# Fast Fourier Transform

Selected Topics in Algorithms

ΑΛΜΑ, ΣΗΜΜΥ

# But what is FFT?

Nothing more than a clever computation of a function

- $FFT : \mathbb{C}^n \to \mathbb{C}^n, FFT(a) = V \cdot a$

$V$ is an invertible $n \times n$ matrix and we get

- $IFFT : \mathbb{C}^n \to \mathbb{C}^n, IFFT(y) = V^{-1} \cdot y$

Main use: Computing convolution

- $(a_0, a_1, \ldots, a_{n-1})$
- $(b_0, b_1, \ldots, b_{n-1})$

Return $(a_0 b_0, a_0 b_1 + a_1 b_0, \ldots, \sum_{i=0}^{n-1} a_i b_{n-1-i}, \ldots, \sum_{i=0}^{2(n-1)} a_i b_{2n-1-i})$

# Polynomials

## Representations of polynomials

Polynomial of degree $n-1$ can be described using

- the coefficients $(a_0, a_1, \ldots, a_{n-1})$, or
- $n$ evaluatons on $n$ different points

e.g., $A(x) = 3 + x + 2x^2$ can be uniquely defined by vector $(3, 1, 2)$ or by points $(-1, 4), (0, 3), (1, 6)$.

## Operations on polynomials

- evaluating polynomials
- adding polynomials
- multiplying polynomials

# Representations vs Time

When given as coefficient vector $(a_0, \ldots, a_{n-1})$

- evaluation: $O(n)$ operations, $A(x) = a_0 + x(a_1 + x(a_3 + \ldots) \ldots)$
- addition: $O(n)$ operations, $(a_0 + b_0)x^0 + \ldots + (a_{n-1} + b_{n-1})x^{n-1}$
- multiplication: k-th term is $\sum_{i=0}^{k-1} a_i b_{k-1-i}$, naively $O(n^2)$

When given $n$ evaluations $(x_0, y_0), \ldots, (x_{n-1}, y_{n-1})$

- evaluation: $O(n^2)$ using interpolation
- addition: $O(n)$, $(x_i, A(x_i)), (x_i, B(x_i)) \rightarrow (x_i, A(x_i) + B(x_i))$
- multiplication: $O(n)$, $(x_i, A(x_i)), (x_i, B(x_i)) \rightarrow (x_i, A(x_i)B(x_i))$

FFT: quick jump from vector representation to evaluations representation

# Evaluating Polynomials

Let $(a_0, \ldots, a_{n-1})$ represent a polynomial. How to get evaluation?

- pick $X = \{x_0, x_1, \ldots, x_{n-1}\}$
- compute for every $i$, $a_0 x_i^0 + a_1 x_i^1 + \ldots + a_{n-1} x_i^{n-1}$, or, all together,

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \ldots & x_{n-1}^{n-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{bmatrix}$$

# Divide and Conquer

Consider $A(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1}$ (wlog assume $n = 2^l$)

- Let $A_{even}(x) = a_0 + a_2 x + a_4 x^2 + \ldots + a_{n-2} x^{n/2-1}$
- Let $A_{odd}(x) = a_1 + a_3 x + a_5 x^2 + \ldots + a_{n-1} x^{n/2-1}$

$$A(x) = A_{even}(x^2) + x A_{odd}(x^2)$$

Idea: Recursively evaluate $A_{even}$ and $A_{odd}$ for points in $X^2 = \{x_0^2, \ldots, x_{n-1}^2\}$

- if evaluated, extra $O(|X|)$ to combine solutions

In total: $T(n, |X|) = 2T(n/2, |X^2|) + O(n + |X|)$

$|X| = n$ and we expect $|X^2| = n$. But $X$ is our choice + $\exists X : |X^2| = |X|/2$ (!!!)

In total: $T(n) = 2T(n/2) + O(n)$

# Picking $X$: $|X^2| = |X|/2$ in every recursion

Roots of unity

- square root: $\{1, -1\}$
- $1^{1/4}$: $\{1, -1, i, -i\}$
- $1^{1/8}$: $\{1, -1, i, -i, \pm\frac{\sqrt{2}}{2}(1+i), \pm\frac{\sqrt{2}}{2}(-1+i)\}$
- $1^{1/n}$: $\{e^{\frac{k}{n}2\pi i}\}_{k=1\ldots n}$ (we care for $n = 2^l$)

Key fact: The even $n$-th roots of unity coincide with the $n/2$-roots of unity

$$(e^{\frac{k}{n}2\pi i})^2 = e^{\frac{2k}{n}2\pi i} = e^{\frac{k}{n/2}2\pi i}$$

# But what is FFT? (Revisited)

Nothing more than a function : $\mathbb{C}^n \to \mathbb{C}^n$, $FFT(a) = V \cdot a$, where

$$V = \begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \dots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \dots & x_{n-1}^{n-1} \end{bmatrix}$$

with $x_k = e^{\frac{k}{n}2\pi i}$, computed using the mentioned divide and conquer idea

Interestingly $V$ is invertible with a very nice structure

$$V^{-1} = \frac{1}{n}\bar{V}$$

where $\bar{V}$ is the convex conjugate of $V$

# The Inverse FFT

Still, nothing more than a function : $\mathbb{C}^n \to \mathbb{C}^n$, $IFFT(y) = V^{-1} \cdot y$, where

$$V^{-1} = \frac{1}{n} \begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \dots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \dots & x_{n-1}^{n-1} \end{bmatrix}$$

with $x_k = e^{-\frac{k}{n}2\pi i}$, computed using the mentioned divide and conquer idea

<u>Proof of $\frac{1}{n}V\bar{V} = I \Leftrightarrow V\bar{V} = nI$:</u>

$$p_{jk} = \sum_{m=0}^{n-1} e^{m\frac{j}{n}2\pi i} e^{-k\frac{m}{n}2\pi i} = \sum_{m=0}^{n-1} e^{(j-k)\frac{m}{n}2\pi i} = \begin{cases} n, & j = k \\ \frac{\left(e^{(j-k)2\pi i/n}\right)^n - 1}{e^{(j-k)2\pi i/n} - 1} = 0, & j \neq k \end{cases}$$

# Applications

## Multiplication of polynomials in $O(n \log n)$

- Given the coefficients of $A(x)$ and $B(x)$ compute $A^* = FFT(A)$ and $B^* = FFT(B)$
- Compute $C^* = A^* B^*$ (pointwise)
- The coefficients of $C(x) = A(x)B(x)$ are simply $IFFT(C^*)$

## String matching in $O(n \log n)$

- Consider text 100110100111001100110001001110010100101100 10 . . .
- and a string 10011101100 of length $k$.
- Change all 0's to $-1$'s
- Reverse the string and add 0's to match text's length
- Let both strings define polynomials
- Coefficient $k$ for $x^m$ in their product implies that the string is matched in positions $m - k + 1$ to $m$