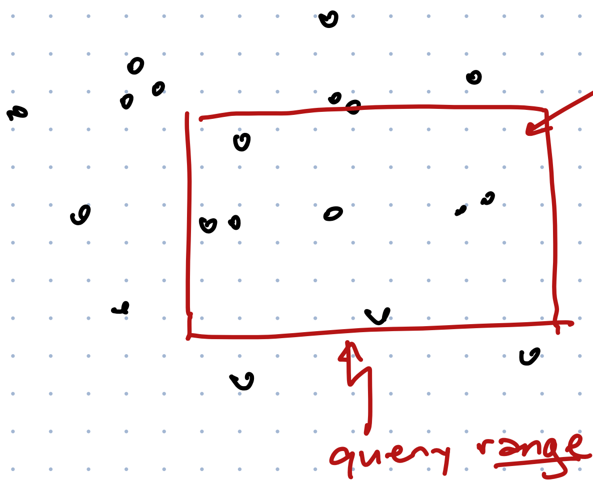
Geometric range searching

Orthogonal



- How many pts?
- ~~Total wt pts?~~
- Min. priority pt?
- List all pts.

$\boxed{\text{kd-tree}}$

$O(n)$ space ← good

$O(\sqrt{n})$ time ← okay

Range tree

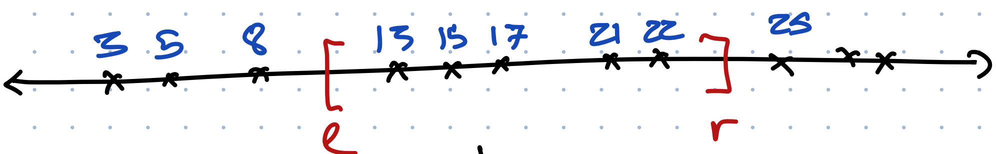
1D	$O(n)$ space	$O(\log n)$ time
2D	$O(n \log n)$ space	$O(\log^2 n)$ time $= O((\log n)^2)$
3D	$O(n \log^2 n)$ space	$O(\log^3 n)$ time
⋮		

1D range searching

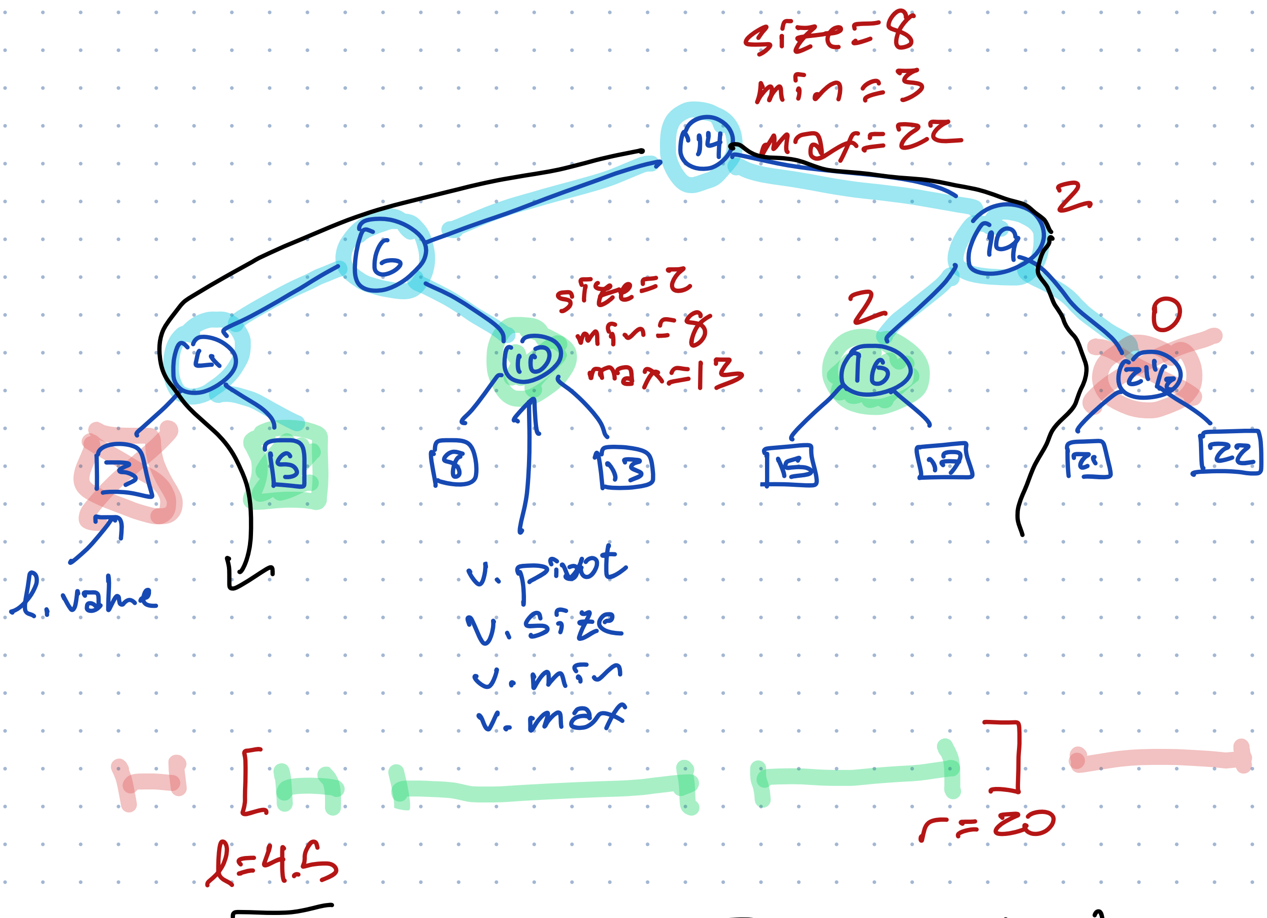
X = data = points in $\mathbb{R}$ = real numbers

q = query = range $[\ell, r]$ $\quad \ell, r \in \mathbb{R}$, $\ell < r$

answer: $\#(X \cap q)$



1D range tree = balanced ~~binary~~ search tree
with values in X at leaves
intermediate pivot values
at interior nodes



Tree subdivides any interval into O(logn) canonical ranges [v.min, v.max]

Query algorithm finds O(logn) nodes whose canonical ranges make up the query range, uses them to assemble answer in O(log n) time

Any decomposable function

$$F(A \cup B) = F(A) \diamond F(B)$$

2D range tree

P = points in $\mathbb{R}^2$ $\quad (p.x, p.y)$

query q = rectangle (ℓ, r, b, t)

$[\ell, r] \times [b, t]$

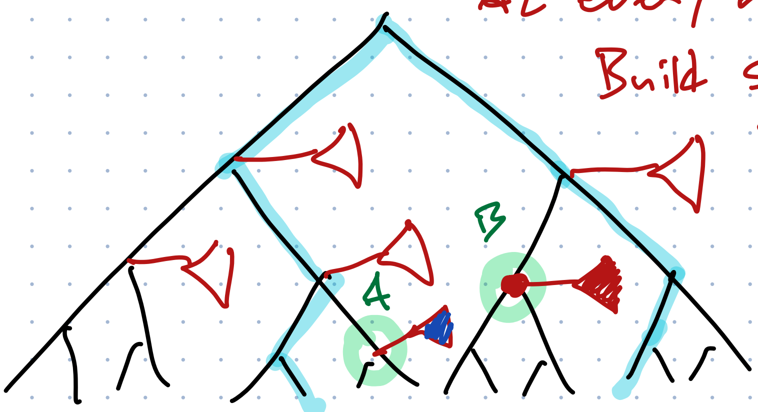
$= \{(x,y) \mid \ell \le x \le r \text{ and } b \le y \le t\}$

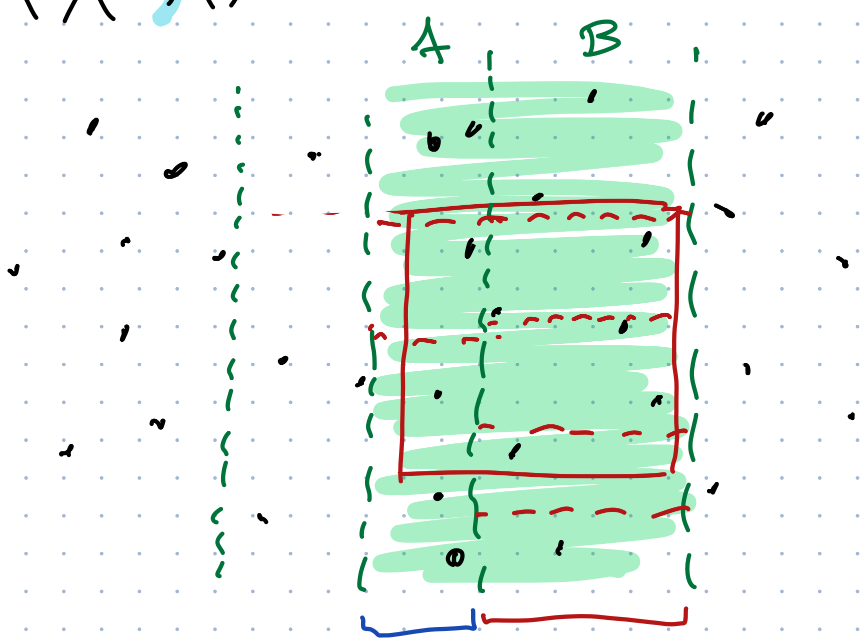
Primary structure = 1d range tree over x-coords of P

At every node v in primary tree

Build secondary structure

= 1d range tree over y-coords of P_v





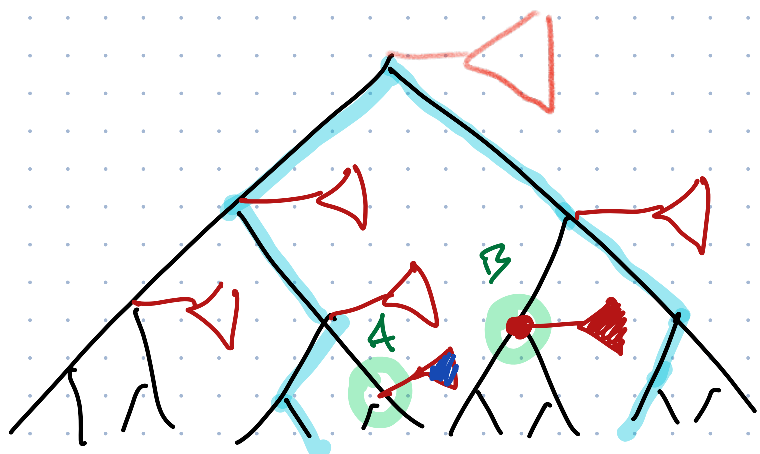
2d range query =

1d query over x-coords

for each canonical subset ← $O(\log n)$

1d query over the y-coords ← $O(\log n)$

$O(\log^2 n)$ time



(x,y)

Each point in P appears in P_v for all v along search path for x

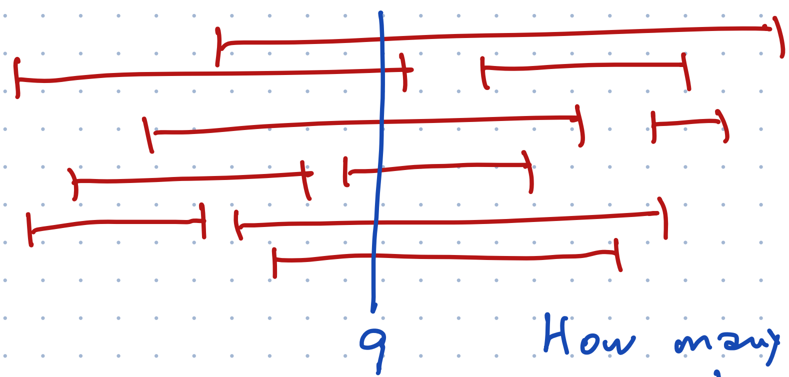
$\Rightarrow$ in $O(\log n)$ secondary structs

$\Rightarrow O(n \log n)$ space

Range-stabbing queries

Given S = set of intervals/segments [s.l, s.r]

Query q = point in $\mathbb{R}$



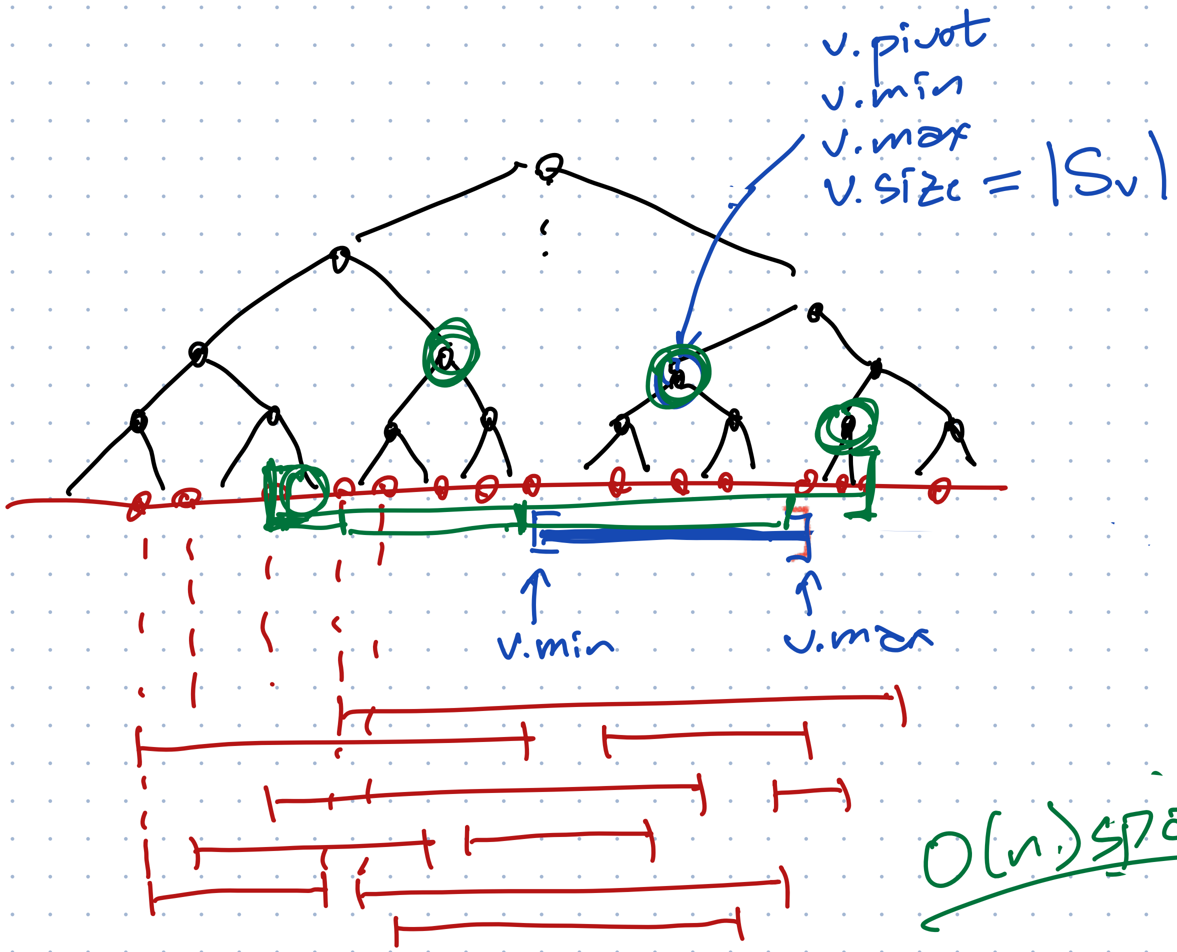
How many segments in S contain q?

Segment tree

Balanced BST over endpts of S

(interior nodes — endpts

leaves — gaps btwn endpts



O(n) space

Binary search tree splits any segment $s \in S$
into $O(\log n)$ canonical subsegments
just like range tree

S_v = subset of segments in S that include
v's canonical range in their decomposition

point-stabbing query (q) =
binary search for q
$\Rightarrow O(\log n)$ nodes
sum up v.size for those nodes v.
$O(\log n)$ time