Lecture 27

CSE 331

Apr 8, 2020

Mid-term temp grade assigned

note @538

Mid-semester temp grades released

Your temp letter grades have been assigned. To calculate your grade, you must first calculate your raw sc

- Add up your HW scores from HW1-4 to calculate H (out of a max of 400)
- Let Q be your quiz 1 score (out of a max of 10)
- Let M be your mid-term score (out of a max of 100).

Then R is calculated as follows (out of a maximum possible of 38.5):

$$R = H * \frac{35}{400} * \frac{4}{10} + Q * \frac{2.5}{10} + M * \frac{22}{100}$$

(The above does not fully follow the grading policy since it does not drop any HW score and does not sub
the course, I think the above is fine as a proxy.)

Here are the stats of the raw score:

- Average: 19.12
- Median: 19.82
- Max: 35.02

Now to calculate your letter grade, read it off from the following map:

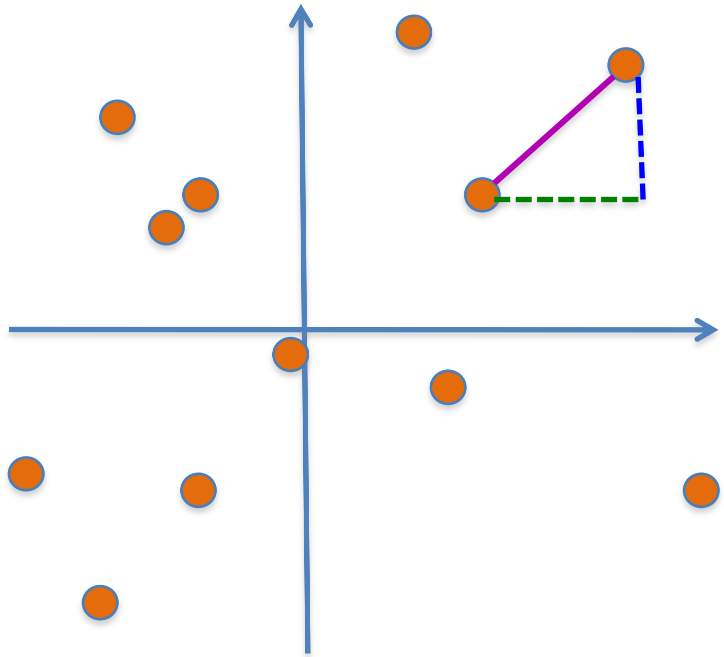
- A: $R \in [34.65, 38.5]$
- A-: $R \in [26.57, 34.65)$
- B+: $R \in [23.49, 26.57)$
- B: $R \in [19.12, 23.49)$
- B-: $R \in [17.33, 19.12)$
- C+: $R \in [15.40, 17.33)$
- C: $R \in [13.48, 15.40)$
- C-: $R \in [11.94, 13.48)$
- D+: $R \in [10.40, 11.94)$
- D: $R \in [8.09, 10.40)$
- F: $R \in [0, 8.09)$

Closest pairs of points

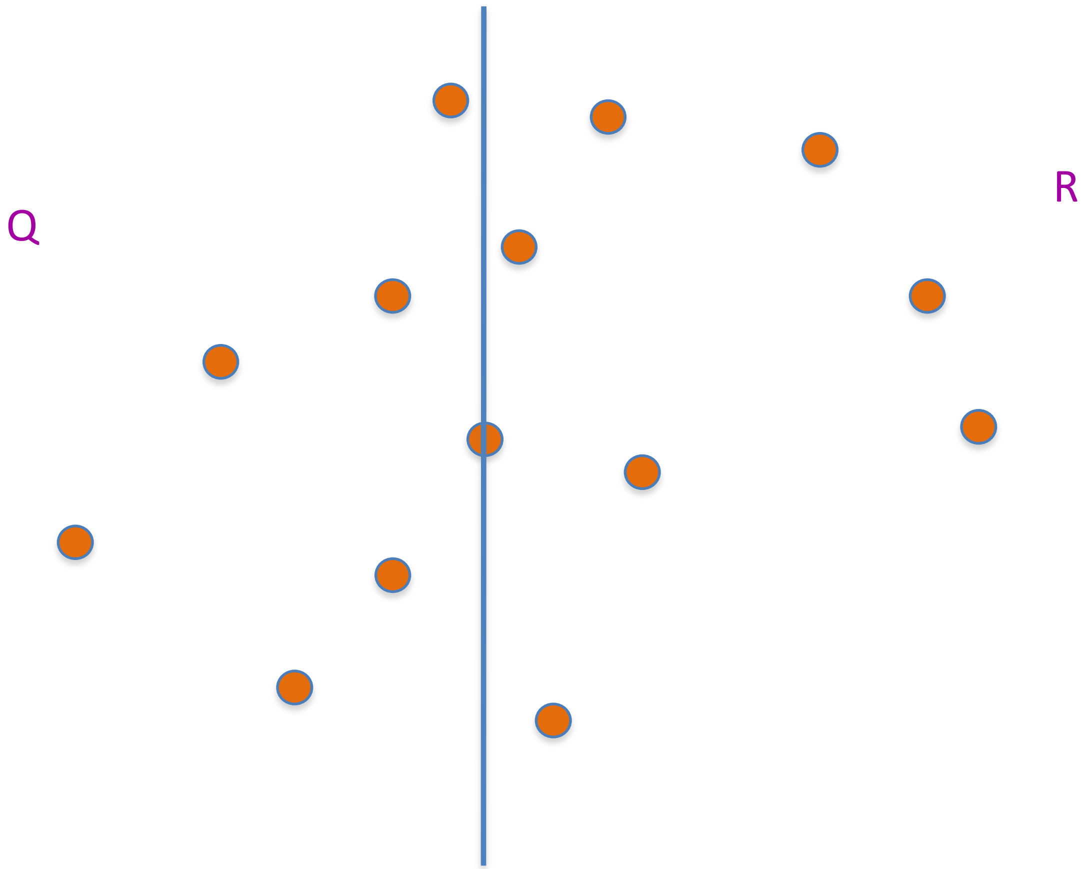
Input: n 2-D points $P = \{p_1,\ldots,p_n\}$; $p_i=(x_i,y_i)$

$$d(p_i,p_j) = ((x_i-x_j)^2+(y_i-y_j)^2)^{1/2}$$

Output: Points p and q that are closest

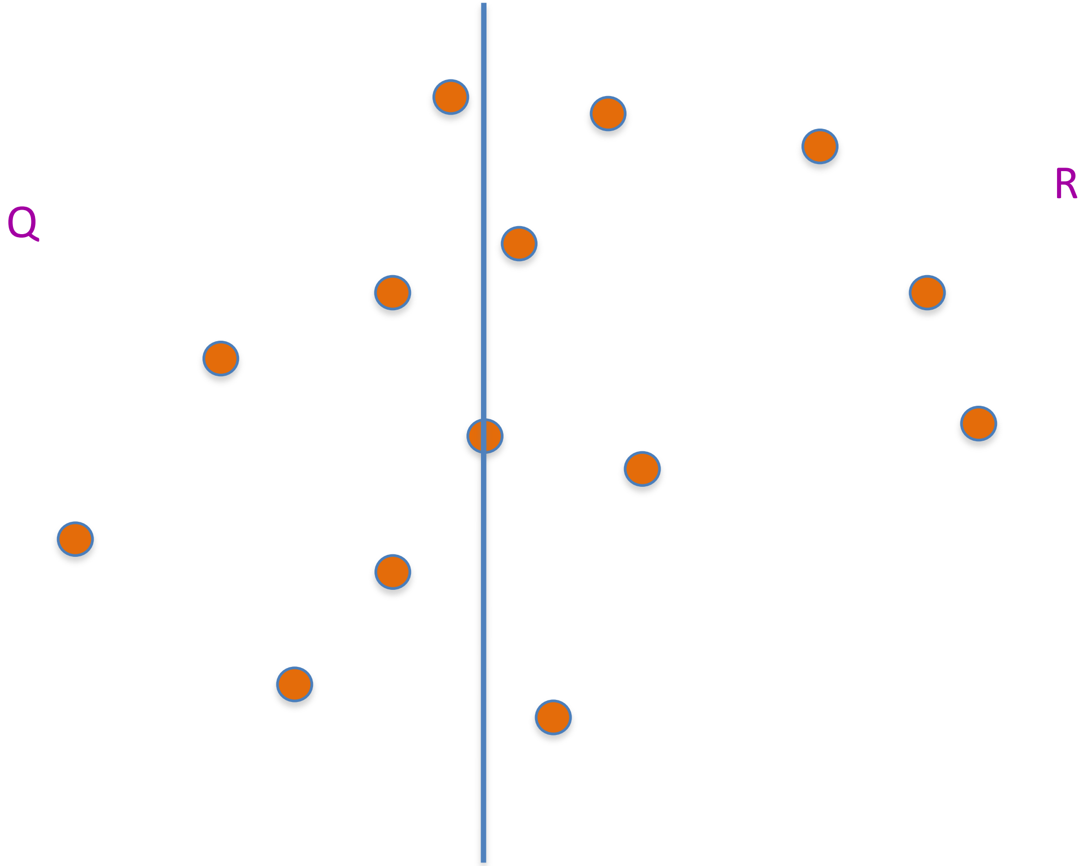


Dividing up P



First n/2 points according to the x-coord

Recursively find closest pairs



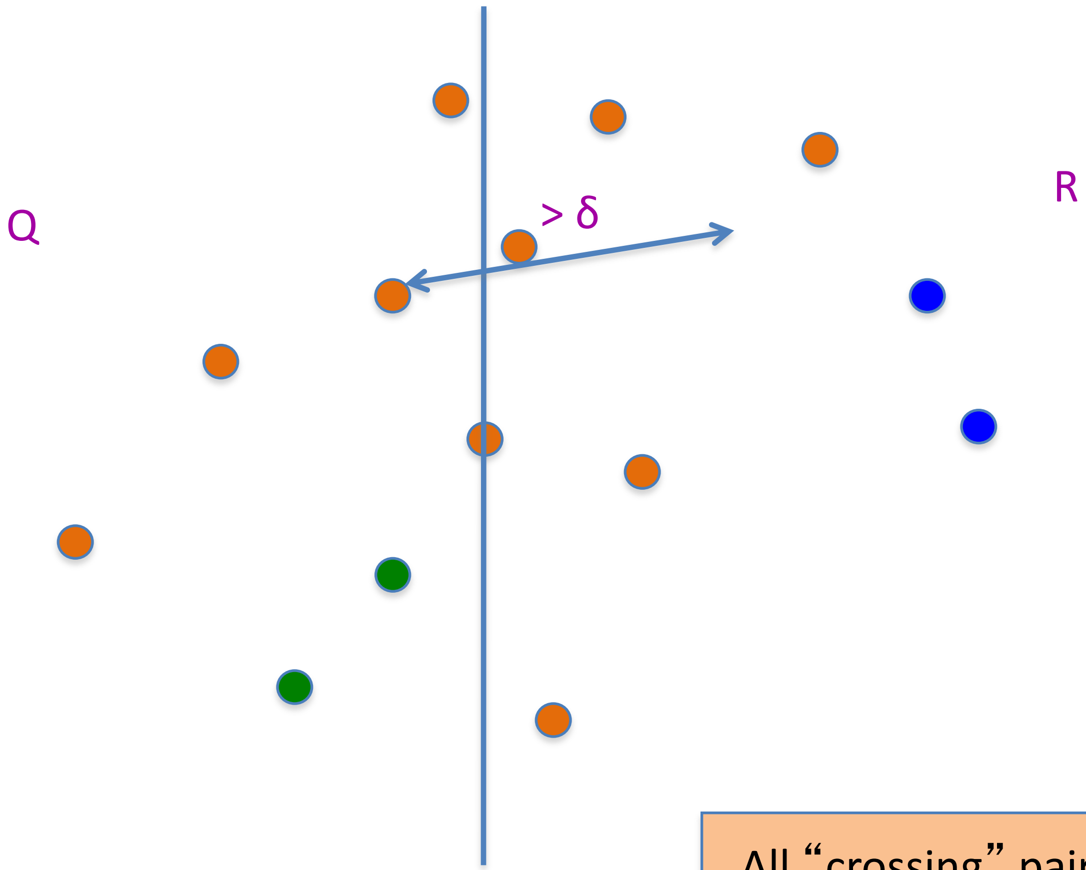
δ = min (**blue**, **green**)

An aside: maintain sorted lists

P_x and P_y are P sorted by x-coord and y-coord

Q_x, Q_y, R_x, R_y can be computed from P_x and P_y in $O(n)$ time

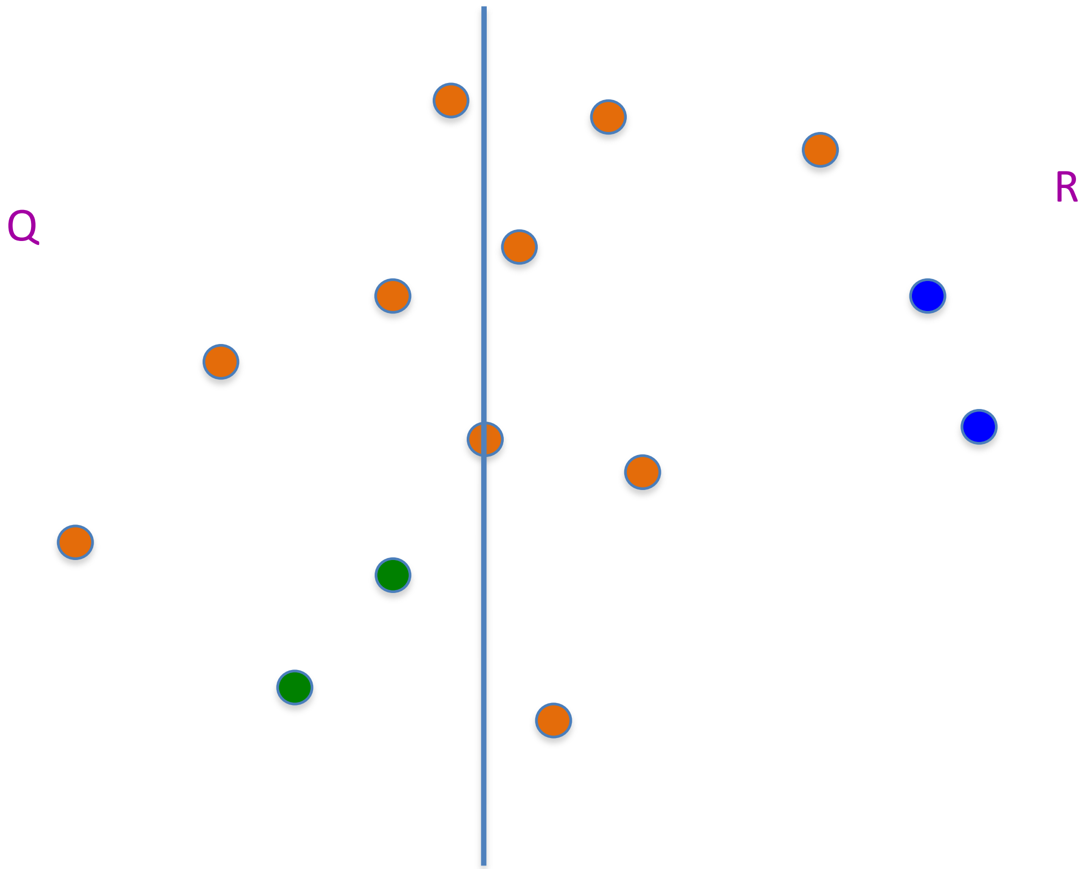
An easy case



All "crossing" pairs have distance > δ

δ = min (**blue**, **green**)

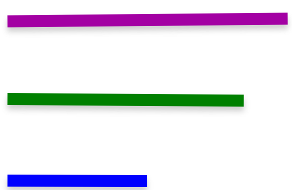
Life is not so easy though

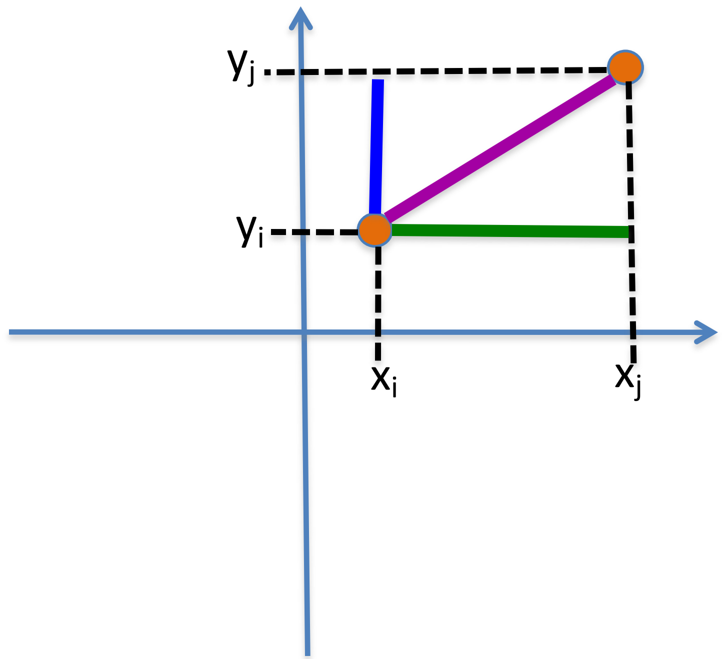


δ = min (**blue**, **green**)

Euclid to the rescue (?)

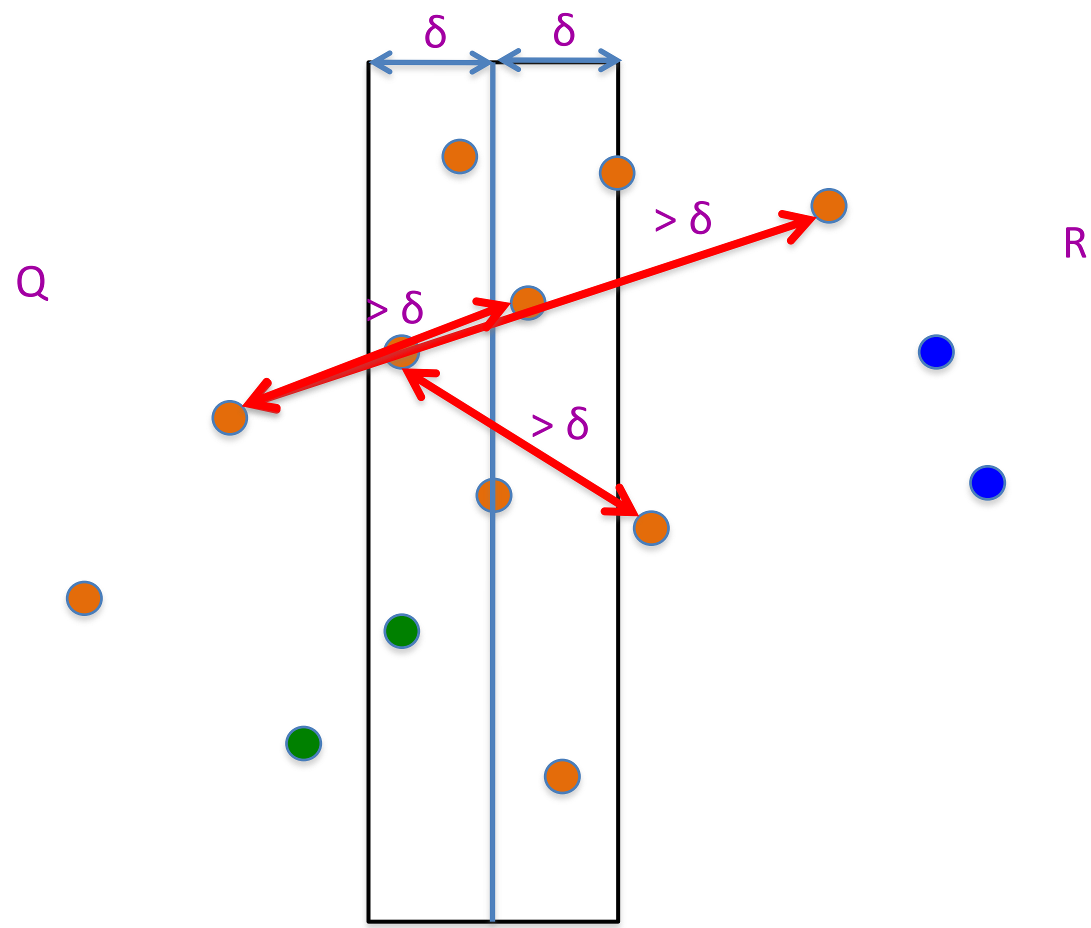
$$d(p_i,p_j) = ((x_i-x_j)^2+(y_i-y_j)^2)^{1/2}$$





The distance is larger than the **x** or **y**-coord difference

Life is not so easy though



δ = min (**blue**, **green**)

All we have to do now

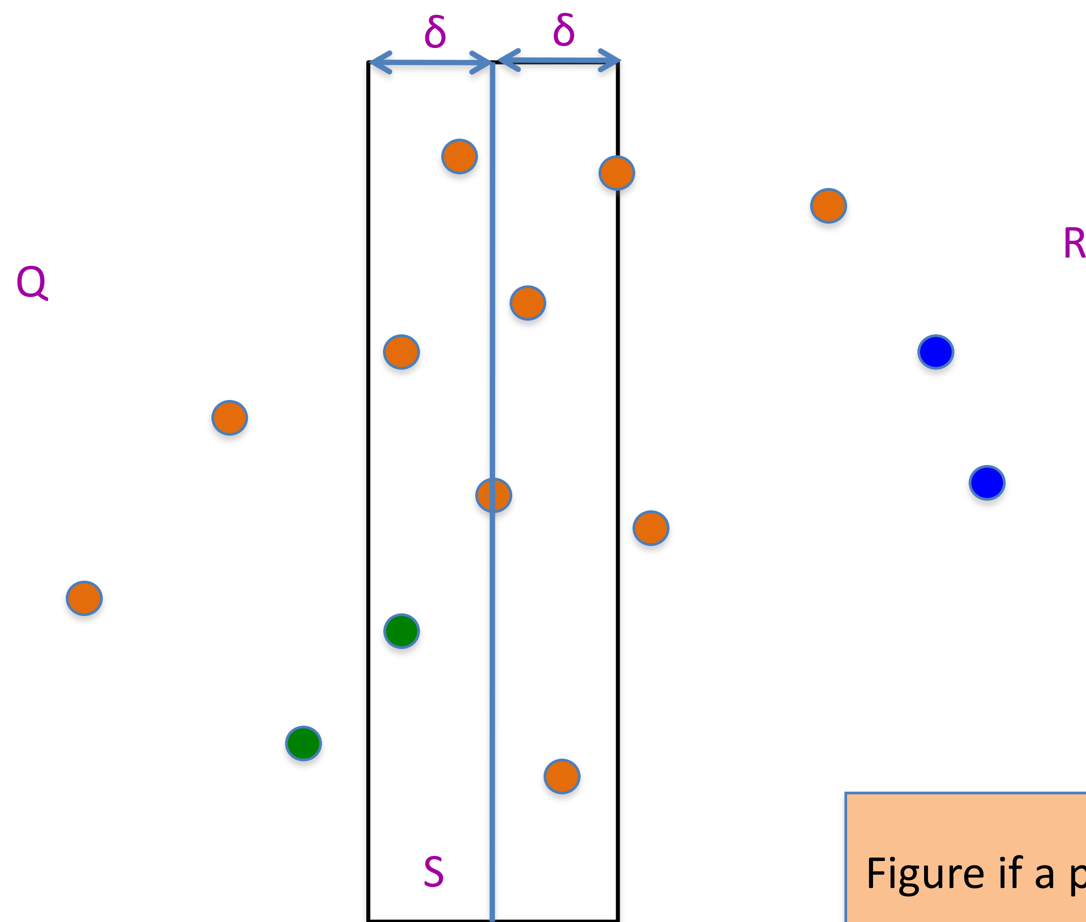


Figure if a pair in S has distance < δ

δ = min (**blue**, **green**)

The algorithm so far…

Input: n 2-D points $P = \{p_1,\dots,p_n\}$; $p_i=(x_i,y_i)$

$O(n \log n) + T(n)$

Sort P to get P_x and P_y — $O(n \log n)$

Closest-Pair (P_x, P_y)

If $n < 4$ then find closest point by brute-force

Q is first half of P_x and R is the rest — $O(n)$

Compute Q_x, Q_y, R_x and R_y — $O(n)$

(q_0,q_1) = Closest-Pair (Q_x, Q_y)

(r_0,r_1) = Closest-Pair (R_x, R_y)

$\delta = \min(\, d(q_0,q_1), d(r_0,r_1)\,)$ — $O(1)$

S = points (x,y) in P s.t. $|x - x^*| < \delta$ — $O(n)$

return Closest-in-box (S, (q_0,q_1), (r_0,r_1)) — Assume can be done in $O(n)$

$T(< 4) = c$

$T(n) = 2T(n/2) + cn$

$O(n \log n)$ overall

Rest of today's agenda

Implement Closest-in-box in $O(n)$ time