Closest pair of points
Announcements

Moodle submission
(oops on last homework)

TA posted guidelines
Closest pair of points

Today we will find the closest pair of points.

Ensures planes do not crash!
Closest pair of points

We will use a divide-and-conquer algorithm to do this

Idea: Split the points in half (vertically) then find...
1. Closest pair of points in left half
2. Closest pair of points in right half
3. Closest pair of points between halves
Closest pair of points

1. and 2. found using recursion

Base case: 3 or fewer points
If so, brute force check (3 checks)

To solve 3. (between splits), we use some tricky geometry...
Closest pair of points

Let $\partial_p$ be the shortest distance for a set of points $P$

If $L$ and $R$ are the left and right halves respectively, let $\partial = \min(\partial_L, \partial_R)$

If there is a closer pair, they must be within $\partial$ of split between $L$ and $R$
Closest pair of points
Closest pair of points
Closest pair of points

$\partial = \partial_L < \partial_R$
Closest pair of points

\[ \partial = \partial_L < \partial_R \]

width = \(2\partial\)
Closest pair of points

distance = $\partial$

on top (1 L, 1 R)

$\partial = \partial_L < \partial_R$

width = $2\partial$

check 8 lower points
Closest pair of points

As the rectangle around the vertical line is width $2\partial$

A point only needs to check the next 7 points (not all points in rectangle)

It is impossible to have more than 8 points in this small an area
Closest pair of points
Closest pair of points

Closest-pair-of-points(P)

\[ X = P \text{ sorted by x coordinate (increasing)} \]
\[ Y = P \text{ sorted by y coordinate (increasing)} \]

return RecursiveCPoP(X, Y)
Closest pair of points

RecursiveCPoP(X, Y)

if(|P| ≤ 3)
    return min distance (brute force)
L = first half of X, R = second half of X
Y_L = points in Y that are in L
∂_L = RecursiveCPoP(L, Y_L)
∂_R = RecursiveCPoP(R, Y-Y_L)
∂ = min(∂_L, ∂_R)
Y' = all points within ∂ of L/R split
for each p in Y'
    for p' as up to next 7 points in Y'
        if distance(p,p') < ∂
            ∂ = distance(p,p')
return ∂
Closest pair of points

Runtime?
Closest pair of points

Runtime?
O(n) to split/copy parts of array
O(n) for comparing points in Y'
So each recursive problem is:
\[ T(n) = 2 \ T(n/2) + O(n) \]

Thus O(n \ lg \ n)
(same cost as initial sort)
Closest pair of points

3D closest point?


- Sun
- Earth
- Moon’s orbit
- Closest Approach

Time ticks in UTC
Jan. 26, 08:00
Jan. 27, 00:00
08:00
3.1 lunar distances
Closest pair of points

We can still use divide and conquer!
1. Split into L and R as before
2. Recursion has same base case (brute force on small set)
3. Only difference is in combine: check 7 points -> check 15 points
   These 15 points need to be checked in 2 dimensions (y and z)
Closest pair of points

Thus the running time is:
\[ T(n) = 2 \ T(n/2) + O(n \ lg \ n) \]
\[ = O(n \ lg^2 n) \]

So for a D dimensional point, we can find closest in:
\[ T(n,d) = 2 \ T(n/2, d) + T(n, d-1) \]
\[ = O(n \ lg^{d-1} n) \]