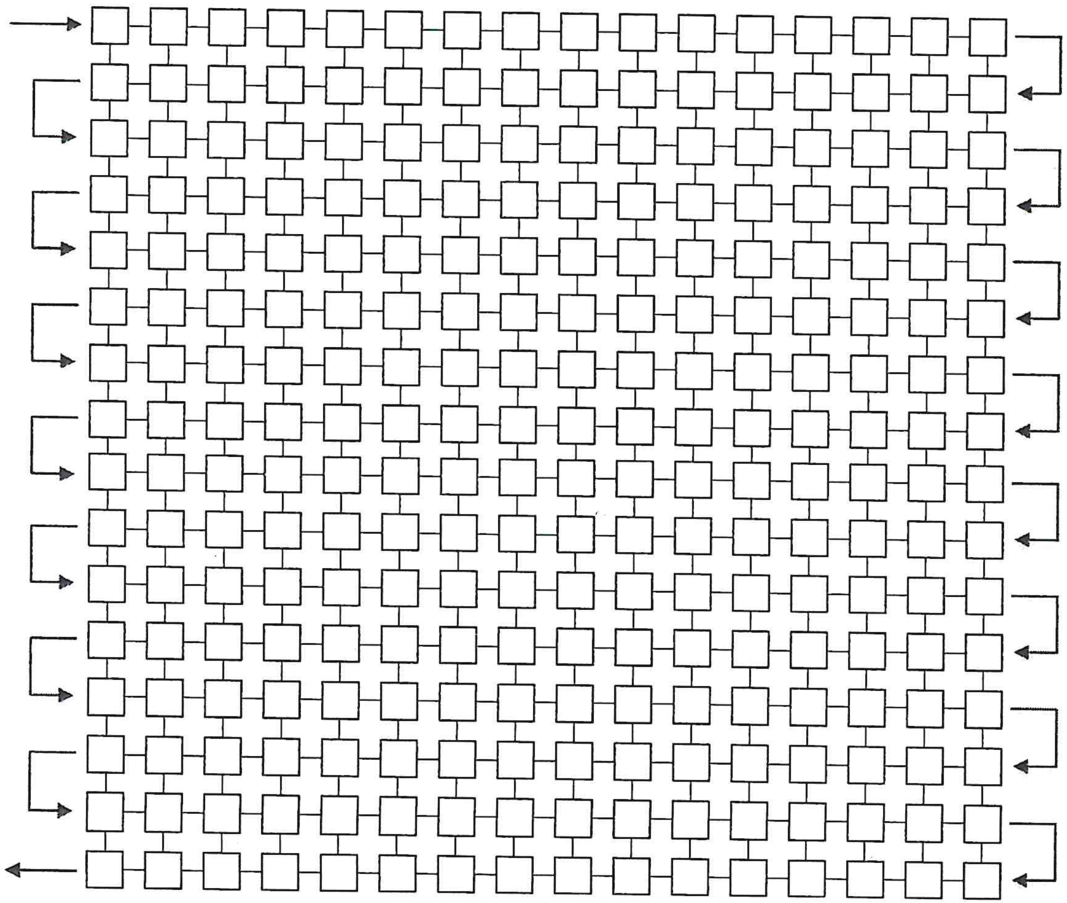
MESH n PROCESSORI: lato $\sqrt{n}$



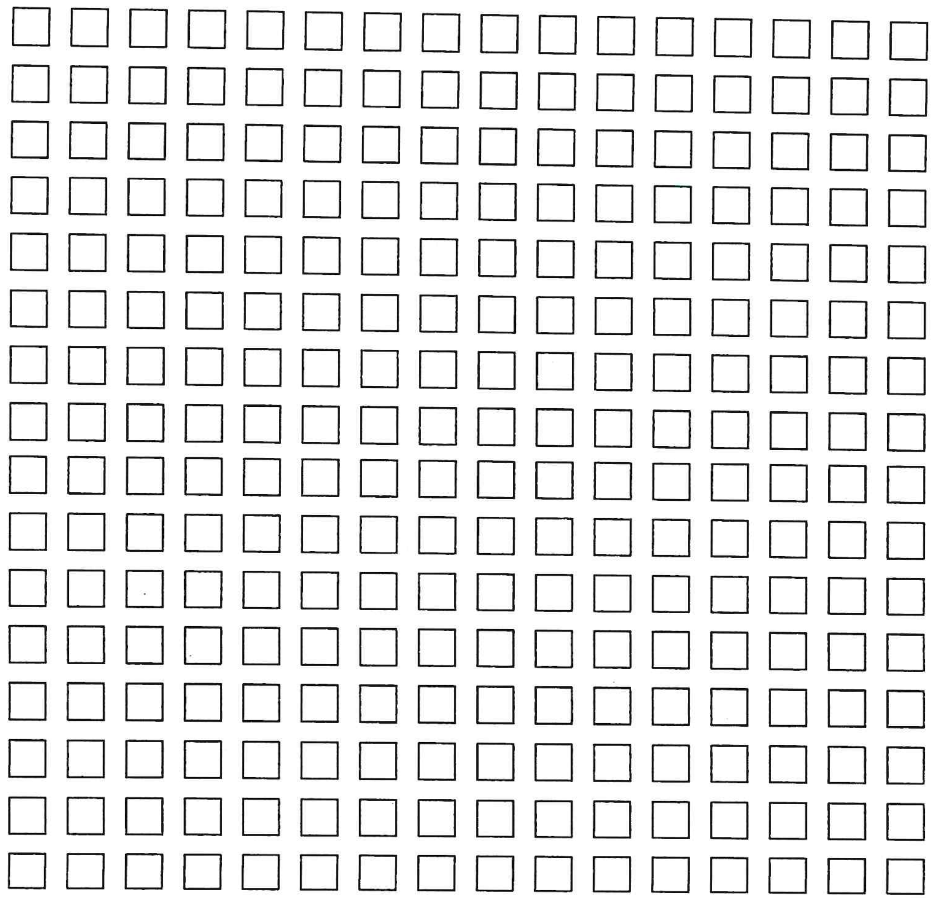
ORDINAMENTO

INPUT
n numeri distribuiti
uno per processore

OUTPUT
n numeri ordinati
secondo il percorso
a *serpente*

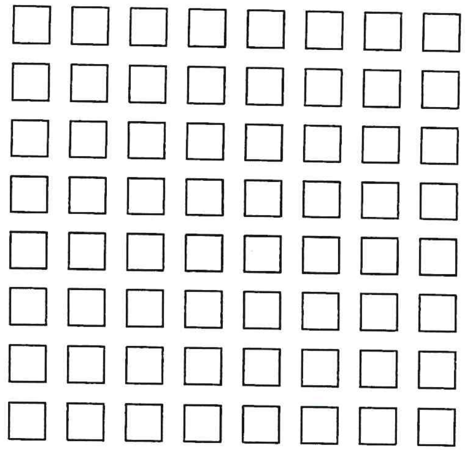
ORDINAMENTO LS3

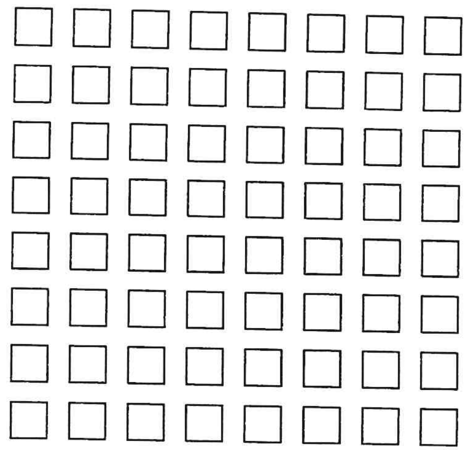
M



ORDINAMENTO LS3: DIVIDI

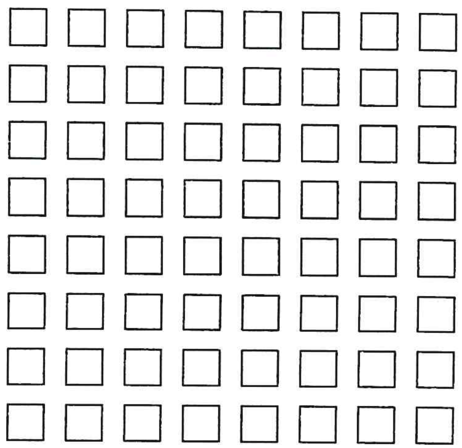
M_1

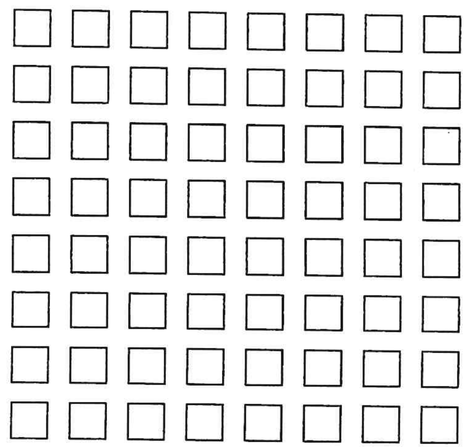




M_2

M_3

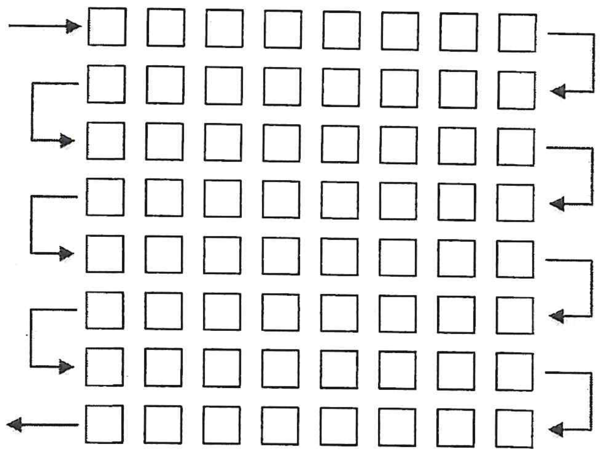


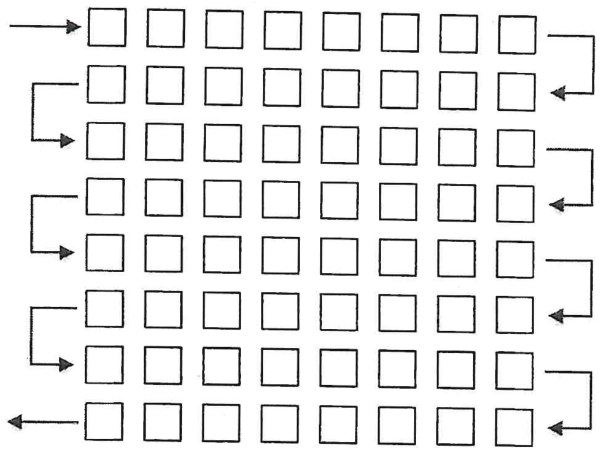


M_4

ORDINAMENTO LS3: ORDINA

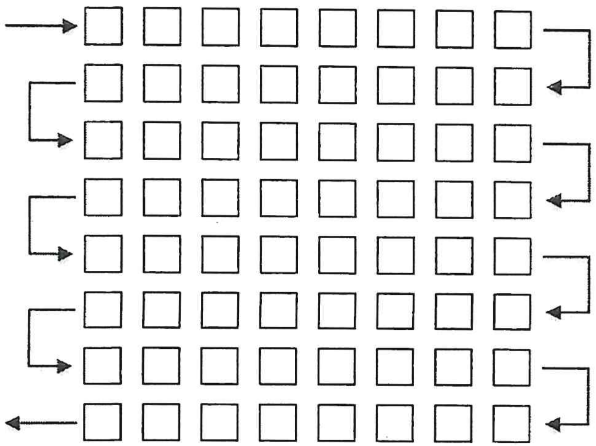
M_1

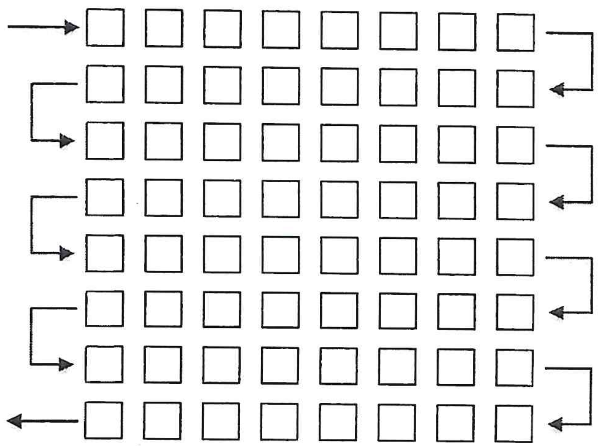




M_2

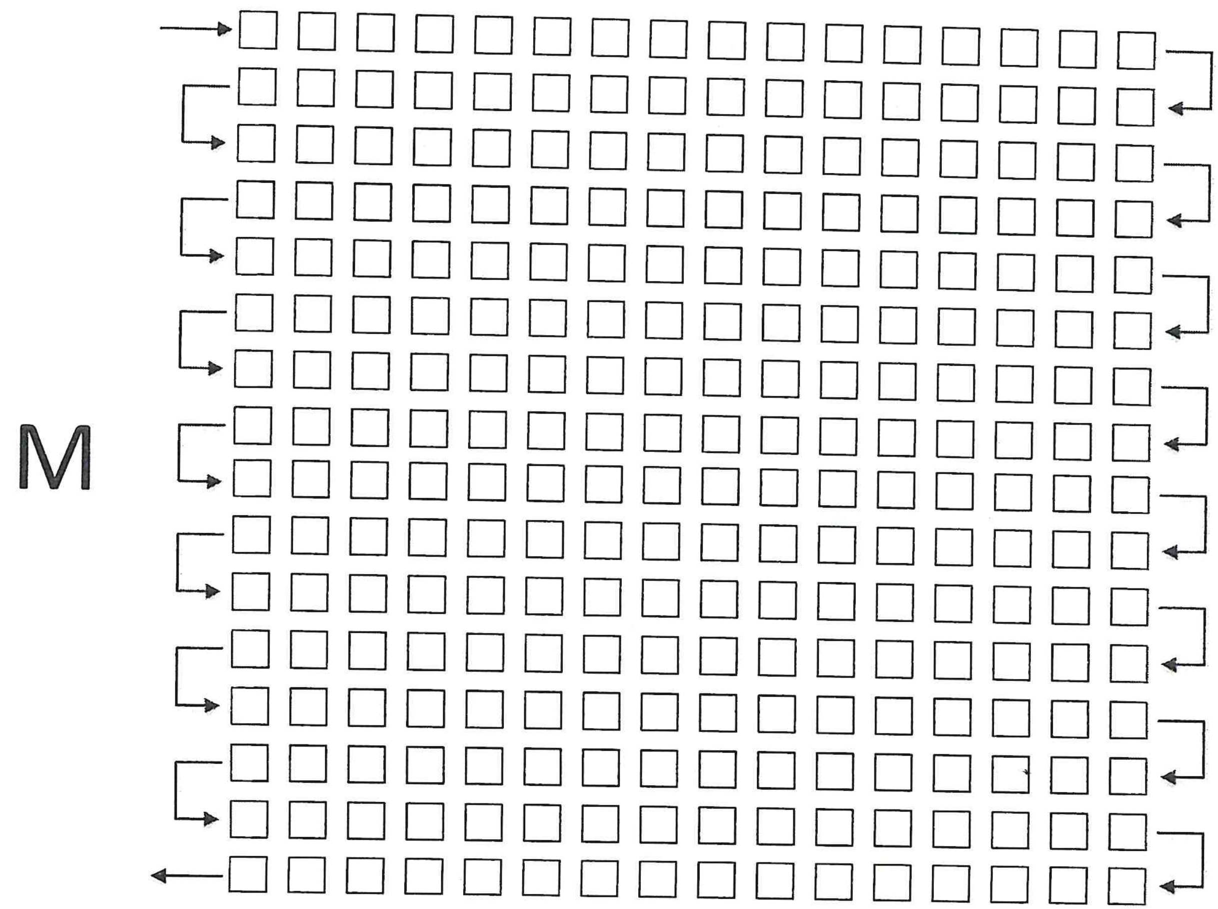
M_3





M_4

ORDINAMENTO LS3: FONDI



ORDINAMENTO LS3 parallelo

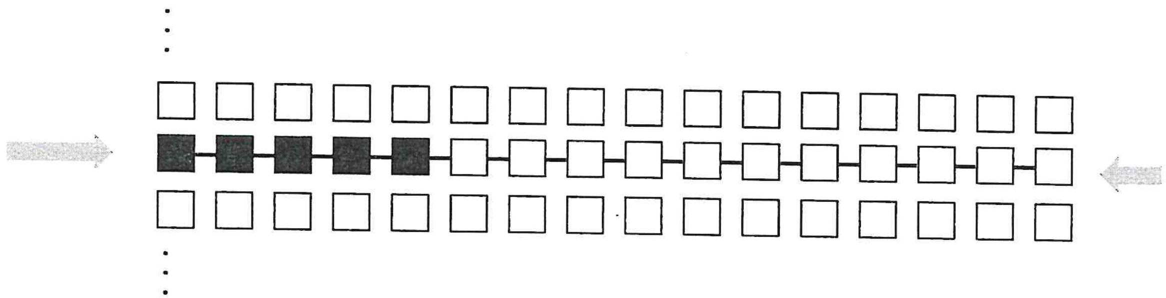
$$M = \begin{matrix} M_1 & M_2 \\ M_3 & M_4 \end{matrix}$$

procedure LS3sort(M)

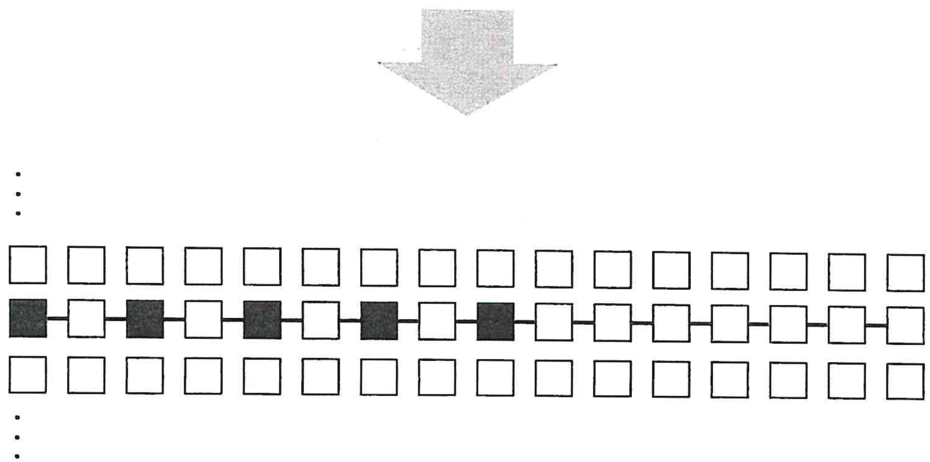
    if (|M| == 1)
        return M;
    else
        LS3sort(M_1); //in parallelo
        LS3sort(M_2);
        LS3sort(M_3);
        LS3sort(M_4);
        LS3merge(M_1 , M_2 , M_3 , M_4); ⇐ come costruirla?

ROUTINE PARALLELE PER LS3merge: SHUFFLE

SHUFFLE(i)



i-esima riga della mesh come array lineare di processori

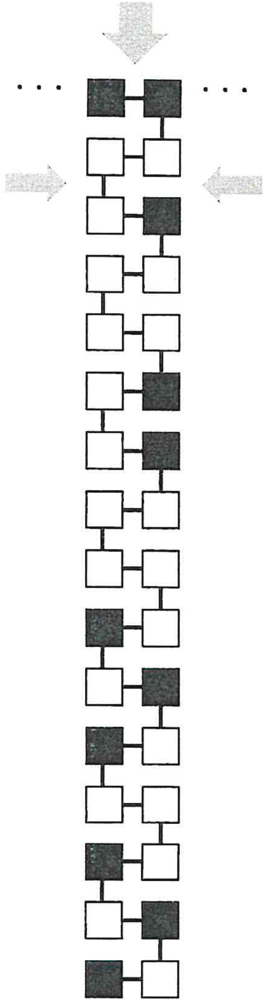


ROUTINE PARALLELE PER LS3merge: ODD-EVEN

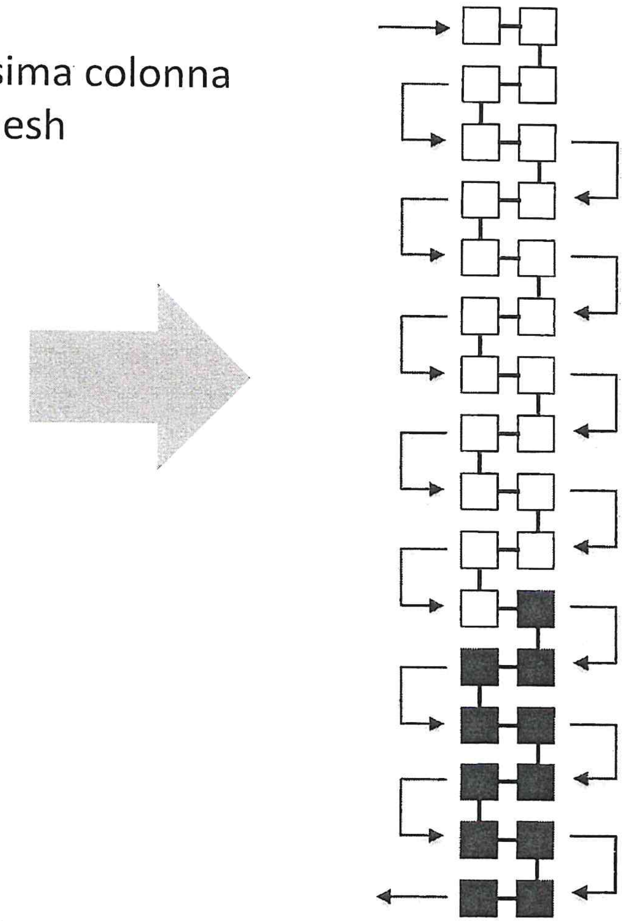
ODD-EVEN(i, i + 1)

i-esima colonna della mesh

(i+1)-esima colonna della mesh



array lineare dei processori sulla serpentina tra le due colonne



LS3merge

procedure LS3merge(M_1 , M_2 , M_3 , M_4) $M = \begin{matrix} M_1 & M_2 \\ M_3 & M_4 \end{matrix}$

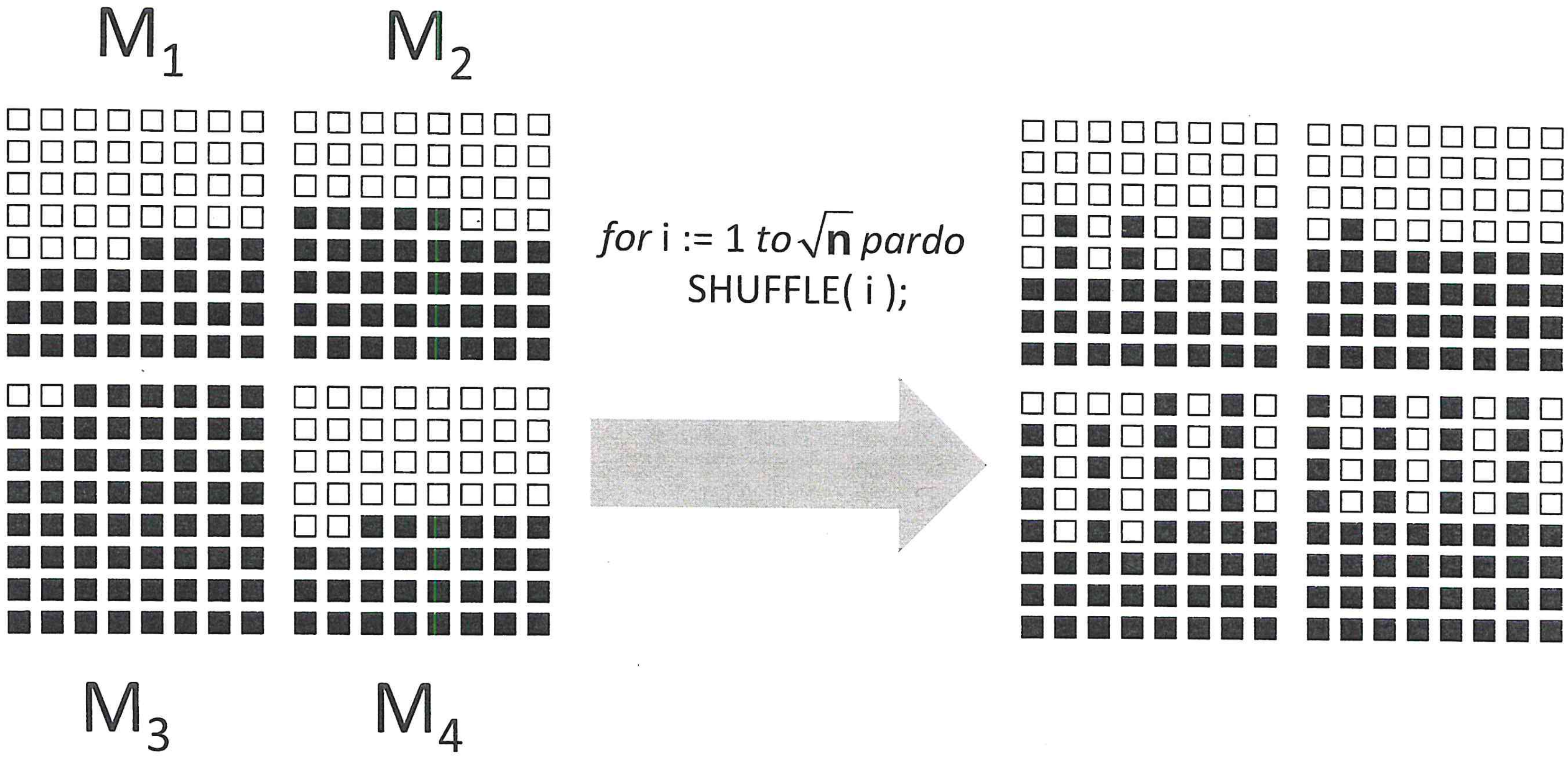
ordinate

for i := 1 *to* $\sqrt{\mathbf{n}}$ *pardo*
SHUFFLE(i);

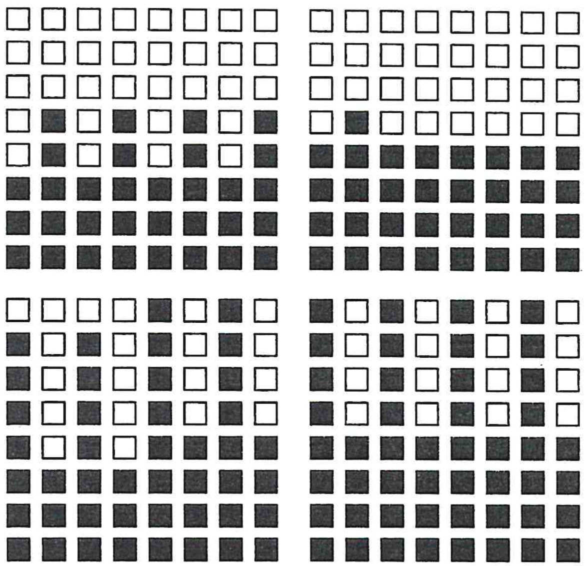
for i := 1 *to* $\sqrt{\mathbf{n}}/2$ *pardo*
ODD-EVEN(2i - 1, 2i);

esegui i primi $2\sqrt{\mathbf{n}}$ passi di ODD-EVEN
sull'intera mesh (a serpente);

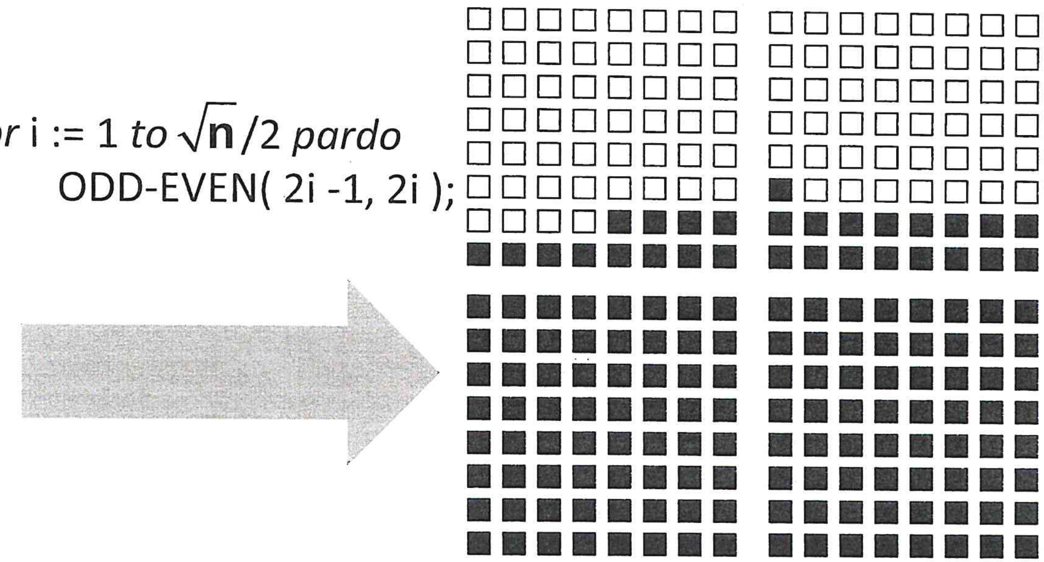
LS3merge(M_1 , M_2 , M_3 , M_4): SHUFFLE TIME



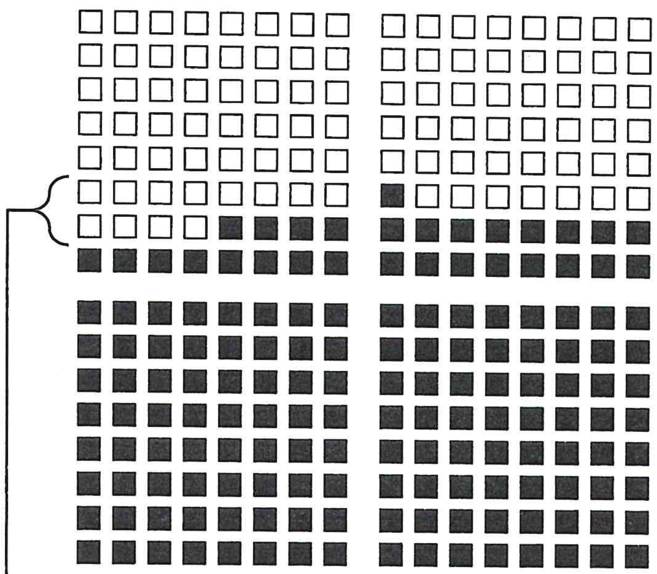
LS3merge(M_1 , M_2 , M_3 , M_4): ODD-EVEN TIME



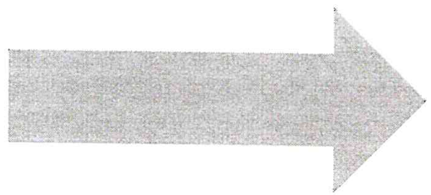
for i := 1 *to* $\sqrt{\mathbf{n}}/2$ *pardo*
ODD-EVEN(2i -1, 2i);

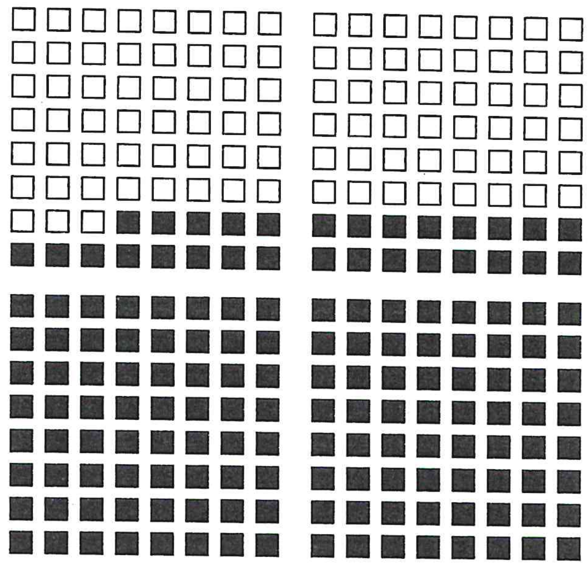


LS3merge(M_1 , M_2 , M_3 , M_4): FINAL ODD-EVEN



esegui i primi $2\sqrt{\mathbf{n}}$ passi di ODD-EVEN sull'intera mesh (a serpente);





la parte "disordinata" sta in una striscia di altezza 2

TEMPO PER LS3merge parallela

	Tempo
for i := 1 *to* $\sqrt{\mathbf{n}}$ *pardo* SHUFFLE(i);	$O(\sqrt{\mathbf{n}})$
for i := 1 *to* $\sqrt{\mathbf{n}}/2$ *pardo* ODD-EVEN(2i - 1, 2i);	$O(\sqrt{\mathbf{n}})$
esegui i primi $2\sqrt{\mathbf{n}}$ passi di ODD-EVEN sull'intera mesh (a serpente);	$O(\sqrt{\mathbf{n}})$

$$T_{merge}(n) = h\sqrt{\mathbf{n}}$$

RISOLVIAMO

$$T(n) = \begin{cases} 1 & \text{se } n=1 \\ T\left(\frac{n}{4}\right) + h\sqrt{n} & \text{altrimenti} \end{cases}$$

la ricorsione si ferma quando $n = 4$

$$T(n) = T\left(\frac{n}{4}\right) + h\sqrt{n} = T\left(\frac{n}{4^2}\right) + h\sqrt{\frac{n}{4}} + h\sqrt{n} =$$

$$= T\left(\frac{n}{4^3}\right) + h\sqrt{\frac{n}{4^2}} + h\sqrt{\frac{n}{4}} + h\sqrt{n} = \cdots = \sum_{i=0}^{\log_4 n - 1} h\sqrt{\frac{n}{4^i}} + 1 =$$

cambio della base di log.

$$= h\sqrt{n} \sum_{i=0}^{\log_4 n - 1} \sqrt{\frac{1}{4^i}} + 1 = h\sqrt{n} \sum_{i=0}^{\frac{\log n}{\log 4} - 1} \frac{1}{\underset{\downarrow\ (\sqrt{4})^i}{2^i}} + 1 = h\sqrt{n} \sum_{i=0}^{\frac{\log n}{2} - 1} \left(\frac{1}{2}\right)^i + 1$$

$$= h\sqrt{n} \left(\frac{\overbrace{1 - \left(\frac{1}{2}\right)^{\frac{\log n}{2}}}^{1 - x^{n+1}}}{\underbrace{\frac{1}{2}}_{1 - x}} \right) + 1 = 2h\sqrt{n}\left(1 - \frac{1}{\sqrt{n}}\right) + 1 = O\left(\sqrt{n}\right)$$

soluzione serie geom. $\sum_{i=0}^{n} x^i$

CONCLUDENDO: ORDINAMENTO LS3 parallelo

processori: $p(n) = n$ tempo: $T(n) = O(\sqrt{\mathbf{n}})$

efficienza: $\dfrac{n \log n}{n\sqrt{\mathbf{n}}} \longrightarrow 0$ *minimo teorico*

Possiamo migliorare l'efficienza riducendo i processori

con una versione del

BITONIC SORT su mesh

processori: $p(n) = O(\log^2 n)$

tempo: $T(n) = O(n / \log n)$

efficiente