CSCI 429 RMQ → LCA

09-13

RMQ (range minimum queries)

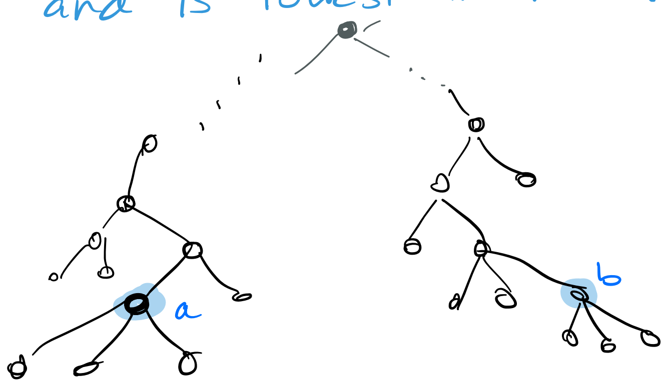
Given: Array $A[1..n]$, ℓ, r, $1 \le \ell \le r \le n$

Find: index i, $\ell \le i \le r$, such that $A[i]$ is min in range $A[\ell..r]$.

LCA (Lowest Common Ancestor)

Given: Rooted tree T, nodes a, b

Find: node that is ancestor of both a and b and is lowest in tree.



Want:
$O(1)$ query
$O(n)$ space

Assume: Tree is static

Possible project:

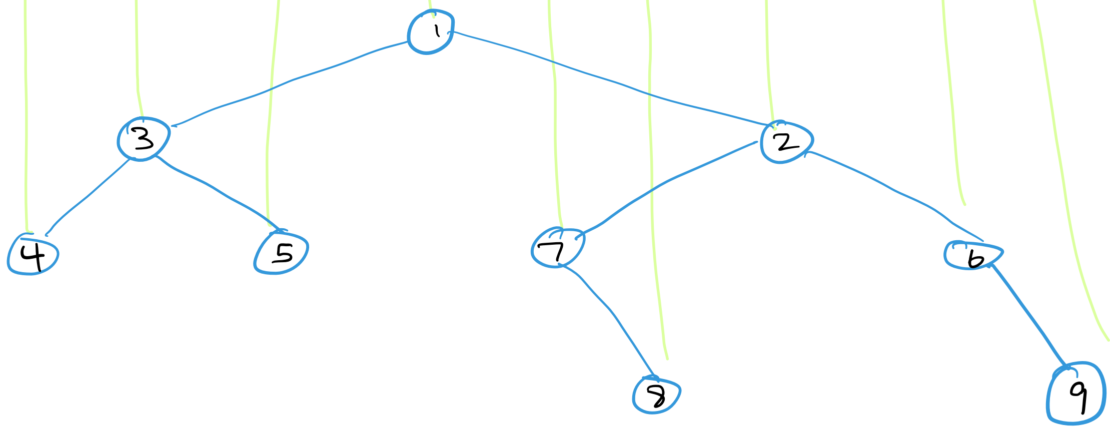
- Read and teach us the dynamic version (Cole & Hariharan, 1984) see notes on weekbyweek

RMQ reduces to LCA

- Make the Cartesian Tree
- a = node for $A[l]$
- b = node for $A[r]$
- find LCA — that is the Range Minimum.

(we can do LCA on *any* rooted tree, but when we do it on Cartesian trees, we get the Range minimum)

	1	2	3	4	5	6	7	8	9
A =	4	3	5	1	7	8	2	6	9



RMQ(6, 8) = 7
RMQ(5, 6) = 5
RMQ(1, 3) = 2
RMQ(3, 6)

If we can solve LCA in O(1),
we can solve RMQ in O(1).

Lowest Common Ancestor (LCA)

Want: $O(1)$

Method: Reduce to RMQ. (what ???)

... a special case.

RMQ$\pm$1 = adjacent values in array can only differ by ± 1

Reduce LCA to RMQ$\pm$1:

Given: rooted tree T

Produce: an instance of RMQ$\pm$1 so that RMQ$\pm$1 queries give us the answers to our LCA queries on T.

Step 1: conduct Euler tour of T to construct list of node depths.

4 3 5 1 7 8 2 6 9

1

Sept 18 '23

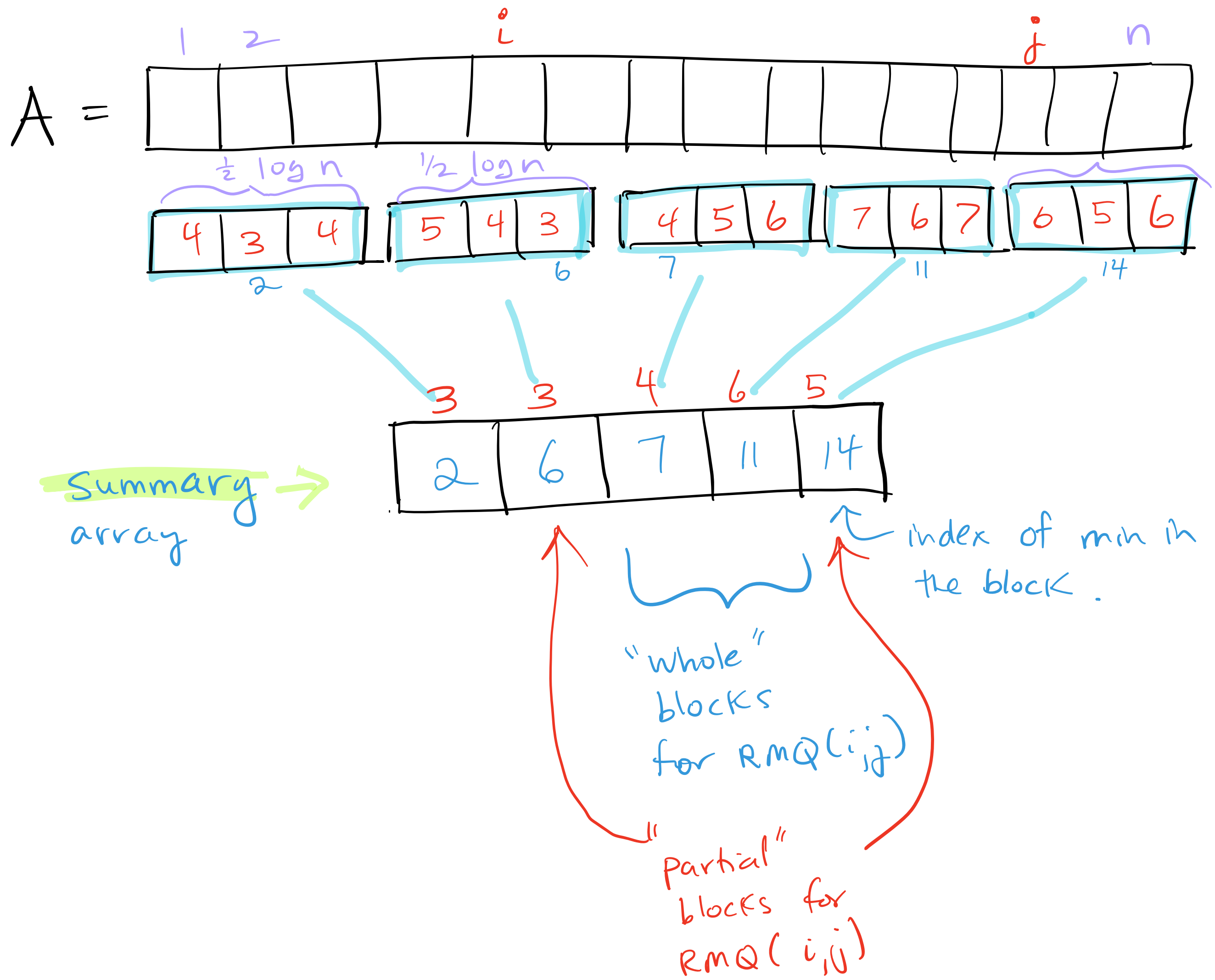
Observation: the new array is ± 1, so if we can query $RMQ^{\pm 1}$ quickly, what will that accomplish for us?

Claim: The $RMQ^{\pm 1}$ on the Cartesian-depth array is the LCA of the original T.

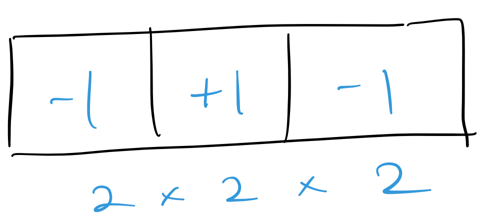
Proof: for you to do.

[Hint: I would try structural induction]

Solve RMQ$^{\pm 1}$ in $O(1)$ time.



Lets look at a block.



how many can there be?

Indeed, mapping $0 \to -1$

$1 \to +1$

we have simply an alphabet-change of the set of all bitstrings of length $\boxed{\frac{1}{2} \lg n}$.

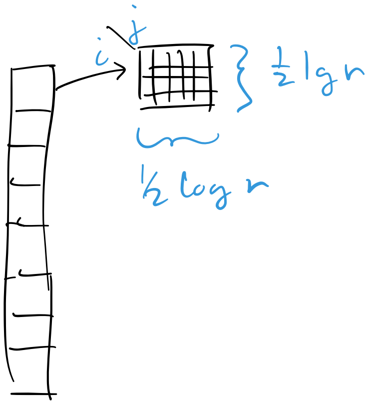
$\Rightarrow \exists$ distinct block types

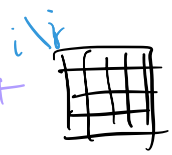
[Note: This is true only for RMQ± 1

- what is it for "general" RMQ?]

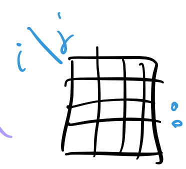
Boundary blocks

Have a Look-up Table for different block types.



Find correct 

: $O(\quad)$

Find correct cell in 

: $O(\quad)$

Size of this DS: $O(\quad)$

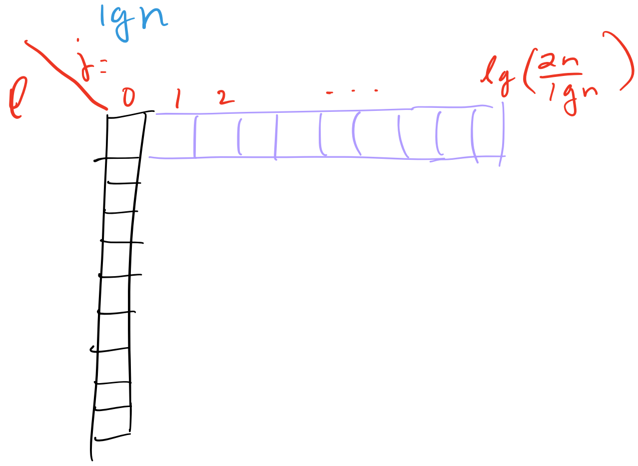
The range of whole blocks :

The Summary Array

- Build a Sparse Table for the Summary Array.
- Values are not necessarily RMQ± 1-type.

$n/\frac{1}{2}\lg n$ blocks $= 2\frac{n}{\lg n}$ blocks

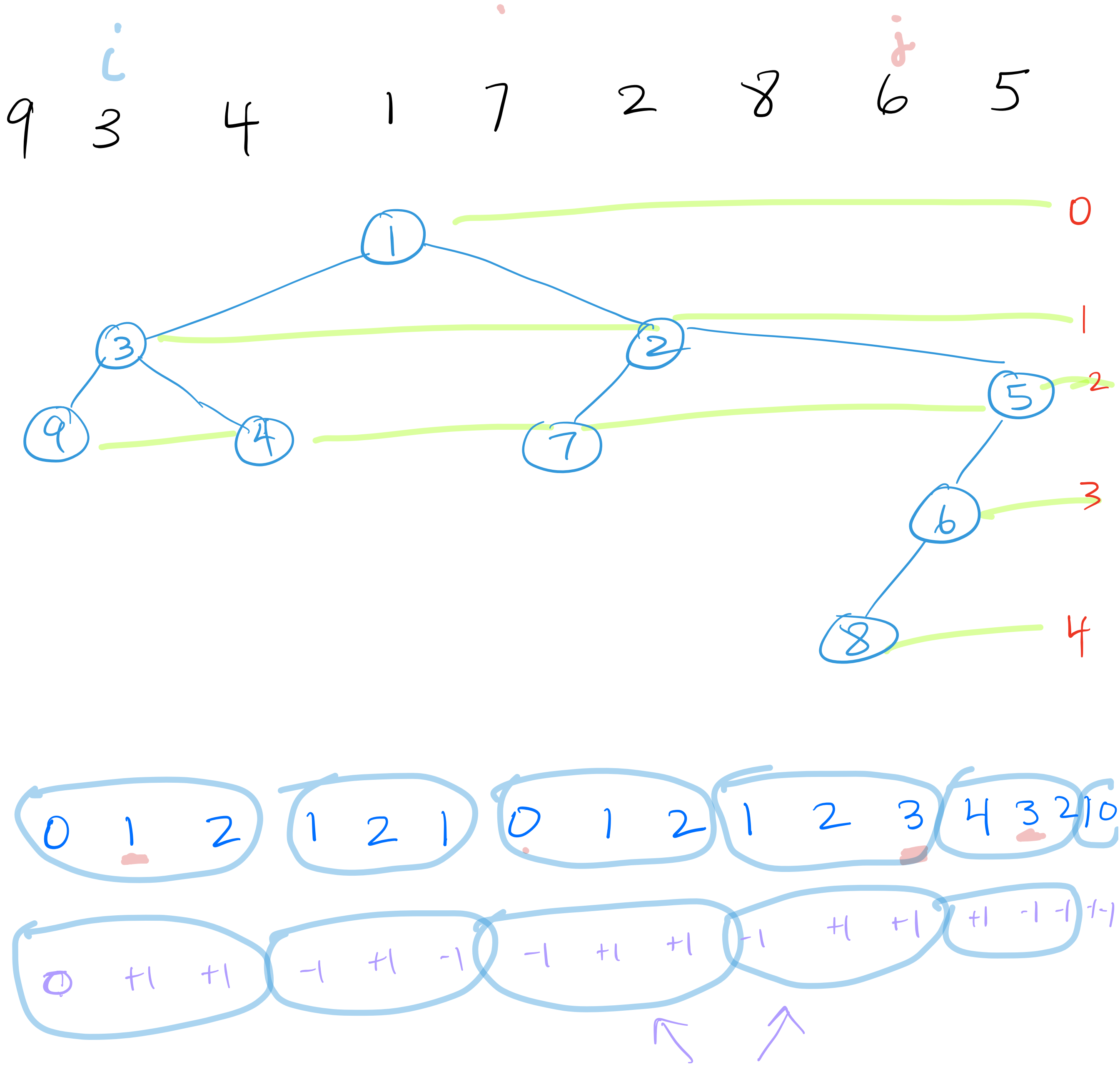
Sparse Table is $2\frac{n}{\lg n}$ *



$$\lg\left(\frac{2n}{\lg n}\right) = \lg n + \lg 2 - \lg\lg n \in \Theta(\lg n)$$

$\therefore n$

Sparse Table	n'	$n' = \frac{2n}{\lg n}$
precompute	$O(n' \lg n')$	For you to do
size	$O(n' \lg n')$	
query	$O(1)$	$O(1)$



Same category of block
– will yield same query answers (in terms of offsets)

- Construct a look-up table for each
  - i, j can take one of the $\frac{\lg n}{2}$ possible values

$\therefore \left(\frac{\lg^2 n}{4}\right)$ possible queries within a block-type

$\exists \ \frac{2n}{\lg n}$ blocks, but only $2^{\frac{\lg n}{2}}$ types of blocks.

$\therefore \sqrt{n}$ types of blocks $\times \ \frac{\lg^2 n}{4}$ is table size.

$\forall$ entry in the table is of size $\lg \lg n$ (why?)

$\Rightarrow$ size of table is $\sqrt{n} \lg^2 n \lg\lg n$.

ie is $o(n)$ bits. $\blacksquare$