Prepared By

Md. Shahrukh Islam

Student ID: 1805098

Offline 8

CSE 204

2-D Second Closest Pair

First we approach to get closest pair

• We partition S[Array of points] into S1, S2 by vertical line defined by median x-coordinate in S.

• Then we recursively compute closest pair distances d1 and d2. Set d = min(d1, d2).

• Now we compute the closest pair with one point each in S1 and S2.

In each candidate pair (p, q), where p ∈ S1 and q ∈ S2, the points p, q must both lie within d of median.

• At this point, complications arise, which weren't present in 1D. It's entirely possible that all n/2 points of S1 (and S2) lie within δ of `.

• We show that points in P1, P2 (d strip around median) have a special structure, and solve the conquer step faster.

Conquer Step

• Consider a point p ∈ S1. All points of S2 within distance d of p must lie in a d × 2d rectangle R.

• How many points can be inside R if each pair is at least δ apart?

• In 2D, this number is at most 7

• So, we only need to perform 7 × n/2 distance comparisons!

• We don't have an O(n log n) time algorithm yet. Why?

Conquer Step Pairs

• In order to determine at most 7 potential mates of p, project p and all points of P2 onto line `.

• Pick out points whose projection is within d of p; at most seven.

• We can do this for all p, by walking sorted lists of P1 and P2, in total O(n) time.

• The sorted lists for P1, P2 can be obtained from pre-sorting of S1, S2.

- Final recurrence is $T(n) = 2T(n/2) + O(n)$, which solves to $T(n) = O(n \log n)$.

One function call gives us the minimum distance as well as the points.

Let point1 and point2 are points of nearest.

Then solve closest points again without point1 and suppose it’s d1.

Again solve closest points without point2 and suppose it’s d2.

Our answer is min(d1, d2) and we can also get the points as well.

So total $3T(n) = 3(2T(n/2) + O(n)) = O(3n\log n) = O(n\log n)$