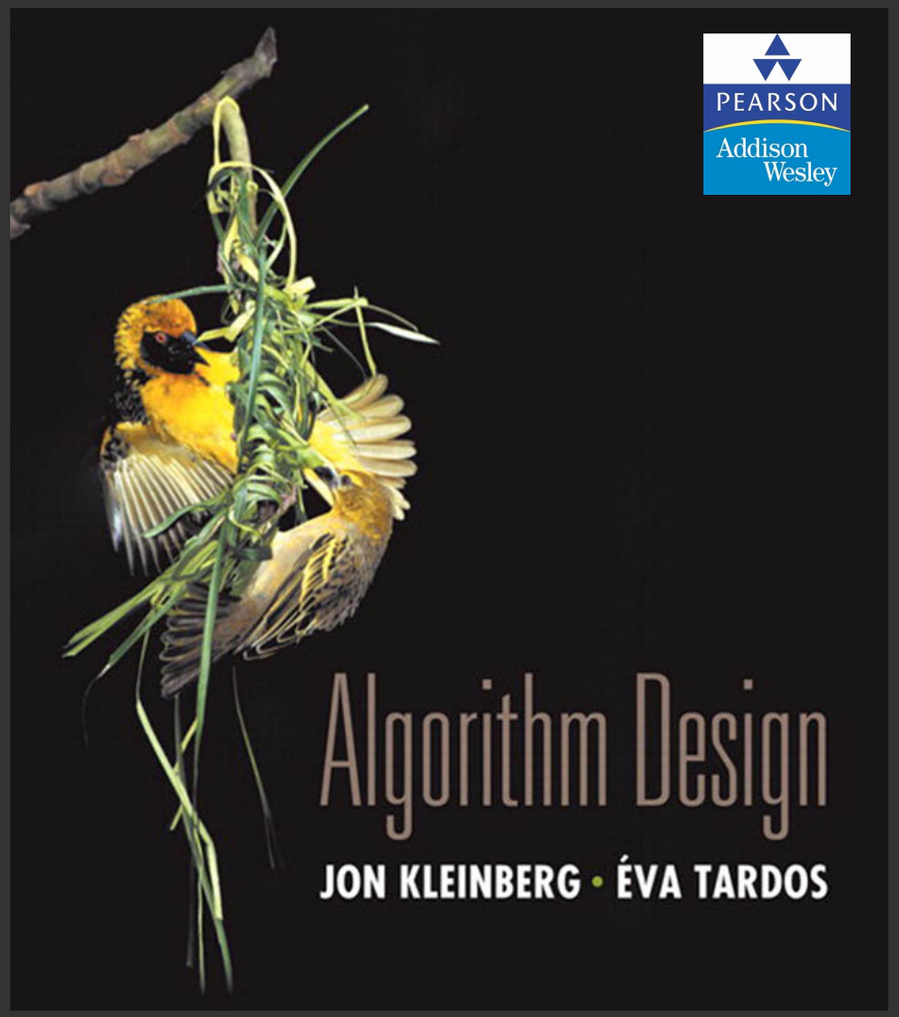


Divide et Impera II

- *teorema master (master theorem)*
- *prodotto di interi*
- ~~*prodotto di matrici*~~
- *convoluzione e FFT*

Lecture slides by Kevin Wayne

http://www.cs.princeton.edu/~wayne/kleinberg-tardos

Last updated on 31/03/26 13:18

Ricorrenze divide et impera

Obiettivo. Ricetta per risolvere ricorrenze divide et impera di forma comune:

$$T(n) = a\,T\left(\frac{n}{b}\right) \;+\; f(n)$$

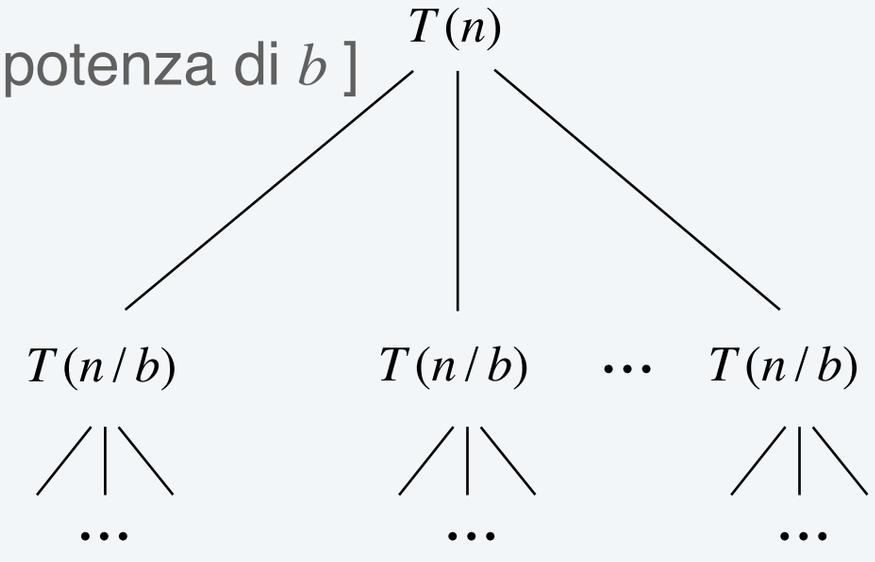
con $T(0) = 0$ e $T(1) = \Theta(1)$.

Notazione.

- $a \geq 1$ è il numero di sottoproblemi.
- $b \geq 2$ è il fattore di diminuzione della taglia di ogni sottoproblema.
- $f(n) \geq 0$ è il lavoro svolto per dividere e ricombinare i sottoproblemi.

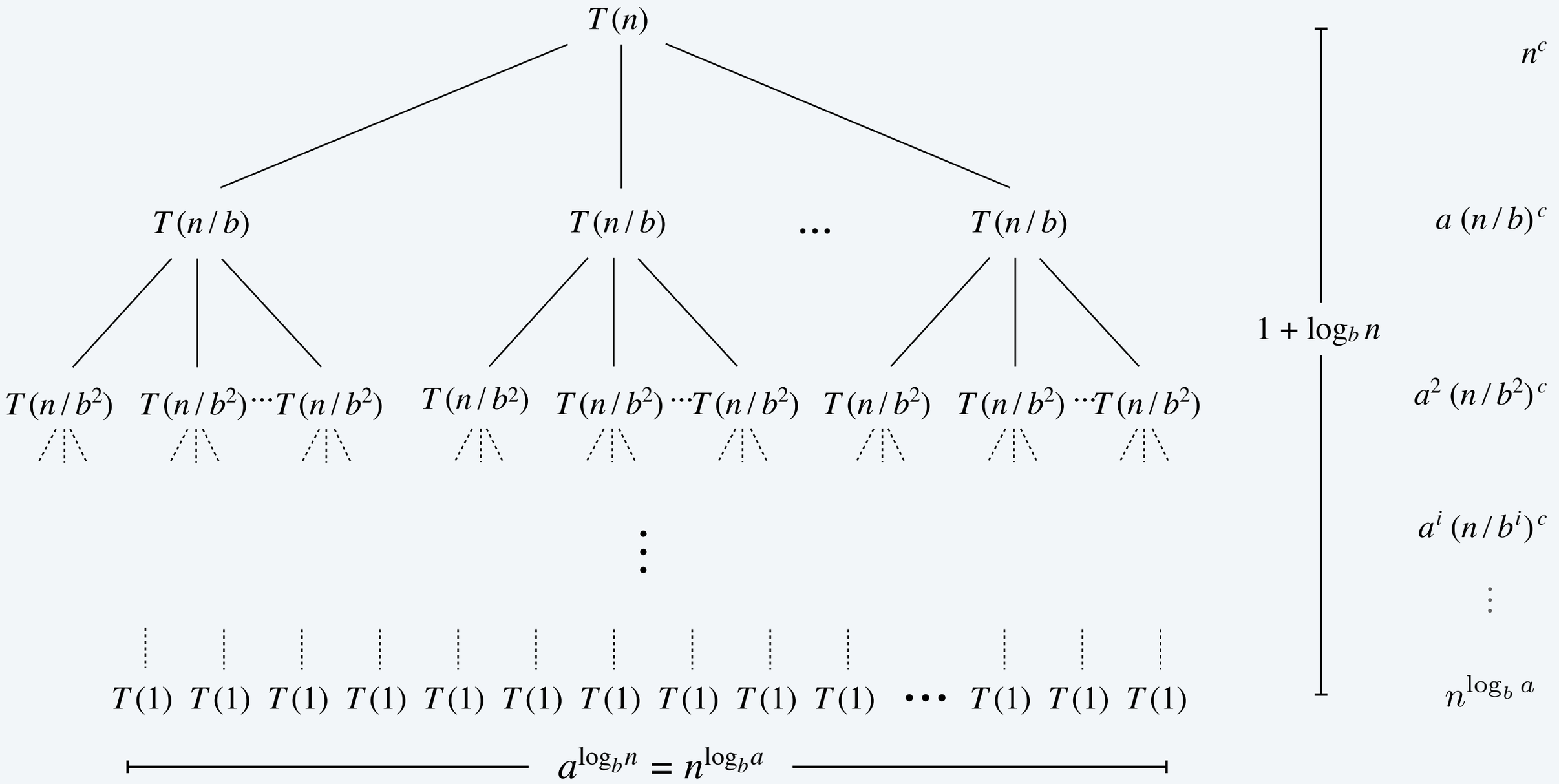
Albero di ricorsione. [assumendo che n sia una potenza di b]

- a = fattore di diramazione.
- a^i = numero di sottoproblemi al livello i.
- $1 + \log_b n$ livelli.
- n / b^i = taglia dei sottoproblemi al livello i.



Ricorrenze divide et impera: albero della ricorsione

Si supponga che $T(n) = a\,T(n/b) + n^c$ con $T(1) = 1$, con n una potenza di b.



$r = a / b^c$ $\qquad T(n) = n^c \sum_{i=0}^{\log_b n} r^i$

Ricorrenze divide et impera: analisi dell'albero della ricorsione

Si supponga che $T(n) = a\,T(n / b) + n^c$ con $T(1) = 1$, con n una potenza di b.

Sia $r = a / b^c$. Notiamo che $r < 1$ sse $c > \log_b a$.

$$T(n) \;=\; n^c \sum_{i=0}^{\log_b n} r^i \;=\; \begin{cases} \Theta(n^c) & \text{if } r < 1 \quad c > \log_b a \\ \Theta(n^c \log n) & \text{if } r = 1 \quad c = \log_b a \\ \Theta(n^{\log_b a}) & \text{if } r > 1 \quad c < \log_b a \end{cases}$$

costo dominato dalla radice

costo distribuito equamente nell'albero

costo dominato dalle foglie

Serie geometriche.

- Se $0 < r < 1$, allora $1 + r + r^2 + r^3 + \ldots + r^k \;\le\; 1 / (1 - r)$.
- Se $r = 1$, allora $1 + r + r^2 + r^3 + \ldots + r^k \;=\; k + 1$.
- Se $r > 1$, allora $1 + r + r^2 + r^3 + \ldots + r^k \;= (r^{k+1} - 1) \;/ (r - 1)$.

Ricorrenze divide et impera: teorema master (teorema dell'esperto)

Teorema master. Siano $a \geq 1$, $b \geq 2$, e $c > 0$ e sia $T(n)$ una funzione definita sui naturali che soddisfa la ricorrenza

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^c)$$

con $T(0) = 0$ e $T(1) = \Theta(1)$, dove n/b può essere $\lfloor n/b \rfloor$ o $\lceil n/b \rceil$. Allora,

Caso 1. Se $c < \log_b a$, allora $T(n) = \Theta(n^{\log_b a})$.

Caso 2. Se $c = \log_b a$, allora $T(n) = \Theta(n^c \log n)$.

Caso 3. Se $c > \log_b a$, allora $T(n) = \Theta(n^c)$.

Bozza dimostrazione.

- Si dimostra prima per b intero e n una potenza intera di b.
- Si estende il dominio della ricorrenza ai reali (o ai razionali).
- Si trattano gli arrotondamenti. ← al più 2 livelli extra nell'albero della ricorsione

$$\begin{aligned} \lceil\lceil\lceil n/b\rceil/b\rceil/b\rceil\rceil &< n/b^3 + (1/b^2 + 1/b + 1) \\ &\leq n/b^3 + 2 \end{aligned}$$

Ricorrenze divide et impera: teorema master (teorema dell'esperto)

Teorema master. Siano $a \geq 1$, $b \geq 2$, e $c > 0$ e sia $T(n)$ una funzione definita sui naturali che soddisfa la ricorrenza

$$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^c)$$

con $T(0) = 0$ e $T(1) = \Theta(1)$, dove n/b può essere $\lfloor n/b \rfloor$ o $\lceil n/b \rceil$. Allora,

Caso 1. Se $c < \log_b a$, allora $T(n) = \Theta(n^{\log_b a})$.

Caso 2. Se $c = \log_b a$, allora $T(n) = \Theta(n^c \log n)$.

Caso 3. Se $c > \log_b a$, allora $T(n) = \Theta(n^c)$.

Estensioni.

- Si possono rimpiazzare i Θ con O.
- Si possono rimpiazzare i Θ con Ω.
- Si possono rimpiazzare le condizioni iniziali con $T(n) = \Theta(1)$ per ogni $n \leq n_0$ e richiedere che la ricorrenza valga solo per ogni $n > n_0$.

Ricorrenze divide et impera: teorema master (teorema dell'esperto)

Teorema master. Siano $a \geq 1$, $b \geq 2$, e $c > 0$ e sia $T(n)$ una funzione definita sui naturali che soddisfa la ricorrenza

$$T(n) \;=\; a\,T\left(\frac{n}{b}\right) \;+\; \Theta(n^c)$$

con $T(0) = 0$ e $T(1) = \Theta(1)$, dove n/b può essere $\lfloor n/b \rfloor$ o $\lceil n/b \rceil$. Allora,

Caso 1. Se $c < \log_b a$, allora $T(n) = \Theta(n^{\log_b a})$.

Caso 2. Se $c = \log_b a$, allora $T(n) = \Theta(n^c \log n)$.

Caso 3. Se $c > \log_b a$, allora $T(n) = \Theta(n^c)$.

Esempio 1. $T(n) = 3\,T(\lfloor n/2 \rfloor) + 5n$.

- $a = 3,\ b = 2, c = 1, \log_b a < 1.58$.
- $T(n) = \Theta(n^{\log_2 3}) = \mathrm{O}(n^{1.58})$.

Ricorrenze divide et impera: teorema master (teorema dell'esperto)

Teorema master. Siano $a \geq 1$, $b \geq 2$, e $c > 0$ e sia $T(n)$ una funzione definita sui naturali che soddisfa la ricorrenza

$$T(n) \;=\; a\,T\left(\frac{n}{b}\right) \;+\; \Theta(n^c)$$

con $T(0) = 0$ e $T(1) = \Theta(1)$, dove n/b può essere $\lfloor n/b \rfloor$ o $\lceil n/b \rceil$. Allora,

Caso 1. Se $c < \log_b a$, allora $T(n) = \Theta(n^{\log_b a})$.

Caso 2. Se $c = \log_b a$, allora $T(n) = \Theta(n^c \log n)$.

Caso 3. Se $c > \log_b a$, allora $T(n) = \Theta(n^c)$.

si possono "mischiare" arrotondamenti in alto e in basso

Es. 2. $T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 17n.$

- $a = 2,\ b = 2, c = 1, \log_b a = 1.$
- $T(n) = \Theta(n \log n).$

Ricorrenze divide et impera: teorema master (teorema dell'esperto)

Teorema master. Siano $a \geq 1$, $b \geq 2$, e $c > 0$ e sia $T(n)$ una funzione definita sui naturali che soddisfa la ricorrenza

$$T(n) \;=\; a\,T\left(\frac{n}{b}\right) \;+\; \Theta(n^c)$$

con $T(0) = 0$ e $T(1) = \Theta(1)$, dove n/b può essere $\lfloor n/b \rfloor$ o $\lceil n/b \rceil$. Allora,

Caso 1. Se $c < \log_b a$, allora $T(n) = \Theta(n^{\log_b a})$.

Caso 2. Se $c = \log_b a$, allora $T(n) = \Theta(n^c \log n)$.

Caso 3. Se $c > \log_b a$, allora $T(n) = \Theta(n^c)$.

Es. 3. $T(n) = 48\,T(\lfloor n/4 \rfloor) + n^3$.

- $a = 48,\ b = 4, c = 3,\ \log_b a \;>\; 2.79$.
- $T(n) = \Theta(n^3)$.

Situazioni in cui il teorema master non è applicabile

- Il numero di sottoproblemi non è costante.

$$T(n) = nT(n/2) + n^2$$

- Il numero di sottoproblemi è minore di 1.

$$T(n) = \frac{1}{2}T(n/2) + n^2$$

- Il lavoro per dividere e ricombinare i sottoproblemi non è $\Theta(n^c)$.

$$T(n) = 2\,T(n/2) + n \log n$$

Divide et impera II: quiz 1

Consideriamo la seguente ricorrenza. Quale caso del teorema master è applicabile?

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 3T(\lceil n/2 \rceil) + \Theta(n) & \text{if } n > 1 \end{cases}$$

A. Caso 1: $T(n) = \Theta(n^{\log_2 3}) = O(n^{1.585})$.

B. Caso 2: $T(n) = \Theta(n \log n)$.

C. Caso 3: $T(n) = \Theta(n)$.

D. Il teorema master non è applicabile.

Divide et impera II: quiz 2

Consideriamo la seguente ricorrenza. Quale caso del teorema master è applicabile?

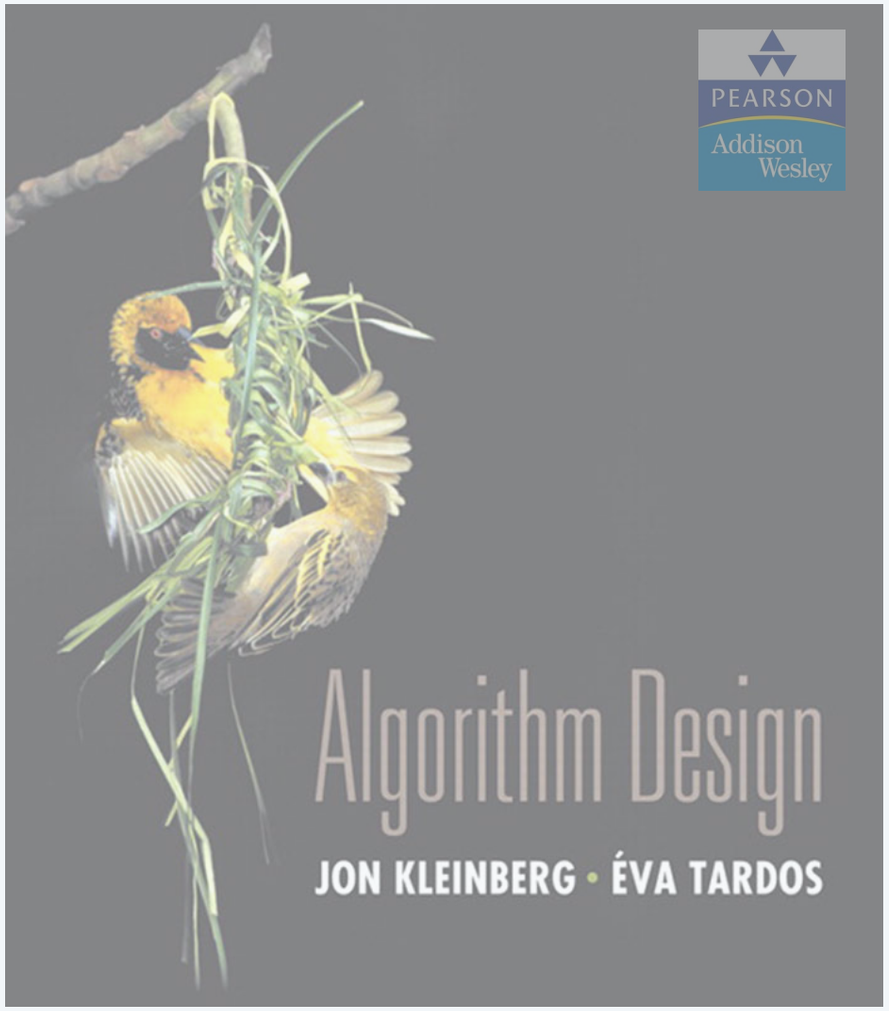
$$T(n) = \begin{cases} 0 & \text{if } n \leq 1 \\ T(\lfloor n/5 \rfloor) + T(n - 3\lfloor n/10 \rfloor) + \frac{11}{5}n & \text{if } n > 1 \end{cases}$$

A. Caso 1: $T(n) = \Theta(n)$.

B. Caso 2: $T(n) = \Theta(n \log n)$.

C. Caso 3: $T(n) = \Theta(n)$.

D. Il teorema master non è applicabile.



SECTION 5.5

DIVIDE ET IMPERA II

- *master theorem*
- *prodotto di interi*
- *matrix multiplication*
- *convolution and FFT*

Addizione e sottrazione intera

Addizione. Dati due interi a n-bit a e b, calcolare $a + b$.

Sottrazione. Dati due interi a n-bit a e b, calcolare $a - b$.

Algoritmi di scuola elementare. $\Theta(n)$ operazioni su bit. ← "complessità a livello di bit" (invece della word RAM)

	1	1	1	1	1	1	0	1
	1	1	0	1	0	1	0	1
+	0	1	1	1	1	1	0	1
1	0	1	0	1	0	0	1	0

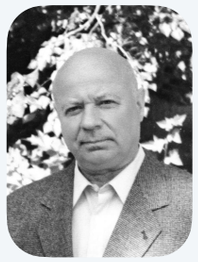
Nota. Gli algoritmi classici per l'addizione e la sottrazione sono asintoticamente ottimali.

Prodotto di interi

Prodotto. Dati due interi a n-bit a e b, calcolare $a \times b$.

Algoritmo classico per la moltiplicazione. $\Theta(n^2)$ operazioni su bit.

	1	1	0	1	0	1	0	1
×	0	1	1	1	1	1	0	1
	1	1	0	1	0	1	0	1
0	0	0	0	0	0	0	0	
	1	1	0	1	0	1	0	1
		1	1	0	1	0	1	0	1						
			1	1	0	1	0	1	0	1					
				1	1	0	1	0	1	0	1				
					1	1	0	1	0	1	0	1			
						1	1	0	1	0	1	0	1		
							0	0	0	0	0	0	0	0	
0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1



Congettura. [Kolmogorov 1956] L'algoritmo classico è asintoticamente ottimo.

Teorema. [Karatsuba 1960] La congettura è falsa.

Prodotto: divide et impera

Per moltiplicare due interi a n-bit x e y:

- Dividi x e y in bit alti (più significativi) e bassi (meno significativi).
- Moltiplica quattro interi a $\frac{1}{2}n$-bit, ricorsivamente.
- Somma e trasla i bit per ottenere il risultato.

$$m = \lceil n / 2 \rceil$$

$$a = \lfloor x / 2^m \rfloor \qquad b = x \bmod 2^m$$

$$c = \lfloor y / 2^m \rfloor \qquad d = y \bmod 2^m$$

← usa lo scorrimento di bit per calcolare 4 termini

$$x\,y = (2^m a + b)(2^m c + d) = \underset{1}{2^{2m} ac} + \underset{2}{2^m (bc} + \underset{3}{ad)} + \underset{4}{bd}$$

Es. $x = 1\,0\,0\,0\,1\,1\,0\,1$ $\quad y = 1\,1\,1\,0\,0\,0\,0\,1$

a b c d

Prodotto: divide et impera

MULTIPLY(x, y, n)

IF $(n = 1)$

  RETURN $x \times y$.

ELSE

  $m \leftarrow \lceil n / 2 \rceil$.

  $a \leftarrow \lfloor x / 2^m \rfloor$; $b \leftarrow x \bmod 2^m$. ← $\Theta(n)$

  $c \leftarrow \lfloor y / 2^m \rfloor$; $d \leftarrow y \bmod 2^m$.

  $e \leftarrow$ MULTIPLY(a, c, m).

  $f \leftarrow$ MULTIPLY(b, d, m).

  $g \leftarrow$ MULTIPLY(b, c, m).

  $h \leftarrow$ MULTIPLY(a, d, m). ← $4\,T(\lceil n / 2 \rceil)$

  RETURN $2^{2m}\, e + 2^m\, (g + h) + f$. ← $\Theta(n)$

Divide et impera II: quiz 3

Quante operazioni servono per il prodotto di due interi a n-bit usando l'algoritmo divide et impera appena descritto?

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 4T(\lceil n/2 \rceil) + \Theta(n) & \text{if } n > 1 \end{cases}$$

A. $T(n) = \Theta(n^{1/2})$.

B. $T(n) = \Theta(n \log n)$.

C. $T(n) = \Theta(n^{\log_2 3}) = \mathrm{O}(n^{1.585})$.

D. $T(n) = \Theta(n^2)$.

L'idea di Karatsuba

Per moltiplicare due interi a n-bit x e y:

- Dividi x e y in bit alti (più significativi) e bassi (meno significativi).
- Per calcolare il termine di mezzo $bc + ad$, usa l'identità:

$$bc + ad = ac + bd - (a - b)(c - d)$$

- Moltipica solo tre interi a $\frac{1}{2}n$-bit, ricorsivamente.

$$m = \lceil n / 2 \rceil$$

$$a = \lfloor x / 2^m \rfloor \qquad b = x \bmod 2^m$$

$$c = \lfloor y / 2^m \rfloor \qquad d = y \bmod 2^m$$

$$x = \underbrace{1\,0\,0\,0}_{a}\ \underbrace{1\,1\,0\,1}_{b}$$

$$y = \underbrace{1\,1\,1\,0}_{c}\ \underbrace{0\,0\,0\,1}_{d}$$

termine di mezzo

$$x\,y = (2^m a + b)(2^m c + d) = 2^{2m} ac + 2^m (bc + ad) + bd$$

$$= 2^{2m} ac + 2^m (ac + bd - (a - b)(c - d)) + bd$$

1 1 3 2 3

Algoritmo di Karatsuba per il prodotto di interi

KARATSUBA-MULTIPLY(x, y, n)

IF $(n = 1)$

  RETURN $x \times y$.

ELSE

  $m \leftarrow \lceil n / 2 \rceil$.

  $a \leftarrow \lfloor x / 2^m \rfloor$; $b \leftarrow x \bmod 2^m$.

  $c \leftarrow \lfloor y / 2^m \rfloor$; $d \leftarrow y \bmod 2^m$.

  $\Theta(n)$

  $e \leftarrow$ KARATSUBA-MULTIPLY(a, c, m).

  $f \leftarrow$ KARATSUBA-MULTIPLY(b, d, m).

  $g \leftarrow$ KARATSUBA-MULTIPLY$(|a - b|, |c - d|, m)$.

  $3\,T(\lceil n / 2 \rceil)$

  Cambia il segno di g se necessario.

  RETURN $2^{2m} e + 2^m (e + f - g) + f$.

  $\Theta(n)$

Analisi dell'algoritmo di Karatsuba

Proposizione. L'algoritmo di Karatsuba's effettua $O(n^{1.585})$ operazioni su bit per moltiplicare due interi a n-bit.

Dim. Si applica il Caso 1 del teorema master alla ricorrenza:

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 3T(\lceil n/2 \rceil) + \Theta(n) & \text{if } n > 1 \end{cases}$$

$$\Longrightarrow \quad T(n) = \Theta(n^{\log_2 3}) = O(n^{1.585})$$

In pratica.

- Si usa la base 32 o 64 (anziché la base 2).
- Più rapido dell'algoritmo classico da circa 320–640 bit in su.

Operazioni aritmetiche di complessità equivalente al prodotto

Prodotto di interi. Dati due interi a n-bit, calcolare il loro prodotto.

problema aritmetico	formula	complessità a livello di bit
prodotto di interi	$a \times b$	$M(n)$
quadrato di interi	a^2	$\Theta(M(n))$
divisione intera	$\lfloor a / b \rfloor$, $a \bmod b$	$\Theta(M(n))$
radici quadrate intere	$\lfloor \sqrt{a} \rfloor$	$\Theta(M(n))$

$$ab = \frac{(a+b)^2 - a^2 - b^2}{2}$$

problemi di aritmetica intera aventi la stessa complessità a livello di bit della moltiplicazione intera (M(n))

Storia della complessità asintotica della moltiplicazione intera

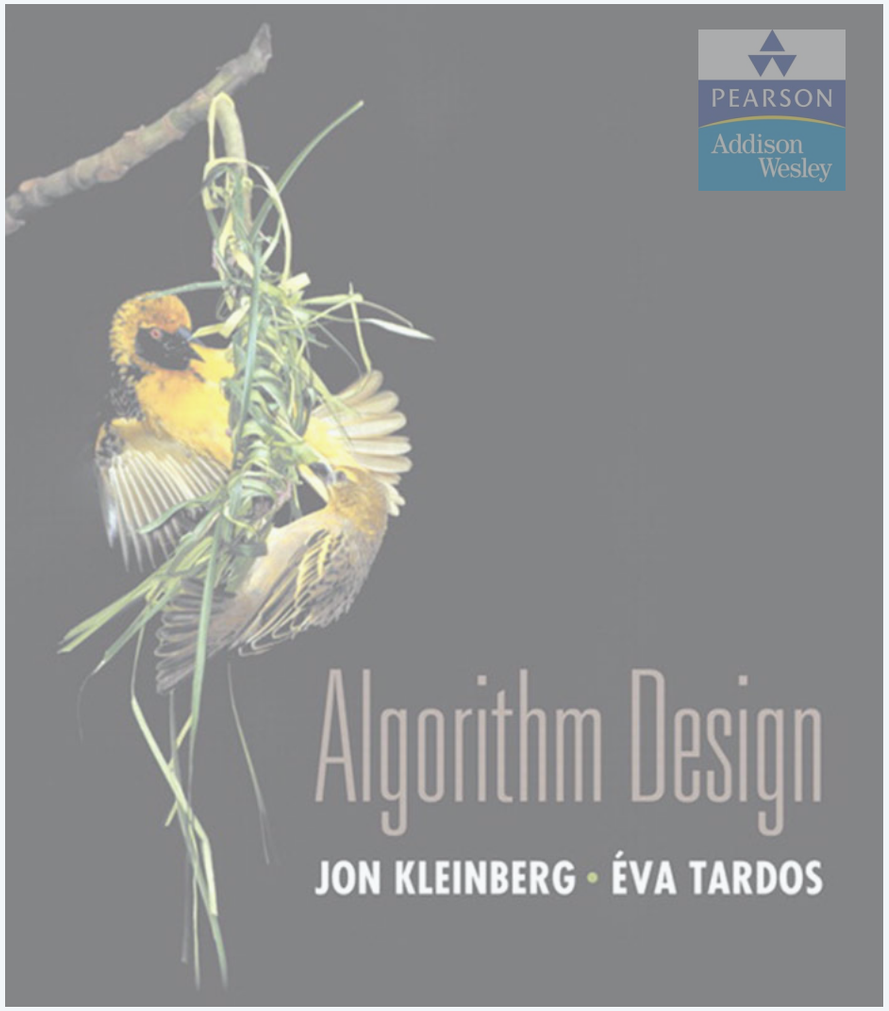
anno	algoritmo	operazioni su bit
12xx	**classico**	$O(n^2)$
1962	**Karatsuba-Ofman**	$O(n^{1.585})$
1963	**Toom-3, Toom-4**	$O(n^{1.465})$, $O(n^{1.404})$
1966	**Toom-Cook**	$O(n^{1+\varepsilon})$
1971	**Schönhage-Strassen**	$O(n \log n \cdot \log\log n)$
2007	**Fürer**	$n \log n \, 2^{O(\log^* n)}$
2019	**Harvey-van der Hoeven**	$O(n \log n)$
		$O(n)$

numero di operazioni su bit per moltiplicare di due interi a n-bit

Nota. La libreria GNU Multiple Precision library usa uno dei primi cinque algoritmi a seconda del valore di n.

GMP
«Arithmetic without limitations»

usata da Maple, Mathematica, gcc, crittografia, ...



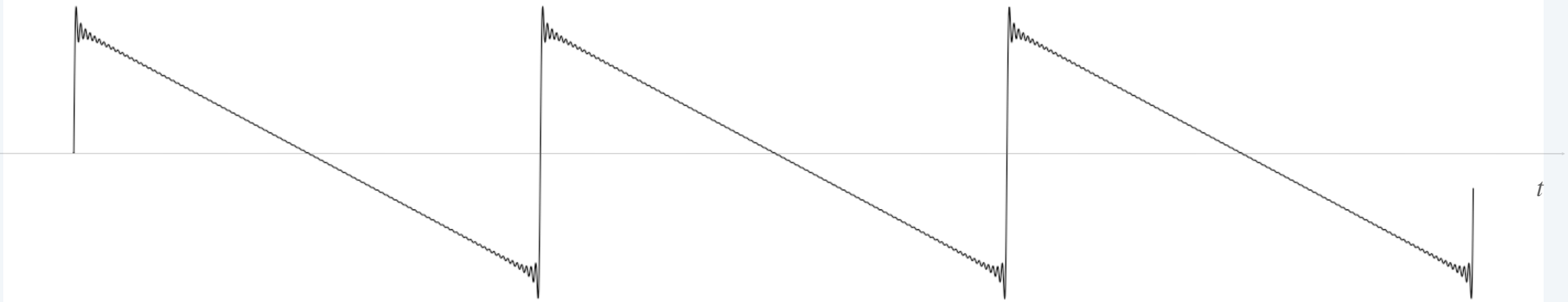
SECTION 5.6

DIVIDE ET IMPERA II

- *master theorem*
- *integer multiplication*
- *matrix multiplication*
- *convoluzione e FFT*

Analisi di Fourier

Teorema di Fourier. [Fourier, Dirichlet, Riemann] Qualunque funzione periodica sufficientemente liscia può essere espressa come somma di una serie di sinusoidi.



$$y(t) = \frac{2}{\pi} \sum_{k=1}^{n} \frac{\sin kt}{k} \qquad n = 100$$

Identità di Eulero

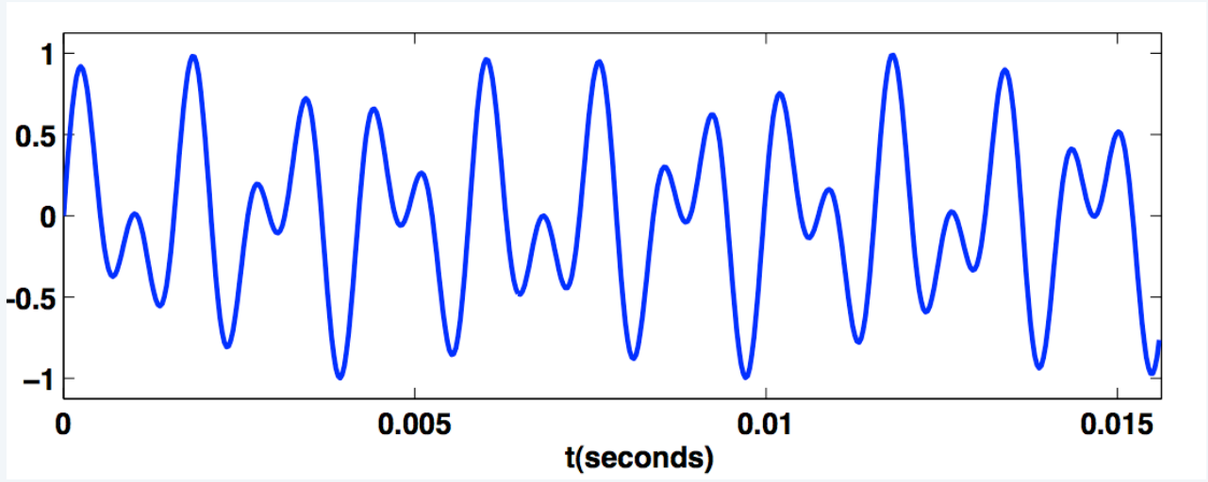
Identità di Eulero. $e^{ix} = \cos x + i \sin x$.

Sinusoidi. Somma di seni e coseni = somma di esponenziali complessi.

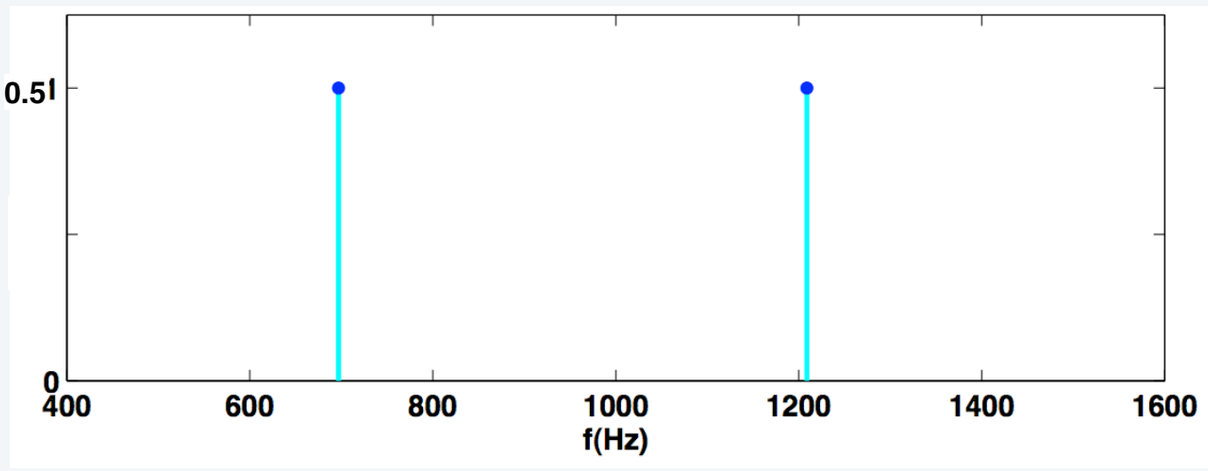
Dominio del tempo vs. dominio della frequenza

Segnale. [pulsante 1 del telefono] $y(t) = \frac{1}{2}\sin(2\pi \cdot 697\ t) + \frac{1}{2}\sin(2\pi \cdot 1209\ t)$

Dominio del tempo.



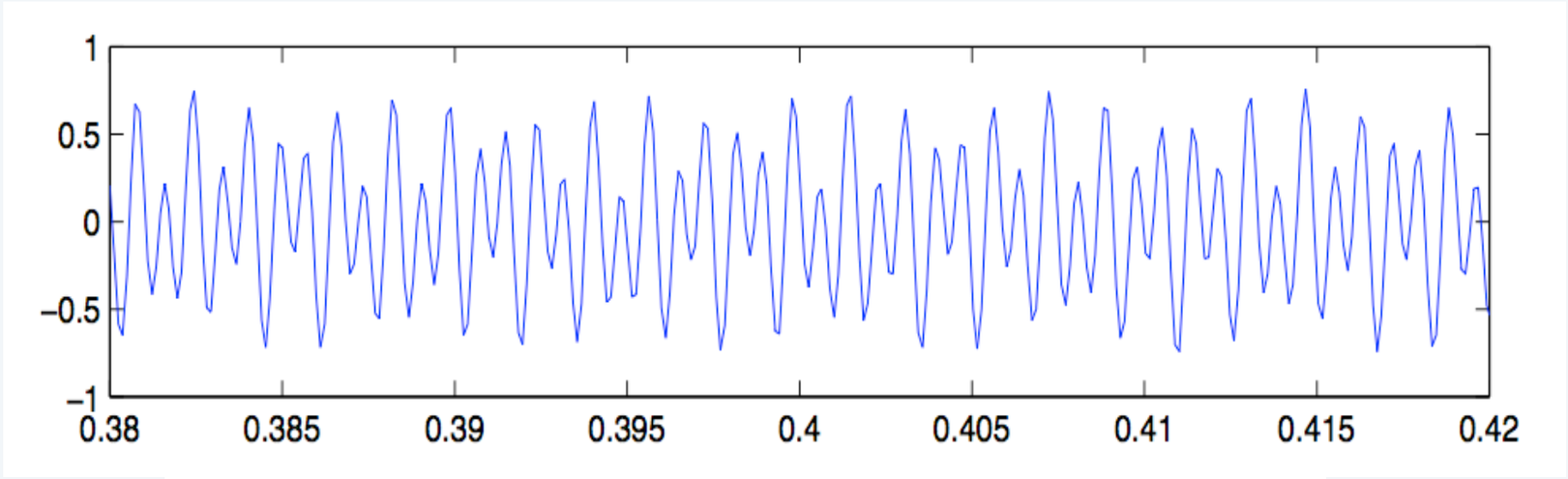
Dominio della frequenza.



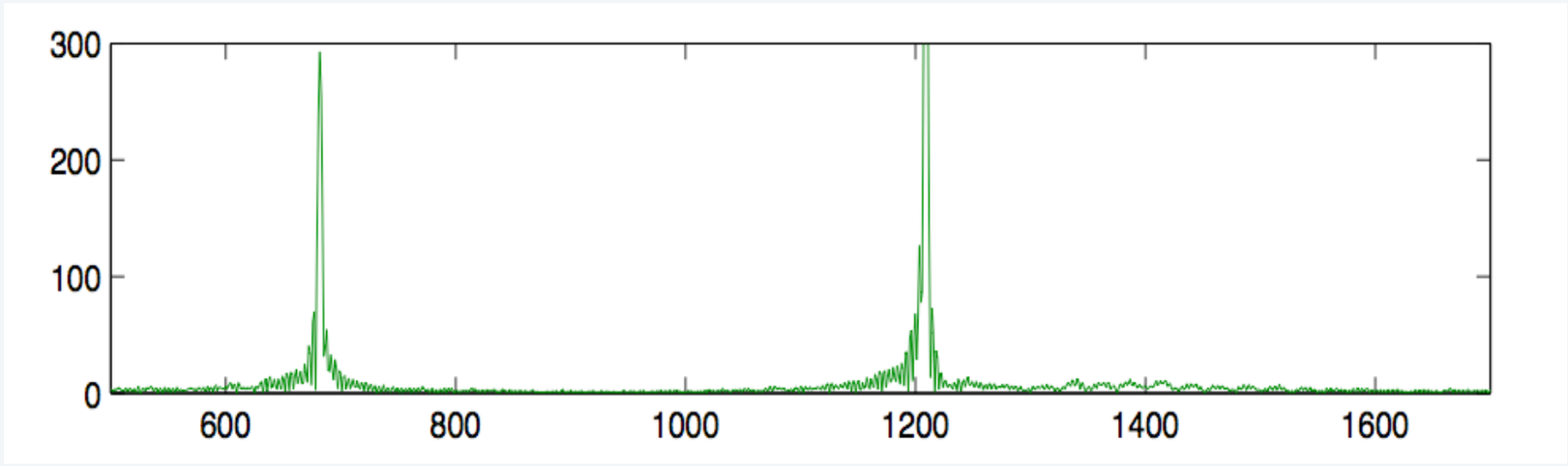
Riferimento: Cleve Moler, Numerical Computing with MATLAB

Dominio del tempo vs. dominio della frequenza

Segnale. [registrazione, 8192 campioni al secondo]



Magnitudo della trasformata discreta di Fourier.



Riferimento: Cleve Moler, Numerical Computing with MATLAB

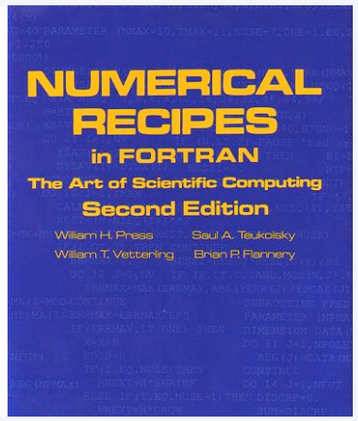
Trasformata di Fourier veloce: Fast Fourier Transform (FFT)

FFT. Modo rapido di convertire un segnale dal dominio del tempo al dominio della frequenza.

Punto di vista alternativo. Modo rapido di moltiplicare e valutare polinomi.

seguiamo questo approccio

> *" If you speed up any nontrivial algorithm by a factor of a million or so the world will beat a path towards finding useful applications for it. "* — *Numerical Recipes*



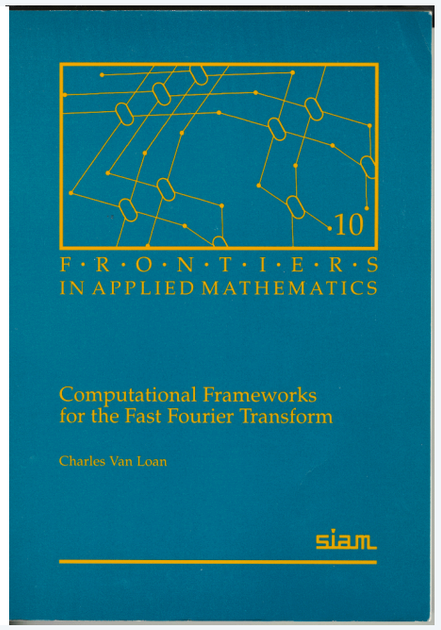
Fast Fourier transform: applicazioni

Applicazioni.

- Ottica, acustica, fisica quantistica, telecomunicazioni, radar, sistemi di controllo, elaborazione dei segnali, riconoscimento del parlato, compressione dati, elaborazione di immagini, sismologia, spettrometria, …
- Media digitali. [DVD, JPEG, MP3, H.264]
- Diagnostica medica. [MRI, CT, PET scans, ultrasound]
- Soluzioni numeriche dell'equazione di Poisson.
- Moltiplicazione di interi e di polinomi.
- Algoritmo quantistico di fattorizzazione di Shor.
- …

" The FFT is one of the truly great computational developments of [the 20th] century. It has changed the face of science and engineering so much that it is not an exaggeration to say that life as we know it would be very different without the FFT. "

— Charles van Loan



Fast Fourier transform: breve storia

Gauss (1805, 1866). Analizzò il moto periodico dell'asteroide Ceres.

Runge–König (1924). Posero le fondamenta teoriche.

Danielson–Lanczos (1942). Algoritmo efficiente, cristallografia a raggi x.

Cooley–Tukey (1965). Individuazione di test nucleari in Unione Sovietica e tracciamento di sottomarini. Hanno riscoperto e popolarizzato la FFT.

An Algorithm for the Machine Calculation of Complex Fourier Series

By James W. Cooley and John W. Tukey

An efficient method for the calculation of the interactions of a 2^m factorial experiment was introduced by Yates and is widely known by his name. The generalization to 3^m was given by Box et al. [1]. Good [2] generalized these methods and gave elegant algorithms for which one class of applications is the calculation of Fourier series. In their full generality, Good's methods are applicable to certain problems in which one must multiply an N-vector by an $N \times N$ matrix which can be factored into m sparse matrices, where m is proportional to $\log N$. This results in a procedure requiring a number of operations proportional to $N \log N$ rather than N^2.

Importanza non compresa appieno fino all'avvento dei calcolatori digitali.

Polinomi: rappresentazione tramite i coefficienti

Polinomi univariati. [rappresentazione a coefficienti]

$$A(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1}$$
$$B(x) = b_0 + b_1 x + b_2 x^2 + \ldots + b_{n-1} x^{n-1}$$

Addizione. $O(n)$ operazioni aritmetiche.

$$A(x) + B(x) = (a_0 + b_0) + (a_1 + b_1)x + \ldots + (a_{n-1} + b_{n-1}) x^{n-1}$$

Valutazione. $O(n)$ usando il metodo di Horner.

$$A(x) = a_0 + (x(a_1 + x(a_2 + \ldots + x(a_{n-2} + x(a_{n-1}))\ldots))$$

```
val = 0.0
for j in range(n-1, -1, -1):
    val = a[j] + (x * val)
```

Moltiplicazione (convoluzione lineare). $O(n^2)$ tramite forza bruta.

$$A(x) \times B(x) = \sum_{i=0}^{2n-2} c_i x^i \quad \text{where} \quad c_i = \sum_{j=0}^{i} a_j b_{i-j}$$

Divide et impera II: quiz 6

Quale fu l'argomento della tesi di dottorato di Gauss?

A. L'eliminazione Gaussiana.

B. La Fast Fourier transform.

C. Il teorema dei numeri primi.

D. Il teorema integrale di Cauchy.

E. Il teorema fondamentale dell'algebra.

F. Le funzioni che preservano gli angoli.

G. Il metodo dei minimi quadrati.

H. La geometria non-euclidea.

I. La costruzione dell'eptadecagono regolare tramite riga e compasso.

Un titolo modesto per una dissertazione dottorale

DEMONSTRATIO NOVA
THEOREMATIS
OMNEM FVNCTIONEM ALGEBRAICAM
RATIONALEM INTEGRAM
VNIVS VARIABILIS
IN FACTORES REALES PRIMI VEL SECUNDI GRADVS
RESOLVI POSSE

AVCTORE
CAROLO FRIDERICO GAVSS
HELMSTADII
APVD C. G. FLECKEISEN. 1799

1.

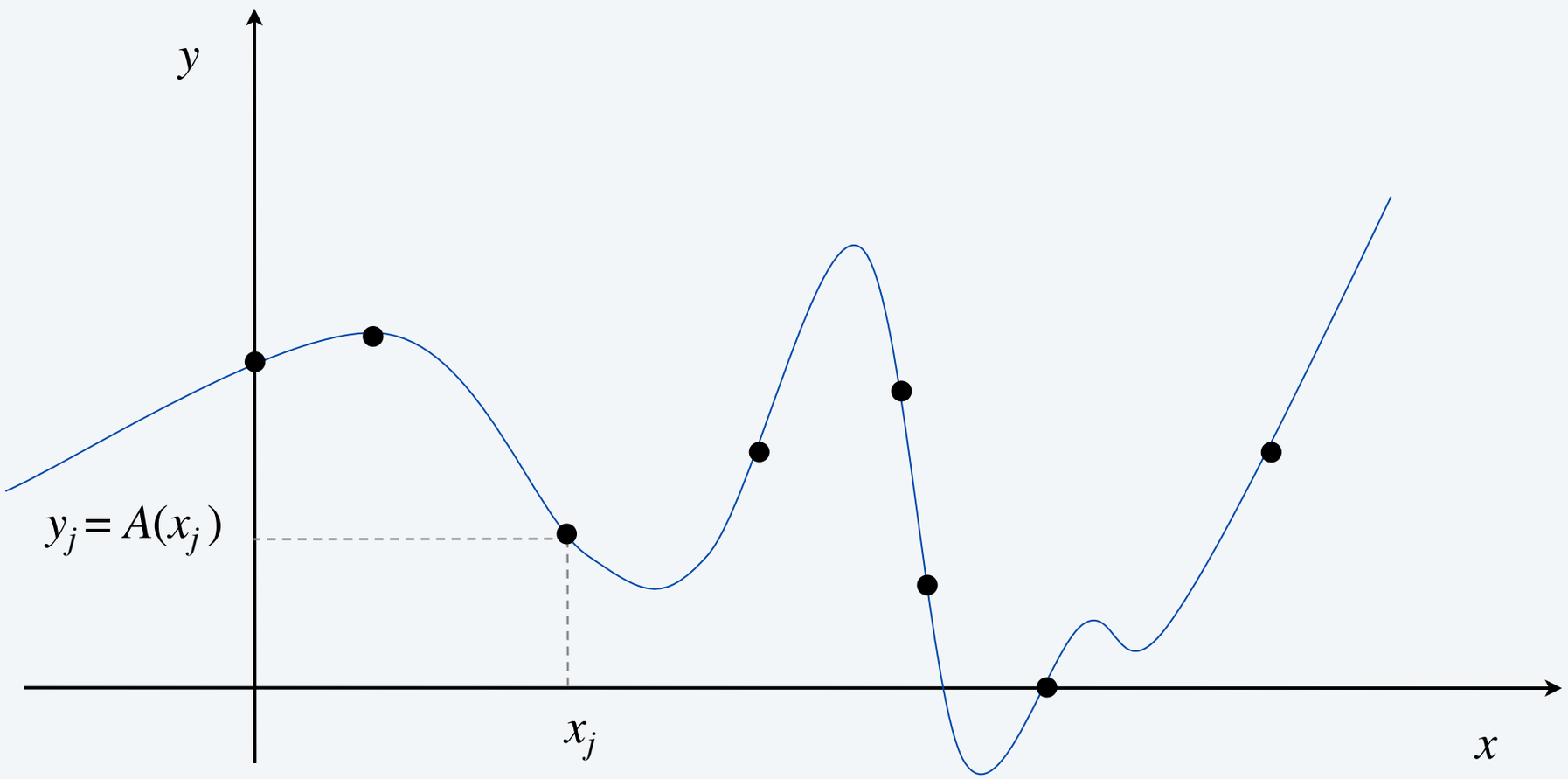
Quaelibet aequatio algebraica determinata reduci potest ad formam $x^m+Ax^{m-1}+Bx^{m-2}+$ etc. $+M=0$, ita vt m sit numerus integer positiuus. Si partem primam huius aequationis per X denotamus, aequationique $X=0$ per plures valores inaequales ipsius x satisfieri supponimus, puta ponendo $x=\alpha$, $x=\beta$, $x=\gamma$ etc. functio X per productum e factoribus $x-\alpha$, $x-\beta$, $x-\gamma$ etc. diuisibilis erit. Vice versa, si productum e pluribus factoribus simplicibus $x-\alpha$, $x-\beta$, $x-\gamma$ etc. functionem X metitur: aequationi $X=0$ satisfiet, aequando ipsam x cuicunque quantitatum α, β, γ etc. Denique si X producto ex m factoribus talibus simplicibus aequalis est (siue omnes diuersi sint, siue quidam ex ipsis identici): alii factores simplices praeter hos functionem X metiri non poterunt. Quamobrem aequatio m^{ti} gradus plures quam m radices habere nequit; simul vero patet, aequationem m^{ti} gradus *pauciores* radices habere posse, etsi X in m factores simplices resolubilis sit:

“ Nuova dimostrazione del teorema che ogni funzione algebrica razionale intera in una variabile si fattorizza in fattori reali di primo o secondo grado. ”

Polinomi: rappresentazione punto-valore

Teorema fondamentale dell'algebra. Un polinomio univariato di grado n a coefficienti complessi ha esattamente n radici complesse.

Corollario. Un polinomio univariato di grado $n-1$ $A(x)$ è individuato in modo univoco dalla sua valutazione su n punti x distinti.



Polinomi: rappresentazione punto-valore

Polinomi univariati. [rappresentazione punto-valore]

$$A(x): \ (x_0, y_0), \ldots, (x_{n-1}, y_{n-1})$$

$$B(x): \ (x_0, z_0), \ldots, (x_{n-1}, z_{n-1})$$

Addizione. $O(n)$ operazioni aritmetiche.

$$A(x) + B(x): \ (x_0, y_0 + z_0), \ldots, (x_{n-1}, y_{n-1} + z_{n-1})$$

Prodotto. $O(n)$, ma $A(x)$ e $B(x)$ sono rappresentati con $2n$ punti.

$$A(x) \times B(x): \ (x_0, y_0 \times z_0), \ldots, (x_{2n-1}, y_{2n-1} \times z_{2n-1})$$

Valutazione. $O(n^2)$ usando la formula di Lagrange.

$$A(x) \ = \ \sum_{k=0}^{n-1} y_k \, \frac{\prod_{j \neq k} (x - x_j)}{\prod_{j \neq k} (x_k - x_j)}$$

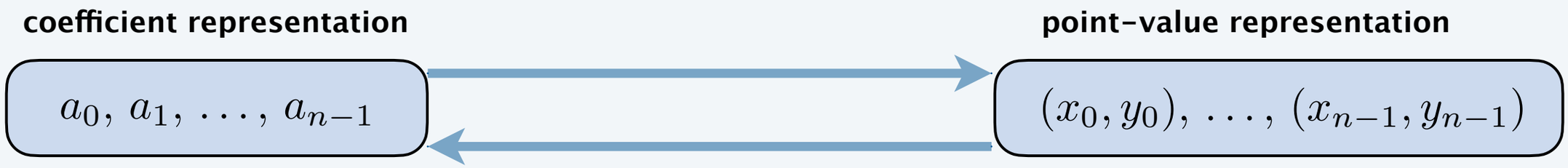
← non si usa in pratica

Conversione tra le due rappresentazioni

Tradeoff. Valutazione rapida o prodotto rapido? Vorremmo entrambe le cose!

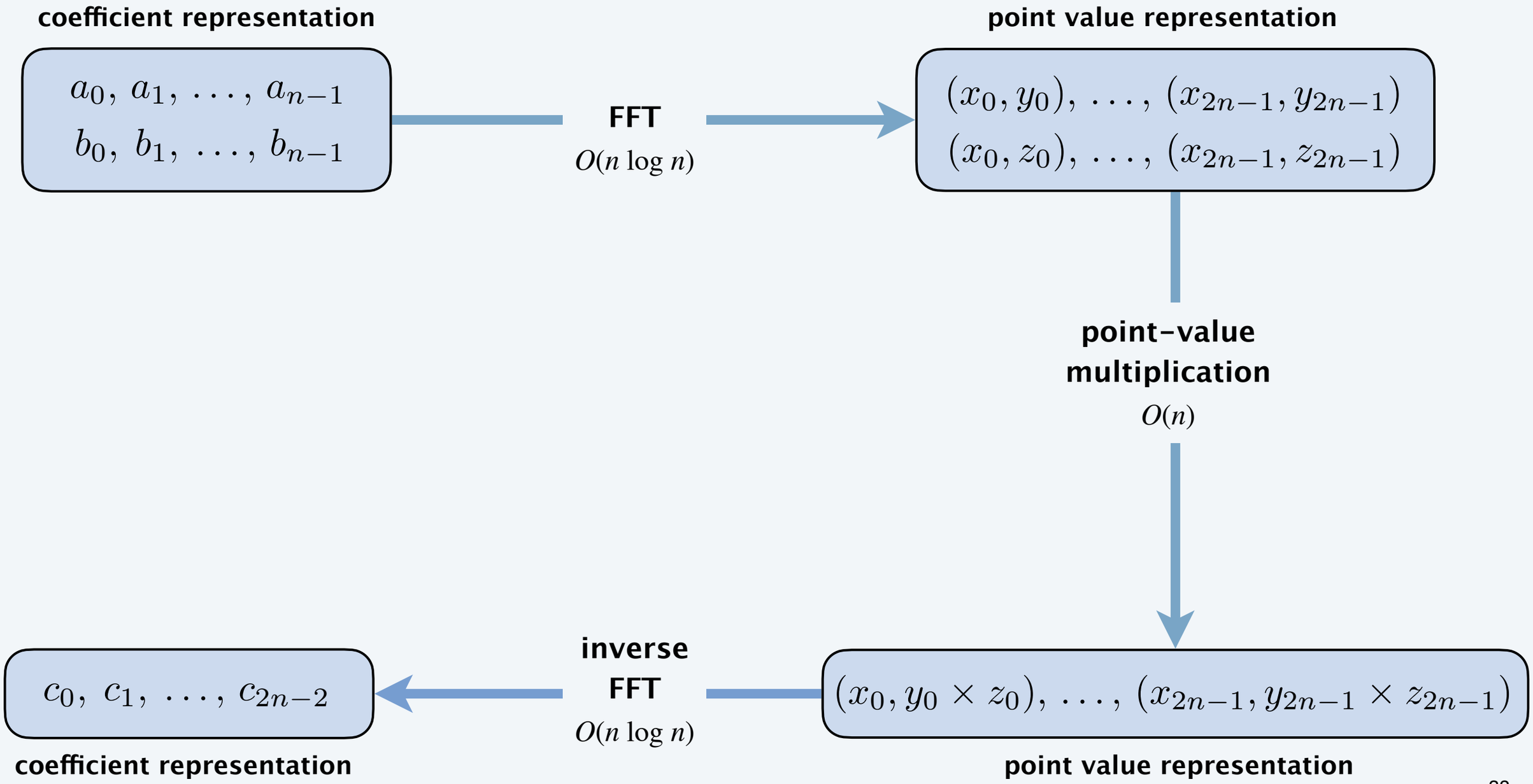
rappresentazione	prodotto	valutazione
a coefficienti	$O(n^2)$	$O(n)$
punto-valore	$O(n)$	$O(n^2)$

Obiettivo. Conversione efficiente tra le due rappresentazioni $\Rightarrow$ così che tutte le operazioni possano essere rapide.



Conversione tra le due rappresentazioni

Applicazione. Prodotto di polinomi (rappresentazione a coefficienti).



Conversione tra le due rappresentazioni: metodo a forza bruta

A coefficienti $\Rightarrow$ punto-valore. Dato un polinomio $A(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$, valutarlo in n punti distinti $x_0, \ldots, x_{n-1}$.

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

Tempo di esecuzione. $O(n^2)$ tramite prodotti matrice-vettore (o n metodi di Horner).

Conversione tra le due rappresentazioni: metodo a forza bruta

Punto-valore ⇒ a coefficienti. Dati n punti distinti $x_0, \ldots, x_{n-1}$ e valori $y_0, \ldots, y_{n-1}$, trovare il polinomio (unico) $A(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$, che ha i valori dati nei punti dati.

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

la matrice di Vandermonde è invertibile sse gli x_i sono distinti

Tempo di esecuzione. $O(n^3)$ tramite eliminazione gaussiana.

o $O(n^{2.38})$ tramite fast matrix multiplication

Divide et impera II: quiz 7

Quale metodo divide et impera si può usare per il prodotto di polinomi?

$$A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7.$$

A. Dividi il polinomio in termini di grado alto e di grado basso.

$$A_{low}(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3.$$
$$A_{high}(x) = a_4 + a_5 x + a_6 x^2 + a_7 x^3.$$

B. Dividi il polinomio in termini di grado pari e di grado dispari.

$$A_{even}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3.$$
$$A_{odd}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3.$$

C. Sia A che B.

D. Né A né B.

Divide et impera

Decimazione nel tempo. Divide i termini in grado pari e grado dispari.

- $A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7.$
- $A_{even}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3.$
- $A_{odd}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3.$
- $A(x) = A_{even}(x^2) + x\, A_{odd}(x^2).$

Cooley–Tukey radix 2 FFT

Decimazione in frequenza. Divide i termini in grado alto e grado basso.

- $A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7.$
- $A_{low}(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3.$
- $A_{high}(x) = a_4 + a_5 x + a_6 x^2 + a_7 x^3.$
- $A(x) = A_{low}(x) + x^4\, A_{high}(x).$

Sande–Tukey radix 2 FFT

Da rappresentazione a coefficienti a punto-valore: intuizione

A coefficienti ⇒ punto-valore. Dato un polinomio $A(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$, valutarlo in n punti distinti $x_0, \ldots, x_{n-1}$. ← possiamo scegliere noi quali!

Divide. Spezza il polinomio in termini di grado pari e dispari.

- $A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7.$
- $A_{even}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3.$
- $A_{odd}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3.$
- $A(x) = A_{even}(x^2) + x A_{odd}(x^2).$
- $A(-x) = A_{even}(x^2) - x A_{odd}(x^2).$

Intuizione. Scegli i due punti ± 1.

- $A(1) = A_{even}(1) + 1 A_{odd}(1).$
- $A(-1) = A_{even}(1) - 1 A_{odd}(1).$

← **Si può valutare un polinomio di grado n-1 in 2 punti valutando due polinomi di grado ½n - 1 in 1 solo punto.**

Da rappresentazione a coefficienti a punto-valore: intuizione

A coefficienti ⇒ punto-valore. Dato un polinomio $A(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$, valutarlo in n punti distinti $x_0, \ldots, x_{n-1}$. ← possiamo scegliere noi quali!

Divide. Spezza il polinomio in termini di grado pari e dispari.

- $A(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 + a_6 x^6 + a_7 x^7.$
- $A_{even}(x) = a_0 + a_2 x + a_4 x^2 + a_6 x^3.$
- $A_{odd}(x) = a_1 + a_3 x + a_5 x^2 + a_7 x^3.$
- $A(x) = A_{even}(x^2) + x A_{odd}(x^2).$
- $A(-x) = A_{even}(x^2) - x A_{odd}(x^2).$

Intuizione. Scegli quattro punti complessi: $\pm 1, \pm i$.

- $A(1) = A_{even}(1) + 1 A_{odd}(1).$
- $A(-1) = A_{even}(1) - 1 A_{odd}(1).$
- $A(i) = A_{even}(-1) + i A_{odd}(-1).$
- $A(-i) = A_{even}(-1) - i A_{odd}(-1).$

← **Si può valutare un polinomio di grado n-1 in 4 punti valutando due polinomi di grado ½n - 1 in 2 punti.**

Trasformata discreta di Fourier

A coefficienti $\Rightarrow$ punto-valore. Dato un polinomio $A(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$, valutarlo in n punti distinti $x_0, \ldots, x_{n-1}$. $\longleftarrow$ possiamo scegliere noi quali!

Idea chiave. Scegli $x_k = \omega^k$ dove ω è una radice n-esima dell'unità.

$$y_k = A(\omega^k) \longrightarrow \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(n-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \omega^{3(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-1} \end{bmatrix}$$

DFT

Fourier matrix F_n

Radici dell'unità

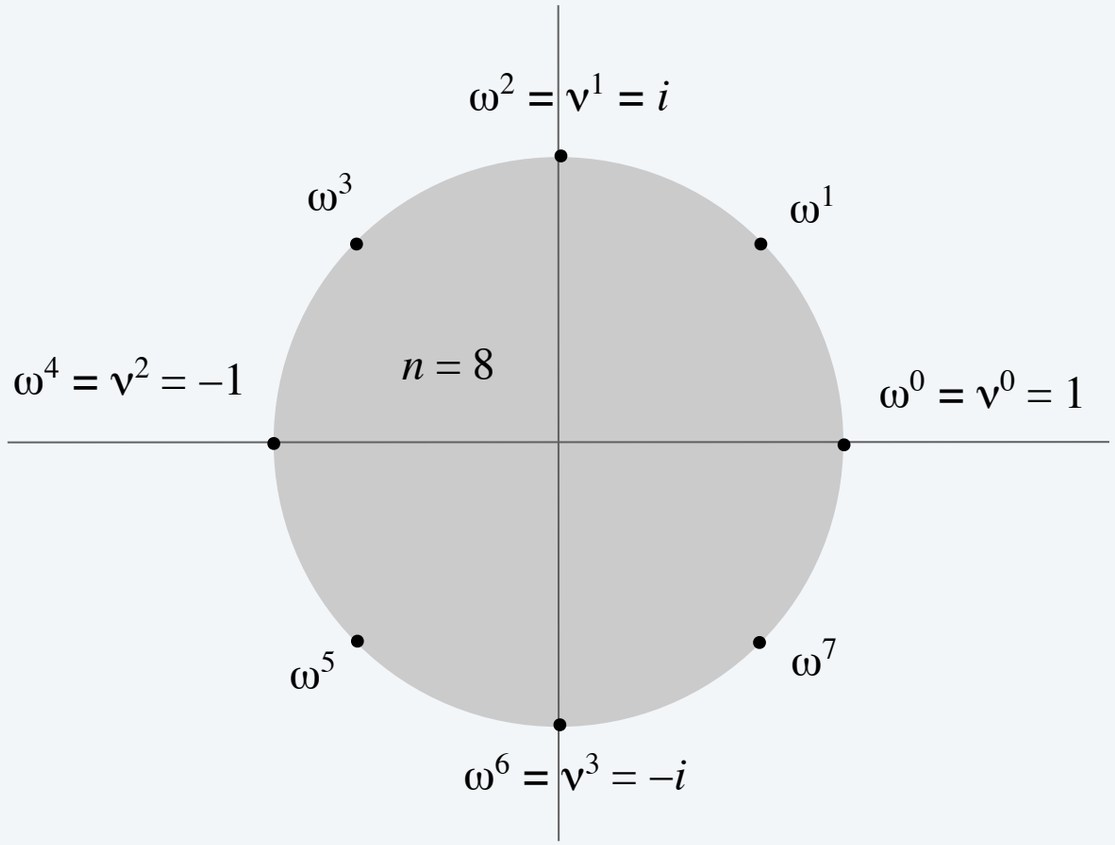
Def. Una radice n-esima dell'unità è un numero complesso x tale che $x^n = 1$.

Fatto. Le radici n-esime dell'unità sono: $\omega^0, \omega^1, \ldots, \omega^{n-1}$ dove $\omega = e^{2\pi i / n}$.

Dim. $(\omega^k)^n = (e^{2\pi i k / n})^n = (e^{\pi i})^{2k} = (-1)^{2k} = 1$.

alternativa comune: $\omega = e^{-2\pi i / n}$

Fatto. Le radici $\frac{1}{2}n$-esime dell'unità sono: $\nu^0, \nu^1, \ldots, \nu^{n/2-1}$ dove $\nu = \omega^2 = e^{4\pi i / n}$.



Fast Fourier transform

Obiettivo. Valutare un polinomio di grado $n - 1$, $A(x) = a_0 + \ldots + a_{n-1}\, x^{n-1}$ nelle radici n-esime dell'unità: $\omega^0, \omega^1, \ldots, \omega^{n-1}$.

Divide. Spezza il polinomio in termini di grado pari e di grado dispari.

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

Impera. Valuta $A_{even}(x)$ e $A_{odd}(x)$ nelle radice $\frac{1}{2}n$-esime dell'unità: $\nu^0, \nu^1, \ldots, \nu^{n/2-1}$.

Combina.

$\nu^k = (\omega^k)^2$

- $y_k = A(\omega^k) = A_{even}(\nu^k) + \omega^k\, A_{odd}(\nu^k), \quad 0 \le k < n/2.$
- $y_{k+\frac{1}{2}n} = A(\omega^{k+\frac{1}{2}n}) = A_{even}(\nu^k) - \omega^k\, A_{odd}(\nu^k), \quad 0 \le k < n/2.$

$A(-\omega^k)$

FFT: implementazione

Obiettivo. Valutare un polinomio di grado $n-1$, $A(x) = a_0 + \ldots + a_{n-1}\, x^{n-1}$ nelle radici n-esime dell'unità: $\omega^0, \omega^1, \ldots, \omega^{n-1}$.

- $y_k = A(\omega^k) = A_{even}(\nu^k) + \omega^k A_{odd}(\nu^k), \quad 0 \le k < n/2.$
- $y_{k+\frac{1}{2}n} = A(\omega^{k+\frac{1}{2}n}) = A_{even}(\nu^k) - \omega^k A_{odd}(\nu^k), \quad 0 \le k < n/2.$

FFT$(n, a_0, a_1, a_2, \ldots, a_{n-1})$

IF $(n = 1)$ RETURN a_0.

$(e_0, e_1, \ldots, e_{n/2-1}) \leftarrow$ FFT$(n/2, a_0, a_2, a_4, \ldots, a_{n-2})$.

$(d_0, d_1, \ldots, d_{n/2-1}) \leftarrow$ FFT$(n/2, a_1, a_3, a_5, \ldots, a_{n-1})$.

← $2\,T(n/2)$

FOR $k = 0$ TO $n/2 - 1$.

$\omega^k \leftarrow e^{2\pi i k/n}$.

$y_k \leftarrow e_k + \omega^k d_k$.

$y_{k+n/2} \leftarrow e_k - \omega^k d_k$.

← $\Theta(n)$

RETURN $(y_0, y_1, y_2, \ldots, y_{n-1})$.

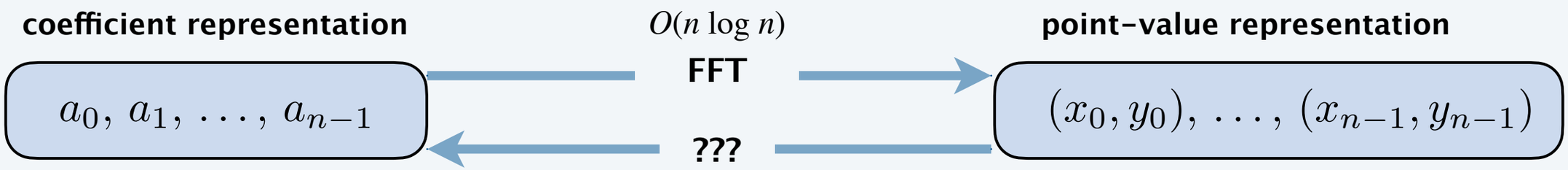
FFT: riassunto

Teorema. L'algoritmo FFT valuta un polinomio di grado $n - 1$ in ciascuna radice n-esima dell'unità con $O(n \log n)$ operazioni aritmetiche e spazio aggiuntivo $O(n)$.

assume che n sia una potenza di 2

Dim.

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 2T(n/2) + \Theta(n) & \text{if } n > 1 \end{cases}$$



Divide et impera II: quiz 8

Nel calcolo della FFT di (a_0, a_1, a_2, …, a_7), quali sono i primi due coefficienti coinvolti in una operazione aritmetica?

A. a_0 e a_1.

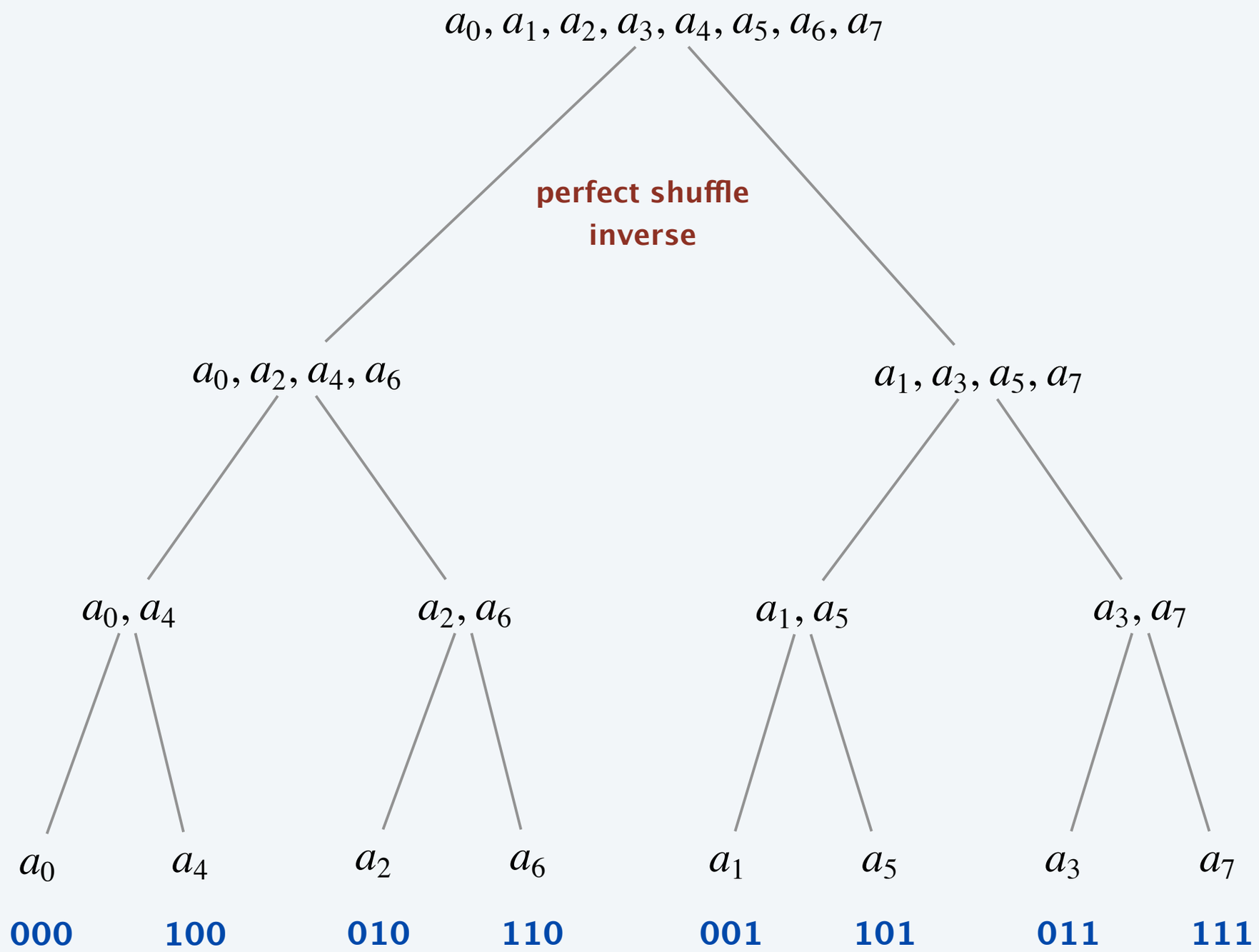
B. a_0 e a_2.

C. a_0 e a_4.

D. a_0 e a_7.

E. Nessuna delle precedenti.

FFT: albero della ricorsione



"bit-reversed" order

FFT: decomposizione della matrice di Fourier

Punto di vista equivalente. La FFT attua una decomposizione ricorsiva della matrice di Fourier.

Fourier matrix

$$F_n = \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(n-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \omega^{3(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{bmatrix} \qquad a = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{n-1} \end{bmatrix}$$

$$I_n = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} \qquad D_n = \begin{bmatrix} \omega^0 & 0 & 0 & \dots & 0 \\ 0 & \omega^1 & 0 & \dots & 0 \\ 0 & 0 & \omega^2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \omega^{n-1} \end{bmatrix}$$

DFT

$$y = F_n\, a = \begin{bmatrix} I_{n/2} & D_{n/2} \\ I_{n/2} & -D_{n/2} \end{bmatrix} \begin{bmatrix} F_{n/2}\, a_{even} \\ F_{n/2}\, a_{odd} \end{bmatrix}$$

Inversa della trasformata discreta di Fourier

Punto-valore ⇒ a coefficienti. Dati n punti distinti $x_0, \ldots, x_{n-1}$ e valori $y_0, \ldots, y_{n-1}$, trovare il polinomio unico $a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$, che ha i valori dati nei punti dati.

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(n-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \omega^{3(n-1)} & \cdots & \omega^{(n-1)(n-1)} \end{bmatrix}^{-1} \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-1} \end{bmatrix}$$

Inverse DFT

Fourier matrix inverse $(F_n)^{-1}$

Inversa della trasformata discreta di Fourier

Proposizione. L'inversa della matrice di Fourier F_n è data dalla formula:

$$G_n = \frac{1}{n}\begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega^{-1} & \omega^{-2} & \omega^{-3} & \cdots & \omega^{-(n-1)} \\ 1 & \omega^{-2} & \omega^{-4} & \omega^{-6} & \cdots & \omega^{-2(n-1)} \\ 1 & \omega^{-3} & \omega^{-6} & \omega^{-9} & \cdots & \omega^{-3(n-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(n-1)} & \omega^{-2(n-1)} & \omega^{-3(n-1)} & \cdots & \omega^{-(n-1)(n-1)} \end{bmatrix}$$

F_n / $\sqrt{n}$ è una matrice unitaria

Conseguenza. Per calcolare la FFT inversa, si applica lo stesso algoritmo ma si usa $\omega^{-1} = e^{-2\pi i / n}$ come radice principale n-esima dell'unità (e si divide il risultato per n).

Moltiplicazione di polinomi

Teorema. Dati due polinomi $A(x) = a_0 + a_1 x + \ldots + a_{n-1} x^{n-1}$ e $B(x) = b_0 + b_1 x + \ldots + b_{n-1} b^{n-1}$ di grado $n - 1$, li possiamo moltiplicare usando $O(n \log n)$ operazioni aritmetiche.

si aggiungono zeri per rendere n una potenza di 2

Dim.