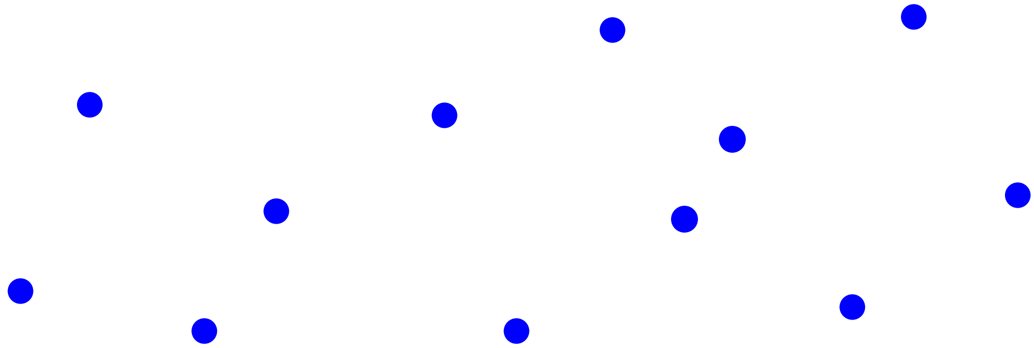
Closest Pair

One-Shot Problem

Given a set P of N points, find p,q $\in$ P, such that the distance d(p, q) is minimum.

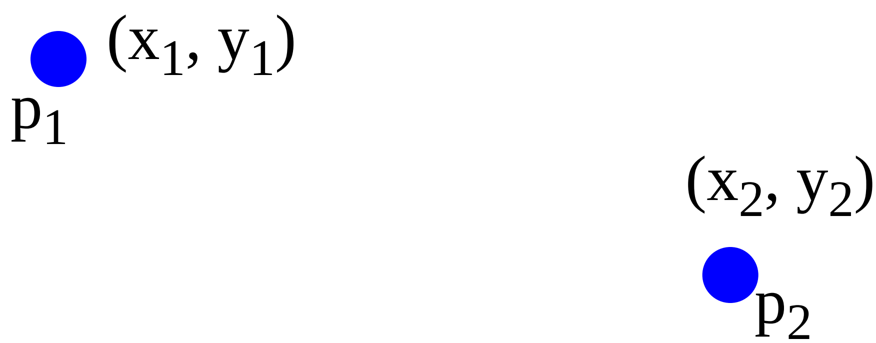


Algorithms for determining the closest pair:

1. Brute Force O(N^2)
2. Divide and Conquer O(N log N)
3. Sweep-Line O(N log N)

Brute Force Algorithm

Compute all the distances $d(p,q)$ and select the minimum distance.

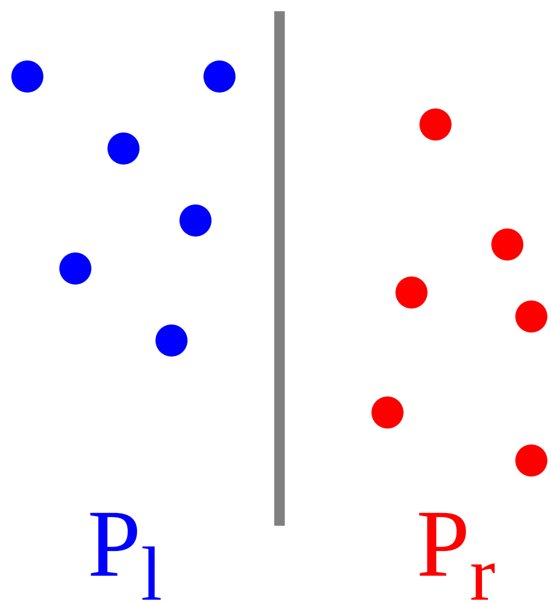


$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Time Complexity: $O(N^2)$

Divide and Conquer Algorithm

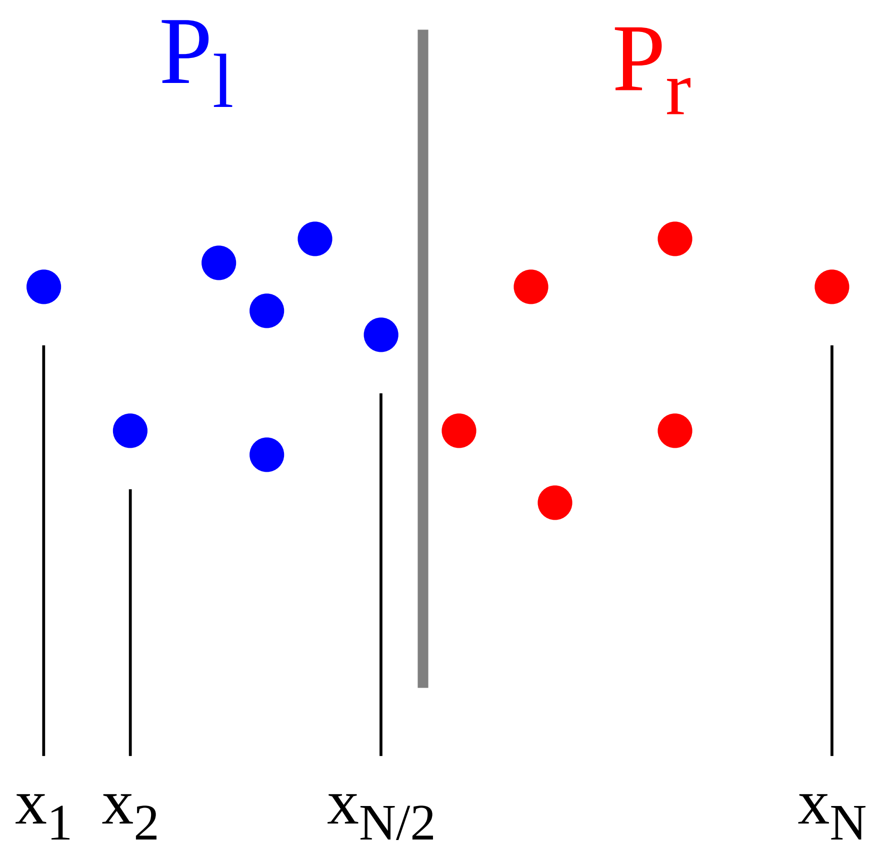
Idea: A better method! Sort points on the x-coordinate and divide them in half. Closest pair is either in one of the halves or has a member in each half.



Divide and Conquer Algorithm

Phase 1: Sort the points by their x-coordinate:

$p_1\ p_2 \cdots p_{N/2} \cdots p_{N/2+1} \cdots p_N$



Divide and Conquer Algorithm

Phase 2:

Recursively compute closest pairs and minimum distances, d_l, d_r in

$$P_l = \{ P_1, p_2, \ldots , P_{N/2} \}$$
$$P_r = \{ P_{N/2+1}, \ldots , P_N \}$$

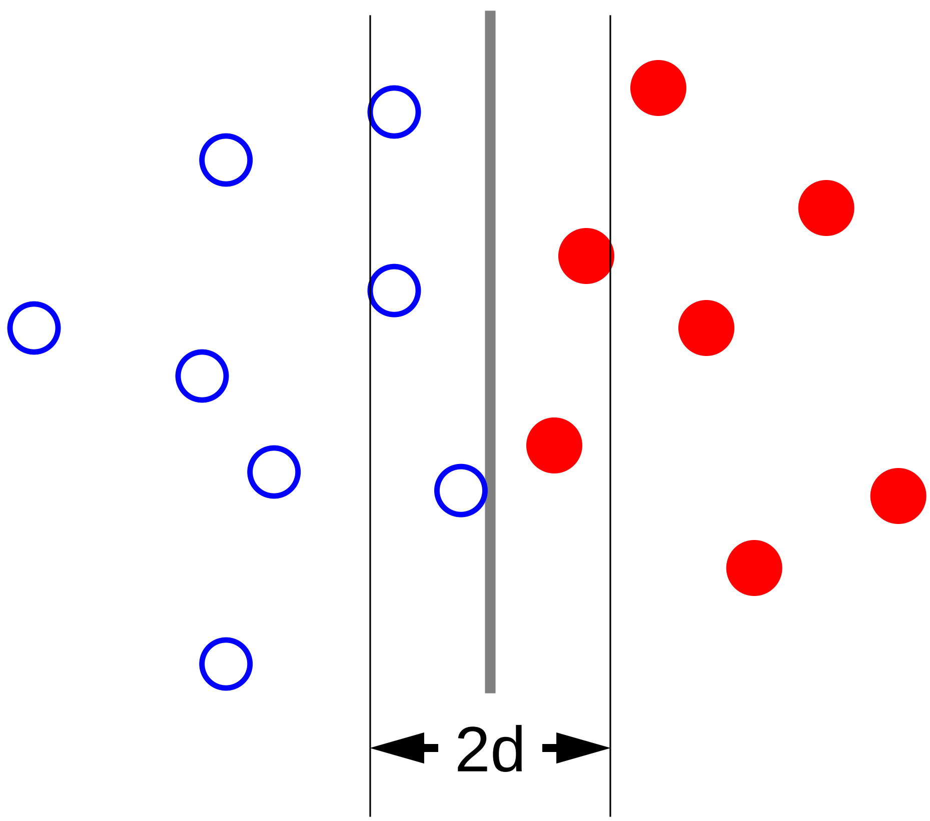
Find the closest pair and closest distance in central strip of width 2d, where

$$d = \min(d_l, d_r)$$

in other words...

Divide and Conquer Subproblem

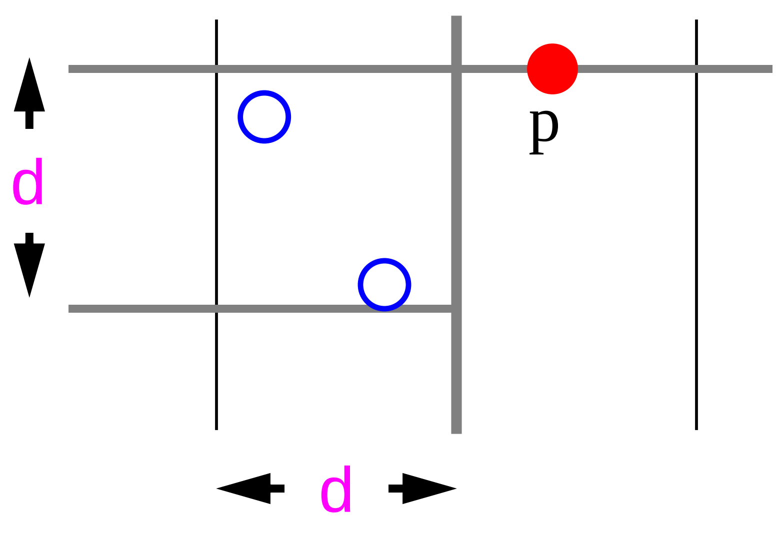
- Find the closest (○, ●) pair in a strip of width 2d, knowing that no (○,○) or (●,●) pair is closer than d.



Subproblem Solution

- For each point p in the strip, check distances d(p, q), where p and q are of different colors and:

$$y(p) - d \le y(q) \le y(p)$$



- There are no more than four such points!

Time Complexity

If we sort by y-coord each time:

$$\boxed{\begin{array}{l} T(N) = 2\,T(N/2) + N \log N \\ T(1) = 1 \end{array}}$$

$$\begin{aligned} T(N) &= 2\,T(N/2) + N \log N && (1)\\ &= 4\,T(N/4) + 2\,(N/2) \log (N/2) + N \log N \\ &= 4\,T(N/4) + N\,(\log N - 1) + N \log N && (2)\\ &\quad \ldots \\ &= 2^K T(N/2^K) + \\ &\quad N(\log N + (\log N - 1) + \ldots + (\log N - K + 1)) && (K)\\ &\quad \ldots \end{aligned}$$

stop when $N/2^K = 1 \longrightarrow K = \log N$

$$\begin{aligned} &= N + N\,(1 + 2 + 3 + \ldots + \log N) && (\log N)\\ &= N + N\,((\log N + 1) \log N) / 2 \end{aligned}$$

$$= \mathbf{O(\,N \log^2 N\,)}$$

Improved Algorithm

Idea:

- Sort all the points by y-coordinate once
- Before recursive calls, partition the sorted list into two sorted sublists for the left and right halves
- After computation of closest pair, merge back sorted sublists

Time Complexity of Improved Algorithm

Phase 1:

Sort by x and y coordinate: O(N log N)

Phase 2:

Partition:	O(N)
Recur:	2 T(N/2)
Subproblem:	O(N)
Merge:	O(N)

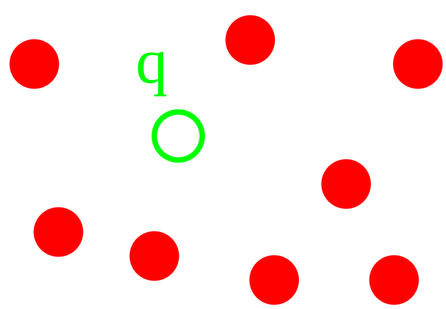
$$T(N) = 2\,T(N/2) + N = O(N \log N)$$

Total Time: O(N log N)

Closest Points

Repetitive Mode Problem

- Given a set S of sites, answer queries as to what is the closest site to point q.



I.e. which post office is closest?

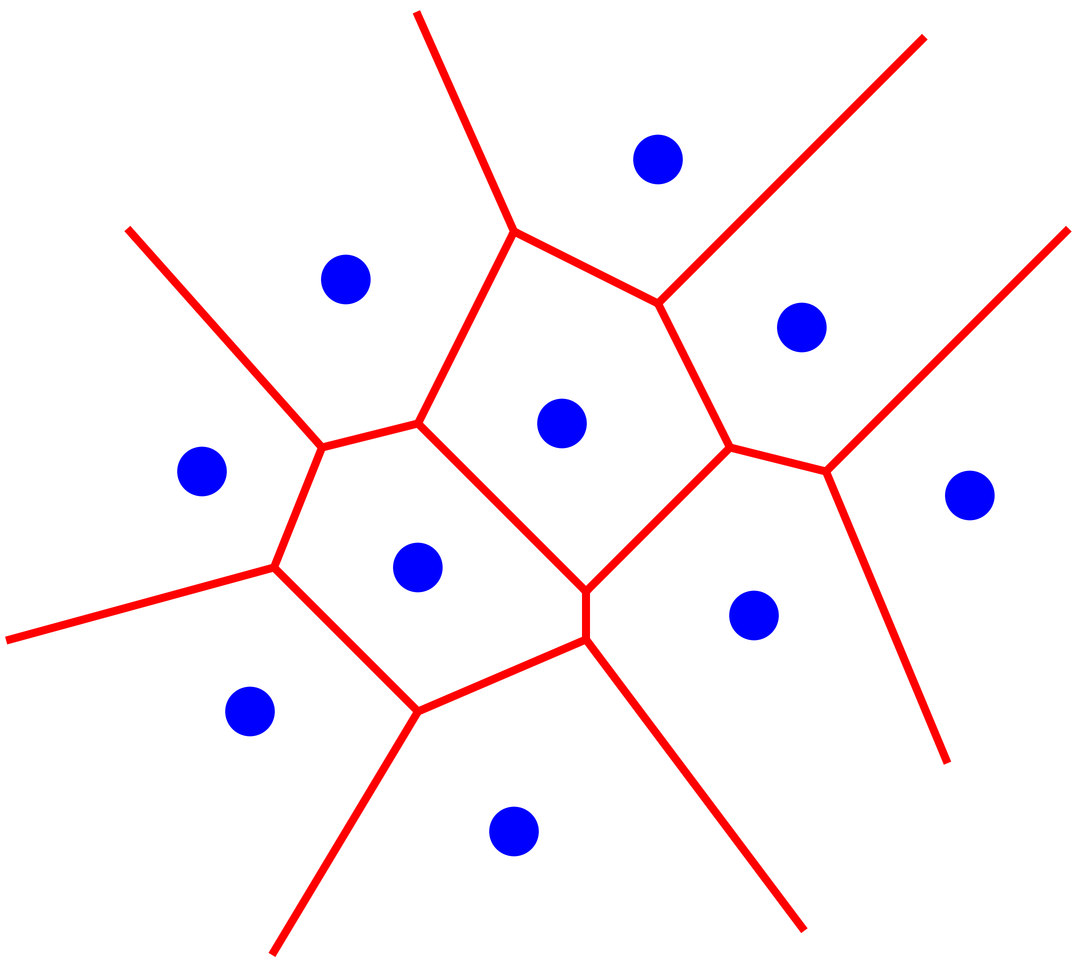
Voronoi Diagram

$S = \{ s_1, s_2, \ldots , s_N \}$
Set of all points in the plane called ***sites***.

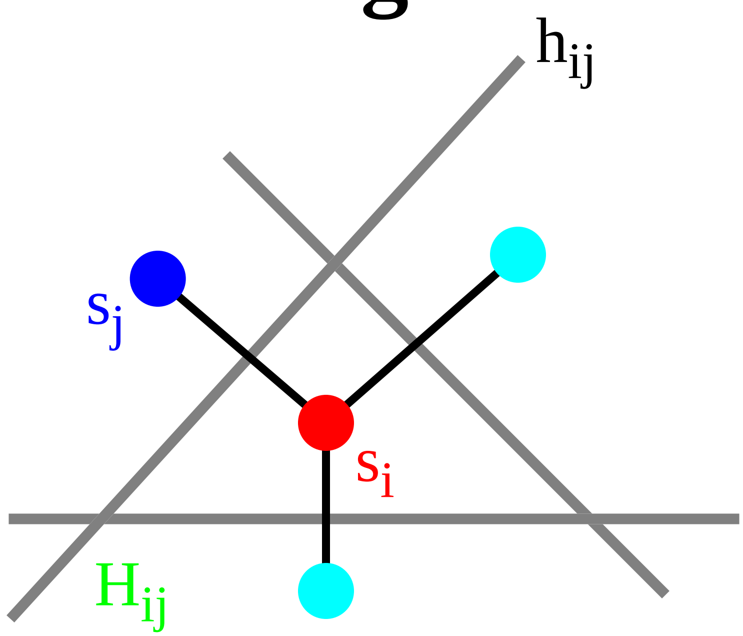
Voronoi region of s_i:
$V(s_i)=$
$\{ p: d(p, s_i) \leq d(p, s_j), \forall\, j \neq i \}$

Voronoi diagram of S:
Vor(S)= partition of plane into the regions $V(s_i)$

Voronoi Diagram Example



Constructing a Voronoi Diagram



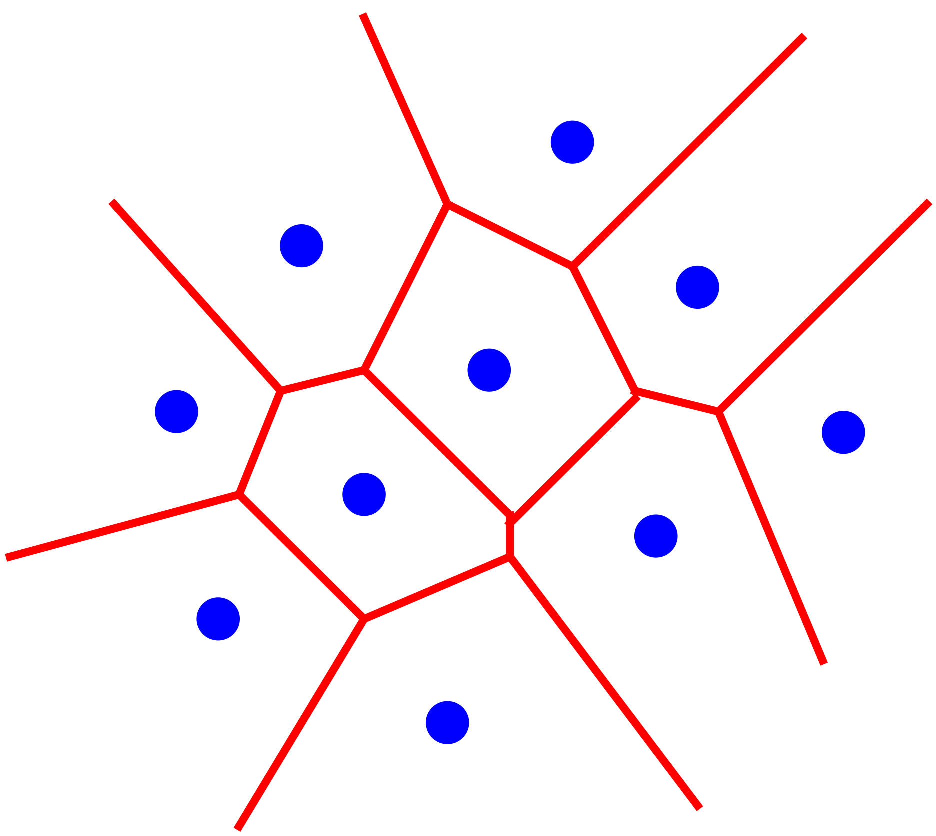
h_{ij}: ***perpendicular bisector*** of segment (s_i, s_j)

H_{ij}: half-plane delimited by h_{ij} and containing s_i

H_{ij}= { p: p is closer to s_i than s_j}

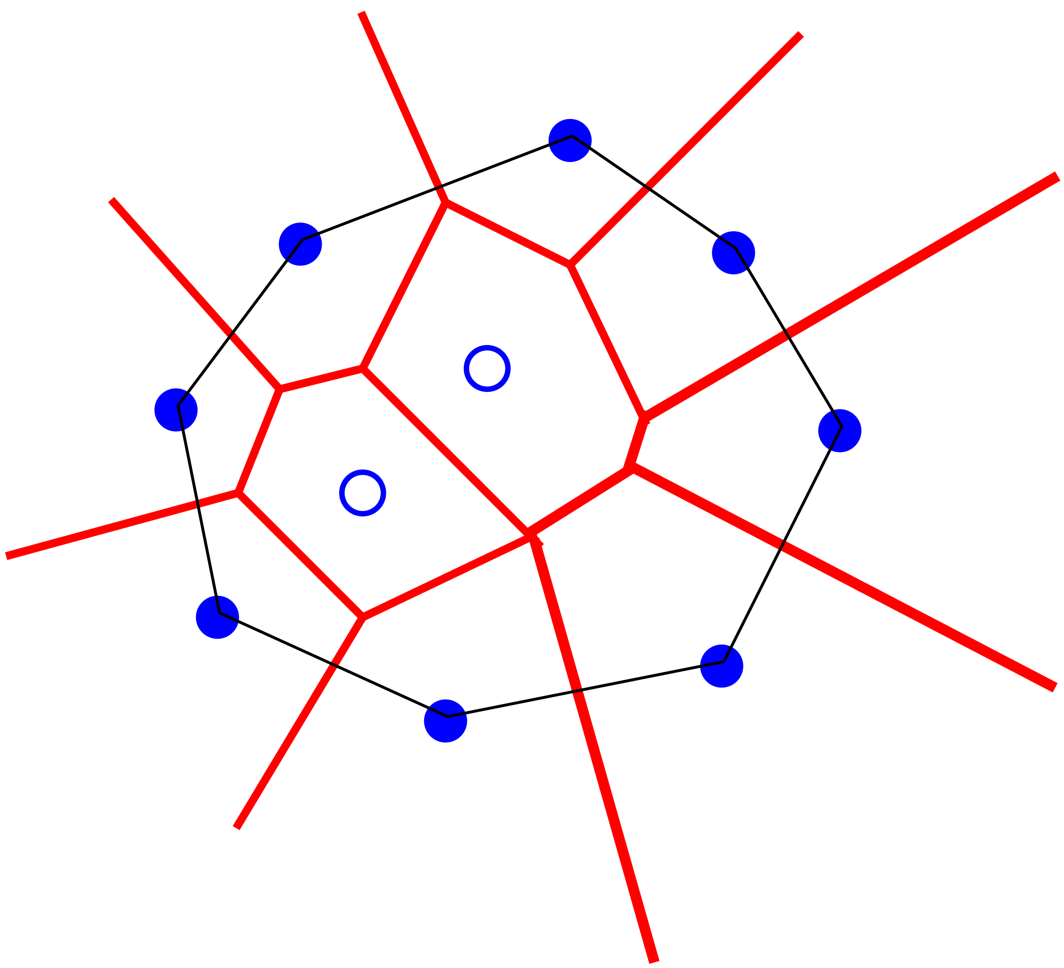
Constructing a Voronoi Diagram

$$V(S_i) = \bigcap_{\substack{j \geq 1 \\ j \neq i}}^{N} H_{ij} \quad \ldots \quad \text{Convex!}$$



Voronoi Diagram and Convex Hull

Sites in unbounded regions of the Voronoi Diagram are exactly those on the convex hull!



Constructing Voronoi Diagrams

There is a divide and conquer algorithm for constructing Voronoi diagrams with O(N log N) time complexity

It's too difficult for CS 16, but don't give up.

Your natural desire to learn more on algorithms and geometry can be fulfilled.

Geometry is Big Fun!

Want to know more about geometric algorithms and explore 3rd, 4th, and higher dimensions?

Take CS 252: Computational Geometry

(offered in Sem. II, 1998)