Calcolo Parallelo e Distribuito

A. Murli

a.a. 2005-2006

Problema

Progettare un algoritmo parallelo per l'ordinamento di un vettore su un calcolatore MIMD a memoria distribuita con p processori

Esempio:

- input

L= (25,7,1,9,81,3,28,12,6,20)

- output

L= (1,3,6,7,9,12,20,25,28,81)

(ordinamento crescente di 10 numeri interi)

Qual è l'algoritmo sequenziale?

- selection sort
- insertion sort
- bubble sort
- ……

$O(n) \leq T(n) \leq O(n^2)$ confronti

- Bitonic-merge sort

$O(n \log^2 n)$ confronti

- Quick sort

$O(n\log n) \leq T(n) \leq O(n^2)$ confronti

- Merge sort
- Heap sort

$O(n\log n)$ confronti

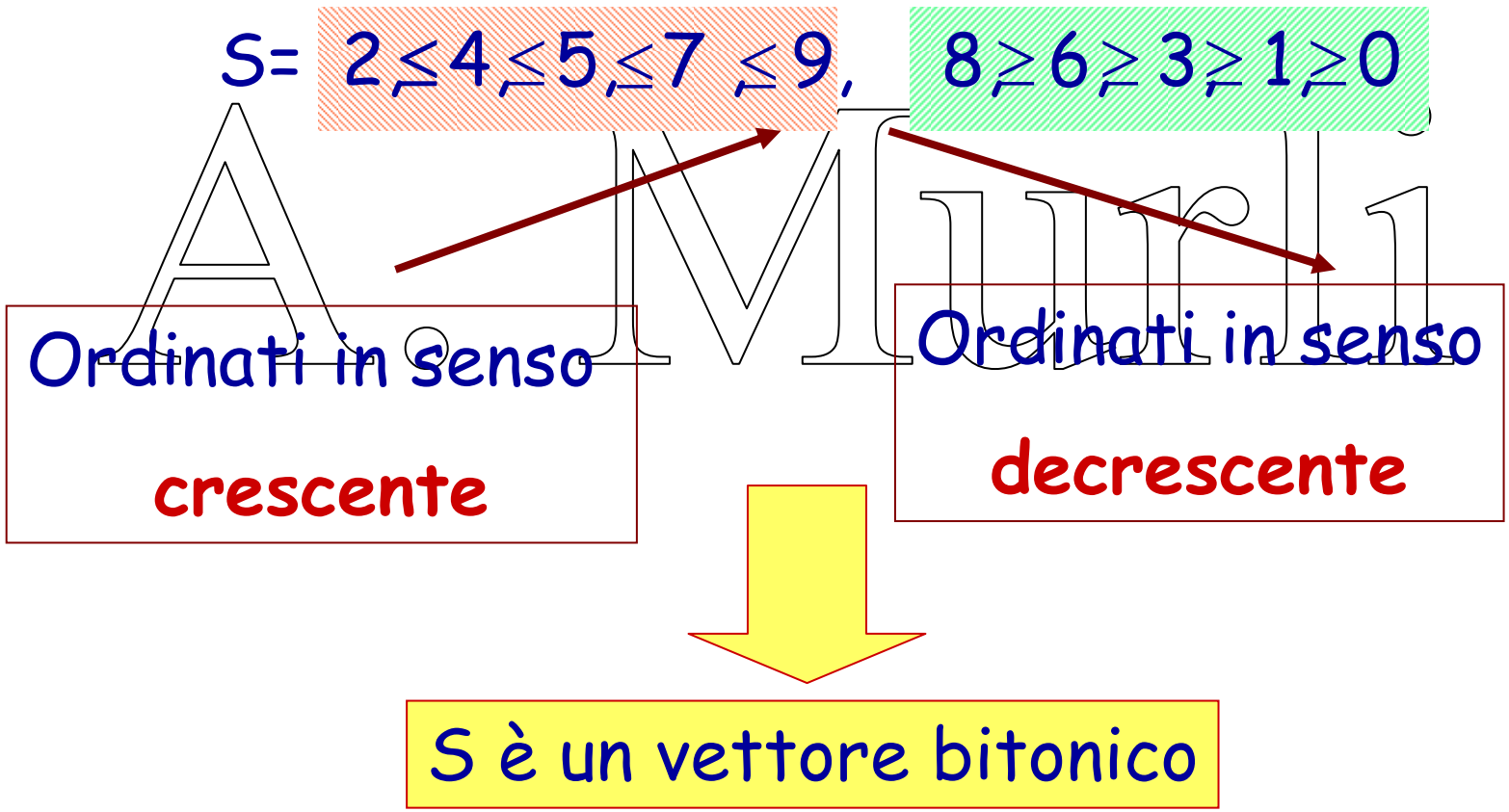
Punti da analizzare:

1. algoritmo per l'ordinamento di un vettore **"bitonico" mediante il** Bitonic Sort

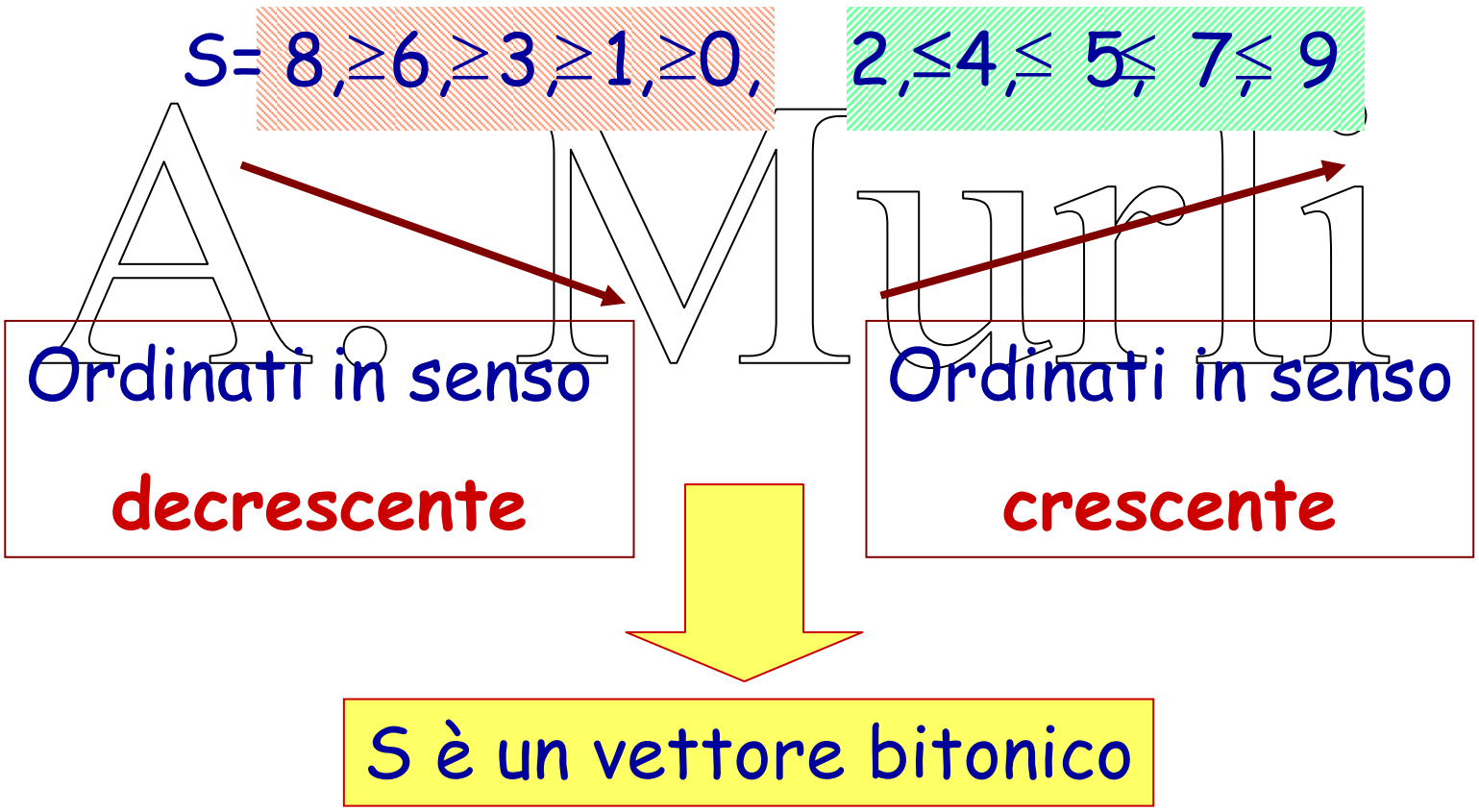
2. Algoritmo per l'ordinamento un vettore **qualsiasi mediante il** General Bitonic Sort

3. Algoritmo **Parallelo** per l'ordinamento di un **vettore qualsiasi** (Parallel General Bitonic Sort)

Cos' è un vettore bitonico ?



Cos' è un vettore bitonico ?



Cos' è un vettore bitonico ?

$S = s_1\, s_2\, s_3 \ldots\ldots s_n$ è un vettore bitonico

Definizione

Esiste un indice k $(1 \le k \le n)$ tale che

$S = s_1\, s_2\, s_3\, s_k \ldots\ldots s_n$

$s_1 \le s_2 \le s_3 \le \ldots\ldots \le s_k$ $\quad$ $s_{k+1} \ge s_{k+2} \ge s_{k+3} \ge \ldots\ldots \ge s_n$

(se k=n il vettore si dice MONOTONICO)

Cos' è un vettore bitonico ?

$S = s_1\, s_2\, s_3 \ldots\ldots s_n$ è un vettore bitonico

Definizione

Esiste un indice k $(1 \leq k \leq n)$ tale che

$S = s_1\, s_2\, s_3\, s_k \ldots\ldots s_n$

$s_1 \geq s_2 \geq s_3 \geq \ldots\ldots \geq s_k$

$s_{k+1} \leq s_{k+2} \leq s_{k+3} \leq \ldots\ldots \leq s_n$

(se k=n il vettore si dice MONOTONICO)

Come ordinare un vettore bitonico?

BBS: IDEA

Fase di SORT

partizionare il vettore

di lunghezza $n = 2^q$

in un insieme di coppie bitoniche

di lunghezza 2

Come ordinare una sequenza bitonica?

BBS: IDEA

Fase di MERGE

Ordinare le coppie bitoniche

in modo da avere un vettore

globalmente ordinato

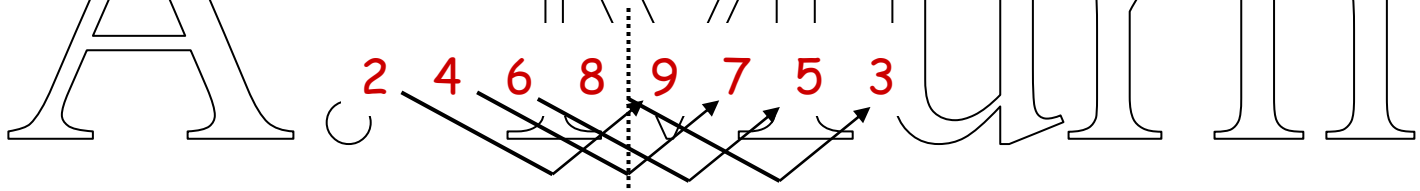
BBS: Ordinamento crescente di un vettore bitonico

Esempio: $n=8=2^3$

2 4 6 8 | 9 7 5 3

è un vettore bitonico

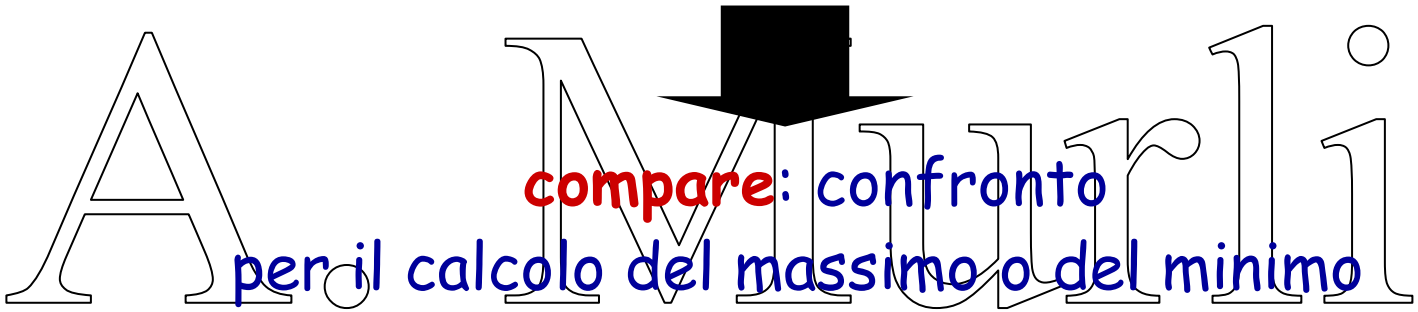
Passo 1 – si confrontano gli elementi a distanza d=n/2=8/2=4.



e si spostano i più piccoli a sinistra e i più grandi a destra

BBS: Ordinamento di una lista bitonica

ad ogni passo e per ogni coppia,
si effettuano operazioni di
compare-exchange

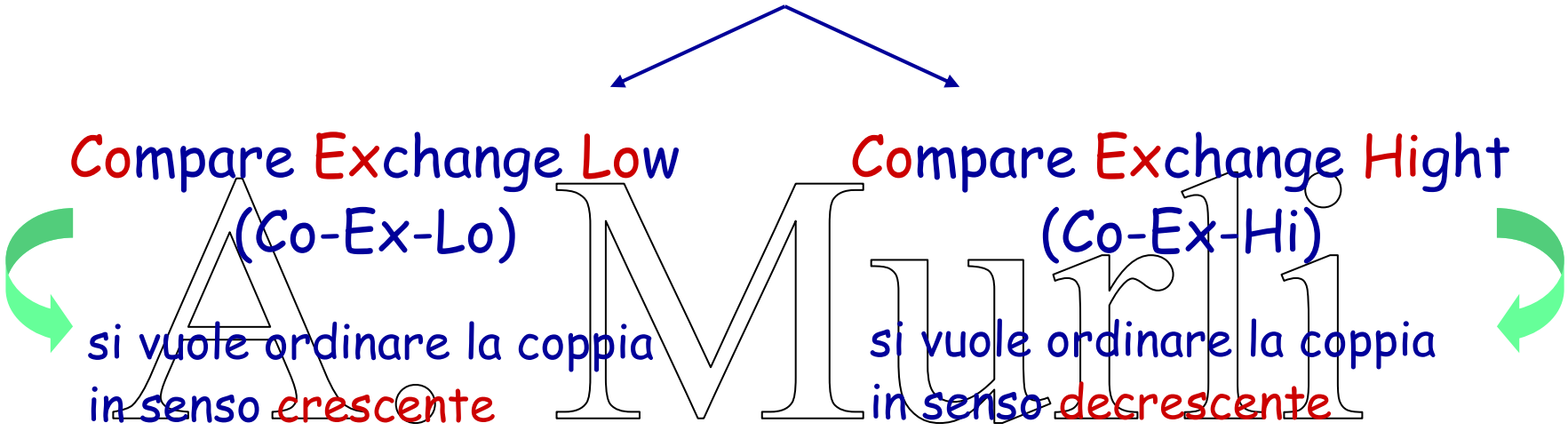


compare: confronto
per il calcolo del massimo o del minimo

exchange: Scambio
per la costruzione della coppia ordinata

BBS: Ordinamento di una lista bitonica

due tipi di **compare-exchange**



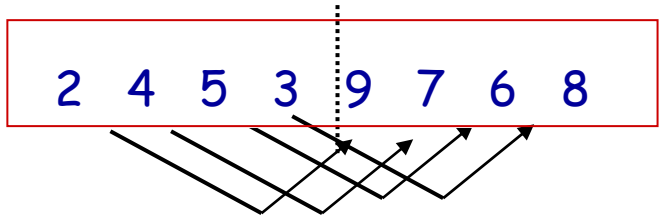
(min, max)

(max, min)

BBS:Ordinamento crescente di una lista bitonica

Applichiamo il Co-Ex-Lo 2 4 6 8 9 7 5 3

passo 1



min (2, 9) = 2

max (2, 9) = 9

min (4, 7) = 4

max (4, 7) = 7

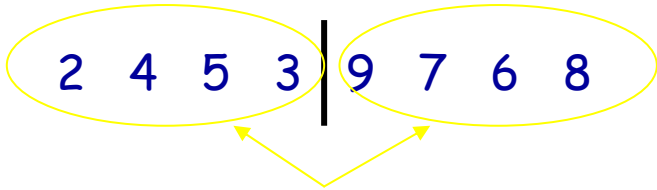
min (6, 5) = 5

max (6, 5) = 6

min (8, 3) = 3

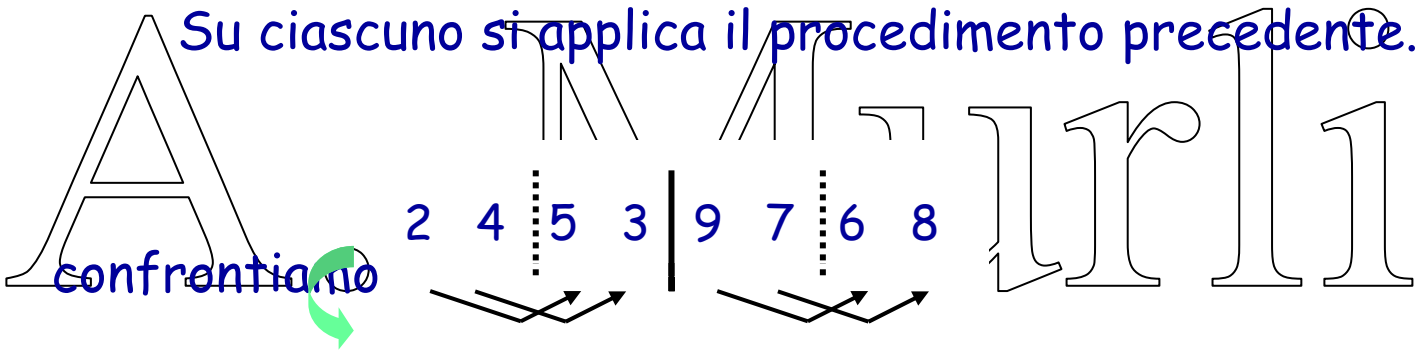
max (8, 3) = 8

BBS: Ordinamento crescente di una lista bitonica

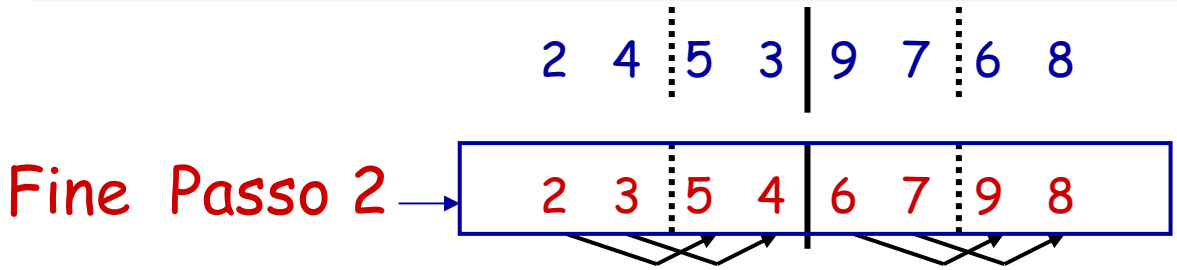


Passo 2 Si considerano i 2 sottovettori bitonici costruiti al Passo 1

Su ciascuno si applica il procedimento precedente.

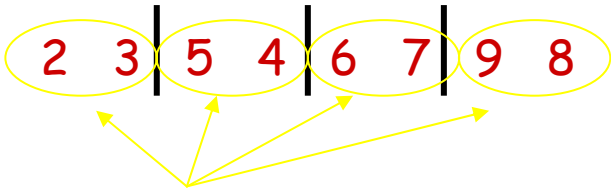


BBS:Ordinamento crescente di una lista bitonica



min (2, 5) = 2
max (2, 5) = 5
min (9, 6) = 6
max (9, 6) = 9
min (4, 3) = 3
max (4, 3) = 4
min (7, 8) = 7
max (7, 8) = 8

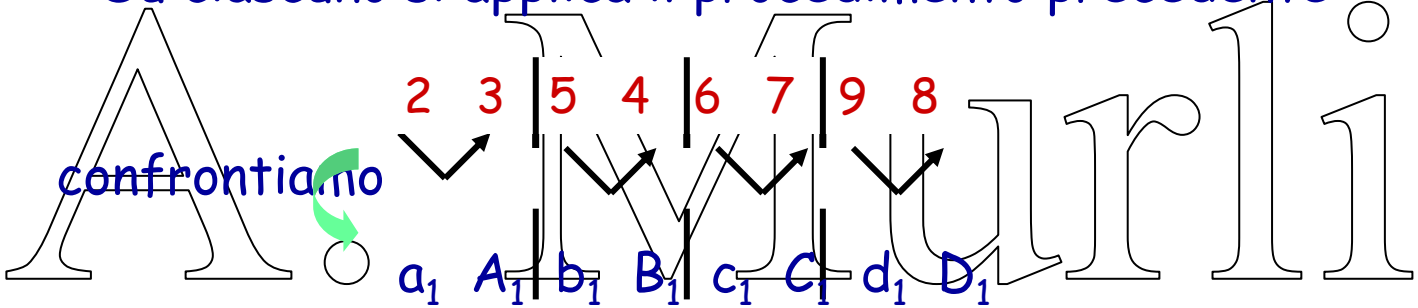
BBS:Ordinamento crescente di una lista bitonica



Passo 3

Si considerano i 4 sottovettori bitonici ottenuti dal passo 2

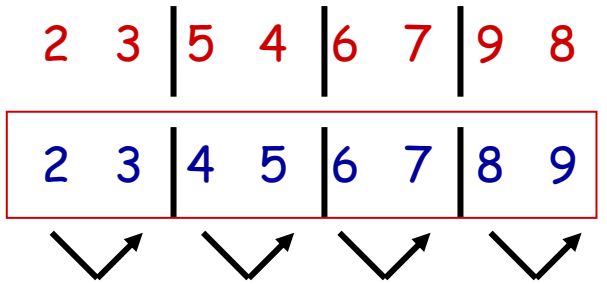
Su ciascuno si applica il procedimento precedente



(a_1, A_1) —Co-Ex-Lo→ (a_1^1, A_1^1) $\quad$ (c_1, C_1) —Co-Ex-Lo→ (c_1^1, C_1^1)

(b_1, B_1) —Co-Ex-Lo→ (b_1^1, B_1^1) $\quad$ (d_1, D_1) —Co-Ex-Lo→ (d_1^1, D_1^1)

BBS:Ordinamento crescente di una lista bitonica



Fine III passo

Lista ordinata

min (2, 3) = 2

max (2, 3) = 3

min (5, 4) = 4

max (5, 4) = 5

min (6, 7) = 6

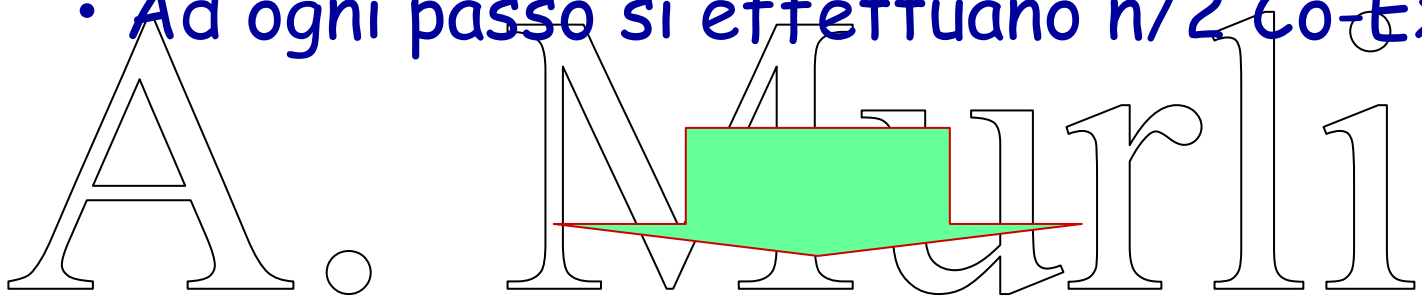
max (6, 7) = 7

min (9, 8) = 8

max (9, 8) = 9

Qual è la complessità di tempo?

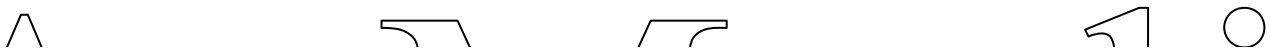
- L'algoritmo è costituito da $\log_2 n$ passi
- Ad ogni passo si effettuano n/2 Co-Ex



La complessità di tempo dell'algoritmo BBS è $O(n \log_2 n)$

Punti da analizzare:

1. Descrizione dello schema per l'ordinamento **Sequenziale** di una **vettore bitonico** (Bitonic Sort)



2. Descrizione dello schema per l'ordinamento **Sequenziale** di un **vettore qualsiasi** (General Bitonic Sort)

3. Descrizione dello schema per l'ordinamento **Parallelo** di un **vettore qualsiasi** (Parallel General Bitonic Sort)

Come ordinare un vettore qualsiasi ?

GBS: IDEA

Trasformare il vettore assegnato in uno bitonico

Come ordinare un vettore qualsiasi ?

GBS: IDEA

Ordinare il vettore bitonico con il BBS

GBS: Ordinamento di un vettore

Per ordinare un vettore di lunghezza $n=2^p$ utilizzando il GBS si distinguono due fasi:

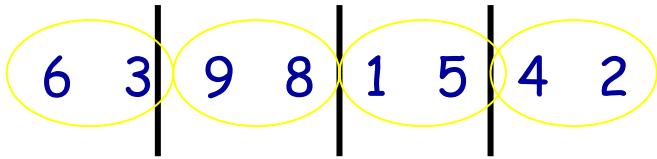
Fase a) Rendere bitonico il vettore ⇨ primi p-1 passi

Fase b) Ordinare il vettore costruito nella fase a). ⇨ p-esimo passo

⇩

$\log_2 n = p$ passi.

GBS: STEP 1 (fase a)



Per trasformare il vettore in uno bitonico partizioniamolo il

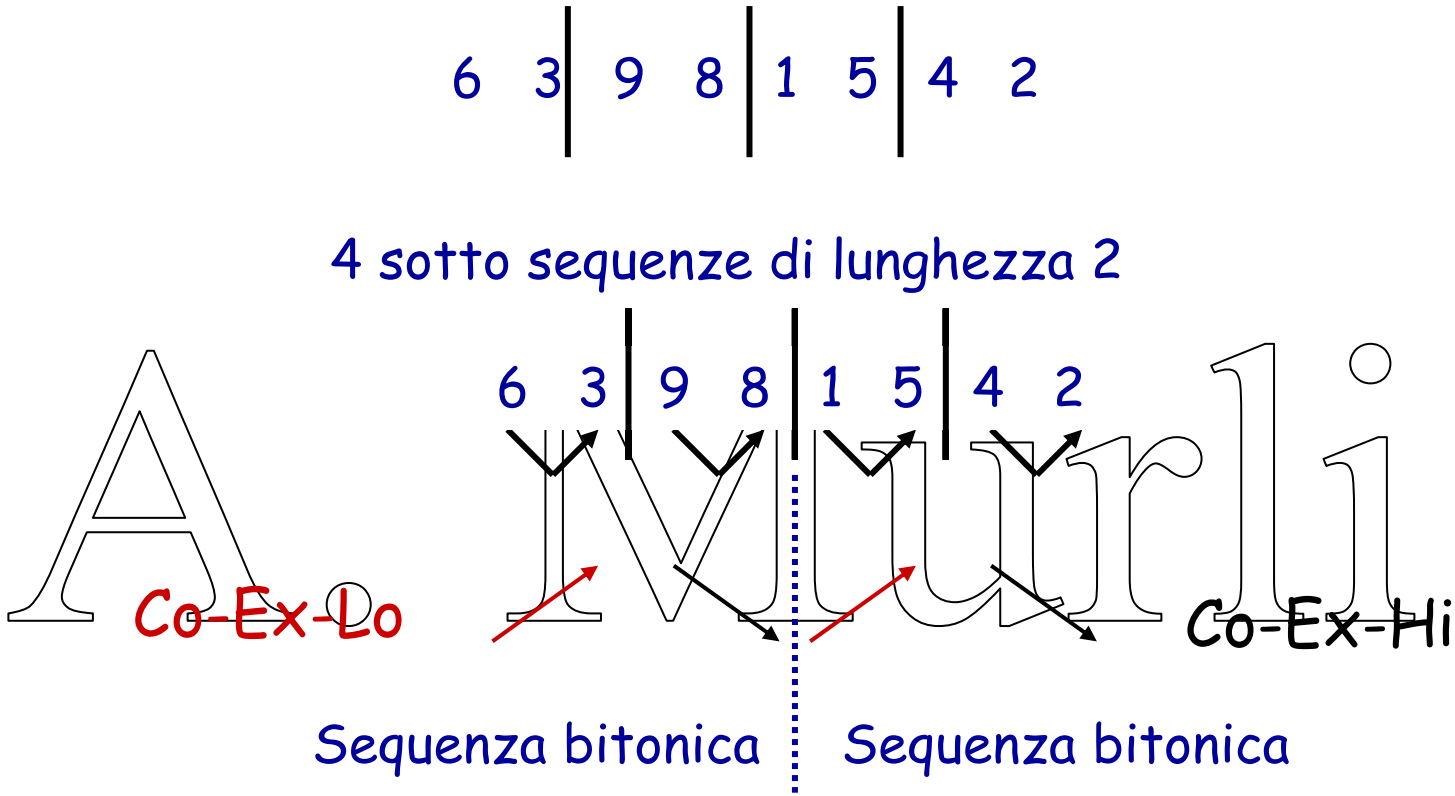
4 coppie

ovvero in 4 sottosequenze bitoniche di lungh = 2

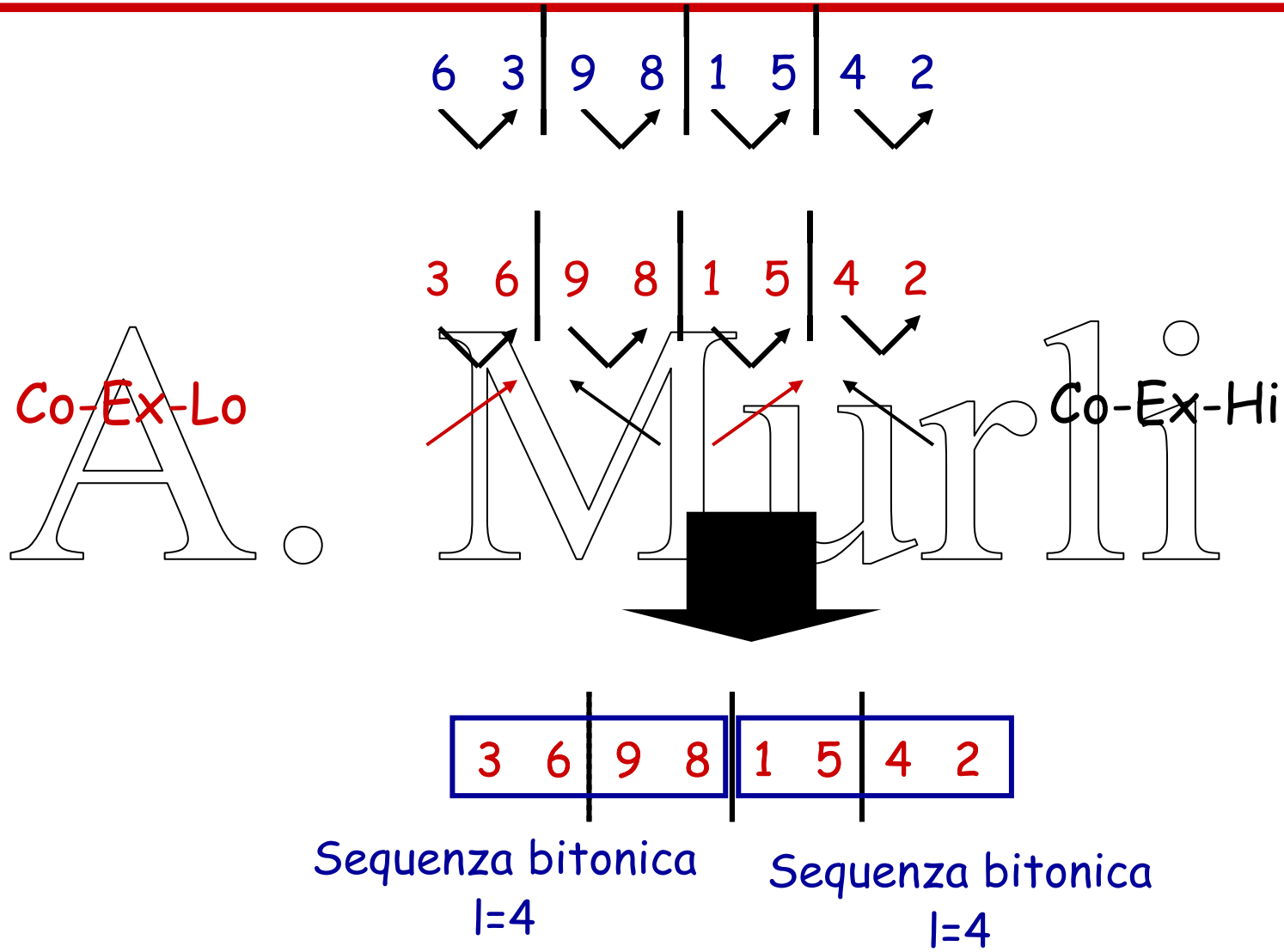
e ordiniamole in modo da avere

2 sottosequenze bitoniche di lungh = 4

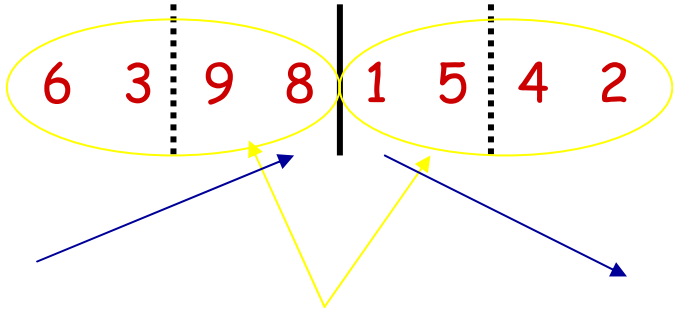
GBS: STEP 1


6 3 9 8 1 5 4 2
4 sotto sequenze di lunghezza 2
6 3 9 8 1 5 4 2
A Murli
Co-Ex-Lo
Co-Ex-Hi
Sequenza bitonica
Sequenza bitonica

GBS: STEP 1


6 3 9 8 1 5 4 2
3 6 9 8 1 5 4 2
Co-Ex-Lo
Co-Ex-Hi
A. Murli
3 6 9 8 1 5 4 2
Sequenza bitonica
l=4
Sequenza bitonica
l=4

GBS: STEP 2 (fase a)



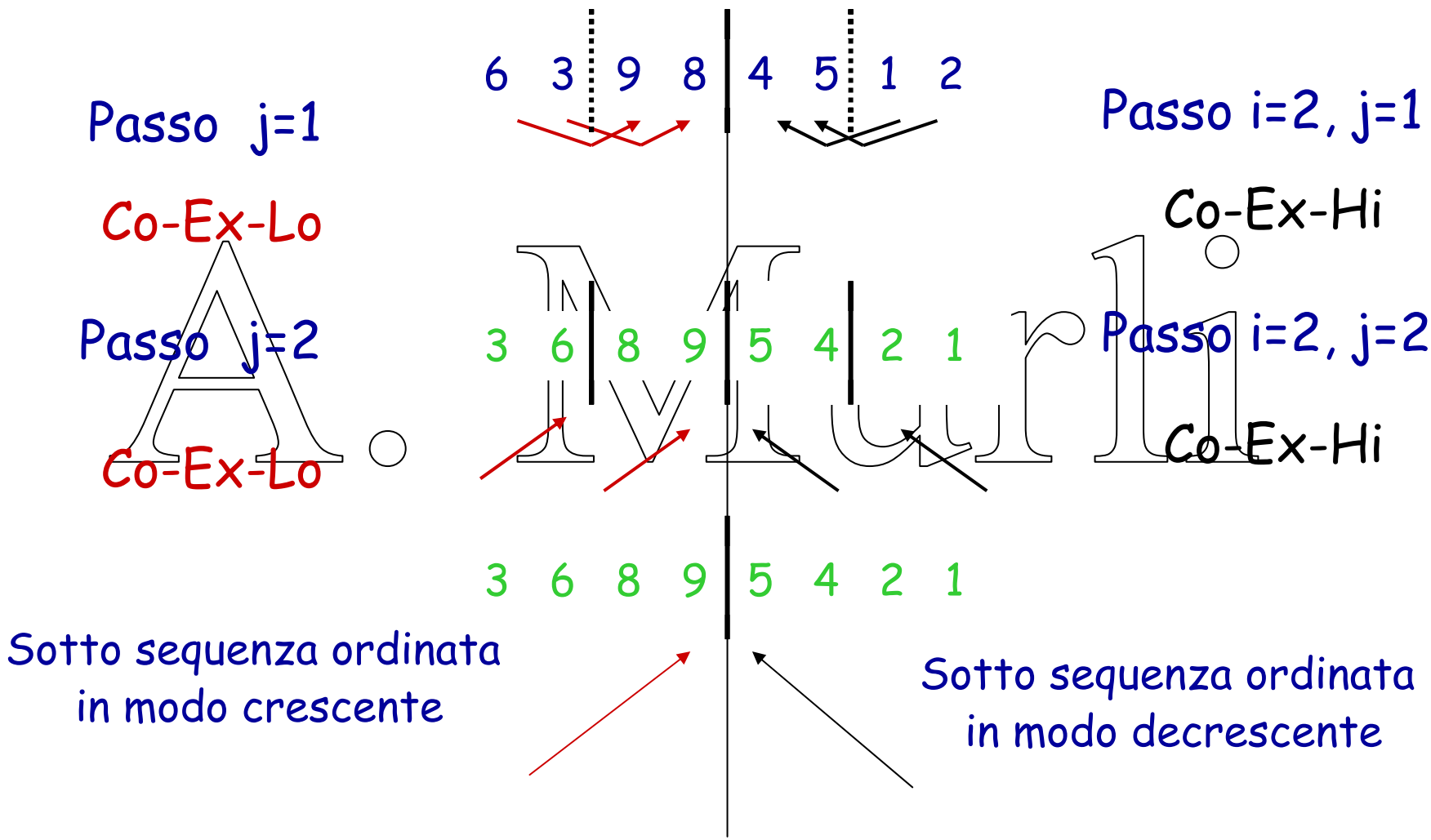
Ordiniamo ciascuno dei 2 sottovettori (con il BBS)

in modo da avere un unico vettore bitonico

Il primo in ordine crescente

il secondo in ordine decrescente.

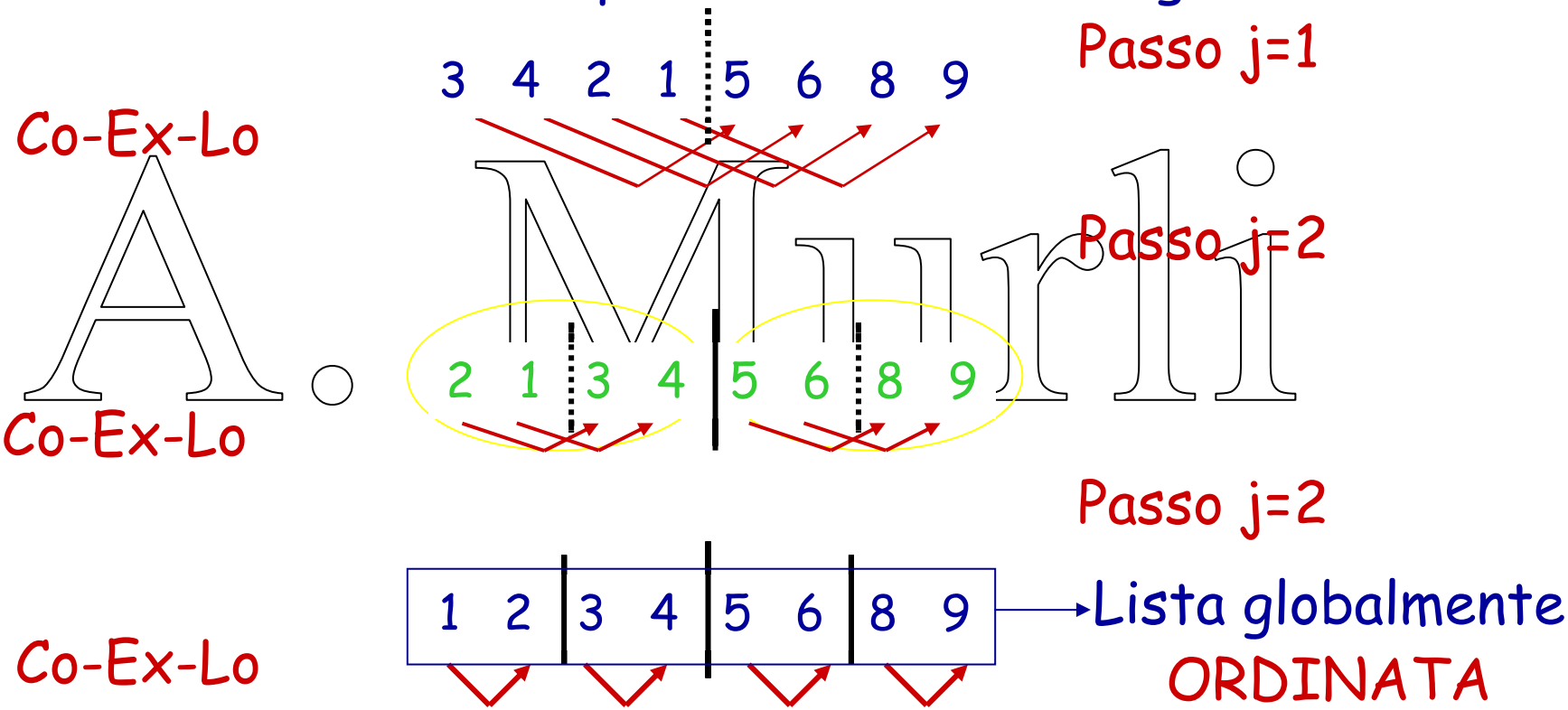
GBS: STEP 2


6 3 9 8 4 5 1 2
Passo j=1
Co-Ex-Lo
Passo i=2, j=1
Co-Ex-Hi
3 6 8 9 5 4 2 1
Passo j=2
Co-Ex-Lo
Passo i=2, j=2
Co-Ex-Hi
3 6 8 9 5 4 2 1
Sotto sequenza ordinata in modo crescente
Sotto sequenza ordinata in modo decrescente

GBS: STEP 3 (fase b)

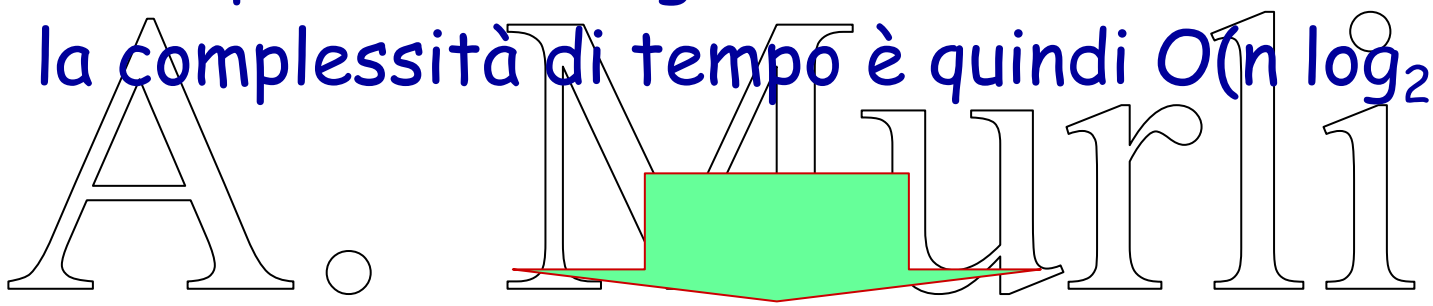
Passo j=1,2,3

Ordinamento di una sequenza bitonica di lunghezza l=8



Qual è la complessità di tempo?

- L'algoritmo è costituito da $\log_2 n$ passi
- Ad ogni passo si utilizza l'algoritmo BBS su sequenze di lunghezza minore di n, la complessità di tempo è quindi $O(n \log_2 n)$



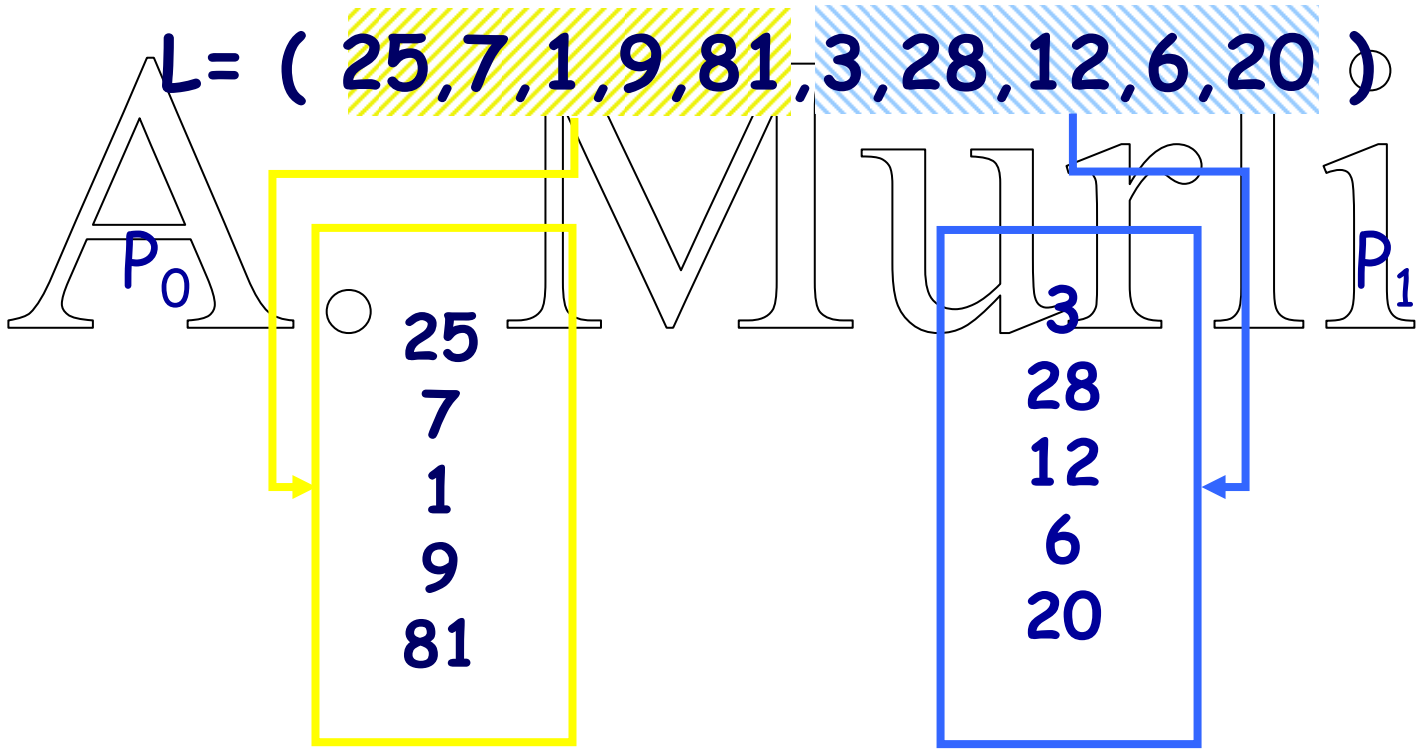
La complessità di tempo dell'algoritmo GBS è $O(n \log_2^2 n)$

Punti da analizzare:

1. Descrizione dello schema per l'ordinamento **Sequenziale** di una **lista bitonica** (Bitonic Sort)
2. Descrizione dello schema per l'ordinamento **Sequenziale** di una **lista generica** (General Bitonic Sort)
3. Descrizione dello schema per l'ordinamento **Parallelo** di una **lista generica** (Parallel General Bitonic Sort)

Esempio: p=2

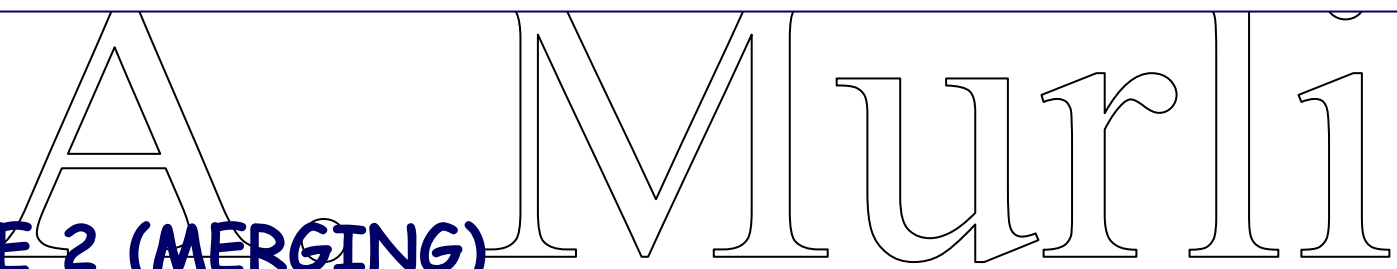
- Distribuiamo il vettore



Strategia generale

FASE 1 (SORTING LOCALE)

Ciascun processore ordina il proprio vettore utilizzando un **algoritmo di ordinamento**

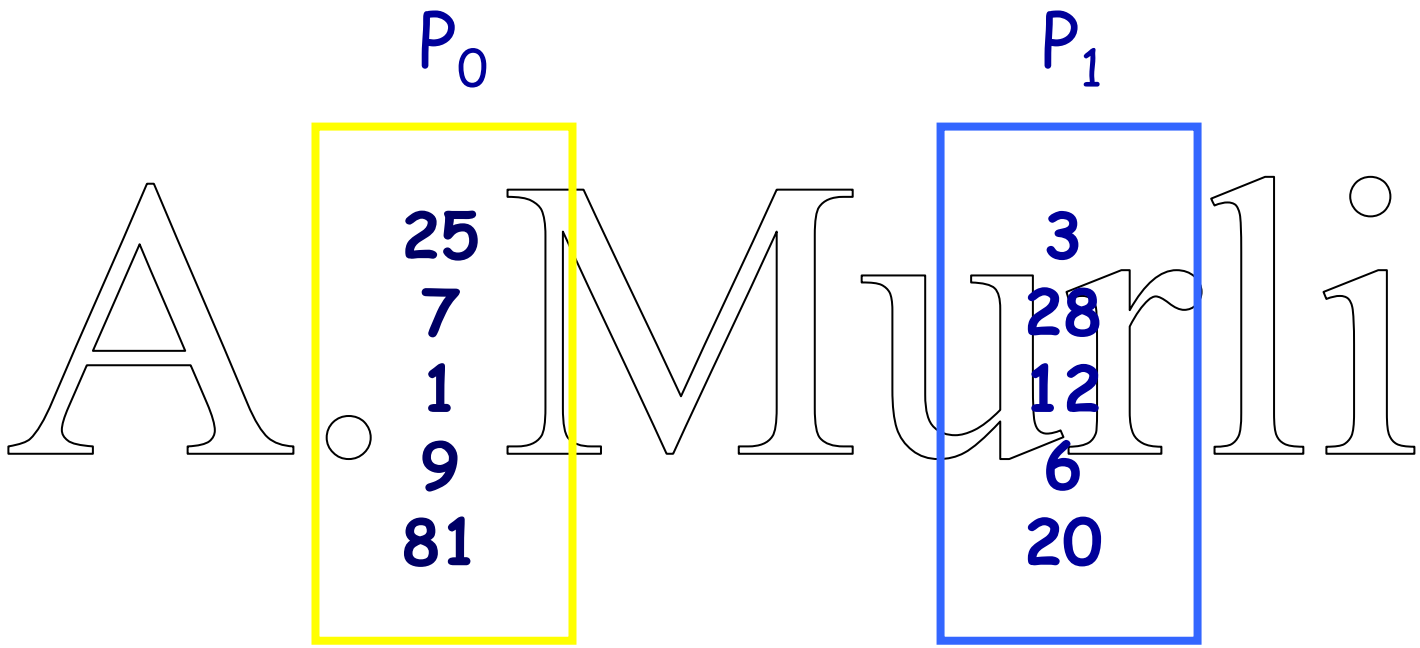


FASE 2 (MERGING)

I vettori ordinati sono opportunamente combinati in modo da ottenere un vettore globalmente ordinato

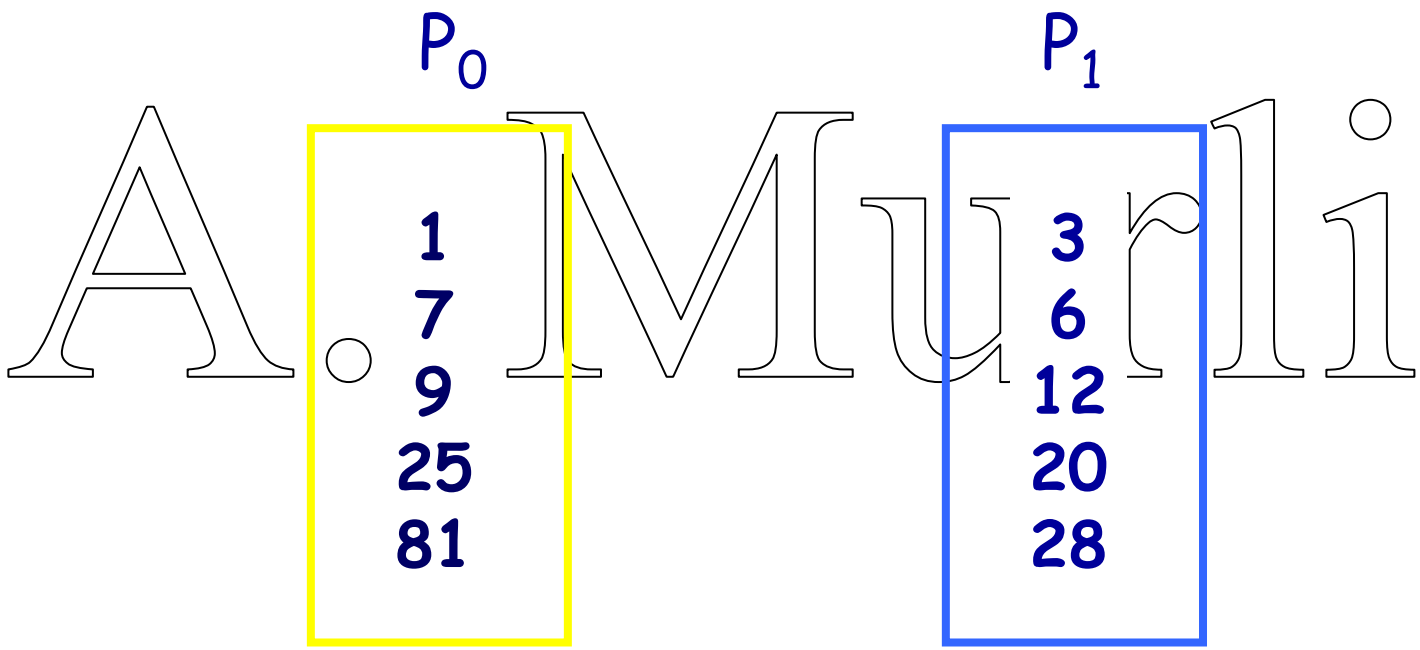
PGBS: Ordinamento di un vettore

FASE 1 (SORTING LOCALE)



Ogni processore ordina in modo crescente
il proprio vettore

PGBS: Ordinamento di un vettore

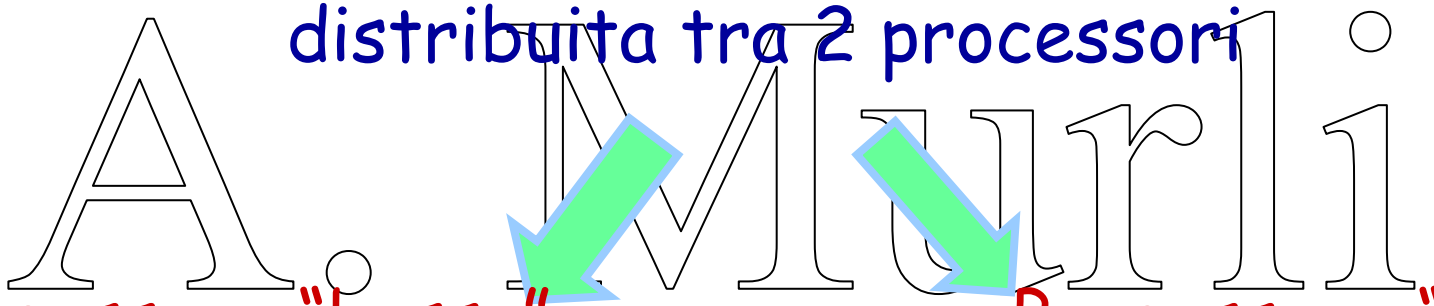


Fine Fase SORTING LOCALE

PGBS:Ordinamento di un vettore

FASE 2 (MERGING)

l'operazione di
COMPARE EXCHANGE viene applicata alla sequenza distribuita tra 2 processori



Processore "basso" Processore "alto"

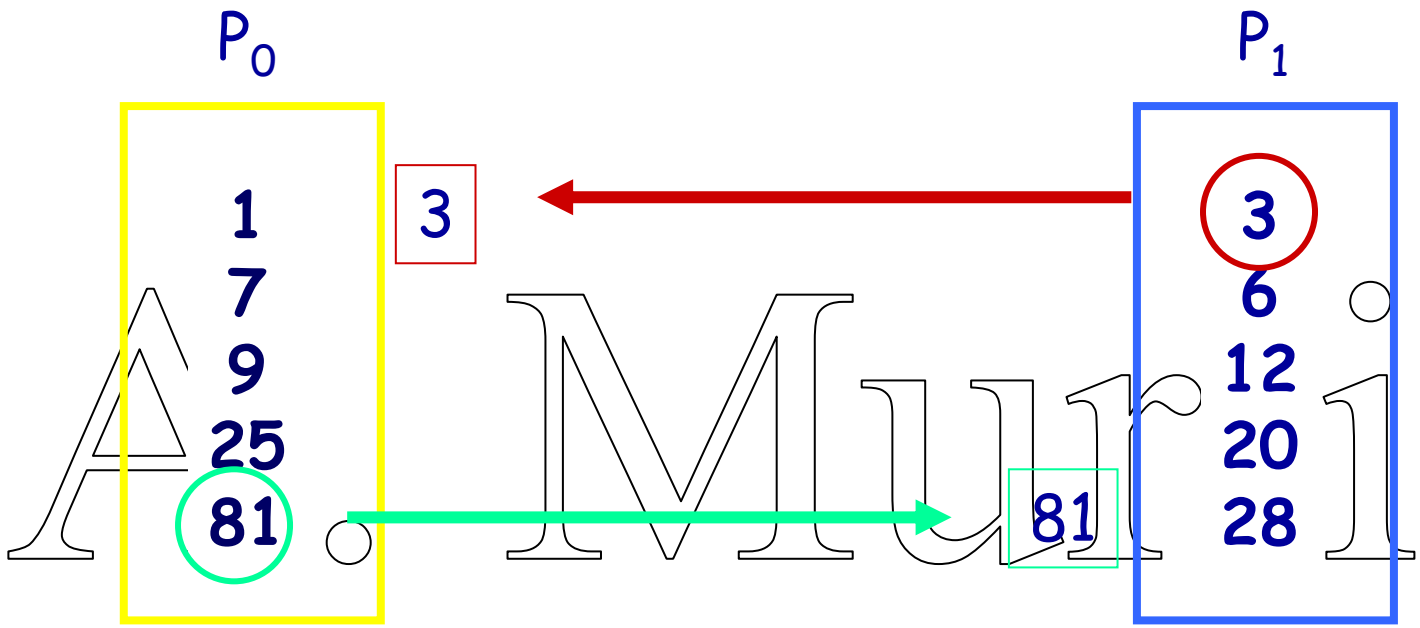
Il processore P_0 contiene gli elementi più piccoli

Il processore P_1 contiene gli elementi più grandi

Co-Ex-Lo

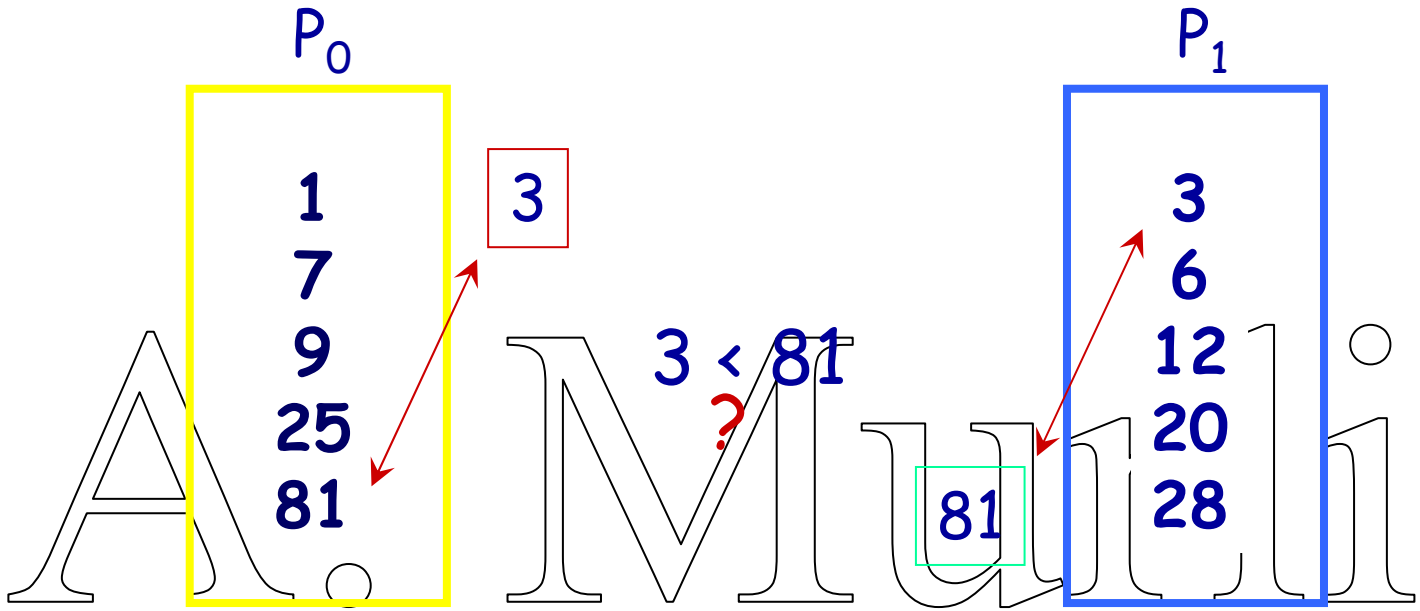
Co-Ex-Hi

I PASSO: step 1.0 → COMUNICAZIONE



Step 1.0: P_0 invia a P_1 il proprio max = 81,
mentre P_1 invia a P_0 il proprio min = 3

I PASSO : step 1.1 → CONFRONTO

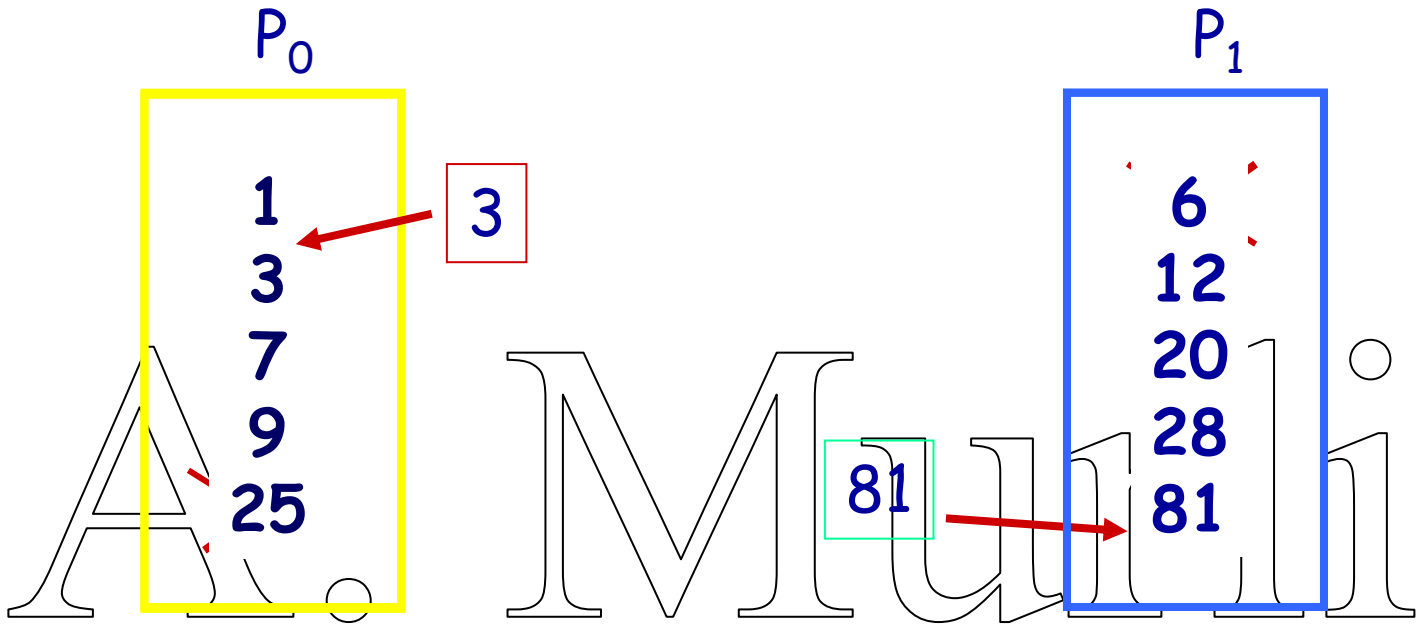


IN PARALLELO

Step 1.1: P_0 confronta 3 con 81: se è minore lo inserisce;

P_1 confronta 81 con 3:se è maggiore lo inserisce

I PASSO: step 1.2 → INSERIMENTO

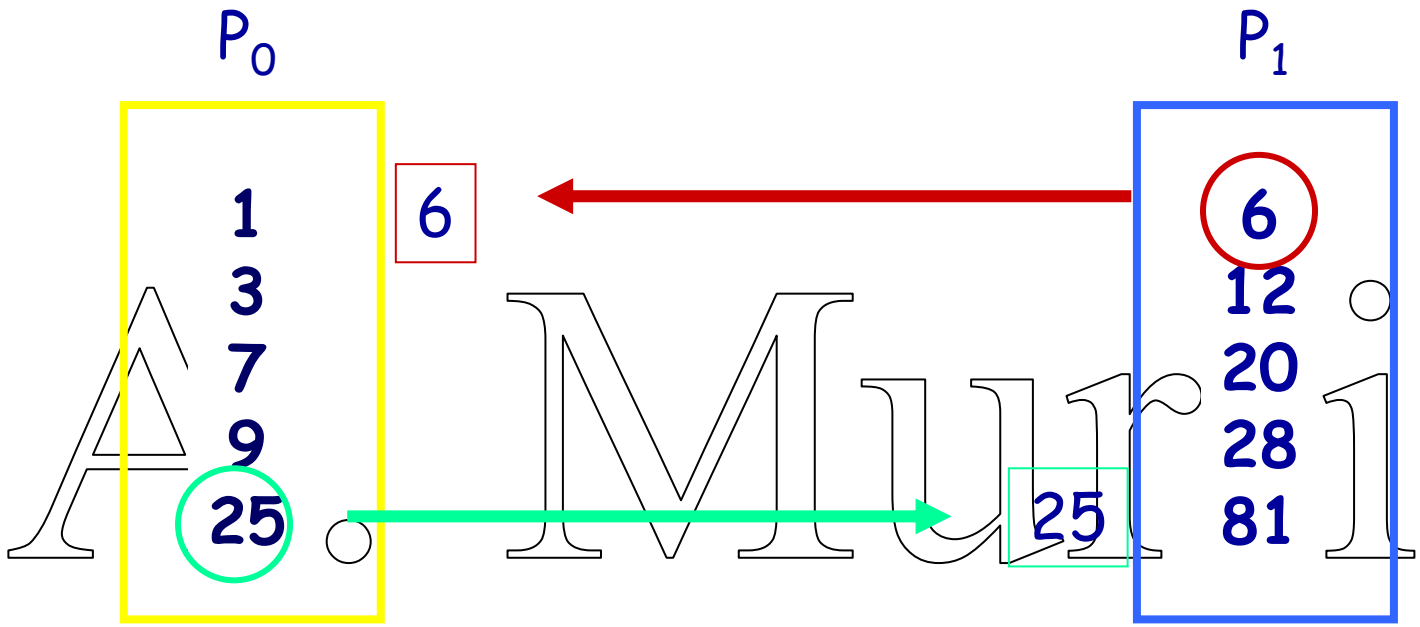


IN PARALLELO

Step 1.2: Ciascun processore

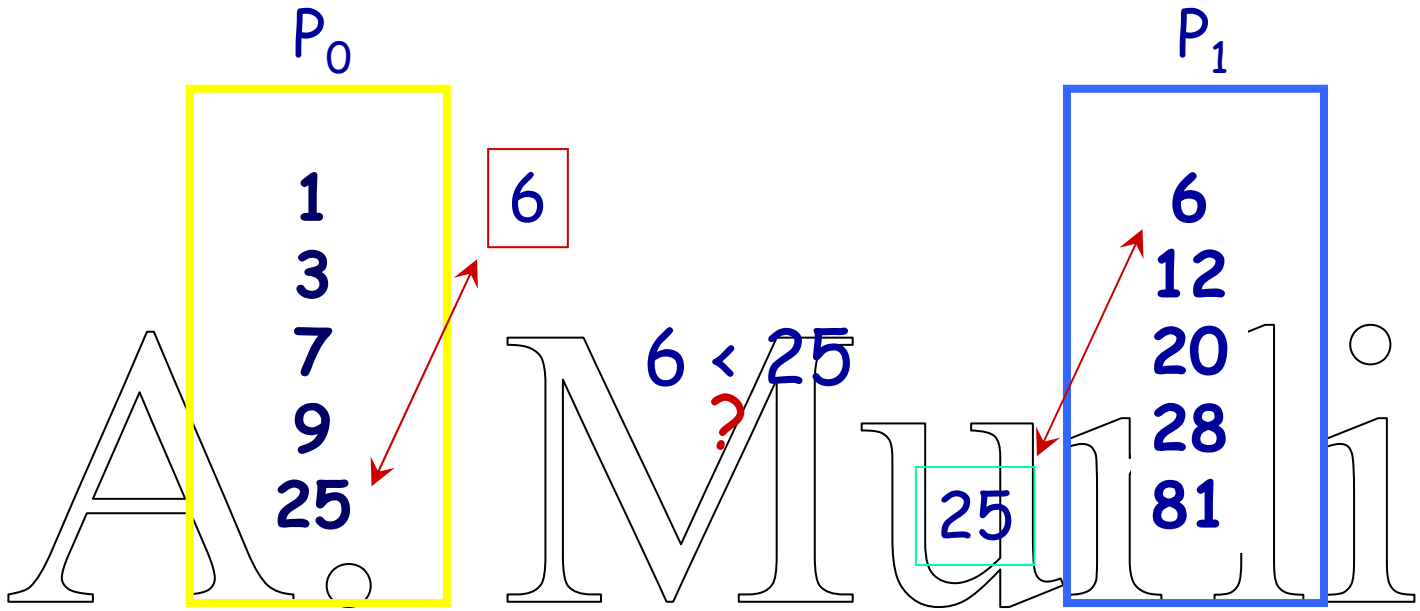
1) inserisce il numero ricevuto ,
2) elimina il numero inviato
3) riordina il vettore

II PASSO: step 2.0 → COMUNICAZIONE



Step 1.0: P_0 invia a P_1 il proprio max = 25,
mentre P_1 invia a P_0 il proprio min = 6

II PASSO: step 2.1 → CONFRONTO

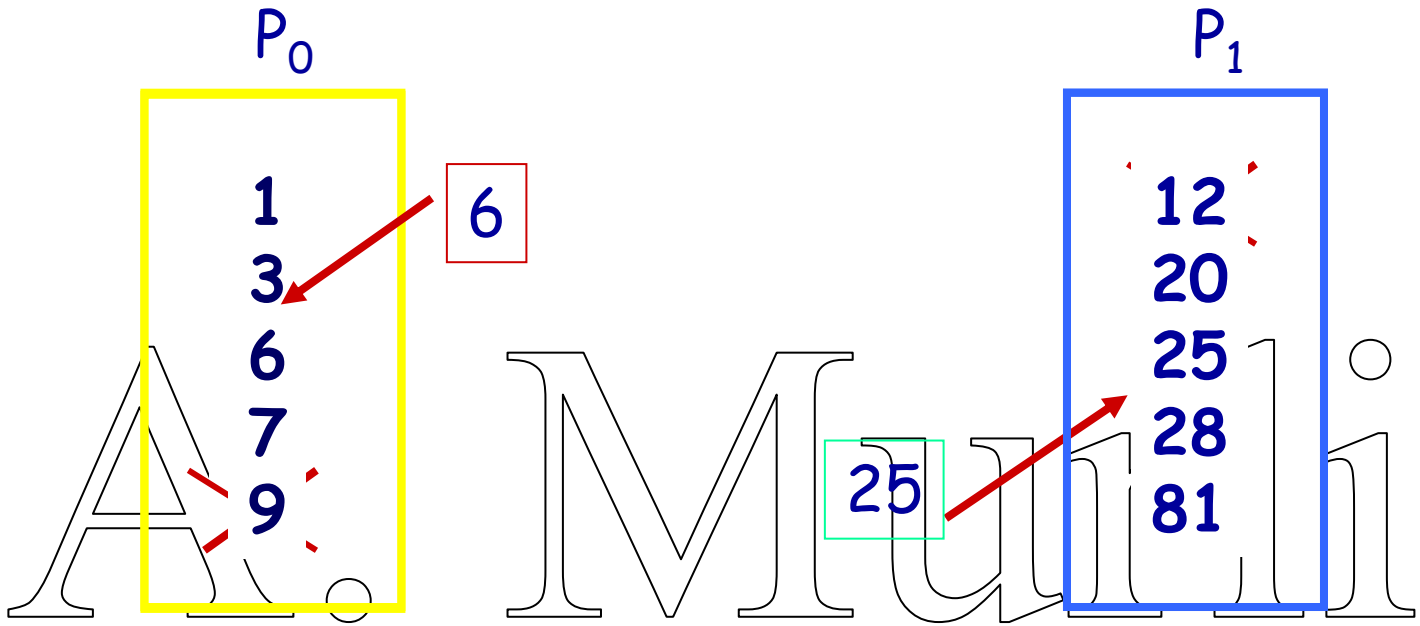


IN PARALLELO

Step 1.1: P_0 confronta 6 con 25: se è minore lo inserisce;

P_1 confronta 25 con 6: se è maggiore lo inserisce

II PASSO: step 2.2 → INSERIMENTO

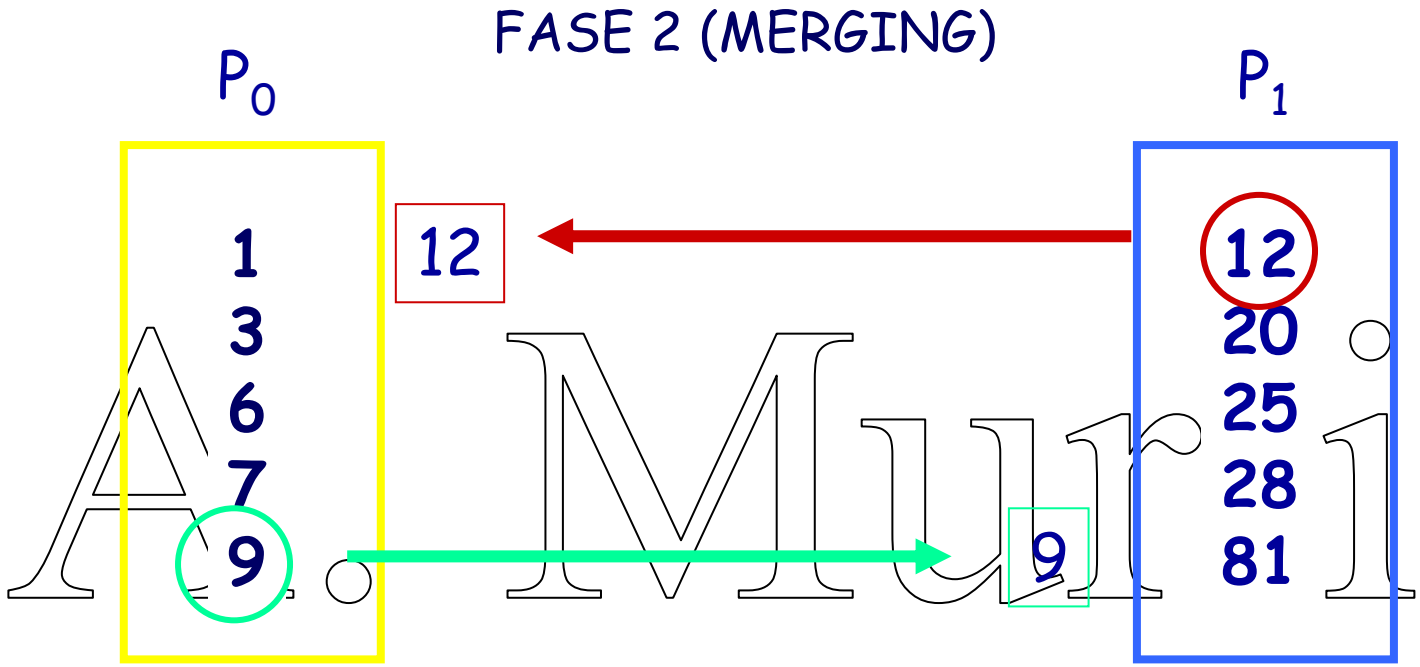


IN PARALLELO

Step 1.2: Ciascun processore

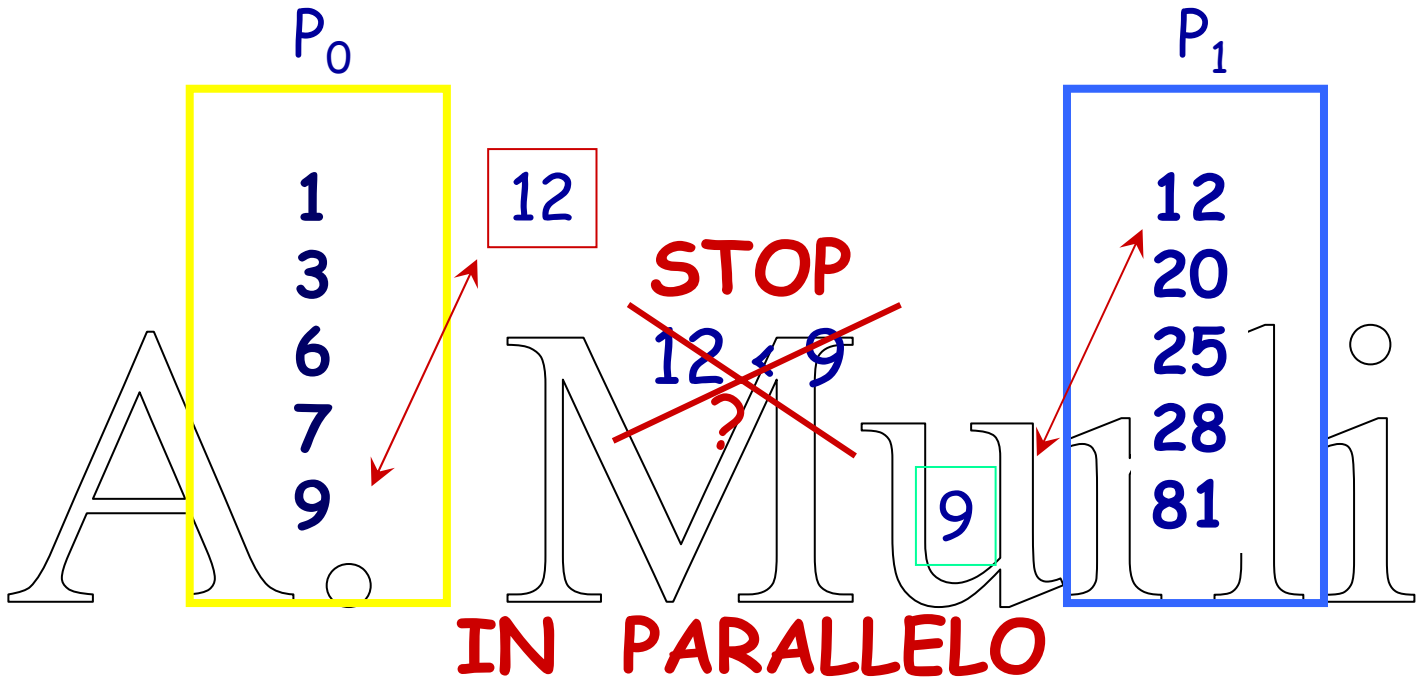
1) inserisce il numero ricevuto
2) Elimina il numero inviato
3) riordina il vettore

III PASSO: step 3.0 → COMUNICAZIONE



Step 1.0: P_0 invia a P_1 il proprio max = 9,
mentre P_1 invia a P_0 il proprio min = 12

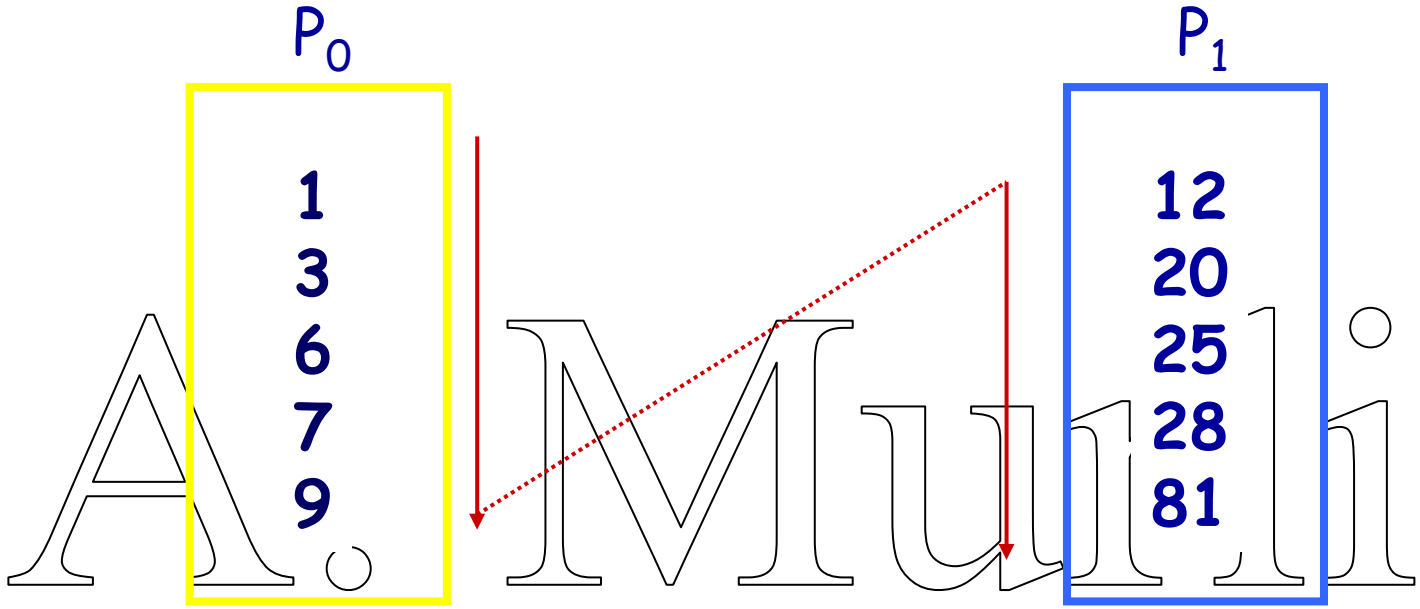
III PASSO: step 3.1 → CONFRONTO



Step 1.1: P_0 confronta il numero 12 con 9: non è minore e termina l' esecuzione.

P_1 confronta il numero 9 con 12: non è maggiore e termina l'esecuzione

PGBS:Ordinamento parallelo di un vettore



I 5 numeri più **piccoli** sono in P_0
mentre i 5 più **grandi** sono in P_1

Le due sottoliste iniziali sono globalmente ordinate secondo l'ordine crescente dell'identificativo dei processi

Finora abbiamo risolto il problema dell'ordinamento nel caso p=2.

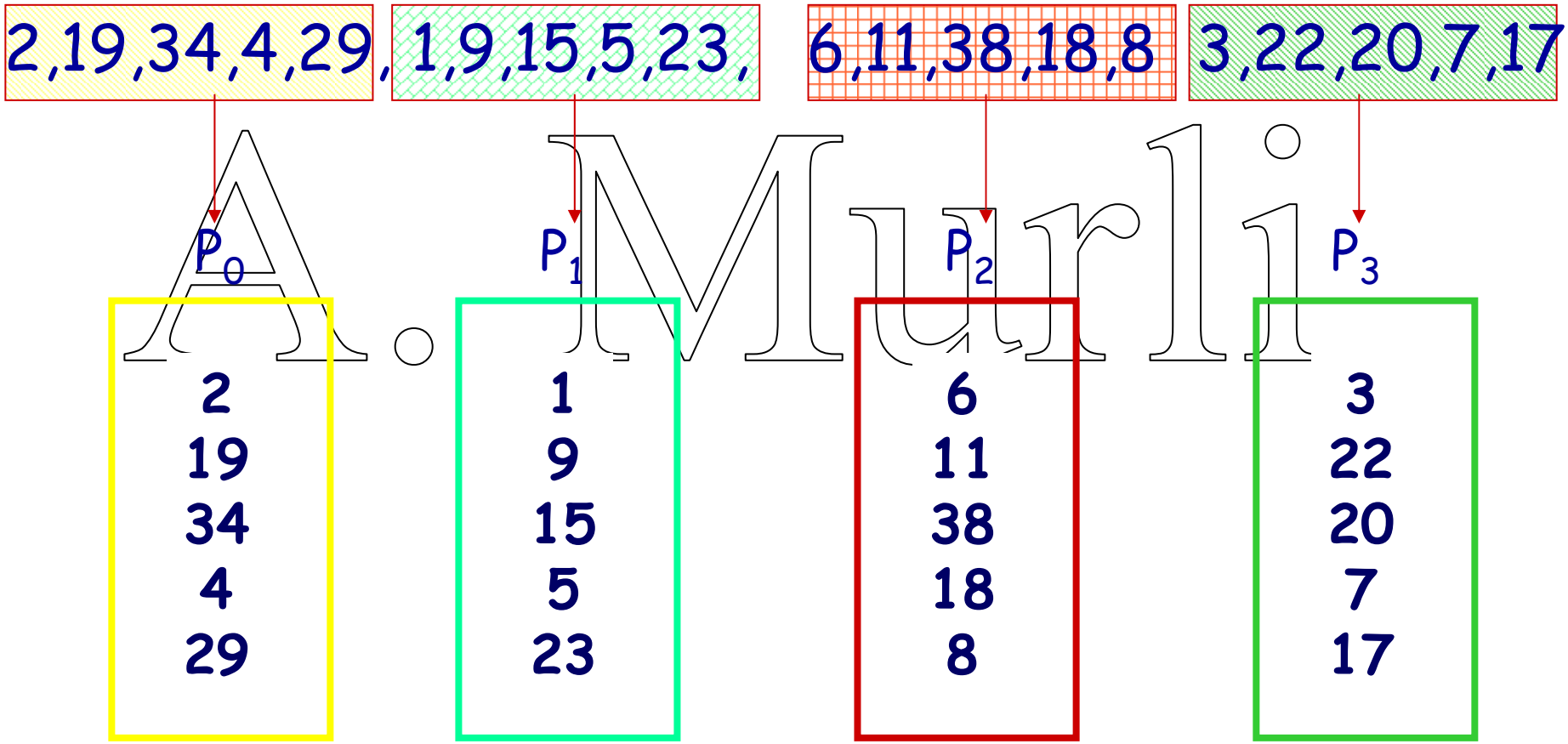
Come fare nel caso

p>2

?

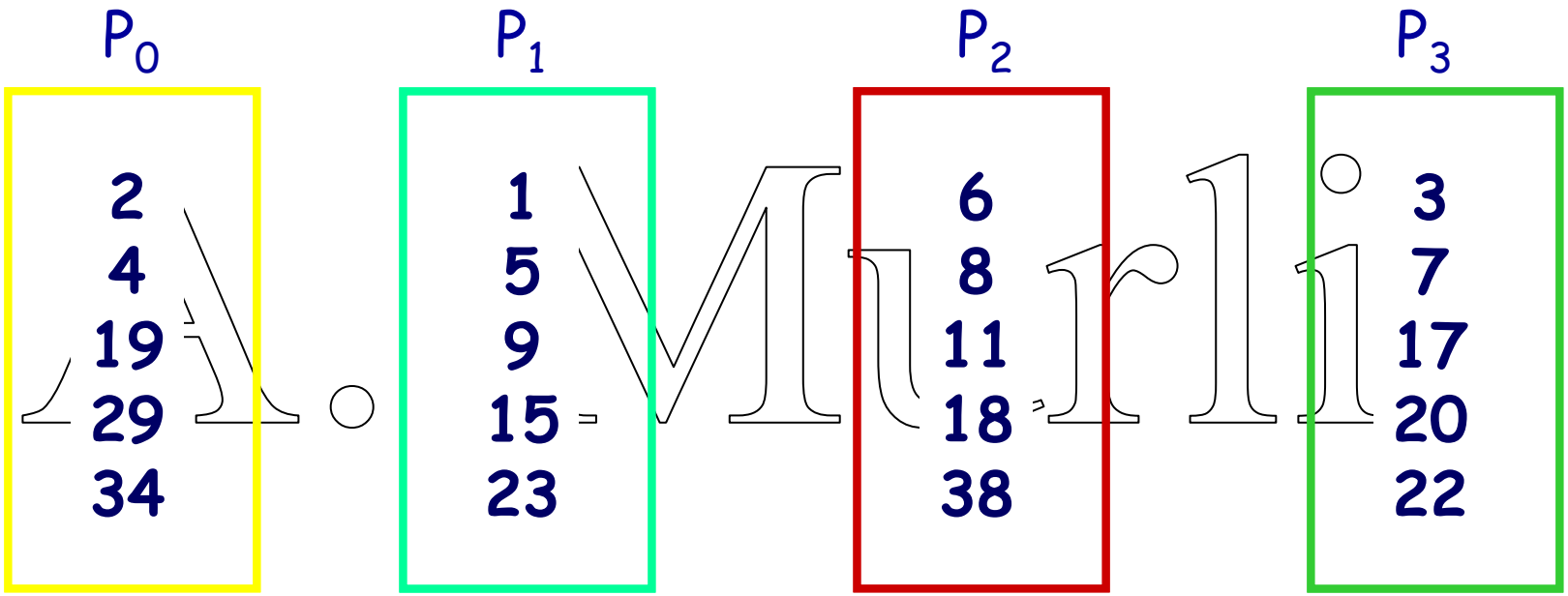
Esempio: p=4, n=20

Passo 1: distribuzione dati



Esempio: p=4, n=20

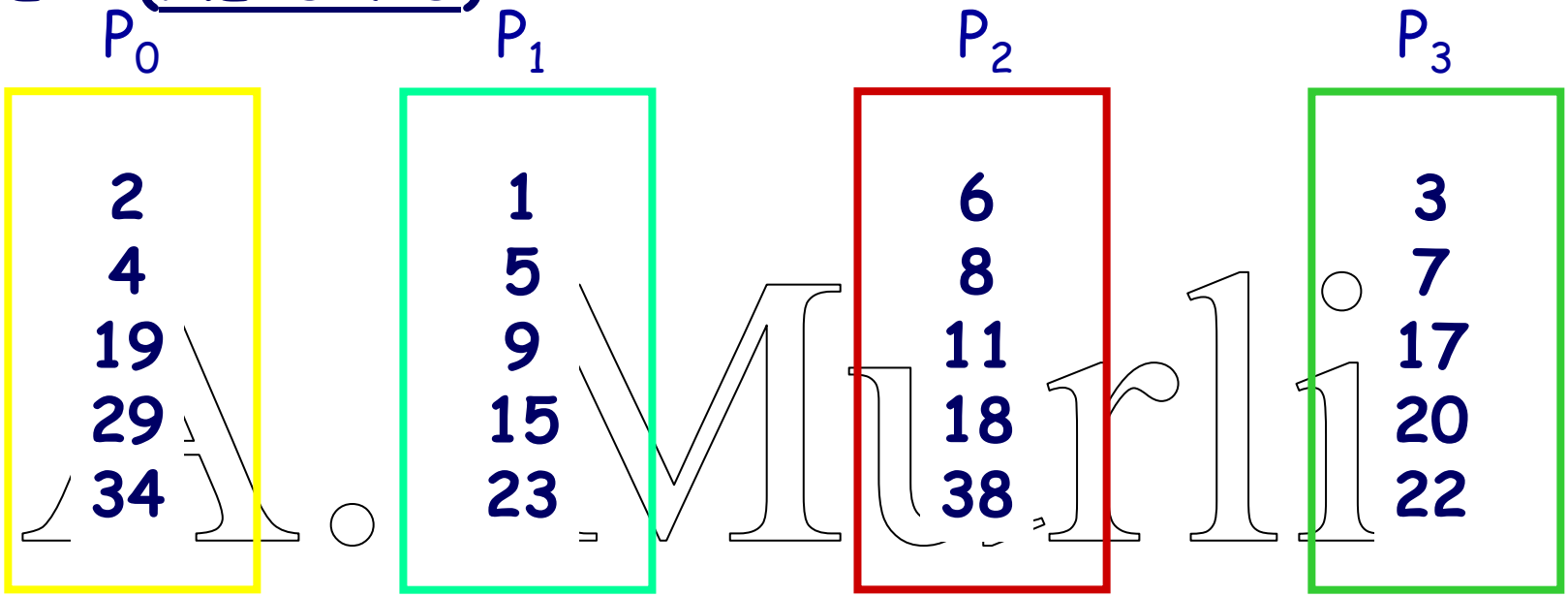
FASE 1 (SORTING LOCALE)



Ogni processore ordina in modo crescente
la propria sottolista

Esempio: p=4, n=20

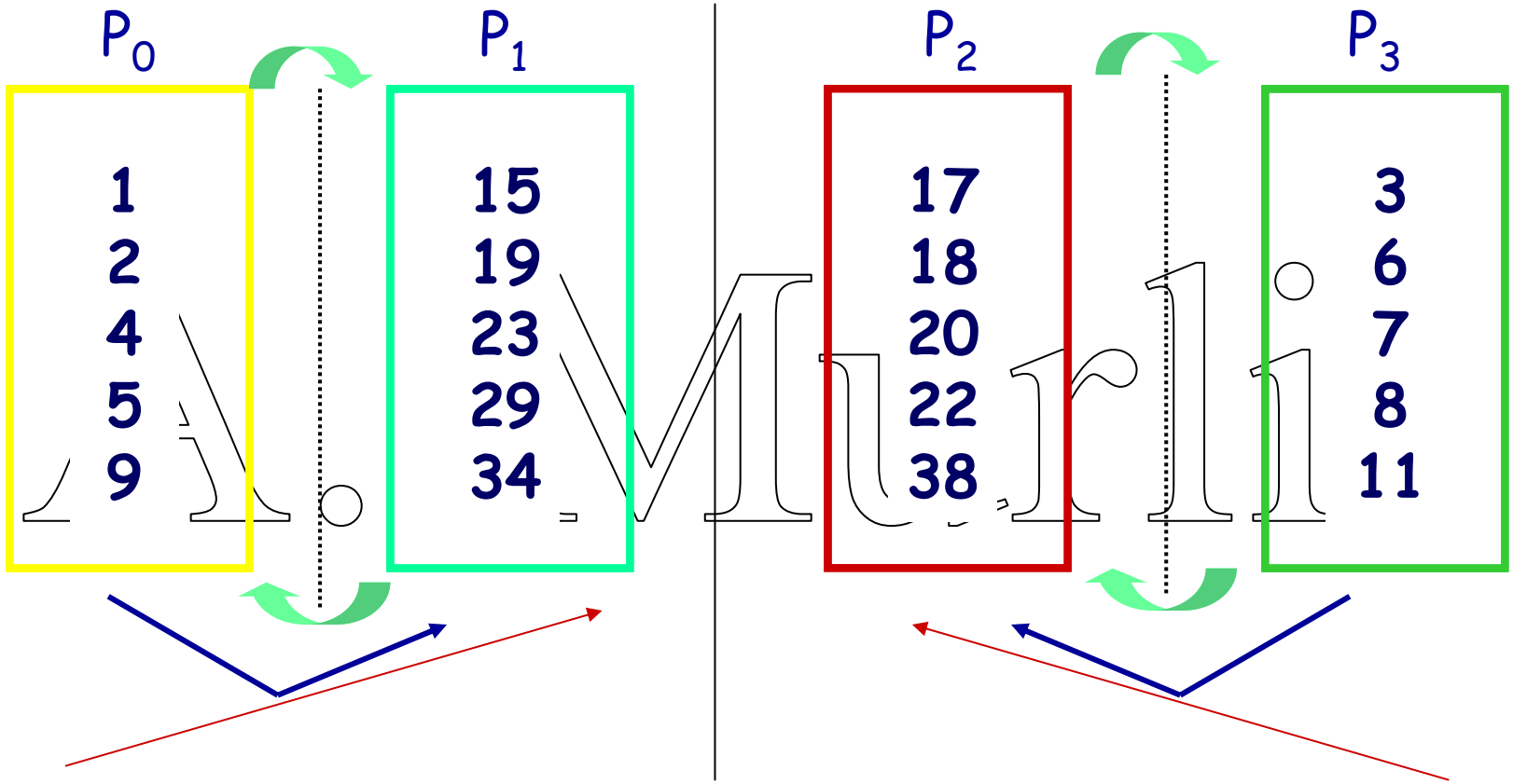
FASE 2 (MERGING)



Si applica l'algoritmo GBS a tutto il vettore

Esempio: p=4, n=20

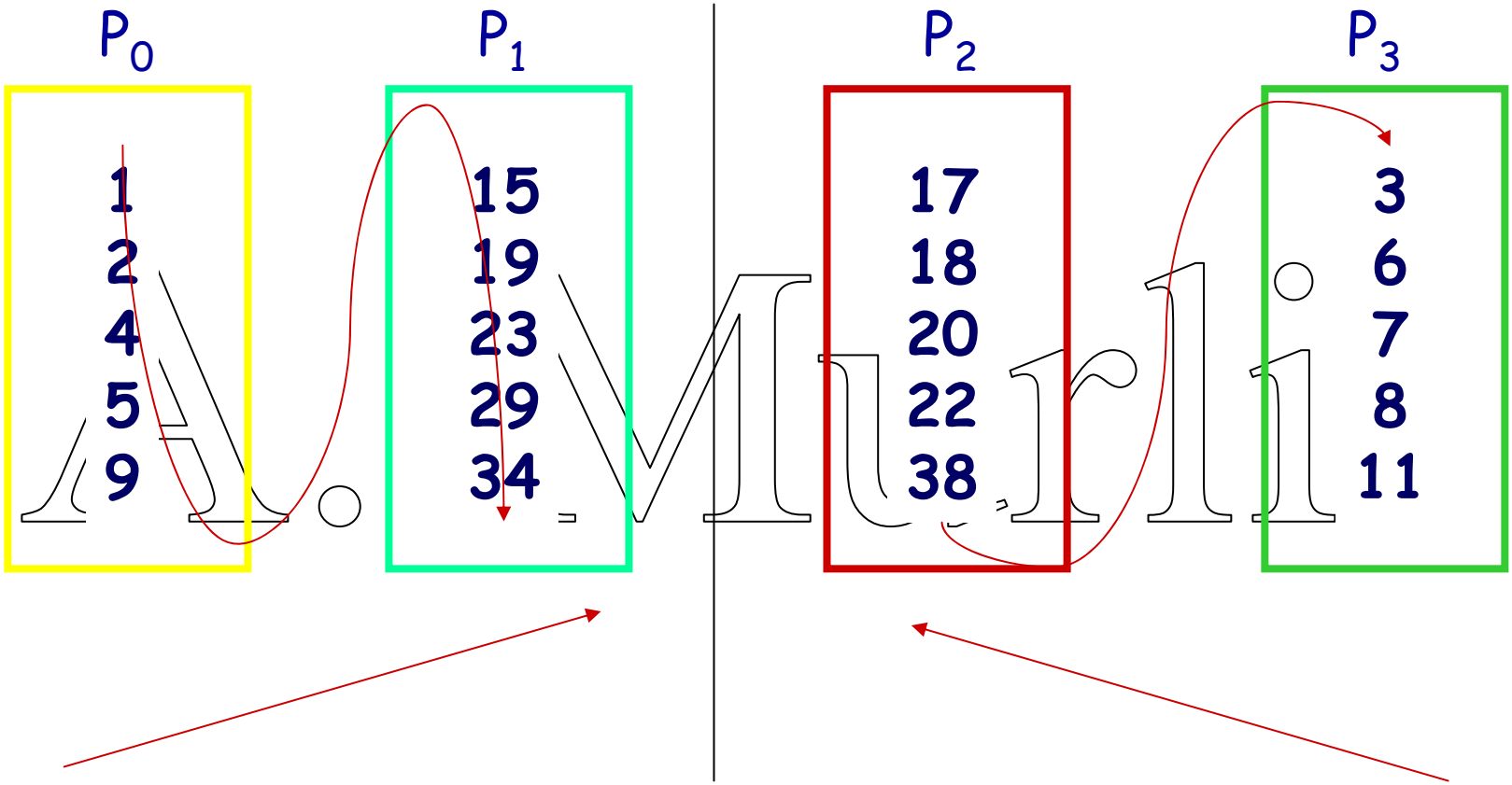
FASE 2 (MERGING)



Passo i=1 (j=1)

Esempio: p=4, n=20

FASE 2 (MERGING)

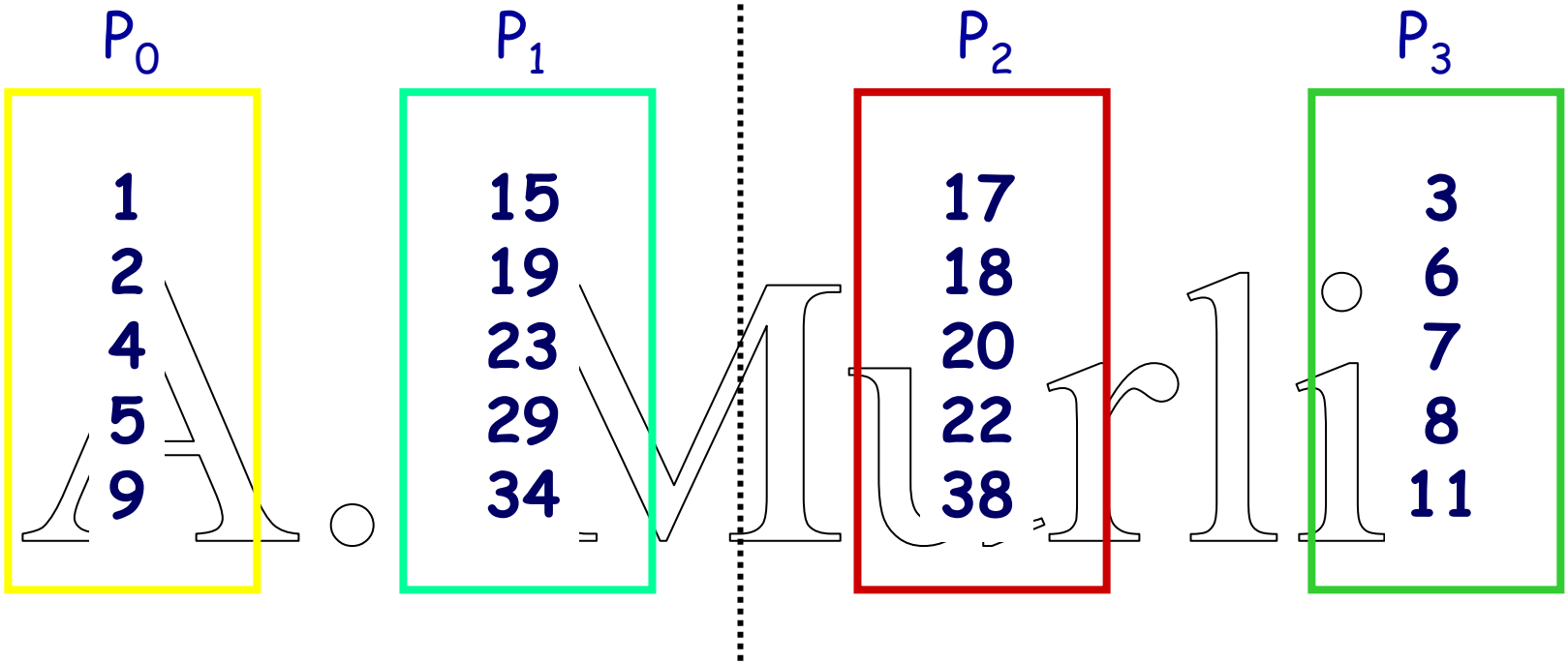


Fine passo i=1 e j=1

Il vettore è bitonico

Esempio: p=4, n=20

FASE 2 (MERGING)

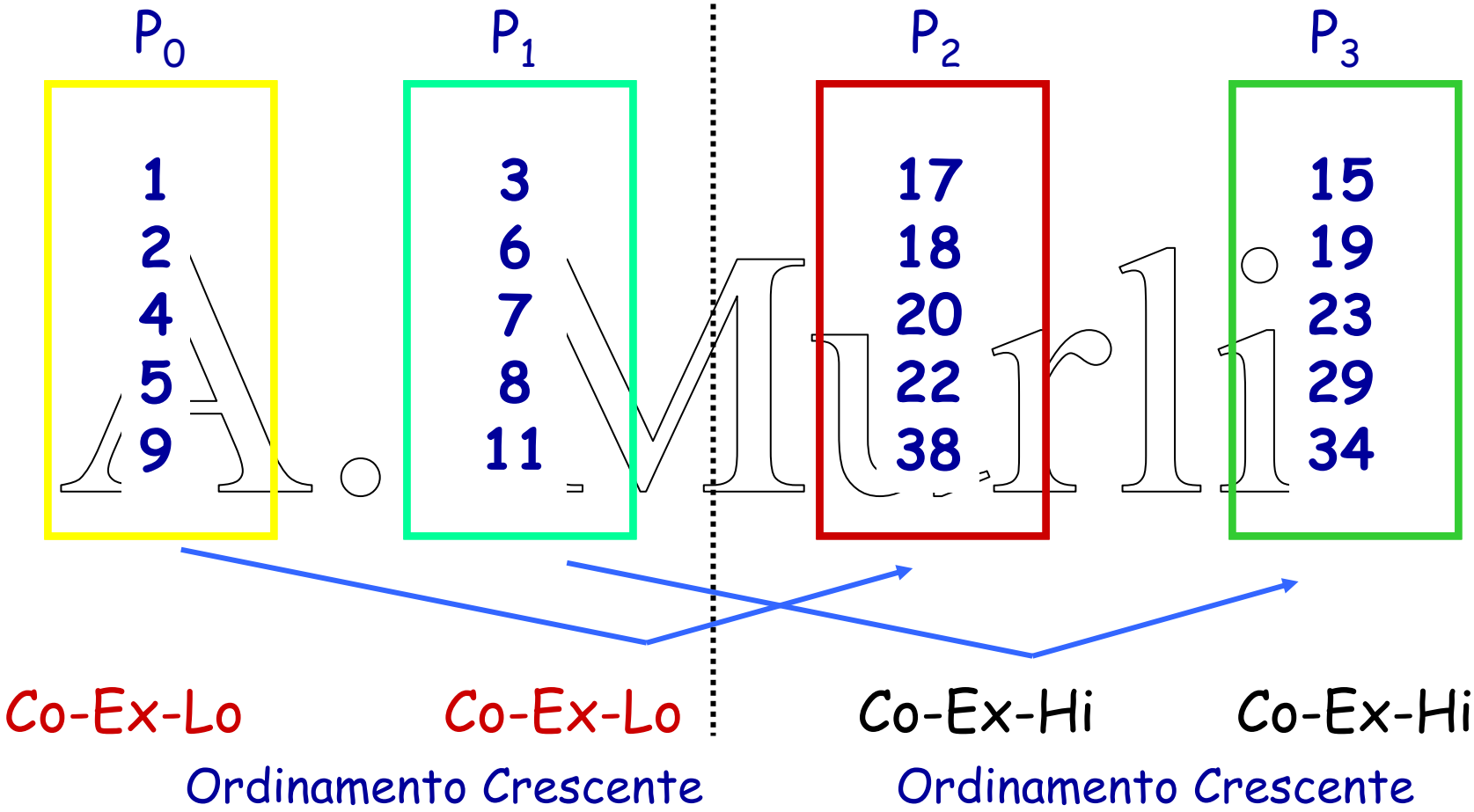


Passo i=2

ordiniamo globalmente il vettore

Esempio: p=4, n=20

FASE 2 (MERGING)

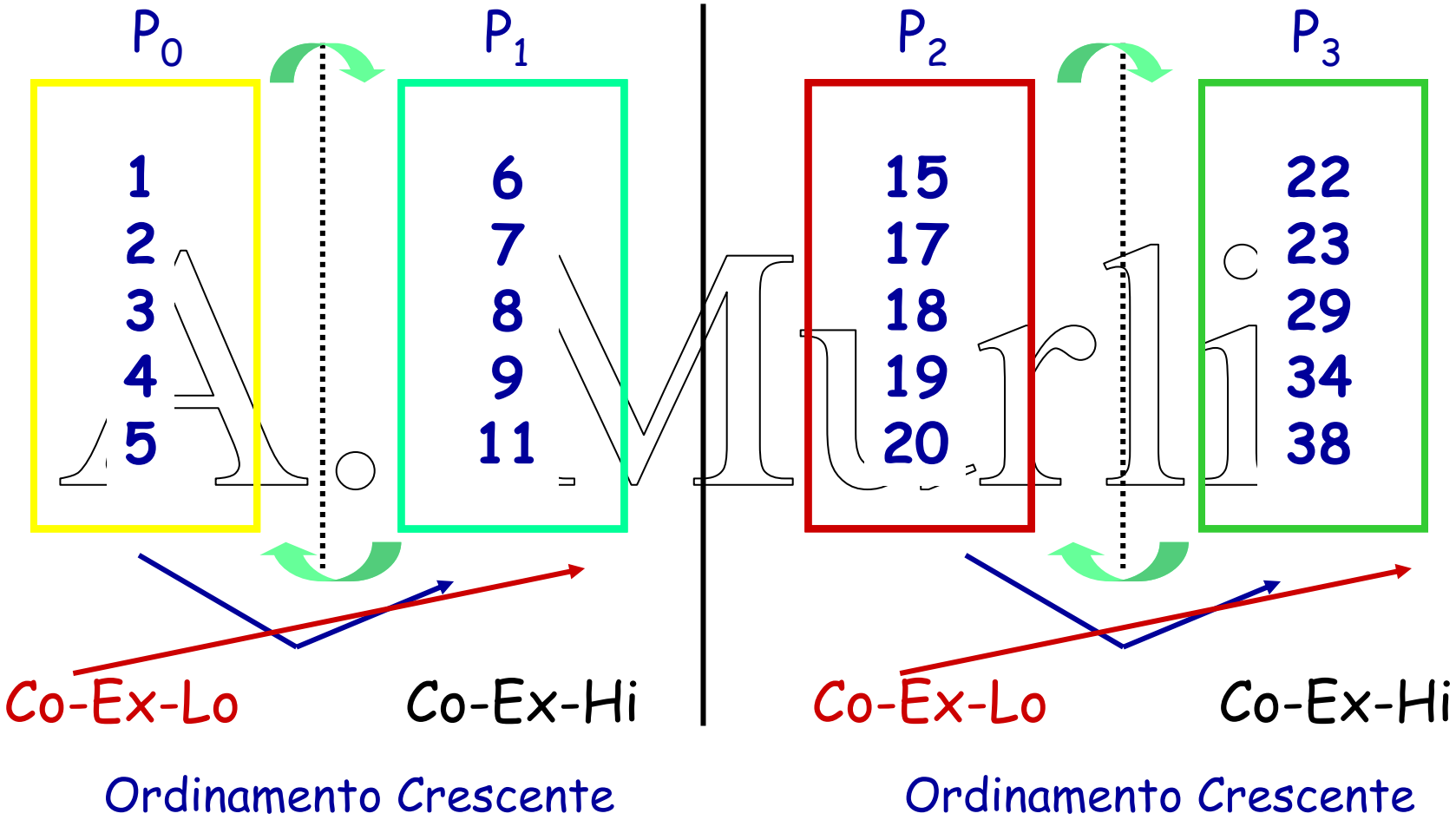


Passo i=2 (j=1)

Esempio: p=4, n=20

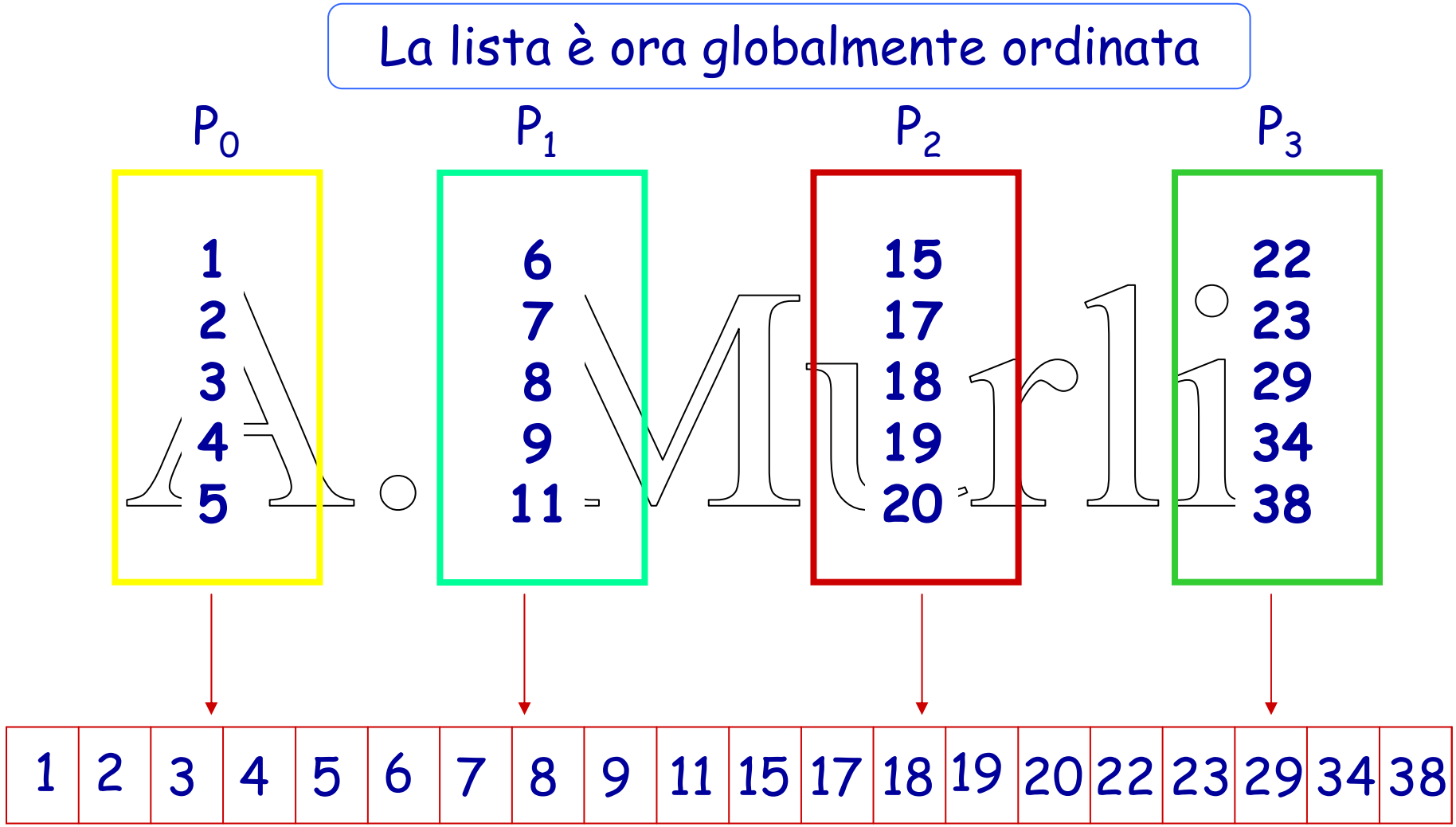
FASE 2 (MER

La lista è ora globalmente ordinata



Passo i=2 (j=2)

Esempio: p=4, n=20



A. Murli

FINE LEZIONE