Closest Pairs of Points O(nlogn)

One-dimensional version:

Input: n numbers $p_1, ..., p_n \in Z$ (e.g. 1, 3, -2, 0)

Output: find i, j, i≠j, such that $|p_i - p_j|$ is as small as possible (e.g. 0 and 1 are closest)

Naive algorithm: try all possible combinations i and j — $\Theta(n^2)$

Better algorithm: sort numbers in non-decreasing order and check adjacent numbers — $\Theta(nlogn) + \Theta(n) = \Theta(nlogn)$

Two-dimensional version (Euclidean distance $dist(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$):

For simplicity, assume no two points have the same x coordinate.

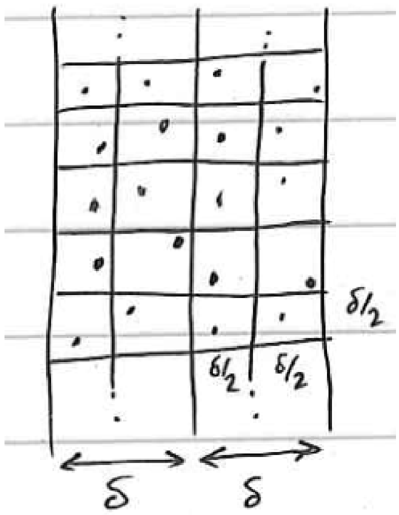
CLOSESTPAIR:

1. Divide points into two halves using a vertical line L (based on median of x coordinates)
2. Conquer: recursively compute CLOSESTPAIR on left half and right half. Let $\delta = min(dist)$
3. Combine
    a. Take points within distance δ of L
    b. Sort them by y coorinates. Call the sorted points $q_1, ..., q_n$.
    c. For each point q_i, calculate its distance to $q_{i+1}, ..., q_{i+11}$
4. Output the closest pair seen among the recursive calls and the cross pairs (Step 3)

Claim: If $|i - j| > 11$, then $dist(q_i, q_j) > \delta$.

Proof: Partition the slab into squares of side length $\frac{\delta}{2}$. Each square can contain at most one point, because the distance between any two points is δ.

Therefore, if $i < j - 11$, the y coordinate of q_i is less than that of q_j by more than δ.



Runtime:

Naively, $T(n) = 2T(\frac{n}{2}) + \theta(nlogn) = \theta(nlog^2 n)$ ($\theta(nlogn)$—sorting)

Actually, we can sort once for all

$T(n) = 2T(\frac{n}{2}) + \theta(n) = \theta(nlogn)$