Divide-and-conquer

- Divide a problem of size n
- Conquer: "a" subproblems of size "$\frac{1}{b}$" of the size of the original problem
- Combine

$$T(n) = \text{Divide}(n) + a \cdot T\left(\frac{n}{b}\right) + \text{Combine}(n)$$

$a \geq 1$
$b > 1$

$$\boxed{T(n) = a \cdot T\left(\frac{n}{b}\right) + f(n)} \rightarrow \text{solve with Master Theorem}$$

- **Algorithm with running time $T(n) = \Theta(n \lg n)$**

Divide-and-conquer:

- Conquer: algorithm recurses on 2 subproblems of size $\frac{n}{2}$
- Divide and Combine take $O(n)$

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

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + cn \qquad c = \text{const}$$

case 2 Master Theorem $\Rightarrow \boxed{T(n) = \Theta(n \cdot \lg n)}$

- **Algorithm with running time $T(n) = \Theta(n)$**

Divide-and-conquer:

- Conquer: algorithm recurses on 1 subproblem of size $\frac{n}{2}$
- Divide and Combine take $O(n)$

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

$$T(n) = T\left(\frac{n}{2}\right) + cn \qquad c = \text{const}$$

cn vs $n^{\log_2 1} = n^0 = 1$
$cn = \Omega(n^{\varepsilon})$ for $\varepsilon = 1$
Regularity condition checks

$\Rightarrow$ case 3 of the Master Theorem $\boxed{T(n) = \Theta(n)}$

- Algorithm with running time $T(n) = \Theta(\lg n)$

Divide-and-conquer:

$\begin{cases} \bullet \text{Conquer: algorithm recurses on 1 subproblem of size } \frac{n}{2} \\ \bullet \text{Divide and Combine take } \Theta(1) \end{cases}$

$$T(n) = T\left(\frac{n}{2}\right) + \Theta(1)$$

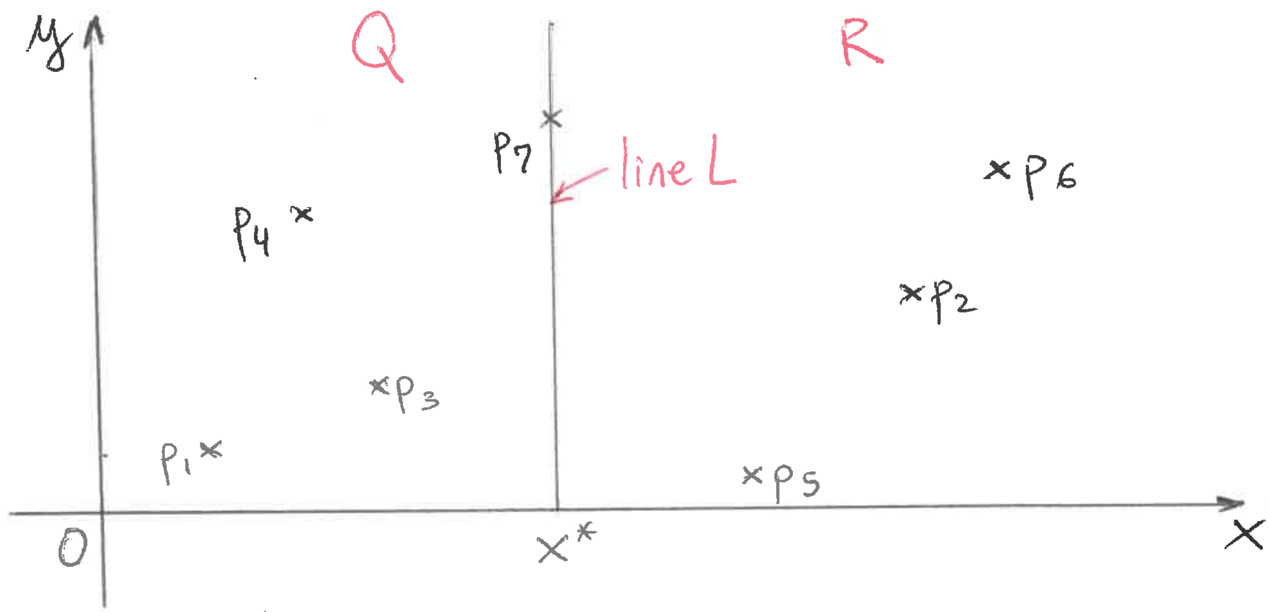
$$T(n) = T\left(\frac{n}{2}\right) + c \qquad c\text{-const}$$

$$c \text{ vs } n^{\log_2 1} = n^0 = 1$$

$c = \Theta(1) \Rightarrow$ case 2 of the Master Theorem

$$\boxed{T(n) = \Theta(\lg n)}$$

Finding the closest pair of points.



$n = 7$ points

$P = \langle p_1, p_2, p_3, p_4, p_5, p_6, p_7 \rangle$

- sort P by the x- and y- coordinate $\quad O(n \cdot \lg n)$

This operation is performed <u>only once</u>, before divide-and-conquer

$P_x = \langle \overbrace{p_1, p_4, p_3, p_7}^{Q}, \overbrace{p_5, p_2, p_6}^{R} \rangle$ sort points by x-coordinate

$P_y = \langle p_5, p_1, p_3, p_2, p_4, p_6, p_7 \rangle$ sort points by y-coordinate

$$n = \underbrace{\lceil \tfrac{n}{2} \rceil}_{\text{number of points in } Q} + \underbrace{\lfloor \tfrac{n}{2} \rfloor}_{\text{number of points in } R}$$

Divide step $O(n)$

$Q_x = \langle p_1, p_4, p_3, p_7 \rangle$

$R_x = \langle p_5, p_2, p_6 \rangle$

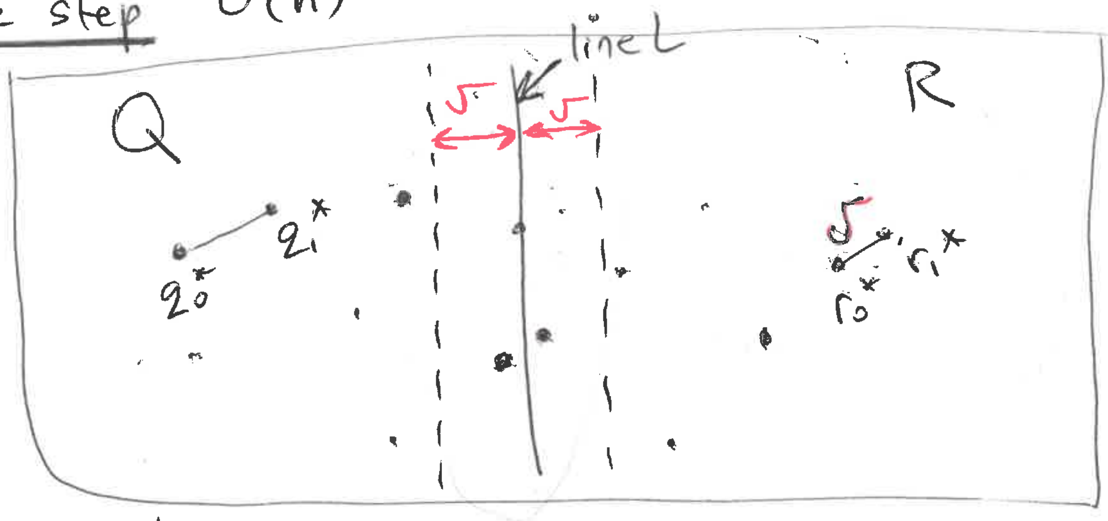
$Q_y = \langle p_1, p_3, p_4, p_7 \rangle$

$R_y = \langle p_5, p_2, p_6 \rangle$

- use a flag array to remember which points are in Q and R

1	2	3	4	5	6	7
0	1	0	0	1	1	0

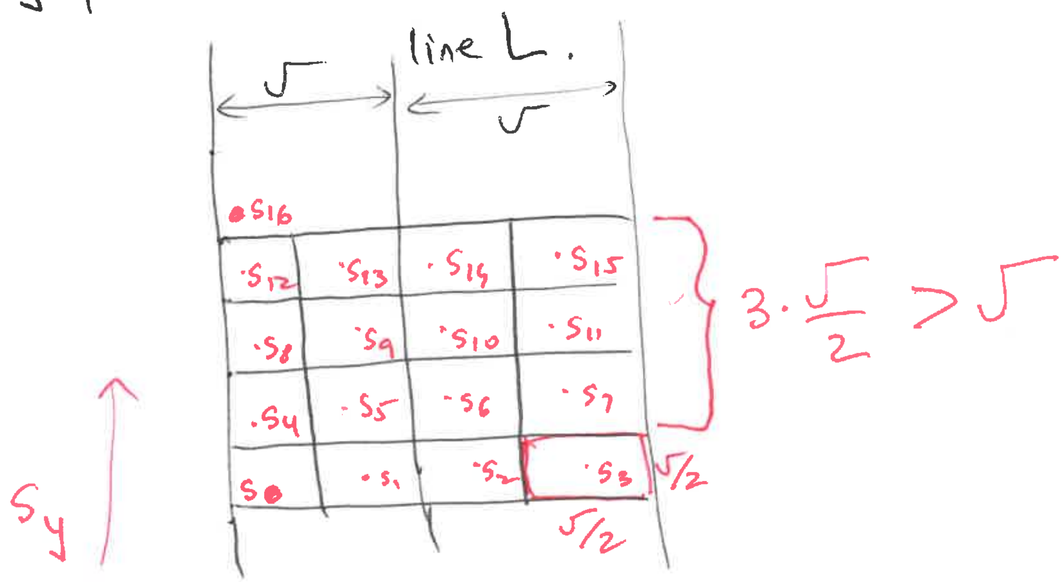
Combine step $O(n)$



set S of points

S_y – elements listed in increasing order of y-coordinate

- check if there are two points $s, s' \in S$ such that $d(s,s') < \delta$
- how many points do we need to check in S_y?



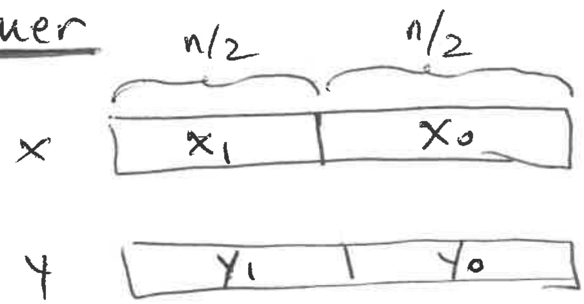
$S_y = \langle - - - s, s_1, s_2, s_3, - - s_{15} \rangle$

- check only the next 15 points in S_y
- the 16th point is separated by at least 3 rows, therefore the distance $> 3 \cdot \frac{\delta}{2} > \delta$

Integer Multiplication

$x \cdot y$ x and y have n bits

Divide-and-conquer



$$x \cdot y = (x_1 \cdot 2^{n/2} + x_0)(y_1 \cdot 2^{n/2} + y_0) =$$
$$= x_1 \cdot y_1 \cdot 2^n + (x_1 y_0 + x_0 y_1) 2^{n/2} + x_0 y_0$$

- Divide: $\Theta(1)$
- Conquer: $4 \cdot T(\frac{n}{2})$
- Combine: $\Theta(n)$

$$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + \Theta(n)$$
$$T(n) = 4 \cdot T\left(\frac{n}{2}\right) + cn$$
$$cn \text{ vs } n^{\log_2 4} = n^2$$
$$cn = O(n^{2-\varepsilon}) \quad \varepsilon = 1$$

Case 1 Master Theorem $\Rightarrow$ $\boxed{T(n) = \Theta(n^2)}$

Better (more efficient) solution

$$p = (x_1 + x_0)(y_1 + y_0) = x_1 y_1 + x_1 y_0 + x_0 y_1 + x_0 y_0$$
$$x \cdot y = x_1 \cdot y_1 \cdot 2^n + (p - x_0 y_0 - x_1 y_1) \cdot 2^{n/2} + x_0 y_0$$

This solution uses only 3 subproblems

- Divide: $\Theta(n)$
- Conquer: $3 \cdot T(\frac{n}{2})$
- Combine: $\Theta(n)$

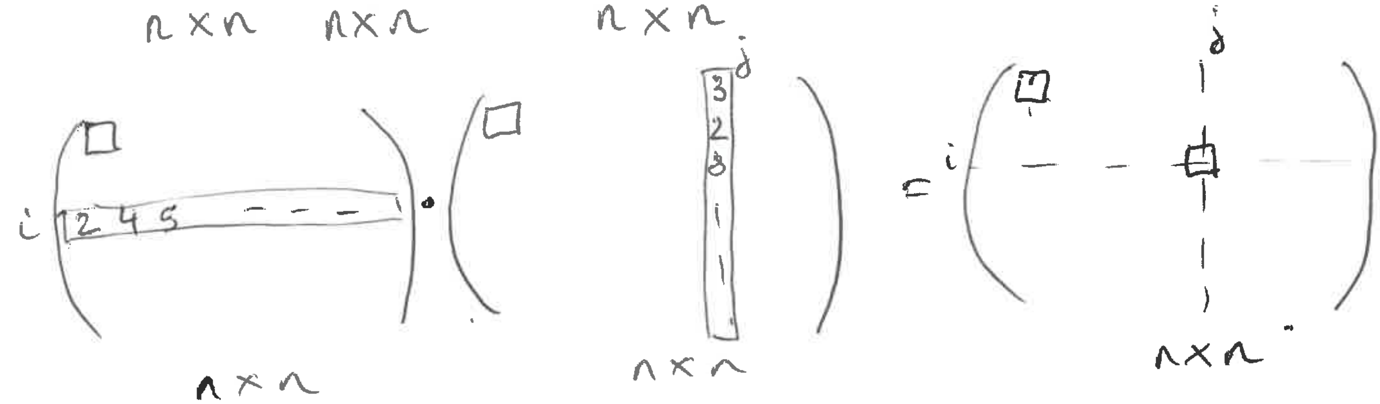
$$T(n) = 3 \cdot T\left(\frac{n}{2}\right) + \Theta(n)$$

Case 1 Master Theorem

$$\boxed{T(n) = \Theta(n^{1.59})}$$

Matrix product

$$\underset{n\times n}{A} \cdot \underset{n\times n}{B} = \underset{n\times n}{C}$$



$A \cdot B$ takes $\Theta(n^3)$

$A + B$ takes $\Theta(n^2)$

Divide-and-conquer

Multiply two matrices of size $n \times n$

$n=16$

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \cdot \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}b_{11}+a_{12}b_{21} & a_{11}b_{12}+a_{12}b_{22} \\ a_{21}b_{11}+a_{22}b_{21} & a_{21}b_{12}+a_{22}b_{22} \end{pmatrix}$$

Divide: $\Theta(1)$

Conquer: $8 \cdot T\left(\frac{n}{2}\right)$

Combine: $\Theta(n^2)$

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

$$T(n) = 8 \cdot T\left(\frac{n}{2}\right) + c \cdot n^2 \qquad c = \text{const}$$

$$cn^2 = O(n^{3-\varepsilon}) \qquad \varepsilon = 2$$

no improvement in RT!

case 1 of the Master Theorem $\Rightarrow$ $\boxed{T(n) = \Theta(n^3)}$

• *Note*: more efficient algorithm – Strassen's algorithm – with $\boxed{T(n) = \Theta(n^{2.8})}$