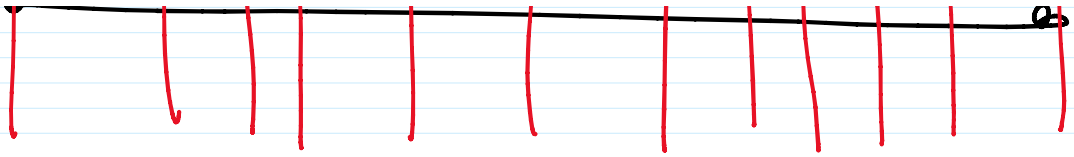
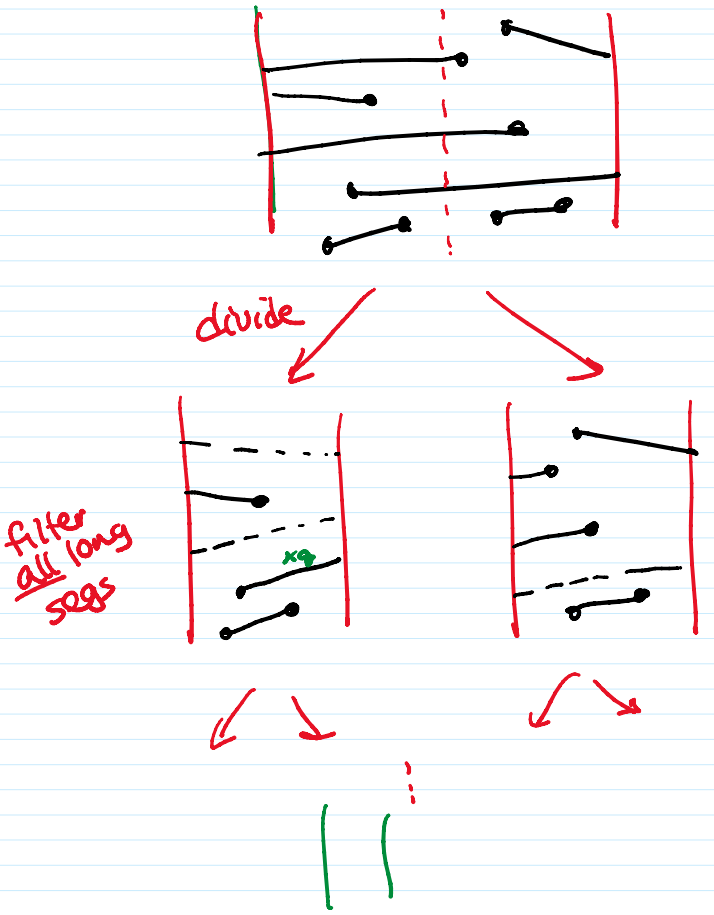


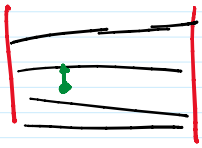
$Q(n) = O(\log n)$ (binary search in x then in y)

$S(n) = O(n^2)$ worst-case

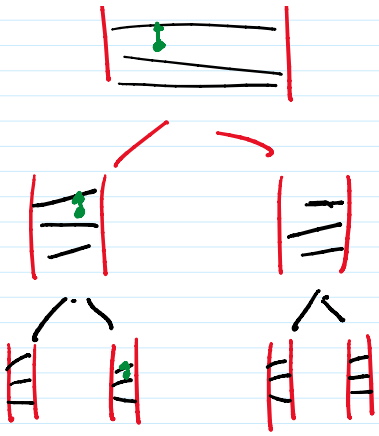
$P(n) = O(n^2 \log n)$

## Divide & Conquer Method ⇒ Segment Trees



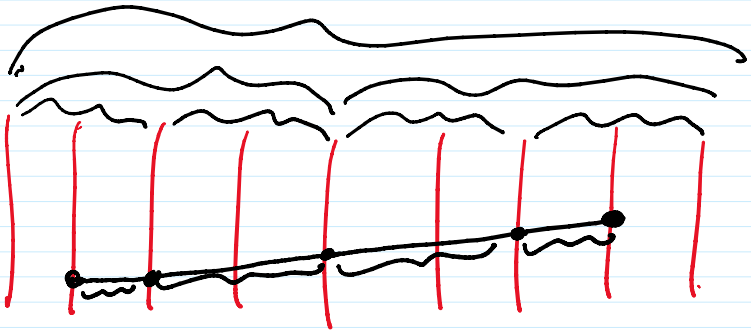



$P(n) = \boxed{O(n \log^2 n)}$



$S(n) = \boxed{O(n \log n)}$
because each seg is stored in $\leq 2 \log n$ nodes

$Q(n) = \boxed{O(\log^2 n)}$
because binary search in $O(\log n)$ nodes along a path



Refinement: (Edelsbrunner, Guibas, Stolfi '86)

$P(n) = O(n \log n)$, $S(n) = O(n)$,
$Q(n) = O(\log n)$

(optimal)

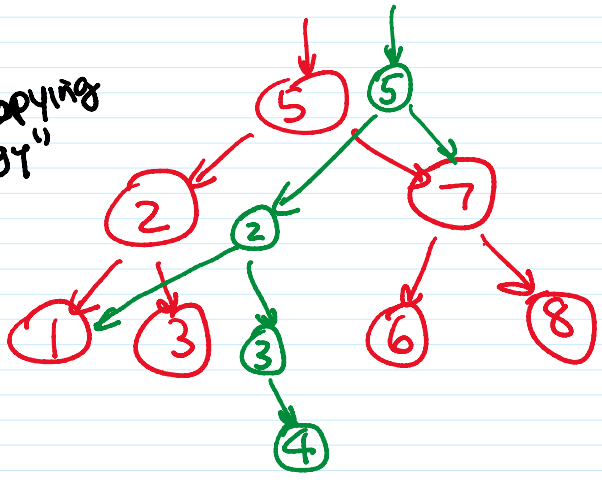
fractional cascading

---

## Sweep Method ⇒ Persistent Search Trees

remember "history" of the search tree T

e.g.
"path-copying strategy"




insert(4)

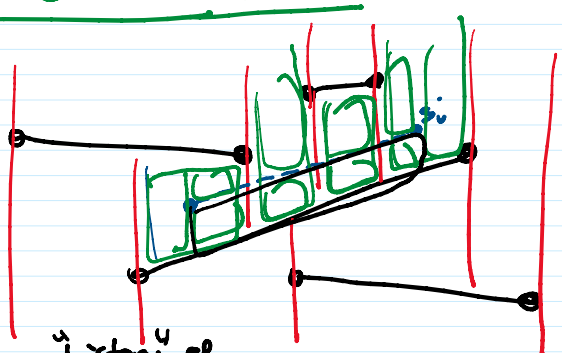
$P(n) = O(n \log n)$
$S(n) = O(n \log n)$ ⟷ $O(n)$
$Q(n) = O(\log n)$

optimal (smaller consts)

Refinement: (Sleator, Tarjan '86)

---

## Rand. Incremental Method

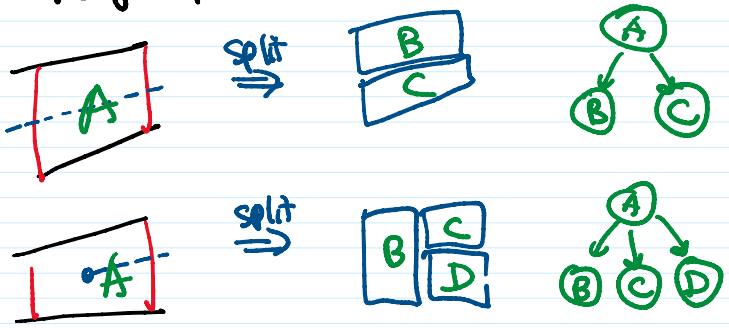


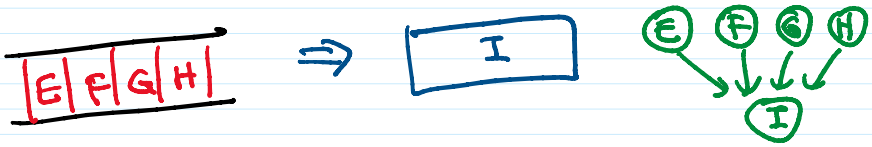

maintain a "history" of the vertical/"trapezoidal decomposition" as we insert segs in rand order


preproc(S):

0. let $s_1, .., s_n$ be a rand order of S
1. for $i = 1, .., n$ do {
2. locate trapezoid containing left endpt of $s_i$
3. generate all trapezoids cut by $s_i$ by "walking"
4. Split these trapezoids & link old to new ones
5. merge trapezoids w. common sides " " " "

}









query algm - follow links from top down
in history dag
(directed acyclic graph)