Closest pair problem

Given: N points in the plane
Goal: Find a pair with smallest Euclidean distance between them.

Fundamental geometric primitive.

- Graphics, computer vision, geographic information systems, molecular modeling, air traffic control.
- Special case of nearest neighbor, Euclidean MST, Voronoi.

fast closest pair inspired fast algorithms for these problems

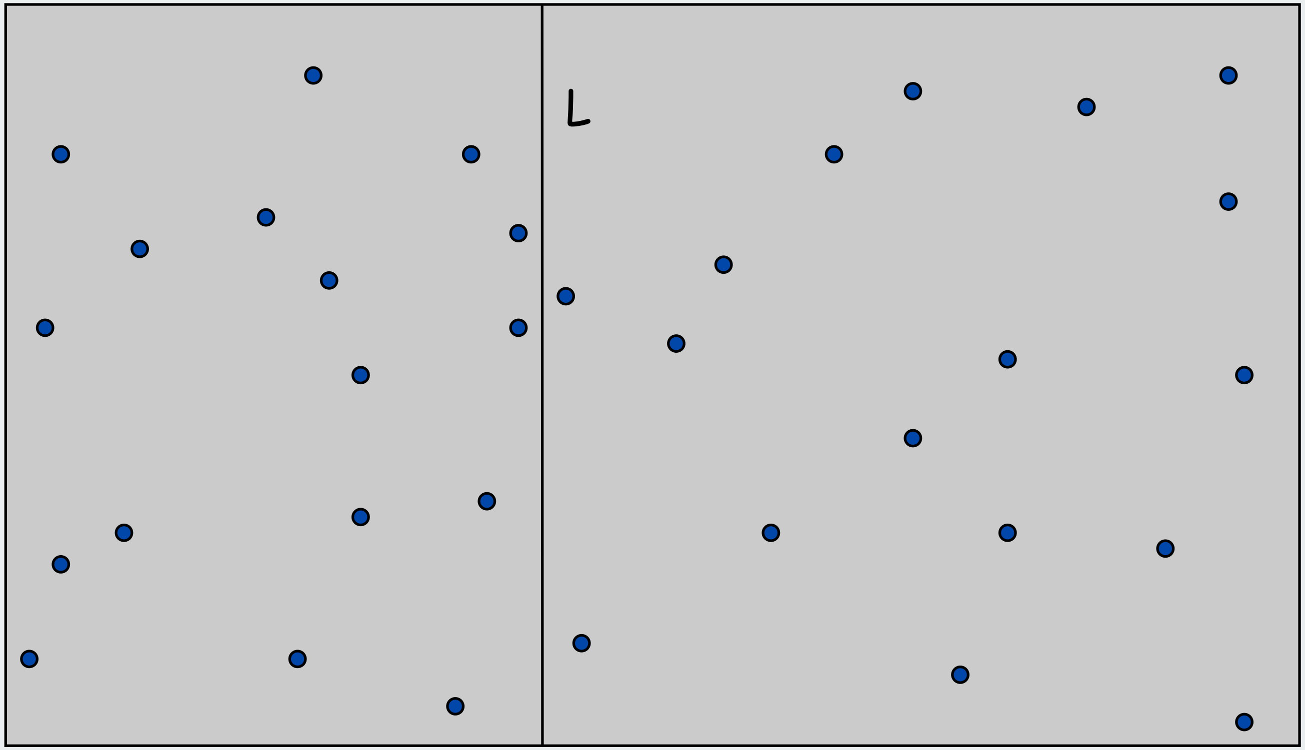
Brute force.
Check all pairs of points p and q with $\Theta(N^2)$ distance calculations.

1-D version. O(N log N) easy if points are on a line.

Degeneracies complicate solutions.
as usual for geometric algs
[assumption for lecture: no two points have same x coordinate]

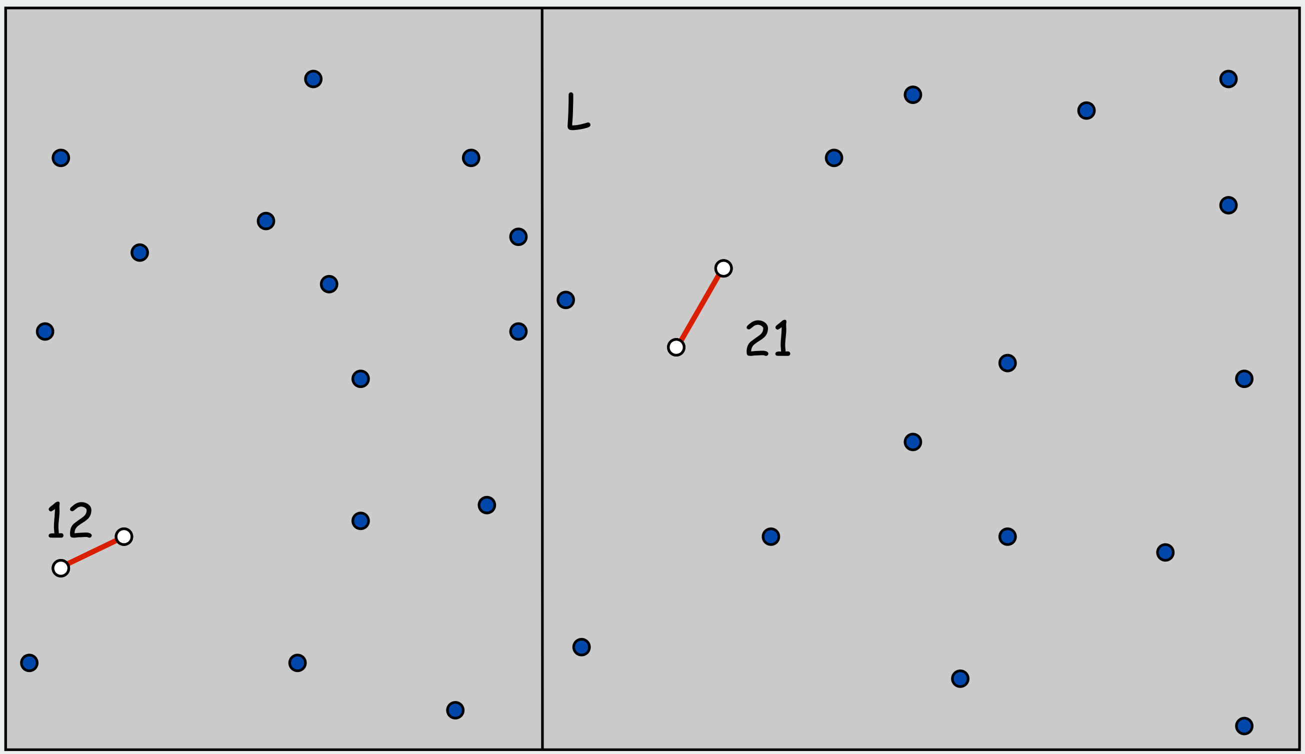
Algorithm.

- Divide: draw vertical line L so that roughly $\frac{1}{2}N$ points on each side.



Algorithm.

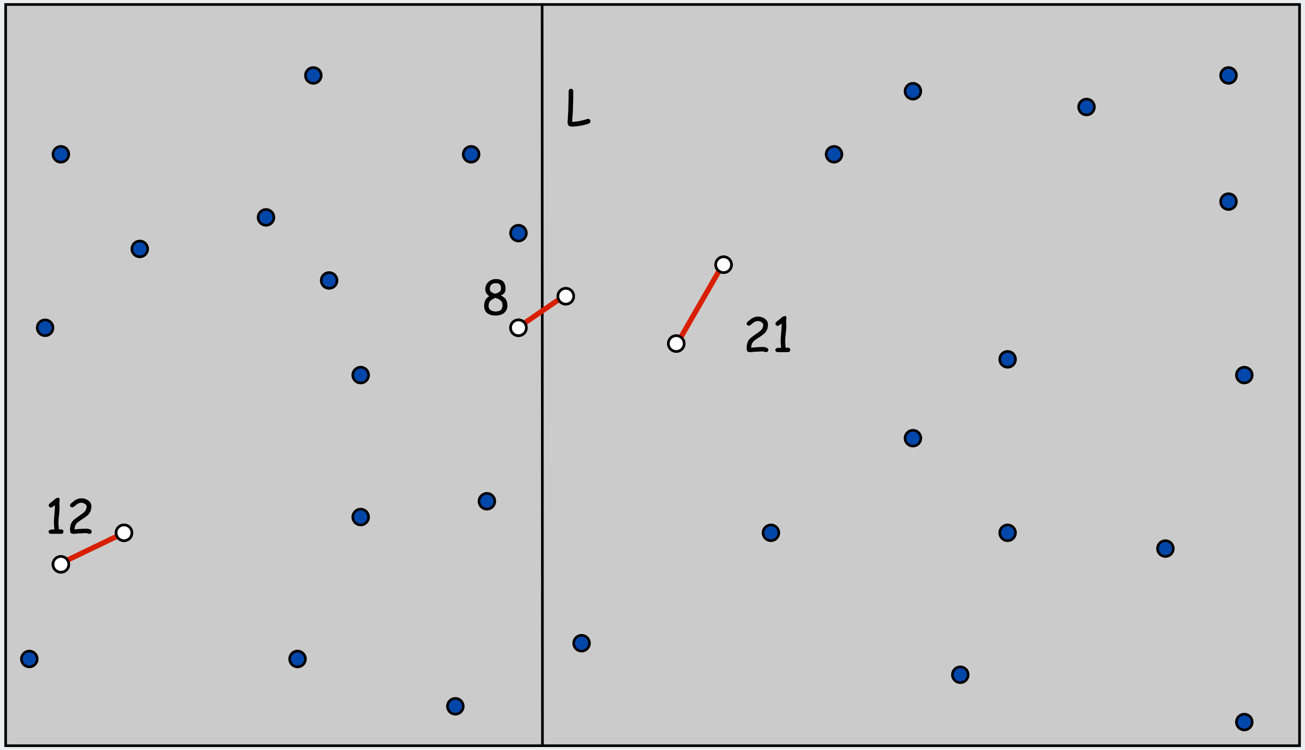
- Divide: draw vertical line L so that roughly $\frac{1}{2}N$ points on each side.
- Conquer: find closest pair in each side recursively.



Algorithm.

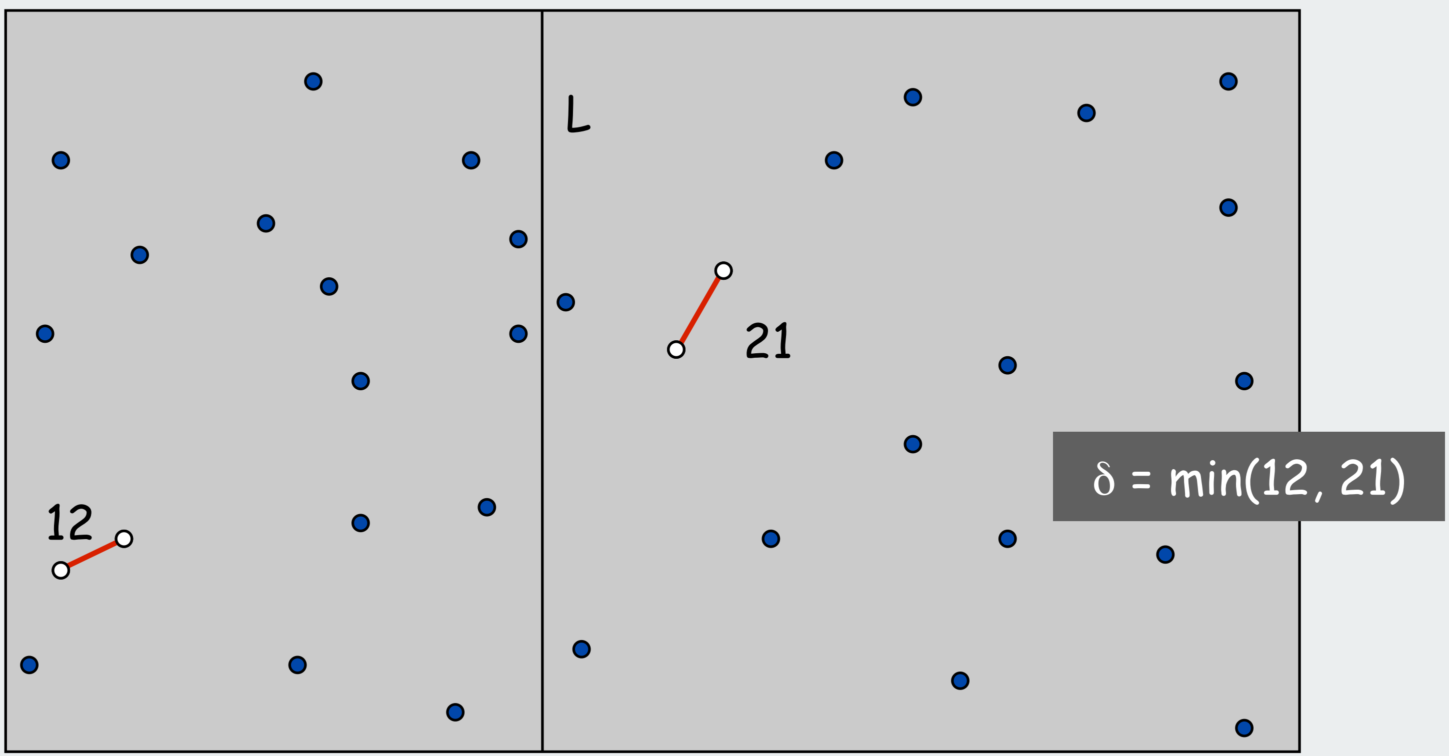
- Divide: draw vertical line L so that roughly $\frac{1}{2}N$ points on each side.
- Conquer: find closest pair in each side recursively.
- Combine: find closest pair with one point in each side.
- Return best of 3 solutions.

seems like $\Theta(N^2)$



Closest Pair of Points

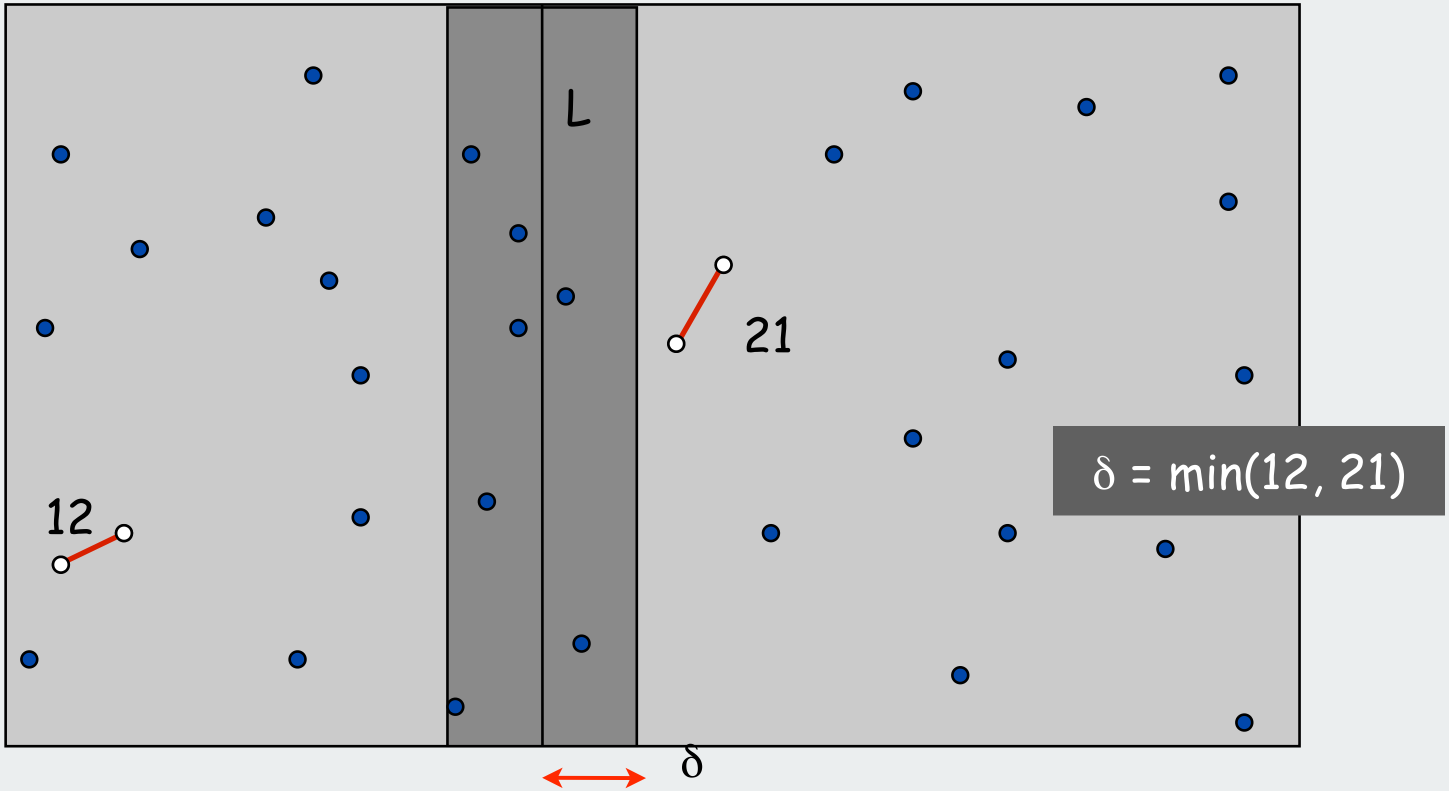
Find closest pair with one point in each side, assuming that distance < δ.



Closest Pair of Points

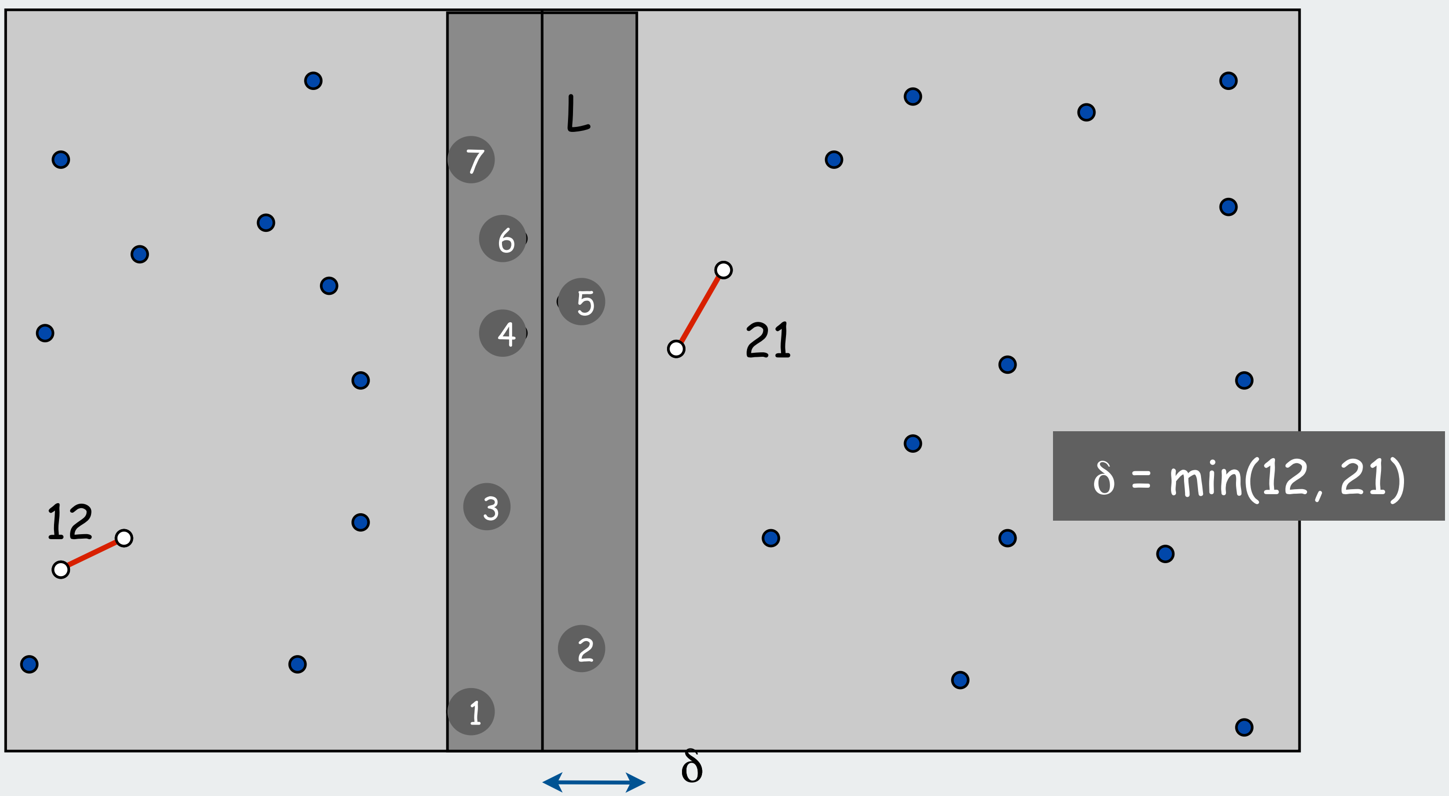
Find closest pair with one point in each side, assuming that distance < δ.

- Observation: only need to consider points within δ of line L.



Find closest pair with one point in each side, assuming that distance < δ.

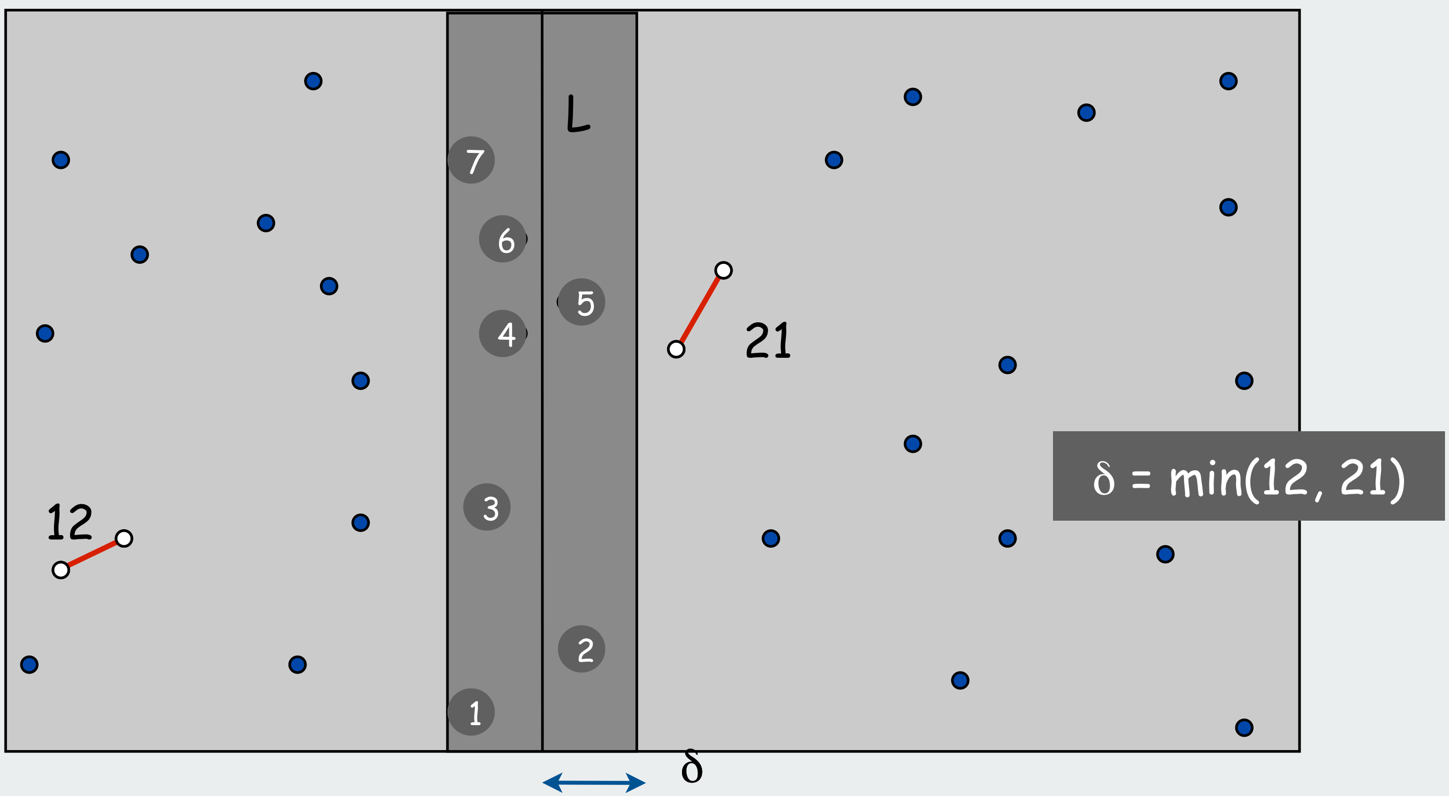
- Observation: only need to consider points within δ of line L.
- Sort points in 2δ-strip by their y coordinate.



Closest Pair of Points

Find closest pair with one point in each side, assuming that distance < δ.

- Observation: only need to consider points within δ of line L.
- Sort points in 2δ-strip by their y coordinate.
- Only check distances of those within 11 positions in sorted list!



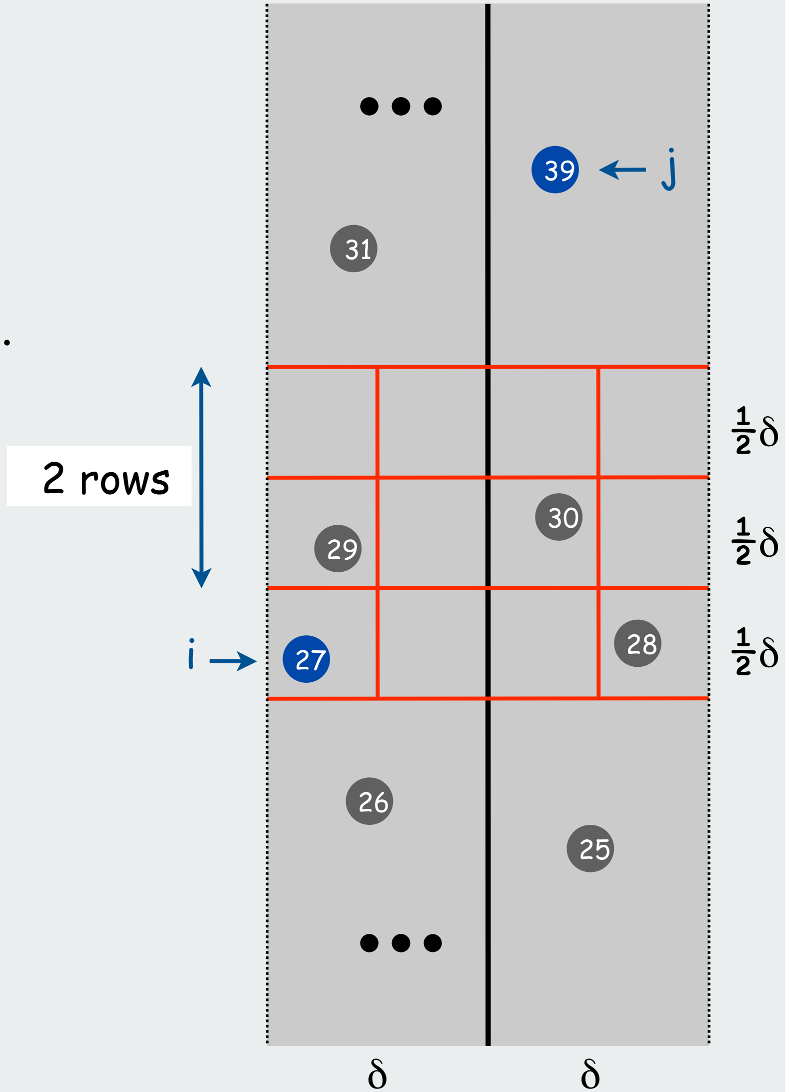
Closest Pair of Points

Def. Let s_i be the point in the 2δ-strip, with the i^{th} smallest y-coordinate.

Claim. If $|i - j| \geq 12$, then the distance between s_i and s_j is at least δ.

Pf.

- No two points lie in same $\frac{1}{2}\delta$-by-$\frac{1}{2}\delta$ box.
- Two points at least 2 rows apart have distance $\geq 2(\frac{1}{2}\delta)$. ▪



Fact. Still true if we replace 12 with 7.

Closest Pair Algorithm

```
Closest-Pair(p₁, …, pₙ)
{
   Compute separation line L such that half the points
   are on one side and half on the other side.

   δ₁ = Closest-Pair(left half)
   δ₂ = Closest-Pair(right half)
   δ  = min(δ₁, δ₂)

   Delete all points further than δ from separation line L

   Sort remaining points by y-coordinate.

   Scan points in y-order and compare distance between
   each point and next 11 neighbors. If any of these
   distances is less than δ, update δ.

   return δ.
}
```

Step	Cost
Compute separation line	$O(N \log N)$
Recursive calls	$2T(N / 2)$
Delete	$O(N)$
Sort	$O(N \log N)$
Scan	$O(N)$

Closest Pair of Points: *Analysis*

Algorithm gives upper bound on running time

Recurrence

$$T(N) \leq 2T(N/2) + O(N \log N)$$

Solution

$$T(N) = O(N (\log N)^2)$$

avoid sorting by y-coordinate from scratch

Upper bound. Can be improved to $O(N \log N)$.

Lower bound. In quadratic decision tree model, any algorithm for closest pair requires $\Omega(N \log N)$ steps.