Closest Pair of Points

Slides by Carl Kingsford

Feb. 19, 2014

Based on AD Section 5.4
Finding closest pair of points

**Problem.** Given a set of points \( \{p_1, \ldots, p_n\} \) in the plane find the pair of points \( \{p_i, p_j\} \) that are closest together.
Brute force gives an $O(n^2)$ algorithm: just check every pair of points.

Can we do it faster? Seems like no: don’t we have to check every pair?

In fact, we can find the closest pair in $O(n \log n)$ time.

What’s a reasonable first step?
Divide

Split the points with line $L$ so that half the points are on each side.

Recursively find the pair of points closest in each half.
Merge: the hard case

Let $d = \min\{d_{\text{left}}, d_{\text{right}}\}$.

- $d$ would be the answer, except maybe $L$ split a close pair!
If there is a pair \( \{p_i, p_j\} \) with \( \text{dist}(p_i, p_j) < d \) that is split by the line, then both \( p_i \) and \( p_j \) must be within distance \( d \) of \( L \).

Let \( S_y \) be an array of the points in that region, sorted by decreasing \( y \)-coordinate value.
Slab Might Contain All Points

Let $S_y$ be an array of the points in that region, sorted by decreasing $y$-coordinate value.

$S_y$ might contain all the points, so we can’t just check every pair inside it.

**Theorem.** Suppose $S_y = [p_1, \ldots, p_m]$. If $\text{dist}(p_i, p_j) < d$ then $j - i \leq 15$.

In other words, if two points in $S_y$ are close enough in the plane, they are close in the array $S_y$. 
Proof, 1

Divide the region up into squares with sides of length $d/2$:

How many points in each box?
Proof, 1

Divide the region up into squares with sides of length $d/2$:

How many points in each box?

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
</tbody>
</table>

At most 1 because each box is completely contained in one half and no two points in a half are closer than $d$. 
Proof, 2

Suppose the theorem is false and two points $x,y$ with $\text{dist}(x,y) < d$ are separated by $> 15$ indices.

Then, at least 3 full rows separate them (the packing shown is the smallest possible).

But the height of 3 rows is $> 3d/2$, which is $> d$.

So the two points are father than $d$ apart. Contradiction!
Linear Time Merge

Therefore, we can scan $S_y$ for pairs of points separated by $< d$ in linear time.

ClosestPair($P_x$, $P_y$):

\[
\text{if } |P_x| == 2: \text{ return dist}(P_x[1], P_x[2]) \quad // \text{ base}
\]

\[
d_1 = \text{ClosestPair}(\text{FirstHalf}(P_x, P_y)) \quad // \text{ divide}
\]

\[
d_2 = \text{ClosestPair}(\text{SecondHalf}(P_x, P_y))
\]

\[
d = \min(d_1, d_2)
\]

\[
S_y = \text{points in } P_y \text{ within } d \text{ of } L \quad // \text{ merge}
\]

For $i = 1, \ldots, |S_y|$:

\[
\text{For } j = 1, \ldots, 15:
\]

\[
\quad d = \min(\text{dist}(S_y[i], S_y[i+j]), d)
\]

Return $d$
Total Running Time

- Divide set of points in half each time: $O(\log n)$ depth recursion

- Merge takes $O(n)$ time.

- Recurrence: $T(n) \leq 2T(n/2) + cn$

- Same as MergeSort $\implies O(n \log n)$ time.