

Calcolo Parallelo e Distribuito

Prodotto Matrice-Vettore
Strategia 3
Legge di W-A

Docente: Prof. L. Marcellino

Tutor: Prof. P. De Luca

PROBLEMA: Prodotto Matrice-Vettore

Progettazione
di un algoritmo parallelo
per architettura MIMD

per il calcolo del prodotto
di una matrice A per un vettore b :

matrice A : N righe, M colonne
Vettore b : M elementi

III STRATEGIA

Decomposizione 1: BLOCCHI di RIGHE

+

Decomposizione 2: BLOCCHI di COLONNE

=

Decomposizione 3: BLOCCHI Righe&Colonne

Abbiamo calcolato...

Calcolo **di speedup ed efficienza** (def classica)

... in tutte le possibilità!

Abbiamo calcolato...

isoefficienza

Ma...

I conti devono essere rifatti nel caso in cui
 $\text{mod}(N, q) \neq 0$ e/o $\text{mod}(M, p) \neq 0$

... molte altre osservazioni si potrebbero aggiungere, ma quelle le faremo all'esame!

Calcolo **di speedup**
(def Ware Amdahl-generalizzata)

III Strategia: speed-up/efficienza (**W-A**)

matrice A: N righe, M colonne
Vettore b: M elementi

In sequenziale:

$$T_1(NM) = N[2M-1] \text{ operazioni}$$

Per calcolare lo speedup con la legge di W-A, la prima domanda che mi devo fare è se per questa strategia di parallelizzazione posso esattamente distinguere la parte parallela
(nella fase di calcolo locale lavorano tutti i processori)
e la parte sequenziale
(la collezione dei risultati avviene in maniera sequenziale)

Quasi sempre fasi a parallelismo medio

$$S_p = \frac{1}{\alpha_1 + \sum_{k=2}^{p-1} \frac{\alpha_k}{k} + \frac{\alpha_p}{p}}$$

III Strategia: speed-up/efficienza (**W-A**)

matrice A: N righe, M colonne
Vettore b: M elementi

In sequenziale:

$$T_1(NM) = N[2M-1] \text{ operazioni}$$

In parallelo:

1 fase (tutta parallela)

Calcolo prodotti parziali

$N/q [2M/p - 1]$ operazioni



contemporaneamente

fatto da $q \times p$ processori/core

$q \times p \frac{N}{q} [2M/p - 1]$
delle $N[2M-1]$ operazioni



$$\alpha_{q \times p} = q \times p \frac{\frac{N}{q} [\frac{2M}{p} - 1]}{N[2M - 1]}$$

III Strategia: speed-up/efficienza (**W-A**)

matrice A: N righe, M colonne

Vettore b: M elementi

In sequenziale:

$$T_1(NM) = N[2M-1] \text{ operazioni}$$

**Per procedere con i calcoli bisogna
fissare dei valori per q e p**

III Strategia:

Collezione dei risultati

$6 = q \times p = 2 \times 3$ $p = 3$
I strategia

P_{00}	P_{01}	P_{02}
P_{10}	P_{11}	P_{12}

$8 = q \times p = 2 \times 4$ $p = 4$
II strategia

P_{00}	P_{01}	P_{02}	P_{03}
P_{10}	P_{11}	P_{12}	P_{13}

III Strategia:

Collezione dei risultati

$6 = q \times p = 2 \times 3$ $p = 3$
I strategia

P_{00}

P_{01}

P_{02}

P_{10}

P_{11}

P_{12}

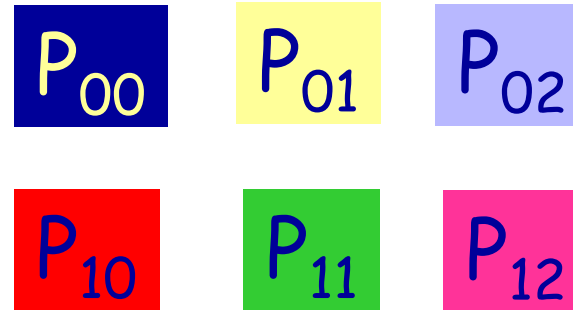
III Strategia: speed-up/efficienza (**W-A**)

In sequenziale:

$$T_1(NM) = N[2M-1]$$

$$6=q \times p=2 \times 3 \quad p=3$$

matrice A: N righe, M colonne
Vettore b: M elementi



In parallelo:

1 fase (tutta parallela)

Calcolo prodotti parziali

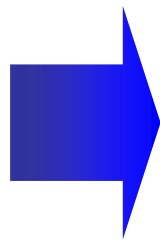
$N/q [2M/p - 1]$ operazioni



contemporaneamente

fatto da $q \times p$ processori/core

$q \times p \ N/q [2M/p - 1]$
delle $N[2M-1]$ operazioni



$$\alpha_6 = 6 \frac{\frac{N}{2} [\frac{2M}{3} - 1]}{N[2M-1]}$$

III Strategia:

Collezione dei risultati

$6 = q \times p = 2 \times 3$ $p = 3$
I strategia

P_{00}	P_{01}	P_{02}
P_{10}	P_{11}	P_{12}

Parallelismo medio:

Non esiste nessuna fase in cui lavorano contemporaneamente 5, 4, 3 processori/core



$$\alpha_5 = \alpha_4 = \alpha_3 = 0$$

III Strategia:

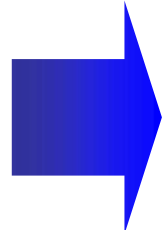
Collezione dei risultati

$6=q \times p=2 \times 3$ $p=3$
I strategia

P_{00}	P_{01}	P_{02}
P_{10}	P_{11}	P_{12}

Parallelismo medio:

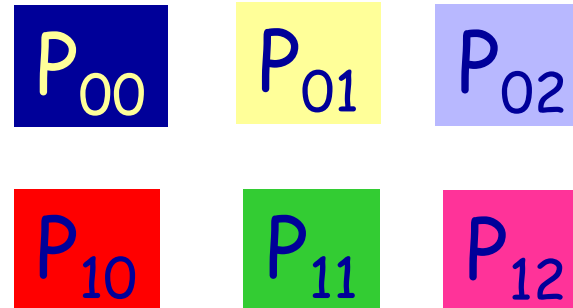
Esiste, invece, una fase in cui, i due processori/core lavorano in sequenziale per effettuare la somma dei contributi, ovvero vettori di dimensione $N/2$ (lungo le colonne) contemporaneamente (per 2)


$$\alpha_2 = 2 \frac{(3-1) \frac{N}{2}}{N[2M-1]}$$

III Strategia:

Collezione dei risultati

$6=q \times p=2 \times 3$ $p=3$
I strategia



Parallelismo medio:

È la fase in cui, i due processori P_{00} e P_{10} della griglia calcolano la somma dei contributi (per le colonne) contemporaneamente. in
Tutti i passi della I strategia

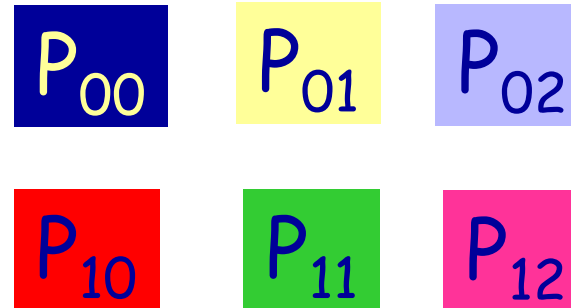


$$\alpha_2 = 2 \frac{(3-1) \frac{N}{2}}{N[2M-1]}$$

III Strategia:

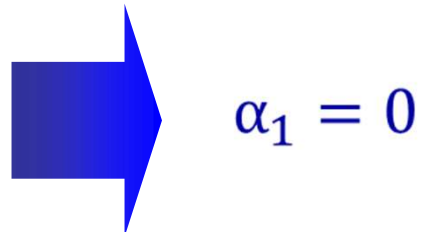
Collezione dei risultati

$6 = q \times p = 2 \times 3$ $p = 3$
I strategia



Parallelismo medio:

Non esiste nessuna fase in cui lavora 1 processore/core



III Strategia:

Collezione dei risultati

$6 = q \times p = 2 \times 3$ $p = 3$
I strategia

P_{00}	P_{01}	P_{02}
P_{10}	P_{11}	P_{12}

Attenzione:

Per essere sicura di aver fatto bene i conti devo sommare tutti i numeratori e ottenere un numero uguale al denominatore comune!

$$\alpha_6 = 6 \frac{\frac{N}{2} \left[\frac{2M}{3} - 1 \right]}{N[2M - 1]} \quad \alpha_5 = \alpha_4 = \alpha_3 = 0$$
$$\alpha_2 = 2 \frac{(3 - 1) \frac{N}{2}}{N[2M - 1]} \quad \alpha_1 = 0$$

III Strategia:

Collezione dei risultati

$6=q \times p=2 \times 3$ $p=3$
I strategia

P_{00}	P_{01}	P_{02}
P_{10}	P_{11}	P_{12}

Non resta che sostituire i valori
calcolati nella formula generalizzata

$$S_p = \frac{1}{\alpha_1 + \sum_{k=2}^{p-1} \frac{\alpha_k}{k} + \frac{\alpha_p}{p}}$$

III Strategia:

Collezione dei risultati

$8 = q \times p = 2 \times 4$ $p = 4$
II strategia

P_{00}	P_{01}	P_{02}	P_{03}
P_{10}	P_{11}	P_{12}	P_{13}

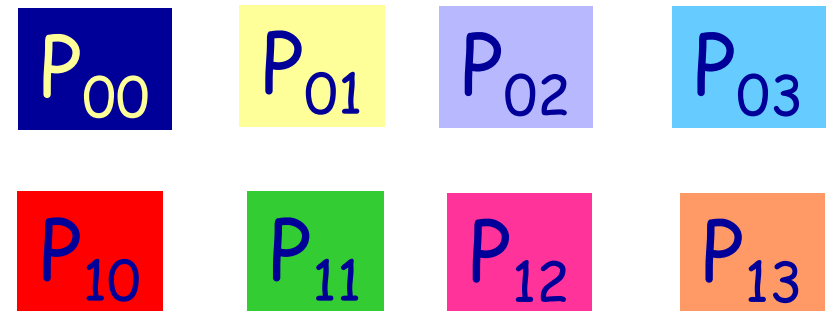
III Strategia: speed-up/efficienza (**W-A**)

In sequenziale:

$$T_1(NM) = N[2M-1]$$

matrice A: N righe, M colonne
Vettore b: M elementi

$$8=q \times p=2 \times 4 \quad p=4$$



In parallelo:

1 fase (tutta parallela)

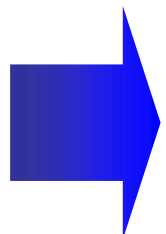
Calcolo prodotti parziali

$N/q [2M/p - 1]$ operazioni

contemporaneamente

fatto da $q \times p$ processori/core

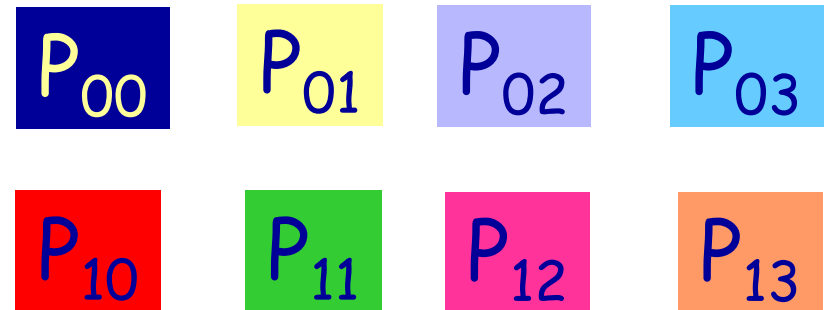
$q \times p \ N/q [2M/p - 1]$
delle $N[2M-1]$ operazioni


$$\alpha_8 = 8 \frac{\frac{N}{2} [\frac{2M}{4} - 1]}{N[2M-1]}$$

III Strategia:

Collezione dei risultati

$8 = q \times p = 2 \times 4$ $p = 4$
II strategia



Parallelismo medio:

Non esiste nessuna fase in cui lavorano contemporaneamente 7, 6, 5 processori/core



$$\alpha_7 = \alpha_6 = \alpha_5 = 0$$

III Strategia:

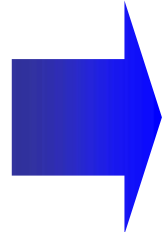
Collezione dei risultati

$8=q \times p=2 \times 4$ $p=4$
II strategia

P_{00}	P_{01}	P_{02}	P_{03}
P_{10}	P_{11}	P_{12}	P_{13}

Parallelismo medio:

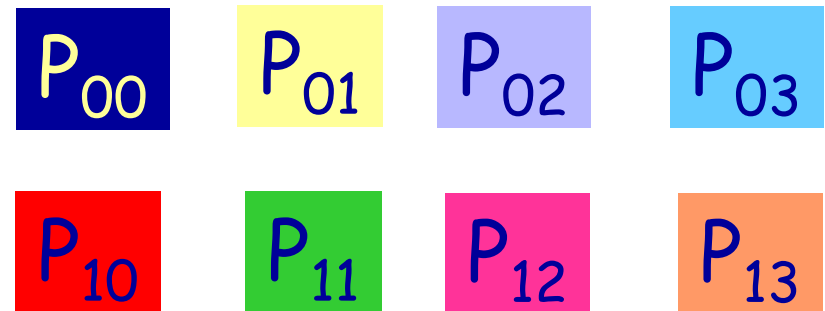
Esiste, invece, una fase (1passo II strategia) in cui quattro processori/core lavorano contemporaneamente per effettuare la somma parziale dei contributi, ovvero vettori di dimensione $N/2$ (lungo le colonne)


$$\alpha_4 = 4 \frac{\frac{N}{2}}{N[2M - 1]}$$

III Strategia:

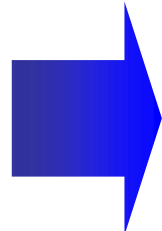
Collezione dei risultati

$8=q \times p=2 \times 4$ $p=4$
II strategia



Parallelismo medio:

Processori $P_{00}, P_{10}, P_{02}, P_{12}$ della griglia
fase (1 passo II strategia)
contemporaneamente p
vettori, ovvero vettori di dimer
Primo passo della II strategia

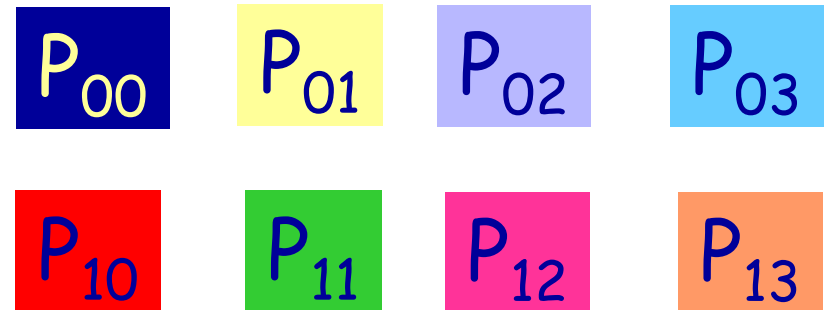


$$\alpha_4 = 4 \frac{\frac{N}{2}}{N[2M - 1]}$$

III Strategia:

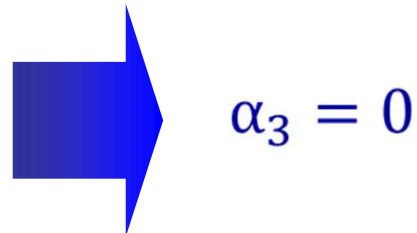
Collezione dei risultati

$8=q \times p=2 \times 4$ $p=4$
II strategia



Parallelismo medio:

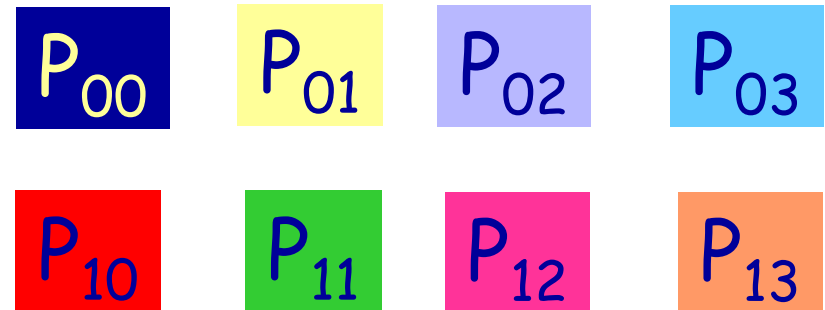
Non esiste nessuna fase in cui lavorano contemporaneamente 3 processori/core



III Strategia:

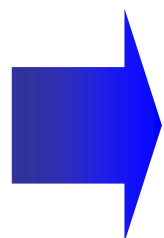
Collezione dei risultati

$8=q \times p=2 \times 4$ $p=4$
II strategia



Parallelismo medio:

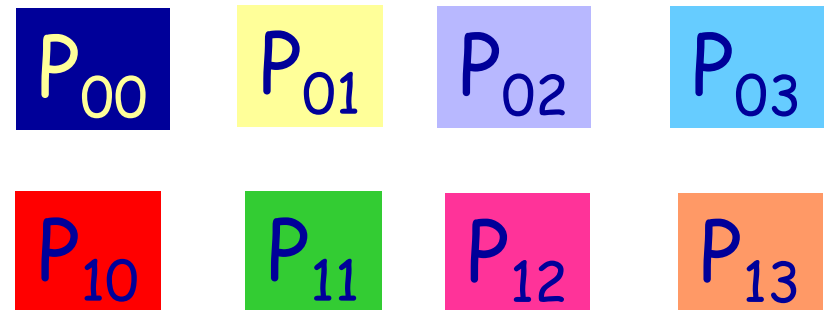
Esiste, infine, una fase (2passo II strategia) in cui due processori/core lavorano contemporaneamente per effettuare la somma parziale dei contributi al 1passo, ovvero vettori di dimensione $N/2$ (lungo le colonne)


$$\alpha_2 = 2 \frac{\frac{N}{2}}{N[2M - 1]}$$

III Strategia:

Collezione dei risultati

$8=q \times p=2 \times 4$ $p=4$
II strategia



Parallelismo medio:

È la seconda fase (2passo II strategia) in cui due processori contigui della griglia (ad esempio P_{00} e P_{10}) lavorano contemporaneamente per effettuare la riduzione dei dati, ovvero vettori di dimensione $\frac{N}{2}$.

Processori
 P_{00} , P_{10}
della griglia

Secondo passo
della II strategia

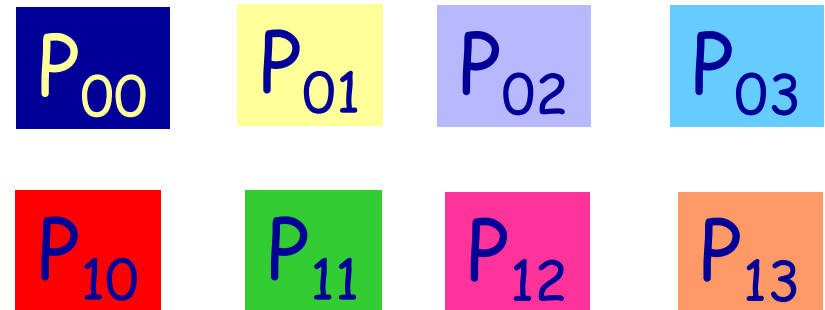


$$\alpha_2 = 2 \frac{\frac{N}{2}}{N[2M - 1]}$$

III Strategia:

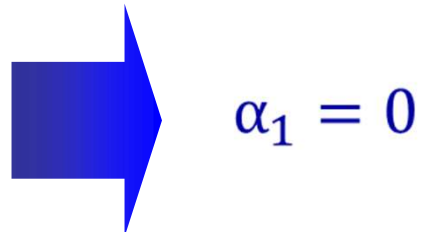
Collezione dei risultati

$8=q \times p=2 \times 4$ $p=4$
II strategia



Parallelismo medio:

Non esiste nessuna fase in cui lavora 1 processore/core



III Strategia:

Collezione dei risultati

$8=q \times p=2 \times 4$ $p=4$
II strategia

P_{00}	P_{01}	P_{02}	P_{03}
P_{10}	P_{11}	P_{12}	P_{13}

Attenzione:

Per essere sicura di aver fatto bene i conti devo sommare tutti i numeratori ed ottenere un numero uguale al denominatore comune!

$$\alpha_8 = 8 \frac{\frac{N}{2} \left[\frac{2M}{4} - 1 \right]}{N[2M - 1]} \quad \alpha_7 = \alpha_6 = \alpha_5 = 0 \quad \alpha_2 = 2 \frac{\frac{N}{2}}{N[2M - 1]}$$
$$\alpha_4 = 4 \frac{\frac{N}{2}}{N[2M - 1]} \quad \alpha_1 = 0$$
$$\alpha_3 = 0$$

III Strategia:

Collezione dei risultati

$8=q \times p=2 \times 4$ $p=4$
II strategia

P_{00}	P_{01}	P_{02}	P_{03}
P_{10}	P_{11}	P_{12}	P_{13}

Non resta che sostituire i valori
calcolati nella formula generalizzata

$$S_p = \frac{1}{\alpha_1 + \sum_{k=2}^{p-1} \frac{\alpha_k}{k} + \frac{\alpha_p}{p}}$$

III Strategia: speed-up/efficienza (**W-A**)

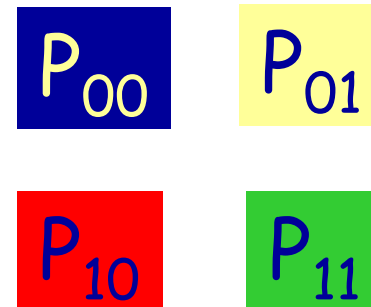
Ultimo esercizio:

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

In sequenziale:

$7[2 \cdot 9 - 1] = 119$ operazioni

$$4 = q \times p = 2 \times 2 \quad p = 2$$



III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

In sequenziale:

$7[2 \cdot 9 - 1] = 119$ operazioni

$4 = q \times p = 2 \times 2$

Vediamo le dimensioni che ogni processore/core deve trattare

$$\dim[A_{00}] = (7/2 + 1) \times (9/2 + 1)$$

$$\dim[b_0] = 9/2 + 1$$

$$\dim[A_{10}] = (7/2) \times (9/2 + 1),$$

$$\dim[A_{01}] = (7/2 + 1) \times (9/2)$$

$$\dim[A_{11}] = (7/2) \times (9/2)$$

$$\dim[b_1] = 9/2$$

P_{00}

P_{01}

P_{10}

P_{11}

III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

In sequenziale:

$7[2 \cdot 9 - 1] = 119$ operazioni

$4 = q \times p = 2 \times 2$

Vediamo le dimensioni che ogni processore/core deve trattare:

$$\dim[A_{00}] = 4 \times 5$$

$$\dim[b_0] = 5$$

$$\dim[A_{10}] = 3 \times 5, \dim[A_{01}] = 4 \times 4$$

$$\dim[A_{11}] = 3 \times 4$$

$$\dim[b_1] = 4$$

P_{00}

P_{01}

P_{10}

P_{11}

III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

$7[2 \cdot 9 - 1] = 119$ operazioni

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

$4 = q \times p = 2 \times 2$

In parallelo:

Ognuno dei 4 processori effettua un prodotto di tipo matrice-vettore con strutture di diverse dimensioni, riportando i valori nell'ordine:



P_{00}

P_{01}

P_{10}

P_{11}

$$\alpha_4 = \frac{1 \cdot 4[2 \cdot 5 - 1] + 1 \cdot 4[2 \cdot 4 - 1] + 1 \cdot 3[2 \cdot 5 - 1] + 1 \cdot 3[2 \cdot 4 - 1]}{119}$$

III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

$7[2 \cdot 9 - 1] = 119$ operazioni

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

$4 = q \times p = 2 \times 2$

Parallelo parziale (I-II strategia stessi conti se $p=2$):

Nessuna fase in cui lavorano contemporaneamente 3
processori/core



$$\alpha_3 = 0$$

III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

$7[2 \cdot 9 - 1] = 119$ operazioni

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

$$4 = q \times p = 2 \times 2$$

Parallelo parziale (I-II strategia stessi conti se $p=2$):

Due processori/core (sulle due righe), contemporaneamente effettuano la somma dei contributi dei vettori distribuiti lungo le colonne

$$\dim[A_{00}] = 4 \times 5$$

$$\dim[b_0] = 5$$

$$\dim[A_{10}] = 3 \times 5, \dim[A_{01}] = 4 \times 4$$

$$\dim[A_{11}] = 3 \times 4$$

$$\dim[b_1] = 4$$

P_{00}

P_{01}

$$4 = \dim[r_0] = \dim[s_0]$$

P_{10}

P_{11}

$$3 = \dim[r_1] = \dim[s_1]$$

III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

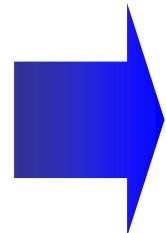
$7[2 \cdot 9 - 1] = 119$ operazioni

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

$4 = q \times p = 2 \times 2$

Parallelo parziale (I-II strategia stessi conti se $p=2$):

Due processori/core (sulle due righe), contemporaneamente effettuano la somma dei contributi dei vettori distribuiti lungo le colonne


$$\alpha_2 = \frac{\overset{\boxed{P_{00}}}{1} \cdot 4 + \overset{\boxed{P_{10}}}{1} \cdot 3}{119} = \frac{7}{119}$$

III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

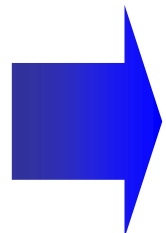
$7[2 \cdot 9 - 1] = 119$ operazioni

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

$4 = q \times p = 2 \times 2$

Parallelo parziale (I-II strategia stessi conti se $p=2$):

Nessuna fase sequenziale


$$\alpha_1 = 0$$

III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

$7[2 \cdot 9 - 1] = 119$ operazioni

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

$4 = q \times p = 2 \times 2$

Attenzione:

Per essere sicura di aver fatto bene i conti devo sommare tutti i numeratori e trovarmi il denominatore comune!

$$\alpha_4 = \frac{112}{119} \quad \alpha_3 = 0 \quad \alpha_2 = \frac{7}{119} \quad \alpha_1 = 0$$

III Strategia: speed-up/efficienza (**W-A**)

Ultimo esercizio:

$7[2 \cdot 9 - 1] = 119$ operazioni

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

$4 = q \times p = 2 \times 2$

**Non resta che sostituire i valori
calcolati nella formula generalizzata**

$$S_p = \frac{1}{\alpha_1 + \sum_{k=2}^{p-1} \frac{\alpha_k}{k} + \frac{\alpha_p}{p}}$$

matrice A: 7 righe, 9 colonne
Vettore b: 9 elementi

Esercizio svolto in aula

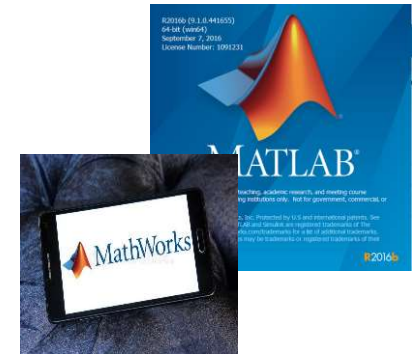
$$8=q \times p=4 \times 2 \quad p=2$$

Lo so che è la migliore!!!

P_{00}	P_{01}
P_{10}	P_{11}
P_{20}	P_{21}
P_{30}	P_{31}

Prodotto Matrice-Vettore: la strategia di parallelizzazione implementazione semplificata del codice

In ambiente **MATLAB**



Parallel Computing Toolbox

- insieme di **funzioni** MATLAB, di alto livello e facile utilizzo
- insieme di **costrutti linguistici** per risolvere problemi di calcolo intensivo e grande quantità di dati, utilizzando processori **multicore** e **GPU**

Il Toolbox

- messa a disposizione degli utenti dal 2016 con la release MATLAB: R2016b

Prodotto Matrice-Vettore:

I strategia di parallelizzazione

implementazione semplificata del codice

In ambiente **MATLAB**

➤ Attualmente nel Parallel Computing Toolbox, è supportata una opportuna sezione che permette di gestire il numero **di Worker (thread)** tra cui è possibile dividere il carico di lavoro, al fine di ottimizzarne le prestazioni.



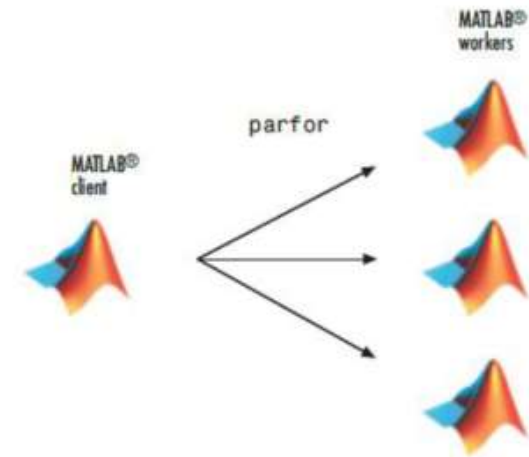
**worker
(thread)**

Prodotto Matrice-Vettore:

la strategia di parallelizzazione

implementazione semplificata del codice

In ambiente **MATLAB**



MATLAB permette di generare un gruppo di unità di lavoro indipendenti, detti **worker** (il carico di lavoro di ogni worker è distribuito tra i core).

Il modo più semplice per farlo è utilizzare il costrutto:

parpool

Per *lanciare* un pool di worker si utilizza la funzione **parpool**: il valore restituito è una variabile di tipo pool

Prodotto Matrice-Vettore:

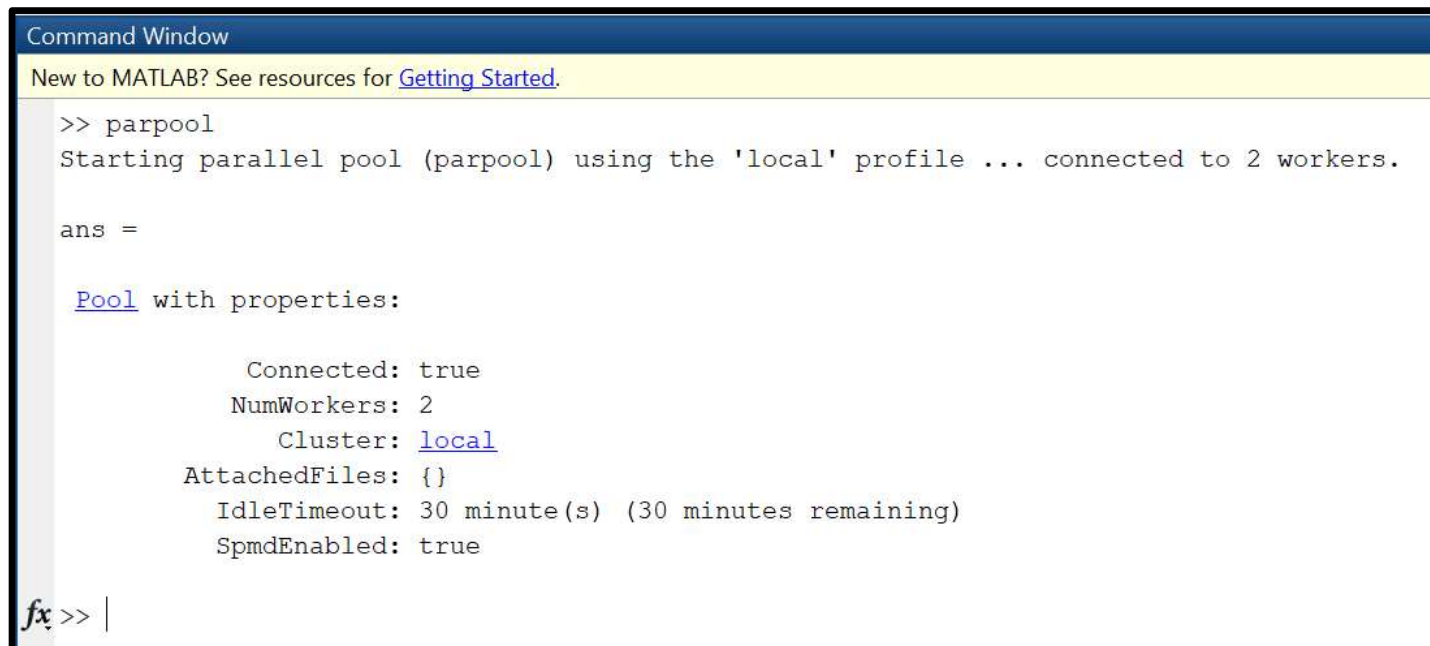
I strategia di parallelizzazione

implementazione semplificata del codice

In ambiente **MATLAB**

Il costrutto *parpool*

[è possibile che il parallel Computing Toolbox debba essere istallato]



```
Command Window
New to MATLAB? See resources for Getting Started.

>> parpool
Starting parallel pool (parpool) using the 'local' profile ... connected to 2 workers.

ans =

    Pool with properties:
        Connected: true
        NumWorkers: 2
        Cluster: local
        AttachedFiles: {}
        IdleTimeout: 30 minute(s) (30 minutes remaining)
        SpmdEnabled: true

fx >> |
```

La funzione accetta come parametri in ingresso *un profilo* e *il numero di worker*; se non vengono specificati, viene utilizzato il profilo locale con un numero di worker pari al numero di core (fisici) della macchina.

Prodotto Matrice-Vettore:

la strategia di parallelizzazione

implementazione semplificata del codice

In ambiente **MATLAB**

Per terminare l'esecuzione in parallelo si utilizza il comando **delete(gcp)**.

```
>> parpool
Starting parallel pool (parpool) using the 'local' profile ... co

ans =

    Pool with properties:

        Connected: true
      NumWorkers: 2
         Cluster: local
AttachedFiles: {}|
  IdleTimeout: 30 minute(s) (30 minutes remaining)
    SpmdEnabled: true

>> delete(gcp)
Parallel pool using the 'local' profile is shutting down.
>> |
```

Prodotto Matrice-Vettore:

I strategia di parallelizzazione

implementazione semplificata del codice

In ambiente **MATLAB**

Cicli *for* in parallelo: il costrutto *parfor*

Il più semplice tipo di parallelismo su architettura multicore con MATLAB può essere definito con un ciclo *for* parallelo, utilizzando il costrutto:

parfor

il numero di cicli verrà effettuato in parallelo distribuendo il lavoro ai *worker*, questi vengono creati (al massimo pari al numero di core logici della macchina).

Prodotto Matrice-Vettore:

la strategia di parallelizzazione

implementazione semplificata del codice

In ambiente **MATLAB**

```
>> A=rand(500,500);  
>> x=rand(1,500);  
>> y=A*x'
```

Una versione
sequenziale
più basso
livello

```
>> n=500;  
>> for i=1:n  
y(i)=A(i , :) * x'  
end
```

versione
parallela

```
>> parfor i=1:n  
y(i) = A(i, : ) * x'  
end  
>>
```

Prodotto Matrice-Vettore:

la strategia di parallelizzazione

implementazione semplificata del codice

In ambiente **MATLAB**

Attenzione:

Il ciclo *parfor* ha però delle limitazioni:

- le istruzioni eseguite al suo interno devono essere indipendenti;
- i valori di iterazione devono essere interi consecutivi

Valori di iterazione	Validità
<code>parfor i = 1 : 100</code>	Valido
<code>parfor i = -20 : 20</code>	Valido
<code>parfor i = 1 : 2 : 25</code>	Non valido
<code>parfor i = -7.5 : 7.5</code>	Non valido