Day 12

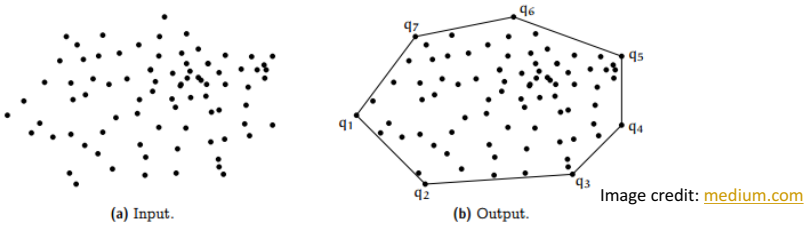
QUICKHULL

Analysis of:

ALGORITHM *EfficientClosestPair*(P, Q)

```
//Solves the closest-pair problem by divide-and-conquer
//Input: An array P of n ≥ 2 points in the Cartesian plane sorted in
//       nondecreasing order of their x coordinates and an array Q of the
//       same points sorted in nondecreasing order of the y coordinates
//Output: Euclidean distance between the closest pair of points
if n ≤ 3
    return the minimal distance found by the brute-force algorithm
else
    copy the first ⌈n/2⌉ points of P to array P_l
    copy the same ⌈n/2⌉ points from Q to array Q_l
    copy the remaining ⌊n/2⌋ points of P to array P_r
    copy the same ⌊n/2⌋ points from Q to array Q_r
    d_l ← EfficientClosestPair(P_l, Q_l)
    d_r ← EfficientClosestPair(P_r, Q_r)
    d ← min{d_l, d_r}
    m ← P[⌈n/2⌉ − 1].x
    copy all the points of Q for which |x − m| < d into array S[0..num − 1]
    dminsq ← d^2
    for i ← 0 to num − 2 do
        k ← i + 1
        while k ≤ num − 1 and (S[k].y − S[i].y)^2 < dminsq
            dminsq ← min((S[k].x − S[i].x)^2 + (S[k].y − S[i].y)^2, dminsq)
            k ← k + 1
return sqrt(dminsq)
```

Convex Hull Problem



(a) Input. (b) Output.

Image credit: medium.com

- Input: points $P_i = (x_i, y_i)$
- Solution: list of points making up convex hull, in (say) CCW order

Convex Hull Problem: Rubber-band edition

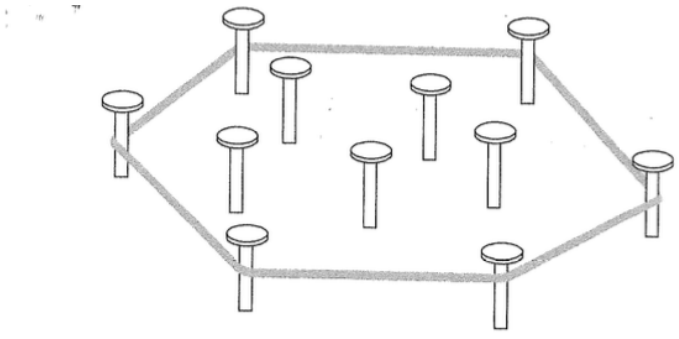
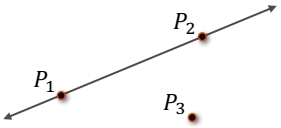


FIGURE 3.5 Rubber-band interpretation of the convex hull.

Geometric Calculation

The main geometric computation we'll need to make: on what side of a line P_1P_2 does a point $P_3 = (x_3, y_3)$ lie?



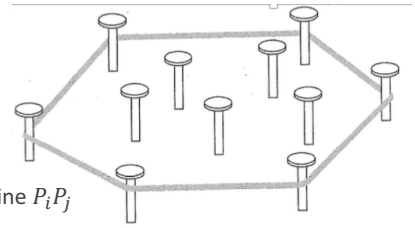
Equation of line P_1P_2 :

$$(x_2 - x_1)(y - y_1) - (y_2 - y_1)(x - x_1) = 0$$

Plug in (x_3, y_3) to LHS and see whether the result is positive (P_3 is on one side), negative (P_3 is on other side), or 0 (P_3 is on the line).

Brute Force Solution

A pair P_i, P_j is adjacent in the convex hull, if all other points are on one side of P_iP_j.



Algorithm sketch:

- Loop over all pairs (P_i, P_j) $\Theta(N^2)$
  - Loop over all other points P_k $\Theta(N)$
    - Run point-line relation on P_k versus line P_iP_j $\Theta(1)$
  - If all other points on the same side, P_i, P_j are adjacent in the convex hull.
- At the end, stitch together the convex hull list by using the set of adjacent pairs found.

Runtime: $\Theta(N^3)$

Quickhull

1) Sort points by x-coordinates

2) Draw a line between the leftmost points P_1 and the rightmost point P_n

- Notice that P_1 and P_n are part of the convex hull solution
- The line separates the points into two sets:
  - S_1: those above/left of the line and
  - S_2: those below/right of the line

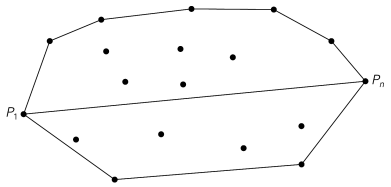


FIGURE 4.8 Upper and lower hulls of a set of points

Quickhull

3) We define the *upper hull* as consisting of line segments with vertices at P_1, some of the points in S_1, and P_n

4) We define the *lower hull* as consisting of line segments with vertices at P_1, some of the points in S_2, and P_n

Notice: The upper and lower hull can be constructed independently

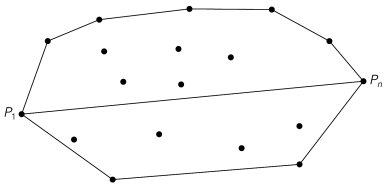


FIGURE 4.8 Upper and lower hulls of a set of points

Quickhull

5) If S_1 is empty, the upper hull is the line segment P_1 to P_n

6) If S_1 is not empty then identify P_{max} as the point that is at max distance from line P_1P_n

- Notice that P_{max} is on convex hull.
- Points inside triangle $P_{max}P_1P_n$ cannot be on convex hull.

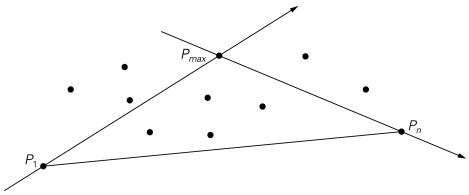


FIGURE 4.9 The idea of quickhull

Upper Hull

Break into two subproblems:

- left of P_1P_{max}
- left of $P_{max}P_n$

Notice the direction of the arrows.

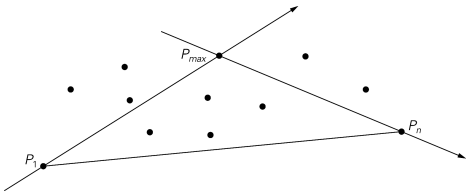


FIGURE 4.9 The idea of quickhull