MTAT.03.286: Advanced Methods in Algorithms **University of Tartu**

Lecture 2

Instructor: Dr. Vitaly Skachek

Fast multiplication of polynomials

Algorithm: fast multiplication of polynomials

Input: Polynomials $A(x) = \sum_{i=0}^{m} a_i x^i$ and $B(x) = \sum_{i=0}^{m} b_i x^i$.

Output: Polynomial $C(x) = \sum_{i=0}^{m} c_i x^i$, such that $C(x) = A(x) \cdot B(x)$.

1. **Point selection.** Select distinct points $x_0, x_1, \cdots, x_{n-1}$, $n \geq 2m+1$.
2. **Evaluation.** Compute $A(x_0), A(x_1), \cdots, A(x_{n-1})$ and $B(x_0), B(x_1), \cdots, B(x_{n-1})$.
3. **Multiplication.** For $i = 0, 1, \cdots, n-1$, compute $C(x_i) = A(x_i) \cdot B(x_i)$.
4. **Interpolation.** Recover $C(x) = \sum_{i=0}^{n-1} c_i x^i$.

Definition 1 *For any positive integer n, the n-th* root of unity *is a solution (complex number) to the equation $z^n = 1$.*

Matrix representation

The following equation holds:

$$\begin{pmatrix} A(x_0) \\ A(x_1) \\ A(x_2) \\ \vdots \\ A(x_{n-1}) \end{pmatrix} = \begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1} \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{pmatrix}.$$

The matrix above belongs to the class of Vandermonde matrices. It is invertible, if all $x_0, x_1, \cdots, x_{n-1}$ are distinct.

We will choose $x_0, x_1, \cdots, x_{n-1}$ to be n distinct n-th roots of unity. For this selection of points,

the matrix above becomes

$$M_n(\omega) \triangleq \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{pmatrix} .$$

The matrix $M_n(\omega)$ is symmetric. Multiplication of a vector by such matrix is called *Fast Fourier Transform*.

The following theorem tells us that the inverse of the matrix $M_n(\omega)$ has a form very similar to the form of the matrix $M_n(\omega)$.

Theorem 2 *We have*

$$(M_n(\omega))^{-1} = \frac{1}{n} M_n(\omega^{-1}) .$$

It follows from the theorem that interpolation is equivalent to multiplication by $M_n(\omega^{-1})$, and is itself Fast Fourier Transform operation.

Proof. We compute the product $M_n(\omega) \cdot M_n(\omega^{-1})$. The elements on the main diagonal satisfy (for all $i = 0, 1, \cdots, n-1$):

$$\begin{aligned} \Big(M_n(\omega) \cdot M_n(\omega^{-1})\Big)_{i,i} &= \sum_{k=0}^{n-1} \omega^{ik} \cdot \omega^{-ik} \\ &= \sum_{k=0}^{n-1} 1 \\ &= n . \end{aligned}$$

For all $i = 0, 1, \cdots, n-1$ and $j = 0, 1, \cdots, n-1$, such that $i \neq j$, we obtain

$$\begin{aligned} \Big(M_n(\omega) \cdot M_n(\omega^{-1})\Big)_{i,j} &= 1 \cdot 1 + \omega^i \cdot \omega^{-j} + \omega^{2i} \cdot \omega^{-2j} + \cdots \omega^{i(n-1)} \cdot \omega^{-j(n-1)} \\ &= \sum_{k=0}^{n-1} \omega^{k(i-j)} \\ &= \frac{1 - \omega^{n(i-j)}}{1 - \omega^{i-j}} , \end{aligned} \tag{1}$$

where the last transition is obtained by summing up the elements in the gemetric series. Here, $1 - \omega^{i-j} \neq 0$ and $1 - \omega^{n(i-j)} = 0$. Therefore, the expression in (1) is equal to zero. □

The element in row j and column k of $M_n(\omega)$ is ω^{jk}, for $j = 0, 1, \cdots, n-1$, and for $k = 0, 1, \cdots, n-1$. Now, consider the elements in rows j and $j + n/2$ for $j = 0, 1, \cdots, n/2 - 1$, and in columns $2k$ and $2k + 1$ for $k = 0, 1, \cdots, n/2 - 1$.

	Column $2k$	Column $2k+1$
Row j	ω^{2jk}	ω^{j+2jk}
Row $j+n/2$	ω^{2jk}	$-\omega^{j+2jk}$

We obtain

$$\begin{pmatrix} A(\omega^0) \\ A(\omega^1) \\ A(\omega^2) \\ \vdots \\ A(\omega^{n-1}) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{pmatrix}$$

$$= \begin{pmatrix} M_{n/2}(\omega^2) \cdot \begin{pmatrix} a_0 \\ a_2 \\ \vdots \\ a_{n-2} \end{pmatrix} + \omega^j \cdot M_{n/2}(\omega^2) \cdot \begin{pmatrix} a_1 \\ a_3 \\ \vdots \\ a_{n-1} \end{pmatrix} \\ M_{n/2}(\omega^2) \cdot \begin{pmatrix} a_0 \\ a_2 \\ \vdots \\ a_{n-2} \end{pmatrix} - \omega^j \cdot M_{n/2}(\omega^2) \cdot \begin{pmatrix} a_1 \\ a_3 \\ \vdots \\ a_{n-1} \end{pmatrix} \end{pmatrix} .$$

Complexity analysis

Let n be a power of 2, and let $T(n)$ be time required to substitute n roots of unity into polynomial of degree $\leq n-1$. Then, the complexity of this operation is given by

$$T(n) = 2T(n/2) + O(n) = O(n \log n) .$$

The algorithm cocnsists of two evaluations of polynomials, $O(n \log n)$ each, of n multiplications, and of interpolation, which requires additional $O(n \log n)$ time. The total complexity is, therefore, $O(n \log n)$.