COL 351: Analysis and Design of Algorithms

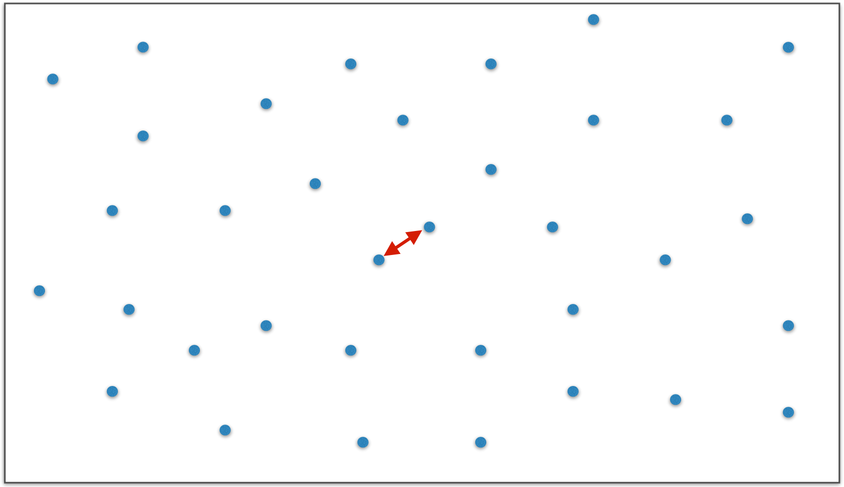
Lecture 25

Closest Pair of Points (or Minimum pairwise distance)

Given: A set P of n points in x-y plane.

Output: A pair of points in P at minimum distance, or $\min_{a \neq b \in P} distance(a, b)$.

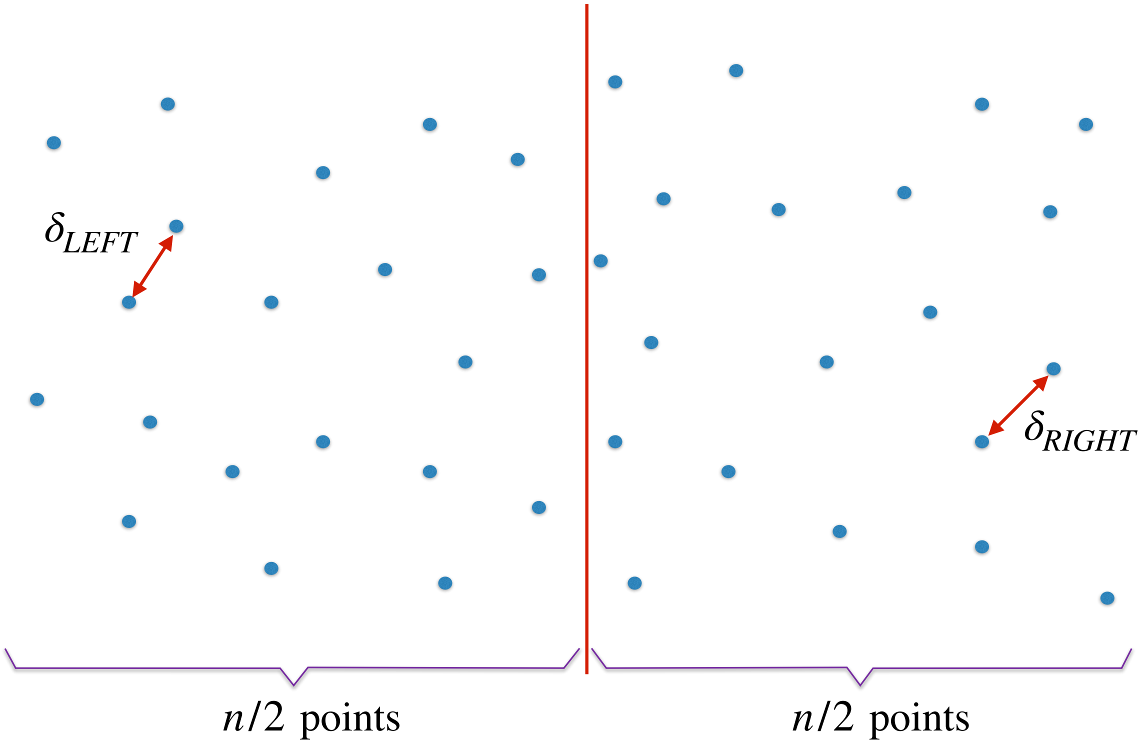
Example:



TRIVIAL : $O(n^2)$

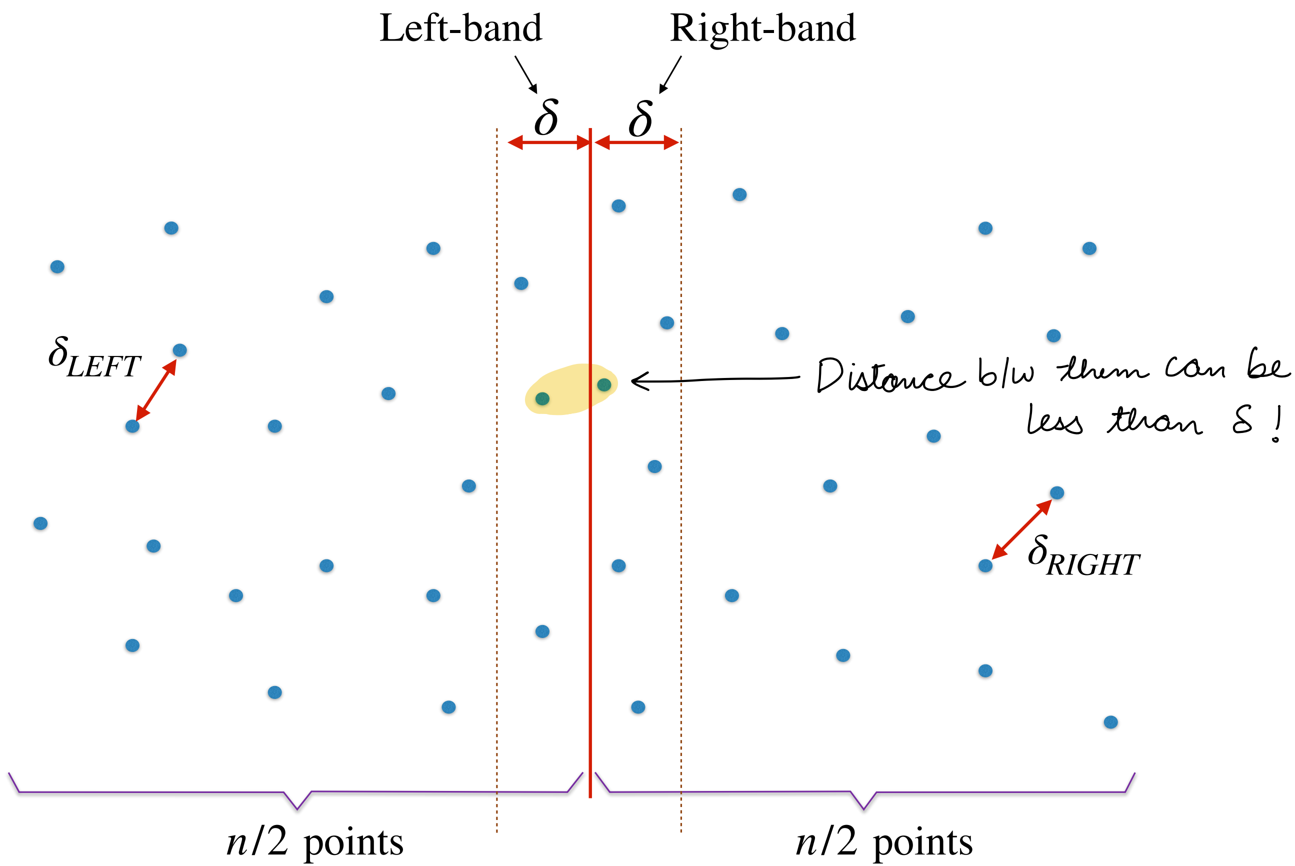
Divide and Conquer (Divide step)

Time to divide
= Time to find Median
= $O(n)$



$\delta = \min(\delta_{LEFT}, \delta_{RIGHT})$

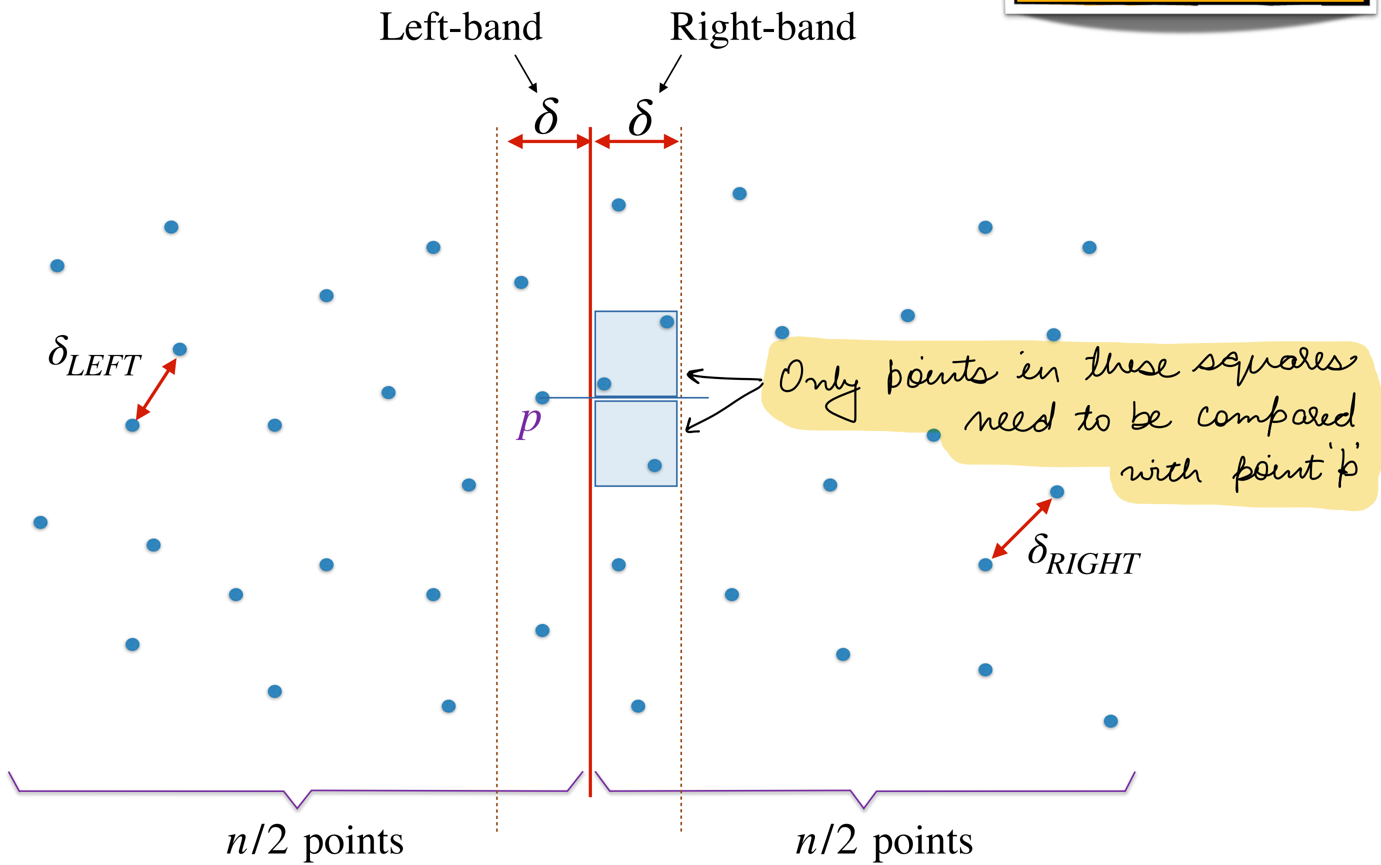
Divide and Conquer



$\delta = \min(\delta_{LEFT}, \delta_{RIGHT})$

Divide and Conquer

Both squares contain at most 4 points



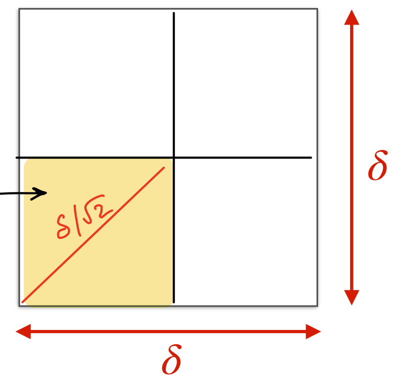
$\delta = \min(\delta_{LEFT}, \delta_{RIGHT})$

Subproblem

Given: A square satisfying all points in it are separated by distance at least δ.

Ques: How many points can lie in interior of square?

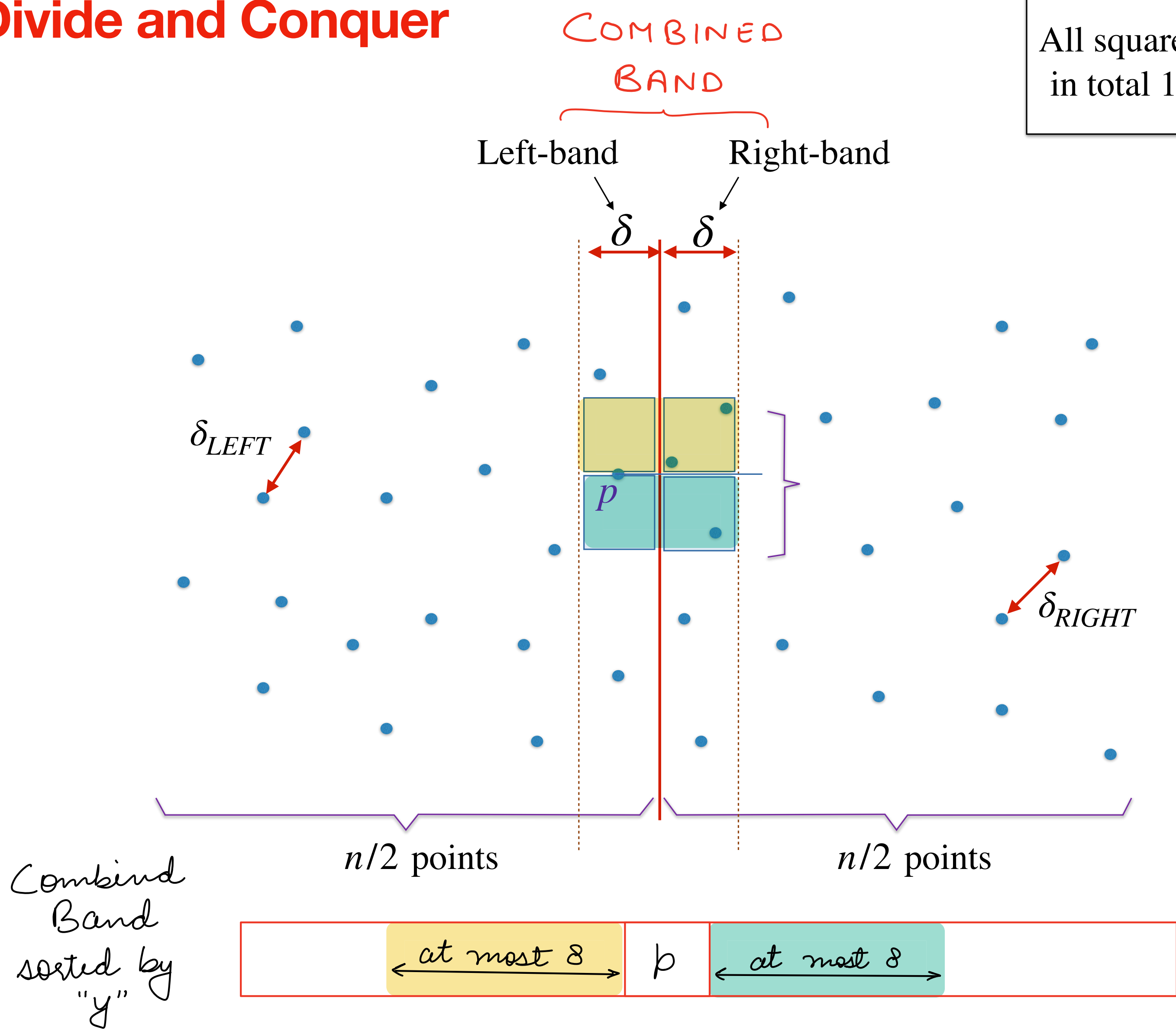
At most one point lies here



Ans: 4

Divide and Conquer

All squares contain in total 16 points!



Algorithm

$$T(n) = \underbrace{O(n)}_{\text{Divide}} + \underbrace{2\,T(n/2)}_{\text{Subproblems}} + \underbrace{O(n \log n)}_{\text{Combine}}$$

Can be improved to $O(n)$

MinPairwiseDistance(P)

1. **If** ($|P| = 1$) return ∞
2. x_{MED} = Median of points in P according to x-coordinate
3. (P_{LEFT} , P_{RIGHT}) = Partition of P by x_{MED}
4. δ_{LEFT} = MinPairwiseDistance(P_{LEFT})
5. δ_{RIGHT} = MinPairwiseDistance(P_{RIGHT})
6. $\delta = \min(\delta_{LEFT}\ ,\ \delta_{RIGHT})$
7. L-band = δ-length band of P_{LEFT}
8. R-band = δ-length band of P_{RIGHT}
9. C = Points in (L-band $\cup$ R-band) sorted w.r.t. to y-coordinate .
10. **ForEach**($p \in C$):

    $C(p)$ = Points in C whose y-coordinate differ from that of p by at most δ.

    δ_p = minimum distance b/w p and points in $C(p)$.

    If ($\delta_p < \delta$): $\delta = \delta_p$
11. Return δ

$C(p)$ is subset of at most 8 predecessors and 8 successors of p in 'C'

when points are sorted by y coordinate

Recurrence relation is $T(n) = 2T(n/2) + cn$, but why?

During preprocessing we can sort points by y-coordinate

This will give us an $O(n \log n)$ term overall.

Minimum pairwise distance

Result: Given a set P of n points in x-y plane, we can compute $\min_{a \neq b \in P} distance(a, b)$ in $O(n \log n)$ time.

Ques: Can you get $O(n \log_2(n))$ bound without using Median finding algorithm?

Yes! During preprocessing we can sort according to x coordinate as well.

Randomized Quick Sort

$RandQuickSort(L)$

x = Random element of list L;

Initialise L1 and L2 to be empty lists;

For each ($y \in L \backslash x$):
    If ($y \leqslant x$) : L1.append(y);
    If ($y > x$) : L2.append(y);

Return $RandQuickSort$(L1) $\circ\ x \circ$ $RandQuickSort$(L2);

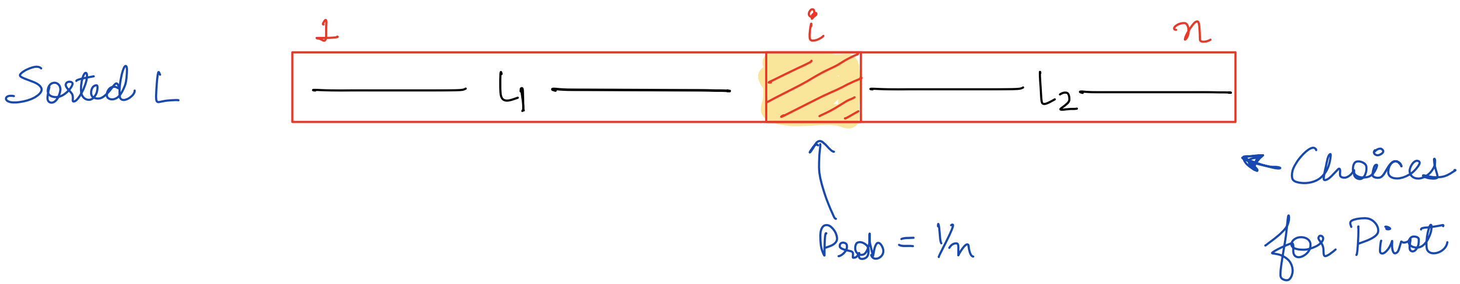
$T(n)$:= Expected time to sort n elements

How to bound $T(n)$?

$$T(n) = \frac{1}{n} \sum_{i=1}^{n} \Big(T(i-1) + T(n-i) \Big) + cn$$

To show:

$$T(n) \leqslant dn \log_2 n$$



If Pivot has rank i then

$$\text{size}(L1) = i-1$$

$$\text{size}(L2) = n-i$$

$$T(n) = \frac{1}{n} \sum_{i=1}^{n} \Big(T(i-1) + T(n-i) \Big) + cn$$

To show:
$T(n) \leqslant dn \log_2 n$

How to ANALYZE ?

Hint :

$T(i) \leq i \log(n/2)$ for $i \leq n/2$

$T(i) \leq i \log(n)$ for $n/2 < i \leq \frac{n}{2}$

$$T(n) = \frac{1}{n}\sum_{i=1}^{n}\Big(T(i-1) + T(n-i)\Big) + cn$$

To show:
$T(n) \leqslant dn\log_2 n$

$$= \frac{2}{n}\left(\sum_{i=1}^{n} T(i-1)\right) + cn$$

$$= \frac{2d}{n}\left(\sum_{i=1}^{n/2} T(i-1) + \sum_{i=\frac{n}{2}+1}^{n} T(i-1)\right) + cn$$

$$\leqslant \frac{2d}{n}\left(\sum_{i=1}^{n/2} (i-1)\underbrace{\log_2(n/2)}_{\log_2(n)-1} + \sum_{i=\frac{n}{2}+1}^{n} (i-1)\log_2 n\right) + cn$$

$$= \frac{2d}{n}\left(\sum_{i=1}^{n} (i-1)\log_2 n - \sum_{i=1}^{n/2} (i-1)\right) + cn$$

$$\underbrace{= \frac{2d}{n}\frac{(n-1)(n)}{2}\log_2(n) - \frac{2d}{n}\frac{n}{2}\frac{(n-2)}{4} + cn}_{\leqslant\; dn\log_2(n)}$$

for $d = 4c$.