January 22, 2025 10:38 AM

CS 473 Algorithms

http://courses.grainger.illinois.edu/cs473/sp2025

10 HWs	(each with 3 problems)	30%
Midterm 1	(Feb 24 Mon 7p-9p)	20%
Midterm 2	(Apr 7 Mon 7p-9p)	20%
Final Exam		30%

- no late HW accepted (but will drop 5 lowest problem scores)
- may work in groups of ≤3
- read academic integrity page

Topics

- Divide & Conquer (e.g. FFT)
- Dynamic Programming
- Randomized Algs
- Optimization Problems: matching, flows, LP
- NP-Completeness and lower bounds
- Approx. Algs

DIVIDE & CONQUER

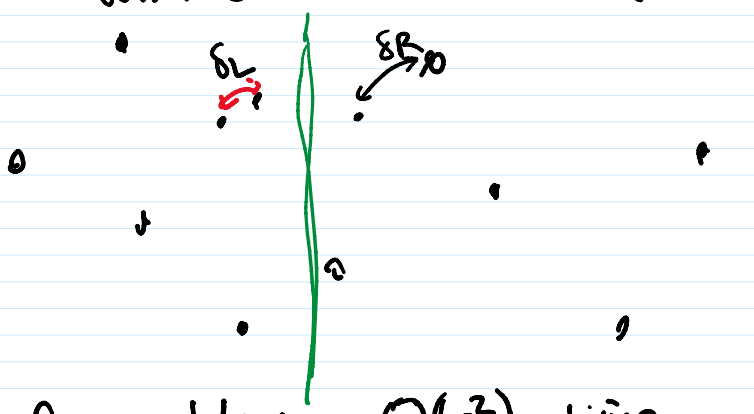
divide into subproblems of same type
Solve each subproblem recursively
Combine

e.g. quicksort, mergesort

Closest Pair

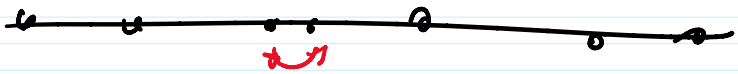
Problem Given set P of n points in 2D (x-coord, y-coord),
find $p, q \in P$ $(p \neq q)$
with smallest distance $d(p,q) = \sqrt{(p.x - q.x)^2 + (p.y - q.y)^2}$

find $p, q \in P$ $(p \neq q)$
with smallest distance $d(p,q) = \sqrt{(p.x - q.x)^2 + (p.y - q.y)^2}$



brute-force algm: $O(n^2)$ time better?

in 1D:



$O(n \log n + n) = \boxed{O(n \log n)}$

sort — check consecutive pairs

in 2D?

Shamos' Algm (1975)

idea 0 - divide by median vertical line

ClosestPair(P)

1. if $|P| \leq$ const (3) return answer by brute-force
2. x_m = median x-coord — $O(n)$ time
3. $P_L = \{p \in P : p.x \leq x_m\}$
4. $P_R = \{p \in P : p.x > x_m\}$ — $O(n)$ time
5. δ_L = ClosestPair(P_L)
6. δ_R = ClosestPair(P_R)

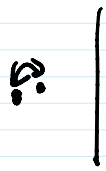
5. $\delta_L = \text{ClosestPair}(P_L)$

6. $\delta_R = \text{ClosestPair}(P_R)$

7. $\delta = \min(\delta_L, \delta_R)$

⋮

Case1.



δ_L is ans

Case2:

δ_R is ans

Case3:

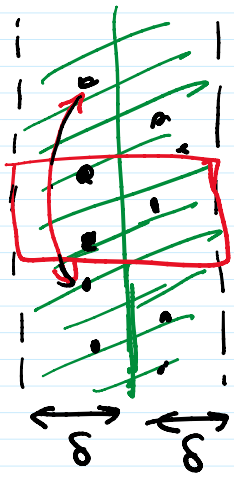
?

need to check pairs between P_L & P_R

$\frac{n}{2} \times \frac{n}{2} = \Theta(n^2)$ pairs

1st idea - only need to check pairs inside strip of width 2δ

$x_m - \delta \leq x \leq x_m + \delta$

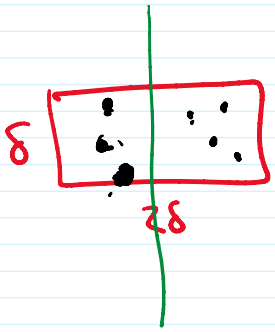


2nd idea - only need to check pairs inside some $2\delta \times \delta$ rectangle

$x_m - \delta \leq x \leq x_m + \delta$

$t \leq y \leq t + \delta$

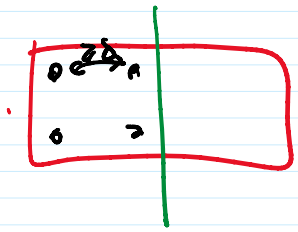
for some t.

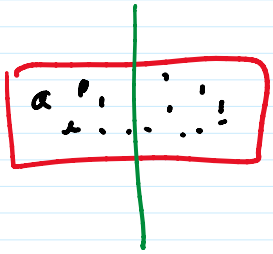


Lemma Each such $2\delta \times \delta$ rectangle

Lemma Each such $2\delta \times \delta$ rectangle may contain ≤ 8 points.

Pf:





left $\delta \times \delta$ square can contain ≤ 4 pts because any 2 pts have dist $\geq \delta_L \geq \delta$.

right $\delta \times \delta$ square " " "

□

$\vdots$

$O(n \log n)$ { 8. $\langle p_1, \ldots, p_l \rangle$ = points in $\{p \in P : x_m - \delta \leq p.x \leq x_m + \delta\}$ sorted by y

$O(7n) = O(n)$ { 9. for $i = 1$ to l do
10. for $j = i+1, i+2, \ldots$, as long as $p_j.y \leq p_i.y + \delta$
$\delta = \min\{\delta, d(p_i, p_j)\}$

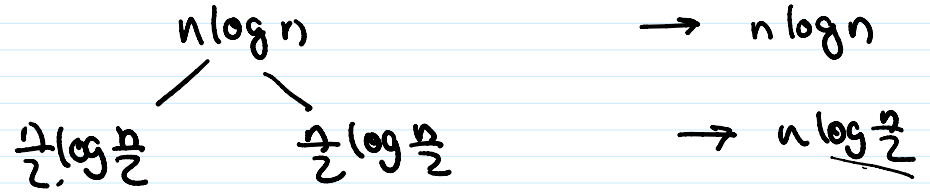
↑ up to ≤ 7 iterations

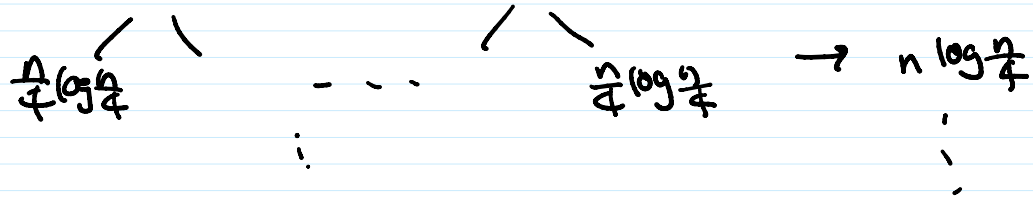
11. return δ

Analysis:

$$T(n) = \begin{cases} 2T\left(\frac{n}{2}\right) + O(n \log n) & \text{if } n \geq 3 \\ O(1) & \text{else.} \end{cases}$$

$\Rightarrow$





total: $\boxed{O(n\log^2 n)}$

Rmk- can be improved by pre-sorting all y's

$\Rightarrow \quad T(n) = 2T(\frac{n}{2}) + O(n)$

$\Rightarrow \quad O(n\log n) + \underbrace{O(n\log n)}_{\text{pre-sort}} = \boxed{O(n\log n)}$

(3D : $O(n\log n)$
or any const dim)