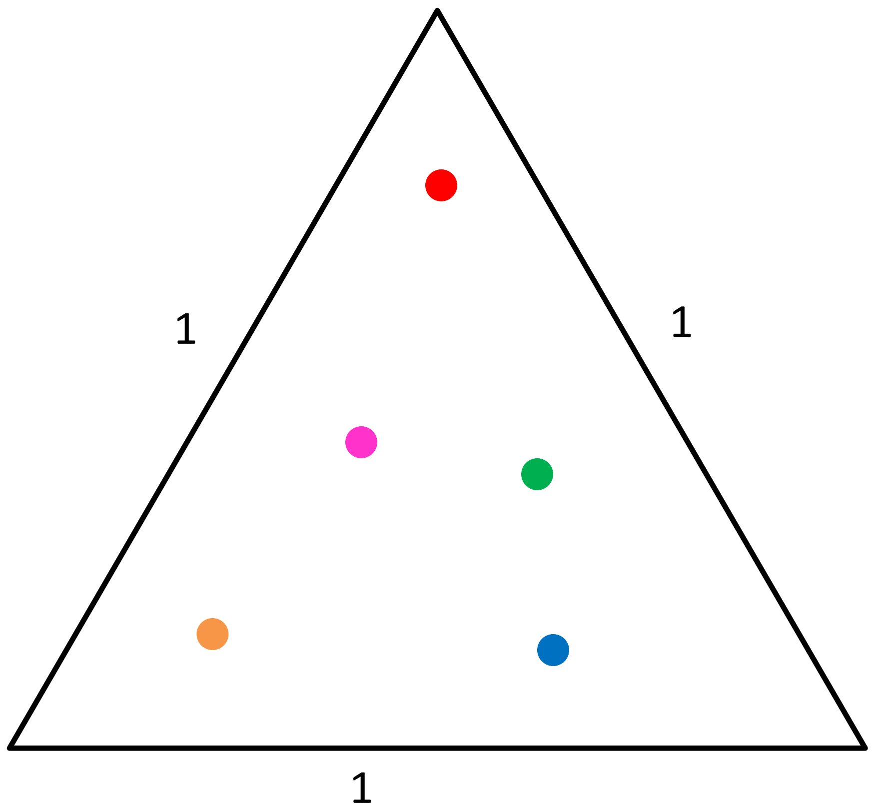
CS4102 Algorithms

Nate Brunelle

Fall 2017

Warm up

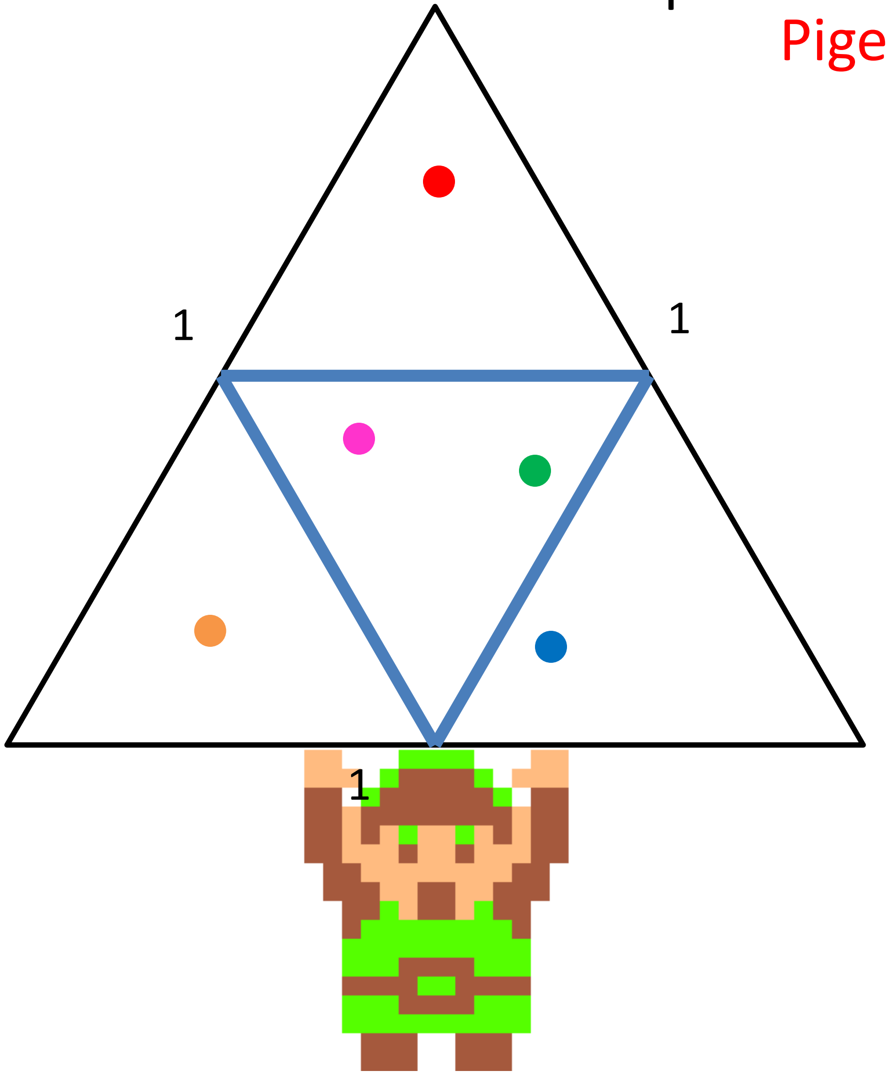
Given 5 points on the unit equilateral triangle, show there's always a pair distance $\leq \frac{1}{2}$ apart



If points p_1, p_2 in same quadrant, then $\delta(p_1, p_2) \le \frac{1}{2}$

Given 5 points, two must share the same quadrant

Pigeonhole Principle!



Today's Keywords

- Divide and Conquer
- Recurrences
- Master's Theorem
- Substitution Method
- Closest Pair of Points

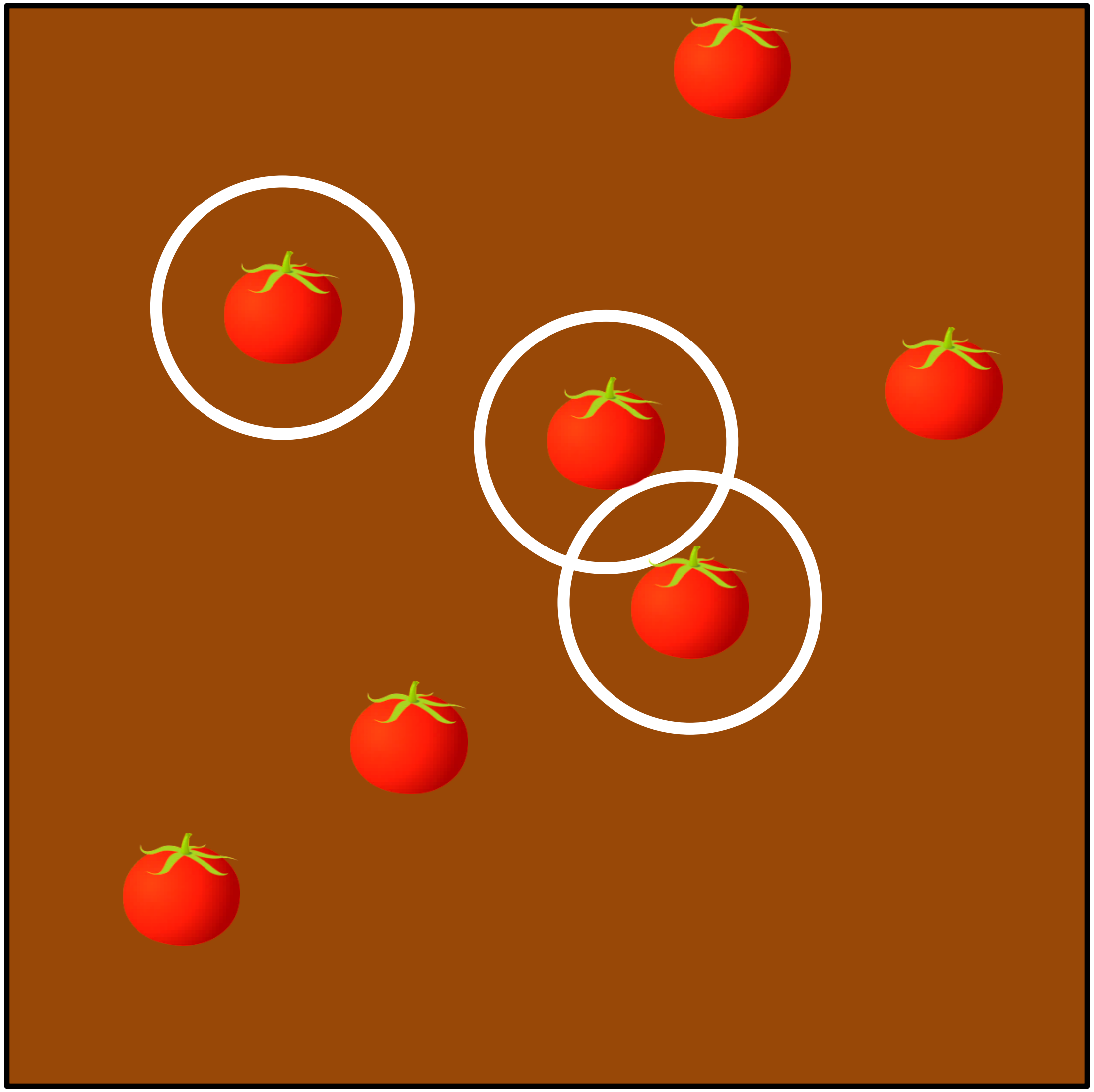
CLRS Readings

- Chapter 4

Homeworks

- Hw1 due 11pm this Friday!
- Hw2 due 11pm Friday, February 16!
  - Released at about 5pm today
  - Programming (use Python!)
  - Divide and conquer
  - Closest pair of points

My Garden

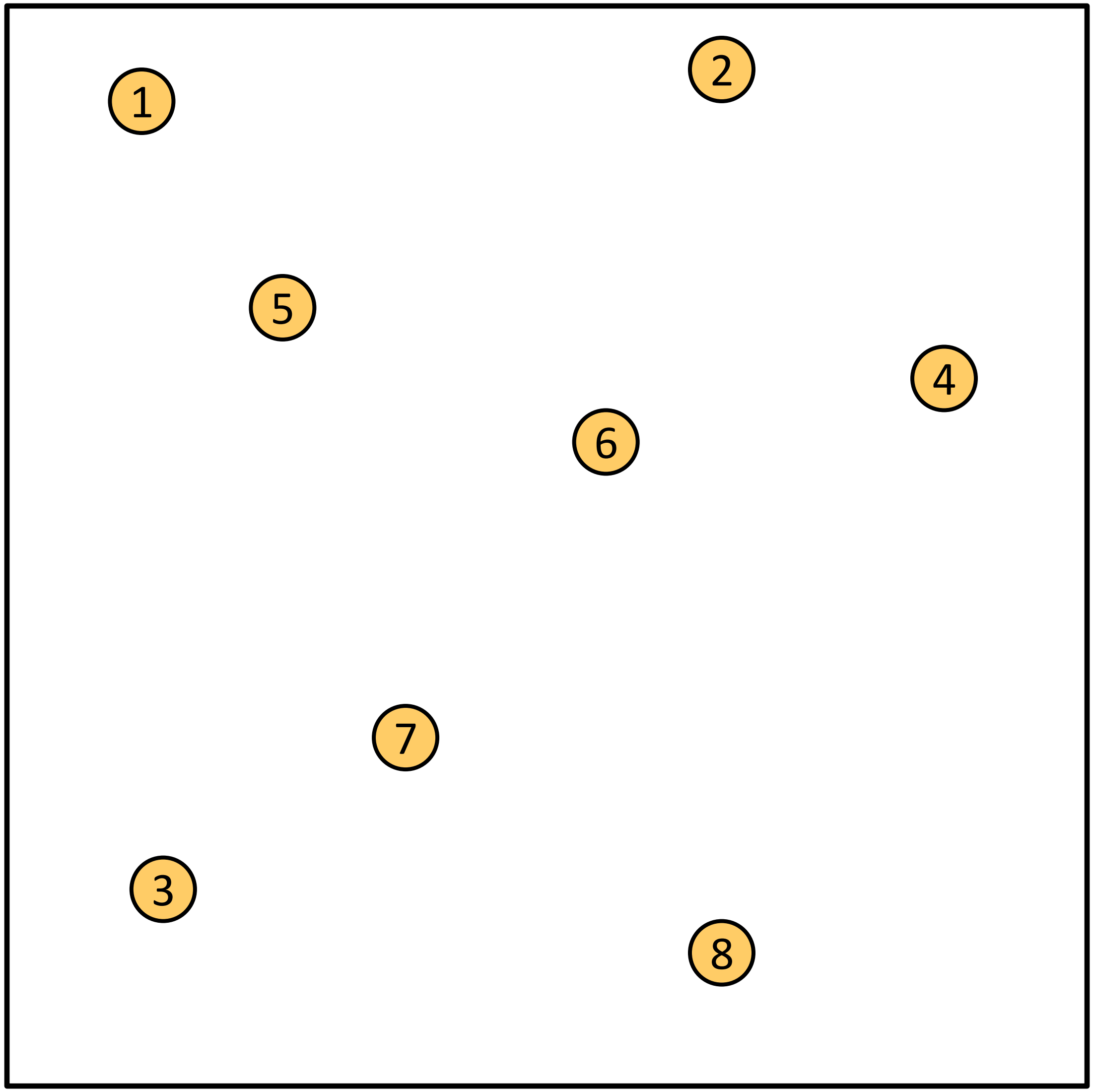


Need to find:
Closest Pair of Tomatoes

Closest Pair of Points

Given:
A list of points

Return:
Pair of points with smallest distance apart

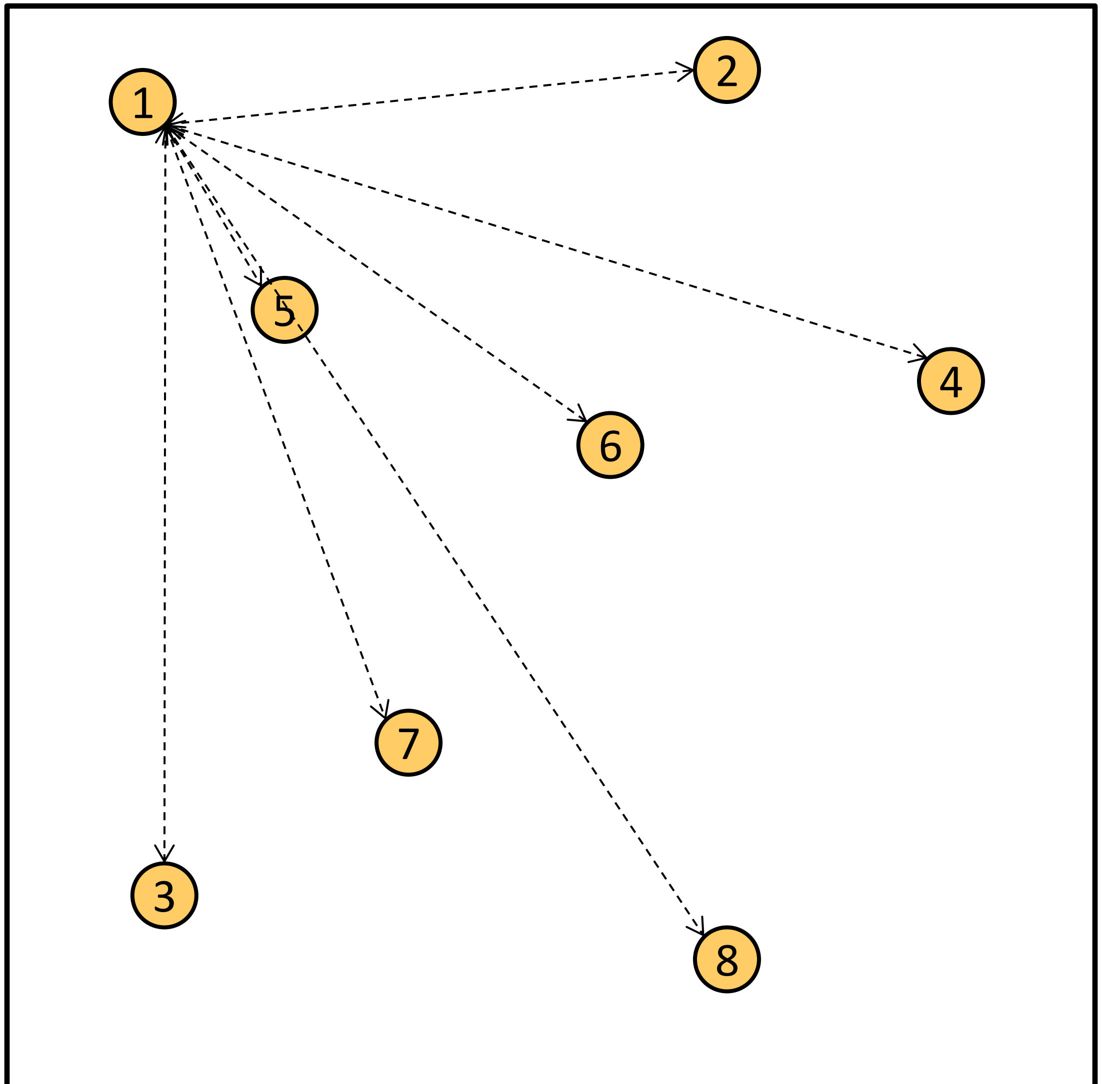


Closest Pair of Points: Naïve

Given:
A list of points

Return:
Pair of points with smallest distance apart

Algorithm: $O(n^2)$
Test every pair of points, return the closest.

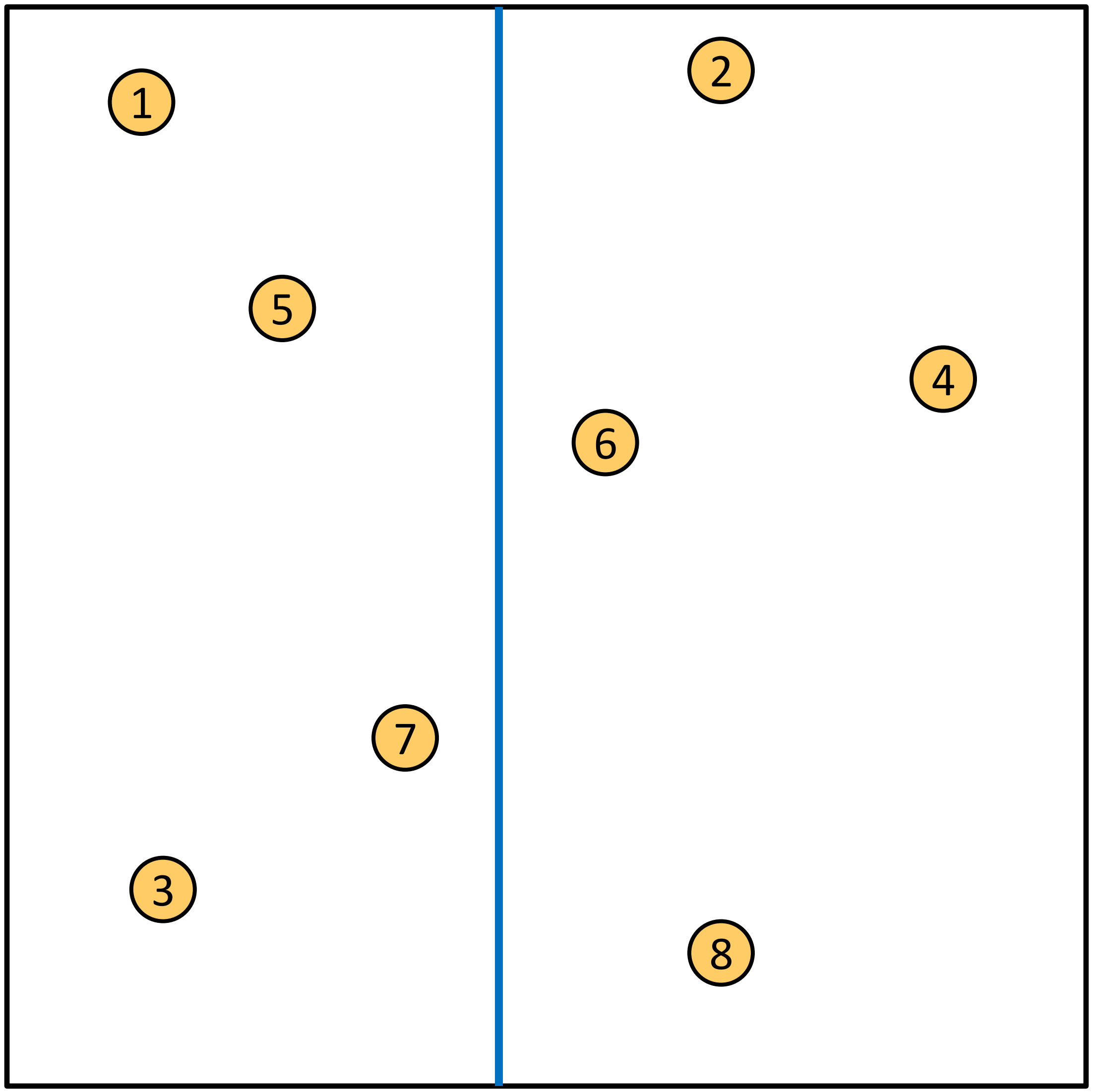


Closest Pair of Points: D&C

Divide: How?

At median x coordinate

Conquer:



Closest Pair of Points: D&C

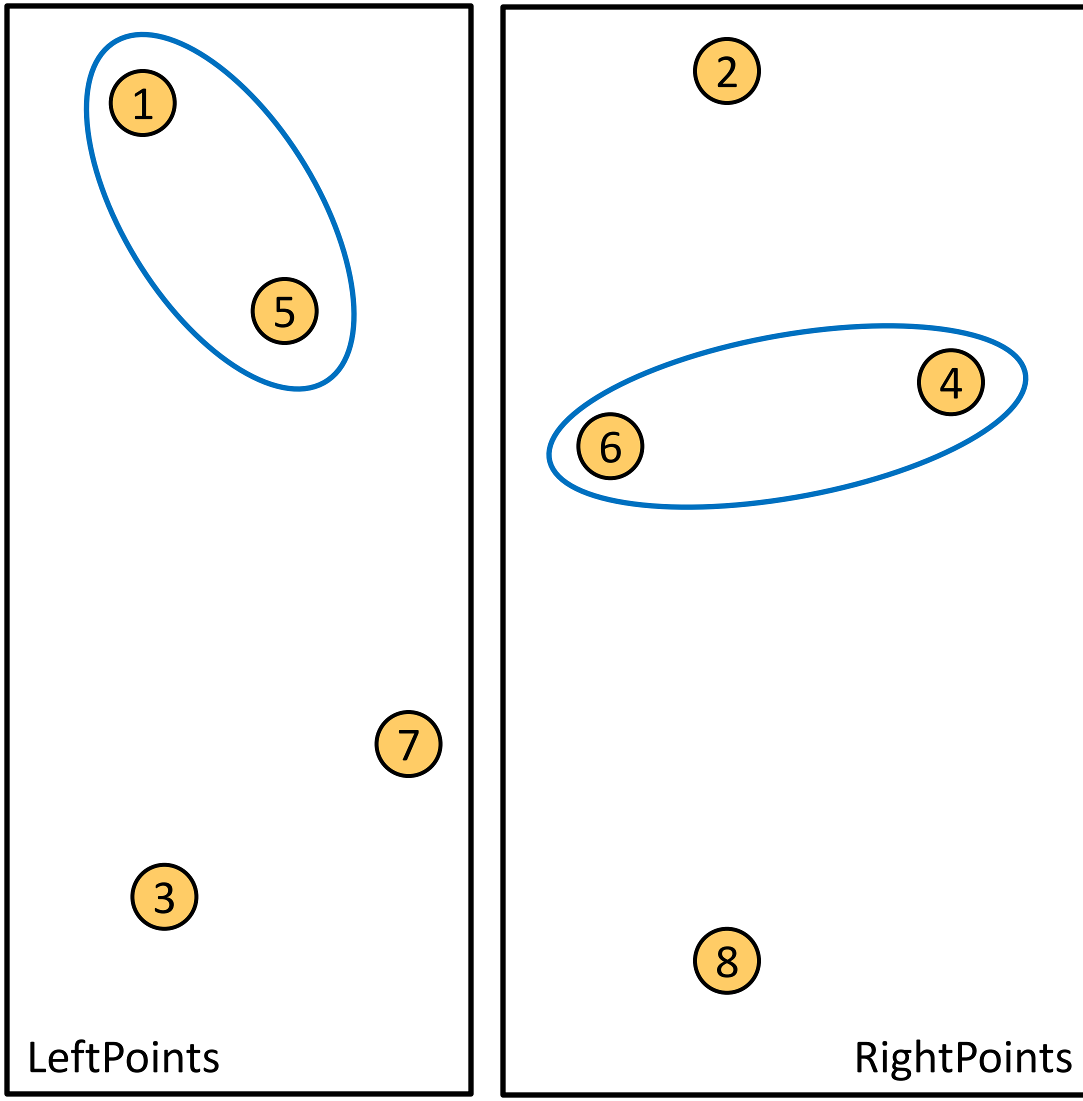
Divide:

At median x coordinate

Conquer:

Recursively find closest pairs from Left and Right

Combine:

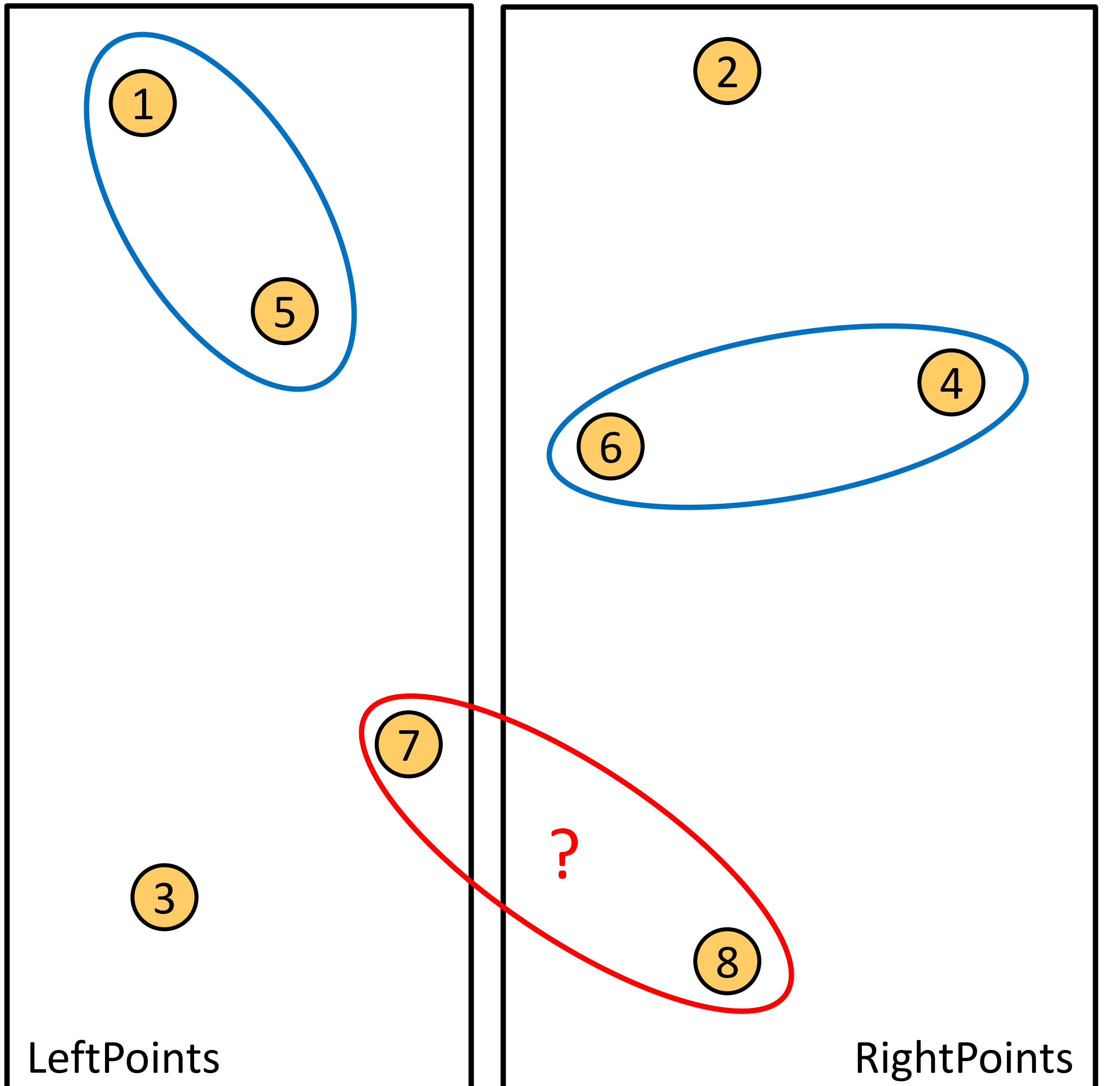


Closest Pair of Points: D&C

Divide:
At median x coordinate

Conquer:
Recursively find closest pairs from Left and Right

Combine:
Return min of Left and Right pairs Problem?



Closest Pair of Points: D&C

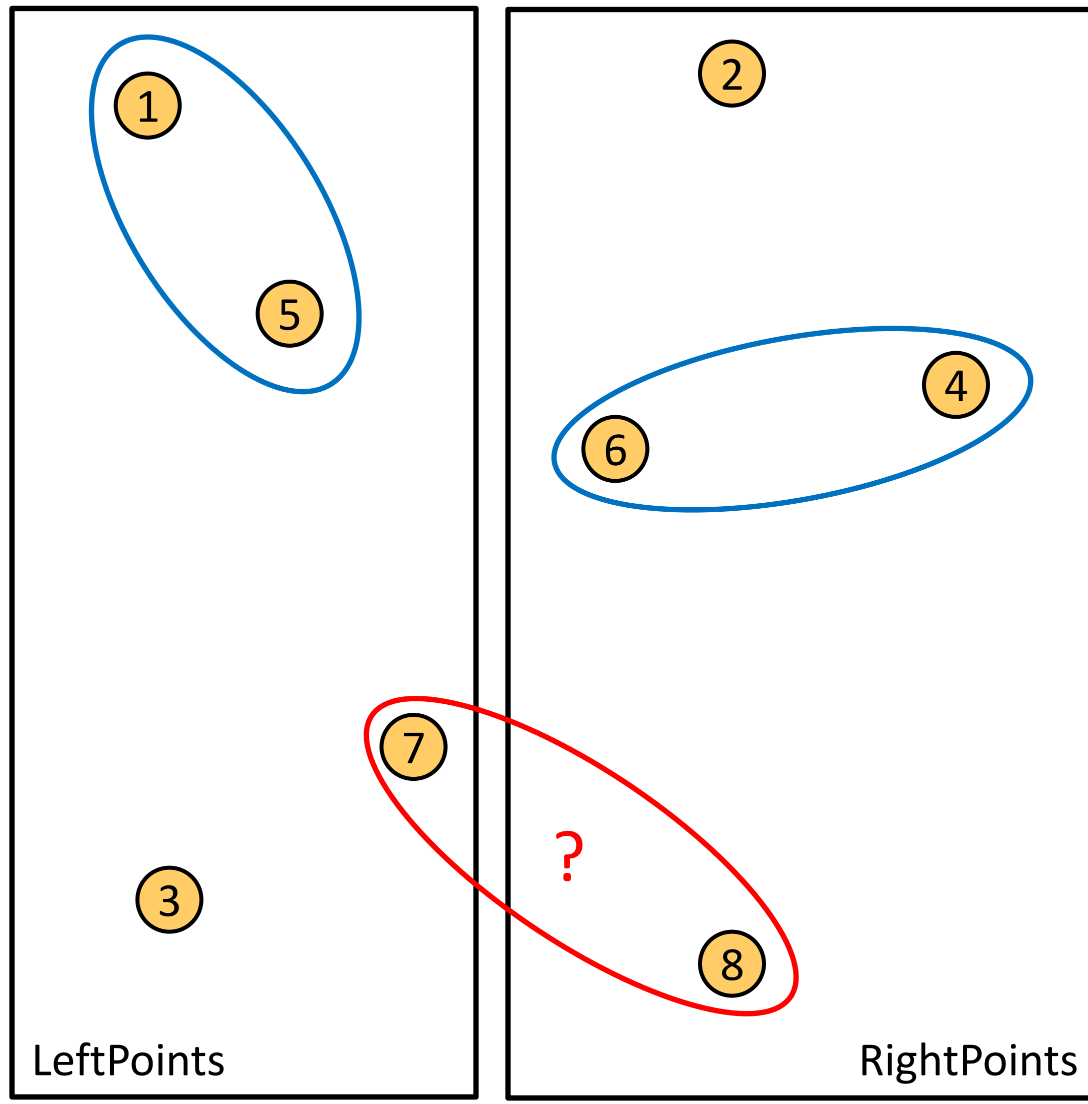
Combine:

2 Cases:

1. Closest Pair is completely in Left or Right

2. Closest Pair Spans our "Cut"

Need to test points across the cut



Spanning the Cut

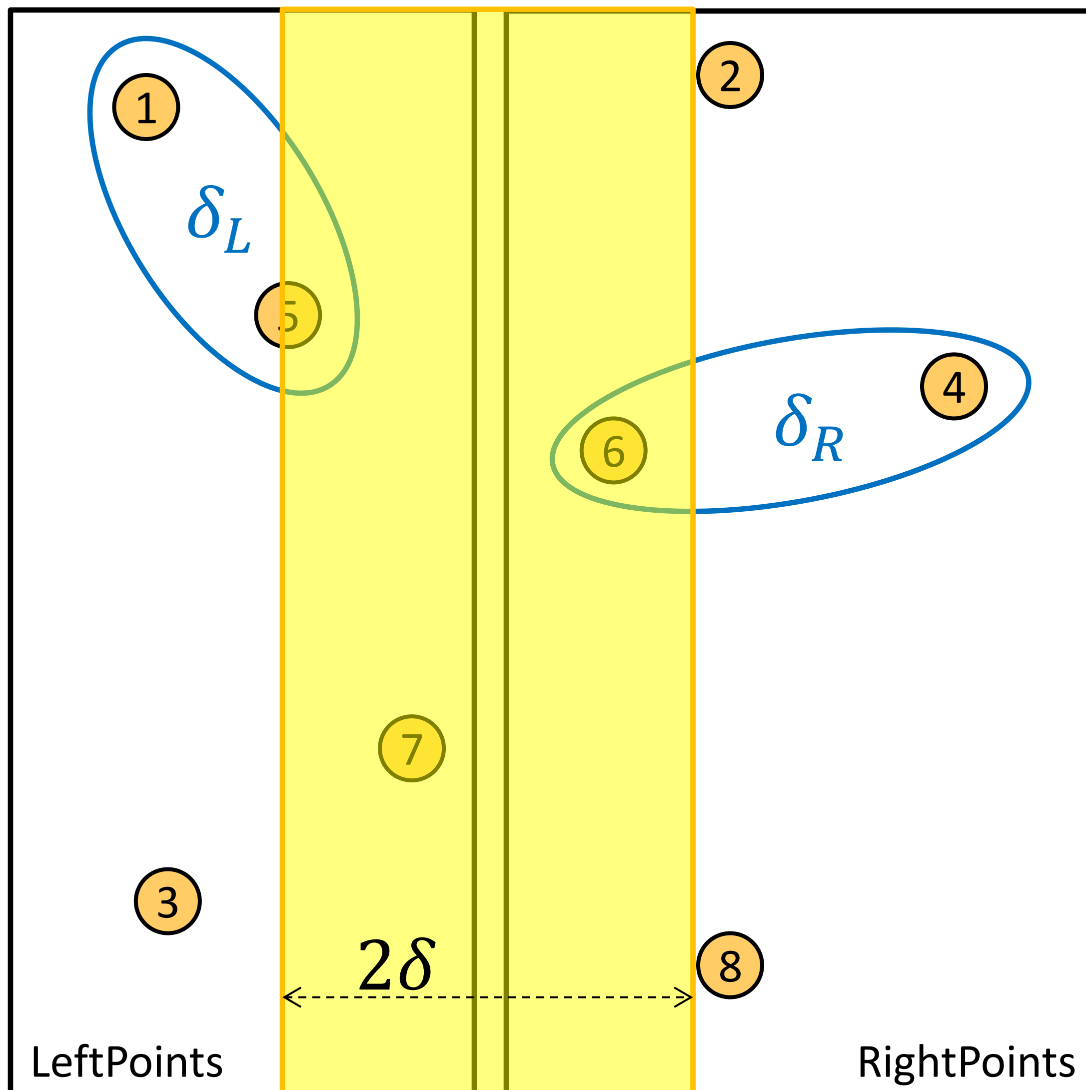
Combine:

2. Closest Pair Spanned our "Cut"

Need to test points across the cut

Compare all points within $\delta = \min\{\delta_L, \delta_R\}$ of the cut.

How many are there?



Spanning the Cut

Combine:

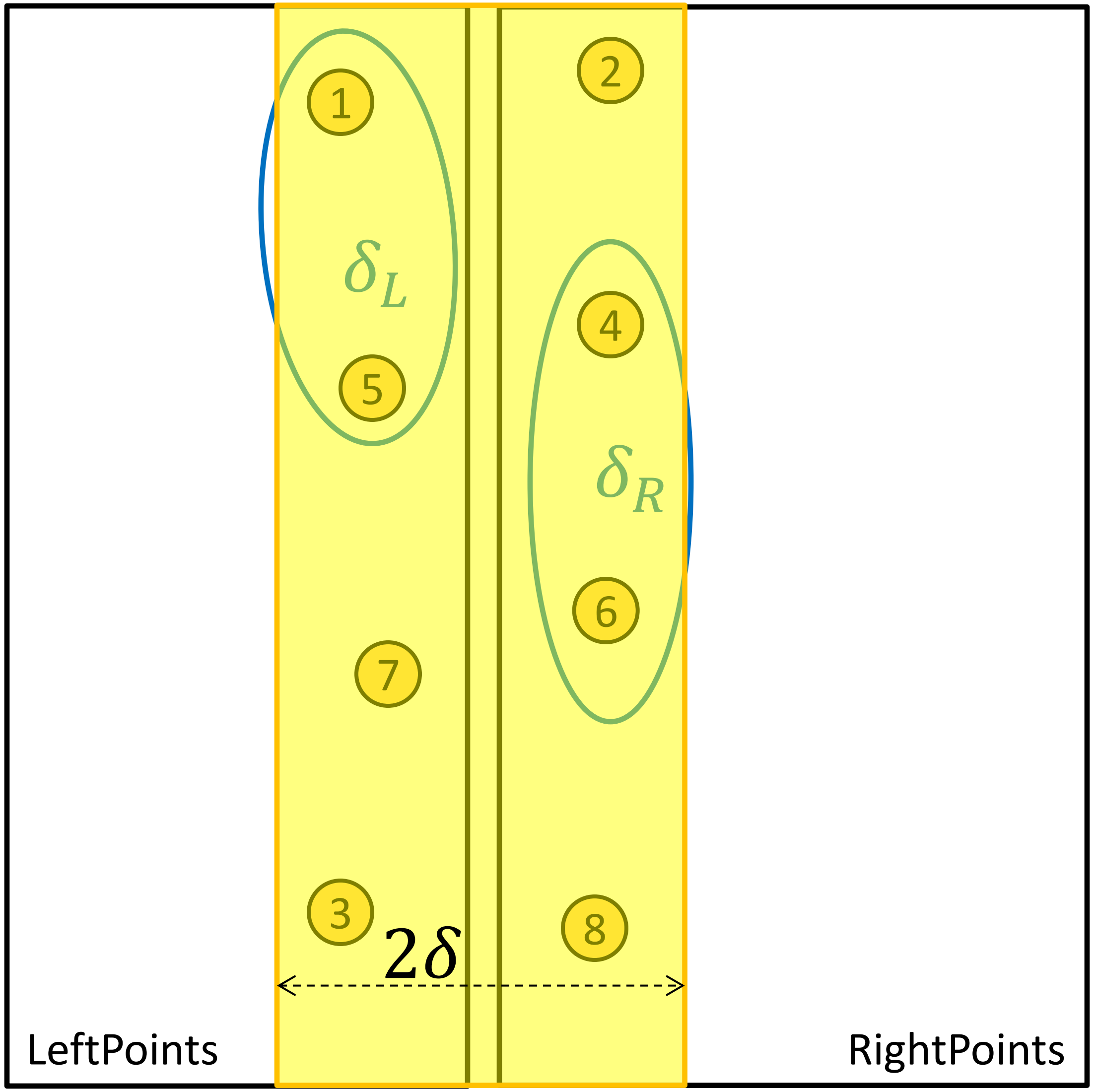
2. Closest Pair Spanned our "Cut"

Need to test points across the cut

Compare all points within $\delta = \min\{\delta_L, \delta_R\}$ of the cut.

How many are there?

$$T(n) = 2T\left(\frac{n}{2}\right) + \left(\frac{n}{2}\right)^2 = \Theta(n^2)$$



Spanning the Cut

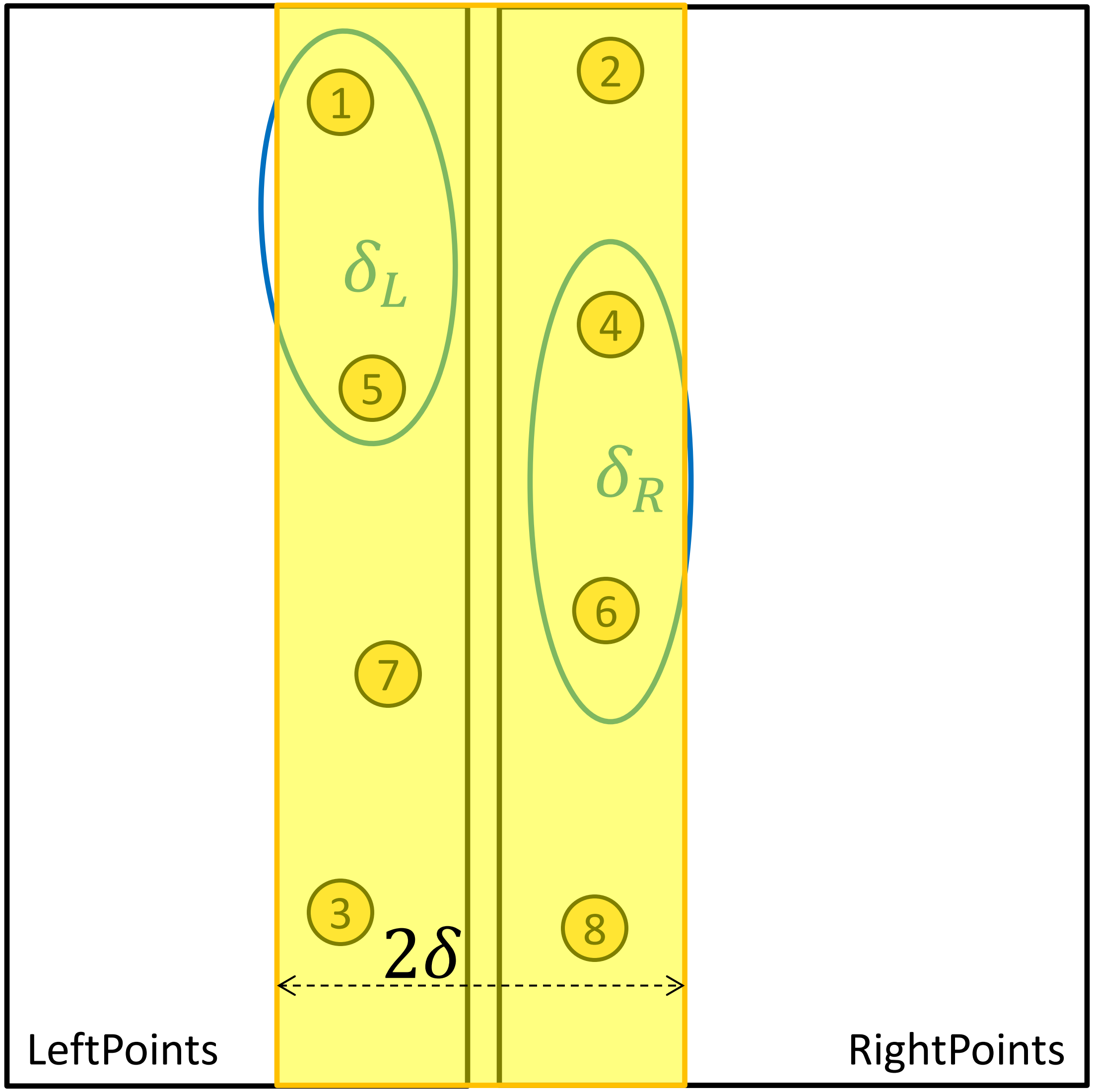
Combine:

2. Closest Pair Spanned our "Cut"

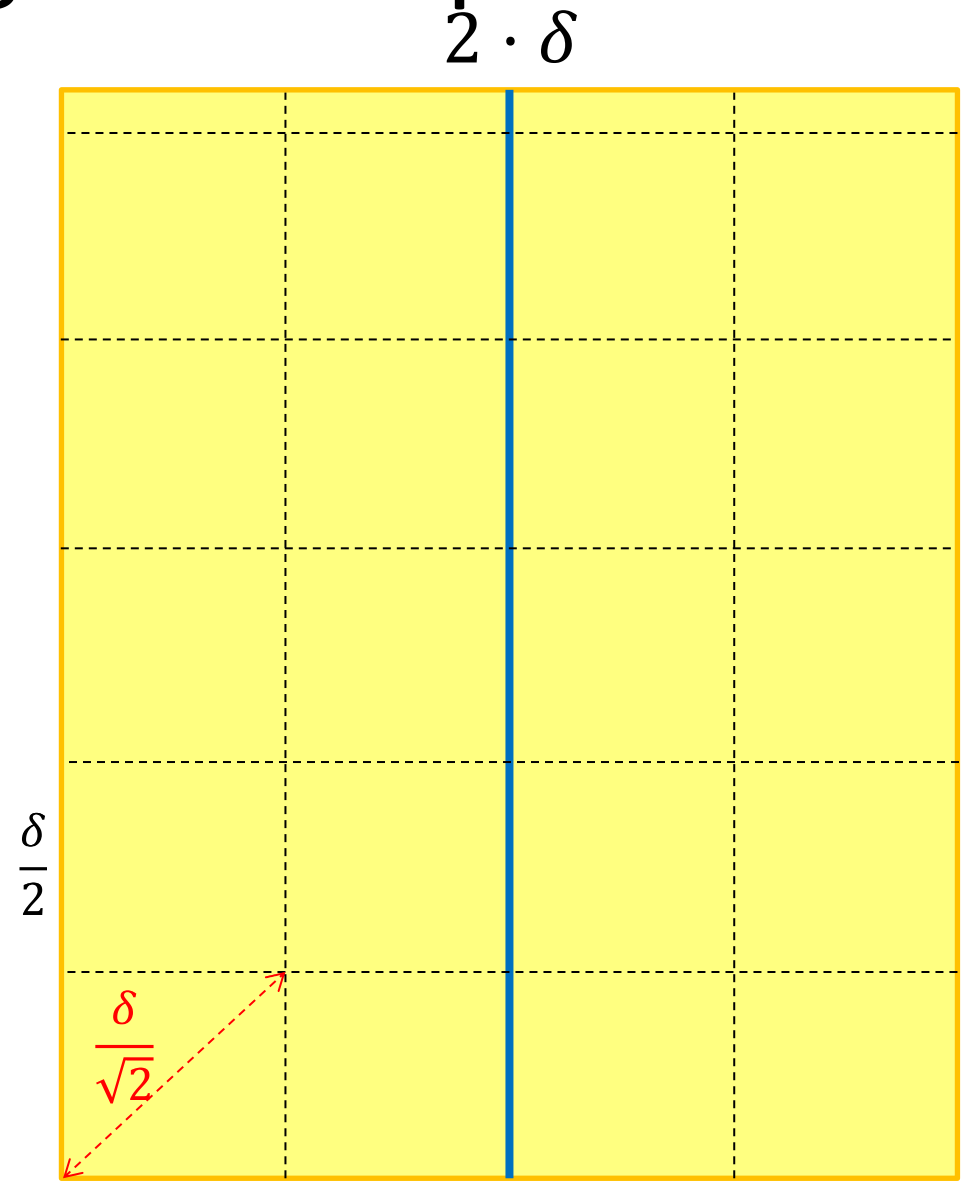
Need to test points across the cut

We don't need to test all pairs!

Only need to test points within δ of one another



Reducing Search Space



Combine:

2. Closest Pair Spanned our "Cut"

Need to test points across the cut

Divide the "runway" into square cubbies of size $\frac{\delta}{2}$

Each cubby will have at most 1 point!

Reducing Search Space

Combine:

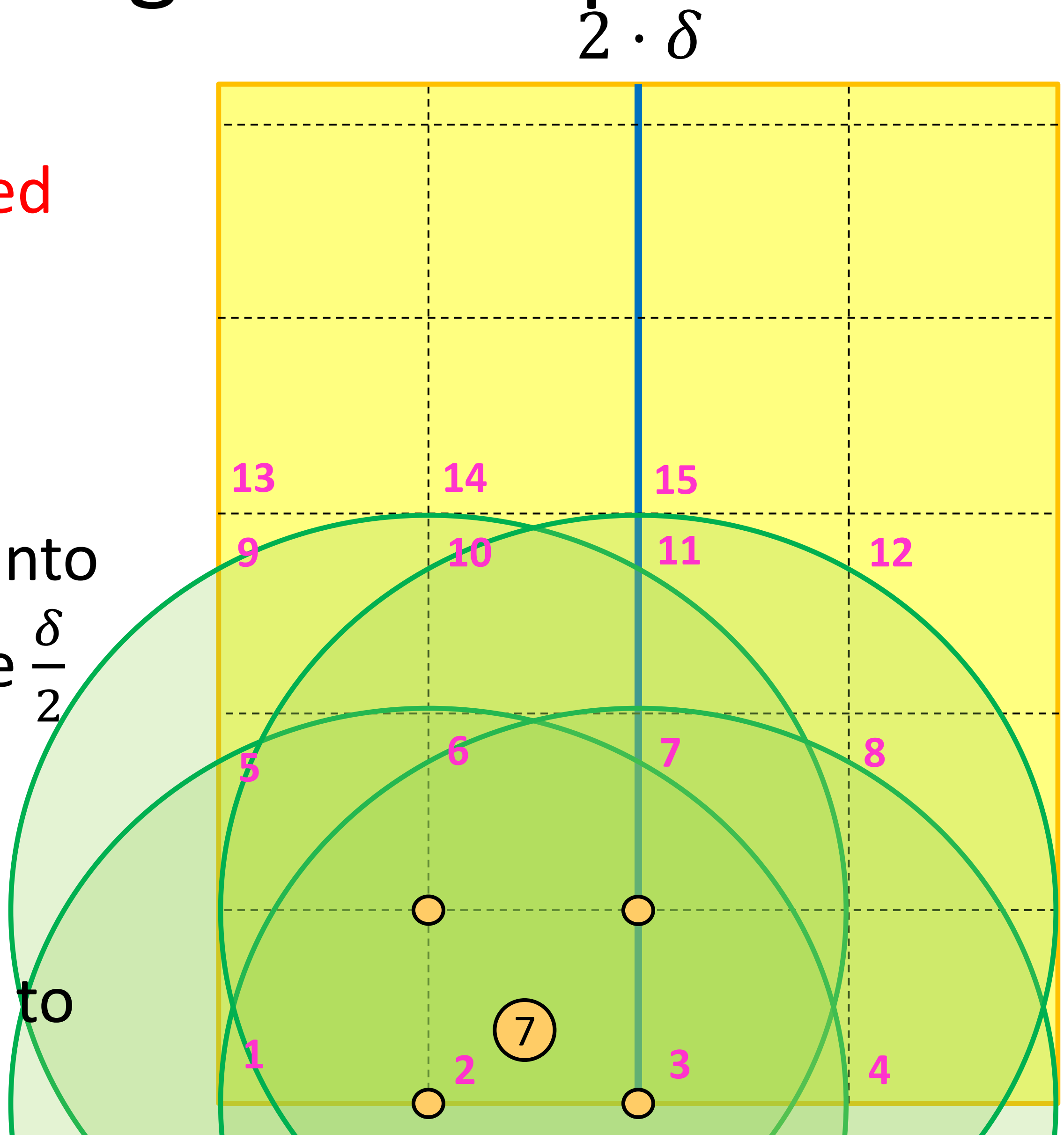
2. Closest Pair Spanned our "Cut"

Need to test points across the cut

Divide the "runway" into square cubbies of size $\frac{\delta}{2}$

How many cubbies could have a point $< \delta$ away?

Each point compared to ≤ 15 other points



Closest Pair of Points: D&C

0. Sort points by x

1. Divide: At median x

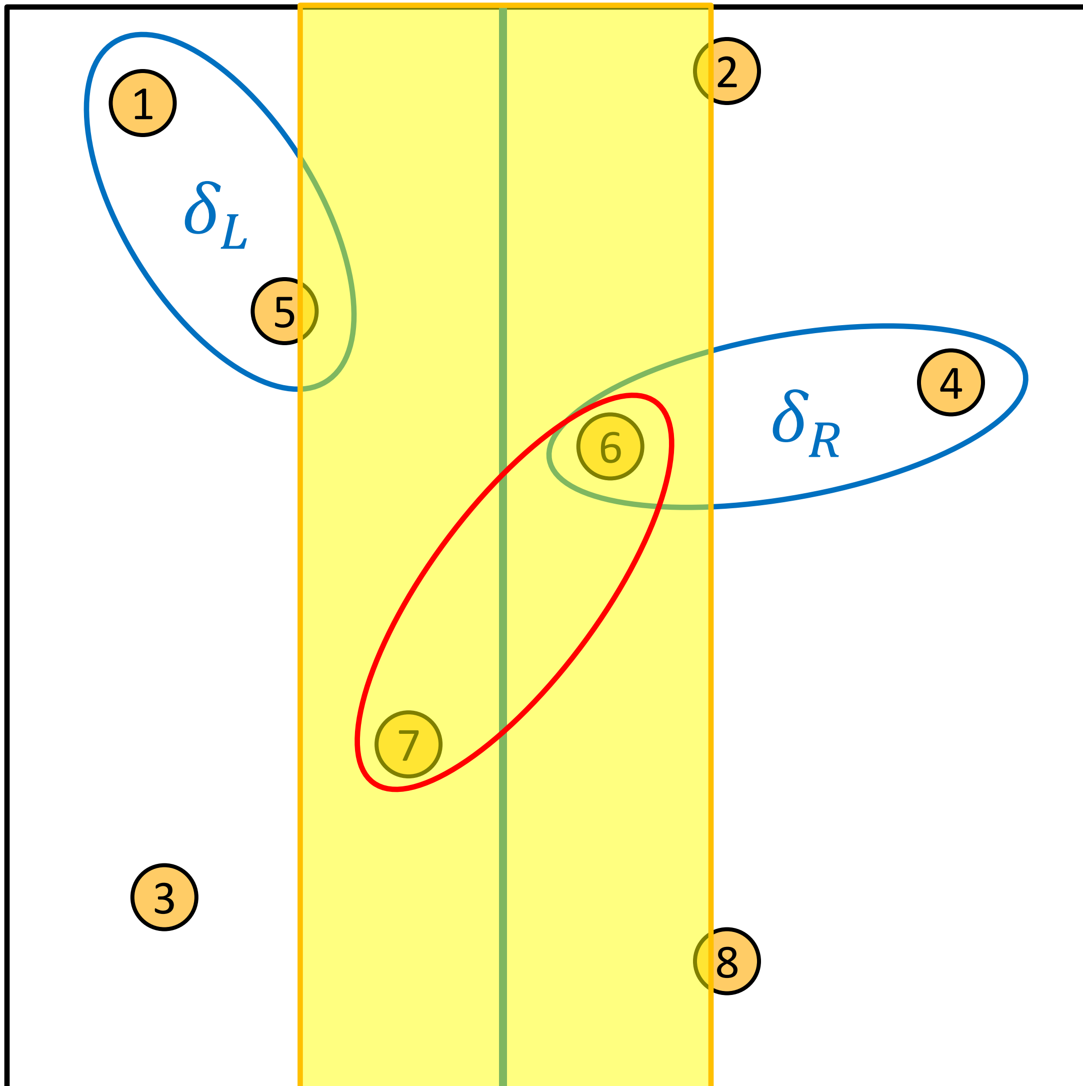
2. Conquer: If >2 points Recursively find closest pair on left and right

3. Combine:

   a. List points in "runway" in order according to y value

   b. Compare each point to the next 15 above it, save best found

   c. Return min from left, right, and 3b



Closest Pair of Points: D&C

0. Sort points by x

1. Divide: At median x

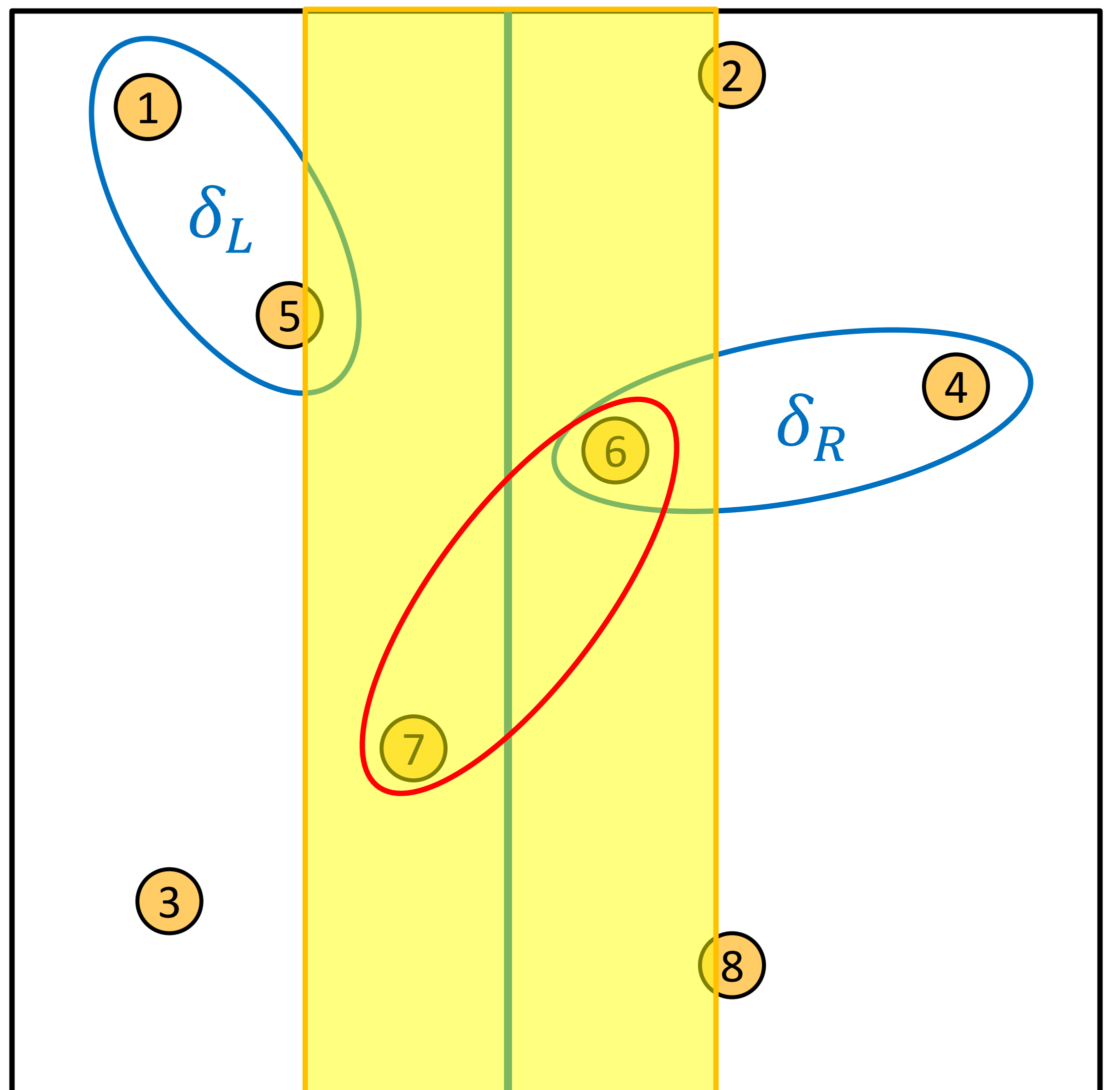
2. Conquer: If >2 points Recursively find closest pair on left and right

3. Combine:

   a. List points in “runway” in order by y

   b. Compare each runway point to the next 15 runway points, save closest pair

   c. Return min from left, right, and 3b



Listing points in "Runway"

- Given: y-sorted lists from left and right
- Return: y-sorted points in "runway"
- Target run tie? $O(n)$

Left, sorted by y

1	5	7	3

Right, sorted by y

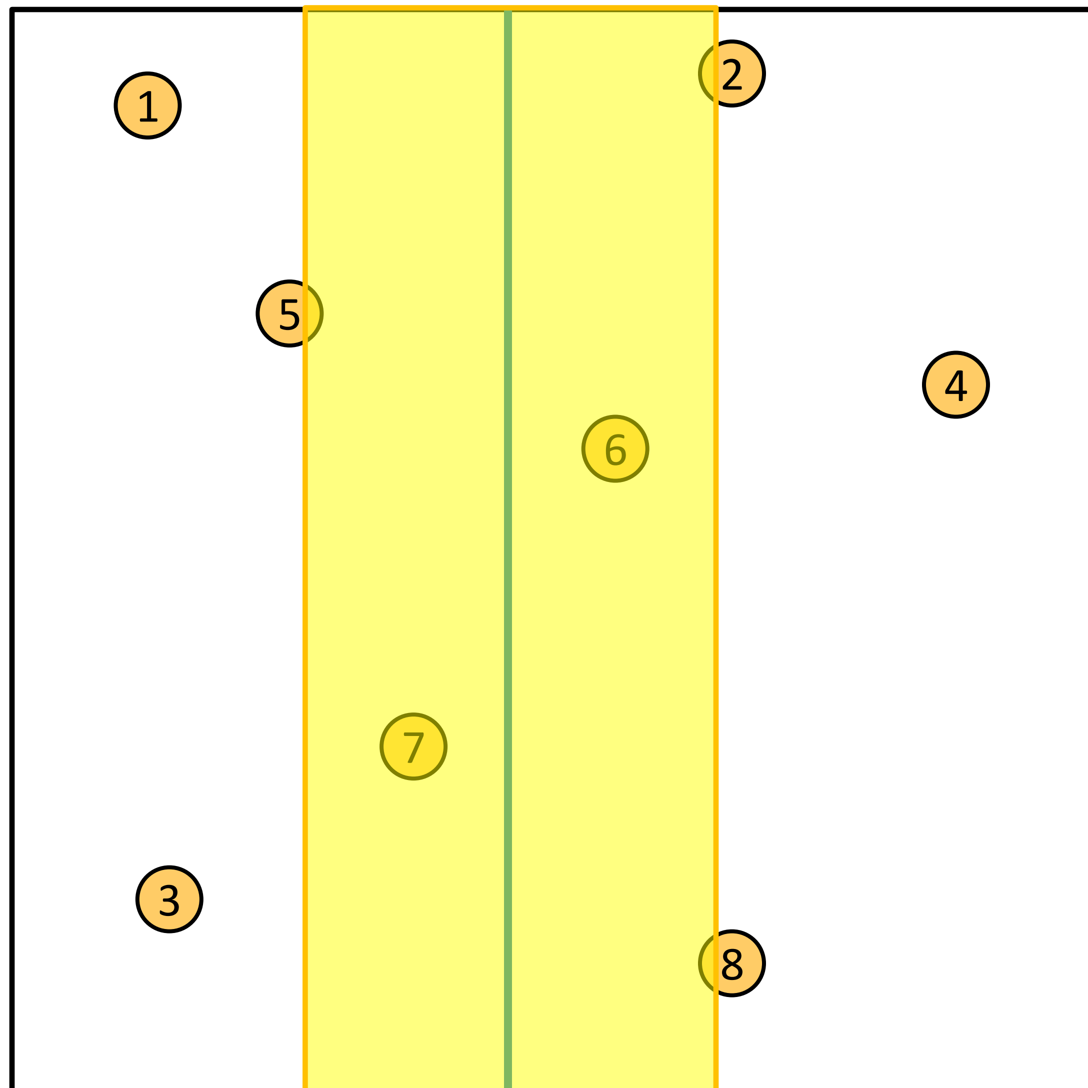
2	4	6	8

Merged, sorted by y

2	1	5	4	6	7	3	8

Runway, still sorted by y!

2	5	6	7	8



Run Time

0. Sort points by x $\Theta(n \log n)$

1. Divide: At median x $\Theta(1)$

2. Conquer: If >2 points, Recursively find closest pair on left and right $T\left(\frac{n}{2}\right)$

3. Combine:
    a. Merge points to sort by y $\Theta(n)$

    b. Compare each runway point to the next 15 runway points, save closest pair $\Theta(n)$

    c. Return y-sorted points and min from left, right, and 3b $\Theta(1)$

$$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$$

Case 2!

$$T(n) = \Theta(n \log n)$$

Matrix Multiplication

$$\begin{array}{cc} & n \\ n & \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \end{array} \times \begin{bmatrix} 2 & 4 & 6 \\ 8 & 10 & 12 \\ 14 & 16 & 18 \end{bmatrix}$$

$$= \begin{bmatrix} 2+16+42 & 4+20+48 & 6+24+54 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$$

$$= \begin{bmatrix} 60 & 72 & 84 \\ 132 & 162 & 192 \\ 204 & 252 & 300 \end{bmatrix}$$

Run time? $O(n^3)$

Matrix Multiplication D&C

Multiply $n \times n$ matrices (A and B)

Divide:

$$A = \left[\begin{array}{cc|cc} a_1 & a_2 & a_3 & a_4 \\ a_5 & a_6 & a_7 & a_8 \\ \hline a_9 & a_{10} & a_{11} & a_{12} \\ a_{13} & a_{14} & a_{15} & a_{16} \end{array}\right] \qquad B = \left[\begin{array}{cc|cc} b_1 & b_2 & b_3 & b_4 \\ b_5 & b_6 & b_7 & b_8 \\ \hline b_9 & b_{10} & b_{11} & b_{12} \\ b_{13} & b_{14} & b_{15} & b_{16} \end{array}\right]$$

Matrix Multiplication D&C

Multiply $n \times n$ matrices (A and B)

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \qquad B = \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix}$$

Combine:

$$AB = \begin{bmatrix} A_{1,1}B_{1,1} + A_{1,2}B_{2,1} & A_{1,1}B_{1,2} + A_{1,2}B_{2,2} \\ A_{2,1}B_{1,1} + A_{2,2}B_{2,1} & A_{2,1}B_{1,2} + A_{2,2}B_{2,2} \end{bmatrix}$$

Run time? $T(n) = 8T\left(\frac{n}{2}\right) + \boxed{4\left(\frac{n}{2}\right)^2}$ Cost of additions

Matrix Multiplication D&C

$$T(n) = 8T\left(\frac{n}{2}\right) + 4\left(\frac{n}{2}\right)^2$$

$$T(n) = 8T\left(\frac{n}{2}\right) + n^2$$

$a = 8, b = 2, f(n) = n^2$

Case 1!

$n^{\log_b a} = n^{\log_2 8} = n^3$

$T(n) = \Theta(n^3)$

We can do better…

Matrix Multiplication D&C

Multiply $n \times n$ matrices (A and B)

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \qquad B = \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix}$$

$$\boxed{AB = \begin{bmatrix} A_{1,1}B_{1,1} + A_{1,2}B_{2,1} & A_{1,1}B_{1,2} + A_{1,2}B_{2,2} \\ A_{2,1}B_{1,1} + A_{2,2}B_{2,1} & A_{2,1}B_{1,2} + A_{2,2}B_{2,2} \end{bmatrix}}$$

Idea: Use a Karatsuba-like technique on this

Strassen's Algorithm

Multiply $n \times n$ matrices (A and B)

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \qquad B = \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix}$$

Calculate:

$$Q_1 = (A_{1,1} + A_{2,2})(B_{1,1} + B_{2,2})$$
$$Q_2 = (A_{2,1} + A_{2,2})B_{1,1}$$
$$Q_3 = A_{1,1}(B_{1,2} - B_{2,2})$$
$$Q_4 = A_{2,2}(B_{2,1} - B_{1,1})$$
$$Q_5 = (A_{1,1} + A_{1,2})B_{2,2}$$
$$Q_6 = (A_{2,1} - A_{1,1})(B_{1,1} + B_{1,2})$$
$$Q_7 = (A_{1,2} - A_{2,2})(B_{2,1} + B_{2,2})$$

Find AB:

$$\begin{bmatrix} Q_1 + Q_4 - Q_5 + Q_7 & Q_3 + Q_5 \\ Q_2 + Q_4 & Q_1 - Q_2 + Q_3 + Q_6 \end{bmatrix}$$

$$\begin{bmatrix} A_{1,1}B_{1,1} + A_{1,2}B_{2,1} & A_{1,1}B_{1,2} + A_{1,2}B_{2,2} \\ A_{2,1}B_{1,1} + A_{2,2}B_{2,1} & A_{2,1}B_{1,2} + A_{2,2}B_{2,2} \end{bmatrix}$$

Number Mults.: 7 Number Adds.: 18

$$T(n) = 7T\left(\frac{n}{2}\right) + \frac{9}{2}n^2$$

Strassen's Algorithm

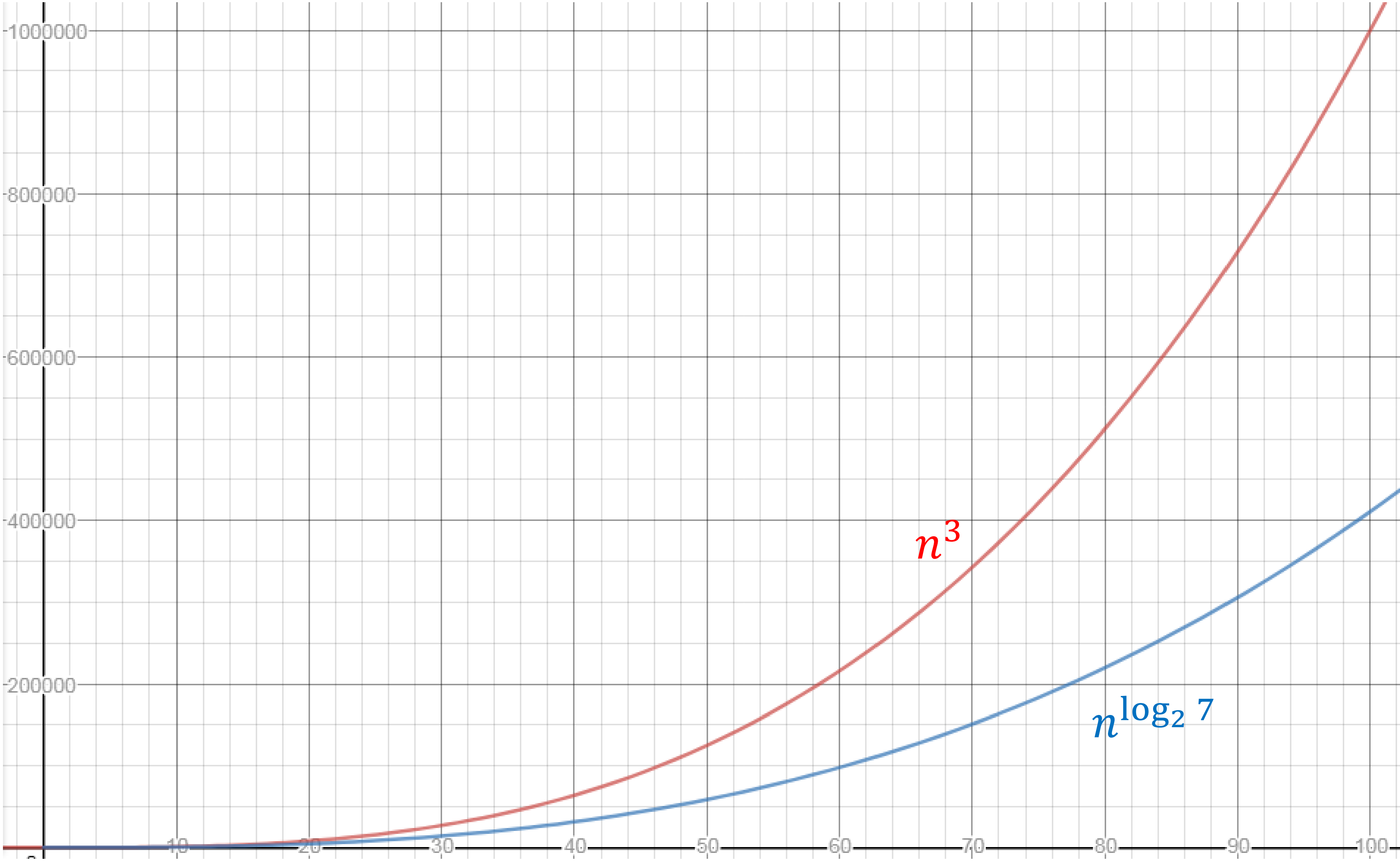
$$T(n) = 7T\left(\frac{n}{2}\right) + \frac{9}{2}n^2$$

$$a = 7, b = 2, f(n) = \frac{9}{2}n^2$$

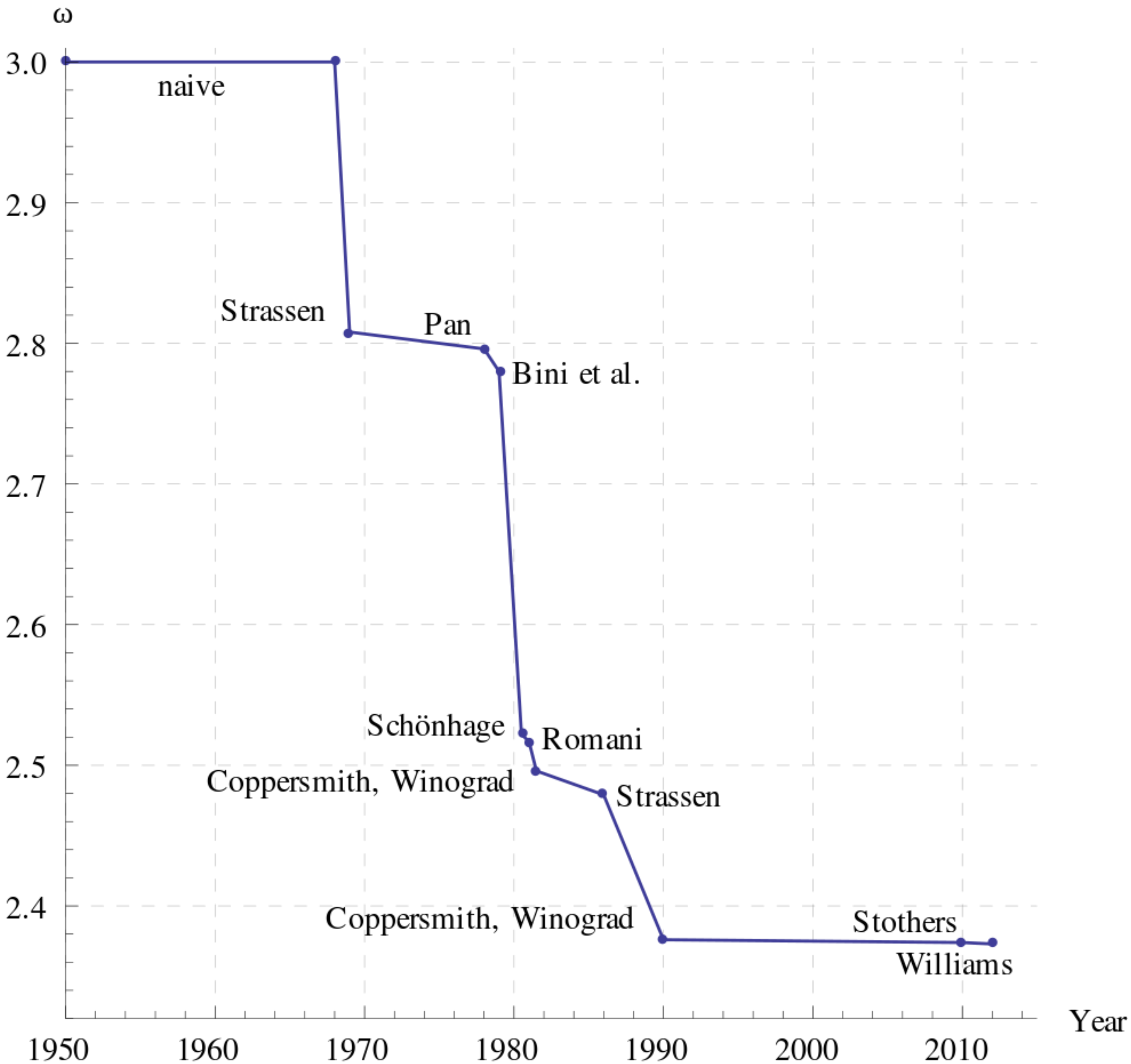
Case 1!

$$n^{\log_b a} = n^{\log_2 7} \approx n^{2.807}$$

$$T(n) = \Theta\left(n^{\log_2 7}\right) \approx \Theta(n^{2.807})$$


1000000
800000
600000
400000
200000
0
10
20
30
40
50
60
70
80
90
100
n^3
$n^{\log_2 7}$

Is this the fastest?



Best possible is unknown

May not even exist!