From previous lectures:
Karatsuba: If $M(k)$ is the number of bit operations for k-bit x k-bit then

$$M(2k) \leq 3M(k) + 10k.$$

Proposition. Let $l \in \mathbb{Z}^+$. Then there is an algorithm which multiplies two 2^l bits numbers in at most $10\cdot(3^l - 2^l)$ bit operations.

Proof: By induction

$l=1$. We can multiply 2 bits number by 2-bits number in 10 operations (Long Mult.) ✓

Assume is true for l and prove for $l+1$.

$$M(2^{l+1}) \leq 3M(2^l) + 10\cdot 2^l \leq [\text{assumption}]$$
$$\leq 3\cdot 10(3^l - 2^l) + 10\cdot 2^l$$
$$= 10(3^{l+1} - 3\cdot 2^l + 2^l) = 10(3^{l+1} - 2^{l+1}).$$

Induction is complete. $\boxtimes$

Proposition: There is an algorithm which multiplies two k-bits numbers in at most $30(k^{\log_2(3)})$ bit operations.

Proof. Consider minimal l such that $k \leq 2^l$

$l = \lceil \log_2(k) \rceil < \log_2(k) + 1$

~~At~~ We add zeroes at the beginning of both numbers, so the number of bits in both numbers becomes 2^l.

Then by the previous proposition,

$$M(k) \leq M(2^l) \leq 10(3^l - 2^l) < 10 \cdot 3^l$$

$$< 10 \cdot 3^{\log_2(k)+1} = \cancel{30 \cdot k^{\log_3 2}}$$

$$= 30 \cdot (2^{\log_2 3})^{\log_2 k} = 30 \cdot k^{\log_2(3)}.$$

$\boxtimes$

§12.2 Big O notation.

Definition: Let $f(k), g(k)$ be two positive valued functions ove positive (integer) numbers.
We say that "$f(k)$ is $O(g(k))$" if:
There are positive numbers N, C such that

$$f(k) \leq C g(k) \text{ for all } k \geq N.$$

Examples:

(a) $10k^3 + 100k^2 + 1000k$ is $O(k^3)$

Indeed, $100k^2 \leq 10k^3$ if $k \geq 10$

$1000k \leq 10k^3$ if $k \geq 10$

In total, $10k^3 + 100k^2 + 1000k \leq 30k^3$ for $k \geq 10$.
We can choose $N = 10$, $C = 30$.

(b) $\frac{k^4}{1000}$ is not $O(k^3)$

because $\frac{k^4}{1000} \leq Ck^3 \iff k \leq 1000\,C$.

Therefore $\frac{k^4}{1000}$ is never $\leq Ck^3$ for an arbitrarily large k.

(c) 2^k is not $O(k^3)$

since $\lim_{k \to \infty} \frac{2^k}{k^3} = \infty$.

We use this notation in the following way: $f(k)$ is the number (maximal) of bit operations to perform some computations if the length of input is k bits. We want to estimate $f(k)$ by $g(k)$.

For multiplication we have: (for k bits $\times$ k bits).

Long multiplikation : $k^2 - k$ bit operations which is $O(k^2)$

Karatsuba : $30 \cdot (k^{\log_2(3)})$ bit operations which is $O(k^{\log_2(3)})$.

$\log_2(3) < 2$ therefore $O(k^{\log_2(3)})$ is also $O(k^2)$ but not vice versa. $\Rightarrow$ Karatsuba is more efficient than long mult. according to this measurement.

The fastest known multiplication algorithm is due to Schönhage - Strassen. It requires $O(k \cdot \log k \cdot \log(\log k))$ bit operations to multiply two k-digit numbers.

Definition. An algorithm is said to be of <u>polynomial time</u> if there exists positive a such that the number of bit operations required for the algorithm with the length of input $\le k$ is $O(k^a)$.