Closest Pair of Points

Computational Geometry.

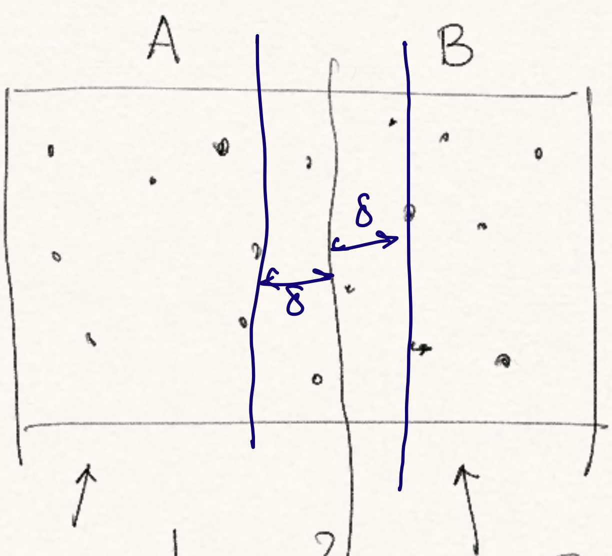
Given a set of points $P_1, \dots, P_n$ ($\mathbb{R}^2$), we want to find points that are "closest" in terms of Euclidean distance.

↳ Can we come up with a better than trivial algo.

$O\left(\binom{n}{2}\right) = O(n^2)$.

(x_1, y_1) (x_2, y_2)

$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$



$\mathbb{P} = \{P_1, \dots, P_n\}$

$\mathbb{P}_x = \{P_1, \dots, P_n\}$ s.t x

$\delta_A = \min\{\delta(P_i, P_j) \mid P_i, P_j \in A, i \neq j\}$

$\delta_B = \min\{\delta(P_i, P_j) \mid P_i, P_j \in B, i \neq j\}$.

$\delta = \min\{\delta_A, \delta_B\}$.

Merge task: Check if there are pairs P_i, P_j s.t $P_i \in A, P_j \in B$ and $\delta(P_i, P_j) < \delta$.

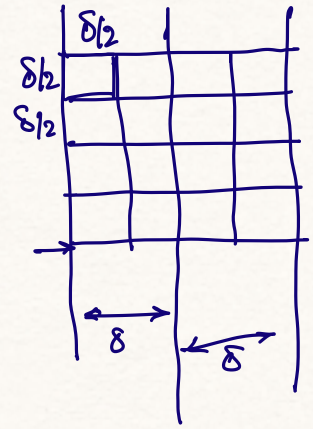
Claim: If such pairs (P_i, P_j) exist across the "border" (line $x = x^*$ where x^* is the max. x coordinate of all points in A) s.t $P_i \in A$ and $P_j \in B$, then P_i and P_j lie in a band of width 2δ with border as the center.

Let S be the set of points in the band. ← can be obtained using linear scan of $\mathbb{P}_x$

↳ Obtain S_y ← sorted based on incr. y-values.

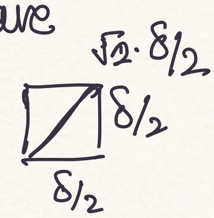
W.L.O.G: we can assume that no points have the x-coordinate or y-coordinate.

Claim: This 4×4 squares each of side $\frac{\delta}{2}$ can at most contain 16 points over all and each square can contain ≤ 1 point.



Each square of size $\frac{\delta}{2} \times \frac{\delta}{2}$ can have at most one point.

1. Max distance between 2 points in the same square can at most be $\sqrt{2} \cdot \frac{\delta}{2} = \frac{\delta}{\sqrt{2}} < \delta$



2. For any 2 points on the same side, the min distance is $\delta \leftarrow \delta = \min\{\delta_A, \delta_B\}$.

From contradiction, each $\delta/2 \times \delta/2$ square can contain at most one point.

Starting from the point a with least y-value in the band, draw the corresponding 4×4 grid.

- Find distance from a to all other points in the grid. (there are at most 15 other points)
- take the minimum and store it in an array <u>MinValues</u>

Repeat it for all points in the band in the increasing order of their y-values.

Compute the min value in the array MinValues.

Algorithm:

→ Sort the points based on their x-values and y-values.

→ Construct 2-smaller instances A and B each with $\frac{n}{2}$ points.

- → Recurse on sets A and B individually.
- → δ_A = min dist of points in A
- → δ_B : min dist of points in B

→ Set $\delta = \min\{\delta_A, \delta_B\}$

→ Construct a line $x = x^*$ where x^* = max x-value of all points in A.

→ Construct a band and the corresponding set of points S that has width δ on either side of $x = x^*$.

→ Consider the set of points in the band in the incr. order of y-values. (call this ordered seq: S_y).

→ For all $u \in S_y$:

- draw the 4x4 grid whose bottom is at the y-value of u.
- Pick all points in S that lie in that grid. (only need to check 15 points in S_y).
- compute the dist from u to every other point in the grid and store it in array D.

→ Let $\delta' = \min\limits_{u \in S_y} \{ D[u] \}$

→ If $\delta' < \delta$ return δ' and the corr. points else return the pair giving us δ.

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + \underbrace{T_{merge}}_{(15+c)\,n}$$

$O(n)$