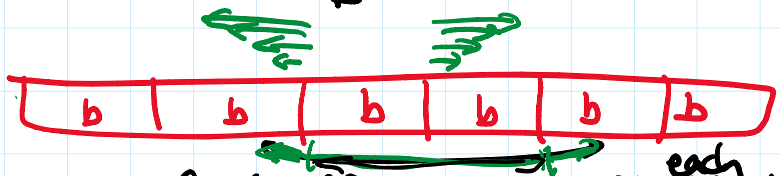
divide into $\frac{n}{b}$ blocks of size b



store prefix/suffix mins inside each block $\leftarrow O(n)$

use Method 6 inside each block

$\leftarrow O(\frac{n}{b} \cdot b \log b)$

recurse!

use Method 6 for the mins of all blocks

$\leftarrow O(\frac{n}{b} \log \frac{n}{b})$

query time $\boxed{O(1)}$

Space/prep time $O(n + \frac{n}{\cancel{b}} \cdot \cancel{b} \log b + \frac{n}{b} \log \frac{n}{b})$

set $b = \log n \Rightarrow \boxed{O(n \log\log n)}$

<u>Method 8</u>: recursion!

use recursion inside each block

rest same

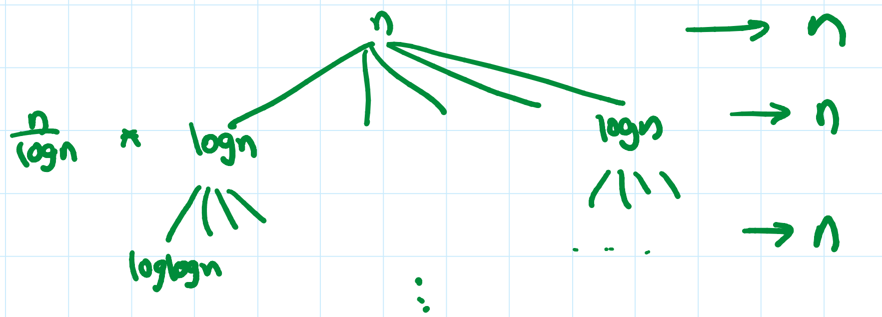
space/preproc time

$$S(n) = O(n) + O(\frac{n}{b} \log \frac{n}{b}) + \frac{n}{b} S(b)$$

Set $b = \log n$

$$S(n) = \frac{n}{\log n} S(\log n) + O(n)$$

$\rightarrow n$



Space/ preproc time $\boxed{O(n \log^* n)}$ iterated log

query time $\boxed{O(1)}$

Method 9

bootstrap again!

$$S(n) = O(n) + O\left(\frac{n}{b} \log^* \frac{n}{b}\right) + \frac{n}{b} S(b)$$

Set $b = \log^* n$

$\Rightarrow$ $S(n) = O(n) + \frac{n}{\log^* n} S(\log^* n)$

$\Rightarrow$ space/ preproc time $\boxed{O(n \log^{**} n)}$

query time $\boxed{O(1)}$

bootstrap ℓ times

space/prep $O(n \log^{\overbrace{** \cdots *}^{\ell}} n)$

query time $O(\ell)$

Def inverse Ackermann fn

Def inverse Ackermann fn

$\alpha(n)$ = smallest ℓ s.t. $\log^{\overbrace{**\cdots*}^{\ell}} n \le$ const

query time $\boxed{O(\alpha(n))}$

space/prep $\boxed{O(n)}$ (Yao '82)

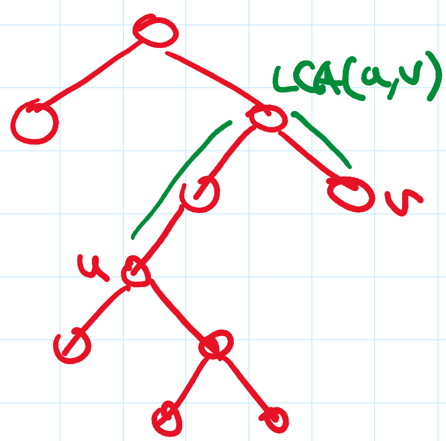
[optimal for semigroup op]

Related Problem (Lowest Common Ancestor (LCA))

Given binary tree T with n nodes,

build data structure to answer following query:

find nodes u, v, find LCA



Thm RMQ reduces to LCA.

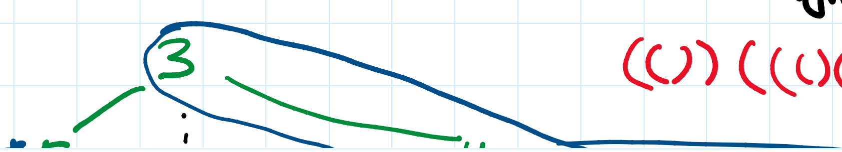
Pf: Given $a_1, \ldots, a_n$, define Cartesian tree:

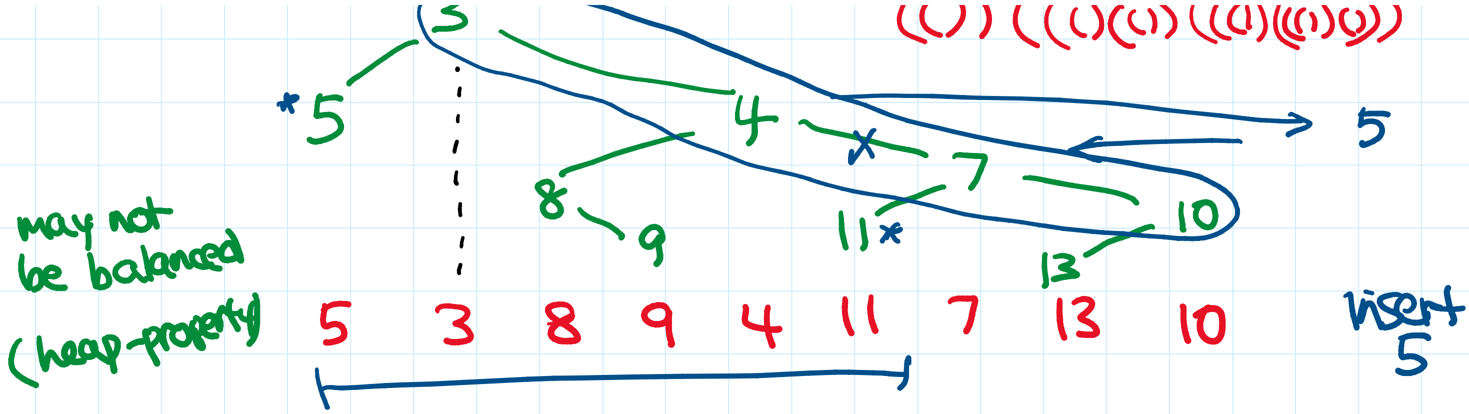
root is min

recurse on all elems to its left

" " right

(()) ((()(()) ((())((())()))

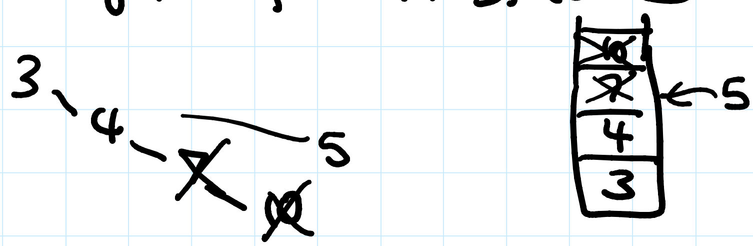




RMQ → LCA in Cartesian tree!

Cartesian tree can be built in $O(n)$ time:

e.g. insert elements from left to right
maintain rightmost path in stack S



to insert a_i:

while ($S.top() > a_i$) $S.pop()$

delete 1 edge, insert 2 edges

$S.push(a_i)$.

(amortized analysis!)

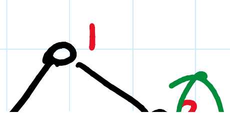
total time $O(n + \#\text{pops})$

$\leq O(n + \#\text{pushes})$

$= \boxed{O(n)}$. □

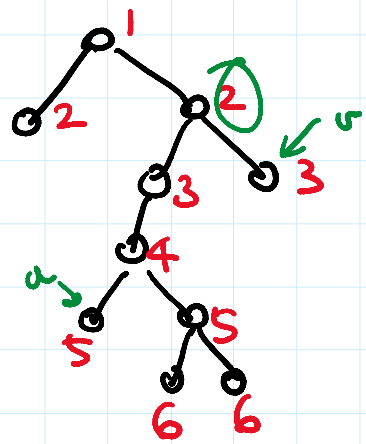
Thm LCA reduces to RMQ.

Pf:



Given tree

Pf:



Given tree

look depth values

take in-order traversal

2 1 [5 4 6 5 6 3 2 3]

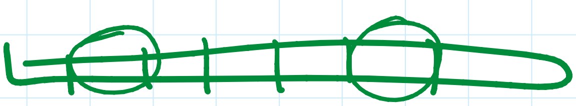
Method 10 (Final)

(Harel, Tarjan '84 / Schieber, Vishkin '88) Bender, Farach-Colton '00)

divide into $\frac{n}{b}$ blocks of size b as before

how to solve subproblems of size b directly?

precompute all answers for all inputs of size b.



\# inputs of size b

$= \#$ binary trees of size b

$\leq 2^{4b}$

can encode binary tree as sequence of $\leq 4b$ bits

Query by table lookup

query by table lookup

ans$[\underline{T}, i, j]$ table size $O(2^{4b} b^2)$

space/ preproc time

$$\le O\left(n + \frac{n}{b}\log\frac{n}{b}\right) + O(2^{4b} b^3) + O\left(\frac{n}{b}\cdot b\right)$$

query time $\boxed{O(1)}$

Set $b = \frac{\log n}{8}$

$\Downarrow$

$\boxed{O(n)}$ space/preproc time