Divide & Conquer

Q : Cost of manipulating integers

A : ① Modern $a \leftarrow b \cdot c$ as primitive operation

② Multiplication can be done $O(n^{\log_2 3})$ time

For this course, these gives two options for cost models :

① n-bit arithmetic operations as unit op.

② n- bit arithmetic ops using $n^{\Theta(1)}$ steps. (tedious)

③ $O(\log n)$ bit integer arithmetic.

↳ Convention : for problems on integers, assume $O(\log n)$ bits. ?

algo can use $O(\log n)$ arithmetics as unit cost.

Closest pair:

Description:

Assume x_i's and y_i's are disjoint

Proposition in $O(n^2)$

algo: $\min_{i,j} \text{dist}(P_i, P_j)^2$

complexity: $\binom{n}{2}$ pairs $\Rightarrow O(n^2)$

Prop: one-dimensional in $O(n\log n)$

Pf: idea: sorting

algo: input: $x_1, \ldots, x_n$

sort to: $\hat{x}_1 \leq \cdots \leq \hat{x}_n$

→ output: $\min_i$

prop: 2D closest pair in $O(n(\lg n)^2)$

Pf: idea ①: sorting $p_1, \dots, p_n$ by x-coordinates. $\longrightarrow$ fails

idea ②: divide and conquer

def $A, B \subseteq P$, $\quad dist(A,B) = \min\limits_{a \neq b} dist(a,b)$

$\Rightarrow$ $dist(P,P)$ is closest pair on P

def: $L, R \subseteq P$ where

$$L = \{ p_i : x_i \le \hat{x}_{\lfloor \frac{n}{2} \rfloor} \}$$

$$R = \{ p_i : x_i > \hat{x}_{\lfloor \frac{n}{2} \rfloor} \}$$

Lemma: $|L| = \lfloor \frac{n}{2} \rfloor$, $|R| = \lceil \frac{n}{2} \rceil$

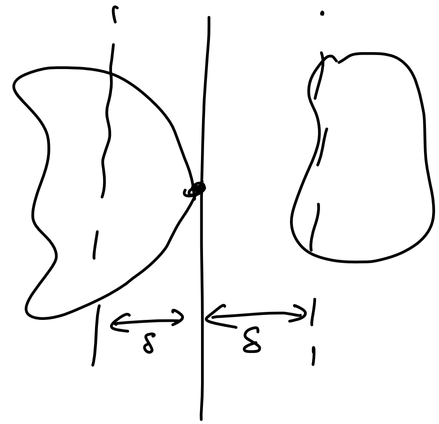
Lemma:

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

$dist(R,R)$, $dist(L,L)$ $\longrightarrow$ two recursive calls

$dist(L,R)$ $\longrightarrow$ hard ?

def: $S_\delta = \{ p_i : \hat{x}_{\lfloor \frac{n}{2} \rfloor} - \delta \le x_i \le \hat{x}_{\lfloor \frac{n}{2} \rfloor} + \delta \}$ δ-margin median strip of P



Lemma:

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

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

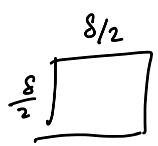
Is easier? $\Longrightarrow$ yes, we can use sorting in y-axis

prop: Sort S_δ by y-coord.

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

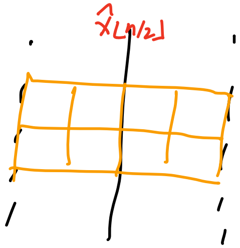
if $dist(\tilde{p}_i, \tilde{p}_j) \le \delta$, then $\boxed{|i-j| \le 8}$ Constant!

Pf: claim: any $\frac{\delta}{2} \times \frac{\delta}{2}$ box contains ≤ 1 points from L

Pf: 

box, if contains two points $p, q \in L$, then $\delta < \frac{\delta}{\sqrt{2}} \rightarrow$ contradiction.

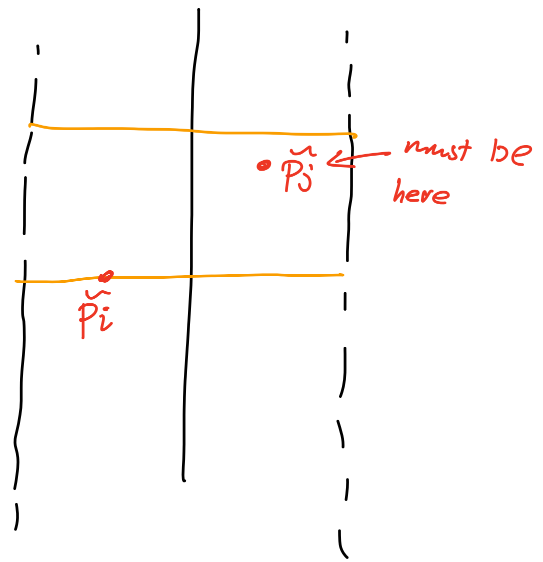
claim: any $2\delta \times \delta$ box B centered around $\hat{x}_{\lfloor \frac{n}{2} \rfloor}$ contains ≤ 8 points

Pf: 

at most 1 point in each box

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

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



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

$\Rightarrow \tilde{y}_j \le \tilde{y}_i + \delta \Rightarrow \tilde{p}_j \in B, \tilde{p}_i \in B$

$\overset{\text{claim}}{\Longrightarrow} |B \cap S_\delta| \le 8$

$\tilde{p}_i, \tilde{p}_j \in B \Rightarrow |i - j| \le 8$ in y-coord sorted order in S_δ

Algo for 2D closest pair in $O(n(\lg n)^2)$:

- If $|P| \le 3$, brute force $O(1)$
- sort P by x-coord and get P_x $O(n\log n)$
- Partition P into L and R (find median and classify all points) $O(n)$
- Recursively compute $\left\{ \begin{array}{ll} \text{dist}(L,L) & T(\frac{n}{2}) \\ \text{dist}(R,R) & T(\frac{n}{2}) \end{array} \right.$
- Let δ = min of $O(1)$
- Compute S_δ $O(n)$ by check if x-coord of p_i is in range
- sort S_δ by y-coord $O(n\log n)$
- compute closest pair in S_δ $O(n)$
- Output min $\{ \ , \delta$ $O(1)$

Complexity: $T(n) \le 2T(\frac{n}{2}) + O(n\log n) \le O(n(\lg n)^2)$

Thm: 2D closest pair in $O(n\log n)$

idea: sorting in every recursive call is wasteful.

→ instead, sort by x and y coord once in the beginning

Then, in each recursion, we can construct L_x, L_y, R_x, R_y in $O(n)$ (L_x and R_x are easy as we have median. Then, L_y and R_y can be constructed during partition.)

Complexity: $T(n) \le O(n\log n) + R(n)$

$$R(n) \le 2R\left(\frac{n}{2}\right) + O(n) \le O(n\log n)$$

Remark: can be done in $O(n)$ using randomization.