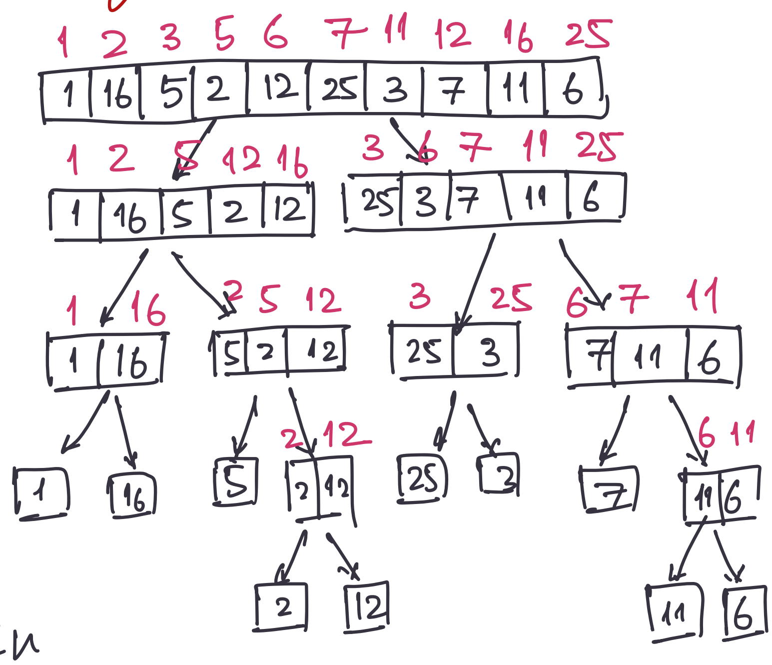
Divide and Conquer

→ Merge sort



$$T(n) = T(n_1) + T(n_2) + T_{merge}(n)$$

$$n_1 \simeq n_2$$

$\underline{n = 2^k}$

$$\hookrightarrow T(n) = 2\,T\left(\frac{n}{2}\right) + cn$$

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

$$= 4\,T\left(\frac{n}{4}\right) + 2\cdot c\cdot n$$

$$= 4\left[2\,T\left(\frac{n}{8}\right) + c\cdot\frac{n}{4}\right] + 2\cdot cn\,.$$

$$= 2^k\cdot T\left(\frac{n}{2^k}\right) + k\cdot c\cdot n\,.$$

$$= \underbrace{n\cdot 1} + \underbrace{c\cdot n\log n} = O(n\log n)$$

$T(1) = 1$

$k = \log_2 n$

$n = 17,\ 32$

$c\cdot \underline{n\log n}$

$C\cdot 32\cdot\log 32$

$\exists$ a const C_0, $\forall n \geq n_0$, $T(n) \leq C_0\cdot n\log n\,.$

Integer multiplication:

Given integers A and B, each of which is a n-bit number, we want to compute $C = A\cdot B$.

MSB

$$A = (a_{n-1}, a_{n-2}, \ldots, a_0)$$

LSB

$$B = (b_{n-1}, b_{n-2}, \ldots, b_0)$$

Trivial way

$O(n^2)$

$$\begin{array}{r} 52 \\ \times\ 13 \\ \hline 156 \\ 52\times \\ \hline 6 \end{array}$$

$$A = \sum_{i=0}^{n-1} a_i \cdot 2^i \qquad B = \sum_{i=0}^{n-1} b_i \cdot 2^i$$

$$A \cdot B = \left(\sum_{i=0}^{n-1} a_i \cdot 2^i\right)\left(\sum_{j=0}^{n-1} b_j \cdot 2^j\right) = \sum_{i,j} a_i b_j \cdot 2^{i+j} = \sum_k \Big(\sum_{\substack{i,j \\ i+j=k}} a_i b_j\Big) 2^k$$

$$A = A_0 + 2^{n/2} \cdot A_1$$
$$B = B_0 + 2^{n/2} \cdot B_1$$

A_0, A_1, B_0, B_1 are $\leq \frac{n}{2}$ bit numbers

$$\underbrace{a_0 \cdot 2^0 + \cdots + a_{\frac{n}{2}-1} \cdot 2^{\frac{n}{2}-1}}_{A_0} + a_{\frac{n}{2}} \cdot 2^{\frac{n}{2}} + a_{\frac{n}{2}+1} \cdot 2^{\frac{n}{2}+1} + \cdots + a_{n-1} \cdot 2^{n-1}$$

$A \cdot B$ multiplication of n-bit no.s

$$= (A_0 + 2^{n/2} \cdot A_1)(B_0 + 2^{n/2} \cdot B_1)$$
$$= A_0 B_0 + 2^{n/2}(A_1 B_0 + A_0 B_1) + 2^n (A_1 B_1).$$

$$a_{\frac{n}{2}} \cdot 2^{\frac{n}{2}} + a_{\frac{n}{2}+1} \cdot 2^{\frac{n}{2}+1} + \cdots + a_{n-1} \cdot 2^{n-1}$$
$$= 2^{n/2}\Big(\underbrace{a_{\frac{n}{2}} \cdot 2^0 + a_{\frac{n}{2}+1} \cdot 2 + \cdots + a_{n-1} \cdot 2^{\frac{n}{2}-1}}_{A_1}\Big)$$

$A_i B_j$ are mult. of $\leq \frac{n}{2}$ bit no.s

$$M(n) = \underset{\substack{\uparrow \\ 3}}{4} \cdot M\left(\frac{n}{2}\right) + c \cdot n = 4\left[4 M\left(\frac{n}{4}\right) + c \cdot \frac{n}{2}\right] + c \cdot n$$

$$= 4^k \cdot M\left(\frac{n}{2^k}\right) + c \cdot n \,(2^k - 1)$$

$$= O(n^2)$$

$n^{\log_2 3}$

4 mult $\left(\frac{n}{2}\right)$
+ 2-linear bit shifts
+ 4 additions.

$A_0 B_0$
$A_1 B_1$
$A_0 B_1$
$A_1 B_0$

$$\mathrm{I} - (A_0 + A_1) \cdot (B_0 + B_1)$$
$$\mathrm{II} - A_0 \cdot B_0$$
$$\mathrm{III} - A_1 B_1$$

$\to$ 3 mult,

$A_0 B_0 + A_1 B_1 + A_1 B_0 + A_0 B_1$

$\to$ Karatsuba's method.

$$AB = \mathrm{II} \cdot 2^0 + (\mathrm{I} - \mathrm{II} - \mathrm{III}) \cdot 2^{n/2} + 2^n \cdot \mathrm{III}.$$

$$M(n) = 3 \cdot M\left(\frac{n}{2}\right) + O(n) \rightarrow n^{\log_2 3}$$

$$\left.\begin{array}{l} A = A_0 + A_1 \cdot 2^{n/3} + A_2 \cdot 2^{2n/3} \\ B = B_0 + B_1 \cdot 2^{n/3} + B_2 \cdot 2^{2n/3} \end{array}\right\} \; 9 \text{ mult} \rightarrow n^{\log_3 9} \rightarrow n^2$$

$$\rightarrow \underbrace{O(n \log n \log\log n)}_{} \leftarrow O^*(n \log n) \quad 2^{\log^* n}$$

Discrete Fourier Transform. (Fast Fourier Transform)

Strassen's matrix multiplication.

$$[C]_{n \times n} = [A]_{n \times n} \times [B]_{n \times n} \qquad C_{ij} = \sum_{k=1}^{n} A_{ik} \cdot B_{kj}$$

$O(n^3)$ "trivial"

$$\begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} \begin{bmatrix} B_1 & B_2 \\ B_3 & B_4 \end{bmatrix} = \begin{bmatrix} A_1B_1 + A_2B_3 & A_1B_2 + A_2B_4 \\ A_3B_1 + A_4B_3 & A_3B_2 + A_4B_4 \end{bmatrix}$$

$$T(n) = 8T\left(\frac{n}{2}\right) + O(n^2) = O(n^3) \rightarrow n^{\log_2 7}$$

$M_1 = (A_{11} + A_{22})(B_{11} + B_{12})$

$M_2 = (A_{21} + A_{22}) B_{11}$

$M_3 = A_{11} (B_{12} - B_{22})$

$M_4 = A_{22} (B_{21} - B_{11})$

$M_5 = (A_{11} + A_{12}) \cdot B_{22}$

$M_6 = (A_{21} - A_{11}) \cdot (B_{11} + B_{12})$

$M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$

$$C = \begin{bmatrix} M_1 + M_4 - M_5 + M_7 & M_3 + M_5 \\ M_2 + M_4 & M_1 - M_2 + M_3 + M_6 \end{bmatrix}$$

Best known algo $\rightarrow$ $n^{2.38\ldots}$ [Alman-Vassilevska Williams]

[LeGall]

"Faster univariate polynomial mult.

$\Downarrow$

Faster integer mult."