



Calcolo Parallelo e Distribuito

I parametri di valutazione di un algoritmo parallelo

Docente: Prof. L. Marcellino

Tutor: Prof. P. De Luca



Previously on... CPD

Previously on... CPD

- **Classificazione di Flynn**
 - Temporale → SIDS/MISD
 - Spaziale → SIMD
 - **asincrono → MIMD (SM) multicore**
- **Macchine multicore e libreria openMP per la programmazione parallela in questo ambiente**
- **I° nucleo computazionale: somma tra due vettori di lunghezza N – decomposizione del dominio (**algoritmi full parallel**)**
- **II° nucleo computazionale: somma di N numeri – decomposizione del dominio – collezione dei risultati (**1-2 strategia**)**

Ho le strategie per la parallelizzazione, ho gli strumenti (macchina multicore e libreria openMP)...



...vale la pena implementare in un software la strategia di parallelizzazione pensata per l'algoritmo?



Analisi algoritmo parallelo

Analisi algoritmo parallelo

Valutare l'efficienza di un **algoritmo**
in ambiente di calcolo parallelo



Cosa si intende per
EFFICIENZA DI UN ALGORITMO ?

Efficienza di un algoritmo sequenziale

- **COMPLESSITA' COMPUTAZIONALE $T(N)$**
Numero di operazioni eseguite dall'algoritmo

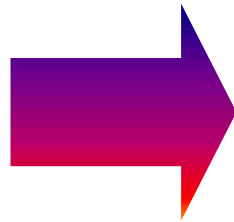
- **COMPLESSITA' SPAZIALE $S(N)$**
Numero di variabili utilizzate dall'algoritmo

Di cosa stiamo parlando?

$$\tau = k \cdot T(N) \cdot \mu$$

μ = tempo di esecuzione di 1 op. f.p.

$$T(N) = T_1(N)$$



$$T_1(N) = \text{complessità computazionale} \\ \text{1 processore}$$

Tempo di
esecuzione
software
sequenziale

$$\tau_1 = T_1(N) \cdot K$$

$t_{\text{calc}} = \mu$ tempo di esecuzione di
una addizione

somma di due vettori di dimensione N

Su un calcolatore monoprocesso la somma è calcolata eseguendo le N addizioni una per volta secondo un ordine prestabilito

$$c_0 := a_0 + b_0$$

$$c_1 := a_1 + b_1$$

...

...

$$c_{N-1} := a_{N-1} + b_{N-1}$$



Esempio:

calcolo della **somma di vettori di dimensione $N=16$**

ALGORITMO SEQUENZIALE

numero di addizioni = 16

passi temporali = 16

a



+

b



Passi
temporali $i=0,\dots,15$

complessità di
tempo

$T(N)=N$ addizioni

In un algoritmo sequenziale

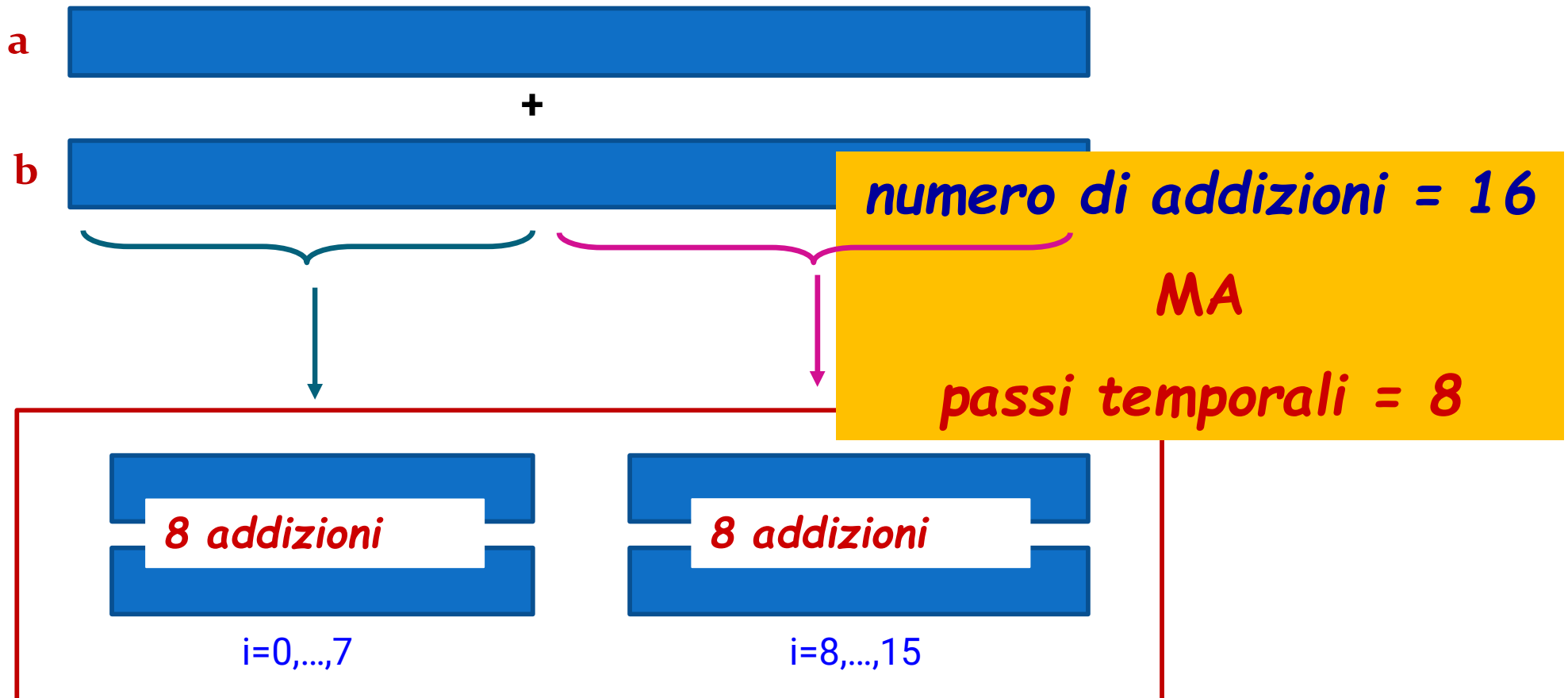
Il numero complessivo di operazioni
è uguale al
numero dei passi temporali

Esempio:

calcolo della **somma di vettori di dimensione $N=16$**

ALGORITMO PARALLELO

2 core

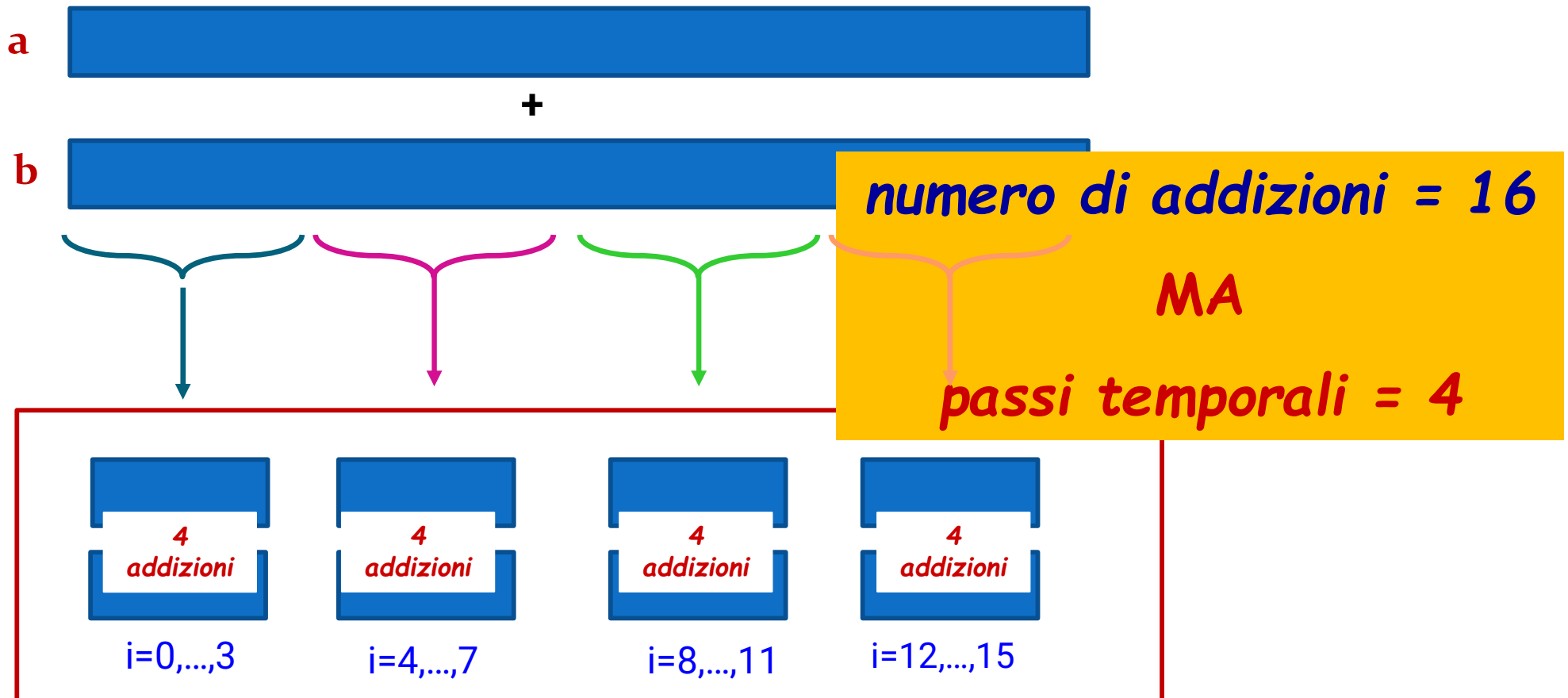


Esempio:

calcolo della **somma di vettori di dimensione $N=16$**

ALGORITMO PARALLELO

4 core



Nell'algoritmo parallelo della somma

Il numero delle operazioni
non è uguale
al numero dei passi temporali

Infatti ...

Un calcolatore parallelo è in grado di eseguire più operazioni

concorrentemente

(allo stesso passo temporale)



Il tempo di esecuzione **non è proporzionale** alla complessità di tempo
(ovvero **non dipende soltanto** dal numero di operazioni fl. p. effettuate)



La complessità di tempo **non è adatta** a misurare
l'efficienza di un algoritmo parallelo

... e allora

Che cosa si intende per
efficienza
di un algoritmo
in ambiente parallelo?

In generale

Con p core ci aspettiamo che T_1 sia
 p volte T_p

ovvero

ci aspettiamo di ridurre di
 p volte il tempo di
esecuzione

In questo caso le cose vanno proprio così...

Misuriamo di quanto **si riduce** il tempo di
esecuzione su **p** processori
rispetto al tempo di esecuzione
su **1** processore...

Esempio:

calcolo della **somma di vettori di dimensione $N=16$**

ALGORITMO SEQUENZIALE

numero di addizioni = 16

passi temporali = 16

a



+

b



Passi
temporali $i=0,\dots,15$

$$T_1(16)=16 \ t_{\text{calc}}$$

complessità di
tempo

$$T(N)=N \text{ addizioni}$$

Esempio:

calcolo della **somma di vettori di dimensione $N=16$**

ALGORITMO PARALLELO
2 core

numero di addizioni = 16
MA

a 
+

passi temporali = 8

b 

$$T_2(16) = 8 t_{\text{calc}}$$



$i=0,\dots,7$



$i=8,\dots,15$

complessità
computazionale
2 processori

Esempio:

calcolo della **somma di vettori di dimensione $N=16$**

ALGORITMO PARALLELO

4 core

numero di addizioni = 16

MA

passi temporali = 4

a



+

b



$$T_4(16) = 4 t_{\text{calc}}$$



$i=0, \dots, 3$



$i=4, \dots, 7$



$i=8, \dots, 11$



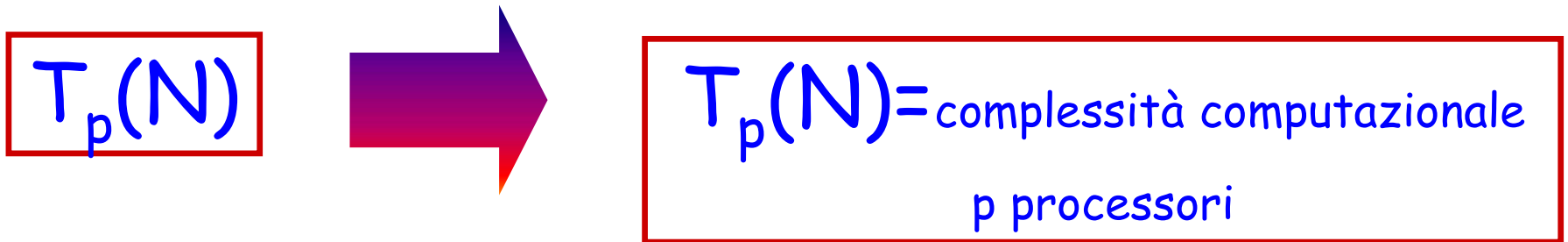
$i=12, \dots, 15$

complessità
computazionale
4 processori

Attenzione:

$$\tau = k \cdot T_p(N) \cdot \mu$$

μ = tempo di esecuzione di 1 op. f.p.



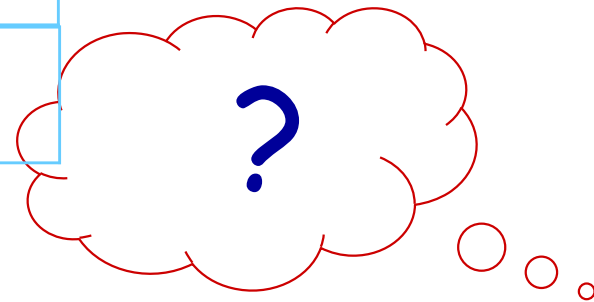
Tempo di
esecuzione
software
parallelo

$$\tau_p = T_p(N) \cdot K$$

$t_{calc} = \mu$ tempo di esecuzione di
una addizione

In sintesi...

p	$T_p(16)$
1	$16 t_{\text{calc}}$
2	$8 t_{\text{calc}}$
4	$4 t_{\text{calc}}$
8	$2 t_{\text{calc}}$
p	?



In generale quanto vale T_p ?

In generale: calcolo di $T_p(N)$

ALGORITMO PARALLELO della somma di due
vettori di dimensione N - posto p processori

$$p=1 \quad T_1=16 \, t_{\text{calc}}$$

$$p=2 \quad T_2=8 \, t_{\text{calc}}$$

$$p=4 \quad T_4=4 \, t_{\text{calc}}$$

$$p=8 \quad T_8=2 \, t_{\text{calc}}$$

.....

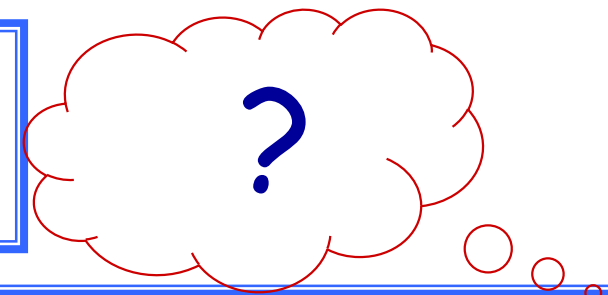
$$N = 16$$

$$T_p(N) = (N/p) \, t_{\text{calc}}$$

Domanda...

p	T_p
1	$16 t_{\text{calc}}$
2	$8 t_{\text{calc}}$
4	$4 t_{\text{calc}}$
8	$2 t_{\text{calc}}$

Qual è l'algoritmo che impiega meno tempo?

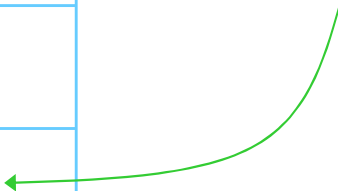


Quanto è più veloce di quello sequenziale?

Esempio: somma di due vettori di dim $N=16$

p	T_p	T_1/T_p
1	$16 t_{\text{calc}}$	1.00
2	$8 t_{\text{calc}}$	2
4	$4 t_{\text{calc}}$	4
8	$2 t_{\text{calc}}$	8

Maggiore
riduzione del
tempo ovvero
maggiore
aumento della
velocità



L'algoritmo su 8 processori è il più veloce
E' più veloce di 8 volte di quello su 1 processore

Speed-up

Si definisce il rapporto T_1 su T_p

$$S_p = \frac{T_1}{T_p}$$

Lo speed up misura la riduzione del
tempo di esecuzione rispetto
all'algoritmo su 1 core

$$S_p < p$$

$$\left(\begin{array}{l} \text{SPEEDUP IDEALE} \\ S_p^{ideale} = p \end{array} \right)$$



Esempio:

calcolo della **somma** di due vettori di dimensione **N**

L'algoritmo parallelo
ha speedup ideale!

$$\left[\begin{array}{l} \text{SPEEDUP IDEALE} \\ s_p^{\text{ideale}} = p \end{array} \right]$$

a



+

b



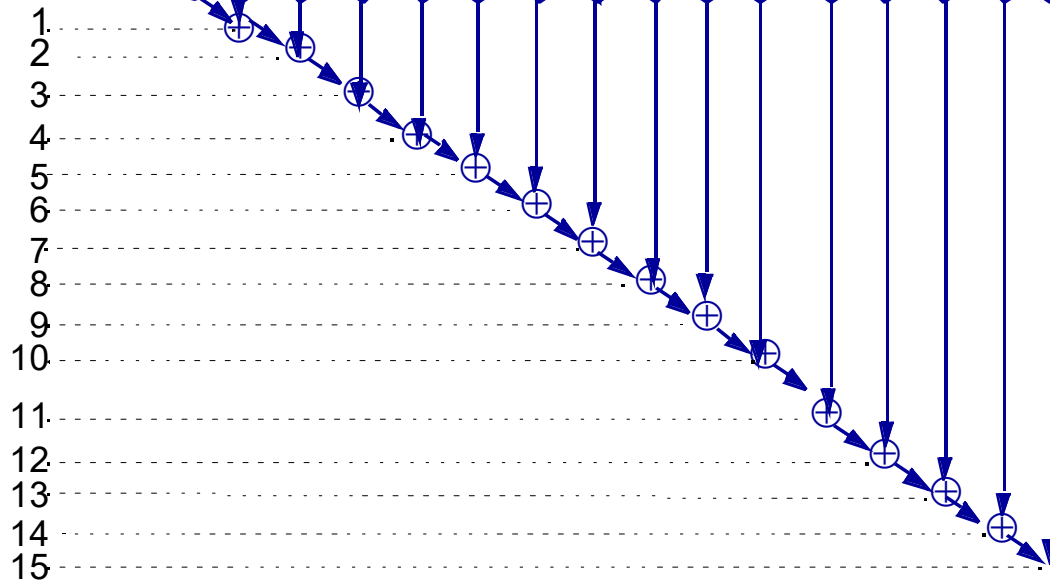
Gli algoritmi full parallel hanno tutti speedup ideale!

*Purtroppo non tutti i problemi possono essere
risolti con algoritmi completamente
parallelizzabili*

Esempio: calcolo della somma di $N=16$ numeri

ALGORITMO SEQUENZIALE

Passi
temporali



numero di addizioni = 15

passi temporali = 15

complessità di
tempo

$T(N) = N - 1$ addizioni

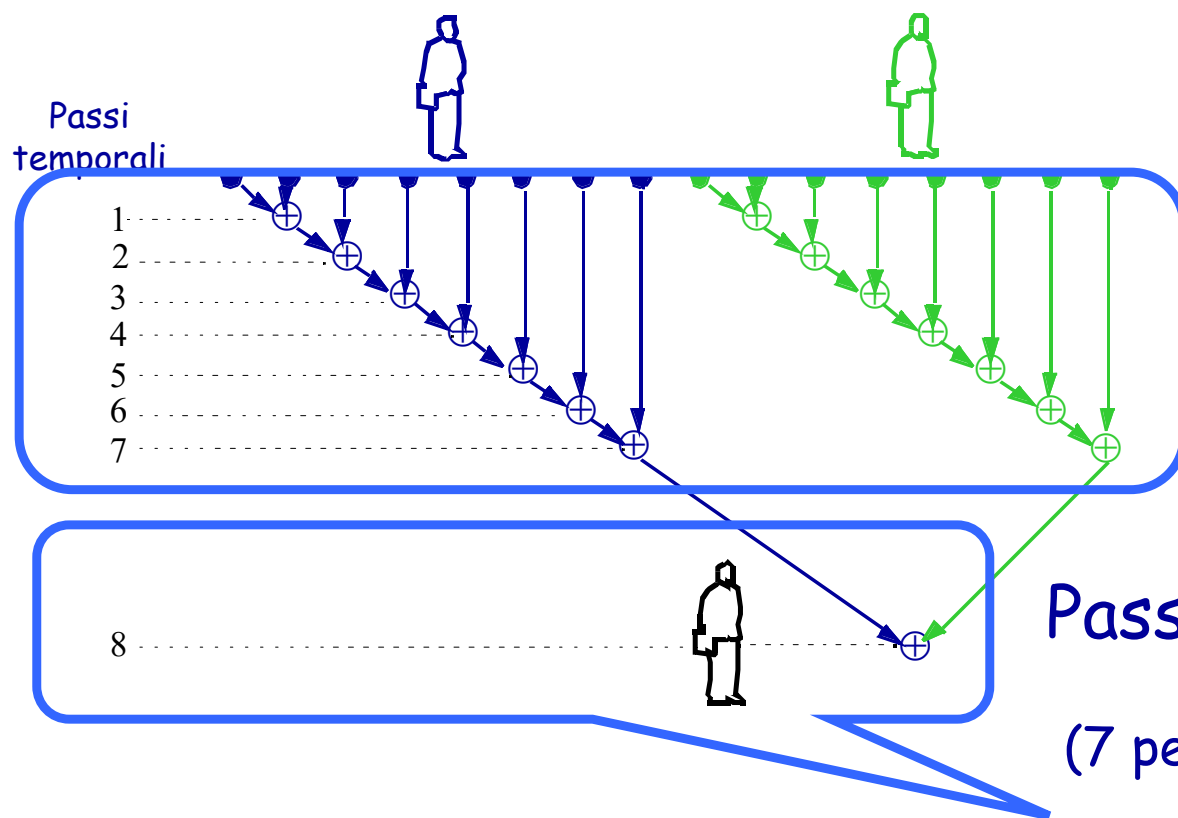
Esempio: calcolo della somma di $N=16$ numeri

ALGORITMO PARALLELO

numero di addizioni = 15

MA

passi temporali = 8



Passi temporali 1-7: **14 addizioni**
(7 per ciascuno core)

• Passo temporale 8: **1 addizione**

Esempio: calcolo della somma di $N=16$ numeri

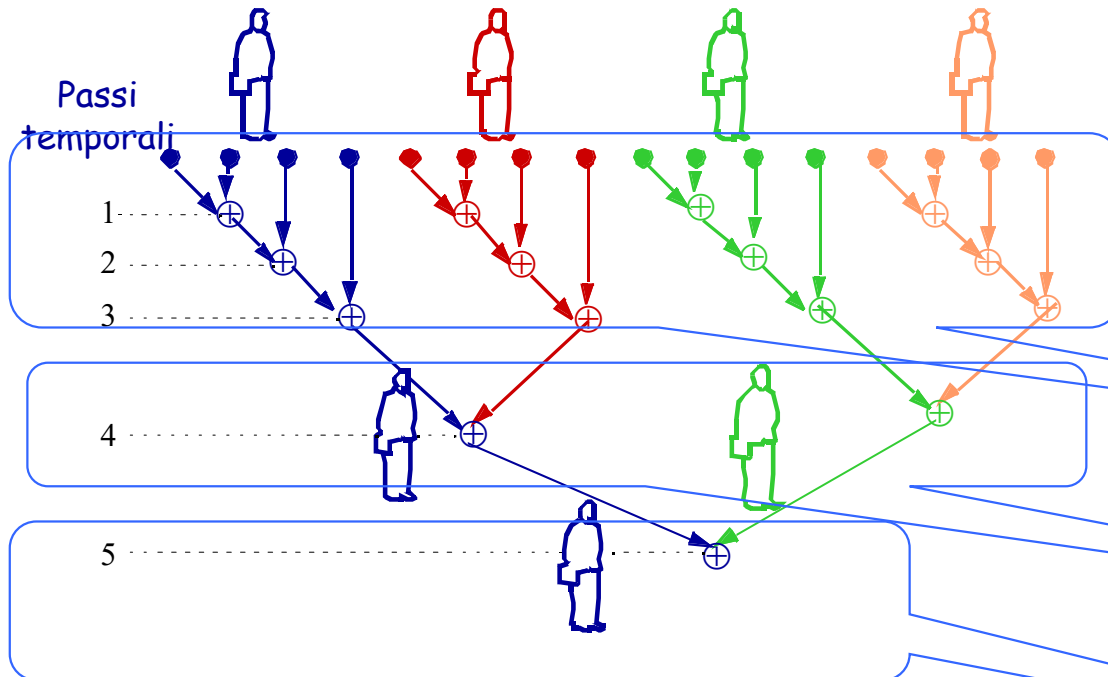
ALGORITMO PARALLELO

II strategia

numero di addizioni = 15

MA

passi temporali = 5



Passi temporali 1-3: 12 addizioni

Passo temporale 4: 2 addizioni

Passo temporale 5: 1 addizione

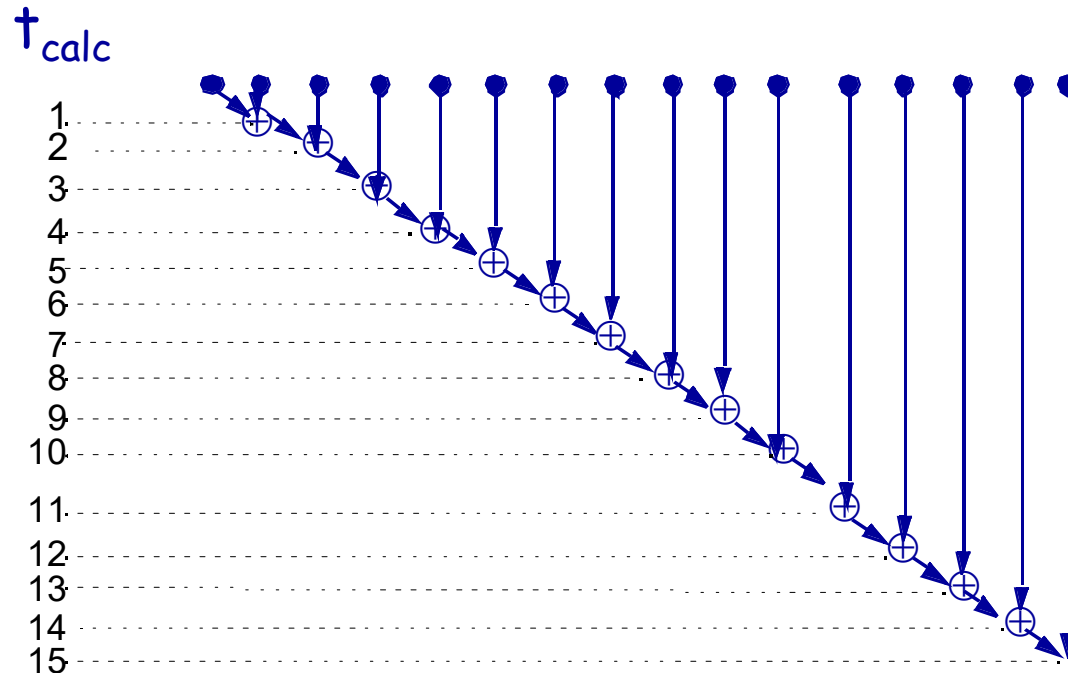
In realtà le cose non vanno proprio così...

Misuriamo di quanto **si riduce** il tempo di
esecuzione su **p** processori
rispetto al tempo di esecuzione
su **1** processore...

Esempio: calcolo della somma di $N=16$ numeri

ALGORITMO SEQUENZIALE

$p=1$



numero di addizioni = 15

$$T_1(16) = 15 t_{calc}$$

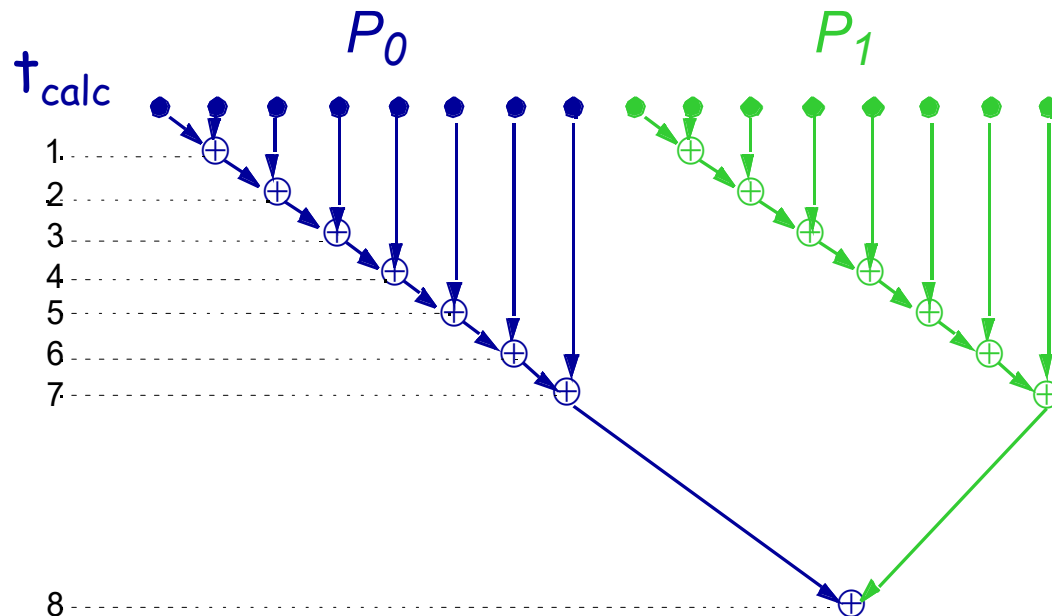
$t_{calc} = \mu$ tempo di esecuzione di una addizione

Esempio: calcolo della somma di $N=16$ numeri

ALGORITMO PARALLELO

$p=2$

numero di addizioni = 15



$$T_2(16) = 8 t_{\text{calc}}$$

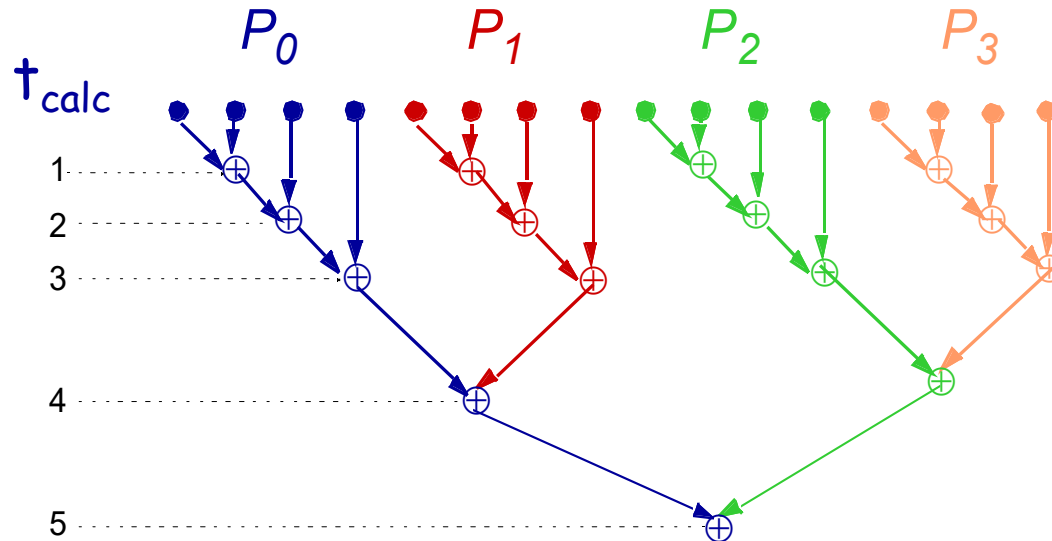
complessità computazionale

2 processori

Esempio: calcolo della somma di $N=16$ numeri

ALGORITMO PARALLELO II strategia $p=4$

numero di addizioni = 15



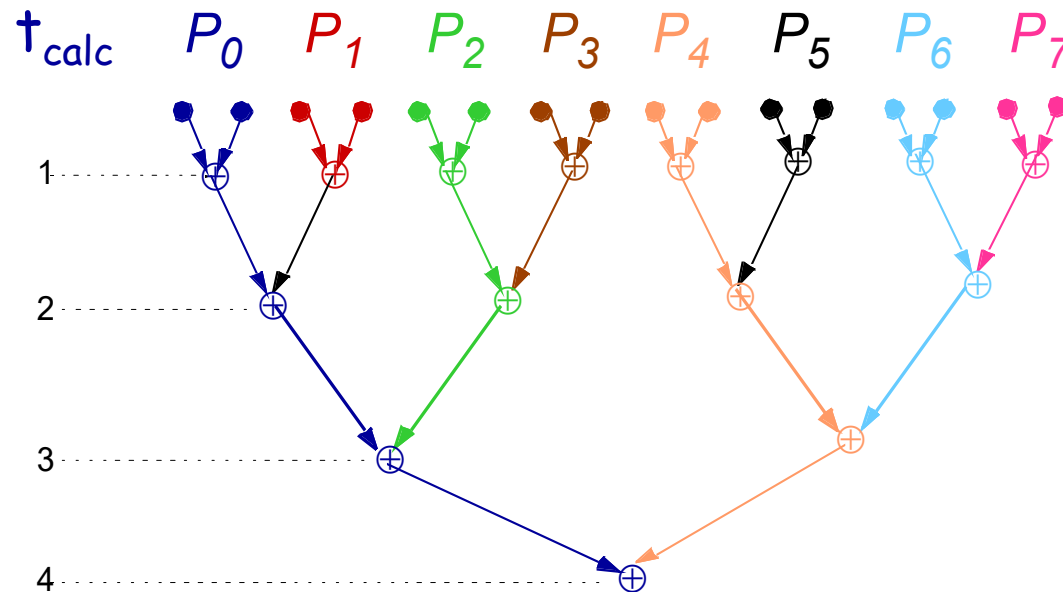
$$T_4(16) = 5 t_{\text{calc}}$$

complessità computazionale
4 processori

Esempio: calcolo della somma di $N=16$ numeri

ALGORITMO PARALLELO II strategia $p=8$

numero di addizioni = 15

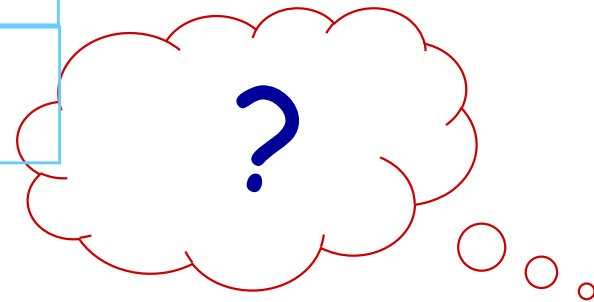


$$T_8(16) = 4 t_{\text{calc}}$$

complessità computazionale
8 processori

In sintesi... II strategia

p	$T_p(16)$
1	15 t_{calc}
2	8 t_{calc}
4	5 t_{calc}
8	4 t_{calc}
p	?



In generale quanto vale T_p ?

In generale: calcolo di $T_p(N)$

ALGORITMO PARALLELO della somma di N numeri
II strategia - posto p processori

$$p=1 \quad T_1=15 \, t_{\text{calc}}$$

$$p=2 \quad T_2=8 \, t_{\text{calc}} = (7+1) \, t_{\text{calc}}$$

$$p=4 \quad T_4=5 \, t_{\text{calc}} = (3+2) \, t_{\text{calc}}$$

$$p=8 \quad T_8=4 \, t_{\text{calc}} = (1+3) \, t_{\text{calc}}$$

.....

$N = 16$

$$T_p(N) = (N/p - 1 + \log_2 p) \, t_{\text{calc}}$$

Domanda...

p	T_p
1	$15 t_{\text{calc}}$
2	$8 t_{\text{calc}}$
4	$5 t_{\text{calc}}$
8	$4 t_{\text{calc}}$

Qual è l'algoritmo che impiega meno tempo?

?

Quanto è più veloce di quello sequenziale?

Esempio: calcolo della somma di $N=16$ numeri

p	T_p	T_1/T_p
1	$15 t_{\text{calc}}$	1.00
2	$8 t_{\text{calc}}$	1.88
4	$5 t_{\text{calc}}$	3.00
8	$4 t_{\text{calc}}$	3.75

Maggiore
riduzione del
tempo ovvero
maggiore
aumento della
velocità

L'algoritmo su 8 processori è il più veloce
E' più veloce di 3.75 volte di quello su 1 processore

Speed-up

Si definisce il rapporto T_1 su T_p

$$S_p = \frac{T_1}{T_p}$$

Lo speed up misura la riduzione del tempo di esecuzione rispetto all'algoritmo su 1 processore

$$S_p < p$$


$$\left(\begin{array}{l} \text{SPEEDUP IDEALE} \\ S_p^{ideale} = p \end{array} \right)$$


Osservazione

$$S_p^{ideale} = \frac{T_1}{T_p} = p$$


$$O_h = (pT_p - T_1) t_{calc}$$

OVERHEAD totale


$$T_p = (O_h + T_1) t_{calc} / p$$

$$S_p = \frac{T_1}{T_p} = \frac{T_1}{(O_h + T_1)/p} = \frac{pT_1}{O_h + T_1} = \frac{p}{\frac{O_h}{T_1} + 1}$$


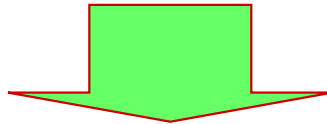
L'OVERHEAD totale misura
quanto lo speed up differisce da quello ideale

L'overhead nell'algoritmo parallelo della somma

II str

$$T_1 = (N-1) t_{\text{calc}}$$

$$T_p = (N/p - 1 + \log_2 p) t_{\text{calc}}$$



$$O_h = p T_p - T_1 = [p (N/p - 1 + \log_2 p) - (N-1)] =$$
$$= (\cancel{N} - p + p \log_2 p - \cancel{N} + 1) = O_h(p \log_2 p)$$

p	O_h
2	$2 t_{\text{calc}}$
4	$8 t_{\text{calc}}$
8	$24 t_{\text{calc}}$
2^k	$p \log_2 p t_{\text{calc}}$

Al crescere di p
l'overhead aumenta!

Quindi

Se si vuole calcolare la somma di 16 numeri
nel minor tempo possibile

l'algoritmo su 8 processori è da preferire

Infatti, aumentando il numero di processori
si riduce

il tempo impiegato per eseguire le operazioni
richieste

QUINDI....

Esempio: calcolo della somma di $N=16$ numeri

p	Speed-up ottenuto	Speed-up ideale
2	1.88	2
4	3.00	4
8	3.75	8

Lo speed-up su 8 processori è il maggiore

MA

Lo speed-up su 2 processori è "il più vicino"
allo speed-up ideale...

Cioè

Ho utilizzato 8 processori per
avere un incremento
di quasi 4 volte

In altre parole

... lo **speed up** non basta a
fornire informazioni sull'efficienza
dell'algoritmo parallelo!

... **e allora?**

Esempio: calcolo della somma di $N=16$ numeri

... se si rapporta lo speed-up al numero di processori...

p	S_p	S_p/p
2	1.88	0.94
4	3.00	0.75
8	3.75	0.47

Rapporto più grande



maggiore sfruttamento dei
processori per $p=2$

In altre parole

l'utilizzo di un maggior numero di processori/core
NON è sempre una garanzia di sviluppo di
algoritmi paralleli "efficaci"

OVVERO

di algoritmi che sfruttano
tutte le risorse della macchina parallela !

Come misurare se e quanto
è stata sfruttato il calcolatore parallelo ?

Efficienza

Si definisce il rapporto E_p su p

$$E_p = \frac{S_p}{p}$$

misura quanto l'algoritmo sfrutta il parallelismo del calcolatore

EFFICIENZA IDEALE

$$E_p^{ideale} = \frac{S_p^{ideale}}{p} = 1$$