Input : set of n points
Output : Finding closest distance between 2 points

Steps

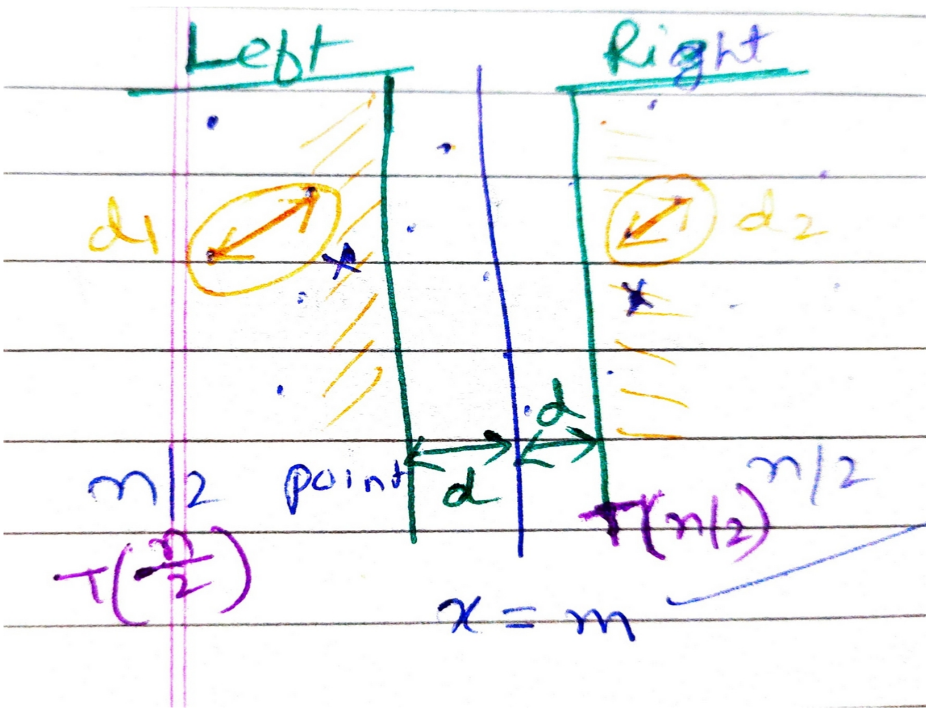
1) Get input. Sort the points according to X coordinate by MergeSort.

2) Now finding Median is easy.

3) Now if x coordinate of the point is less than mid point then it will go the left List. But if x coordinate is equal to mid point's x coordinate then check if point's y coordinate is less than mid point's y coordinate then add to Left List. Now if this Left list's population become bigger than median then add to Right list. So making 2 lists of points in 2 sides of median.

4) Finding smallest distance in left half (Dl) and right half (DR) recursively.

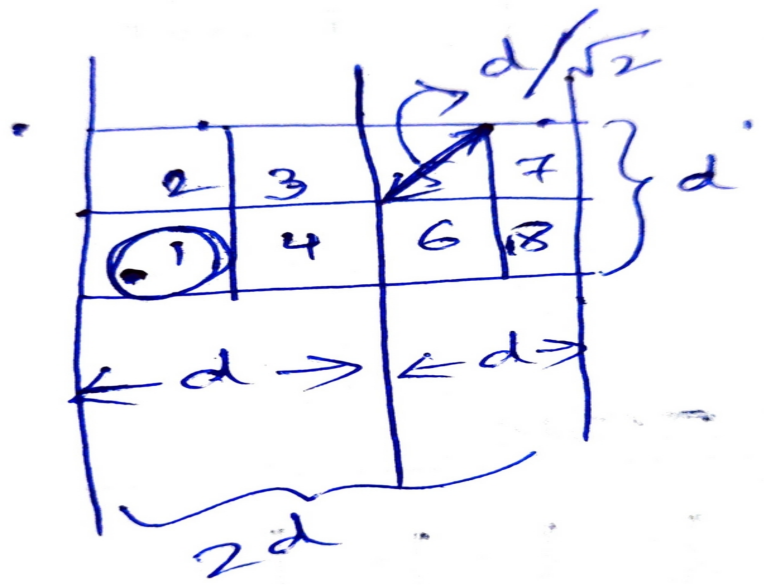
5) Suppose d = Get minimum distance in left half (Dl) and right half (DR).



6) We want to find 2 points whose distance is less than the closest pair points in left and right half. So build a strip of points in both side of median which less than d distance.

7) Find the minimum and closest distance in strip according to Y coordinate. If it differs more than d then we will proceed to next point. If distance between point i & j is less than smallest distance(d) then update. Now for each point we will only look next 7 points. Now as we have taken distance d in 2 sides of median so, the rectangle becomes d by 2d. Now if there is 2 points in different sides of median we can imagine dividing the rectangle vertically by d distance. So assume there can be 8 squares. In one of them resides our particular 1 point. So we are comparing distance between a particular point with its next 7 points, for each point. As they are sorted by y coordinate.

8) Now there are 8 squares where the diagonal in each square is d/square root (2). Now if there are 2 points in each of these boxes then distance between them is less than diagonal length. Now these squares can’t have more than 1 point in them distance less than d by the assumption of the definition of d. It means we will find a pair of points whose distance is less than d if exists.



Time Complexity

1) Get those points sorted by X coordinate by MergeSort → O(nlogn)
2) Find median. → O(1)
3) Break the list of points in 2 halves according to X coordinate. → O(n)
4) Solve the sub problems in 2 regions recursively → 2T(n/2)
5) Sort the points according to Y coordinates → O(nlogn)
6) For each point calculate distance with next 7 points. → O(n)

Now if we assume that points are already sorted by X coordinate simultaneously. And when dividing into sub problems then these points are sorted according to Y coordinate simultaneously. Now we just have to merge these 2 sorted lists and merging can be done in O(n). Then this time complexity can be measured in 2T(n/2) + O(n) = O(nlogn).