August 23, 2023 12:55 AM

CS 598 TMC Advanced Data Structures (F'23)

courses.engr.illinois.edu/cs598tmc

Course Work:

4 HWs	45%
presentation	15%
project	40%

(may work in groups of ≤3)

Prerequisite: strong background in algorithms (CS374)

This is a <u>theory</u> course!

No textbook

Course Topics:

1. Basics (BST, heaps, union-find, ...)
2. Integers (hashing, vEB trees, fusion trees, ..)
3. Geometry (orthogonal range search, point location, ANN / LSH, ...)
4. Graphs (dynamic connectivity, distance oracles, ...)
5. Strings (suffix trees/arrays, ...)
6. Other models (succinct DS, external memory DS

5. Other models

(succinct DS,
external memory DS,
streaming / sketching, ...)

Problem (Range Min Queries (RMQ))

Given sequence of n numbers $a_1, \ldots, a_n$,
build a data structure to answer following query:

given i, j, find min of $a_i, \ldots, a_j$.

$$5,\ 3,\ \underbrace{8,\ 9,\ \boxed{4},\ 10}_{i \qquad\qquad j},\ 7$$

[static problem]

to bound: space, preprocessing time
query time

[other variants: dynamic (update time)
range median / mode,
2D, ...]

Method 1:

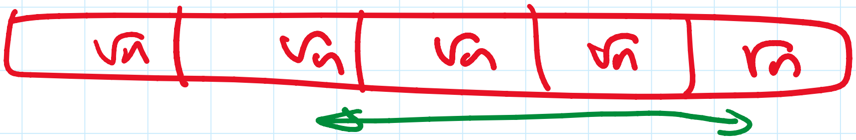
preproc time 0
space $\boxed{O(n)}$
query time $\boxed{O(n)}$

Method 2:

Precompute all answers in table
space $\boxed{O(n^2)}$ preproc time $\boxed{O(n^2)}$
query $\boxed{O(1)}$

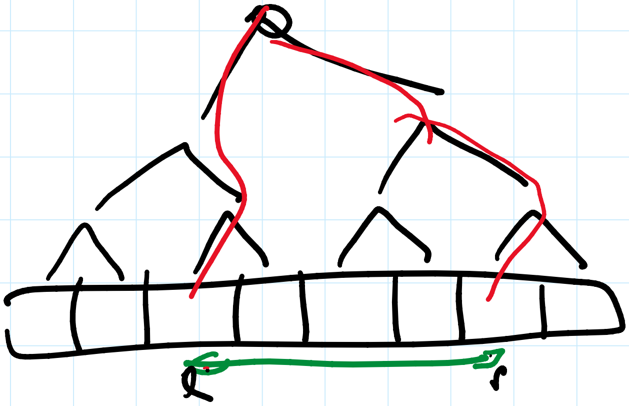
query $\boxed{O(1)}$

Method 3: divide into $\sqrt{n}$ blocks of size $\sqrt{n}$
precompute min of each block



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

Method 4 tree



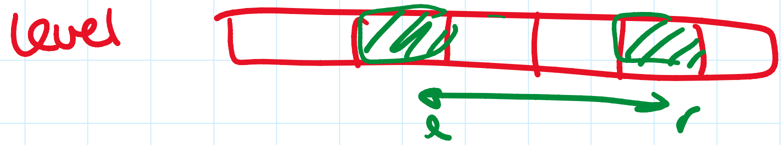
at each node, store min of subtree

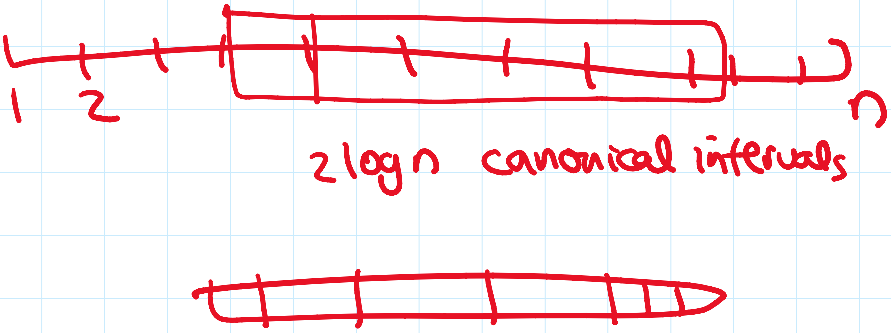
```
preproc($a_\ell, .., a_r$):
    1.  left.preproc($a_\ell, .., a_{\frac{\ell+r}{2}}$)
    2.  right.preproc($a_{\frac{\ell+r}{2}+1}, .., a_r$)
    3.  $m^* = \min(\text{left}.m^*, \text{right}.m^*)$
query(i,j):
    if  $[i,j] \cap [\ell,r] = \phi$   return $\infty$
    if  $[i,j] \supset [\ell,r]$   return $m^*$
    return min( left.query(i,j), right.query(i,j))
```

Space $\boxed{O(n)}$ preproc time $\boxed{O(n)}$

Space $\boxed{O(n)}$ preproc time $\boxed{O(n)}$

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

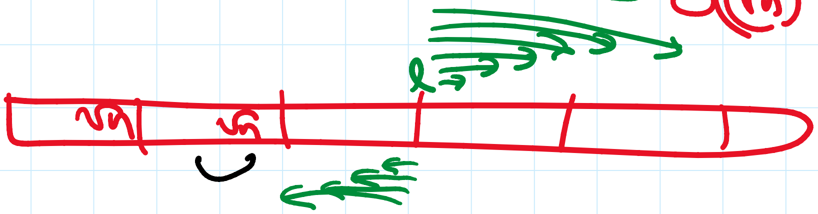




Method 5 divide into $\sqrt{n}$ blocks of size $\sqrt{n}$

precompute answers for (i,j) within each block

$O((\sqrt{n})^2 \cdot \sqrt{n}) = O(n^{3/2})$



for each block boundary l,

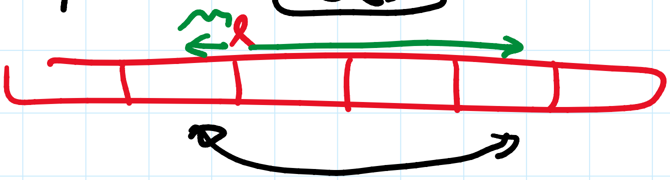
precompute ans for (l,j) $\forall j$

& for (i,l) $\forall i$

$O(n \cdot \sqrt{n}) = O(n^{3/2})$

space/ preproc time $\boxed{O(n^{3/2})}$

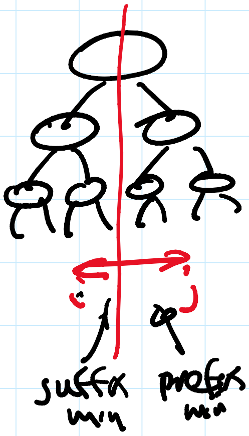
query time $\boxed{O(1)}$



Method 6: tree

at each node, store all prefix mins & suffix mins

```
preproc($a_l, .., a_r$):
    left.preproc($a_l, .., a_{\frac{l+r}{2}}$)
    right.preproc($a_{\frac{l+r}{2}+1}, .., a_r$)
    for j = l to r,   $m^+[j] = \min(a_l, .., a_j)$
    for i = l to r,   $m^-[j] = \min(a_i, .., a_r)$
```

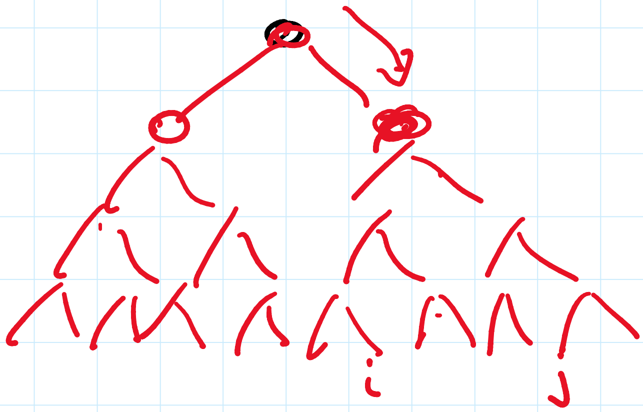


Space/preproc time $\boxed{O(n \log n)}$

$(S(n) = 2S(\frac{n}{2}) + O(n))$

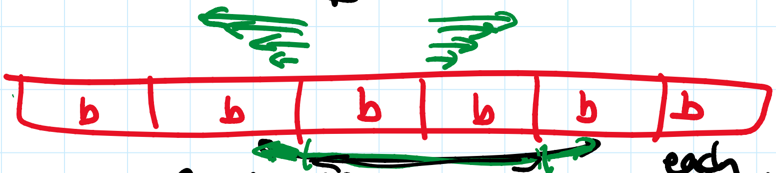
```
query(i, j):
    if   $j \le \frac{l+r}{2}$   return left.query(i,j)
    if   $i > \frac{l+r}{2}$   return right.query(i,j)
    return  $\min($left.$m^-[i]$, right.$m^+[j])$
```

query time $O(\cancel{\log n} + 1) = \boxed{O(1)}$



Method 7: bootstrap

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



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

use Method 6 inside each block

← $O(\frac{n}{b} \cdot b\log b)$

use Method 6 for the mins of all blocks

← $O(\frac{n}{b}\log\frac{n}{b})$

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