Closet Pair of Points: Divide and Conquer

Monday, 21 August 2023 10:43 AM

I/P: Array $S = ((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n))$ of n points in $\mathbb{R}^2$.

$p_i = (x_i, y_i)$

O/P: Pts. p_i, p_j with minimum L_2 distance, i.e., pair of pts. that minimize $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$

Easy algo: for each pair of points i, j, find distance

$d(i,j) = \sqrt{\ldots}$,

o/p pts. w/ minimum distance

Time taken: $O(n^2)$

Will show how to do this in $O(n \log n)$ time, using

Divide & Conquer:
1. Divide problem into two roughly equal subproblems,
2. Solve each subproblem independently (using recursion)
3. Combine solutions to both subproblems, to solve larger problem.

Assume: no 2 pts. have same x-coordinate or same y-coordinate.

Step 1: Sort S by x-coord & y-coord separately

$S^x \leftarrow S$ sorted by x-coord

$S^y \leftarrow S$ sorted by y-coord

Now divide:

$\bar{x} \leftarrow \lfloor \frac{n}{2} \rfloor^{th}$ highest x-coord

$S^L \leftarrow \{p_i : x_i \le \bar{x}\}$

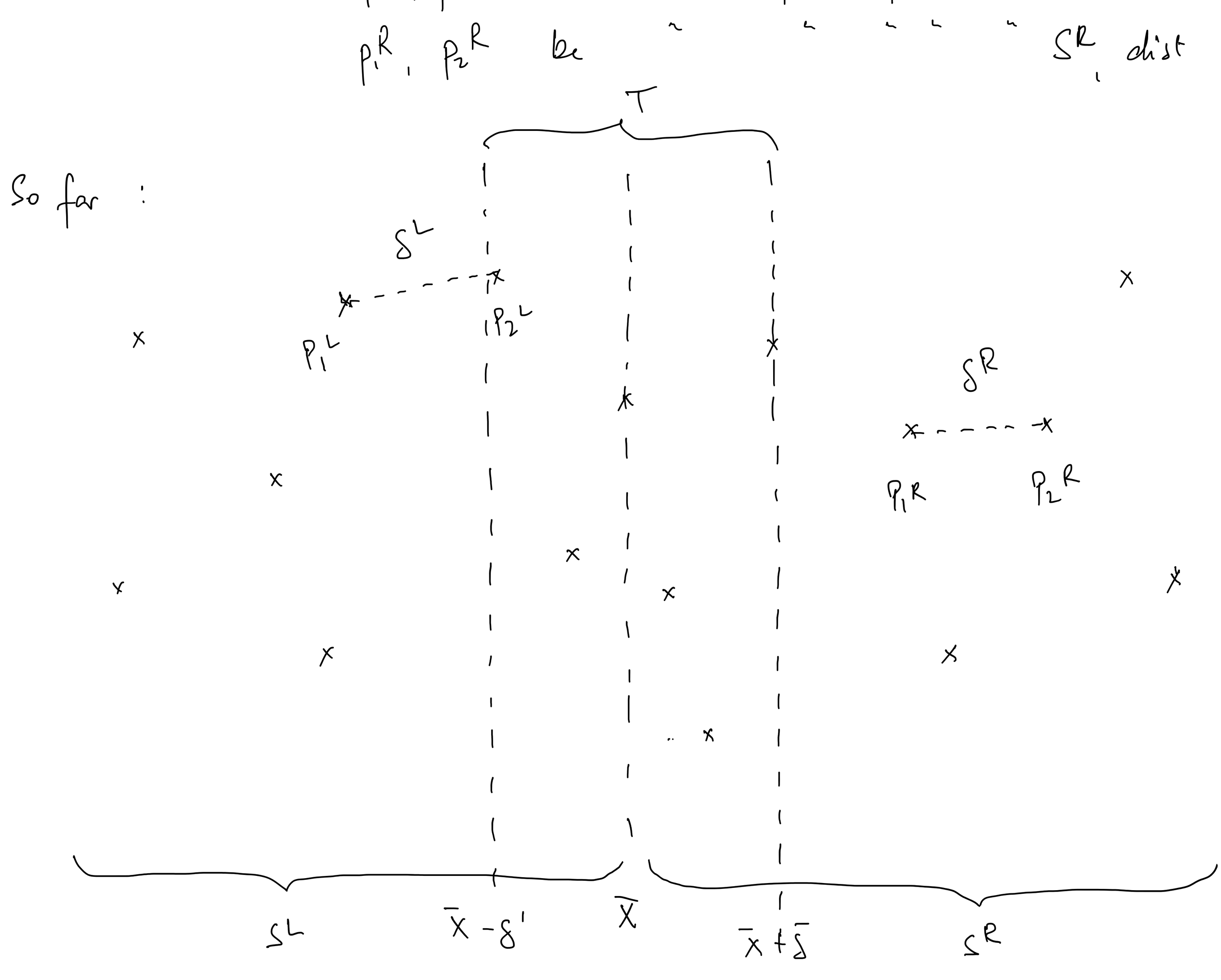
$S^R \leftarrow \{p_i : x_i > \bar{x}\}$

Step 2: Conquer: Solve S^L, S^R recursively

Let p_1^L, p_2^L be closest pair of pts. in S^L, dist δ^L

p_1^R, p_2^R be " " " " S^R, dist δ^R

So far:



Now we need to combine these two solutions.

Let $\delta_{min} \leftarrow \min\{\delta^L, \delta^R\}$

Q. Is there a pair of points p_i, p_j s.t. $d(p_i, p_j) < \delta_{min}$?

If yes, then:

(i) one must be in S^L, the other in S^R

(ii) x-coords must be within δ of each other (& hence $\max\{|x_i - \bar{x}|, |x_j - \bar{x}|\} < \delta_{min}$)

(iii) y-coords must be within δ of each other ($|y_i - y_j| < \delta_{min}$)

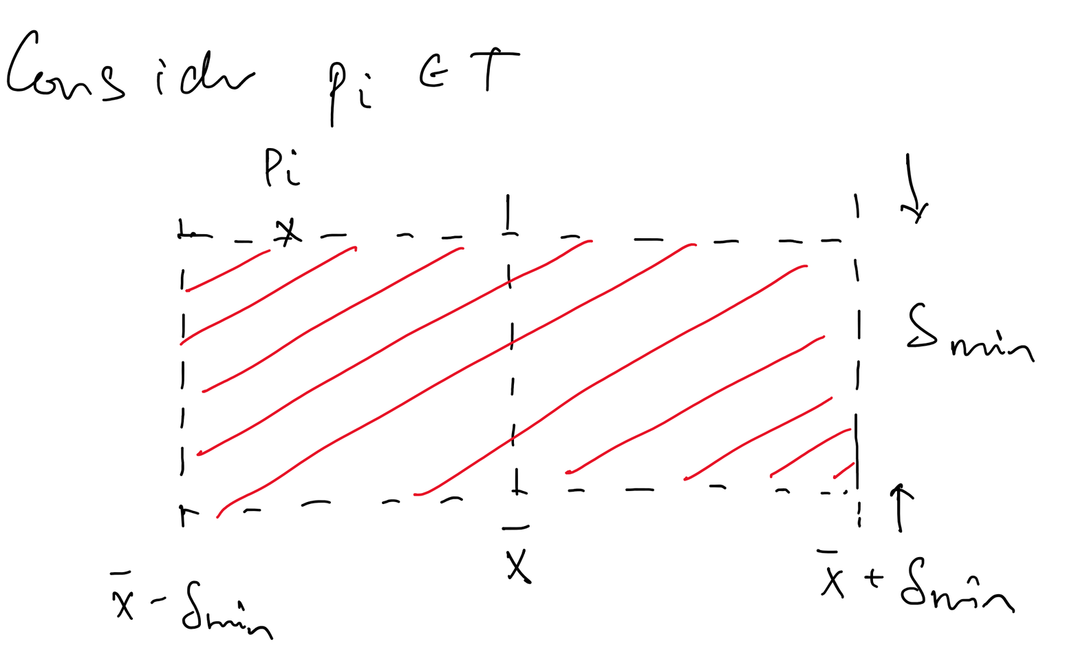
Step 3: Combine: Let $T = \{p_i : |x_i - \bar{x}| \le \delta_{min}\}$

If $\exists\, p_i, p_j$ closer than δ_{min}, BOTH must be in T (why?)

We will consider pts in T in order of decr. y-coord.

i.e., $\forall p_i \in T$, we try to find pts. closer than δ_{min} in T, below p_i.

Consider $p_i \in T$



To find if $\exists p_j$ closer than δ_{min} to p_i, only need to consider points within this box

Q. How many pts. can there be in this box?

A. At most 6 (5 excluding p_i)

So for each $p_i \in T$,
- in order of decreasing y-coordinates,
- look at next 5 pts. in T
- if $\exists p_j$ s.t. $d(p_i, p_j) < \delta_{min}$, store p_i, p_j as current closest pair of pts.

Algo: Find Closest (S)

I/P: Array S of n pts $((x_1, y_1), \ldots, (x_n, y_n))$, $p_i = (x_i, y_i)$

O/P: Closest pair of pts p_i, p_j, dist δ

If $(|S| \le 10)$ use brute force to find closest pair of points

$S^x \leftarrow S$ sorted by x-coord
$S^y \leftarrow S$ sorted by y-coord } $O(n \log n)$

$\bar{x} \leftarrow \lfloor \frac{n}{2} \rfloor^{th}$ highest x-coord

$S^L \leftarrow \{p_i : x_i \le \bar{x}\}$, $S^R \leftarrow \{p_j : x_i > \bar{x}\}$ → $O(n)$

$(p_1^L, p_2^L, \delta^L) \leftarrow$ Find Closest (S^L) → $T(n/2)$

$(p_1^R, p_2^R, \delta^R) \leftarrow$ Find Closest (S^R) → $T(n/2)$

$\delta_{min} \leftarrow \min\{\delta^L, \delta^R\}$

If $(\delta_{min} = \delta^L)$ then $p_1 \leftarrow p_1^L$, $p_2 \leftarrow p_2^L$

else $p_1 \leftarrow p_1^R$, $p_2 \leftarrow p_2^R$

\\ p_1, p_2, δ_{min} store the "current" closest pair of pts. & their distance

$T \leftarrow \{p_i : |x_i - \bar{x}| \le \delta_{min}\}$ ← $O(n)$

$T^y \leftarrow T$ sorted by decreasing y-coordinate ← $O(n \log n)$

For each $p \in T_y$
  $U \leftarrow$ next 6 pts. in T_y
  $\hat{p} \leftarrow$ closest pt. in U to p
  If $d(\hat{p}, p) < \delta_{min}$
    $\delta_{min} \leftarrow d(\hat{p}, p)$
    $(p_1, p_2) \leftarrow (\hat{p}, p)$
} $O(n)$

Return (p_1, p_2, δ_{min})

Time taken:

$T(n) = O(n \log n) + O(n) + 2T(n/2)$

Solving this gives

$T(n) = O(n \log^2 n)$ ...

To get this to $O(n \log n)$, must take sorting outside the recursion.

i.e., sort S into S_x, S_y, & then somehow maintain sorted order throughout.

Then we get

$T(n) = O(n \log n) + T'(n)$

$T'(n) = O(n) + 2T'(n/2) = O(n \log n)$

& hence $T(n) = O(n \log n)$

PROBLEM 1: modify algo, so that it runs in time $O(n \log n)$

PROBLEM 2: remove assumption that no 2 pts. have the same x-coord, y-coord (did we really use this?)

PROBLEM 3: prove that the red box contains at most 6 pts.