CS 473 Algorithms: Lecture 2 (2022-01-20)

logistics:
- signup
  - piazza
  - gradescope
- pset 0 on Friday 5pm

last lecture:
- introduction
- divide and conquer
  - integer multiplication
    - Karatsuba's algo

today: divide and conquer

Q: are we socially distanced?

def: given points P $p_1 = (x_1, y_1), \dots, p_n = (x_n, y_n) \in \mathbb{Z}^2$

the closest pair problem is find

$$\arg\min_{i \neq j} \text{dist}(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

(assuming x_i, y_j distinct)

Q: cost of manipulating integers?

A: programming languages have $a \leftarrow b \cdot c$ as primitive

A: multiplication of n-bit integers done in $O(n^{\log_2 3})$ time

Q: what is a reasonable model of cost of manipulating integers?

A: n-bit arithmetic ops are unit cost

fact: exist algorithms "efficient" in unit cost model, not known to be realistically implementable

$\Rightarrow$ not realistic

A: n-bit arithmetic takes $n^{O(1)}$ time

$\Rightarrow$ tedious

A: $O(\lg n)$-bit arithmetic operation takes unit cost

convention: on problems on set of n-integers, integers are $O(\lg n)$ bits
also can use $O(\lg n)$-bit arithmetic at unit cost
the input size is n

eg: closest pair, eg/non-eg: n-bit multiplication

Prop closest pair solvable in $O(n^2)$ steps

pf: algo: output $\arg\min_{i \neq j} \text{distance}(p_i, p_j)^2$

$$= (x_i - x_j)^2 + (y_i - y_j)^2$$

correctness: $\sqrt{\cdot}$ is monotonic

complexity: $\binom{n}{2} = \Theta(n^2) = O(n^2)$; $O(1)$

∎

Prop: one-dimensional closest pair in $O(n \lg n)$

pf: algo: (1) sort $x_1, \ldots, x_n$ into $\hat{x}_1 \le \cdots \le \hat{x}_n$ $\quad O(n \lg n)$

(2) output: $\min_i \text{dist}(x_i, x_{i+1})$

$$= \sqrt{(x_i - x_{i+1})^2}$$
$$= |x_i - x_{i+1}|$$
$$= x_{i+1} - x_i$$

$O(n)$ $\quad O(1)$

correctness: clear

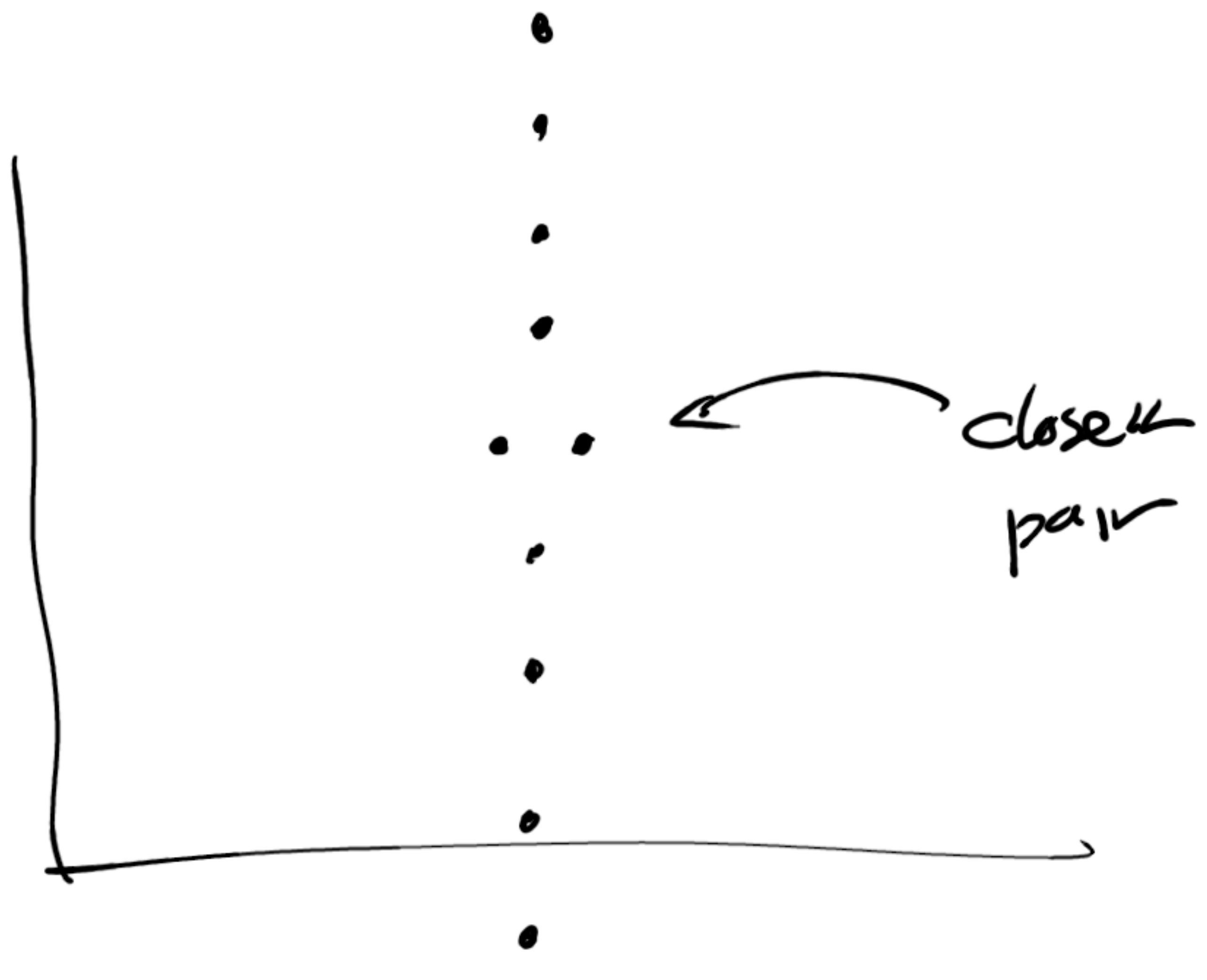
complexity:

$$O(n \lg n) + O(n) = O(n \lg n)$$

∎

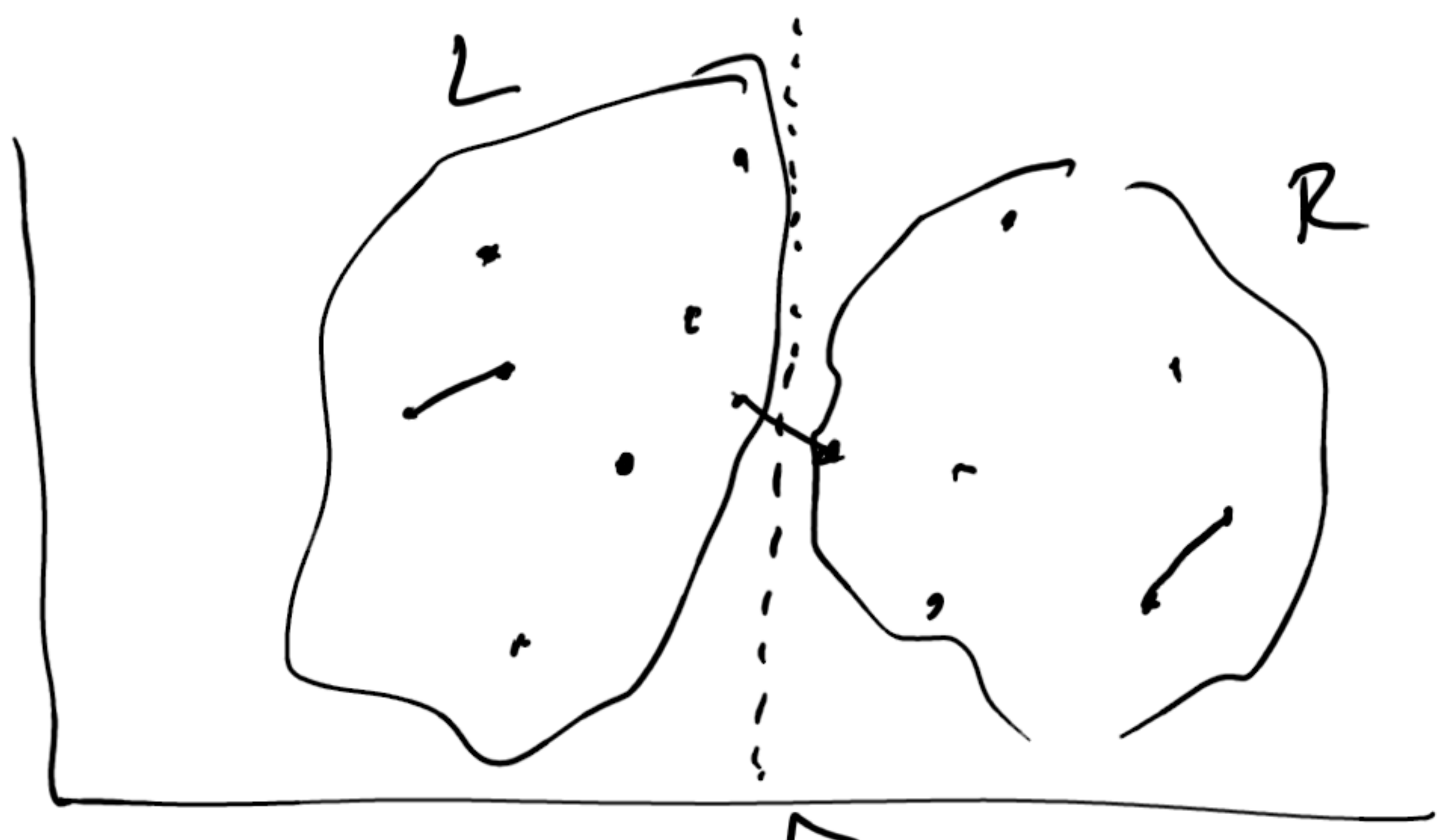
Thm: two-dimensional closest pair in $O(n \lg n)$

idea: sort $p_1, \ldots, p_n$ by x-coord



Thm

idea: divide and conquer



def: $A, B \subseteq P$

$$\text{dist}(A,B) = \min_{\substack{a \neq b \\ a \in A,\ b \in B}} \text{dist}(a,b)$$

$\Rightarrow$ $\text{dist}(P,P)$ is dist closest pair

def: $L, R \subseteq P$ by $L = \{p_i : x_i \leq \hat{x}_{\lfloor n/2 \rfloor}\}$ ← median x-coord

$R = P \setminus L$

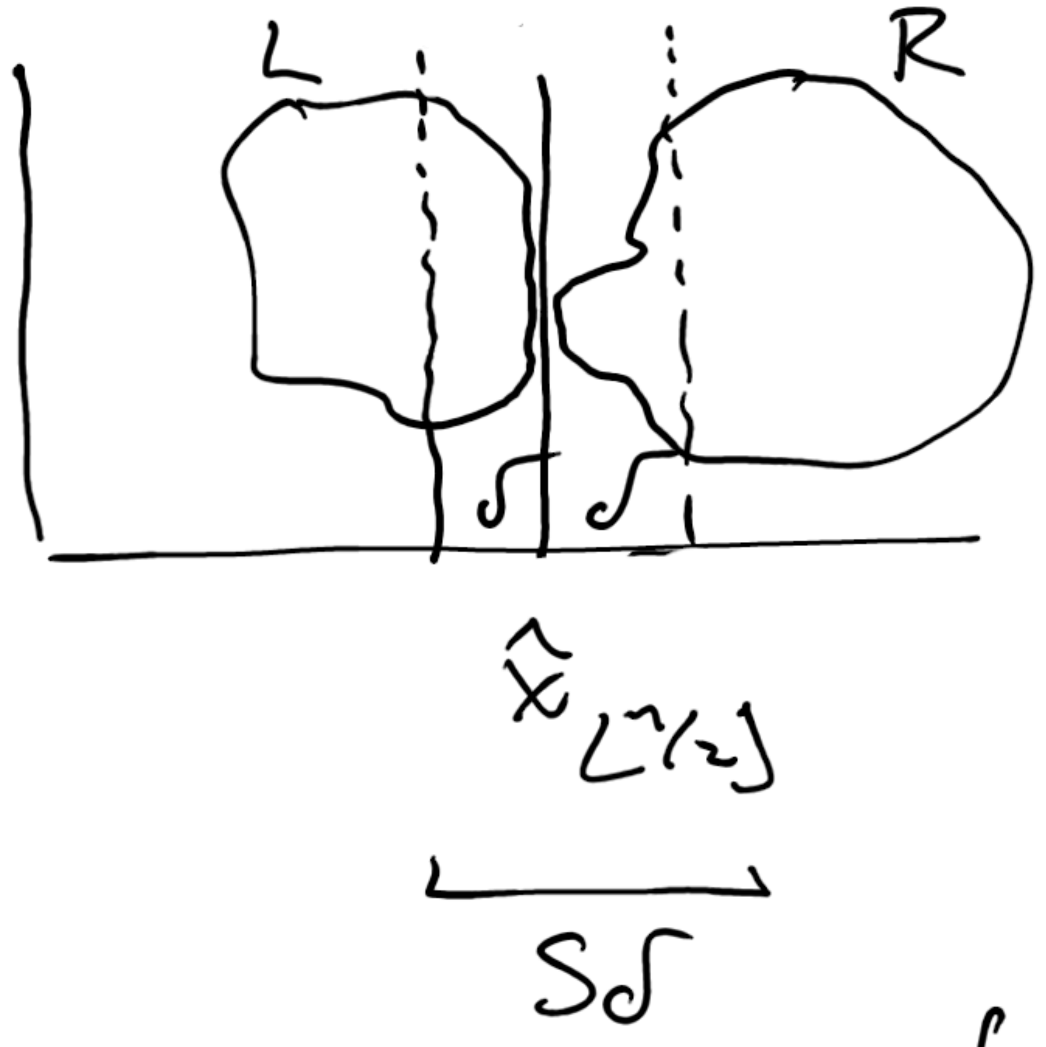
lem: $|L| = \lfloor n/2 \rfloor$, $|R| = \lceil n/2 \rceil$

pf: all x-coord distinct ← assumption

lem: $\text{dist}(P,P) = \min \begin{cases} \text{dist}(L,L) \\ \text{dist}(R,R) \\ \text{dist}(L,R) \end{cases}$

Q: is computing ↗ any easier?

def: $S_\delta = \{p_i : \hat{x}_{\lfloor n/2 \rfloor} - \delta \leq x_i \leq \hat{x}_{\lfloor n/2 \rfloor} + \delta\}$

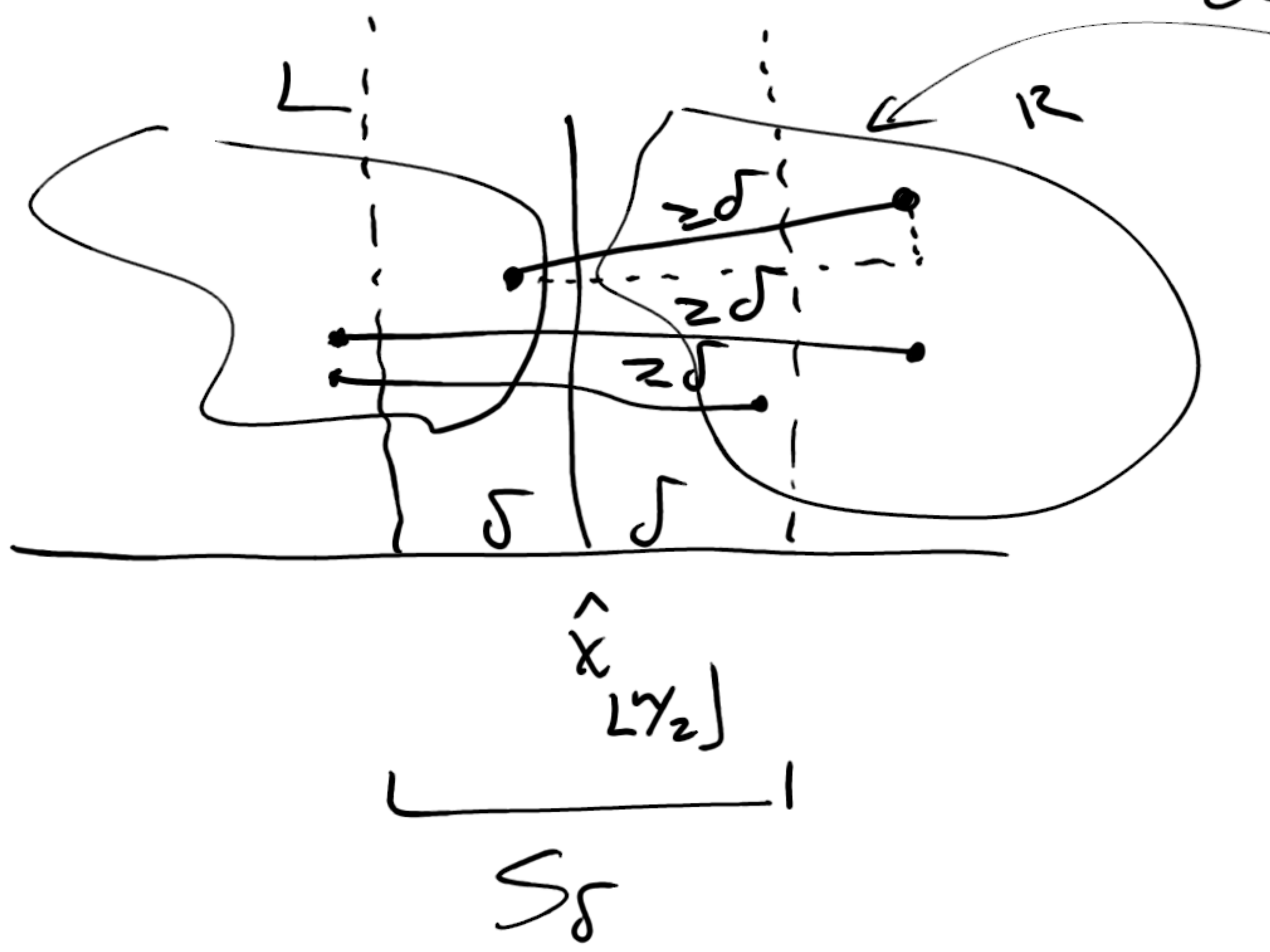


is δ-margin median strip of P

lem: take $\delta = \min \begin{cases} dist(L,L) \\ dist(R,R) \end{cases}$

then $dist(P,P) = \min \begin{cases} \delta \\ dist(L \cap S_\delta, R \cap S_\delta) \end{cases}$

Pf:



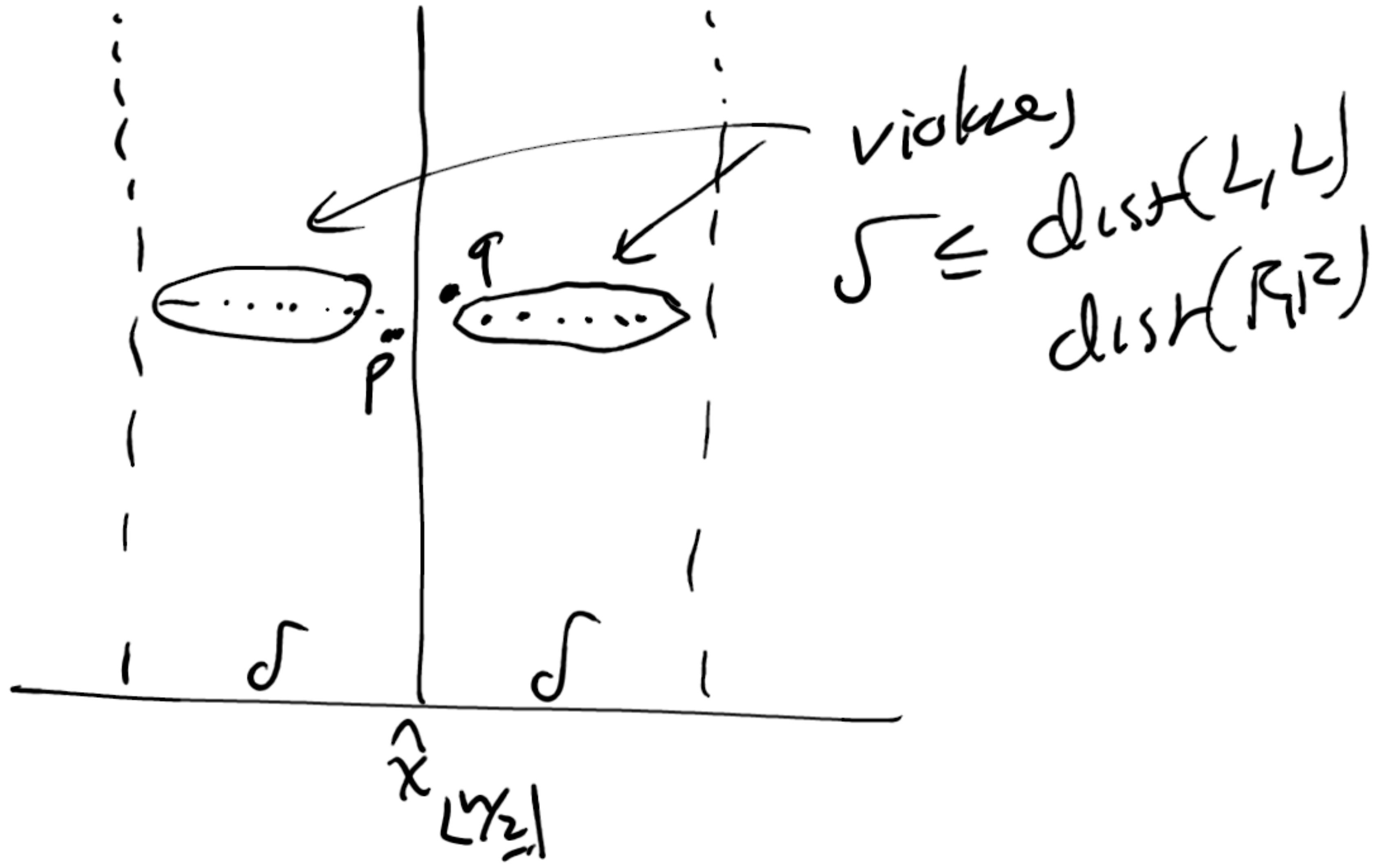
$dist(L, R \setminus S_\delta)$

$dist(L \setminus S_\delta, R)$

$\square$

Q: is computing $dist(L \cap S_\delta, R \cap S_\delta)$ easier?

eg: sort by y-coord



idea: use

$\delta \le dist(L,L)$
$dist(R,R)$ in

computing $dist(L \cap S_\delta, R \cap S_\delta)$

prop.: $\delta = \min \begin{cases} dist(L, L) \\ dist(R, R) \end{cases}$

$S_\delta = (\tilde{p}_i)_i$, $\tilde{p}_i$ sorted by y-coord

if $\tilde{p}_i \in L \cap S_\delta$, $\tilde{p}_j \in R \cap S_\delta$

w/ $dist(\tilde{p}_i, \tilde{p}_j) \leq \delta$

$\Rightarrow |i-j| \leq 8$

thm.: two-dimensional closest pair in $O(n(\log n)^2)$

pf.: algo: (1) if $|P| \leq 3$, solve directly — $O(1)$

(2) sort P by x-coord — $O(n \lg n)$

(3) partition P by $\hat{x}_{\lceil n/2 \rceil}$ into $L \sqcup R$ — $O(n)$

(4) compute $dist(L, L)$ recursively — $T(n/2)$

$dist(R, R)$ recursively — $T(n/2)$

(5) δ min — $O(1)$

(6) compute S_δ — $O(n)$

(7) sort S_δ by y-coord — $O(n \lg n)$

(8) compute $\min_{\substack{\tilde{p}_i \in L \cap S_\delta \\ \tilde{p}_j \in R \cap S_\delta \\ |i-j| \leq 8}} dist(\tilde{p}_i, \tilde{p}_j)$ — $O(8 \cdot n)$ = $O(n)$

(9) output $\min \begin{cases} \delta \\ \text{(result of 8)} \end{cases}$ — $O(1)$

correct.: "clear"

complexity: $T(n)$ = max # steps on n points

$$\leq 2 \cdot T(n/2) + \underbrace{O(n \lg n) + O(n)}_{O(n \lg n)}$$

$$\leq O(n (\log n)^2)$$

∎

prop: $\delta = \min \begin{cases} dist(L, L) \\ dist(R, R) \end{cases}$

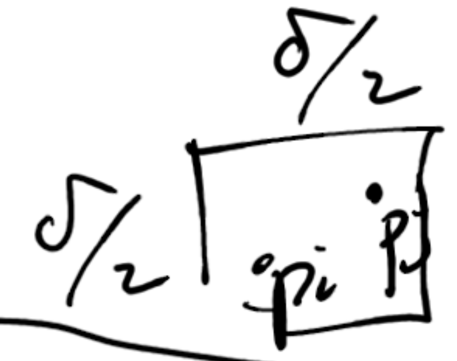
$S_\delta = (\tilde{p}_i)_i$ sorted by y-coord

if $\tilde{p}_i \in L \cap S_\delta$, $\tilde{p}_j \in R \cap S_\delta$

w/ $dist(\tilde{p}_i, \tilde{p}_j) \le \delta$

$\Rightarrow |i - j| \le 8$

pf: **Clm:** any $\delta/2 \times \delta/2$ box contains ≤ 1 point from L (or R)

pf: 

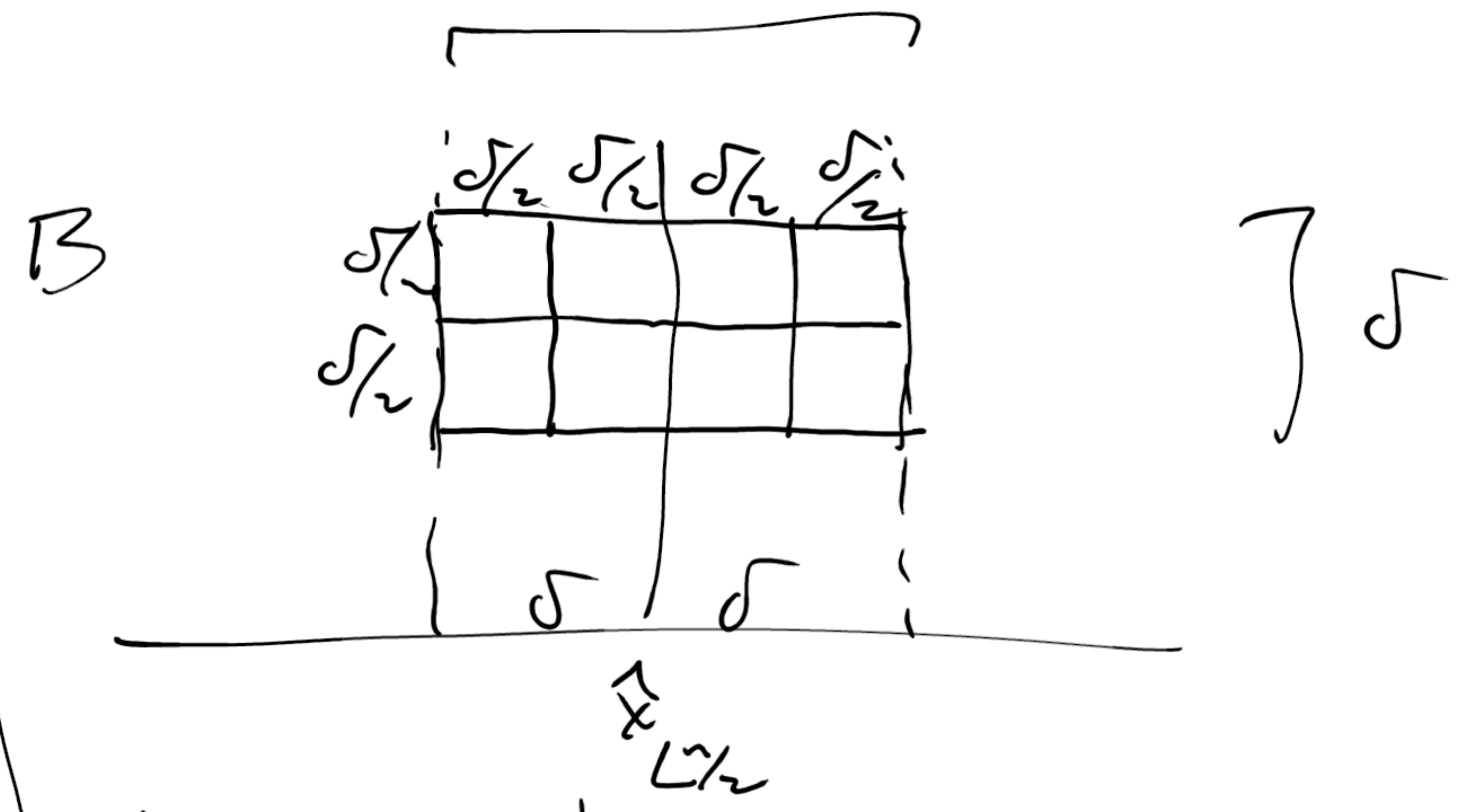
$dist(p, q) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ where $|\cdot| \le \delta/2$ and $|\cdot| \le \delta/2$

$\le \sqrt{\delta^2/4 + \delta^2/4} = \delta/\sqrt{2} < \delta$

$\le dist(L, L), dist(R, R)$

Clm: any $2\delta \times \delta$ box centered around median $\hat{x}_{\lfloor n/2 \rfloor}$ contains ≤ 8 points from S_δ

pf:



$|S_\delta \cap B| = |S_\delta \cap B \cap L| + |S_\delta \cap B \cap R|$ (L, R disjoint)

$|S_\delta \cap B \cap L| \le 4$, $|S_\delta \cap B \cap R| \le 4$

≤ 8

pf of prop:

$\tilde{p}_i = (\tilde{x}_i, \tilde{y}_i) \in L \cap S_\delta$

$\tilde{p}_j = (\tilde{x}_j, \tilde{y}_j) \in R \cap S_\delta$

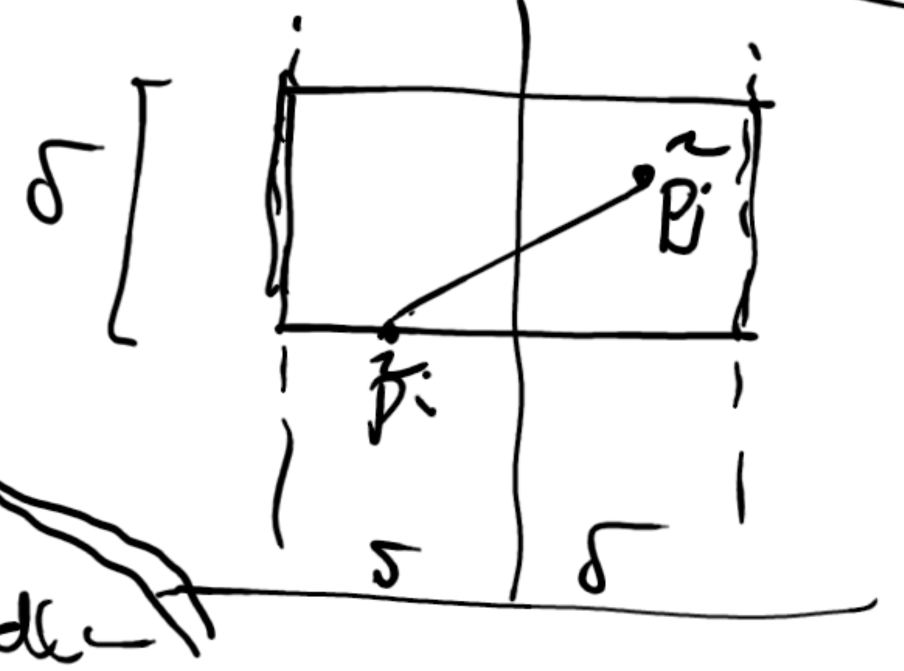
w/ $\tilde{y}_i \le \tilde{y}_j$

$dist(\tilde{p}_i, \tilde{p}_j) \le \delta$

$\Rightarrow \tilde{p}_j \in B$

$|B \cap S_\delta| \le 8$

$\Rightarrow |i - j| \le 8$ in y-coord sorted order



$y = \tilde{y}_i + \delta$

$y = \tilde{y}_i$

∎

thm: twodimension closest pair in $O(n \lg n)$

sketch: idea: sorting on (every) recursive call is wasteful

instead sort P according to x-coord (once at beginning)

y-coord (once at beginning)

modify initial sorts to set L, R according to x-coord, y-coord

correctness: same

complexity:

$$T(n) \le O(n \lg n) + R(n) \le O(n \lg n) + O(n \lg n) = O(n \lg n)$$

$$R(n) \le 2R(n/2) + O(n)$$

$$\le O(n \lg n)$$

∎

rmk: twodimensional closest pair in $O(n)$ time, using randomization

today: divide and conquer: twodimensional closest pair

next lecture: dynamic programming

logistics:
- signup
- pizza
- guidesign
- posted on FSpn