given i or none.

So, consider packing R and P values of a node into a single word

If list has no more than 2^{32} elements, we use 64 bit architectures with each word packing two 32 bit numbers.



64 bits

every architecture assumes atomicity at word level