NMAI 076 - Algebra 2 - spring semester 2024

Homework 3

Deadline 16.05.2024, 15:40

Exercise 1. (5 points) Let $(2, 0, 2, 0)$ be the modular representation of the polynomial f given by DFT_4 in $\mathbb{Z}_{17}$. Compute the polynomial f using the FFT.

Exercise 2. (10 points) Let $M = 2^{u2^{k-1}} + 1$, for some $u, k \geq 1$.

1. Show that $\omega = 2^u$ is a 2^k-th primitive root of $\mathbb{Z}_M$.
2. Show that 2^k is invertible modulo M.
3. Conclude that the FFT algorithm (for computing DFT and IDFT) also works for polynomials from $\mathbb{Z}_M[x]$.

Exercise 3. (15 points) The product $c = a \cdot b$ of two numbers given in binary expansion $a = \sum_{i=0}^{l} a_i 2^i$ and $b = \sum_{i=0}^{l} b_i 2^i$ can be computed as follows: take the polynomials $A(x) = \sum a_i x^i$ and $B(x) = \sum b_i x^i$ over $\mathbb{Z}[x]$ and compute their product $C(x) = \sum_{i=0}^{2l} c_i x^i$ using fast polynomial multiplication in a ring $\mathbb{Z}_p[x]$, for suitable $p > 2(l+1)$. Then clearly $c = C(2)$.

1. "Computing C, and then outputting the coefficients $(c_0, c_1, \ldots, c_{2l})$ returns a binary expansion of c in time $O(l \log(l))$." Why is this statement wrong?
2. How can we modify the above to get a correct algorithm? Give an upper bound to its running time, using that we can add or multiply two numbers in $\mathbb{Z}_p$ (given by their binary expansion) in time $c(p) = O(\log(p)^2)$.
3. Assuming that $p = O(l)$, how does this compare to the running time of "long multiplication" algorithm you know from school?

Remark Combining the observations in Exercise 2 and 3 we can obtain an algorithm for multiplying two integers in time $O(l \log(l) \log\log(l))$. This algorithm is called the *Schönhage-Strassen-Algorithm*, and was the fastest known until 2007.

The main idea of the algorithm is to work in $\mathbb{Z}_M$ with $M = 2^{u2^{k-1}} + 1$, and represent $a = \sum_{i=0}^{l} a_i 2^{ui}$ and $b = \sum_{i=0}^{l} b_i 2^{ui}$, with $a_i, b_i \in \{0, 1, \ldots, 2^u - 1\}$. We then use FFT in $\mathbb{Z}_M$ as described in Exercise 3 to compute $a \cdot b$. Moreover, we can recursively call this routine when compute the coordinate-wise product of $DFT(A) \cdot DFT(B)$.

The running time results from $\mathbb{Z}_M$ being very well-behaved with respect to binary representation of numbers: multiplication by powers of the unit root $\omega = 2^k$ can be simply done by bit-shifts; computing $x \mod M$ is also very fast.

The provably best running time of $O(l \log(l))$ was reached in 2021, by an algorithm of Harvey and van der Hoeven.