Closest pair problem

Input: A list of $n \geq 2$ points $S \subseteq \mathbb{R}^2$.
Output: Points $(a,b), (c,d) \in S$ that are the closest.

- Trivial or brute-force approach?

  Go over all the possible pairs.
  $\#\text{steps} \geq \binom{n}{2} = O(n^2), \Omega(n^2), \Theta(n^2)$.
  (brush-up your asymptotics)
  [Big-Oh, Big-Omega, Theta]

- If $n = 1$ billion $= 10^9$ then n vs. n^2 time is a Huge difference (10^{18} nanosecs > 300 yrs is impossible!)

- Could you find a closest pair faster?

- Hint: Recurse.
  But, how do we divide the points S?

- Sort by the x-coordinate & collect the lower $n/2$ points in the set L.
The remaining $n/2$ points form R.

[Sorting takes $O(n\lg n)$ operations. Why?
Let us revise Merge Sort:
Given set $X = \{x_1, \ldots, x_n\}$ we recursively sort the first $n/2$ numbers to get L & the last $n/2$ to get R.
Now we want to merge $L = \{l_1 \leq \cdots \leq l_{n/2}\}$ & $R = \{r_1 \leq \cdots \leq r_{n/2}\}$:

i) Find the earliest position of l_1 in R & insert it.

ii) From that position onwards, find the position of l_2 & insert it in R.

iii) Continue till you reach $l_{n/2}$ or $r_{n/2}$.

▷ The comparisons done in the Merge Step are only $O(n)$.

▷ The recurrence for time is:

$$T(n) = 2 \cdot T(n/2) + O(n)$$

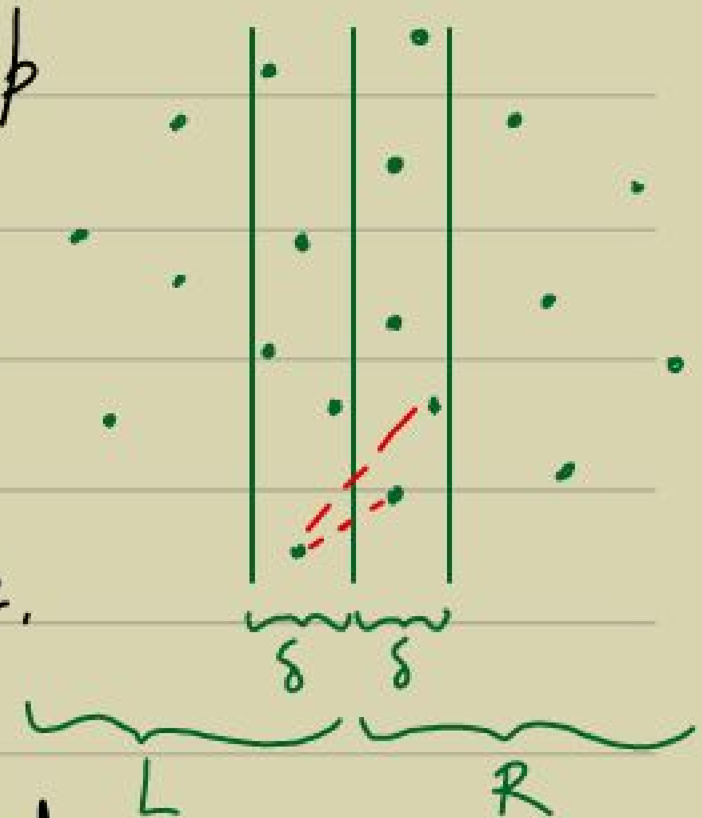
$$\Rightarrow T(n) = O(n\lg n). \quad \square$$
]

- Coming back to Closest Pair: We have left points L & right points R.

CP-dist(L) & CP-dist(R)

- Say, <u>recursively</u> we find closest pair distance δ_L in L & δ_R in R.
- Compute $\delta := \min(\delta_L, \delta_R)$.
- We can find the δ-strip on the left – S_L & the one on the right – S_R.

  How to <u>combine</u>?



distinct values →

- <u>Sort</u> S_R by y-coordinate.
- For each $p \in S_L$ {
  - let y_p be the y-coord.
  - Binary search for points $q \in S_R$ with y-coordinates $(y_p - \delta, y_p + \delta)$.

    [They all can be found in $O(\lg n)$ time.]
  - Compute the distance (p,q) & update δ if required. }
- Output δ.

▷ Time taken $T(n)$ is
- <u>Divide</u> step: $2T(n/2) + O(n \lg n)$
- <u>Combine</u> step: $O(n \lg n) + O(n)$.

$\Rightarrow T(n) = 2T(n/2) + O(n \lg n)$.

$\Rightarrow T(n) = O(n \lg^2 n) = \tilde{O}(n)$. ← soft-Oh

Theorem: There is an $O(n \lg^2 n)$ time algorithm to compute ClosestPair of n points in $\mathbb{R}^2$.

- Note: The time would also depend on the number of bits required to store a point.

- $O(n \lg^2 n)$ is an improvement over $O(n^2)$. Can we do better? Is this a lower bound?

- Suppose we want to reduce it to $O(n \lg n)$. Then, there are two places where we need to optimize:

x-sort S only once?

1) Don't sort S in Divide step.
2) " " S_R in Combine step.

- What are the alternatives?

Alternative Divide Step: (middle after sorting)

We find the x-median of S in $O(n)$ time.

- How can this be done without sorting?

Median of Medians Idea:

- Let Select(S, i) be the function that returns the element of rank i in S.

Generalization helps!

We will recursively define it!

Select(S, i) {

i) Divide the n elements S into $n/5$ groups each of size 5.

ii) Find the median in each group.

iii) Use Select( ) recursively to find the median x of these $n/5$ medians.

iv) Compute the rank k of x in S.

v) Divide S around x: $S_{<x}$ are elements $<x$ & $S_{>x}$ are those $>x$.

vi) If $i=k$ then return x.

Solve & Combine

If $i<k$ " " Select$(S_{<x}, i)$.

If $i>k$ " " Select$(S_{>x}, i-k)$.

- Proof of correctness requires you:
  - to check the base case,
  - to check the recursive calls, &
  - to check the returned value.

- Time complexity $T(n)$ has a recurrence,
  - First recursive call is on $n/5$ size.
  - Second " " " " $S_{<x}$ or $S_{>x}$

$\triangleright$ $\#S_{<x} \geq 3\cdot\left(\frac{1}{2}\cdot\frac{n}{5}-1\right)+2 = \frac{3n}{10}-1$

$\triangleright$ $\#S_{>x} \geq 3n/10 - 1$

$\triangleright$ $\#S_{<x} + \#S_{>x} = n-1$

- Thus, $\#S_{<x}, \#S_{>x} \leq 7n/10$.

$$\Rightarrow \quad T(n) \leq T(n/5) + T\left(\frac{7n}{10}\right) + O(n)$$

$$\Rightarrow \quad T(n) = T(n/5) + T(7n/10) + O(n)$$

Note: $1/5 + 7/10 = 0.9 < 1$

$$\Rightarrow \quad T(n) = O(n).$$

Lemma: Median is computable in $O(|S|)$ time.

Alternative Combine Step:

- We demand that CP-dist(L) & CP-dist(R) give us L & R each sorted by y-coordinate.

- Note that the Combine step of CP-dist(S) could merge the two & y-sort S as well.

$\Rightarrow$ S_R is already sorted by y-coordinate.

$\triangleright$ The new implementation of CP-dist(S) has the recurrence: $T(n) = 2T(n/2) + O(n)$.

[Shamos-Hoey 1975]

Theorem: Closest pair in the plane is computable in $O(n \lg n)$ time.

Qn: What about 1-D ?

Exercise: Write the full pseudocode for this algorithm. This will force you to deal with boundary cases & conditions.