1.3. Multiplying integers

Thm 1.3.1 We can multiply two (binary) integers x, y with n digits in time $O(n)$ on an $O(\log n)$-bit RAM.

~~We can multiply two base 2^k integers x, y with n digits in time $O(n)$~~

Alg (sketch) W.l.o.g. $x, y \geq 0$.

W.l.o.g. $n = 2^k \cdot k$ with $k \geq 1$. ($\Rightarrow k = \Theta(\log n)$).

Write x, y in base 2^k:

$$x = \sum_{i=0}^{2^k - 1} a_i 2^{ki}, \quad y = \sum b_i 2^{ki},$$

$$0 \leq a_i, b_i < 2^k.$$

Let $R = \mathbb{C}$, $t = 2^{k+1}$, ζ_t any prim. t-th root of unity.

By Thm 1.2.1, we can compute

$$c_k := \sum_{\substack{i,j \\ i+j=k}} a_i b_j \quad \text{for } k = 0, \ldots, 2^{k+1} - 1$$

in time $O(2^k \cdot k) = O(n)$, assuming the operations in $\mathbb{C}$ can be done in time $O(1)$. It turns out that it suffices to do the computations in $\mathbb{C}$ with precision $O(n)$ (rounding intermediate results to $O(n)$ digits) and round the result c_k to the nearest integer. (These can be done in $O(1)$ on an $O(\log n)$-bit RAM.)

cf. Schönhage-Strassen: schnelle Mult. großer Zahlen

Now $x \cdot y = \sum_{k=0}^{2^{k+1}-2} c_k \cdot 2^{kk}$, where $0 \leq c_k \leq (k+1) \cdot 2^k \cdot 2^k \leq 2^{3k+1}$

c_k has at most 4 digits in base 2^k.

You can add these 2^{k+1} integers with $O(1)$ nonzero digits in time $O(2^{k+1})$.

Rmk 1.3.2 Harvey and van Hoeven recently showed that you can multiply two binary int. x, y with $< n$ digits in time $O(n \log n)$ on a multitape Turing machine. This is conjectured to be optimal.

[Their algorithm also uses FFT and several ingenious tricks!]

REFERENCE: Fast multiplication and its applications
Daniel J. Bernstein