

# Transformación de dominio y la transformada rápida de Fourier

IIC2283

# Representación de un polinomio

La representación canónica de un polinomio  $p(x)$  no nulo es:

$$p(x) = \sum_{i=0}^{n-1} a_i x^i$$

donde  $n \geq 1$ ,  $a_{n-1} \neq 0$  y el grado de  $p(x)$  es  $n - 1$

Los coeficientes  $a_i$  de  $p(x)$  son números racionales

- Pero vamos a evaluar estos polinomios tanto sobre números racionales como sobre números reales y complejos

Representamos  $p(x)$  a través del vector  $(a_0, \dots, a_{n-1})$

- También podemos representar  $p(x)$  como un vector  $(a_0, \dots, a_{n-1}, 0, \dots, 0)$  donde cada términos  $x^i$  tiene coeficiente 0 si  $i \geq n$

# Suma de polinomios

La suma de dos polinomios  $(a_0, \dots, a_{n-1})$  y  $(b_0, \dots, b_{n-1})$  es un polinomio  $(c_0, \dots, c_{n-1})$  tal que:

$$c_i = a_i + b_i \quad \text{para } i \in \{0, \dots, n-1\}$$

Consideramos a la suma y multiplicación de números en  $\mathbb{C}$  como las operaciones básicas a contar.

- La suma de dos polinomios puede ser calculada en tiempo  $O(n)$

# Multiplicación de polinomios

La multiplicación de dos polinomios  $(a_0, \dots, a_{n-1})$  y  $(b_0, \dots, b_{n-1})$  es un polinomio  $(c_0, \dots, c_{2n-2})$  tal que:

$$c_i = \sum_{k, \ell \in \{0, \dots, n-1\} : k+\ell=i} a_k \cdot b_\ell \quad \text{para } i \in \{0, \dots, 2n-2\}$$

La multiplicación de dos polinomios puede ser calculada en tiempo  $O(n^2)$

► ¿Podemos realizar esta operación en un orden menor?

# Una representación alternativa de un polinomio

Un polinomio  $p(x)$  de grado  $n - 1$  se puede representar de manera única a través de un conjunto de  $n$  pares de puntos-valores:

$$\{(v_0, p(v_0)), (v_1, p(v_1)), \dots, (v_{n-1}, p(v_{n-1}))\},$$

suponiendo que  $v_i \neq v_j$  para  $i \neq j$

## Ejemplo

El polinomio  $p(x) = 1 + x + x^2$  es representado de manera única a través del conjunto de pares de puntos-valores:

$$\{(0, 1), (1, 3), (2, 7)\}$$

y también a través del conjunto de pares de puntos-valores:

$$\{(-2, 3), (0, 1), (5, 31)\}$$

# Una representación alternativa de un polinomio

Un polinomio  $p(x)$  de grado  $n - 1$  también se puede representar de manera única a través de un conjunto de pares de puntos-valores con  $m > n$  elementos:

$$\{(v_0, p(v_0)), \dots, (v_{n-1}, p(v_{n-1})), (v_n, p(v_n)), \dots, (v_{m-1}, p(v_{m-1}))\},$$

suponiendo que  $v_i \neq v_j$  para  $i \neq j$

## Ejemplo

El polinomio  $p(x) = 1 + 2x$  es representado de manera única a través del conjunto de pares de puntos-valores:

$$\{(0, 1), (1, 3), (2, 5)\}$$

y también a través del conjunto de pares de puntos-valores:

$$\{(-2, -3), (0, 1), (5, 11), (7, 15)\}$$

## ¿Por qué es útil la representación basada en puntos-valores?

Sean  $p(x)$  y  $q(x)$  polinomios de grado  $n - 1$  representados por:

$$\begin{aligned} &\{(v_0, p(v_0)), \dots, (v_{n-1}, p(v_{n-1}))\} \\ &\{(v_0, q(v_0)), \dots, (v_{n-1}, q(v_{n-1}))\} \end{aligned}$$

El polinomio  $r(x) = p(x) + q(x)$  es representado por:

$$\{(v_0, p(v_0) + q(v_0)), \dots, (v_{n-1}, p(v_{n-1}) + q(v_{n-1}))\}$$

# ¿Por qué es útil la representación basada en puntos-valores?

Suponga que se agrega  $n$  puntos a las representaciones de  $p(x)$  y  $q(x)$ :

$$\{(v_0, p(v_0)), \dots, (v_{n-1}, p(v_{n-1})), (v_n, p(v_n)), \dots, (v_{2n-1}, p(v_{2n-1}))\}$$

$$\{(v_0, q(v_0)), \dots, (v_{n-1}, q(v_{n-1})), (v_n, q(v_n)), \dots, (v_{2n-1}, q(v_{2n-1}))\}$$

El polinomio  $s(x) = p(x) \cdot q(x)$  es representado por:

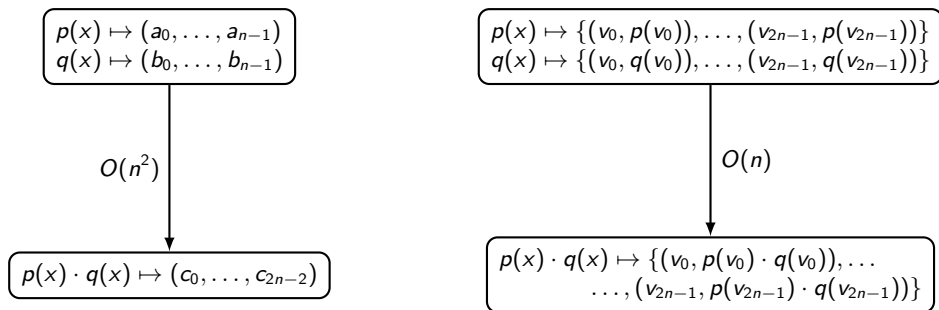
$$\{(v_0, p(v_0) \cdot q(v_0)), \dots, (v_{2n-1}, p(v_{2n-1}) \cdot q(v_{2n-1}))\}$$

¡Podemos sumar y multiplicar polinomios en tiempo  $O(n)$  si están representados por pares de puntos-valores!

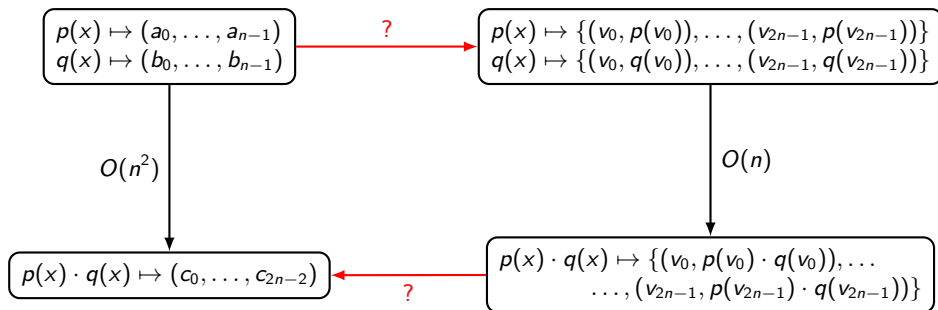
- Suponiendo que usamos los mismos puntos  $v_0, v_1, \dots, v_{2n-1}$



# La situación hasta ahora



# La situación hasta ahora



# De la representación canónica a la de puntos-valores

## Ejercicio

Dado un polinomio  $p(x)$  de grado  $n$  en su representación canónica y un punto  $v$ , de un algoritmo que calcule  $p(v)$  en tiempo  $O(n)$

Podemos entonces pasar de la representación canónica a la de puntos-valores en tiempo  $O(n^2)$

# De la representación puntos-valores a la canónica

Sea  $p(x)$  un polinomio de grado  $n - 1$  dado por una representación punto-valores:

$$\{(v_0, p(v_0)), \dots, (v_{n-1}, p(v_{n-1})), (v_n, p(v_n)), \dots, (v_{m-1}, p(v_{m-1}))\},$$

donde  $m \geq n$

Podemos pasar a la representación canónica de  $p(x)$  utilizando fórmula de Lagrange:

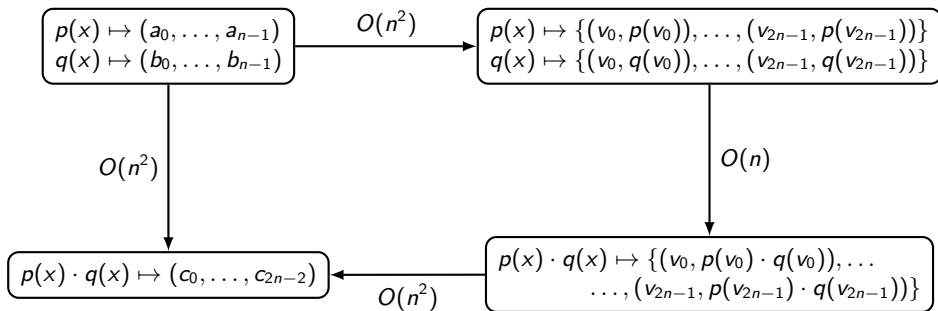
$$p(x) = \sum_{i=0}^{m-1} p(v_i) \cdot \left( \prod_{j \in \{0, \dots, m-1\} : j \neq i} \frac{(x - v_j)}{(v_i - v_j)} \right)$$

# De la representación puntos-valores a la canónica

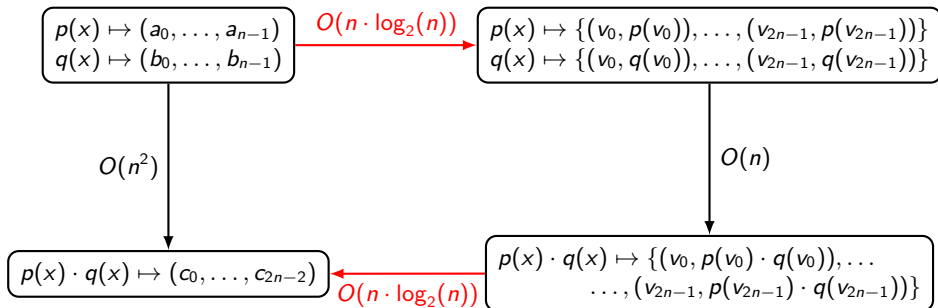
## Ejercicio

Dado un polinomio  $p(x)$  representando por el conjunto de punto-valores  $\{(v_0, p(v_0)), \dots, (v_{2n-1}, p(v_{2n-1}))\}$ , muestre que la fórmula de Lagrange permite construir la forma canónica de  $p(x)$  en tiempo  $O(n^2)$

# Todavía no tenemos un algoritmo más rápido para multiplicar polinomios



# La solución: la transformada rápida de Fourier



# La solución: la transformada rápida de Fourier

La transformada rápida de Fourier nos va a permitir entonces calcular la multiplicación de dos polinomios de grado  $n - 1$  en tiempo  $O(n \cdot \log_2(n))$

- ▶ La idea clave es cómo elegir los puntos  $v_0, \dots, v_{2n-1}$  cuando se calcula la representación como punto-valores de un polinomio de grado  $n - 1$

Los números complejos y las raíces de la unidad juegan un papel fundamental en la definición de la transformada rápida de Fourier.



# La fórmula de Euler

## Teorema

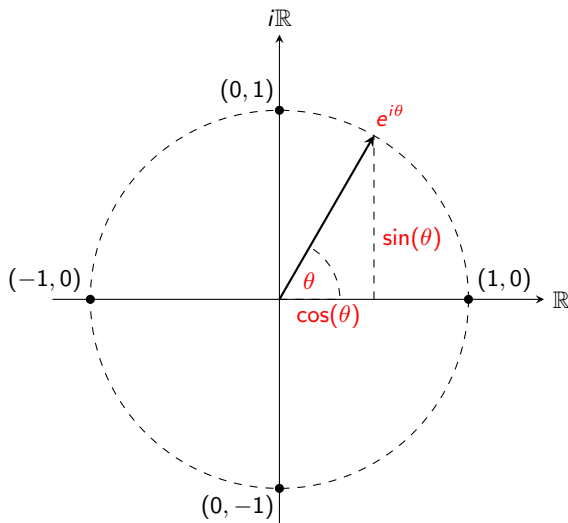
*Para todo número real  $x$ :*

$$e^{ix} = \cos(x) + i \sin(x)$$

Podemos representar entonces a  $e^{i\theta}$  como un vector  $(\cos(\theta), \sin(\theta))$  en el plano complejo.

►  $e^{i\theta}$  es un vector unitario:  $\|e^{i\theta}\| = \cos^2(\theta) + \sin^2(\theta) = 1$

# La fórmula de Euler: interpretación geométrica



# La fórmula de Euler: las raíces de la unidad

Dado  $n \geq 1$ , queremos encontrar las  $n$  raíces del polinomio  $p(x) = x^n - 1$

- ▶ Sabemos que este polinomio tiene  $n$  raíces en los números complejos
- ▶ Llamamos a estos elementos las  $n$ -raíces de la unidad

El componente básico para definir las  $n$ -raíces de la unidad:

$$\omega_n = e^{\frac{2\pi i}{n}}$$

# La fórmula de Euler: las raíces de la unidad

Las  $n$ -raíces de la unidad son  $\omega_n^0, \omega_n^1, \omega_n^2, \dots, \omega_n^{n-1}$

Si  $k \in \{0, \dots, n-1\}$ , tenemos que:

$$\begin{aligned}(\omega_n^k)^n &= ((e^{\frac{2\pi i}{n}})^k)^n \\&= ((e^{\frac{2\pi i}{n}})^n)^k \\&= (e^{2\pi i})^k \\&= (\cos(2\pi) + i \sin(2\pi))^k \\&= 1^k \\&= 1\end{aligned}$$

# La fórmula de Euler: las raíces de la unidad

Además, si  $0 \leq k \leq \ell \leq n-1$ , entonces:

$$\begin{aligned}\omega_n^k = \omega_n^\ell &\Rightarrow \left(e^{\frac{2\pi i}{n}}\right)^k = \left(e^{\frac{2\pi i}{n}}\right)^\ell \\&\Rightarrow \left(e^{\frac{2\pi i}{n}}\right)^{\ell-k} = 1 \\&\Rightarrow \left(e^{\frac{2\pi(\ell-k)i}{n}}\right) = 1 \\&\Rightarrow \cos\left(\frac{2\pi(\ell-k)}{n}\right) + i \sin\left(\frac{2\pi(\ell-k)}{n}\right) = 1 \\&\Rightarrow \cos\left(\frac{2\pi(\ell-k)}{n}\right) = 1 \\&\Rightarrow \frac{\ell-k}{n} = 0 \qquad \text{puesto que } 0 \leq \frac{\ell-k}{n} \leq \frac{n-1}{n} \\&\Rightarrow \ell = k\end{aligned}$$

Por lo tanto:  $\omega_n^0, \dots, \omega_n^{n-1}$  son elementos distintos

# Raíces de la unidad: ejemplos

¿Cuáles son las raíces del polinomio  $x^4 - 1$ ?

- Considerando  $\omega_4 = e^{\frac{2\pi i}{4}} = e^{\frac{\pi i}{2}}$ , tenemos que las 4-raíces de la unidad son:

$$\omega_4^0 = 1$$

$$\omega_4^1 = e^{\frac{\pi i}{2}} = \cos\left(\frac{\pi}{2}\right) + i \sin\left(\frac{\pi}{2}\right) = i$$

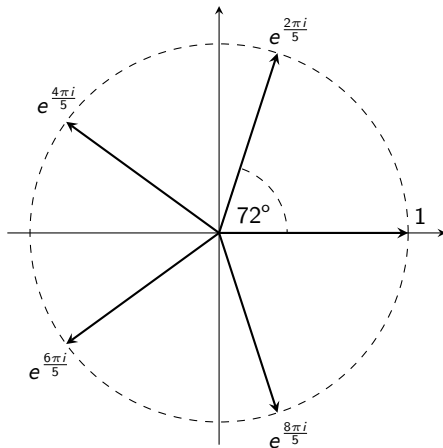
$$\omega_4^2 = (e^{\frac{\pi i}{2}})^2 = e^{\pi i} = \cos(\pi) + i \sin(\pi) = -1$$

$$\omega_4^3 = (e^{\frac{\pi i}{2}})^3 = e^{\frac{3\pi i}{2}} = \cos\left(\frac{3\pi}{2}\right) + i \sin\left(\frac{3\pi}{2}\right) = -i$$

# Raíces de la unidad: otro ejemplo

¿Cuáles son las raíces del polinomio  $x^5 - 1$ ? Considerando  $\omega_5 = e^{\frac{2\pi i}{5}}$ , tenemos que las 5-raíces de la unidad son 1,  $e^{\frac{2\pi i}{5}}$ ,  $e^{\frac{4\pi i}{5}}$ ,  $e^{\frac{6\pi i}{5}}$  y  $e^{\frac{8\pi i}{5}}$

Representación  
geométrica:



# La transformada discreta de Fourier

Dado:  $n \geq 2$  y un polinomio  $p(x) = \sum_{i=0}^{n-1} a_i x^i$

## Definition

La transformada discreta de Fourier (DFT) de  $p(x)$  se define como:

$$[p(\omega_n^0), p(\omega_n^1), \dots, p(\omega_n^{n-1})]$$



# ¿Cómo podemos calcular DFT de manera eficiente?

Representamos  $p(x)$  a través del vector  $\bar{a} = (a_0, \dots, a_{n-1})$

El problema a resolver es calcular de manera eficiente la transformada discreta de Fourier de  $p(x)$ , la cual denotamos como **DFT**( $\bar{a}$ )

- Definimos  $y_k = p(\omega_n^k)$  para cada  $k \in \{0, \dots, n-1\}$ , de manera tal que queremos calcular **DFT**( $\bar{a}$ ) =  $[y_0, \dots, y_{n-1}]$

## Ejercicio

Muestre que **DFT**( $\bar{a}$ ) puede ser calculada en tiempo  $O(n^2)$

# ¿Cómo podemos calcular DFT de manera eficiente?

Suponiendo que  $n$  potencia de 2, calculamos  $\mathbf{DFT}(\bar{a})$  realizando los siguientes pasos:

- (1) Calcular  $\mathbf{DFT}(\bar{a}_0)$  y  $\mathbf{DFT}(\bar{a}_1)$  para dos vectores  $\bar{a}_0$  y  $\bar{a}_1$  de largo  $\frac{n}{2}$
- (2) Combinar  $\mathbf{DFT}(\bar{a}_0)$  y  $\mathbf{DFT}(\bar{a}_1)$  para obtener  $\mathbf{DFT}(\bar{a})$

Es fundamental que el paso (2) sea realizado en tiempo  $O(n)$

## Ejercicio

Muestre que si el paso (2) es realizado en  $c \cdot n$  operaciones, entonces  $\mathbf{DFT}(\bar{a})$  puede ser calculada en tiempo  $O(n \cdot \log_2(n))$

# La descomposición recursiva

Tenemos que:

$$\begin{aligned}p(x) &= \sum_{i=0}^{n-1} a_i x^i \\&= \sum_{i=0}^{\frac{n}{2}-1} a_{2i} x^{2i} + \sum_{i=0}^{\frac{n}{2}-1} a_{2i+1} x^{2i+1} \\&= \sum_{i=0}^{\frac{n}{2}-1} a_{2i} x^{2i} + x \cdot \sum_{i=0}^{\frac{n}{2}-1} a_{2i+1} x^{2i}\end{aligned}$$

Definimos los polinomios:

$$\begin{aligned}q(z) &= \sum_{i=0}^{\frac{n}{2}-1} a_{2i} z^i \\r(z) &= \sum_{i=0}^{\frac{n}{2}-1} a_{2i+1} z^i\end{aligned}$$

# La descomposición recursiva

Tenemos entonces que  $p(x) = q(x^2) + x \cdot r(x^2)$

Para calcular  $[p(\omega_n^0), p(\omega_n^1), \dots, p(\omega_n^{n-1})]$ , tenemos entonces que calcular:

$$\begin{aligned} &[q((\omega_n^0)^2), q((\omega_n^1)^2), \dots, q((\omega_n^{n-1})^2)] \\ &[r((\omega_n^0)^2), r((\omega_n^1)^2), \dots, r((\omega_n^{n-1})^2)] \end{aligned}$$

Pero si  $k \in \{0, \dots, \frac{n}{2} - 1\}$ , entonces tenemos que:

$$\begin{aligned} (\omega_n^{\frac{n}{2}+k})^2 &= \omega_n^{n+2k} \\ &= \omega_n^n \cdot \omega_n^{2k} \\ &= (e^{\frac{2\pi i}{n}})^n \cdot (\omega_n^k)^2 \\ &= (e^{2\pi i}) \cdot (\omega_n^k)^2 \\ &= 1 \cdot (\omega_n^k)^2 \\ &= (\omega_n^k)^2 \end{aligned}$$

# La descomposición recursiva

Por lo tanto para calcular  $[p(\omega_n^0), p(\omega_n^1), \dots, p(\omega_n^{n-1})]$ , basta con calcular:

$$[q((\omega_n^0)^2), q((\omega_n^1)^2), \dots, q((\omega_n^{\frac{n}{2}-1})^2)]$$

$$[r((\omega_n^0)^2), r((\omega_n^1)^2), \dots, r((\omega_n^{\frac{n}{2}-1})^2)]$$

La pregunta clave: ¿si  $\omega_n^0, \omega_n^1, \dots, \omega_n^{n-1}$  son las  $n$ -raíces de la unidad, quiénes son  $(\omega_n^0)^2, (\omega_n^1)^2, \dots, (\omega_n^{\frac{n}{2}-1})^2$ ?

- Dado que  $q$  y  $r$  son polinomios de grado  $\frac{n}{2} - 1$ , ¿quiénes deberían ser  $(\omega_n^0)^2, (\omega_n^1)^2, \dots, (\omega_n^{\frac{n}{2}-1})^2$  para poder realizar las llamadas recursivas a **DFT**?

# Respondiendo la pregunta: una regla de simplificación

Tenemos que  $\omega_{n \cdot \ell}^{k \cdot \ell} = \omega_n^k$  para  $\ell > 0$ , puesto que:

$$\begin{aligned}\omega_{n \cdot \ell}^{k \cdot \ell} &= \left( e^{\frac{2\pi i}{n \cdot \ell}} \right)^{k \cdot \ell} \\ &= \left( e^{\frac{2\pi i \cdot \ell}{n \cdot \ell}} \right)^k \\ &= \left( e^{\frac{2\pi i}{n}} \right)^k \\ &= \omega_n^k\end{aligned}$$

# La respuesta a la pregunta

## Lema

Si  $n \geq 2$  es par, entonces  $(\omega_n^0)^2, (\omega_n^1)^2, \dots, (\omega_n^{\frac{n}{2}-1})^2$  son las  $\frac{n}{2}$ -raíces de la unidad (vale decir, son las raíces del polinomio  $x^{\frac{n}{2}} - 1$ ).

**Demostración:** dado  $k \in \{0, \dots, \frac{n}{2} - 1\}$ , se tiene que

$$\begin{aligned}(\omega_n^k)^2 &= \omega_n^{k \cdot 2} \\ &= \omega_{\frac{n}{2}}^{k \cdot 2} \\ &= \omega_{\frac{n}{2}}^k\end{aligned}\quad \text{por la regla de simplificación}$$

Por lo tanto,  $(\omega_n^0)^2, (\omega_n^1)^2, \dots, (\omega_n^{\frac{n}{2}-1})^2$  son las  $\frac{n}{2}$ -raíces de la unidad

► Puesto que  $(\omega_n^0)^2, (\omega_n^1)^2, \dots, (\omega_n^{\frac{n}{2}-1})^2 = \omega_{\frac{n}{2}}^0, \omega_{\frac{n}{2}}^1, \dots, \omega_{\frac{n}{2}}^{\frac{n}{2}-1}$



# La descomposición recursiva (continuación)

Recuerde que  $\bar{a} = (a_0, \dots, a_{n-1})$ , y defina:

$$\bar{a}_0 = (a_0, a_2, \dots, a_{n-2})$$

$$\bar{a}_1 = (a_1, a_3, \dots, a_{n-1})$$

De los resultados anteriores concluimos que para calcular **DFT**( $\bar{a}$ ), primero tenemos que calcular **DFT**( $\bar{a}_0$ ) y **DFT**( $\bar{a}_1$ )

¿Cómo se construye **DFT**( $\bar{a}$ ) a partir de **DFT**( $\bar{a}_0$ ) y **DFT**( $\bar{a}_1$ )?



# Construyendo $\mathbf{DFT}(\bar{a})$ a partir de $\mathbf{DFT}(\bar{a}_0)$ y $\mathbf{DFT}(\bar{a}_1)$

Sea:

$$\mathbf{DFT}(\bar{a}_0) = [y_{0,0}, y_{0,1}, \dots, y_{0,\frac{n}{2}-1}]$$

$$\mathbf{DFT}(\bar{a}_1) = [y_{1,0}, y_{1,1}, \dots, y_{1,\frac{n}{2}-1}]$$

Para  $k \in \{0, \dots, \frac{n}{2} - 1\}$  tenemos que:

$$\begin{aligned} y_k &= p(\omega_n^k) \\ &= q((\omega_n^k)^2) + \omega_n^k \cdot r((\omega_n^k)^2) \\ &= q(\omega_{\frac{n}{2}}^k) + \omega_n^k \cdot r(\omega_{\frac{n}{2}}^k) \\ &= y_{0,k} + \omega_n^k \cdot y_{1,k} \end{aligned}$$

# Construyendo $\mathbf{DFT}(\bar{a})$ a partir de $\mathbf{DFT}(\bar{a}_0)$ y $\mathbf{DFT}(\bar{a}_1)$

Además, para  $k \in \{0, \dots, \frac{n}{2} - 1\}$  tenemos que:

$$\begin{aligned}y_{\frac{n}{2}+k} &= p(\omega_n^{\frac{n}{2}+k}) \\&= q((\omega_n^{\frac{n}{2}+k})^2) + \omega_n^{\frac{n}{2}+k} \cdot r((\omega_n^{\frac{n}{2}+k})^2) \\&= q(\omega_n^{n+2k}) + \omega_n^{\frac{n}{2}} \cdot \omega_n^k \cdot r(\omega_n^{n+2k}) \\&= q(\omega_n^n \cdot \omega_n^{2k}) + (e^{\frac{2\pi i}{n}})^{\frac{n}{2}} \cdot \omega_n^k \cdot r(\omega_n^n \cdot \omega_n^{2k}) \\&= q(1 \cdot \omega_n^{2k}) + e^{\pi i} \cdot \omega_n^k \cdot r(1 \cdot \omega_n^{2k}) \\&= q(\omega_n^{2k}) - \omega_n^k \cdot r(\omega_n^{2k}) \\&= q(\omega_n^{\frac{k}{2}}) - \omega_n^k \cdot r(\omega_n^{\frac{k}{2}}) \\&= y_{0,k} - \omega_n^k \cdot y_{1,k}\end{aligned}$$

## Construyendo $\mathbf{DFT}(\bar{a})$ a partir de $\mathbf{DFT}(\bar{a}_0)$ y $\mathbf{DFT}(\bar{a}_1)$

Resumiendo, para  $k \in \{0, \dots, \frac{n}{2} - 1\}$  tenemos que:

$$\begin{aligned}y_k &= y_{0,k} + \omega_n^k \cdot y_{1,k} \\ y_{\frac{n}{2}+k} &= y_{0,k} - \omega_n^k \cdot y_{1,k}\end{aligned}$$

Para tener un algoritmo recursivo para calcular **DFT** sólo nos falta el caso base.

- Consideramos  $n = 2$  y un polinomio  $p(x) = a_0 + a_1x$

## Construyendo $\mathbf{DFT}(\bar{a})$ a partir de $\mathbf{DFT}(\bar{a}_0)$ y $\mathbf{DFT}(\bar{a}_1)$

Tenemos que  $\omega_2 = e^{\frac{2\pi i}{2}} = e^{\pi i} = -1$

► Por lo tanto:  $\omega_2^0 = 1$  y  $\omega_2^1 = -1$

Concluimos que:

$$p(\omega_2^0) = a_0 + a_1$$

$$p(\omega_2^1) = a_0 - a_1$$

# Un algoritmo recursivo eficiente para **DFT**

La entrada del algoritmo es un polinomio  $p(x) = \sum_{i=0}^{n-1} a_i x^i$

- ▶ Este polinomio es representado por el vector  $\bar{a} = (a_0, \dots, a_{n-1})$

Suponemos además que  $n \geq 2$  y  $n$  es una potencia de 2

El algoritmo es llamado la transformada rápida de Fourier (FFT)

- ▶ Fue propuesto por Cooley & Tukey

# Un algoritmo recursivo eficiente para **DFT**

**FFT**( $\bar{a}$ )

$(a_0, \dots, a_{n-1}) := \bar{a}$

**if**  $n = 2$  **then**

$y_0 = a_0 + a_1$

$y_1 = a_0 - a_1$

**return**  $[y_0, y_1]$

**else**

$\bar{a}_0 := (a_0, \dots, a_{n-2})$

$\bar{a}_1 := (a_1, \dots, a_{n-1})$

$[y_{0,0}, \dots, y_{0,\frac{n}{2}-1}] := \mathbf{FFT}(\bar{a}_0)$

$[y_{1,0}, \dots, y_{1,\frac{n}{2}-1}] := \mathbf{FFT}(\bar{a}_1)$

$\omega_n := e^{\frac{2\pi i}{n}}$

$\alpha := 1$

**for**  $k := 0$  **to**  $\frac{n}{2} - 1$  **do**

$y_k := y_{0,k} + \alpha \cdot y_{1,k}$

$y_{\frac{n}{2}+k} := y_{0,k} - \alpha \cdot y_{1,k}$

$\alpha := \alpha \cdot \omega_n$

**return**  $[y_0, \dots, y_{n-1}]$

## Ejercicios

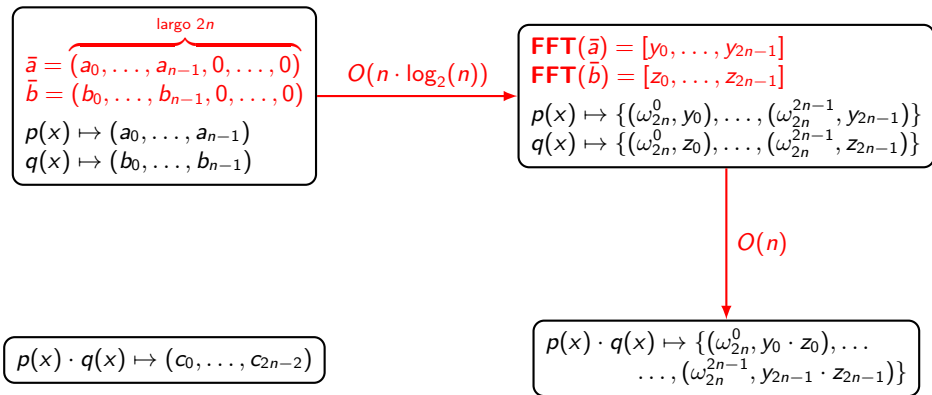
1. Demuestre que **FFT** funciona en tiempo  $O(n \cdot \log_2(n))$ , donde  $n$  es el grado del polinomio de entrada.
2. Sea  $p(x)$  un polinomio representado como  $\bar{a} = (a_0, \dots, a_{n-1})$ , donde  $n$  **no** es una potencia de 2.

Para evaluar  $p(x)$  en  $n$  puntos haga lo siguiente:

- (i) Defina  $\bar{b} = (a_0, \dots, a_{n-1}, 0, \dots, 0)$  de largo  $m = 2^{\lceil \log_2(n) \rceil}$
- (ii) Calcule **FFT**( $\bar{b}$ ) =  $[y_0, \dots, y_{m-1}]$ , y retorne  $[y_0, \dots, y_{n-1}]$

Note que este algoritmo retorna  $[p(\omega_m^0), \dots, p(\omega_m^{n-1})]$  en lugar de **DFT**( $\bar{a}$ ), y demuestre que funciona en tiempo  $O(n \cdot \log_2(n))$

# La nueva situación con el algoritmo **FFT**



Todavía nos falta un algoritmo para calcular la inversa de la transformada discreta de Fourier.



# La transformada discreta de Fourier como matriz

Suponga nuevamente que  $p(x) = \sum_{i=0}^{n-1} a_i x^i$  con  $n \geq 2$

► Para cada  $k \in \{0, \dots, n-1\}$ :  $y_k = p(\omega_n^k) = \sum_{i=0}^{n-1} a_i (\omega_n^k)^i = \sum_{i=0}^{n-1} a_i \cdot \omega_n^{i \cdot k}$

La transformada discreta de Fourier puede ser representada entonces de la siguiente forma en notación matricial:

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_n & \omega_n^2 & \cdots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \cdots & \omega_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \cdots & \omega_n^{(n-1)(n-1)} \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

# La transformada discreta de Fourier como matriz

Definimos la matrix  $M_n$  como:

$$M_n = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega_n & \omega_n^2 & \cdots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \cdots & \omega_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \cdots & \omega_n^{(n-1)(n-1)} \end{pmatrix}$$

Esta matriz va a ser muy útil al momento de definir la inversa de la transformada discreta de Fourier.

# Las matrices $M_n$ y $M_n^*$

Para cada  $i, j \in \{1, \dots, n\}$ , tenemos que  $M_n[i, j] = (\omega_n^{i-1})^{j-1} = \omega_n^{(i-1) \cdot (j-1)}$

- Por lo tanto  $M_n[i, j] = M_n[j, i]$ , de lo cual se deduce que  $M_n$  es una matriz simétrica

Recuerde que la matriz adjunta  $A^*$  de una matriz  $A$  se define como  $A^*[i, j] = \overline{A[j, i]}$

- Donde  $\overline{b + ci} = b - ci$  es el conjugado del número complejo  $b + ci$

Para calcular  $M_n^*$  necesitamos calcular el conjugado de  $\omega_n^k$

# El conjugado de $\omega_n^k$

Tenemos que:

$$\begin{aligned}\overline{(\omega_n^k)} &= \overline{(e^{\frac{2\pi i}{n}})^k} \\&= \overline{e^{\frac{2\pi ki}{n}}} \\&= \overline{\cos\left(\frac{2\pi k}{n}\right) + i \sin\left(\frac{2\pi k}{n}\right)} \\&= \cos\left(\frac{2\pi k}{n}\right) - i \sin\left(\frac{2\pi k}{n}\right) \\&= \cos\left(-\frac{2\pi k}{n}\right) + i \sin\left(-\frac{2\pi k}{n}\right) \\&= e^{\frac{-2\pi ki}{n}} \\&= (e^{\frac{2\pi i}{n}})^{-k} \\&= \omega_n^{-k}\end{aligned}$$

# Las matrices $M_n$ y $M_n^*$ (continuación)

Tenemos entonces que:

$$\begin{aligned}M_n^*[i, j] &= \overline{M_n[j, i]} \\&= \overline{(\omega_n^{j-1})^{i-1}} \\&= \omega_n^{(i-1) \cdot (j-1)} \\&= \omega_n^{-(i-1) \cdot (j-1)}\end{aligned}$$

Vamos a utilizar esta propiedad para calcular  $M_n^* \cdot M_n$

# Las matrices $M_n$ y $M_n^*$ (continuación)

## Lema

Sea  $I_n$  la matriz identidad de tamaño  $n$ . Entonces  $M_n^* \cdot M_n = n \cdot I_n$

**Demostración:** Sea  $A = M_n^* \cdot M_n$

Para  $i \in \{1, \dots, n\}$ , tenemos que:

$$\begin{aligned} A[i, i] &= \sum_{k=1}^n M_n^*[i, k] \cdot M_n[k, i] \\ &= \sum_{k=1}^n \omega_n^{-(i-1) \cdot (k-1)} \cdot \omega_n^{(k-1) \cdot (i-1)} \\ &= \sum_{k=1}^n 1 \\ &= n \end{aligned}$$

# Las matrices $M_n$ y $M_n^*$ (continuación)

Para  $i, j \in \{1, \dots, n\}$  con  $i \neq j$ , tenemos que:

$$\begin{aligned} A[i, j] &= \sum_{k=1}^n M_n^*[i, k] \cdot M_n[k, j] \\ &= \sum_{k=1}^n \omega_n^{-(i-1) \cdot (k-1)} \cdot \omega_n^{(k-1) \cdot (j-1)} \\ &= \sum_{k=1}^n (\omega_n^{-(i-1) + (j-1)})^{k-1} \\ &= \sum_{k=1}^n (\omega_n^{j-i})^{k-1} \\ &= \frac{(\omega_n^{j-i})^n - 1}{\omega_n^{j-i} - 1} \quad \text{ya que } \omega_n^{j-i} \neq 1 \text{ (} j-i \neq 0, j-i \neq n \text{ y } j-i \neq -n) \\ &= \frac{(\omega_n^n)^{j-i} - 1}{\omega_n^{j-i} - 1} = \frac{1^{j-i} - 1}{\omega_n^{j-i} - 1} = \frac{1 - 1}{\omega_n^{j-i} - 1} = 0 \end{aligned}$$



## $M_n^*$ como una permutación de $M_n$

Dado  $i \in \{1, \dots, n\}$  y  $j \in \{2, \dots, n\}$ , tenemos que:

$$\begin{aligned}M_n[i, n+2-j] &= \omega_n^{(i-1) \cdot (n+2-j-1)} \\&= \omega_n^{(i-1) \cdot (n+1-j)} \\&= \omega_n^{n \cdot (i-1) + (i-1) \cdot (1-j)} \\&= \omega_n^{n \cdot (i-1)} \cdot \omega_n^{(i-1) \cdot (1-j)} \\&= (\omega_n^n)^{i-1} \cdot \omega_n^{-(i-1) \cdot (j-1)} \\&= 1^{i-1} \cdot \omega_n^{-(i-1) \cdot (j-1)} \\&= \omega_n^{-(i-1) \cdot (j-1)} \\&= M_n^*[i, j]\end{aligned}$$



## $M_n^*$ como una permutación de $M_n$

Para  $j \in \{2, \dots, n\}$ , concluimos que:

$$\begin{pmatrix} M_n^*[1, j] \\ M_n^*[2, j] \\ \vdots \\ M_n^*[n, j] \end{pmatrix} = \begin{pmatrix} M_n[1, n+2-j] \\ M_n[2, n+2-j] \\ \vdots \\ M_n[n, n+2-j] \end{pmatrix}$$

Vamos a usar esta propiedad para definir  $M_n^*$  como una permutación de  $M_n$

## $M_n^*$ como una permutación de $M_n$

Considere la siguiente matriz de permutación de  $n \times n$ :

$$P_n = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

De la propiedad en la transparencia anterior se concluye que:

$$M_n^* = M_n \cdot P_n$$

# Calculando la inversa de la transformada discreta de Fourier

Tenemos los ingredientes necesarios para calcular la inversa de la transformada discreta de Fourier.

- No necesitamos de un nuevo algoritmo para calcularla

## Teorema

Sea  $n \geq 2$ ,  $\bar{a} = (a_0, a_1, \dots, a_{n-1})$  y  $\mathbf{DFT}(\bar{a}) = [y_0, y_1, \dots, y_{n-1}]$ . Si  $\bar{p} = (y_0, y_{n-1}, \dots, y_1)$ , entonces:

$$\frac{1}{n} \cdot \mathbf{DFT}(\bar{p}) = [a_0, a_1, \dots, a_{n-1}]$$

# La demostración del teorema

Tenemos que:

$$M_n \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

Por lo tanto:

$$P_n \cdot M_n \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = P_n \cdot \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_{n-1} \\ \vdots \\ y_1 \end{pmatrix}$$

# La demostración del teorema

Concluimos que:

$$M_n \cdot P_n \cdot M_n \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = M_n \cdot \begin{pmatrix} y_0 \\ y_{n-1} \\ \vdots \\ y_1 \end{pmatrix}$$

Dado que  $M_n \cdot P_n = M_n^*$ , tenemos que:

$$M_n^* \cdot M_n \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = M_n \cdot \begin{pmatrix} y_0 \\ y_{n-1} \\ \vdots \\ y_1 \end{pmatrix}$$

# La demostración del teorema

Así, puesto que  $M_n^* \cdot M_n = n \cdot I_n$ , tenemos que:

$$n \cdot I_n \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = M_n \cdot \begin{pmatrix} y_0 \\ y_{n-1} \\ \vdots \\ y_1 \end{pmatrix}$$

De lo cual concluimos que:

$$\begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = \frac{1}{n} \cdot M_n \cdot \begin{pmatrix} y_0 \\ y_{n-1} \\ \vdots \\ y_1 \end{pmatrix}$$

# La demostración del teorema

Así, utilizando la notación matricial para la transformada discreta de Fourier concluimos que:

$$[a_0, a_1, \dots, a_{n-1}] = \frac{1}{n} \cdot \mathbf{DFT}(\bar{p})$$



# Multiplicando polinomios en tiempo $O(n \cdot \log n)$

