Algoritmi e Strutture Dati

Lezione 10

19 ottobre 2022

ALGORITMO mergeSort (Array A)

IF lunghezza di A ≤ 1 THEN

| non fare nulla

ELSE

| B ← 1° metà di A
| C ← 2ª metà di A
| mergeSort(B)
| mergeSort(C)
| A ← merge(B, C)

Tecnica "divide–et–impera"

$\mathcal{P}$ Problema

$\mathcal{I}$ Istanza di $\mathcal{P}$

- Se $\mathcal{I}$ è piccola risolvi $\mathcal{P}$ su $\mathcal{I}$ direttamente

  altrimenti

  - dividi $\mathcal{I}$ in istanze di lunghezza minore della lunghezza di $\mathcal{I}$
  - risolvi queste istanze separatamete
  - ricava la soluzione di $\mathcal{I}$ combinando opportunamente le soluzioni ottenute

Tecnica “divide–et–impera”

$\mathcal{P}$ Problema

$\mathcal{I}$ Istanza di $\mathcal{P}$

```
ALGORITMO risolviP ( Istanza I ) → soluzione
  IF |I| ≤ C THEN
    risolvi P su I direttamente
    RETURN soluzione trovata
  ELSE
    dividi I in m istanze ridotte I_1, I_2, ..., I_m
             con |I_j| < |I| per j = 1, ..., m
    sol_1 ← risolviP(I_1)
    sol_2 ← risolviP(I_2)
        ⋮
    sol_m ← risolviP(I_m)
    RETURN combina (sol_1, sol_2, ..., sol_m)
```

Tecnica "divide–et–impera": tempo

```
ALGORITMO risolviP ( Istanza I ) → soluzione
  IF |I| ≤ C THEN
    risolvi P su I direttamente
    RETURN soluzione trovata
  ELSE
    dividi I in m istanze ridotte I_1, I_2, ..., I_m
          con |I_j| < |I| per j = 1, ..., m
    sol_1 ← risolviP (I_1)
    sol_2 ← risolviP (I_2)
      ⋮
    sol_m ← risolviP (I_m)
    RETURN combina (sol_1, sol_2, ..., sol_m)
```

MergeSort

$C = 1$

$m = 2$

$|I_1| = \left\lfloor \frac{|I|}{2} \right\rfloor \qquad |I_2| = \left\lceil \frac{|I|}{2} \right\rceil$

$$T(I) = \begin{cases} \text{costante} \quad \Theta(1) \\ T_{\text{dividi}}(I) + T(I_1) + T(I_2) + \cdots + T(I_m) + \\ \qquad T_{\text{combina}}(sol_1, \ldots, sol_m) \end{cases}$$

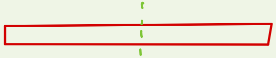
Calcolo del minimo e del massimo in un vettore

Input Vettore A di $n > 0$ elementi

Output Il valore minimo e il valore massimo in A

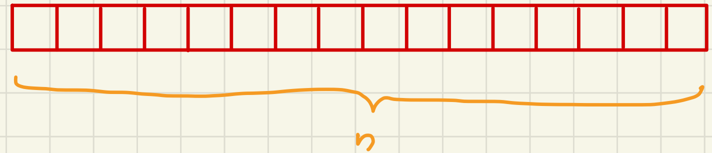
```
ALGORITMO minMax (array A[0..n-1]) → (elemento, elemento)
    min ← A[0]
    max ← A[0]
    FOR i ← 1 TO n-1 DO
        IF A[i] < min THEN min ← A[i]
        IF A[i] > max THEN max ← A[i]
    RETURN (min, max)
```

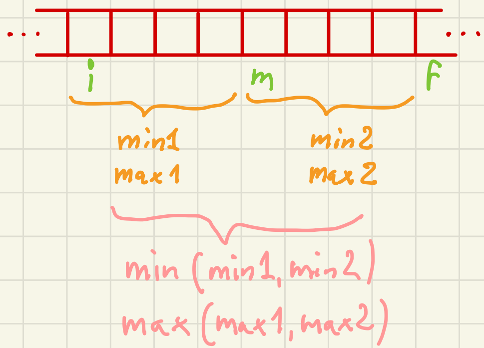
#cfr $2n-2$



$n = 1, 2$

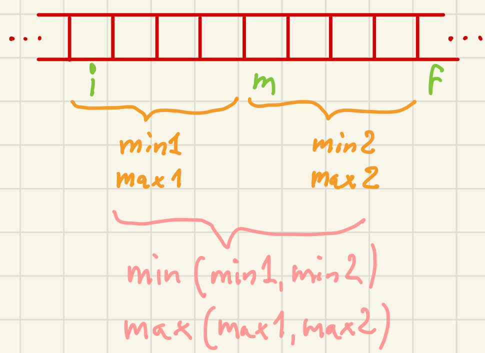
$n > 2$


n


i
m
f
min1
max 1
min2
max 2
min (min1, min2)
max (max1, max2)

```
FUNZIONE minMax (Array A, indice i, indice f) → (elemento, elemento)
    IF f - i = 1 THEN RETURN (A[i], A[i])
    ELSE IF f-i=2 THEN
        IF A[i]<A[i+1] THEN RETURN (A[i], A[i+1])
                       ELSE RETURN (A[i+1], A[i])
    ELSE
        m ← (f+i)/2
        (min1, max1) ← minMax(A, i, m)
        (min2, max2) ← minMax(A, m, f)
        IF  min1<min2  THEN  min ← min1
                       ELSE  min ← min2
        IF  max1>max2  THEN  max ← max1
                       ELSE  max ← max2
        RETURN (min, max)
```

```
ALGORITMO minMax (Array A[0..n-1])
      → (elemento, elemento)
   RETURN   minMax(A, 0, n)
```



```
IF f-i = 1 THEN RETURN (A[i], A[i])
ELSE IF f-i=2 THEN
|       IF A[i]<A[i+1] THEN RETURN (A[i], A[i+1])
|                      ELSE RETURN (A[i+1], A[i])
ELSE
|  m ← (f+i)/2
|  (min1, max1) ← minMax(A, i, m)
|  (min2, max2) ← minMax(A, m, f)
|  IF min1<min2 THEN min ← min1
|               ELSE min ← min2
|  IF max1>max2 THEN max ← max1
|               ELSE max ← max2
|_ RETURN (min, max)
```

$$C(n) = \begin{cases} 0 & \text{se } n = 1 \\ 1 & \text{se } n = 2 \\ C\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + C\left(\left\lceil \frac{n}{2} \right\rceil\right) + 2 & \text{altrimenti} \end{cases}$$

per n potenza di 2

$$C(n) = \begin{cases} 0 & \text{se } n = 1 \\ 1 & \text{se } n = 2 \\ 2C\left(\frac{n}{2}\right) + 2 & \text{altrimenti} \end{cases}$$

$$C(n) = 2C\left(\frac{n}{2}\right) + 2 = 2\left[2C\left(\frac{n}{2^2}\right) + 2\right] + 2$$

$$= 2^2 C\left(\frac{n}{2^2}\right) + 2^2 + 2 = 2^2\left[2C\left(\frac{n}{2^3}\right) + 2\right] + 2^2 + 2$$

$$= 2^3 C\left(\frac{n}{2^3}\right) + 2^3 + 2^2 + 2 = \dots$$

$$= 2^k C\left(\frac{n}{2^k}\right) + \underbrace{2^k + 2^{k-1} + \dots + 2}_{\sum_{i=1}^{k} 2^i \, = \, 2^{k+1} - 2} = 2^k C\left(\frac{n}{2^k}\right) + 2^{k+1} - 2$$

$$\frac{n}{2^k} = 2 \qquad n = 2^{k+1} \qquad \frac{n}{2} = 2^k$$

$$k = \log_2 \frac{n}{2} = \log_2 n - 1$$

$$= \frac{n}{2} \underbrace{C(2)}_{1} + n - 2 = \frac{n}{2} + n - 2 = \frac{3}{2} n - 2$$

$$C(n) = \left\lceil \frac{3}{2} n \right\rceil - 2$$

Equazioni di ricorrenza: teorema fondamentale

(Master Theorem)

Siano $m, a, b', b'', c \in \mathbb{R}^+$ con $a > 1$. L'equazione

$$F(n) = \begin{cases} b' & \text{se } n = 1 \\ mF\left(\frac{n}{a}\right) + b'' n^c & \text{se } n > 1 \end{cases}$$

soddisfa le seguenti relazioni:

$$F(n) = \begin{cases} \Theta(n^c) & \text{se } m < a^c \\ \Theta(n^c \log n) & \text{se } m = a^c \\ \Theta(n^{log_a m}) & \text{se } m > a^c \end{cases}$$

Il risultato può essere esteso anche al caso in cui per $n > 1$ l'equazione data sia della forma

$$F(n) = m_1 F\left(\left\lfloor \frac{n}{a} \right\rfloor\right) + m_2 F\left(\left\lceil \frac{n}{a} \right\rceil\right) + b'' n^c$$

con $m = m_1 + m_2$

Master theorem: esempi

$$H(n) = \begin{cases} 1 & \text{se } n=1 \\ H\left(\frac{n}{2}\right)+1 & \text{altrimenti} \end{cases}$$

$b' = b'' = 1 \qquad m=1 \qquad c=0 \qquad a=2 \qquad a^c = 1 = m$

$$H(n) = \Theta(\lg n)$$

Siano $m, a, b', b'', c \in \mathbb{R}^+$ con $a > 1$. L'equazione

$$F(n) = \begin{cases} b' & \text{se } n = 1 \\ mF\left(\frac{n}{a}\right) + b''n^c & \text{se } n > 1 \end{cases}$$

soddisfa le seguenti relazioni:

$$F(n) = \begin{cases} \Theta(n^c) & \text{se } m < a^c \\ \Theta(n^c \log n) & \text{se } m = a^c \\ \Theta(n^{\log_a m}) & \text{se } m > a^c \end{cases}$$

Master theorem: esempi

$$T(n) = \begin{cases} 100 & \text{se } n=1 \\ 2T\left(\frac{n}{2}\right) + 8n & \text{altrimenti} \end{cases}$$

$m=2 \quad a=2 \quad b'=100 \quad b''=8 \quad c=1 \qquad a^c=2=m$

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

Siano $m, a, b', b'', c \in \mathbb{R}^+$ con $a > 1$. L'equazione

$$F(n) = \begin{cases} b' & \text{se } n = 1 \\ mF\left(\frac{n}{a}\right) + b''n^c & \text{se } n > 1 \end{cases}$$

soddisfa le seguenti relazioni:

$$F(n) = \begin{cases} \Theta(n^c) & \text{se } m < a^c \\ \Theta(n^c \log n) & \text{se } m = a^c \\ \Theta(n^{\log_a m}) & \text{se } m > a^c \end{cases}$$

Master theorem: esempi

$$F(n) = \begin{cases} 1 & \text{se } n = 1 \\ 8F\left(\frac{n}{2}\right) + n^2 & \text{se } n > 1 \end{cases}$$

$m = 8 \quad a = 2 \quad c = 2 \quad a^c = 4 < m = 8$

$$F(n) = \Theta\left(n^{\log_2 8}\right) = \Theta\left(n^3\right)$$

Siano $m, a, b', b'', c \in \mathbb{R}^+$ con $a > 1$. L'equazione

$$F(n) = \begin{cases} b' & \text{se } n = 1 \\ mF\left(\frac{n}{a}\right) + b''n^c & \text{se } n > 1 \end{cases}$$

soddisfa le seguenti relazioni:

$$F(n) = \begin{cases} \Theta(n^c) & \text{se } m < a^c \\ \Theta(n^c \log n) & \text{se } m = a^c \\ \Theta(n^{\log_a m}) & \text{se } m > a^c \end{cases}$$

Master theorem: esempi

$$G(n) = \begin{cases} 1 & \text{se } n=1 \\ 7G\left(\frac{n}{2}\right) + 6n^2 & \text{se } n>1 \end{cases}$$

$m=7 \quad a=2 \quad c=2 \qquad a^c = 4 < m = 7$

$$G(n) = \Theta\left(n^{\log_2 7}\right)$$

$$= \Theta\left(n^{2.81}\right)$$

Siano $m, a, b', b'', c \in \mathbb{R}^+$ con $a > 1$. L'equazione

$$F(n) = \begin{cases} b' & \text{se } n = 1 \\ mF\left(\frac{n}{a}\right) + b''n^c & \text{se } n > 1 \end{cases}$$

soddisfa le seguenti relazioni:

$$F(n) = \begin{cases} \Theta(n^c) & \text{se } m < a^c \\ \Theta(n^c \log n) & \text{se } m = a^c \\ \Theta(n^{\log_a m}) & \text{se } m > a^c \end{cases}$$

Esempio: somma di matrici $n \times n$

somme, prodotti di interi

Quante operazioni "elementari" sono necessarie e sufficienti per calcolare la somma di 2 matrici $n \times n$ di interi?

Input $A = [a_{ij}]$, $B = [b_{ij}]$, matrici $n \times n$ di interi

Output $C = [c_{ij}] = A + B$ somma delle matrici A e B

Definizione di *somma di matrici*:

$$c_{ij} = \underbrace{a_{ij} + b_{ij}}_{\text{1 somma di interi per ciascuna posizione}}$$

n^2 somme in totale

Esempio: prodotto di matrici $n \times n$

Input $A = [a_{ij}]$, $B = [b_{ij}]$, matrici $n \times n$ di interi

Output $C = [c_{ij}] = A \cdot B$ prodotto delle matrici A e B

Definizione di *prodotto di matrici*: $$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj}$$

$\left.\begin{array}{l} n \text{ prodotti} \\ n \text{ somme} \end{array}\right] 2n$ operazioni

Per tutta la matrice $2n \cdot n^2 = 2n^3$ operazioni

Con $O(n^3)$ operazioni calcolo C

Lower bound $\Omega(n^2)$
Calcolo un'operazione
per ogni elemento

Prodotto di matrici

Divide-et-impera "immediato"

$$A = \left(\begin{array}{cc|cc} 1 & 4 & 8 & 3 \\ 2 & 5 & 7 & 1 \\ \hline 11 & 3 & 4 & 0 \\ 1 & 2 & 4 & 3 \end{array} \right)$$

Se $A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$ e $B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$

con A_{ij}, B_{ij} matrici $\frac{n}{2} \times \frac{n}{2}$, allora

$$C = A \cdot B = \begin{bmatrix} A_{11} \cdot B_{11} + A_{12} \cdot B_{21} & A_{11} \cdot B_{12} + A_{12} \cdot B_{22} \\ A_{21} \cdot B_{11} + A_{22} \cdot B_{21} & A_{21} \cdot B_{12} + A_{22} \cdot B_{22} \end{bmatrix}$$

Prodotto di matrici

Divide-et-impera "immediato"

$T(n)$ = # operazioni elementari per moltiplicare 2 matrici $n \times n$

ALGORITMO moltiplica (Matrici $n \times n$ A, B) → Matrice

IF $n = 1$ THEN RETURN $[a_{11} b_{11}]$ → 1 operazione

ELSE

decomponi A e B in matrici $\frac{n}{2} \times \frac{n}{2}$

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \text{ e } B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

calcola le seguenti matrici $\frac{n}{2} \times \frac{n}{2}$:

$$C_{11} = A_{11} \cdot B_{11} + A_{12} \cdot B_{21}$$
$$C_{12} = A_{11} \cdot B_{12} + A_{12} \cdot B_{22}$$
$$C_{21} = A_{21} \cdot B_{11} + A_{22} \cdot B_{21}$$
$$C_{22} = A_{21} \cdot B_{12} + A_{22} \cdot B_{22}$$

4 somme di matrici $\frac{n}{2} \times \frac{n}{2}$ $\quad 4\left(\frac{n}{2}\right)^2 = \cancel{4}\frac{n^2}{\cancel{4}} = n^2$

8 prodotti " " $\frac{n}{2} \times \frac{n}{2}$ $\quad 8T\left(\frac{n}{2}\right)$

RETURN $C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$

$$T(n) = \begin{cases} 1 & \text{se } n = 1 \\ 8T\left(\frac{n}{2}\right) + n^2 & \text{altrimenti} \end{cases} \Longrightarrow T(n) = \Theta(n^3)$$

come algoritmo diretto

Prodotto di matrici

Divide-et-impera migliorato – Algoritmo di Strassen

Se $A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$ e $B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$ calcolo le matrici:

- $M_1 = (A_{21} + A_{22} - A_{11}) \cdot (B_{22} - B_{12} + B_{11})$
- $M_2 = A_{11} \cdot B_{11}$
- $M_3 = A_{12} \cdot B_{21}$
- $M_4 = (A_{11} - A_{21}) \cdot (B_{22} - B_{11})$
- $M_5 = (A_{21} + A_{22}) \cdot (B_{12} - B_{11})$
- $M_6 = (A_{12} - A_{21} + A_{11} - A_{22}) \cdot B_{22}$
- $M_7 = A_{22} \cdot (B_{11} + B_{22} - B_{12} - B_{21})$

7 prodotti di matrici $\frac{n}{2} \times \frac{n}{2}$

24 somme/sottrazioni di matrici $\frac{n}{2} \times \frac{n}{2}$

Allora:

$$C = A \cdot B = \begin{bmatrix} M_2 + M_3 & M_1 + M_2 + M_5 + M_6 \\ M_1 + M_2 + M_4 - M_7 & M_1 + M_2 + M_4 + M_5 \end{bmatrix}$$

Algoritmo di Strassen

$T(n) = \#$ operazioni elementari per moltiplicare 2 matrici $n \times n$

ALGORITMO moltiplica (Matrici $n \times n$ A, B) $\longrightarrow$ Matrice

IF $n = 1$ THEN RETURN $[a_{11} b_{11}]$

1 operazione

ELSE

decomponi A e B in matrici $\frac{n}{2} \times \frac{n}{2}$

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \text{ e } B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

calcola le seguenti matrici $\frac{n}{2} \times \frac{n}{2}$:

$$M_1 = (A_{21} + A_{22} - A_{11}) \cdot (B_{22} - B_{12} + B_{11})$$
$$M_2 = A_{11} \cdot B_{11}$$
$$M_3 = A_{12} \cdot B_{21}$$
$$M_4 = (A_{11} - A_{21}) \cdot (B_{22} - B_{11})$$
$$M_5 = (A_{21} + A_{22}) \cdot (B_{12} - B_{11})$$
$$M_6 = (A_{12} - A_{21} + A_{11} - A_{22}) \cdot B_{22}$$
$$M_7 = A_{22} \cdot (B_{11} + B_{22} - B_{12} - B_{21})$$

7 prodotti matrici $\frac{n}{2} \times \frac{n}{2}$

24 somme/sottrazioni matrici $\frac{n}{2} \times \frac{n}{2}$

e

$$C_{11} = M_2 + M_3$$
$$C_{12} = M_1 + M_2 + M_5 + M_6$$
$$C_{21} = M_1 + M_2 + M_4 - M_7$$
$$C_{22} = M_1 + M_2 + M_4 + M_5$$

$$T(n) = \begin{cases} 1 & \text{se } n = 1 \\ 7T\left(\frac{n}{2}\right) + 6n^2 & \text{altrimenti} \end{cases}$$

RETURN $C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$

$$T(n) = \Theta\left(n^{\log_2 7}\right) = \Theta\left(n^{2.81\ldots}\right)$$

$$24 \left(\frac{n}{2}\right)^2 = \cancel{24}^{6} \frac{n^2}{\cancel{4}}$$

upper bound migliore noto $n^{2.37\ldots}$