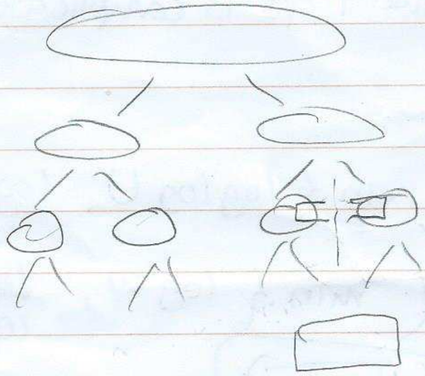
3-sided emptiness

use Cartesian tree

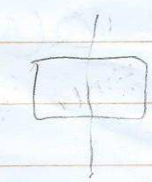
$\Rightarrow$ $\boxed{O(n)}$ space, $\boxed{O(\log\log U)}$ time

by 2 pred/succ searches
+ 1 LCA query

general 4-sided: use range tree



store 3-sided DS at each node



$\boxed{O(n \log n)}$ space

$\boxed{O(\log\log U)}$ time by 2 3-sided queries

(+ k for reporting)

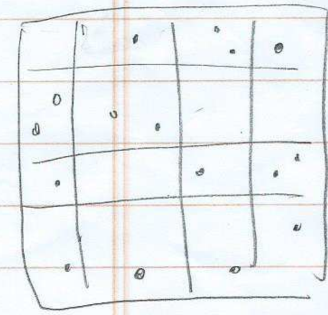
Advanced Method 2: Alstrup, Brodal, Rauhe '00

$O(n \log\log n)$ space, $O((\log\log U)^2)$ time

idea - $\sqrt{n}$-way recursion

form a grid with $\sqrt{\frac{n}{c}} \times \sqrt{\frac{n}{c}}$ cells

where each row/column has $\sqrt{cn}$ pts



recurse in each row & in each column

store 3-sided DS in each row & column ← space $O(n)$

store list L of all nonempty grid cells in 4-sided DS

$$\text{space } O(|L| \log |L|) = O\left(\frac{n}{C} \log \frac{n}{C}\right)$$
$$= O(n) \text{ by setting } C = \log n$$

Analysis of space:

let $s(n) =$ space per pt

$$s(n) \leq 2\, s(\sqrt{Cn}) + O(1)$$

pretend $C=1$

$$\Longrightarrow s(n) = O\left(2^{\log\log n}\right) = O(\log n) \text{ bad!}$$

idea - bit packing + "rank space reduction"

(replace coords by ranks, $U \mapsto n$)

$$s(n) \leq 2\, s(\sqrt{Cn}) + O\left(\cancel{\frac{\log U}{w}}\ \frac{\log n}{w}\right)$$

pretend $C=1$

$$\Longrightarrow s(n) = O\left(\frac{\log n}{w} + 2\frac{\log \sqrt{n}}{w} + 4\frac{\log n^{1/4}}{w} + \dots\right)$$
$$= O\left(\frac{\log n}{w} + \frac{\log n}{w} + \frac{\log n}{w} + \dots\right)$$
$$= O\left(\frac{\log n \log\log n}{w}\right) = O(\log\log n)$$

$(w \geq \log n)$

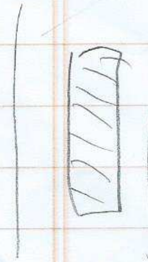
$$\Rightarrow S(n) = \boxed{O(n \log\log n)}$$

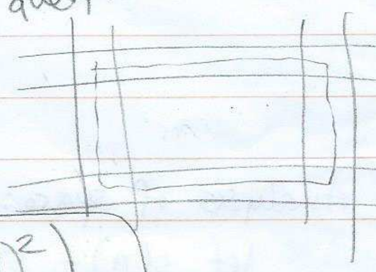
Pred search to locate column/row
+ rank space reduction

Querying (emptiness)

$$Q(n) \le \max \begin{cases} Q(\sqrt{Cn}) + O(\log\log U) \\ O(\log\log U) \end{cases}$$

4 3-sided queries
+ 1 4-sided query in L





$$\Rightarrow \quad Q(n) = \boxed{O((\log\log U)^2)}$$

(+ $k\log\log U$ for report)

<u>Advanced Method 3</u>: C.-Larsen-Patrascu '11

idea - go back to range tree!
- save space by bit packing + succinct DS

(Succinct rank DS)

<u>Fact 1</u> can store array A of n symbols in $[\sigma]$ using $O\left(\frac{n\log\sigma}{w}\right)$ words of space

s.t. can compute $\text{rank}_{A[i]}(i)$ = # times $A[i]$ occurs in $A[1..i]$ in $O(1)$ time

Pf sketch: divide into blocks of size σw

σw | σw | σw | ...

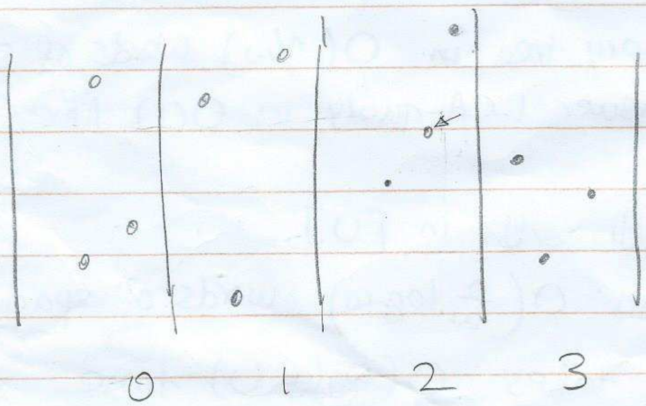
store σ counts per block

$$\Rightarrow \text{space } O\left(\frac{n}{\sigma w} \cdot \sigma\right) = O\left(\frac{n}{w}\right) \text{ words}$$

store answers within each block

$\Rightarrow$ space $O\left(\frac{n}{\sigma w} \cdot \sigma w \cdot \frac{\log(\sigma w)}{w}\right) = O\left(\frac{n \log(\sigma w)}{w}\right)$

can get rid of $\log w$ by more blocking ... $\square$



8th position

1 3 0 0 3 2 3 2 1 0 1 2

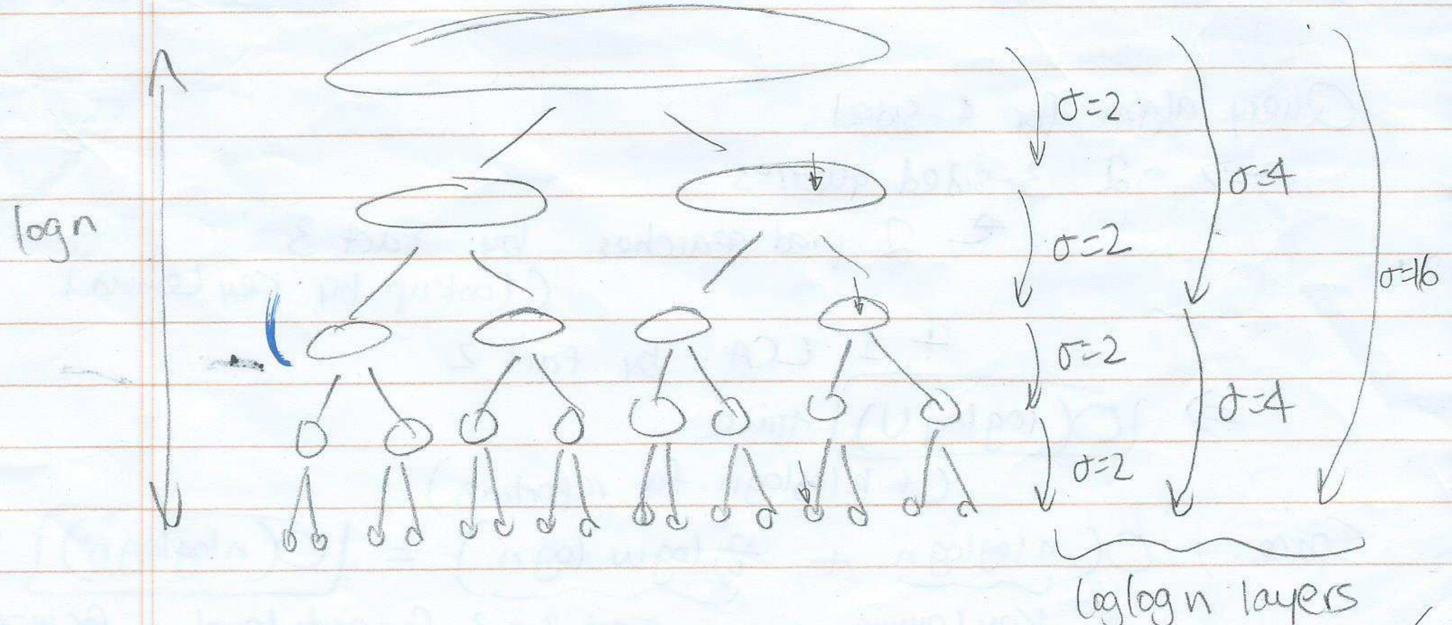
$\text{rank}_2(8) = 2$

8th lowest pt

= 2nd lowest pt in slab 2

<u>Key Lemma</u> can compress 2D range tree in $O(n \log\log n)$ space s.t. given any node v & index i can look up i^{th} element in v's y-sorted list in $O(\log\log n)$ time.

Pf: Use Fact 1 repeatedly



query: follow $O(\log\log n)$ "pointers" (by rank op)

space: $O\left(\frac{n\log 2}{w}\log n + \frac{n\log 4}{w}\frac{\log n}{2} + \frac{n\log 16}{w}\frac{\log n}{4} + \dots\right)$

$= O\left(\frac{n\log n}{w}\log\log n\right) \leq O(n\log\log n)$. □

(succinct LCA)

<u>Fact 2</u> can store any binary tree in $O(n/w)$ words of space s.t. can answer LCA query in $O(1)$ time

(succinct pred)

<u>Fact 3</u> given sorted list $y_1, \dots, y_n$ in $[U]$, can build DS in $O(\frac{n}{w}\log w)$ words of space

s.t. pred search takes $O(\log\log U)$ time $+ O(1)$ lookups of the y_i's.

Pf Sketch: select subset containing every other b elements

build vEB tree for subset $\leftarrow O(\frac{n}{b})$ space $\quad b = w/\log w$

for each block of b elements,

build compressed trie $\leftarrow O(\frac{n}{b}\cdot\frac{b\log w}{w})$ space

(recall fusion tree) □

Query alg'm for 4-sided:

$\Rightarrow$ 2 3-sided queries

$\Leftarrow$ 2 pred searches by Fact 3 (lookup by Key Lemma)

$+$ 1 LCA by Fact 2

$\Rightarrow \boxed{O(\log\log U)}$ time

($+ k\log\log n$ for reporting)

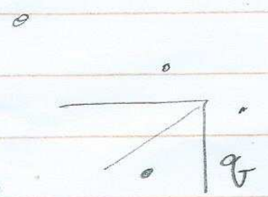
Space: $O(\underbrace{n\log\log n}_{\text{Key Lemma}} + \underbrace{\frac{n}{w}\log w\,\log n}_{\text{Fact 2 \& 3 for each level}}) = \boxed{O(n\log\log n)}$ for $w = \log n$

Rmk: tradeoffs

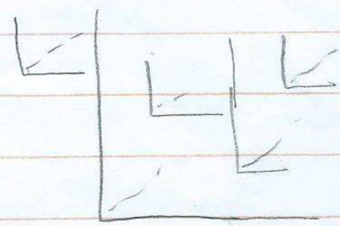
$O(n \log^{\varepsilon} n)$ space, $O(\log\log U + k)$ time

OR $O(n)$ space, $O(\log^{\varepsilon} U + k \log^{\varepsilon} n)$ time

Rmk: 3D dominance (3-sided) emptiness



$\Rightarrow$



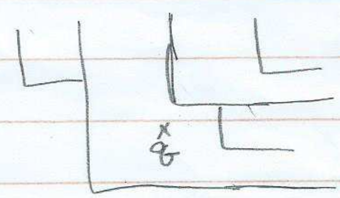
decide if q above staircase polyhedron

$\Downarrow$ project to xy

O(n) space

$O(\log\log U)$ time

(as we'll see...)



2D point location

3D general (6-sided):

$O(n \log^3 n)$ space, $O(\log\log U)$ time

improves to

$O(n \log^{1+\varepsilon} n)$ space, $O(\log\log U)$ time

(+ k for report)

[C.-Larsen-Patrascu '11]

[using recursive grids]

4D: use b-ary range tree

set $b = \log^{\varepsilon} n$

$O(n \log^{1+\varepsilon} n \cdot b \log n)$ space $\Rightarrow$ $O(n \log^{2+\varepsilon'} n)$ space

$O\left(\log\log U \cdot \frac{\log n}{\log b}\right)$ time $\Rightarrow$ $O(\log n)$ time

[Patrascu: lower bound $\Omega\left(\frac{\log n}{\log\log n}\right)$]