Moltiplicazione di Matrici

Sappiamo come funziona la **moltiplicazione tra valori**

$$a \cdot b = ab$$

Aumentando di una dimensione possiamo fare la **moltiplicazione di vettori**, dove i due vettori devono avere **dimensione uguale**

$$A = (a_1, \dots, a_n) \qquad B = (b_1, \dots, b_n)$$

$$A \cdot B = a_1 b_1 + \dots + a_n b_n$$
$$\iff A \cdot B = \sum_{k=1}^{n} a_k b_k$$

Aumentando di un'altra dimensione possiamo fare la **moltiplicazione di matrici**, dove la prima matrice deve avere **tante colonne quante le righe della seconda**

$$\underset{(m,n)}{A} = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \dots & a_{ij} & \dots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} \qquad \underset{(n,k)}{B} = \begin{pmatrix} b_{11} & \dots & a_{1k} \\ \dots & b_{ij} & \dots \\ a_{n1} & \dots & a_{nk} \end{pmatrix}$$

$$A \cdot B = \underset{(m,k)}{C} = \begin{pmatrix} c_{11} & \dots & c_{1k} \\ \dots & c_{ij} = (a_{i1}, \dots, a_{in}) \times (b_{1j}, \dots, b_{nj}) & \dots \\ c_{1m} & \dots & c_{mk} \end{pmatrix}$$
$$\iff c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

Algoritmo

SQUARE-MATRIX-MULTIPLY(A, B)

```
n = A.rows // matrici quadrate
let C be a new n x n matrix

for i = 1 to n
        for j = 1 to n
                c_ij = 0
                for k = 1 to n
                        c_ij = c_ij + (a_ik * b_kj)

return C
```

***Parametri**: A,B=matrici quadrate*
***Return**: matrice risultante*

Costo Computazionale

$$T(n) = \Theta(n^3)$$

Moltiplicazione Ricorsiva

Idea

Quando una matrice quadrata ha come dimensione una potenza del due $n = 2^i$, possiamo **dividere la matrice in 4 sottomatrici** con dimensione $\frac{n}{2}$ le quali formeranno le sottosezioni della risultante

$$n = 4 = 2^2$$

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \iff A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \qquad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

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

Proprietà: Contornamento della Matrice

Se una matrice non è quadrata o non ha dimensione equivalente ad una potenza del due, possiamo sempre **"contornarla" di zeri per modificarne la dimensione** senza modificare il risultato.

Nel caso sia necessario ritornare alla dimensione originale, basterà rimuovere nuovamente le righe e colonne aggiunte

RECMM(A, B)

```
if n = 1
        return a_11 * b_11

C_11 = RECMM(A_11, B_11) + RECMM(A_12, B_21)
C_12 = RECMM(A_11, B_12) + RECMM(A_12, B_22)
C_21 = RECMM(A_21, B_11) + RECMM(A_22, B_21)
C_22 = RECMM(A_21, B_12) + RECMM(A_22, B_22)

return C
```

***Parametri**: A,B=matrici quadrate*
***Return**: matrice risultante*

Costo Computazionale

L'equazione ricorsiva da risolvere sarà

$$T(n) = 8T\left(\frac{n}{2}\right) + \Theta(n^2)$$

L'albero della ricorsione quindi sarà

$$\begin{aligned} T(n) &= n^2 + 8T\left(\frac{n}{2}\right) \\ &= n^2 + 8\left(\frac{n}{2}\right)^2 + 64T\left(\frac{n}{4}\right) \\ &= \dots \end{aligned}$$

$$\begin{aligned} T(n) &= \dots \\ &= \cancel{8^{2^i}}\left(\frac{n^2}{\cancel{4^i}}\right) = 2^i n^2 \\ &= \dots \end{aligned}$$

L'albero avrà altezza

$$\frac{n^2}{4^h} = 1 \iff n^2 = 4^h \iff n = 2^h \iff h = \log_2(n)$$

Quindi

$$\begin{aligned} T(n) &= \sum_{i=0}^{\log_2(n)} 2^i n^2 \\ &= n^2 \sum_{i=0}^{\log_2(n)} 2^i \\ &= n^2 \Theta(2^{\log_2(n)}) \\ &= n^2 \Theta(n) \\ &= \Theta(n^3) \end{aligned}$$

Costo Computazionale

Abbiamo un costo computazionale **pari** alla moltiplicazione iterativa

$$T(n) = \Theta(n^3)$$

Algoritmo di Strassen

> **Idea**
>
> **Abbassare il costo computazionale** della moltiplicazione ricorsiva, aumentando le somme e diminuendo le moltiplicazioni

STRASSEN(A, B)

```
if n = 1
        return a_11 * b_11

P1 = STRASSEN(A_11, B_12 - B_22)
P2 = STRASSEN(A_11 + A_12, B_22)
P3 = STRASSEN(A_21 + A_22, B_11)
P4 = STRASSEN(A_22, B_21 - B_11)
P5 = STRASSEN(A_11 + A_22, B_11 - B_22)
P6 = STRASSEN(A_12 - A_22, B_21 + B_22)
P7 = STRASSEN(A_11 - A_21, B_11 - B_12)

C_11 = P5 + P4 - P2 + P6
C_12 = P1 + P2
C_21 = P3 + P4
C_22 = P5 + P1 - P3 - P7

return C
```

***Parametri**: A,B=matrici quadrate*
***Return**: matrice risultante*

Costo Computazionale

L'equazione ricorsiva sarà

$$T(n) = 7T\left(\frac{n}{2}\right) + \Theta(n^2)$$

Analogamente alla moltiplicazione ricorsiva, l'albero avrà altezza

$$\frac{n^2}{4^h} = 1 \iff n^2 = 4^h \iff n = 2^h \iff h = \log_2(n)$$

Quindi

$$\begin{aligned}
T(n) &= \sum_{i=0}^{\log_2(n)} \left(\frac{7}{4}\right)^i n^2 \\
&= n^2 \sum_{i=0}^{\log_2(n)} \left(\frac{7}{4}\right)^i \\
&= n^2 \Theta\left(\left(\frac{7}{4}\right)^{\log_2(n)}\right) \\
&= n^2 \Theta\left(\left(\frac{7}{4}\right)^{\log_{\frac{7}{4}}(n)\log_2\left(\frac{7}{4}\right)}\right) \\
&= n^2 \Theta\left(n^{\log_2\left(\frac{7}{4}\right)}\right) \\
&= \Theta\left(n^{2+\log_2\left(\frac{7}{4}\right)}\right) \\
&= \Theta\left(n^{\log_2(4)+\log_2\left(\frac{7}{4}\right)}\right) \\
&= \Theta(n^{\log_2(7)})
\end{aligned}$$

> **Costo Computazionale**
>
> Otteniamo un costo computazionale **leggermente minore** alle controparti iterative e ricorsive
>
> $$\Theta(n^{\log_2(7)}) = \Theta(n^{2.80735})$$