CS314 – Divide + Conquer

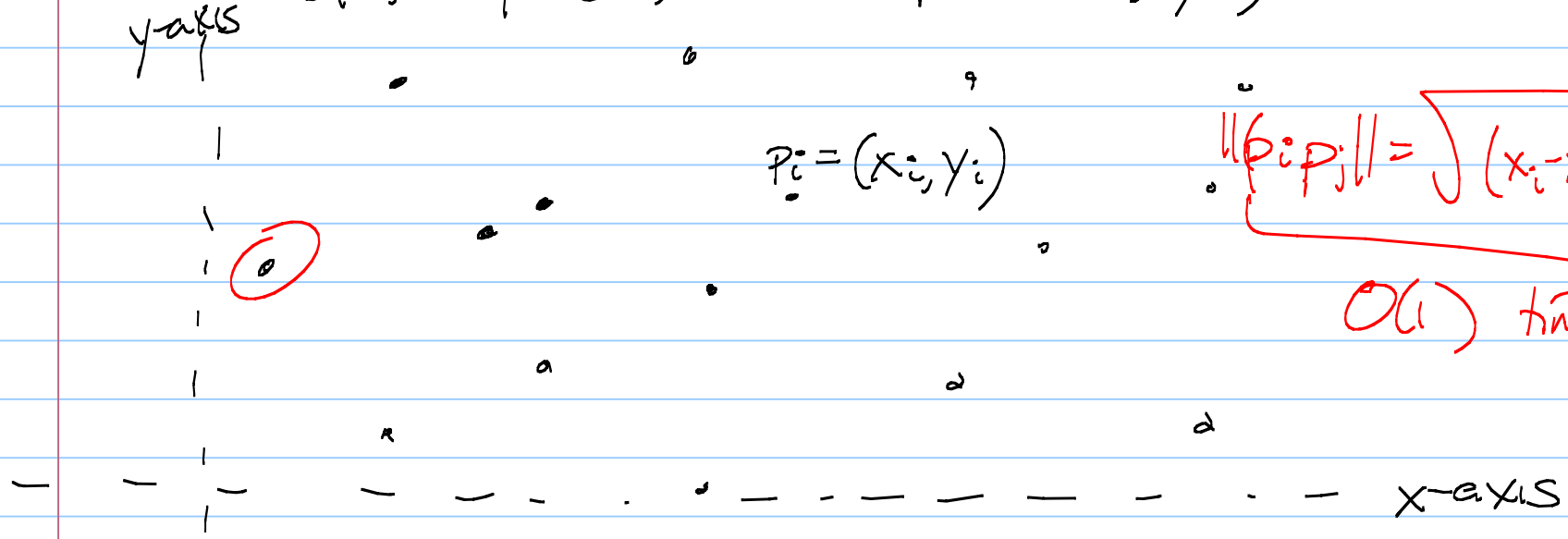
Note Title 2/3/2010

- Oral grading tomorrow
- New homework out tomorrow or Friday
- Comments on HW1:
  - Proof of correctness!

Closest Pair of Points

Let P be a set of points in $\mathbb{R}^2$.

$P = \{p_1, \dots, p_n\}$, and $p_i = (x_i, y_i)$

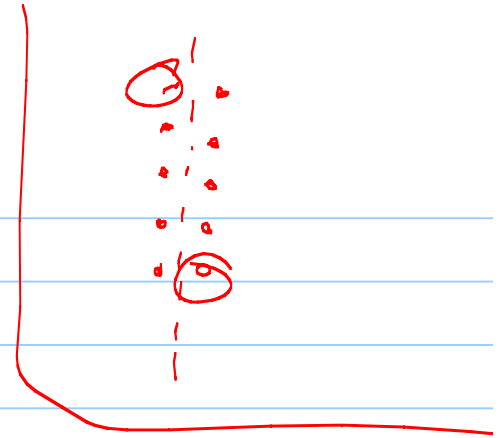


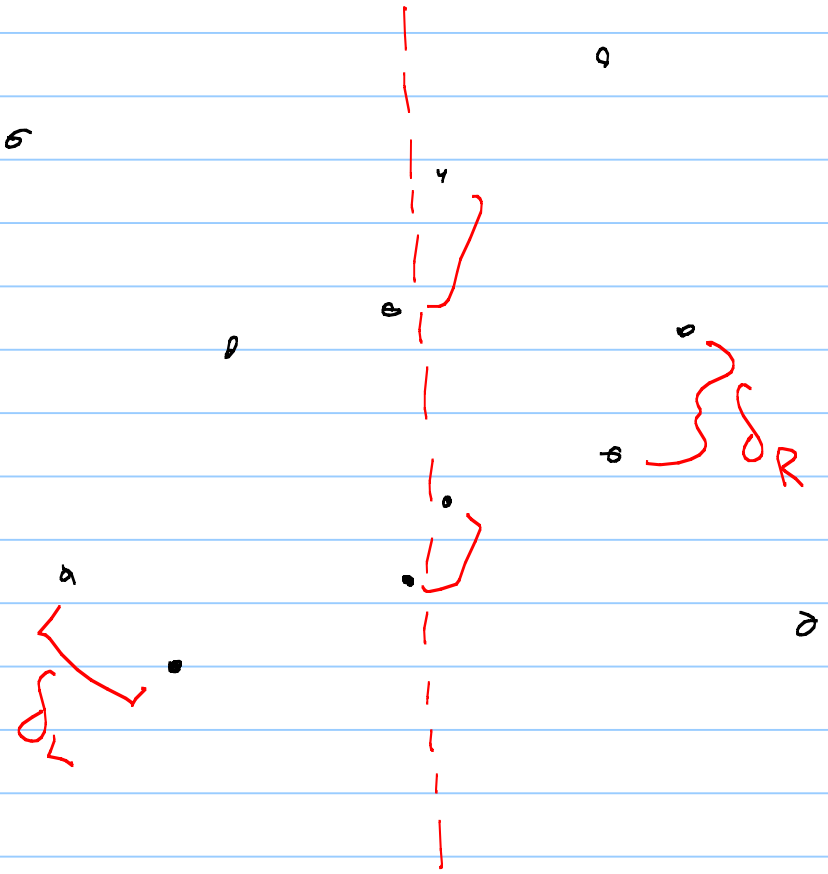
$$\underbrace{\|p_i p_j\| = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}_{O(1) \text{ time}}$$

Q: What is the closest pair of points?

$O(n^2) \leftarrow$ naive

Divide + Conquer Approach





Designing an algorithm

- Assume no 2 points have same x- or y-coord.
- Start by having 2 lists for points P:
  - P_x - points sorted by x-coordinate
  - P_y - points sorted by y-coordinate

  (make sure these have indices of position in other list, so we can keep track easily)

So: Set up recursion

$O(n \log n)$ - At beginning, sort P twice to get P_x & P_y

- Find dividing line ℓ - how? look in $P_x[\frac{n}{2}]$

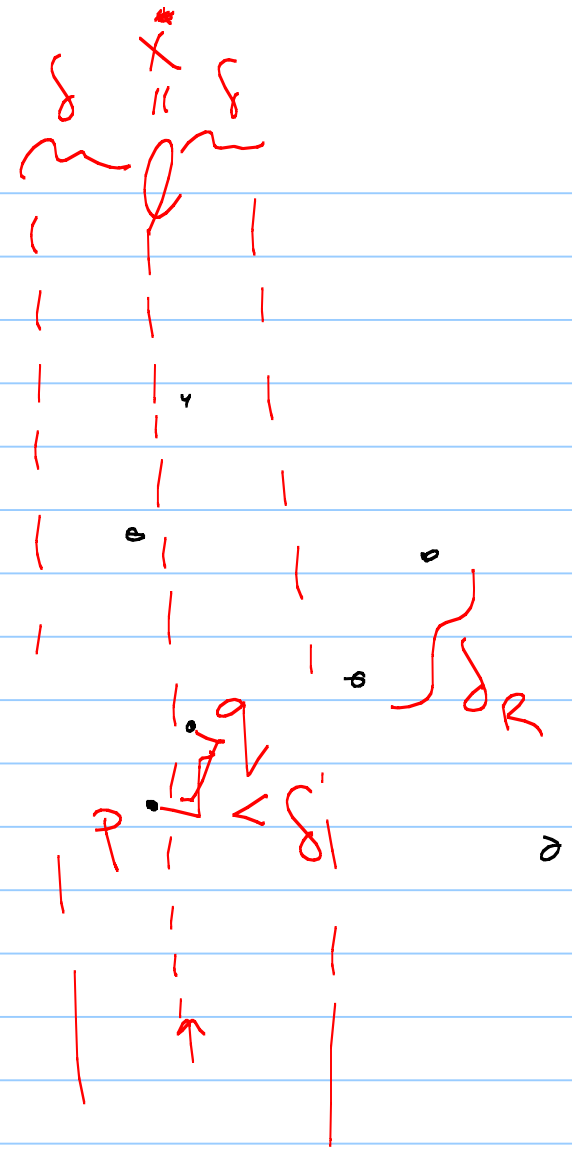
$O(n)$ → & compute R & L - points to left & right of dividing line

$R = P_x[0 \ldots \frac{n}{2}]$, $L = P_x[\frac{n}{2}+1 \ldots n]$

$O(n)$ - Compute R_x & R_y, L_x & L_y ← already ordered in P_y, so $O(n)$

- Recursively compute closest distance in R & in L

→ Now have δ_R & δ_L


δ
x*
δ
δ_R
q
p
< δ'
δ_L

Now what?

<u>Prop</u>: Let $\delta = \min(\delta_R, \delta_L)$. If there is $p \in R$ and $q \in L$ with $d(p,q) < \delta$, then p & q must be within δ of our dividing line ℓ.

<u>pf</u>: Spps $p = (p_x, p_y)$ and $q = (q_x, q_y)$ exist, $d(p,q) < \delta$.

Let our line ℓ be $\underline{\ell = x^*}$.

know: $\underbrace{q_x - p_x < \delta}$

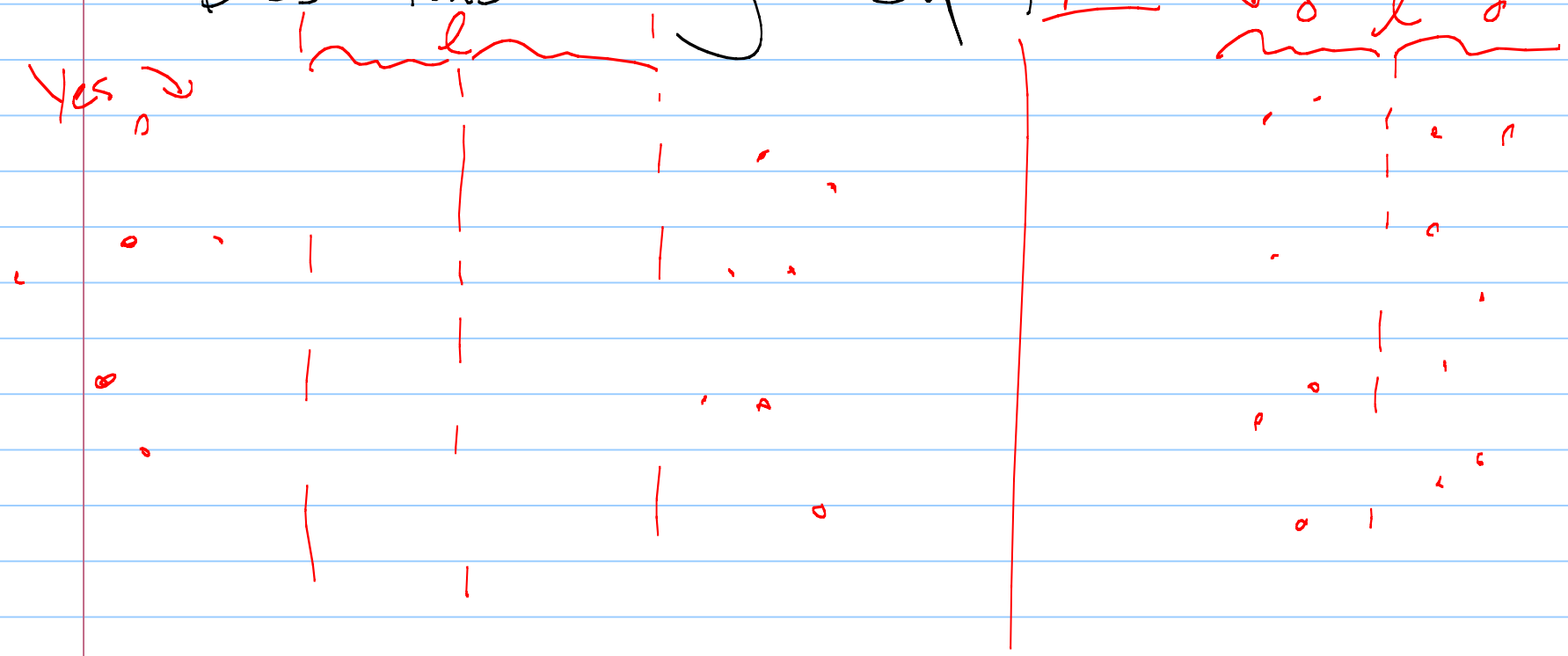
$p_x \le x^* \le q_x$

$\Rightarrow \delta > \underbrace{q_x - p_x} \ge \underbrace{p_x - x^*}$

$\Rightarrow \delta > p_x - x^*$, so p_x is within δ of ℓ $\blacksquare$

So - only need to search a "narrow" band around ℓ.

Does this always help? No:



One more idea:

$O(n)$ → Let S be points of P near ℓ, sorted by y-coordinate.

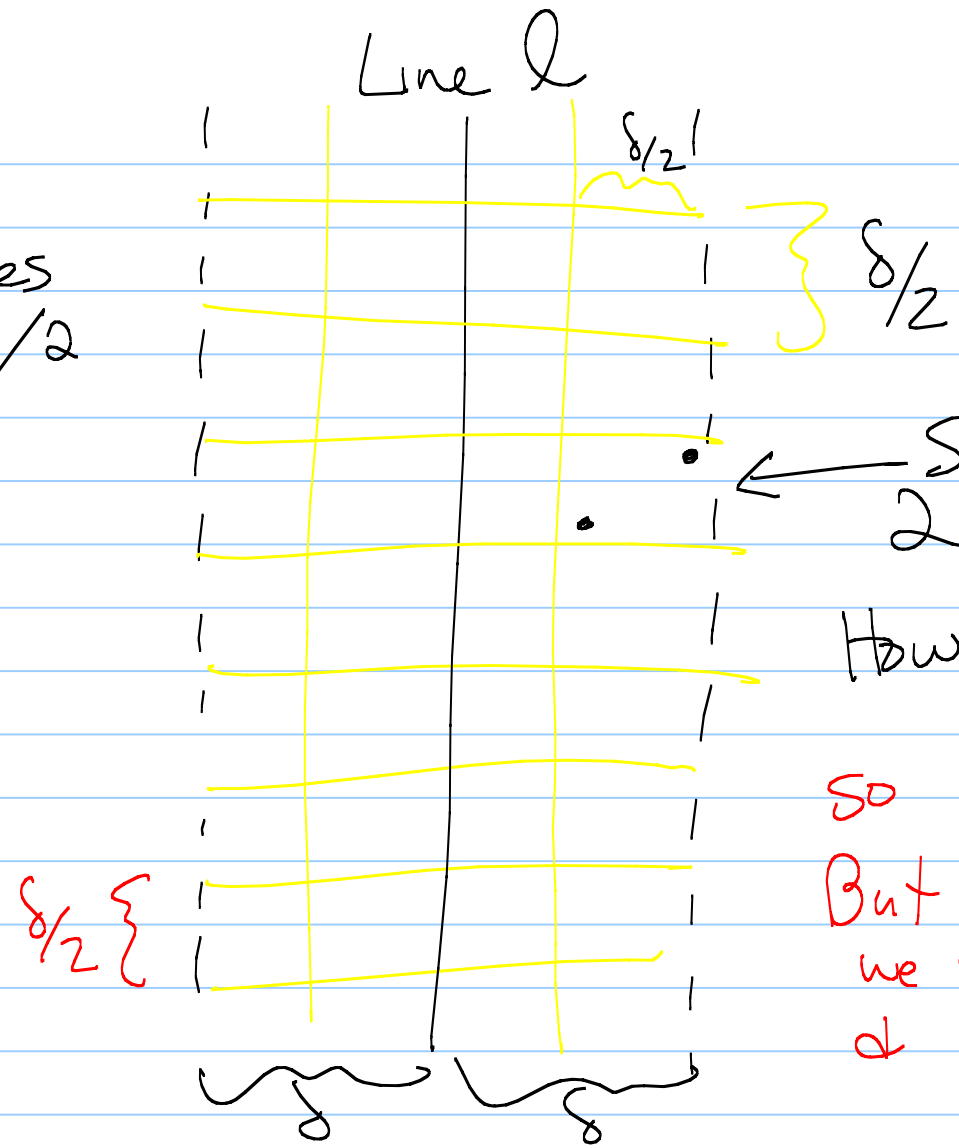
within δ

<u>Key Lemma</u>: If $s, s' \in S$ are within δ of each other, then s & s' are within 5 positions of each other in S.

Why??

Proof:

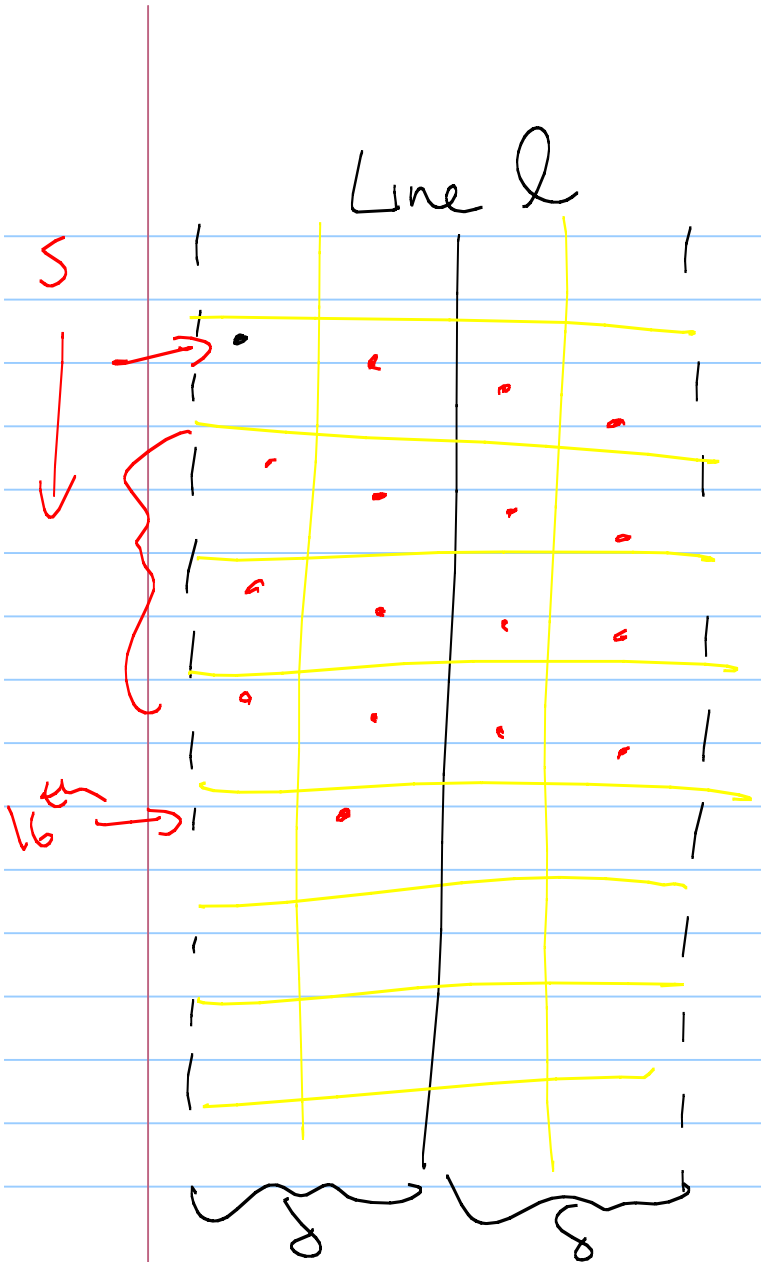
partition into boxes of size $\delta/2$



Spps we have 2 points inside a box. How far apart are they?

so $\leq \frac{\delta}{2}\sqrt{2} < \delta$

But these 2 points are from same side of ℓ, & δ was min distance!



So at most 1 point per box!

Now consider 2 points more than 15 positions away in S.

Nearest that point could be is 3 "boxes" down.

3 empty rows mean that point is <u>at least</u> $\frac{3\delta}{2} > \delta$

So it can't be closer than δ away! $\blacksquare$

Our algorithm:

$15n = O(n)$

After constructing S, compute distance between every $s \in S$ and the next 15 elements.

Key Lemma $\Rightarrow$ These are only possible candidates for getting a distance $< \delta$.

How long does this take?

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

$$T(n) = 2T\left(\frac{n}{2}\right) + \underbrace{O(n)} \Rightarrow T(n) = O(n \log n)$$

Algorithm (sketch) - see p. 230 for pseudocode

- Initial sort & divide P in half
- Two recursive calls (but <u>don't</u> need to sort)
- Compute S & compare each pt to 15 points after it in S

Proof of Correctness:

Induction on n:

Base case - clear

For n points, our IH says δ_R & δ_L are closest distance for points only in R & L, resp.

By our key lemma we look at all possible points $p \in R$ & $q \in L$ with $d(p,q) < \min(\delta_R, \delta_L)$.

Since closest pair of points are either both in L, both in R, or one on each side, we are done - either our recursive calls found distance correctly, or our scan of S found it. $\square$