Algorithm Design Techniques

- Induction
- **Divide and Conquer**
- Dynamic Programming
- Greedy
- Reduction

CS140 15-1

Closest Pair

- Input: Set of points $n \geq 2$ on the plane $\{(x_i,y_i) \mid 1 \leq i \leq n\}$
- Output: Pair of closest points (under Euclidian distance)

CS140 15-2

Closest Pair - Example

- Input: {(1,1),(2,2),(10,1),(10,2)}
- Output: (10,1) and (10,2)

CS140 15-3

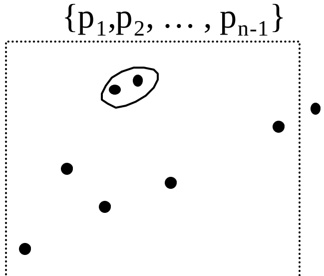
Closest Pair – Algorithm Naïve Approach

- Compute pair-wise distances
- Choose smallest

Requires $O(n^2)$ time

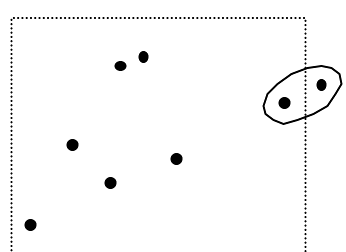
CS140 15-4

Inductive Approach Find Closest Pair in $\{p_1, p_2, \ldots, p_{n-1}\}$



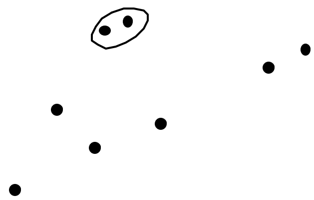
CS140 15-5

Inductive Approach cont. Find Closest point to p_n



CS140 15-6

Inductive Approach cont.
Choose best of best



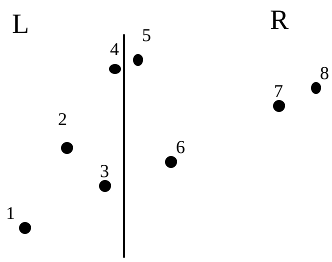
CS140 15-7

Inductive Approach

- $T(n) = T(n-1) + O(n) = O(n^2)$

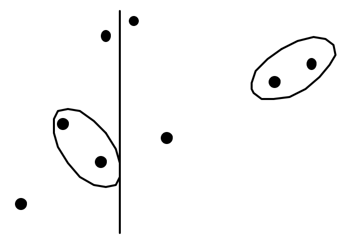
CS140 15-8

Divide and Conquer
Partition by x-coordinate



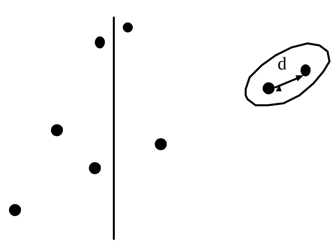
CS140 15-9

Divide and Conquer cont.
Solve sub-problems



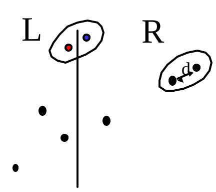
CS140 15-10

Divide and Conquer cont.
Choose best of best



CS140 15-11

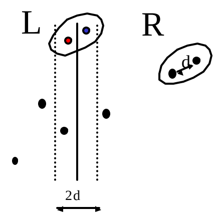
What have we missed?
Suppose p and q are closer than d.



1. Then p ε L and q ε R

CS140 15-12

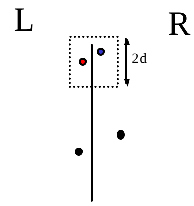
What have we missed?
Suppose p and q are closer than d.



1. Then p ε L and q ε R
2. They are within d of the L-R boundary

CS140 15-13

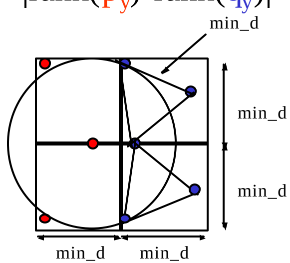
What have we missed?
Suppose p and q are closer than d.



1. Then p ε L and q ε R
2. They are within d of the L-R boundary
3. Their y-coordinates differ by less than d

CS140 15-14

If d(p,q)<min_d then
|rank(p_y)-rank(q_y)|<4



CS140 15-15

Closest-pair(P)

- Let M be the median p_x for p ε P
- P_1={p ε P | $p_x \leq$ M}, P_2={p ε P | p_x > M}
- Let (r,s) be the closest of the pairs Closest-pair(P1) and Closest-pair(P2)
- Let Q={p ε P | d(r,s)>|p_x-M|}
- Sort Q by y-coordinate.
- Find closest pair p_i and p_j ε Q, where 0<|i-j|<4.
- Return best pair found

CS140 15-16

Running time of Closest Pair

- T(n) = 2T(n/2) + n lg(n) = O(n lg(n))

CS140 15-17

Matrix Multiplication

- Input: Two n x n matrices A, B
- Output: The product AB

CS140 15-18

Matrix Multiplication - Example

- Input:

-1	3
2	4

2	-3
2	1

- Output:

4	6
12	-2

CS140 15-19

Matrix Multiplication -Algorithm

```
For i = 1 to n
    For j = 1 to n
        C[i,j]=0
        For k = 1 to n
            C[i,j] = C[i,j] + A[i,k]B[k,j]
```

Requires n^3 multiplications and $n^2(n-1)$ additions.

(Note the input size is $2n^2$.)

CS140 15-20

Divide and Conquer

- Input:

A_{11}	A_{12}
A_{21}	A_{22}

B_{11}	B_{12}
B_{21}	B_{22}

- Output:

C_{11}	C_{12}
C_{21}	C_{22}

$C_{ij} = A_{i1}B_{1j} + A_{i2}B_{2j}$

$T(n)=8T(n/2)+O(n^2)==O(n^3)$

CS140 15-21

Can we do better?

- $T(n)=aT(n/2)+O(n^2)=o(n^3)$ if $a<8$
- Strassen's algorithm:

  $a=7$ and $T(n)=O(n^{\log_2 7})$

CS140 15-22

Strassen's Algorithm

2x2 case in 7 multiplications

a	b
c	d

X

e	g
f	h

=

ae+bf	ag+bh
ce+df	cg+dh

CS140 15-23

Strassen's Algorithm

Rephrase:

a	b	0	0
c	d	0	0
0	0	a	b
0	0	c	d

e
f
g
h

=

ae+bf
ag+bh
ce+df
cg+dh

CS140 15-24

Strassen's Algorithm

$$\begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix} \begin{bmatrix} e \\ f \\ g \\ h \end{bmatrix} = \begin{bmatrix} ae+bf \\ ag+bh \\ ce+df \\ cg+dh \end{bmatrix}$$

MAGIC

- We can do with 7 multiplications …