



# Calcolo Parallelo e Distribuito

---

Il parallelismo dell'ambiente multicore  
MIMD-SM

**Docente:** Prof. L. Marcellino

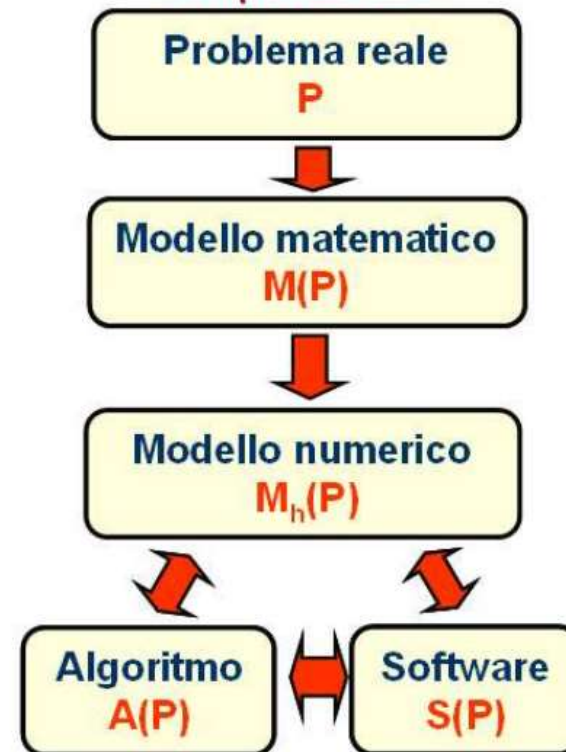
**Tutor:** Prof. P. De Luca

# Modellizzazione di problemi su larga scala



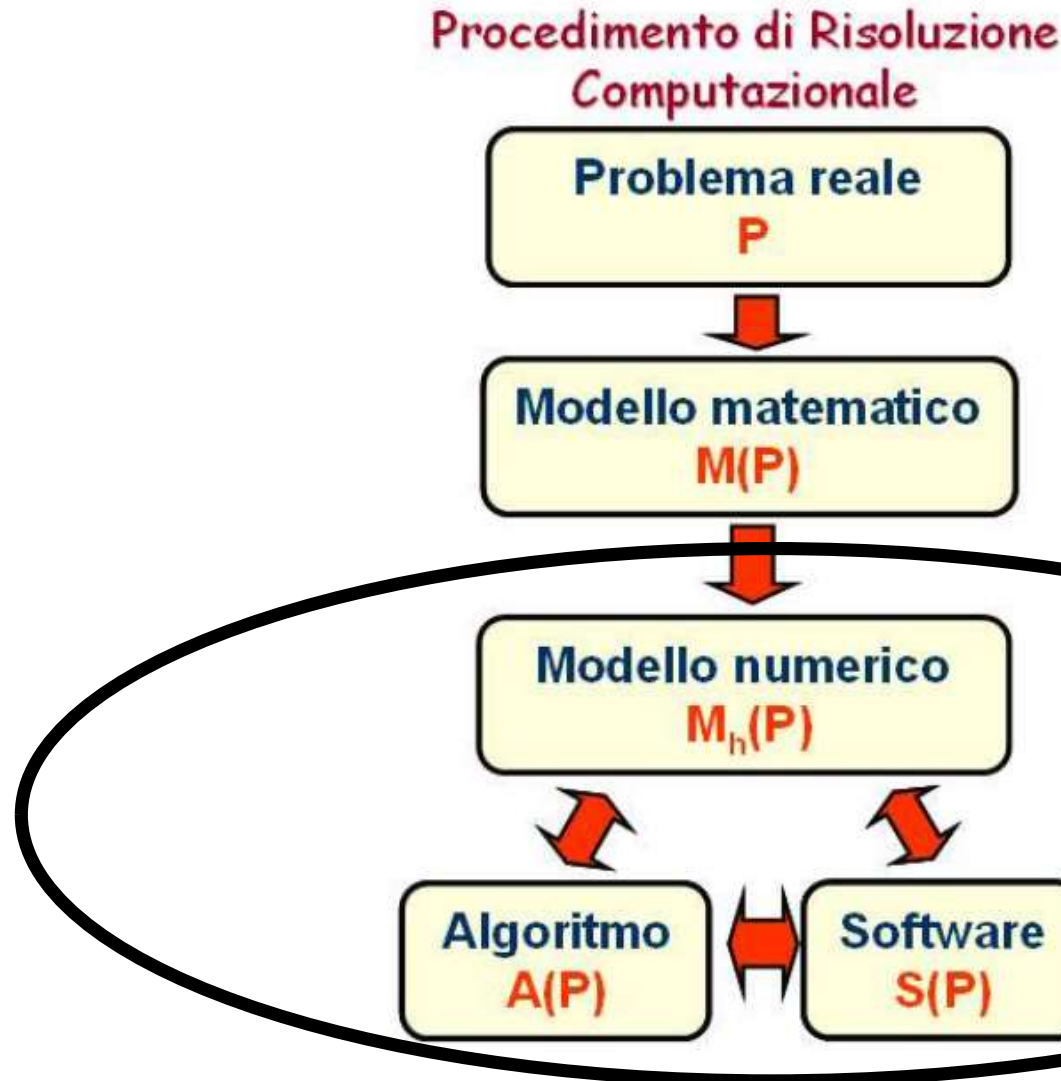
- Ricerca su internet
- Trasporto
- Pubblicità e Marketing
- Servizi bancari e finanziari
- Media e intrattenimento
- Meteorologia
- Assistenza sanitaria
- Sicurezza informatica
- Formazione

## Procedimento di Risoluzione Computazionale



# Il parallelismo delle architetture MIMD

## metodi numerici paralleli



Quando si progetta un algoritmo sequenziale oltre a concentrarsi sul problema è importante tenere conto dell'**esecutore**.  
Lo **sviluppo del software** deve essere ricondotto ad un insieme di passi elementari, per lo specifico ambiente di calcolo.

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzare gli  $N$  passi in modo da distribuirli, eventualmente, a più unità processanti.

Più possibilità:

- ogni unità processante esegue un passo **differente**  
(**decomposizione funzionale**)

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzarlo per individuare **task indipendenti che possano essere processati separatamente e contemporaneamente**.

### Esempio:

Torta con crema pasticciera

- $h=1$  preparare la crema
- $h=2$  preparare il pan di spagna
- $h=3$  preparare la torta con la crema



**... e se siamo in due?**

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzarlo per individuare **task indipendenti che possano essere processati separatamente e contemporaneamente.**

### Esempio: in due

Torta con crema pasticciera



$h=1$	preparare la crema;	$h=2$	preparare il pan di spagna
-------	---------------------	-------	----------------------------

$h=3$  preparare la torta con la crema

**... ma la crema si deve raffreddare!!!**

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzarlo per individuare **task indipendenti che possano essere processati separatamente e contemporaneamente**.

### Esempio: in due

Torta con crema pasticciera



$h=1$  preparare la crema;

$h=2$  preparare il pan di spagna

$h=3$  preparare la torta con la crema

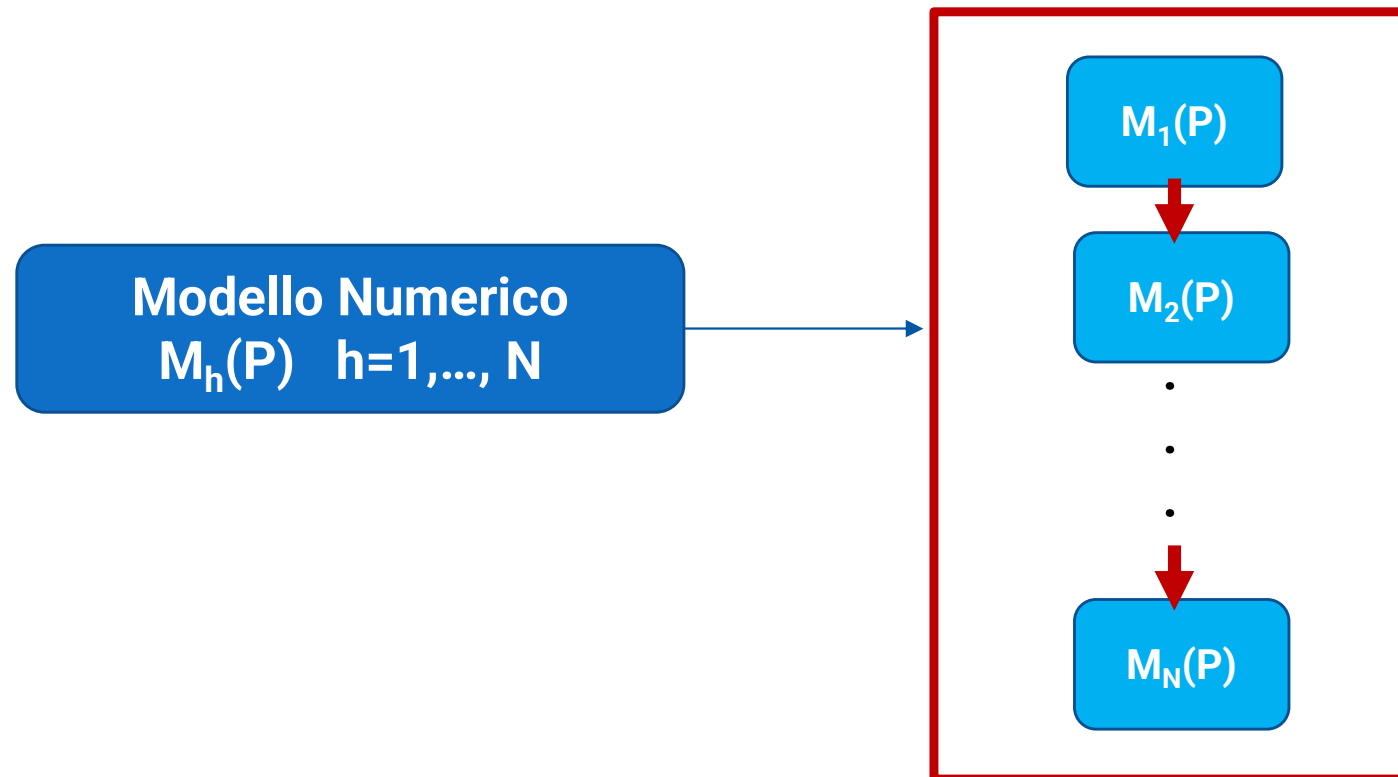
L'esecutore più lento  
potrebbe fare il pan di spagna

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,\dots,N$ ) e analizzarlo per individuare **task indipendenti** che possano essere **processati separatamente e contemporaneamente**.



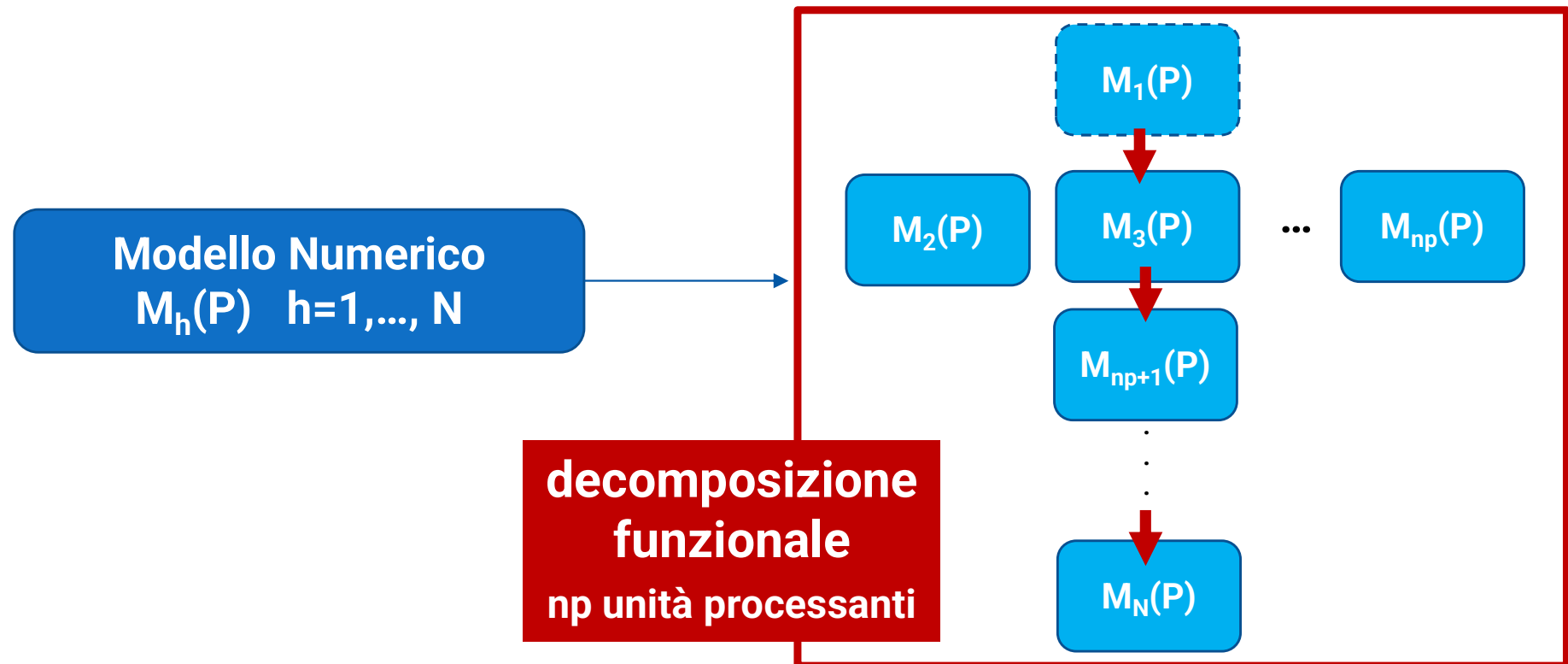


# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,\dots,N$ ) e analizzarlo per individuare **task indipendenti** che possano essere **processati separatamente e contemporaneamente**.



# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli – decomposizione funzionale

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzarlo per individuare **task indipendenti che possano essere processati separatamente e contemporaneamente**.

Decomposizione funzionale = **eseguire elaborazioni differenti e indipendenti contemporaneamente**

Il modello è decomposto in base al lavoro che deve essere svolto.

**Caratteristica fondamentale:**

applicabile a un modello caratterizzato da più nuclei computazionali

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzare gli  $N$  passi in modo da distribuirli, eventualmente, a più unità processanti.

Più possibilità:

- **ogni unità** processante esegue un passo **differente**  
(**decomposizione funzionale**)
- **tutte le unità** processanti eseguono **la stessa** operazione su un sottoinsieme di dati  
(**decomposizione del dominio**)

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,\dots,N$ ) e **suddividere ogni task in più sotto-task uguali e processarli contemporaneamente**, ma **riducendo al minimo la collezione dei risultati locali**.

### Esempio:

Torta con crema pasticciera

- $h=1$  preparare la crema
- $h=2$  preparare il pan di spagna
- $h=3$  preparare la torta con la crema



**... e se siamo in due?**

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e suddividere ogni task in più sotto-task uguali e processarli contemporaneamente, ma riducendo al minimo la collezione dei risultati locali.

### Esempio: in due

Torta con crema pasticciera

$h=1.1+1.2$

in due prepariamo la crema;

$h=2.1+2.2$

in due prepariamo il pan di spagna

$h=3.1+3.2$

in due prepariamo la torta con la crema



... ma la crema si deve raffreddare!!!

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e suddividere ogni task in più sotto-task uguali e processarli contemporaneamente, ma riducendo al minimo la collezione dei risultati locali.

### Esempio: in due

Torta con crema pasticciera



$h=1.1+1.2$

in due prepariamo la crema;

$h=2.1+2.2$

in due prepariamo il pan di Spagna

$h=3.1+3.2$

in due prepariamo la torta con la crema

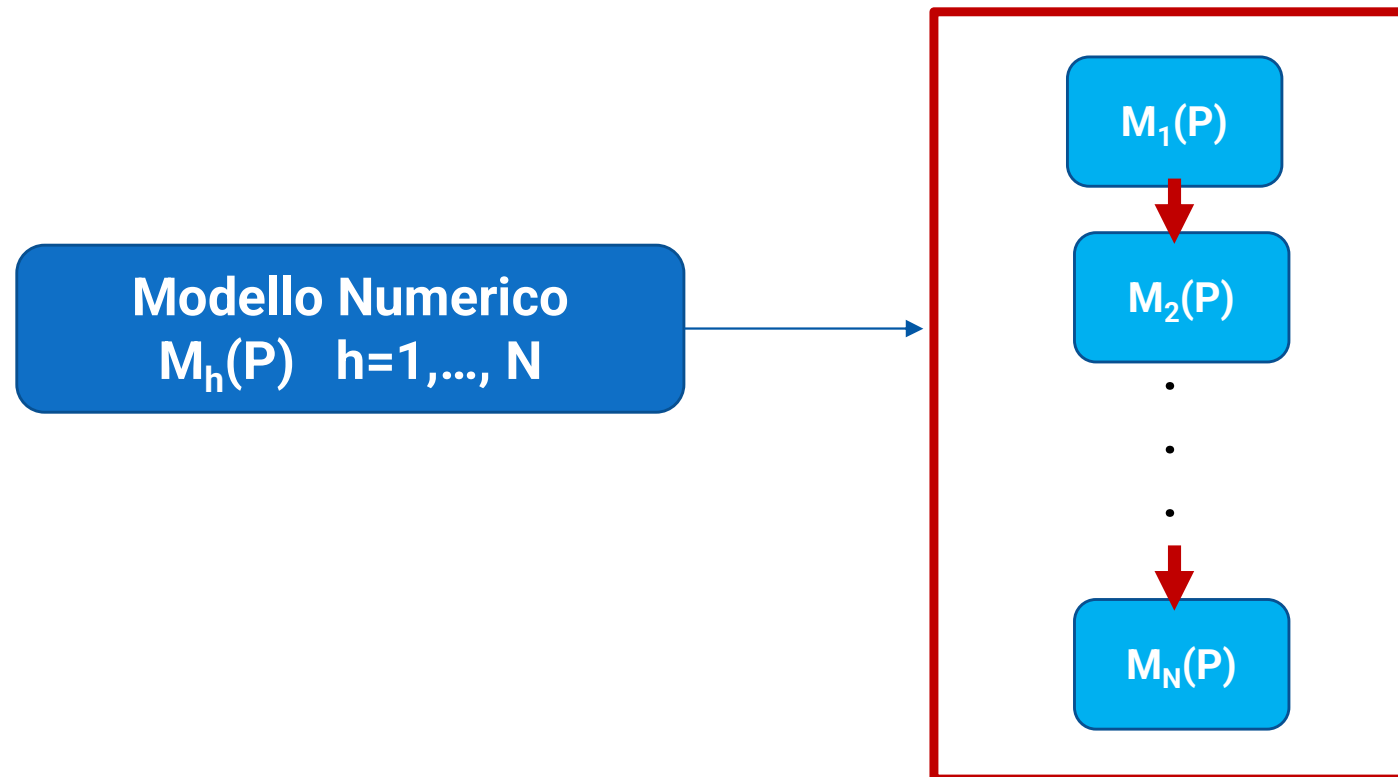
**...in due aspetteremo qualche minuto prima di unire pan di Spagna e crema**

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e **suddividere ogni task in più sotto-task uguali e processarli contemporaneamente**, ma **riducendo al minimo la collezione dei risultati locali**.

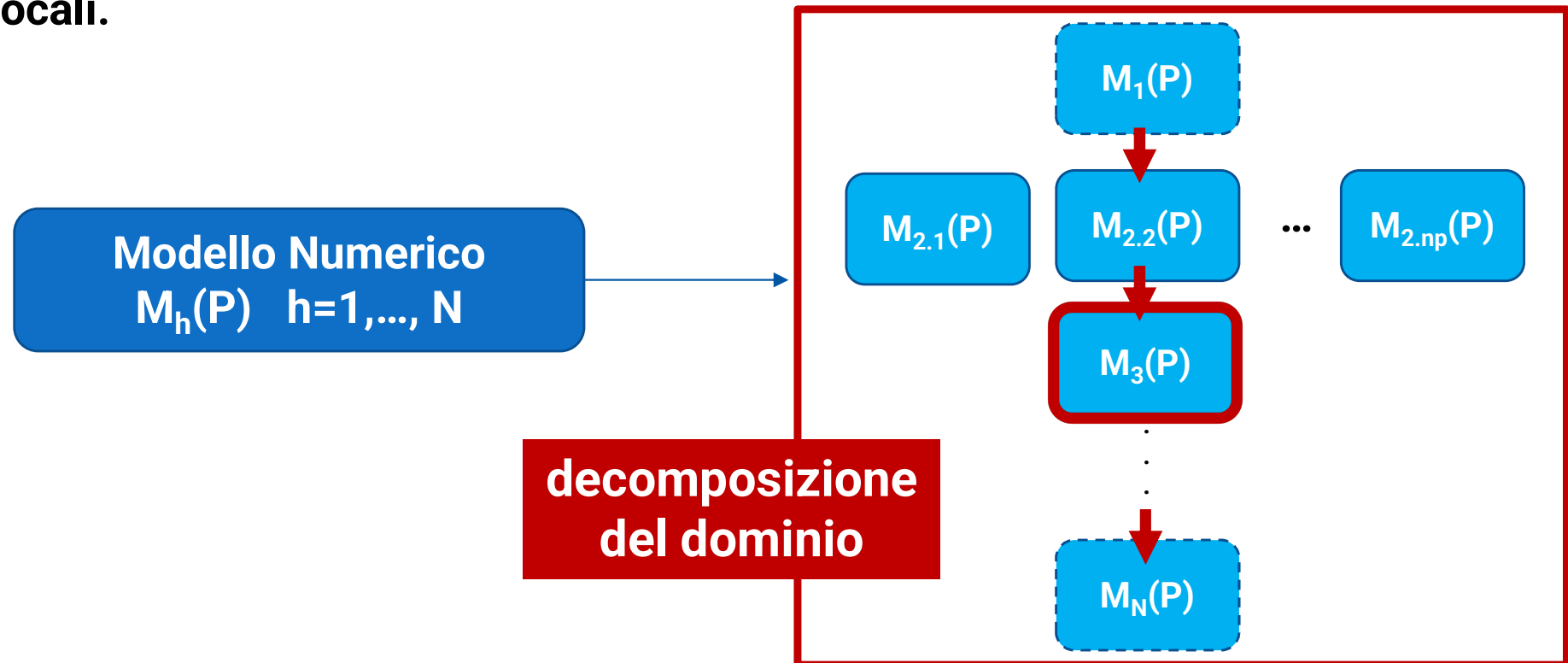


# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e **suddividere ogni task in più sotto-task uguali e processarli contemporaneamente**, ma **riducendo al minimo la collezione dei risultati locali**.





# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli – **decomposizione dominio**

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzarlo per **suddividere ogni task in più sotto-task uguali e processarli contemporaneamente, ma riducendo al minimo la collezione dei risultati locali.**

Decomposizione dominio = **suddividere i dati ed elaborarli tutti allo stesso modo.**

Il modello è decomposto in base al lavoro che deve essere svolto.

**Caratteristica fondamentale:**

vantaggiosa solo quando vi è una **semplice/nessuna** collezione dei risultati locali

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzare gli  $N$  passi in modo da distribuirli, eventualmente, a più unità processanti.

Più possibilità:

- **ogni unità** processante esegue un passo **differente**  
(**decomposizione funzionale**)
- **tutte le unità** processanti eseguono **la stessa** operazione su un sottoinsieme di dati  
(**decomposizione del dominio**)
- **combinazione delle due possibilità precedenti**

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Modello Numerico  
 $M_h(P)$

Ripartire dal modello numerico (modello matematico discretizzato  $h=1,...,N$ ) e analizzare gli  $N$  passi in modo da distribuirli, eventualmente, a più unità processanti.

Più possibilità:

- **ogni unità** processante esegue un passo **differente**  
(**decomposizione funzionale**)
- **tutte le unità** processanti eseguono **la stessa** operazione su un sottoinsieme di dati  
(**decomposizione del dominio**)
- **combinazione delle due possibilità precedenti**

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Proviamo a fare un esempio meno banale...

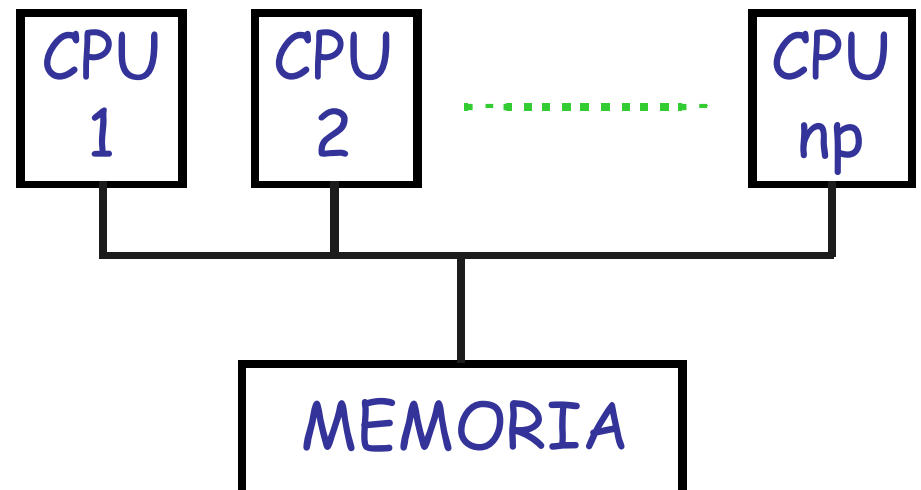
... supponiamo di dover sommare due vettori di grandi dimensioni!

**Input:**  $a = (a_0, a_1, a_2, \dots, a_{N-1})$ ,  $b = (b_0, b_1, b_2, \dots, b_{N-1})$

**Output:**  $c = (a_0 + b_0, a_1 + b_1, a_2 + b_2, \dots, a_{N-1} + b_{N-1})$

su un calcolatore parallelo  
tipo MIMD

A MEMORIA **CONDIVISA**



# Il parallelismo delle architetture MIMD-SM

## somma di due vettori di dimensione $N$

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

$$c_0 := a_0 + b_0$$

$$c_1 := a_1 + b_1$$

...

...

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



## Somma di due vettori di dimensione $N$

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

```
begin
  for i=0 to N-1 do
     $c_i := a_i + b_i;$ 
  endfor
end
```

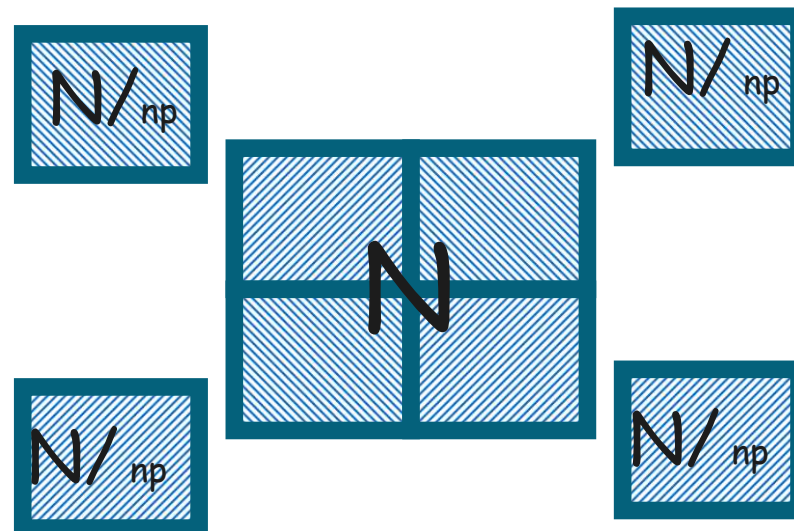
Qual è  
l'ALGORITMO PARALLELO?

# Il parallelismo delle architetture MIMD

## somma di due vettori di dimensione $N$

Se ho a disposizione  $np$  unità processanti,  
come posso procedere sfruttando il concetto di  
calcolo parallelo?

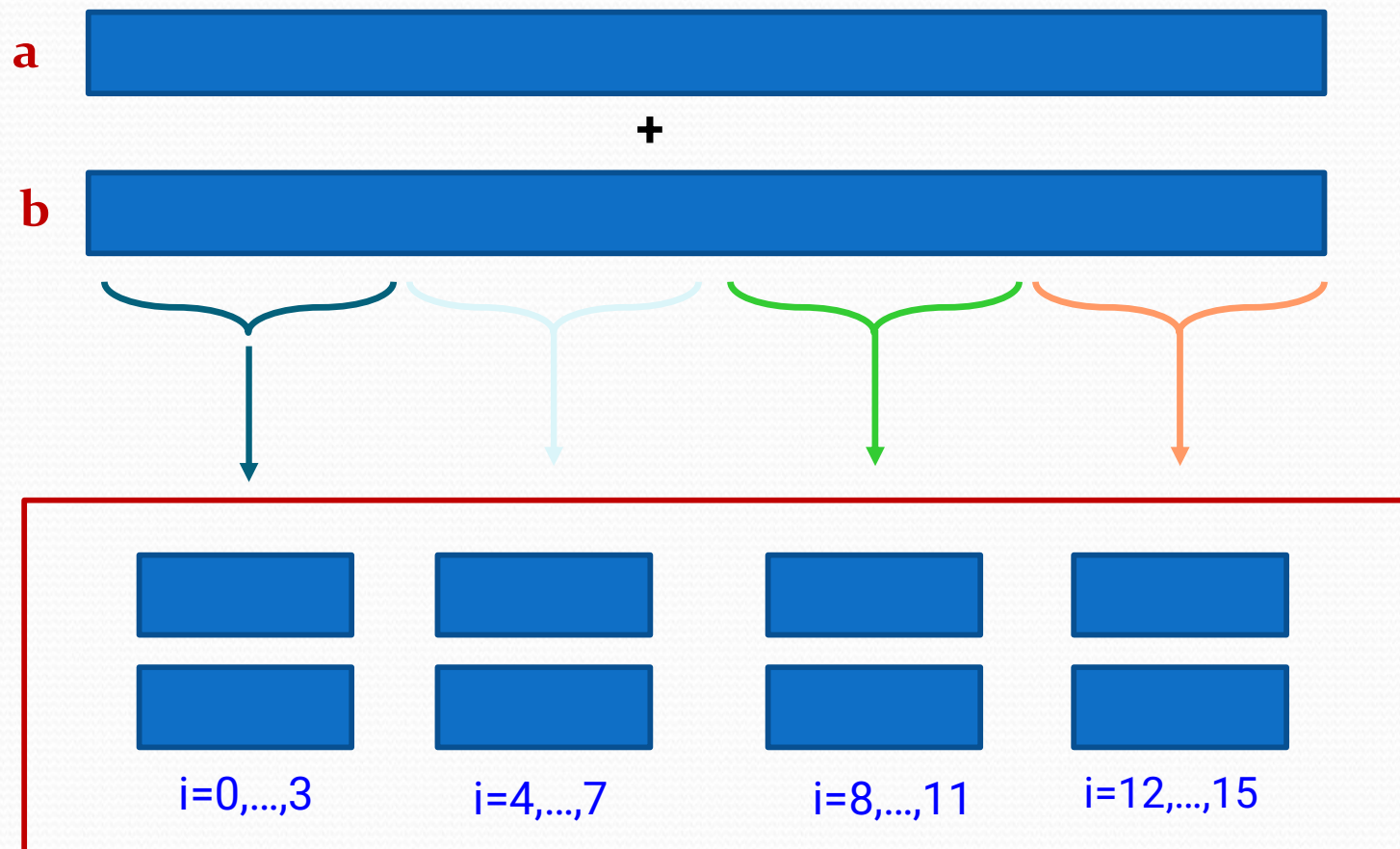
Decomporre un problema di dimensione  $N$  in  $np$  sottoproblemi di  
dimensione  $N/np$  e risolverli **contemporaneamente** usando  $np$  CPU



# Strategia di parallelizzazione

## somma di due vettori di lunghezza N

Esempio:  $N=16$ ,  $np=4$





# Strategia di parallelizzazione

## somma di due vettori di lunghezza N

### IDEA

Suddividere il dominio del problema (i due vettori) e assegnare la somma di sottovettori ad ogni CPU

**I sottovettori sommati possono essere uniti nella memoria shared**

**a formare il vettore risultato**

**PARALLELISMO COMPLETO**

## Strategia di parallelizzazione somma di due vettori di lunghezza N

- Il lavoro sui due vettori è distribuito ai core.
- Ogni core si occupa di sommare due sotto-vettori opportuni dei vettori iniziali.
- I sottovettori sommati possono essere uniti nella memoria shared

**... senza ulteriori operazioni**

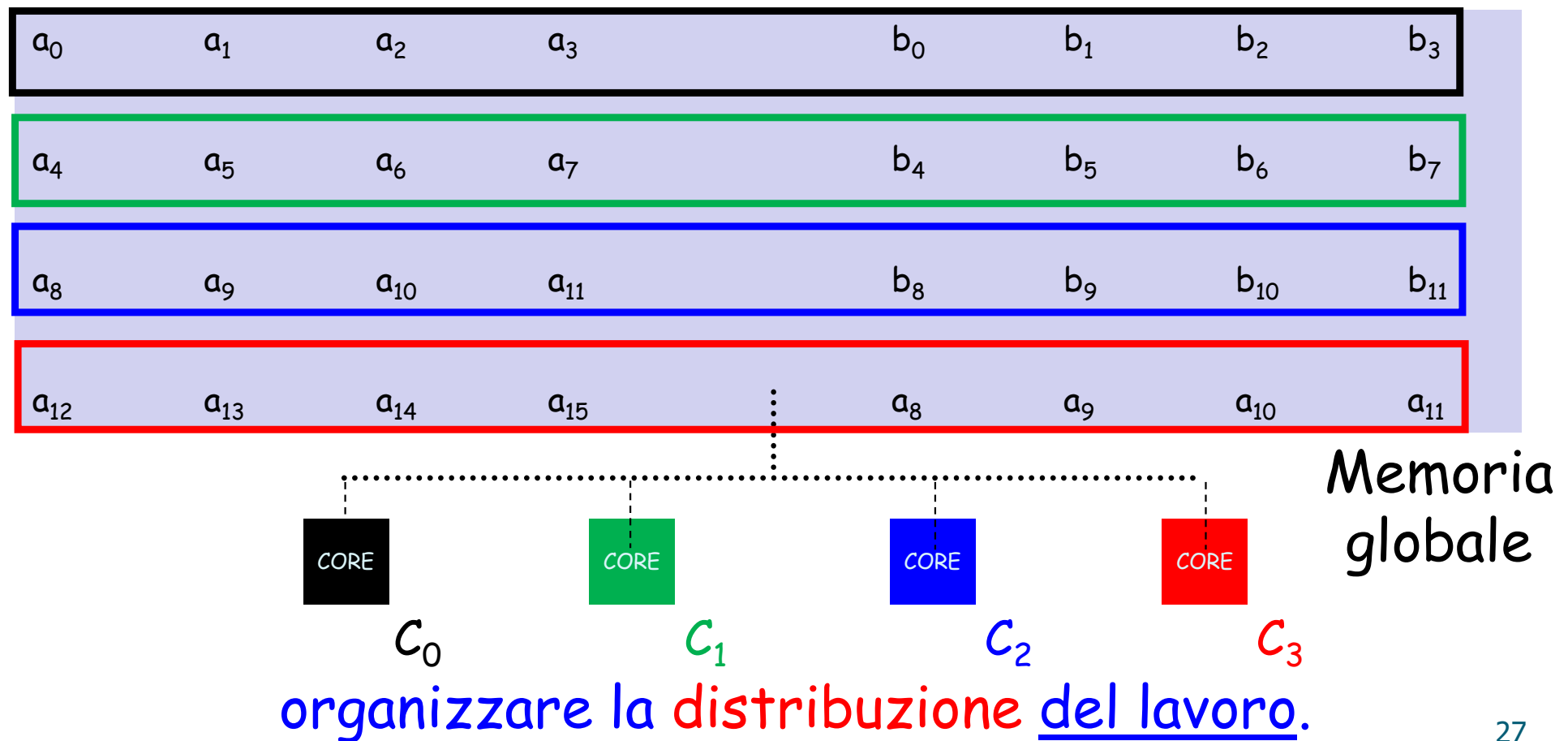
**PROBLEMA COMPLETAMENTE PARALLELIZZABILE  
FULL PARALLEL**

# Somma due vettori di dimensione N

## MIMD Shared Memory

I core possono accedere simultaneamente alla memoria globale su dati differenti

- Esempio:  $N=16$ ,  $p=4$

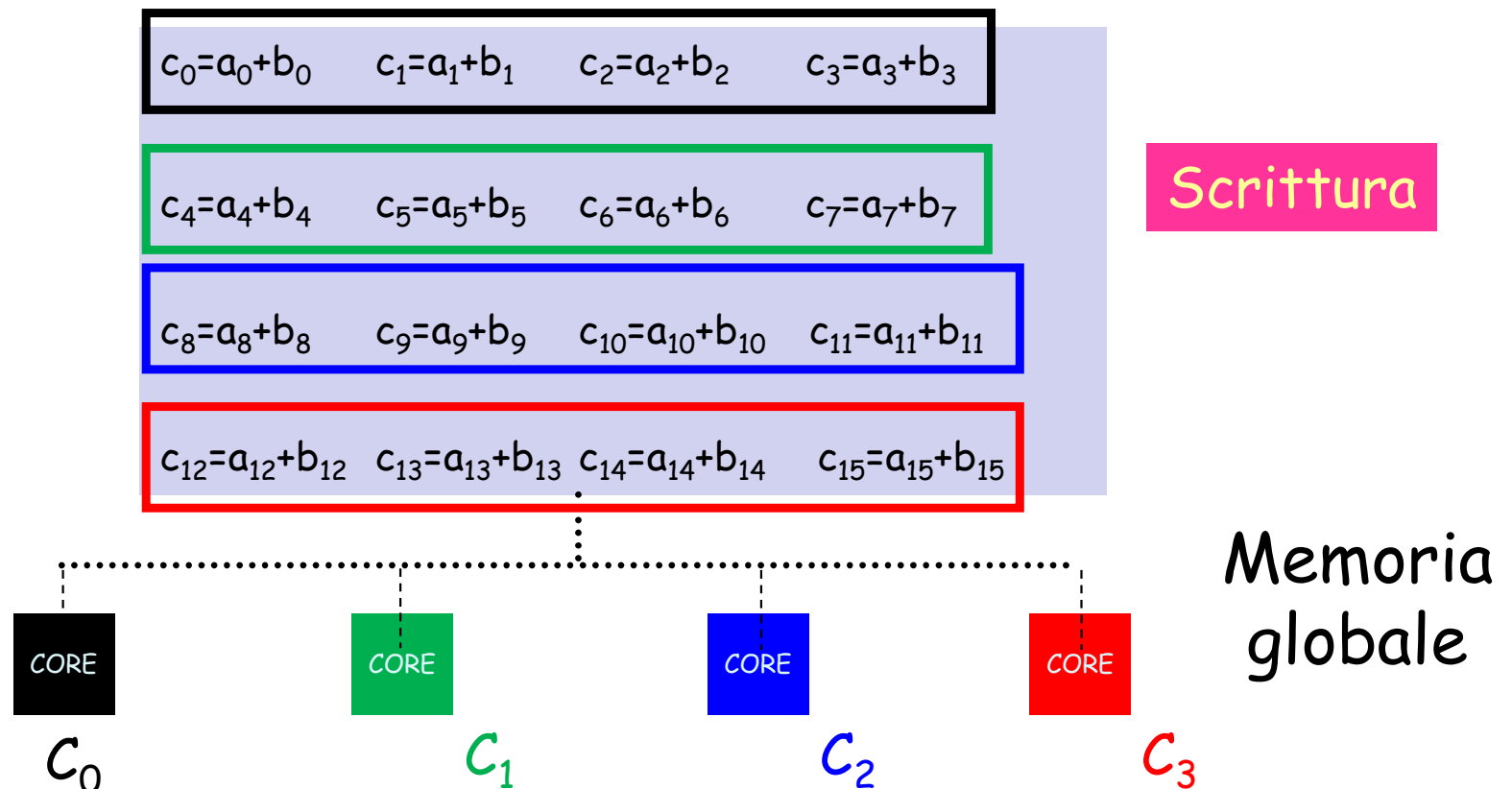


# Somma due vettori di dimensione N

## MIMD Shared Memory

- Esempio:  $N=16$ ,  $p=4$

Calcolo somme  
parziali



# Strategie di parallelizzazione per problemi di tipo element-wise

**PROBLEMA COMPLETAMENTE  
PARALLELIZZABILE**

**FULL PARALLEL**

**nessuna collezione dei risultati**

# Il parallelismo delle architetture MIMD-SM

## metodi numerici paralleli

Quale può essere un problema **ancora meno banale**?

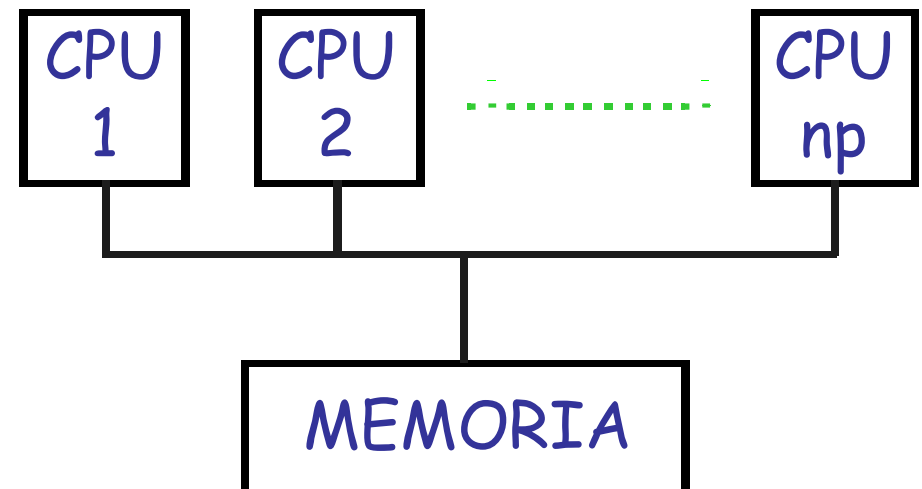
... supponiamo di dover **sommare tra loro** gli elementi  
di un vettore di grandi dimensioni!

**Input:**  $a = (a_0, a_1, a_2, \dots, a_{N-1})$

**Output:**  $\text{sum} = a_0 + a_1 + a_2 + \dots + a_{N-1}$

su un calcolatore parallelo  
tipo MIMD

A MEMORIA **CONDIVISA**



# Il parallelismo delle architetture MIMD-SM

## somma di N numeri

Su un calcolatore monoprocesore la somma è calcolata eseguendo  $N-1$  addizioni una per volta secondo un ordine prestabilito

$\text{sumtot} := a_0$

$\text{sumtot} := \text{sumtot} + a_1$

$\text{sumtot} := \text{sumtot} + a_2$

$\vdots$

$\text{sumtot} := \text{sumtot} + a_{N-1}$



## Somma di N numeri

---

Su un calcolatore monoprocesore la somma è calcolata *eseguendo le  $N-1$  addizioni una per volta* secondo un ordine prestabilito

```
begin
    sumtot := a0;
    for i=1 to N-1 do
        sumtot := sumtot + ai ;
    endfor
end
```

Qual è  
l'ALGORITMO PARALLELO?

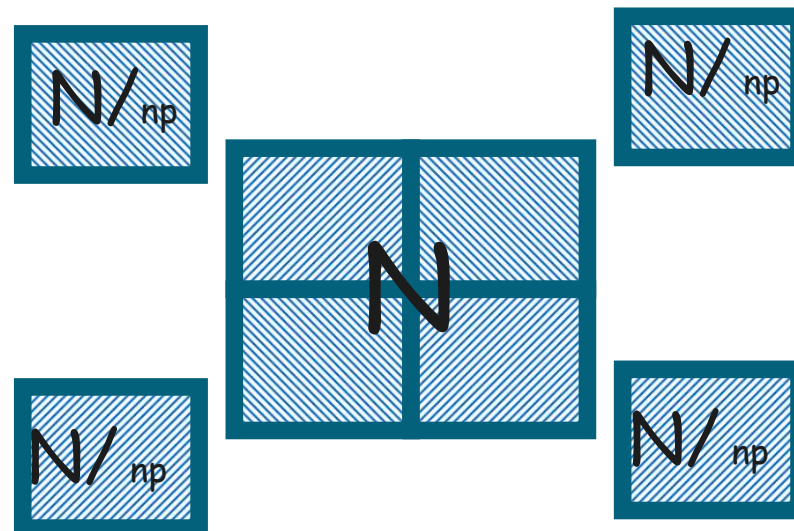


# Il parallelismo delle architetture MIMD

## somma di $N$ numeri

Se ho a disposizione  $np$  unità processanti,  
come posso procedere sfruttando il concetto di  
calcolo parallelo?

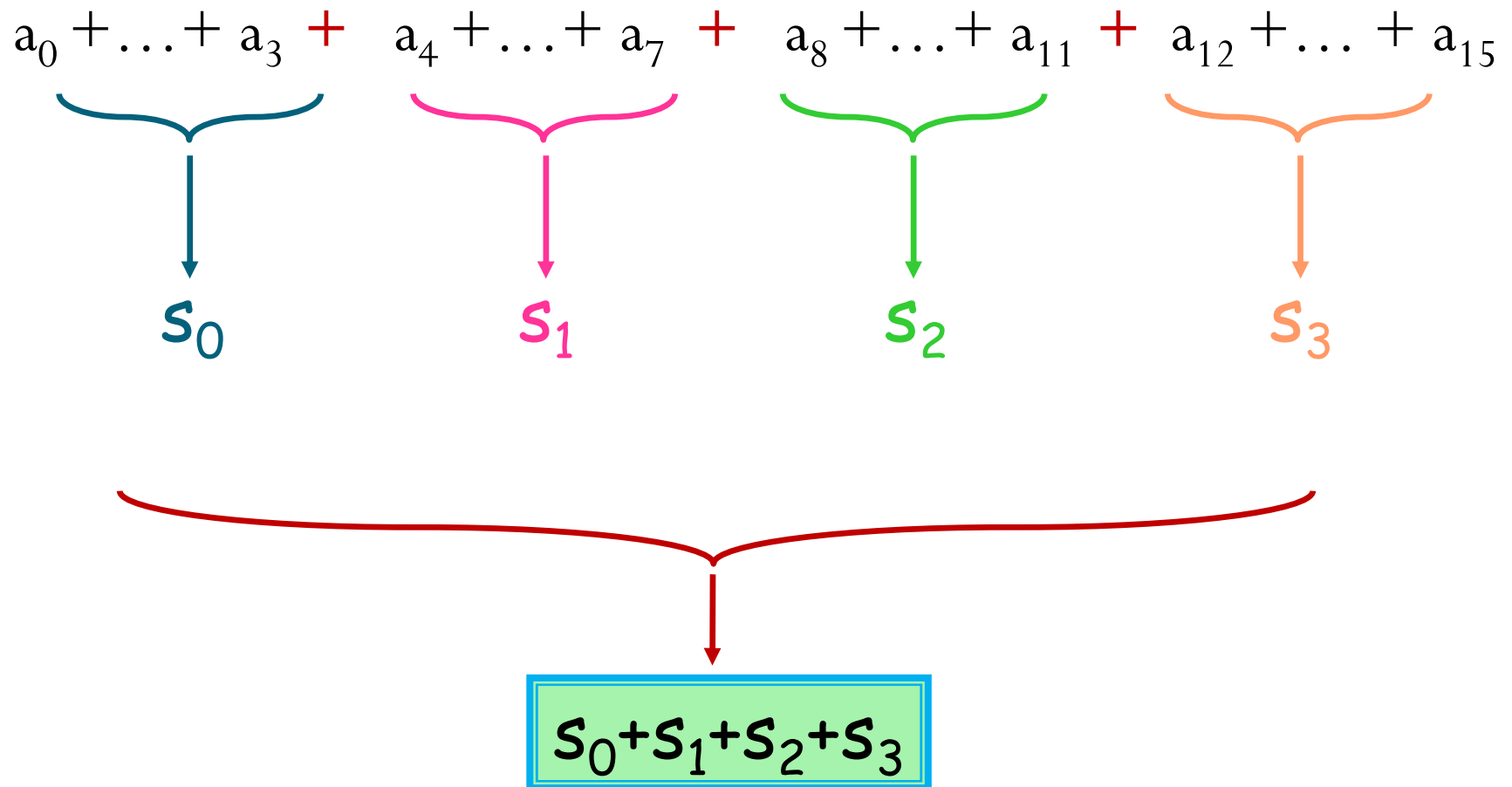
Decomporre un problema di dimensione  $N$  in  $np$  sottoproblemi di  
dimensione  $N/np$  e risolverli **contemporaneamente** usando  $np$  CPU



# Il parallelismo delle architetture MIMD

## somma N numeri

Esempio:  $N=16$ ,  $np=4$

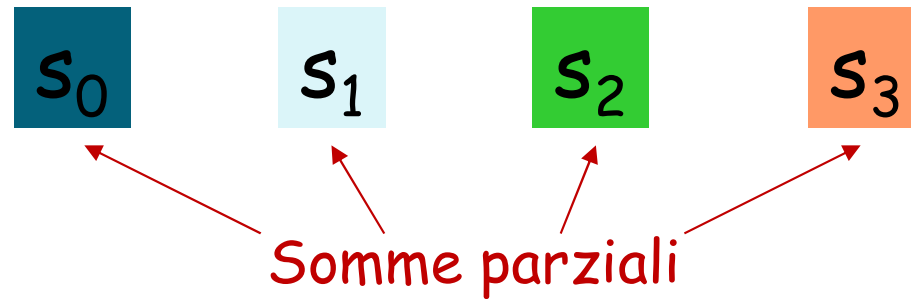


# Il parallelismo delle architetture MIMD

## somma N numeri

### IDEA

Suddividere la somma in somme parziali ed assegnare ciascuna somma parziale ad una CPU

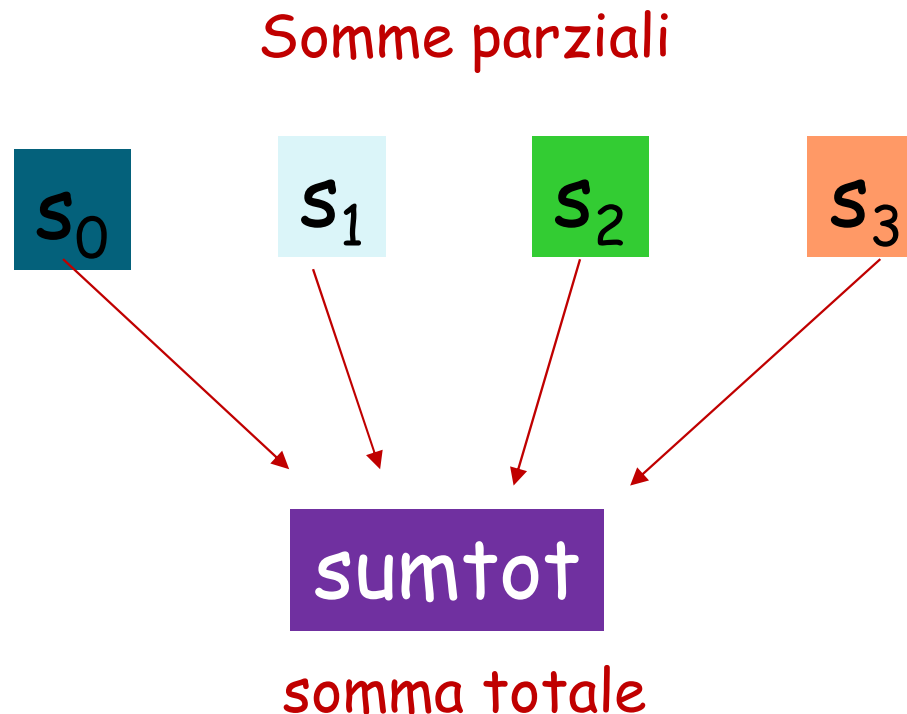


# Il parallelismo delle architetture MIMD

## somma N numeri

### IDEA

Le somme parziali devono poi essere combinate in **modo opportuno** per ottenere la somma totale

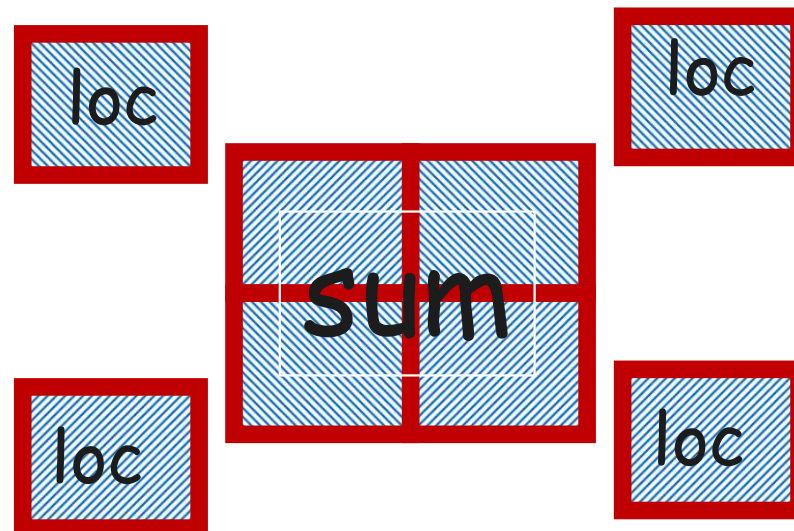


# Il parallelismo delle architetture MIMD

## somma di N numeri

Se ho a disposizione  $np$  unità processanti,  
come posso procedere sfruttando il concetto di  
calcolo parallelo?

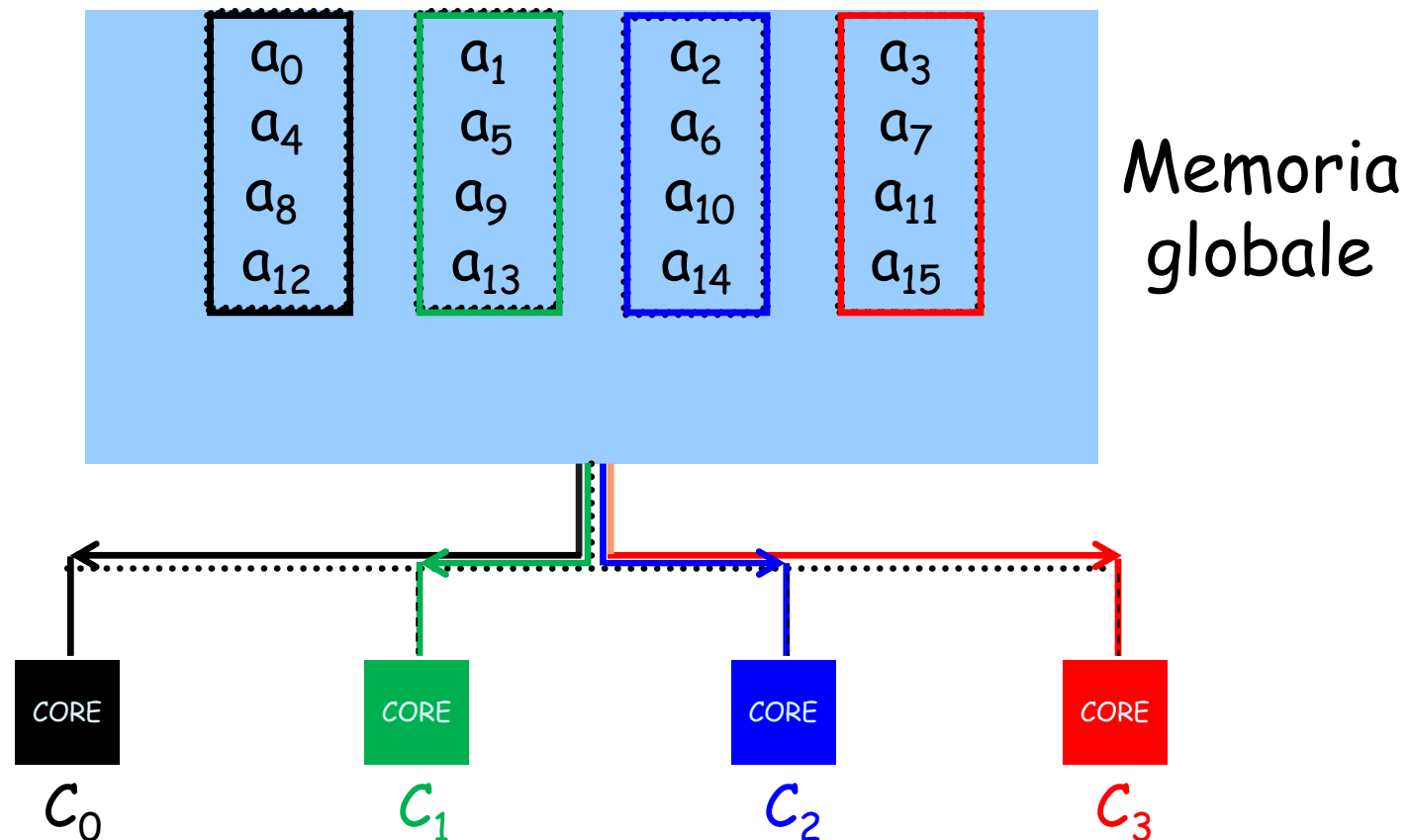
Ho bisogno di collezionare con un'ulteriore operazione i risultati locali



# Somma N numeri - MIMD Shared Memory

I core possono accedere simultaneamente alla memoria globale su dati differenti

- Esempio:  $N=16$ ,  $p=4$

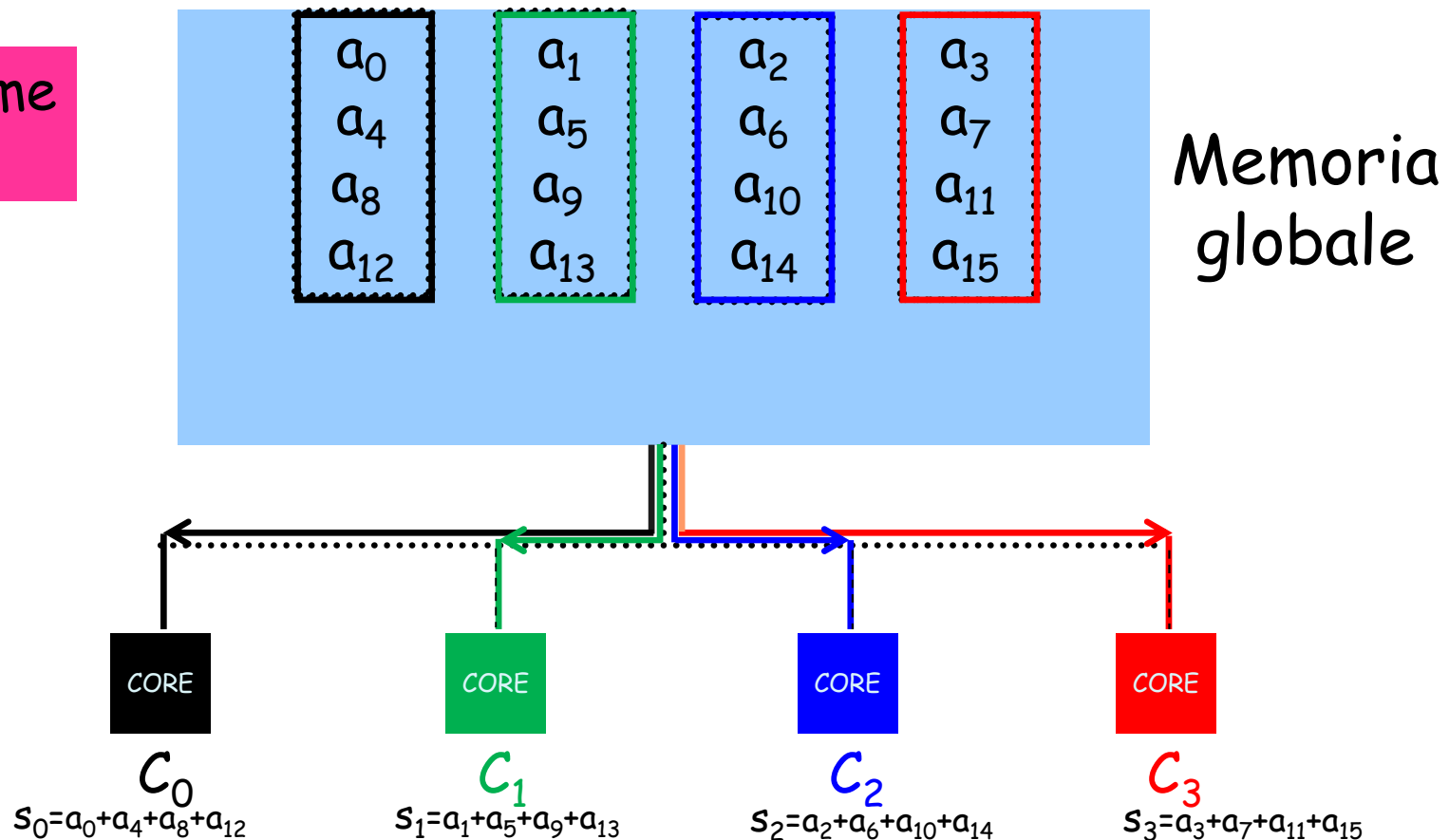


organizzare la distribuzione del lavoro.

# Somma - MIMD Shared Memory

- Esempio:  $N=16$ ,  $p=4$

Calcolo somme  
parziali

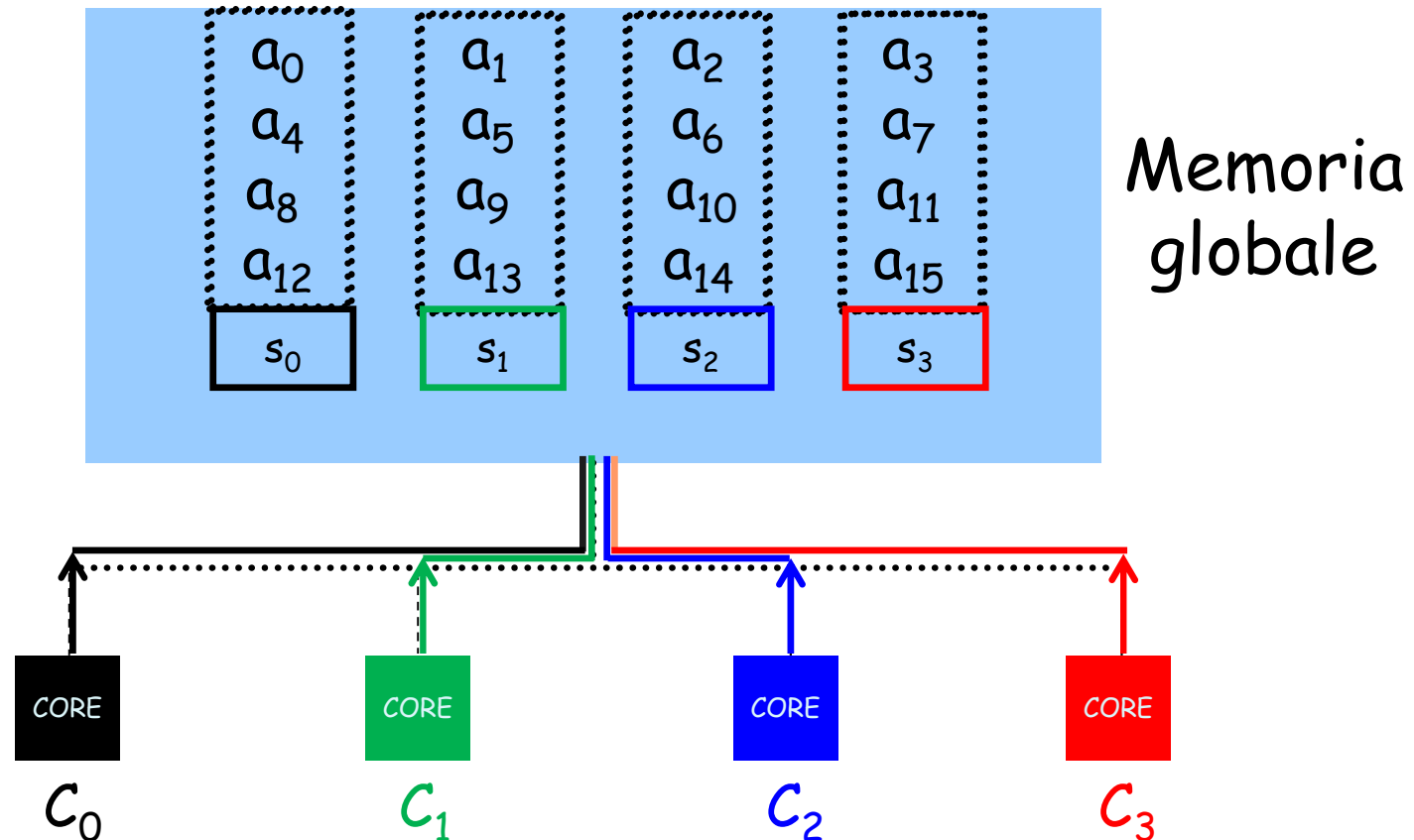


# Somma - MIMD Shared Memory

- Esempio:  $N=16$ ,  $p=4$

I core possono accedere  
simultaneamente alla memoria globale  
su dati differenti

Scrittura







# Somma - MIMD Shared Memory

---

Come calcolare  
la somma totale?



## Somma - MIMD Shared Memory

---

Ogni core  $C_i$ ,  
calcolata la sua somma parziale  $s_i$ ,  
deve addizionare tale valore ad una variabile  
(ad es. ***sumtot***) che conterrà la somma finale.  
I core devono accedere a ***sumtot***  
uno alla volta.



# Esempio: Somma

---

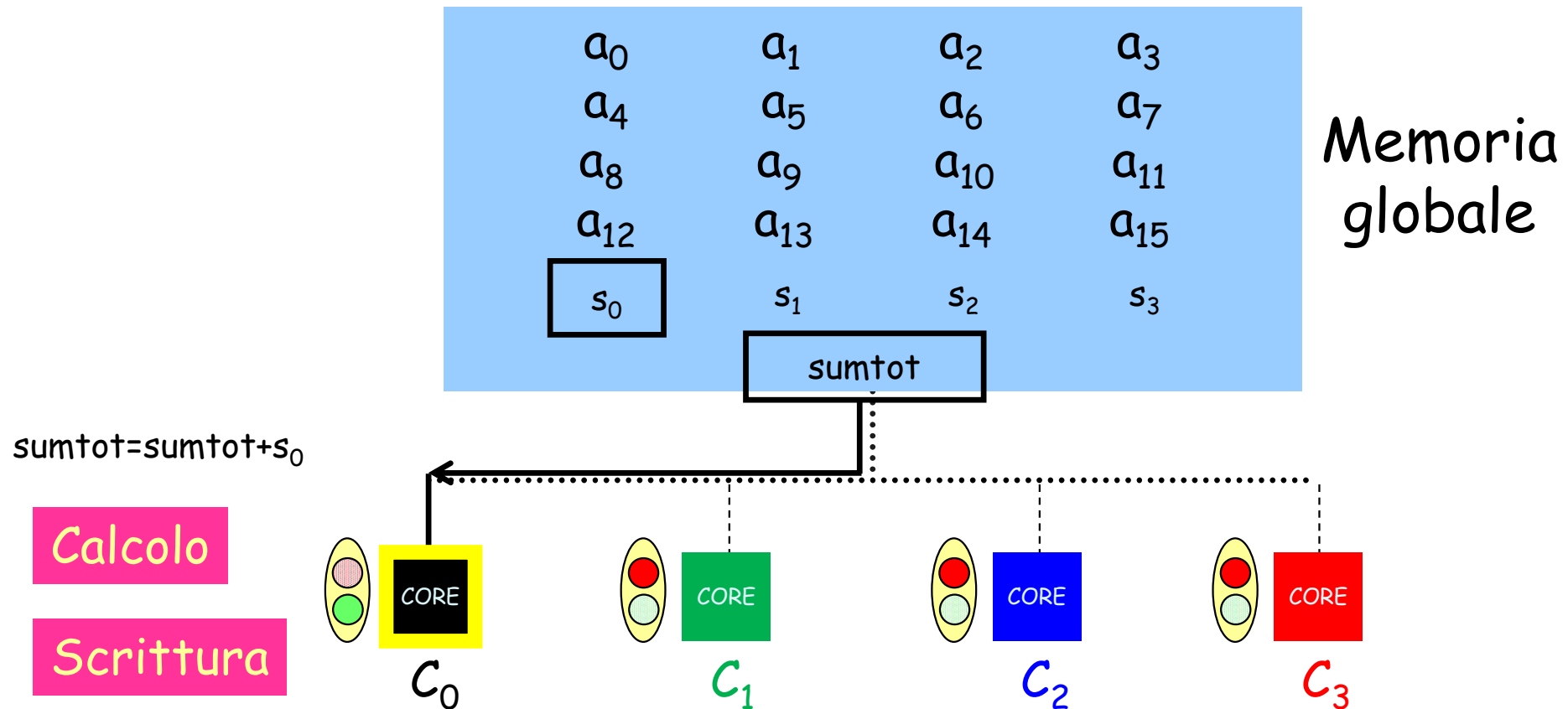
Affinchè il valore di *sumtot* sia **correttamente** aggiornato  
è necessario che ciascun core abbia **accesso esclusivo**  
a tale variabile durante il suo aggiornamento



**Sincronizzazione**  
degli accessi in memoria

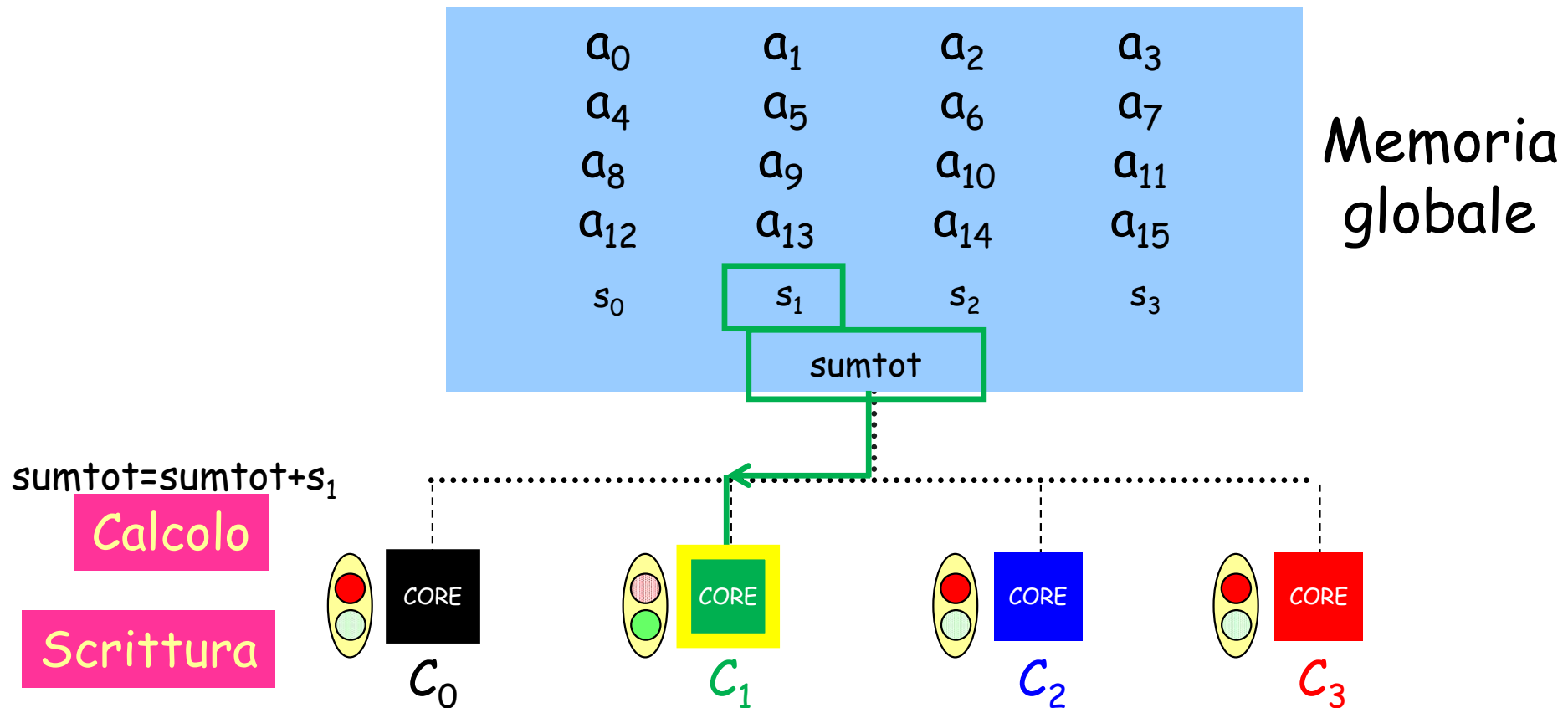
## Esempio: Somma 1strategia

- Esempio:  $N=16$ ,  $p=4$



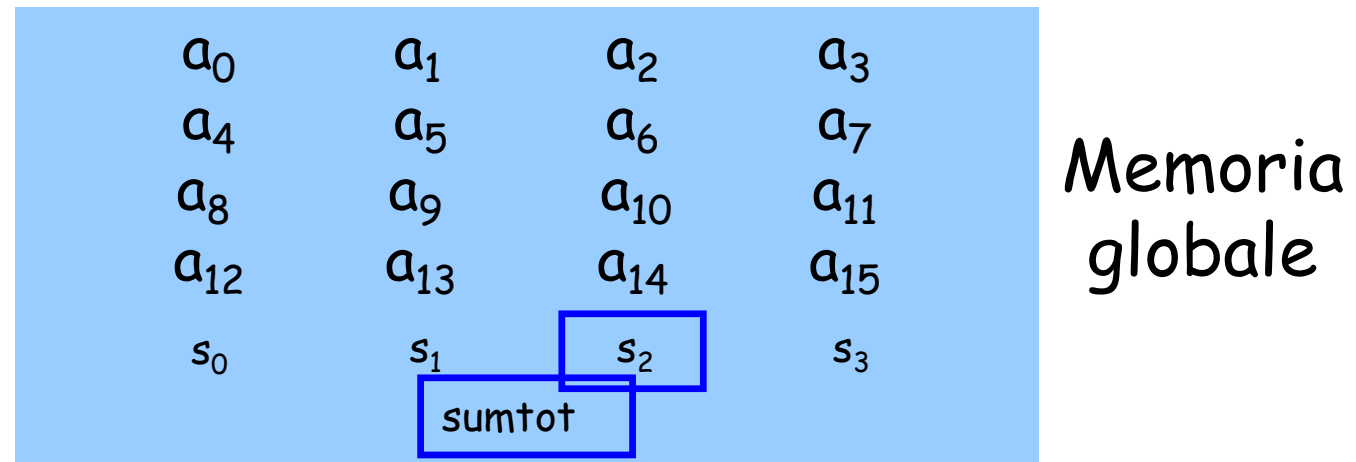
## Esempio: Somma 1strategia

- Esempio:  $N=16$ ,  $p=4$



## Esempio: Somma 1strategia

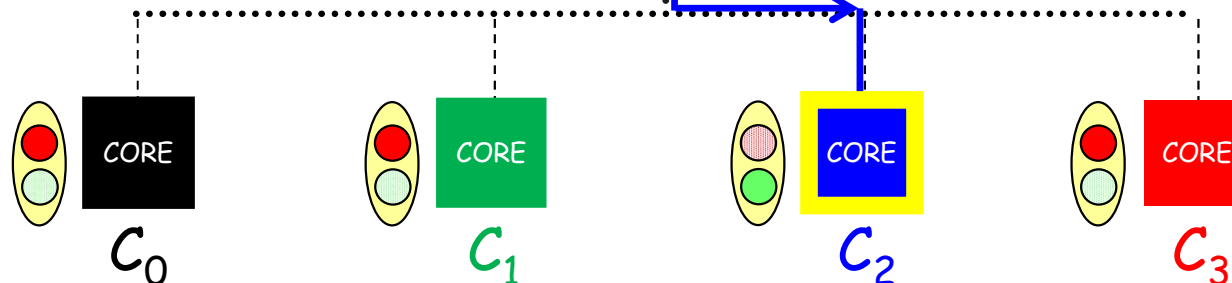
- Esempio:  $N=16$ ,  $p=4$



$\text{sumtot} = \text{sumtot} + s_2$

Calcolo

Scrittura



# Esempio: Somma 1strategia

- Esempio:  $N=16$ ,  $p=4$

$a_0$	$a_1$	$a_2$	$a_3$
$a_4$	$a_5$	$a_6$	$a_7$
$a_8$	$a_9$	$a_{10}$	$a_{11}$
$a_{12}$	$a_{13}$	$a_{14}$	$a_{15}$
$s_0$	$s_1$	$s_2$	$s_3$
	sumtot		

Memoria  
globale

$\text{sumtot} = \text{sumtot} + s_3$

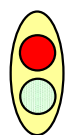
Calcolo

Scrittura



CORE

$C_0$



CORE

$C_1$



CORE

$C_2$

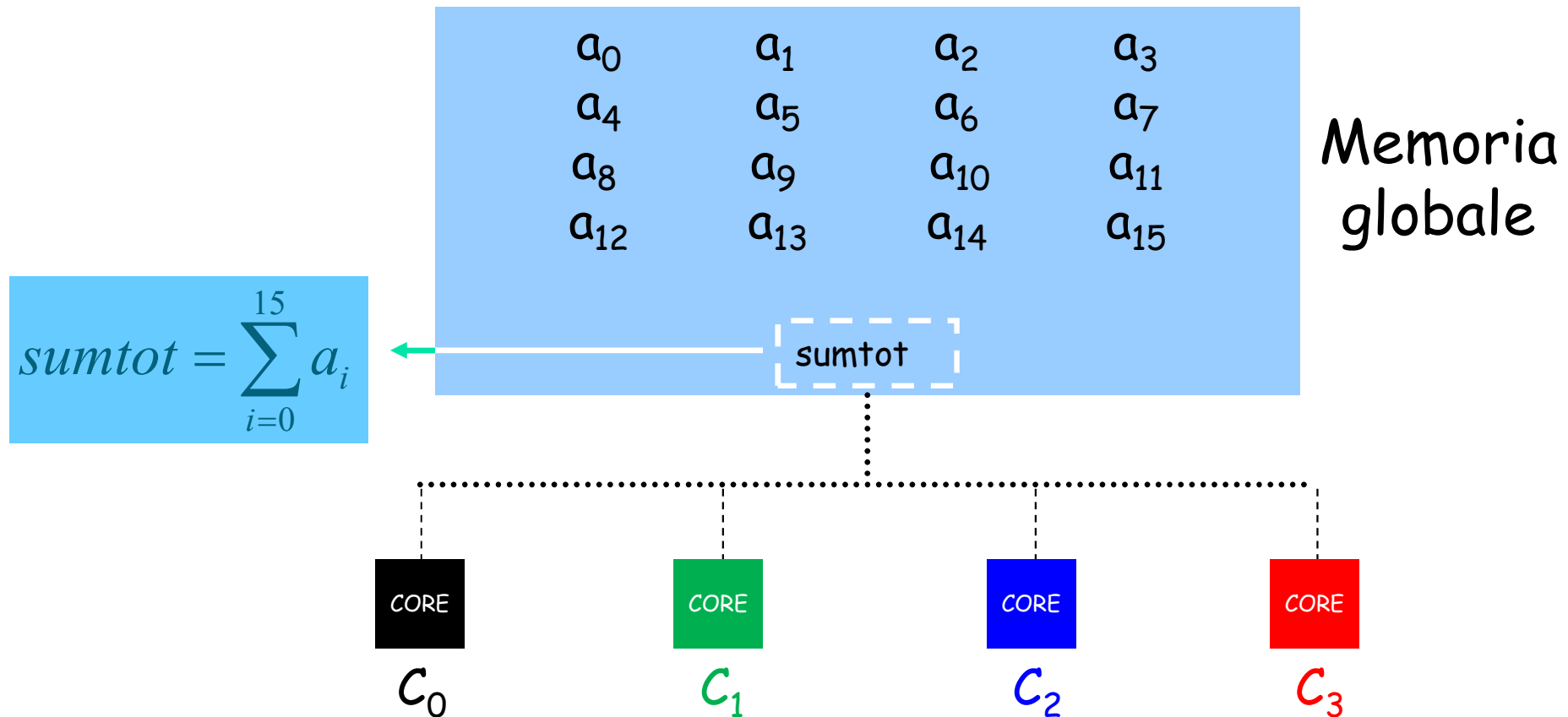


CORE

$C_3$

## Esempio: Somma 1strategia

- Esempio:  $N=16$ ,  $p=4$







## I strategia (MIMD-SM)

---

Ogni core

- calcola la propria somma parziale

Ad ogni passo

- ciascun core aggiunge la propria somma parziale ad un unico valore prestabilito

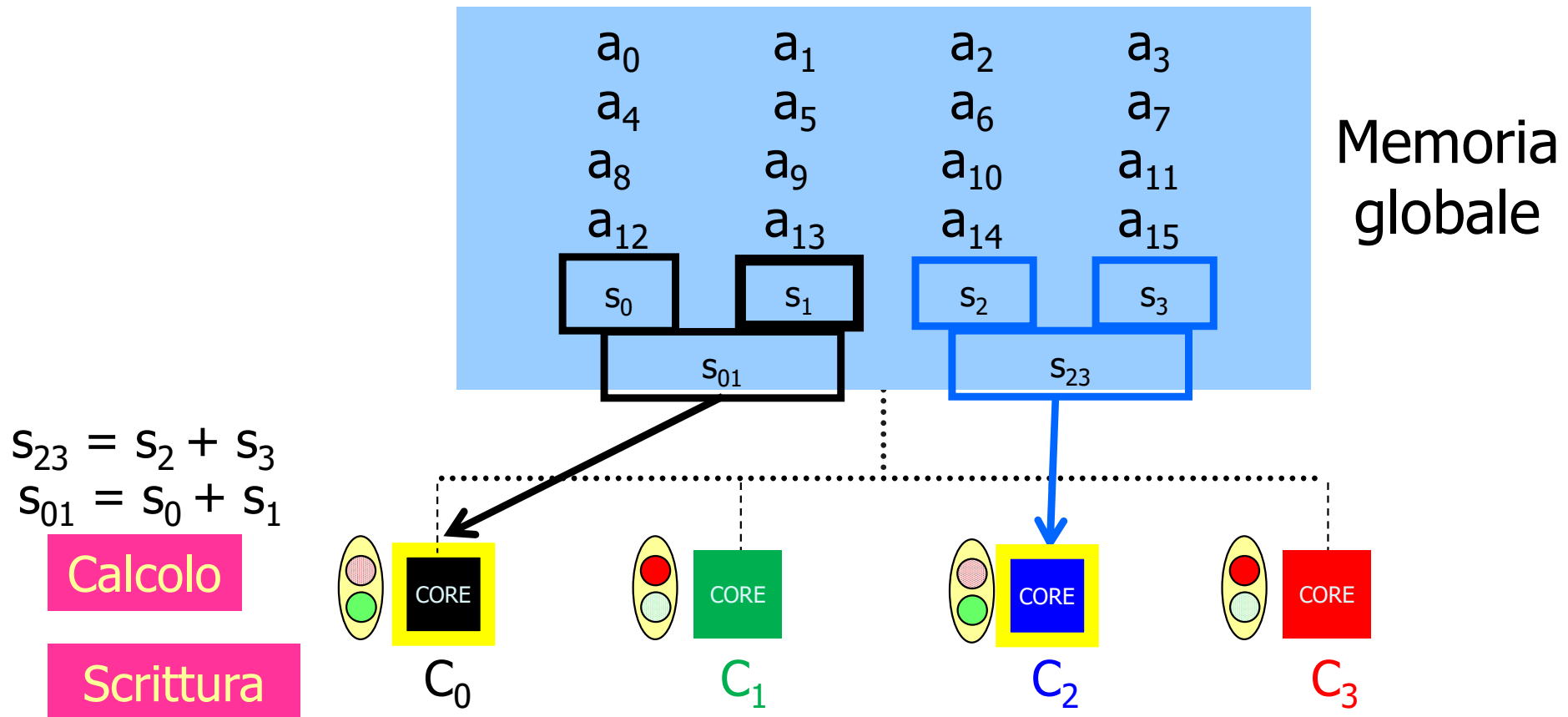
Il valore finale si trova nell'unica memoria condivisa.



Operazioni concorrenti

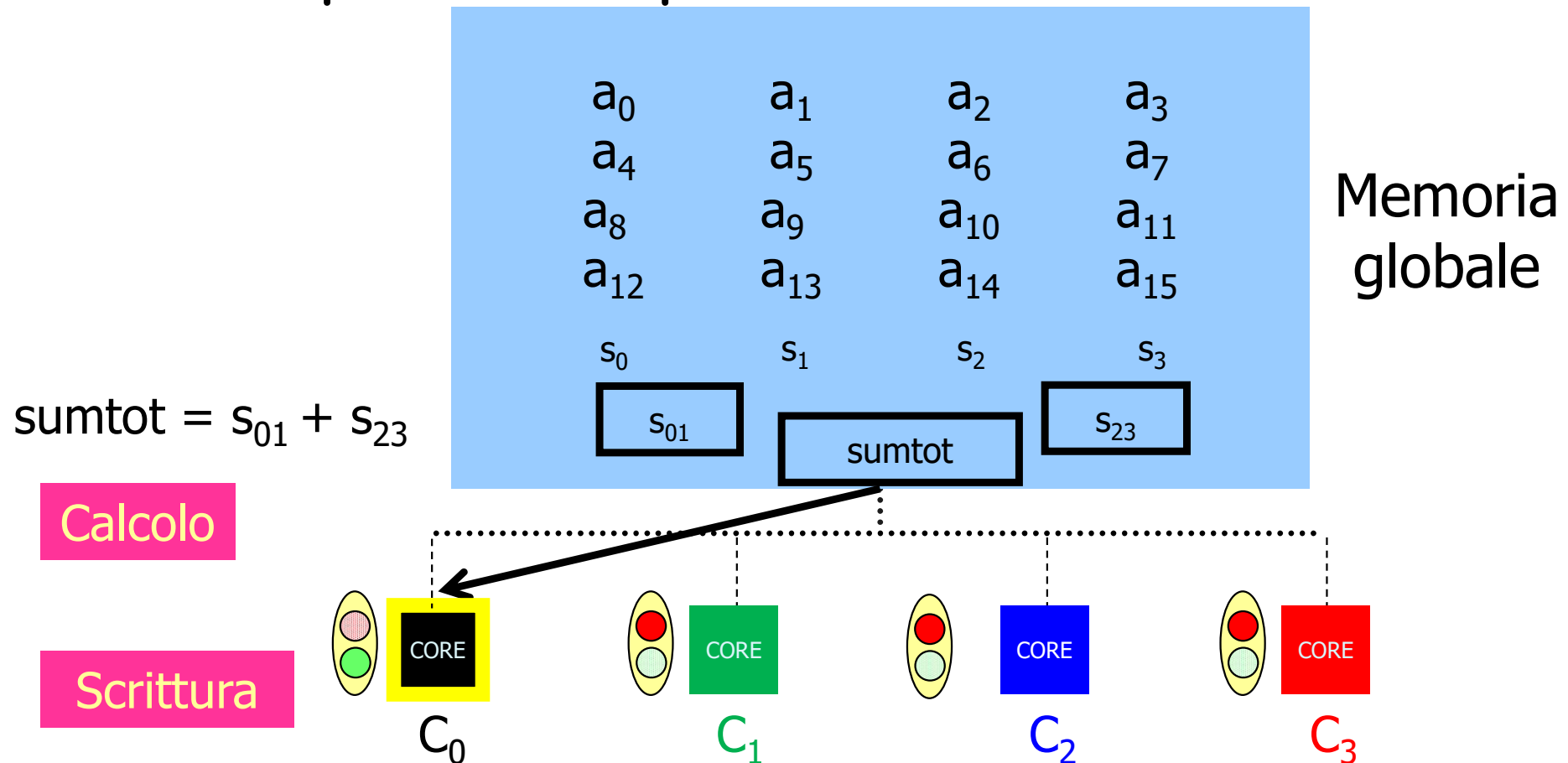
## Somma (II strategia MIMD-SM)

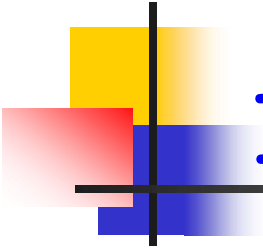
- Esempio:  $N=16$ ,  $p=4$



## Somma (II strategia MIMD-SM)

- Esempio:  $N=16$ ,  $p=4$





## II strategia (MIMD-SM)

Ogni core

- calcola la propria **somma parziale**.

Ad ogni passo,

- la metà dei core (rispetto al passo precedente) calcola un contributo della somma parziale.

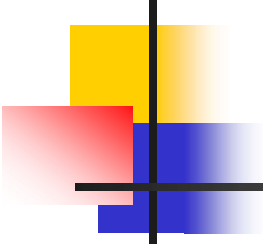
Il valore finale si trova  
nell'**unica memoria condivisa**.



Operazioni concorrenti



Operazioni  
quasi  
concorrenti



---

Esistono diversi strumenti per lo sviluppo di software in ambiente di calcolo MIMD-Shared Memory

OpenMp, Pthreads,  
Windows threads...