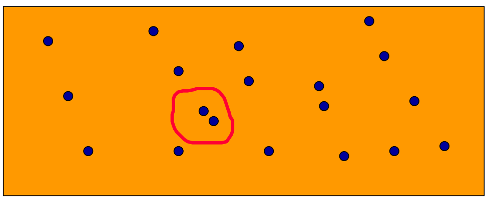
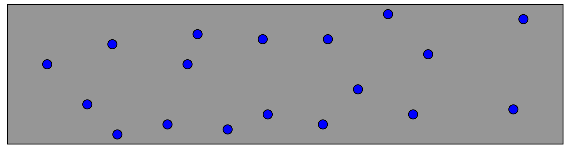
Closest Pair Of Points

- Given n points in 2D, find the pair that are closest.



Applications



- We plan to drill holes in a metal sheet.
- If the holes are too close, the sheet will tear during drilling.
- Verify that no two holes are closer than a threshold distance (e.g., holes are at least 1 inch apart).

Air Traffic Control

- 3D -- Locations of airplanes flying in the neighborhood of a busy airport are known.
- Want to be sure that no two planes get closer than a given threshold distance.

Simple Solution

- For each of the $n(n-1)/2$ pairs of points, determine the distance between the points in the pair.
- Determine the pair with the minimum distance.
- $O(n^2)$ time.

Divide-And-Conquer Solution

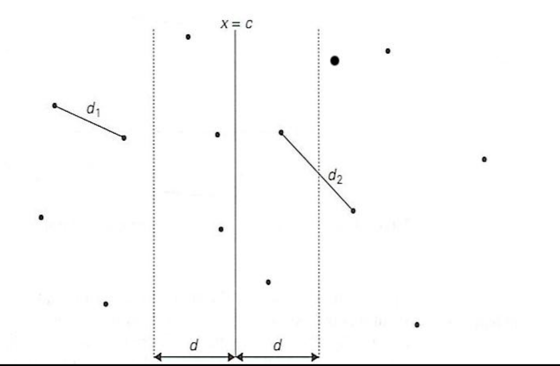
- When n is small, use simple solution.
- When n is large
  - Divide the point set into two roughly equal parts A and B.
  - Determine the closest pair of points in A.
  - Determine the closest pair of points in B.
  - Determine the closest pair of points such that one point is in A and the other in B.
  - From the three closest pairs computed, select the one with least distance.

Closest-Pair Problem by Divide-and-Conquer

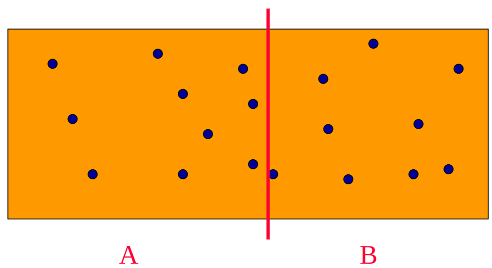
Step 1 Sort the points by x (list one) and then by y (list two).

Step 2 Divide the points given into two subsets A and B by a vertical line $x = c$ so that half the points lie to the left or on the line and half the points lie to the right or on the line.

Step 3 Find recursively the closest pairs for the left and right subsets.

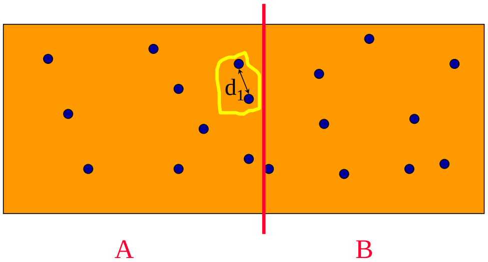


Example



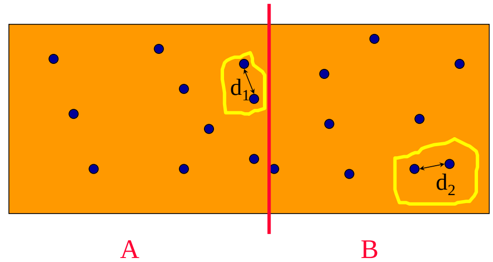
- Divide so that points in A have x-coordinate <= that of points in B.

Example



- Find closest pair in A.
- Let d_1 be the distance between the points in this pair.

Example



- Find closest pair in B.
- Let d_2 be the distance between the points in this pair.

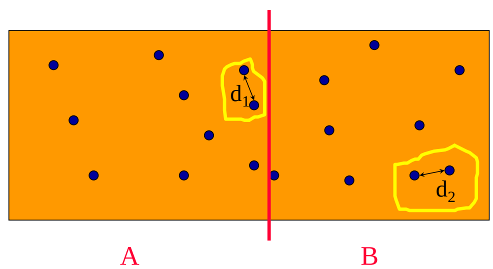
Closest Pair by Divide-and-Conquer (cont.)

Step 4 Set $d = \min\{d_1, d_2\}$

We can limit our attention to the points in the symmetric vertical strip of width 2*d* as possible closest pair. Let R_A and R_B be the subsets of points in the left subset A and of the right subset B, respectively, that lie in this vertical strip. The points in R_A and R_B are stored in increasing order of their *y* coordinates, taken from the second list.

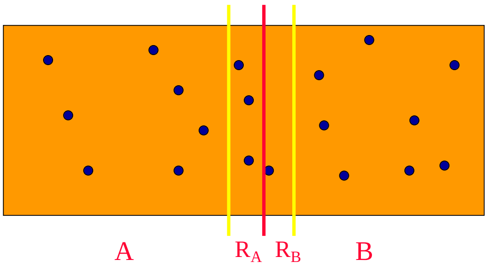
Step 5 For every point $P(x,y)$ in R_A, we inspect points in R_B that may be closer to *P* than *d*.

Example



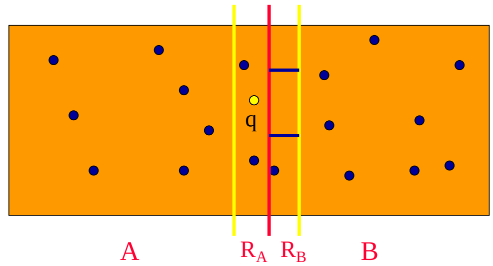
- Let $d = \min\{d_1, d_2\}$.
- Is there a pair with one point in A, the other in B and distance $< d$?

Example



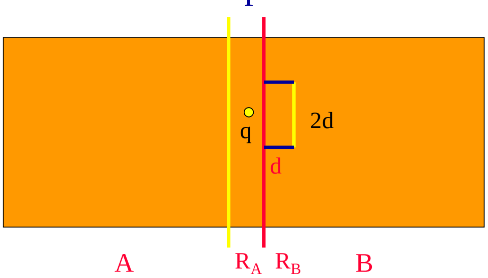
- Candidates lie within d of the dividing line.
- Call these regions R_A and R_B, respectively.

Example



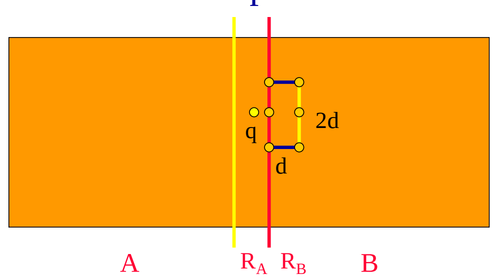
- Let q be a point in R_A.
- q need be paired only with those points in R_B that are within d of q.y.

Example



- Points that are to be paired with q are in a d x 2d rectangle of R_B (comparing region of q).
- Points in this rectangle are at least d apart.

Example



- So the comparing region of q has at most 6 points.
- So number of pairs to check is <= 6| R_A |

Time Complexity

- Create a sorted by x-coordinate list of points.
  - O(n log n) time.
- Create a sorted by y-coordinate list of points.
  - O(n log n) time.
- When you add in the initial sort time for the sort by x and sort by y lists, the overall time remains O(n log n).
- Using these two lists, the required pairs of points from R_A and R_B can be constructed in O(n) time.
- Let n < 4 define a small instance.

Time Complexity

- Let T(n) be the time to find the closest pair (excluding the time to create the two sorted lists). //without sorting
- t(n) = c, n < 4, where c is a constant.
- When n >= 4,

  T(n) = T(ceil(n/2)) + T(floor(n/2)) + an,

  where a is a constant.

  T(n) = 2T(n/2)) + an
- To solve the recurrence, assume n is a power of 2 and use repeated substitution.
- t(n) = O(n log n).