

Indice

| | |
|---|-----------|
| Introduzione | 5 |
| Premessa | 7 |
| 1 L'Algoritmo di rotazione | 9 |
| 1.1 Funzionamento base dell'algoritmo | 9 |
| 1.2 Parallelizzazione dell'algoritmo | 10 |
| 2 I Test | 13 |
| 2.1 Considerazioni sui test | 14 |
| 2.2 MFLOPS | 15 |
| A I risultati | 17 |
| B Il Sorgente | 27 |
| Bibliografia | 40 |

Introduzione

Nel seguente progetto viene presentato un algoritmo di rotazione parallela per immagini PCX true color con una profondità di colore di 24 bit per pixel. Vengono quindi utilizzati 8 bit per ognuno dei 3 canali di colore RGB (Red, Green, Blue). La nostra scelta è caduta in questo progetto in quanto volevamo trasportare le nostre esperienze passate sugli argomenti qui trattati in un ambiente multiprocessore e allo stesso tempo evitare di ricalcare argomenti già fortemente trattati da altri. La scrittura del codice relativo alla rotazione ha richiesto dapprima la realizzazione degli algoritmi di Encoding/Decoding RLE (Run Length Encoding) per il formato PCX della Zsoft in alta risoluzione. La lettura dell'header dell'immagine ha richiesto per i tipi di dato superiori al byte la conversione ENDIAN per poter essere utilizzati e letti correttamente nella macchina parallela. L'algoritmo di rotazione è funzionante con immagini PCX 24bpp di qualsiasi dimensione $width * height$. Il programma prevede inoltre il passaggio attraverso la riga di comando dei seguenti parametri:

- Nome file sorgente PCX;
- Nome file destinazione PCX ;
- Angolo di rotazione normalizzato da 0 a 255 (in modo da usare per esso un solo byte) ;
- Valori R G B appartenenti all'intervallo $[0, 255]$, separati da uno spazio per definire il colore di sfondo dell'immagine ruotata e salvata nel file di destinazione specificato, dopo l'Encoding RLE. Questi ultimi, se non specificati, sono di default a zero.

La fase di scrittura e sviluppo del codice è stata realizzata su piattaforma Linux Debian mentre le prove e i test sono stati sempre eseguiti sulla macchina IBM RS/6000 SP collegandoci ad essa tramite ssh da casa o in laboratorio.

Il programma non evidenzia problemi di nessun tipo e termina sempre con successo la sua esecuzione in base alle direttive preimpostate nei file Job sottoposti allo scheduler per mezzo del sistema di gestione LoadLeveler. I test sono stati eseguiti mettendo in evidenza l'importanza del sottosistema di networking utilizzato in un ambiente multi-processore e mettendo inoltre in evidenza il comportamento dell'algoritmo, sottoposto ad immagini in ingresso di diverse dimensioni. Sono stati variati i seguenti parametri nelle diverse esecuzioni:

- Diversa grandezza dell'immagine PCX (800x600 - 1024x768 - 1191x893 - 2048x1536 - 4000x3000);
- Diversa rete di comunicazione (Switch ad alte prestazioni ed Ethernet);
- Diverso numero di Processori.

E' stato mantenuto fisso l'angolo di rotazione per tutte le prove e il colore di sfondo dietro all'immagine ruotata.

Premessa

Inizialmente si voleva realizzare, oltre la rotazione, anche l'encoding e il decoding di immagini PCX in parallelo ma non si è riusciti ad utilizzare le librerie MPI per la lettura scrittura parallela di file in quanto tale libreria non sembrava essere presente nel sistema; questo non ci ha permesso di valutare i pro e i contro di un loro possibile utilizzo ai fini del programma. Una loro realizzazione in parallelo utilizzando le classiche funzioni offerte dalla libreria `<stdio.h>` risultava apparentemente non appetibile e difficile da praticare. Per quanto riguarda il DECODING in entrambi i casi ci sarebbe stato il problema insito nella natura intrinsecamente seriale della decodifica RLE e all'impossibilità di dividere il file in modo esatto se non pre-processandolo in modo sequenziale. Anche se l'immagine fosse stata aperta da un solo processore e distribuita ai vari processori tramite adeguate funzioni MPI di comunicazione, quali `MPI_Scatter`, questo avrebbe sovraccaricato di molto, come si può ben capire, le comunicazioni del sistema oltre che essere inutile ai fini dell'utilizzo dell'algoritmo di rotazione. L'algoritmo infatti va a calcolare per ogni coordinata la coordinata del pixel che li deve essere ruotato senza necessitare dell'immagine. Ogni processo avrebbe dovuto poi richiedere agli altri processi i valori della triade RGB dei pixel che con la rotazione andavano a finire nei pixel dell'area di memoria di sua competenza comportando un grosso intasamento in comunicazioni, a sfavore anche della scalabilità. Per quanto riguarda la codifica questa avviene per scanline; di solito una scanline equivale alla larghezza (*width*) dell'immagine. Una sua implementazione sarebbe stata possibile nel caso in cui i processori avessero accesso all'immagine, cosa non richiesta dall'algoritmo parallelo di rotazione. Si è quindi optato per la realizzazione in parallelo del solo algoritmo di rotazione. La compressione

e la decompressione vengono quindi effettuate inizialmente dal solo processore ROOT mentre l'algoritmo parallelo di rotazione viene eseguito tra tutti i Job presenti i quali si suddividono virtualmente le aree dell'immagine da ruotare come vedremo di seguito.

Capitolo 1

L'Algoritmo di rotazione

1.1 Funzionamento base dell'algoritmo

L'algoritmo di rotazione si serve di due buffer di memoria, il primo contiene l'immagine da ruotare e ha una dimensione pari a $width*3*height$ mentre il secondo di destinazione, con la stessa dimensione, conterrà l'immagine ruotata. Ricordiamo che la costante 3 è dovuta al fatto che ogni pixel è formato da una triade RGB che occupa 3 byte consecutivi. L'immagine viene ruotata attorno al pixel di coordinate $(width/2, height/2)$, il quale individua il centro dell'immagine, applicando semplicemente una rotazione alle coordinate di ciascun Pixel. Per ogni pixel di coordinate (i, j) con:

```
i=[-width/2..width/2)
j=[-height/2...height/2)
```

si va a determinare quale pixel di coordinate (i', j') dell'immagine sorgente verrà ruotato alla posizione (i, j) dell'immagine di destinazione. Se i valori di (i', j') risultano fuori dallo specchio dell'immagine allora nel pixel (i, j) corrente verrà impostato il valore di RGB di default per lo sfondo. I valori di (i', j') vengono calcolati in base alle seguenti formule di rotazione

```
i'=(i*cos_rad(a)+j*sin_rad(a))>>10
j'=(i*sin_rad(a)-j*cos_rad(a))>>10.
```


dove a è l'angolo di rotazione appartenenti all'intervallo $[0, 255]$. Le funzioni `sin_rad` e `cos_rad` moltiplicano tale valore per $2\pi/256$ e ne calcolano il *cos* e il *sin*. Il motivo di tale scelta è dovuto al fatto di poter così usare una variabile `char` di 1 byte per esprimere l'angolo di rotazione. Da notare che aumentando all'interno della funzione `cos_rad` e `sin_rad` la costante moltiplicativa 1024 e modificando nelle funzioni sopra il valore dello *shift* $\gg 10$ si ottiene un rimpicciolimento dell'immagine ruotata.

1.2 Parallelizzazione dell'algoritmo

L'idea che abbiamo seguito per parallelizzare questo algoritmo è molto semplice. I processori non hanno bisogno dell'immagine o del suo pre-caricamento nel buffer sorgente. Le uniche informazioni di cui ciascun processore necessita per poterla ruotare sono:

- `sin_rad(a)`
- `cos_rad(a)`
- `Width, Height` dell'immagine

Queste informazioni vengono inviate in Broadcast dal task 0 il quale calcola i valori per `cos_rad` e `sin_rad` e legge l'header dell'immagine dal quale ricava *width* e *height*. Ogni Task attivo calcola invece numero delle righe dell'immagine che deve ruotare per mezzo dell'espressione (approssimata per difetto grazie alla funzione C `ceil()`):

```
numrighe=height/ranksize,
```

dove `ranksize` è il numero dei task attivi. In base al rank del task, che va da 0 a $ranksize - 1$, ogni processore riesce quindi a sapere da quale riga (*start*) partire per ruotare le sue *numrighe* dell'immagine. Se l'immagine ha un'altezza (*height*) non divisibile perfettamente per il numero di processori avremmo un resto che è minore del numero dei processori. Queste righe di resto vengono ruotate in blocco dal task 0. E' comunque possibile ottimizzare ulteriormente suddividendo questo numero irrisorio di righe fra i processori attivi. Ciascun processore ruoterà in seguito un numero uguale di

righe dell'immagine . Ognuno istanzia un vettore di grandezza $width * numrighe$ nel quale memorizzerà per ogni locazione, in base all'algoritmo di rotazione sopra descritto, il solo offset del pixel (coincidente con la prima componente R di rosso) che dovrà essere ruotato in quella coordinata dell'immagine. Terminata la fase appena descritta tutti i processori si sincronizzano ed inviano quanto calcolato al processo 0. E' stata quindi utilizzata la funzione della libreria MPI `MPI_Gather`. Questa funzione trasmette da tutti i processi appartenenti ad un comunicatore ad un singolo processo denominato ricevente, nel nostro caso il processo 0.

```
MPI_GATHER(nome-in, quanti, tipo-in, nome-out, quanti, tipo-out, ricevente,
comunicatore, ierr)
```

dove:

- *nome-in* si riferisce all'indirizzo iniziale di memoria di una serie di dati contigui (al limite solo uno) che devono essere spediti.
- *quanti* il quantitativo di questi dati (è una variabile intera)
- *tipo-in* il loro tipo
- *nome-out* si riferisce all'indirizzo iniziale di una zona di memoria da assegnare in modo contiguo alla serie di dati che il ricevente stà ricevendo. Tra queste due zone di memoria, e cioè tra quella occupata dai dati spediti e quella che vanno ad occupare i dati ricevuti non ci devono essere sovrapposizioni, ma questo vale chiaramente solo per il processo ricevente.
- *quanti* ancora il quantitativo di questi dati (è una variabile intera)
- *tipo-out* il tipo dei dati ricevuti
- *ricevente* si riferisce all'intero che caratterizza il processo che riceve il messaggio
- *comunicatore* identifica il gruppo di processi che spedisce il messaggio al ricevente
- *ierr* intero di controllo della funzione

L'uso di questa famiglia di funzioni toglie al programmatore il compito di gestire manualmente (con la possibilità di introdurre errori di sincronizzazione) la comunicazione tra i singoli processori. Una volta ricomposto l'array *i_arr* con le rotazioni da effettuare, il processo 0 copierà in ogni locazione del buffer dell'immagine di destinazione i valori RGB dell'immagine sorgente in base agli offset contenuti nell'array. I valori di G e B si ottengono sommando rispettivamente 1 e 2 al valore dell'offset per R calcolato dai vari processori.

Capitolo 2

I Test

Presentiamo di seguito le tabelle dei test eseguiti sul calcolatore parallelo. Ogni modalità di esecuzione è stata ripetuta 5 volte. I tempi totali sono comprensivi dei tempi di codifica e decodifica RLE oltre che della stessa rotazione. Le modalità di esecuzione per ognuna delle diverse dimensioni dell'immagine sono le seguenti:

1. Switch

- 1 task - 1 nodi Seriale : eseguito su un singolo processore
- 4 task - 1 nodi con shared memory
- 4 task - 1 nodi senza shared memory
- 4 task - 4 nodi - uso un solo processore di 4 nodi diversi
- 10 task - 10 nodi
- 16 task - 16 nodi
- 24 task - 24 nodi
- 24 task - 6 nodi con shared memory

2. Ethernet

- 4 task - 4 nodi
- 10 task - 10 nodi

- 16 task - 16 nodi
- 24 task - 24 nodi
- 24 task - 6 nodi con shared memory

Le tabelle contenenti tutti i risultati ottenuti sono contenute all'interno dell'Appendice A. Come si può notare, ciascuna misurazione è stata effettuata cinque volte, e se ne è considerata la media aritmetica.

2.1 Considerazioni sui test

Guardando le tabelle si può notare l'impatto determinante della rete sul tempo di esecuzione del programma. Se prendiamo in considerazione la rete ethernet a 10 Mbit con 24 processori, per immagini 800x600 possiamo notare in tutte le prove un incremento rispetto alla rete switch entro l'intervallo dal 49% all'82% del tempo di comunicazione e dal 63% al 93% per immagini 2048x1536. All'aumentare della grandezza dell'immagine e del numero di processori ovviamente aumentano anche i tempi di comunicazione. Questo comporta un aumento del tempo di esecuzione totale che per immagini 2048x1536 con 24 processori su 6 nodi arriva ad essere di quasi 20 volte superiore rispetto a quello riscontrato nella rete switch. Nelle varie prove effettuate possiamo notare che nel totale i guadagni in termini di tempo sono molto piccoli, ricordiamo che i tempi totali comprendono anche ENCODING e DECODING. Per quanto riguarda la rotazione si può comunque notare un guadagno nel tempo di esecuzione, in particolare al crescere della dimensione dell'immagine. Osserviamo infatti i tempi rilevati nelle prove extra effettuate su un'immagine di dimensione 4000x3000 che decodificata occupa all'incirca 92 Mbyte. In tali prove, notiamo rispetto all'uso di un solo processore, un aumento dei tempi di comunicazione per 24 processori su 6 NODI di circa l'1%. Per quanto riguarda i tempi di rotazione possiamo invece osservare dei guadagni più apprezzabili di quasi 1,5 secondi rispetto all'uso di un solo processore. I tempi totali sono quelli mostrati nella relativa tabella nell'Appendice A. Possono essere prese in considerazione delle varianti per l'algoritmo attuale il quale prevede l'accesso al buffer dell'immagine da parte di un singolo processore, per copiare i valori dei pixel nella loro giusta locazione. Tali

varianti, come è stato già fatto notare in precedenza, prevedono un maggior carico in comunicazioni. I calcoli per la rotazione eseguiti in parallelo non richiedono un grosso quantitativo di tempo e il guadagno rimane quindi molto basso. Inoltre, come già osservato, anche nel caso in cui una variante prevedesse la decodifica RLE di diverse parti dell'immagine in maniera indipendente fra i vari processori ci sarebbe il problema per questi di richiedere agli altri i valori di tutte le triadi RGB che andrebbero a finire dopo la rotazione nella propria area di competenza. Anche questo aumenterebbe non di poco le comunicazioni fra i processori in modo proporzionale alla dimensione dell'immagine stessa. Nel caso di switch ad alte prestazioni notiamo invece che l'overhead della rete tocca un massimo di 1.5% lasciando intendere i vantaggi del calcolo parallelo. Questa percentuale subisce nella versione attuale solo lievi variazioni in eccesso o in difetto all'aumentare o al diminuire della dimensione dell'immagine. A causa del basso numero di prove effettuate e dei piccoli scostamenti che ci sono fra queste prove con immagini della stessa dimensione si riesce solo intuitivamente a carpire certe informazioni riguardo le prestazioni dell'algoritmo con le varie configurazioni dei Job. Per quasi tutte le immagini sembra comunque di intuire che sotto una certa dimensione nell'intorno di 10 processori ci sia un numero ottimo di processori per i quali i tempi di comunicazione più i tempi di calcolo siano minimi. In teoria ogni scostamento dall'ottimo porterà a tempi all'incirca peggiori. Lo studio effettuato è comunque riuscito ad evidenziare i vantaggi del calcolo parallelo e l'importanza della rete di interconnessione usata per far comunicare i processori. Si sono inoltre potute osservare da vicino le numerose problematiche che si hanno in fase di progettazione e implementazione per ottenere dei buoni risultati in termini di guadagno di tempo.

2.2 MFLOPS

Una alternativa al tempo di esecuzione per la misura delle prestazioni è il cosiddetto MFLOPS (Million of Floating Point Instructions Per Second). Tale indice viene calcolato con la seguente formula e valuta il numero di operazioni in virgola mobile al secondo e non di istruzioni.

$$MFLOPS = \frac{\text{Numero di operazioni in virgola mobile}}{\text{tempo di esecuzione} * 10^6}$$

Il MFLOPS dipende sia dalla macchina che dal programma che si sta usando. Nella macchina parallela è dichiarata una potenza di picco di $1,5MFLOPS$ per CPU. Visto che in ogni nodo ci sono 4 CPU si ha una potenza di picco pari a 6 GFLOPS per nodo. Utilizziamo nel seguito una funzione di costo che tenga conto del numero di operazioni aritmetiche (addizione, sottrazione, moltiplicazione e divisione) che vengono svolte dai diversi nodi. Data un'immagine di dimensione $W * H$, viene detto X il numero canonico di operazioni FP richieste per il calcolo di una funzione complessa, quale \sin e \cos . Abbiamo cercato di valutare il numero di queste operazioni nel caso in cui si utilizzi *un solo processore e un task*. Mediamente per un'esecuzione con un qualsiasi angolo il numero di operazioni richiesto dall'algoritmo di rotazione è

- $8 * W * H + 2 * X + 2$

Ricordiamo che maggiore è il valore di MFLOPS migliori sono le prestazioni. Ponendo $X = 8$, $W = 800$, $H = 600$ otteniamo circa 3,840 MFLOP. Il tempo ottenuto per l'algoritmo di rotazione con un'immagine 800x600 utilizzando 1 TASK in un NODO è di circa 1.6 secondi. La misura analitica in MFLOPS sarà circa :

$$3,840/1.6 = 2,4MFLOPS.$$

Per un'immagine 1024x768 nelle stesse condizioni dettate prima con un processore otteniamo circa 6,291 MFLOPS, quindi

$$6,291/0,4 = 15.72MFLOPS.$$

Per un'immagine 2048x1536 nelle medesime condizioni sopra menzionate con un processore otteniamo circa 25,165 MFLOPS, quindi

$$25,165/1.2 = 20,97MFLOPS.$$

Nel caso in cui si utilizzino *più processori* il numero di operazioni richiesto per ogni iterazione dell'algoritmo è:

- nel nodo 0: $8 * W * (NUMRIGHE + RESTO) + 2 * X + 2$
- negli altri nodi: $8 * W * NUMRIGHE$

Appendice A

I risultati

TEMPI per IMMAGINE di DIMENSIONE 800 x 600 su RETE ETHERNET

TASK=4 NODI=4 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,3585607444 | 1,7409375533 | 1,6332649998 | 1,7252957446 | 48,6299079908 |
| 2 | 3,3451590426 | 1,7198267023 | 1,6127892553 | 1,7323697873 | 48,2126330851 |
| 3 | 3,2440187233 | 1,6185457448 | 1,5079592550 | 1,7360594683 | 46,4842956733 |
| 4 | 3,9660681915 | 2,3455403191 | 2,2349396807 | 1,7311285108 | 56,3515192574 |
| 5 | 3,3138737234 | 1,6948970214 | 1,5843608512 | 1,7295128722 | 47,8099343384 |
| MEDIA | 3,4455360850 | 1,8239494682 | 1,7146628084 | 1,7308732766 | 49,4976580690 |

TASK=24 NODI=24 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 8,7841829788 | 7,1219152126 | 7,0228512764 | 1,7613317024 | 79,9488272660 |
| 2 | 10,0397315957 | 8,3162360638 | 8,2197003188 | 1,8200312769 | 81,8717138047 |
| 3 | 8,7615797869 | 7,1115876595 | 7,0160389361 | 1,7455408508 | 80,0773274540 |
| 4 | 11,2395864893 | 9,5769338298 | 9,4810288297 | 1,7585576596 | 84,3538936130 |
| 5 | 11,0375393617 | 9,4421497872 | 9,3454986168 | 1,6920407449 | 84,6701271952 |
| MEDIA | 9,9725240425 | 8,3137645106 | 8,2170235956 | 1,7555004469 | 82,1843778666 |

TASK=10 NODI=10 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 6,1842247874 | 4,5985670211 | 4,4971574466 | 1,6870673408 | 72,7198250577 |
| 2 | 6,1844359576 | 4,5682992556 | 4,4707265960 | 1,7137093616 | 72,2899651096 |
| 3 | 9,0109414894 | 7,3687597872 | 7,2715894682 | 1,7393520212 | 80,6973330902 |
| 4 | 5,3048955319 | 3,7249158511 | 3,6303209576 | 1,6745745743 | 68,4334109094 |
| 5 | 4,9858234043 | 3,3734143616 | 3,2788154250 | 1,7070079793 | 65,7627669321 |
| MEDIA | 6,3340642341 | 4,7267912553 | 4,6297219787 | 1,7043422554 | 71,9806602198 |

TASK=24 NODI=6 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 11,3648578722 | 9,7043730854 | 9,6073813834 | 1,7574764888 | 84,5358691804 |
| 2 | 14,0892562766 | 12,3932928722 | 12,2923315959 | 1,7969246807 | 87,2461353146 |
| 3 | 11,9799791491 | 10,3117129786 | 10,2118075534 | 1,7681715957 | 85,2406120775 |
| 4 | 7,5541968087 | 5,9108142552 | 5,8155946806 | 1,7386021281 | 76,9849505892 |
| 5 | 9,8748752128 | 8,1891627659 | 8,0890332984 | 1,7858419144 | 81,9152963873 |
| MEDIA | 10,9726330639 | 9,3018711915 | 9,2032297023 | 1,7694033615 | 83,1845727098 |

TASK=16 NODI=16 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 8,1306447873 | 6,5058785104 | 6,4084789362 | 1,7221658511 | 78,8188280733 |
| 2 | 9,0031744679 | 7,3748949999 | 7,2761874469 | 1,7269870210 | 80,8180211636 |
| 3 | 9,0890788299 | 7,4487504254 | 7,3524342552 | 1,7366445747 | 80,8930629035 |
| 4 | 12,1615806383 | 10,5373368084 | 10,4376196805 | 1,7239609578 | 86,8245321147 |
| 5 | 8,1154946806 | 6,4775580850 | 6,3768309576 | 1,7386637230 | 78,5759982426 |
| MEDIA | 9,2999946808 | 7,6688837658 | 7,5703102553 | 1,7296844255 | 81,1860884995 |

TEMPI per IMMAGINE di DIMENSIONE 800 x 600 su RETE SWITCH

TASK=1 NODI=1
NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 1,7824474468 | 0,1592629789 | 0,0013195747 | 1,7811278722 | 0,0007403162 |
| 2 | 1,7448765957 | 0,1596015957 | 0,0013137234 | 1,7435628723 | 0,0752903325 |
| 3 | 1,7963891490 | 0,1596027662 | 0,0013072337 | 1,7950819153 | 0,0727700734 |
| 4 | 1,7736454254 | 0,1595507448 | 0,0013450000 | 1,7723004254 | 0,0758325186 |
| 5 | 1,8093946809 | 0,1592259575 | 0,0013455318 | 1,8080491491 | 0,0743636420 |
| MEDIA | 1,7813506596 | 0,1594488086 | 0,0013262127 | 1,7800244469 | 0,0597993765 |

TASK=10 NODI=10
NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 1,6828001062 | 0,1129107445 | 0,0154470208 | 1,6673530855 | 0,0091793557 |
| 2 | 1,7534780849 | 0,1132049998 | 0,0155743621 | 1,7379037228 | 0,8881982765 |
| 3 | 1,7341057446 | 0,1125537236 | 0,0153996805 | 1,7187060641 | 0,8880473708 |
| 4 | 1,7312937232 | 0,1128143619 | 0,0153092556 | 1,7159844676 | 0,8842667980 |
| 5 | 1,6861071275 | 0,1130431914 | 0,0155648938 | 1,6705422337 | 0,9231260330 |
| MEDIA | 1,7175569573 | 0,1129054042 | 0,0154590426 | 1,7020979147 | 0,7185635668 |

TASK=4 NODI=1
SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 1,7491006386 | 0,1270341489 | 0,0180514893 | 1,7310491493 | 0,0103204406 |
| 2 | 1,7475898934 | 0,1264424466 | 0,0183163830 | 1,7292735104 | 1,0480938965 |
| 3 | 1,7080407448 | 0,1263941489 | 0,0177650005 | 1,6902757443 | 1,0400806021 |
| 4 | 1,7438614895 | 0,1263589361 | 0,0181268088 | 1,7257346807 | 1,0394637943 |
| 5 | 1,7438489362 | 0,1259321279 | 0,0182834039 | 1,7255655323 | 1,0484511339 |
| MEDIA | 1,7384883405 | 0,1264323617 | 0,0181086171 | 1,7203797234 | 0,8372819735 |

TASK=16 NODI=16
NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 1,7553704255 | 0,1137320213 | 0,0171947875 | 1,7381756380 | 0,0097955322 |
| 2 | 1,7560958508 | 0,1137809574 | 0,0172378726 | 1,7388579782 | 0,9816020336 |
| 3 | 1,7482138297 | 0,1144174468 | 0,0176459574 | 1,7305678723 | 1,0093706567 |
| 4 | 1,7451159575 | 0,1148709576 | 0,0175499998 | 1,7275659577 | 1,0056638175 |
| 5 | 1,7391178727 | 0,1141013829 | 0,0172652127 | 1,7218526600 | 0,9927569024 |
| MEDIA | 1,7487827872 | 0,1141805532 | 0,0173787660 | 1,7314040212 | 0,7998377885 |

TASK=4 NODI=1
NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 1,7417467022 | 0,1258180852 | 0,0178438295 | 1,7239028728 | 0,0102447902 |
| 2 | 1,7738554257 | 0,1258443617 | 0,0176575531 | 1,7561978726 | 0,9954336088 |
| 3 | 1,7474196809 | 0,1257236169 | 0,0178622343 | 1,7295574466 | 1,0222063114 |
| 4 | 1,7083787236 | 0,1261770213 | 0,0181954254 | 1,6901832982 | 1,0650697730 |
| 5 | 1,8019247875 | 0,1258705319 | 0,0179239362 | 1,7840008513 | 0,9947105658 |
| MEDIA | 1,7546650640 | 0,1258867234 | 0,0178965957 | 1,7367684683 | 0,8175330098 |

TASK=24 NODI=24
NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 1,8216153192 | 0,1114874468 | 0,0200408513 | 1,8015744679 | 0,0110016923 |
| 2 | 1,8355214891 | 0,1113706382 | 0,0199194676 | 1,8156020215 | 1,0852211615 |
| 3 | 1,7678774467 | 0,1120782979 | 0,0197222342 | 1,7481552125 | 1,1155883153 |
| 4 | 1,7774961703 | 0,1123495747 | 0,0199858511 | 1,7575103192 | 1,1243822332 |
| 5 | 1,8405953189 | 0,1115451062 | 0,0196826593 | 1,8209126596 | 1,0693637596 |
| MEDIA | 1,8086211488 | 0,1117662128 | 0,0198702127 | 1,7887509361 | 0,8811114324 |

TASK=4 NODI=4
NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 1,7488840425 | 0,1192729787 | 0,0119341491 | 1,7369498934 | 0,0068238653 |
| 2 | 1,7459040429 | 0,1197025534 | 0,0118403193 | 1,7340637236 | 0,6781769771 |
| 3 | 1,7032545742 | 0,1201246812 | 0,0123026595 | 1,6909519147 | 0,7223030332 |
| 4 | 1,7934026595 | 0,1198126597 | 0,0116689366 | 1,7817337229 | 0,6506590431 |
| 5 | 1,7471586168 | 0,1193061701 | 0,0118905320 | 1,7352680848 | 0,6805639665 |
| MEDIA | 1,7477207872 | 0,1196438086 | 0,0119273193 | 1,7357934679 | 0,5477053770 |

TASK=24 NODI=6
SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 1,7069288299 | 0,1116491491 | 0,0198760643 | 1,6870527656 | 0,0116443427 |
| 2 | 1,8001598935 | 0,1146158511 | 0,0196753191 | 1,7804845744 | 1,0929761936 |
| 3 | 1,7028587237 | 0,1115754256 | 0,0199165957 | 1,6829421280 | 1,1695976525 |
| 4 | 1,7027273404 | 0,1118047873 | 0,0200560638 | 1,6826712766 | 1,1778787654 |
| 5 | 1,7098241490 | 0,1125672341 | 0,0196772343 | 1,6901469147 | 1,1508338070 |
| MEDIA | 1,7244997873 | 0,1124424894 | 0,0198402554 | 1,7046595319 | 0,9205861522 |

TEMPI per IMMAGINE di DIMENSIONE 1024 x 768 su RETE ETHERNET

TASK=4 NODI=4 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 5,3403555318 | 2,7244070212 | 2,5298575533 | 2,8104979785 | 47,3724556025 |
| 2 | 6,1261226598 | 3,5310025532 | 3,3376036170 | 2,7885190428 | 54,4815016342 |
| 3 | 5,9304760639 | 3,3359774468 | 3,1455677662 | 2,7849082977 | 53,0407294842 |
| 4 | 5,5671787234 | 2,9793206381 | 2,7869489358 | 2,7802297876 | 50,0603460800 |
| 5 | 5,9786721277 | 3,3907189365 | 3,1970718086 | 2,7816003191 | 53,4746134314 |
| MEDIA | 5,7885610213 | 3,1922853192 | 2,9994099362 | 2,7891510851 | 51,6859292465 |

TASK=24 NODI=24 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 21,5036676594 | 18,8470872340 | 18,6734855317 | 2,8301821277 | 86,8386073832 |
| 2 | 21,5580278724 | 18,8888867020 | 18,7155412766 | 2,8424865958 | 86,8147188019 |
| 3 | 18,6487890424 | 15,9953599998 | 15,8236077661 | 2,8251812763 | 84,8505912642 |
| 4 | 19,1735594680 | 16,5519674467 | 16,3776569148 | 2,7959025532 | 85,4179264006 |
| 5 | 19,0375956383 | 16,4180171276 | 16,2467424471 | 2,7908531912 | 85,3403063906 |
| MEDIA | 19,9843279361 | 17,3402637020 | 17,1674067873 | 2,8169211488 | 85,8524300481 |

TASK=10 NODI=10 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 13,3554706383 | 10,7696986170 | 10,5944128721 | 2,7610577662 | 79,3263911026 |
| 2 | 9,0381739361 | 6,4510087234 | 6,2766809573 | 2,7614929788 | 69,4463395116 |
| 3 | 14,4887601063 | 11,8916669148 | 11,7148256383 | 2,7739344680 | 80,8545765988 |
| 4 | 15,6123986170 | 13,0513222339 | 12,8753676596 | 2,7370309574 | 82,4688632124 |
| 5 | 11,2301618084 | 8,6291108511 | 8,4572343617 | 2,7729274467 | 75,3082146628 |
| MEDIA | 12,7449930212 | 10,1585614680 | 9,9837042978 | 2,7612887234 | 77,4808770176 |

TASK=24 NODI=6 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 21,1837525531 | 18,5224507446 | 18,3441092551 | 2,8396432980 | 86,5951828372 |
| 2 | 17,7722827659 | 15,0783264893 | 14,9054607451 | 2,8668220208 | 83,8691401740 |
| 3 | 21,3353232981 | 18,6625391489 | 18,4857231912 | 2,8496001069 | 86,6437453648 |
| 4 | 21,1508404254 | 18,5045472342 | 18,3325252126 | 2,8183152128 | 86,6751620448 |
| 5 | 18,7898609575 | 16,1207535109 | 15,9255536175 | 2,8643073400 | 84,7561014609 |
| MEDIA | 20,0464120000 | 17,3777234256 | 17,1986744043 | 2,8477375957 | 85,7078663763 |

TASK=16 NODI=16 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 18,8463340427 | 16,2397882978 | 16,0678969147 | 2,7784371280 | 85,2574133423 |
| 2 | 19,9181698938 | 17,3268342551 | 17,1533953187 | 2,7647745751 | 86,1193343069 |
| 3 | 21,0799025532 | 18,5362498937 | 18,3600387233 | 2,7198638299 | 87,0973605166 |
| 4 | 12,6339577660 | 10,0583446808 | 9,8830617019 | 2,7508960641 | 78,2261733413 |
| 5 | 15,0098364893 | 12,4263047874 | 12,2508814894 | 2,7589549999 | 81,6190202876 |
| MEDIA | 17,4976401490 | 14,9175043830 | 14,7430548296 | 2,7545853194 | 83,6638603589 |

TEMPI per IMMAGINE di DIMENSIONE 1024 x 768 su RETE SWITCH

TASK=1 NODI=1 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 2,9804611700 | 0,2764661703 | 0,0023730849 | 2,9780880851 | 0,0796213994 |
| 2 | 2,8275447874 | 0,2773597871 | 0,0023731918 | 2,8251715956 | 0,0839311820 |
| 3 | 2,8881227660 | 0,2774950000 | 0,0024459569 | 2,8856768091 | 0,0846901992 |
| 4 | 2,8694903192 | 0,2768527658 | 0,0022275536 | 2,8672627656 | 0,0776288952 |
| 5 | 2,8774647871 | 0,2768761702 | 0,0022509573 | 2,8752138298 | 0,0782271016 |
| MEDIA | 2,8886167659 | 0,2770099787 | 0,0023341489 | 2,8862826170 | 0,0808197555 |

TASK=10 NODI=10 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 2,7908753192 | 0,1983920212 | 0,0232992552 | 2,7675760640 | 0,8348368347 |
| 2 | 2,7853468084 | 0,1979028725 | 0,0232950004 | 2,7620518080 | 0,8363411105 |
| 3 | 2,7811207448 | 0,1992289361 | 0,0233425526 | 2,7577781922 | 0,8393217964 |
| 4 | 2,7737556382 | 0,1980229786 | 0,0231976598 | 2,7505579784 | 0,8363267275 |
| 5 | 2,7735808510 | 0,1979215958 | 0,0233306382 | 2,7502502128 | 0,8411738989 |
| MEDIA | 2,2262197021 | 0,1982936808 | 0,0186245107 | 2,7576428511 | 0,8376000736 |

TASK=4 NODI=1 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 2,7697079787 | 0,2217154256 | 0,0285245741 | 2,7411834046 | 1,0298765915 |
| 2 | 2,8266445743 | 0,2226278721 | 0,0307005320 | 2,7959440423 | 1,0861122159 |
| 3 | 2,9660097871 | 0,2162598935 | 0,0296660641 | 2,9363437230 | 1,0002011525 |
| 4 | 2,9651935105 | 0,2222743619 | 0,0293891490 | 2,9358043615 | 0,9911376421 |
| 5 | 2,9721898935 | 0,2212570212 | 0,0294152128 | 2,9427746807 | 0,9896814759 |
| MEDIA | 2,8999491488 | 0,2208269149 | 0,0295391064 | 2,8704100424 | 1,0194018156 |

TASK=16 NODI=16 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 2,8054823405 | 0,1986706383 | 0,0258688300 | 2,7796135105 | 0,9220813713 |
| 2 | 2,8024158508 | 0,1978921278 | 0,0254893613 | 2,7769264895 | 0,9095495678 |
| 3 | 2,7880619147 | 0,1977940425 | 0,0254107448 | 2,7626511699 | 0,9114125010 |
| 4 | 2,7808655321 | 0,1987656383 | 0,0257843619 | 2,7550811702 | 0,9272063527 |
| 5 | 2,8258512767 | 0,1986792553 | 0,0257257447 | 2,8001255320 | 0,9103715003 |
| MEDIA | 2,8005353830 | 0,1983603404 | 0,0256558085 | 2,7748795744 | 0,9161242586 |

TASK=4 NODI=1 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 2,8265571275 | 0,2202260641 | 0,0284820215 | 2,7980751060 | 1,0076577350 |
| 2 | 2,8251565958 | 0,2209806382 | 0,0285654259 | 2,7965911699 | 1,0111094703 |
| 3 | 2,7644682976 | 0,2209709575 | 0,0294124470 | 2,7350558506 | 1,0639458964 |
| 4 | 2,8201743618 | 0,2161611703 | 0,0289105319 | 2,7912638299 | 1,0251327821 |
| 5 | 2,8579898938 | 0,2226395744 | 0,0288915958 | 2,8290982980 | 1,0109061562 |
| MEDIA | 2,8188692553 | 0,2201956809 | 0,0288524044 | 2,7900168509 | 1,0237504080 |

TASK=24 NODI=24 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 2,8848927661 | 0,1943470214 | 0,0281355318 | 2,8567572343 | 0,9752713219 |
| 2 | 2,8783172339 | 0,1949056385 | 0,0284277662 | 2,8498894677 | 0,9876522934 |
| 3 | 2,9026163830 | 0,1979519150 | 0,0300865958 | 2,8725297872 | 1,0365336589 |
| 4 | 2,8885488298 | 0,1944557447 | 0,0287236173 | 2,8598252125 | 0,9943961141 |
| 5 | 2,8584437233 | 0,1942778723 | 0,0285141491 | 2,8299295742 | 0,9975410361 |
| MEDIA | 2,8825637872 | 0,1951876384 | 0,0230747022 | 2,8537862552 | 0,9982788849 |

TASK=4 NODI=4 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 2,8486732978 | 0,2093464895 | 0,0188342549 | 2,8298390429 | 0,6611588262 |
| 2 | 2,8257542551 | 0,2099278725 | 0,0187298935 | 2,8070243616 | 0,6628281089 |
| 3 | 2,7982874468 | 0,2096249999 | 0,0188705323 | 2,7794169145 | 0,6743600376 |
| 4 | 2,7983758510 | 0,2091589363 | 0,0187731918 | 2,7796026592 | 0,6708602718 |
| 5 | 2,8422220212 | 0,2088147872 | 0,0184246809 | 2,8237973403 | 0,6482491792 |
| MEDIA | 2,8226625744 | 0,2093746171 | 0,0187265107 | 2,8039360637 | 0,6634912847 |

TASK=24 NODI=6 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 2,8819379788 | 0,1945887231 | 0,0282704257 | 2,8536675531 | 0,9809519132 |
| 2 | 2,8973756384 | 0,1946731915 | 0,0284595743 | 2,8689160641 | 0,9822535246 |
| 3 | 2,8991182977 | 0,1946003190 | 0,0281923399 | 2,8709259578 | 0,9724453104 |
| 4 | 2,9213322341 | 0,1949848935 | 0,0285488297 | 2,8927834044 | 0,9772537814 |
| 5 | 2,8680288300 | 0,1969720214 | 0,0281088296 | 2,8399200004 | 0,9800748632 |
| MEDIA | 2,8935585958 | 0,1951638297 | 0,0283159998 | 2,8652425960 | 0,9785958786 |

TEMPI per IMMAGINE di DIMENSIONE 1191 x 893 su RETE ETHERNET

TASK=4 NODI=4 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 8,2619478723 | 4,9711172339 | 4,7025319149 | 3,5594159574 | 56,9179567286 |
| 2 | 7,6852248937 | 4,3389640427 | 4,0704829784 | 3,6147419153 | 52,9650470175 |
| 3 | 7,2369901063 | 3,8971196811 | 3,6266358509 | 3,6103542554 | 50,1124887226 |
| 4 | 7,1844952127 | 3,8913880852 | 3,6229326592 | 3,5615625535 | 50,4271010273 |
| 5 | 7,0504925533 | 3,7581152127 | 3,4876913829 | 3,5628011704 | 49,4673436859 |
| MEDIA | 7,4838301277 | 4,1713408511 | 3,9020549573 | 3,5817751704 | 51,9779874364 |

TASK=24 NODI=24 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 36,5823268085 | 33,2867186170 | 33,0404598941 | 3,5418669144 | 90,3180928515 |
| 2 | 30,6832725534 | 27,2787226595 | 27,0371509574 | 3,6461215960 | 88,1169077071 |
| 3 | 26,7803993619 | 23,3547037235 | 23,1105153190 | 3,6698840429 | 86,2963804489 |
| 4 | 30,8144421277 | 27,4077598939 | 27,1646569150 | 3,6497852127 | 88,1556018519 |
| 5 | 24,8522671277 | 21,4695768086 | 21,2263072345 | 3,6259598932 | 85,4099431871 |
| MEDIA | 29,9425415958 | 26,5594963405 | 26,3158180640 | 3,6267235318 | 87,6593852093 |

TASK=10 NODI=10 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 17,1932810640 | 13,9023741491 | 13,6588632981 | 3,5344177659 | 79,4430292115 |
| 2 | 13,8347715957 | 10,4907160637 | 10,2497542549 | 3,5850173408 | 74,0869061991 |
| 3 | 14,2731047871 | 10,9551017024 | 10,7116958513 | 3,5614089358 | 75,0481132947 |
| 4 | 19,3860392552 | 16,0140176595 | 15,7679548936 | 3,6180843616 | 81,3366499783 |
| 5 | 19,1566306383 | 15,8817664897 | 15,6394569145 | 3,5171737238 | 81,6399147105 |
| MEDIA | 16,7687654681 | 13,4487952129 | 13,2055450425 | 3,5632204256 | 78,3109226788 |

TASK=24 NODI=6 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 28,8832965957 | 25,4506792552 | 25,2085382978 | 3,6747582979 | 87,2772199472 |
| 2 | 28,6458195744 | 25,0885749999 | 24,8344742551 | 3,8113453193 | 86,6949335858 |
| 3 | 27,5461604258 | 24,1444714894 | 23,8874864895 | 3,6586739363 | 86,7180257440 |
| 4 | 30,7293393614 | 27,3827557447 | 27,1260730855 | 3,6032662759 | 88,2741824236 |
| 5 | 32,3505511703 | 28,9703152128 | 28,7245539364 | 3,6259972339 | 88,7915441848 |
| MEDIA | 29,6310334255 | 26,2073593404 | 25,9562252129 | 3,6748082127 | 87,5511811771 |

TASK=16 NODI=16 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 28,8211939360 | 25,4563806381 | 25,2044859577 | 3,6167079783 | 87,4512208400 |
| 2 | 21,7680135109 | 18,4161710637 | 18,1691118083 | 3,5989017026 | 83,4670182432 |
| 3 | 18,6097217021 | 15,3317401065 | 15,0854264894 | 3,5242952127 | 81,0620746021 |
| 4 | 27,1398622342 | 23,8147357446 | 23,5686752128 | 3,5711870214 | 86,8415432966 |
| 5 | 23,6909087235 | 20,3726656383 | 20,1124384045 | 3,5784703190 | 84,8951749348 |
| MEDIA | 24,0059400213 | 20,6783386382 | 20,4280275745 | 3,5779124468 | 84,7434063833 |

TEMPI per IMMAGINE di DIMENSIONE 1191 x 893 su RETE SWITCH

TASK=1 NODI=1 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,8712342551 | 0,3851930851 | 0,0034977661 | 3,8677364890 | 0,0903527368 |
| 2 | 3,6701052126 | 0,3856301063 | 0,0036972342 | 3,6664079784 | 0,1007391884 |
| 3 | 3,7177391490 | 0,3851114891 | 0,0033011700 | 3,7144379790 | 0,0887950936 |
| 4 | 3,7190539362 | 0,3859145746 | 0,0036163831 | 3,7154375531 | 0,0972393283 |
| 5 | 3,7230594680 | 0,3866720214 | 0,0038218086 | 3,7192376594 | 0,1026523655 |
| MEDIA | 3,7402384042 | 0,3857042553 | 0,0035868724 | 3,7366515318 | 0,0959557425 |

TASK=10 NODI=10 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,5993496808 | 0,2784105318 | 0,0326726595 | 3,5666770213 | 0,9077378524 |
| 2 | 3,6027249999 | 0,2777676596 | 0,0321098936 | 3,5706151063 | 0,8912668501 |
| 3 | 3,5996732977 | 0,2776292553 | 0,0320075534 | 3,5676657443 | 0,8891793986 |
| 4 | 3,6349611704 | 0,2803036170 | 0,0325742548 | 3,6023869156 | 0,8961376273 |
| 5 | 3,6170967019 | 0,2786039363 | 0,0323119147 | 3,5847847872 | 0,8933107789 |
| MEDIA | 3,6107611701 | 0,2785430000 | 0,0323352552 | 3,5784259149 | 0,8955265015 |

TASK=4 NODI=1 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,6922680850 | 0,3181348934 | 0,0415203192 | 3,6507477658 | 1,1245207072 |
| 2 | 3,7037119151 | 0,3087943618 | 0,0396336168 | 3,6640782983 | 1,0701052812 |
| 3 | 3,7830490426 | 0,3114241490 | 0,0413468084 | 3,7417022342 | 1,0929493106 |
| 4 | 3,5858143617 | 0,3098374470 | 0,0381691486 | 3,5476452131 | 1,0644485397 |
| 5 | 2,8091670214 | 0,2220876596 | 0,0297519153 | 2,7794151061 | 1,0591009728 |
| MEDIA | 3,5148020852 | 0,2940557022 | 0,0380843617 | 3,4767177235 | 1,0822249623 |

TASK=16 NODI=16 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,6424724467 | 0,2807134043 | 0,0342601063 | 3,6082123404 | 0,9405728327 |
| 2 | 3,6023274469 | 0,2780378724 | 0,0338928725 | 3,5684345744 | 0,9408604023 |
| 3 | 3,6148951063 | 0,2778255318 | 0,0338735103 | 3,5810215960 | 0,9370537545 |
| 4 | 3,5964241489 | 0,2781665958 | 0,0340372343 | 3,5623869146 | 0,9464187997 |
| 5 | 3,5606609576 | 0,2809961704 | 0,0354120210 | 3,5252489366 | 0,9945350429 |
| MEDIA | 3,6033560213 | 0,2791479149 | 0,0342951489 | 3,5690608724 | 0,9518881664 |

TASK=4 NODI=1 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,6567232979 | 0,3101601063 | 0,0408907444 | 3,6158325535 | 1,1182345803 |
| 2 | 3,7541575532 | 0,3073817021 | 0,0381770211 | 3,7159805321 | 1,0169264486 |
| 3 | 3,6642614892 | 0,3078735108 | 0,0384214893 | 3,6258399999 | 1,0485466011 |
| 4 | 3,5806803191 | 0,3094041490 | 0,0398503193 | 3,5408299998 | 1,1129259168 |
| 5 | 3,6668924466 | 0,3039279787 | 0,0384300007 | 3,6284624459 | 1,0480263933 |
| MEDIA | 3,6645430212 | 0,3077494894 | 0,0391539150 | 3,6253891062 | 1,0689319880 |

TASK=24 NODI=24 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,7180699999 | 0,2735285107 | 0,0384851061 | 3,6793707443 | 1,0350828812 |
| 2 | 3,6574264891 | 0,2717150000 | 0,0376627657 | 3,6195361696 | 1,0297613861 |
| 3 | 3,6671100000 | 0,2718605320 | 0,0378527662 | 3,6289249996 | 1,0322233646 |
| 4 | 3,6471359576 | 0,2717452126 | 0,0376575529 | 3,6095795750 | 1,0325239668 |
| 5 | 3,5639671276 | 0,2717118086 | 0,0378768086 | 3,5262497871 | 1,0627709851 |
| MEDIA | 3,6507419148 | 0,2721122128 | 0,0380096597 | 3,6127322551 | 1,0279071188 |

TASK=4 NODI=4 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,6405602130 | 0,2947235105 | 0,0264457446 | 3,6141144684 | 0,7264196446 |
| 2 | 3,6281260636 | 0,2957279787 | 0,0261181914 | 3,6020078722 | 0,7198810341 |
| 3 | 3,5584817021 | 0,2942780852 | 0,0259765962 | 3,5325051059 | 0,7299910001 |
| 4 | 3,6341211700 | 0,2951635108 | 0,0261163833 | 3,6080047867 | 0,7186437116 |
| 5 | 3,6287187233 | 0,2958198935 | 0,0259013830 | 3,6028173403 | 0,7137886675 |
| MEDIA | 3,6180015744 | 0,2951425957 | 0,0261116597 | 3,5918899147 | 0,7217448116 |

TASK=24 NODI=6 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 3,6569844682 | 0,2741325533 | 0,0386992556 | 3,6182852126 | 1,0582286005 |
| 2 | 3,7619802128 | 0,2726043619 | 0,0378903195 | 3,7240898933 | 1,0071908234 |
| 3 | 3,6601231915 | 0,2725440424 | 0,0381850004 | 3,6219381911 | 1,0432709071 |
| 4 | 3,6765562766 | 0,2718498935 | 0,0375563826 | 3,6389998940 | 1,0215097980 |
| 5 | 3,7368488298 | 0,2722171275 | 0,0377173405 | 3,6991314893 | 1,0093354650 |
| MEDIA | 3,6984985958 | 0,2726695957 | 0,0380096597 | 3,6604889361 | 1,0279071188 |

TEMPI per IMMAGINE di DIMENSIONE 2048 x 1536 su RETE ETHERNET

TASK=4 NODI=4 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 20,1365161701 | 13,66756362 | 12,8118415955 | 7,3246745746 | 63,6249164812 |
| 2 | 21,2368324469 | 14,77410255 | 13,9074437234 | 7,3293887235 | 65,4873732144 |
| 3 | 19,6016027660 | 13,08885362 | 12,2321709576 | 7,3694318084 | 62,4039324929 |
| 4 | 19,6677658509 | 13,19535234 | 12,3380312764 | 7,3297345745 | 62,7322461021 |
| 5 | 19,6000879786 | 13,12658713 | 12,2704130847 | 7,3296748939 | 62,6038673807 |
| MEDIA | 20,0485610425 | 13,5704918511 | 12,7119801275 | 7,3365809150 | 63,3704671343 |

TASK=24 NODI=24 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 159,5059661702 | 152,9168136169 | 152,1301014896 | 7,3758646806 | 95,3758063991 |
| 2 | 141,0065985105 | 134,3510337232 | 133,5300039360 | 7,4765945745 | 94,6976987932 |
| 3 | 157,7107189361 | 151,1821384041 | 150,3998136173 | 7,3109053188 | 95,3643573702 |
| 4 | 154,2263722338 | 147,6776158512 | 146,8977648937 | 7,3286073401 | 95,2481490461 |
| 5 | 83,2345241487 | 76,7436704254 | 75,9697267027 | 7,2647974460 | 91,2718940603 |
| MEDIA | 139,1368359999 | 132,5742544042 | 131,7854821279 | 7,3513538720 | 94,3915811338 |

TASK=10 NODI=10 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 53,0582078726 | 46,51984372 | 45,7379576596 | 7,3202502130 | 86,2033594680 |
| 2 | 52,2408092553 | 45,78536085 | 45,0049735110 | 7,2358357443 | 86,1490741674 |
| 3 | 46,6016865957 | 40,15589021 | 39,3792484042 | 7,2224381915 | 84,5017665258 |
| 4 | 53,5800898937 | 47,13682809 | 46,3594421279 | 7,2206477658 | 86,5236363356 |
| 5 | 44,0333908510 | 37,64403894 | 36,8660702126 | 7,1673206384 | 83,7229872605 |
| MEDIA | 49,9028368937 | 43,4483923618 | 42,6695383831 | 7,2332985106 | 85,4201647515 |

TASK=24 NODI=6 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 157,5394003193 | 150,8905680852 | 150,1063206382 | 7,4330796811 | 95,2817646468 |
| 2 | 146,8050247873 | 140,3091005322 | 139,4648279785 | 7,3401968088 | 95,0000370768 |
| 3 | 145,4152868085 | 138,8753406384 | 138,0931928721 | 7,3220939364 | 94,9647013756 |
| 4 | 124,0001950001 | 117,2368580850 | 116,4613357440 | 7,5388592561 | 93,9202843543 |
| 5 | 147,1335285108 | 139,7860837232 | 138,9451357445 | 8,1883927663 | 94,4347200470 |
| MEDIA | 144,1786870852 | 137,4195902128 | 136,6141625955 | 7,5645244897 | 94,7203015001 |

TASK=16 NODI=16 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 116,6803270211 | 110,2041354259 | 109,4208027665 | 7,2595242546 | 93,7782791325 |
| 2 | 98,6456503193 | 92,2416093617 | 91,4574237233 | 7,1882265960 | 92,7130830678 |
| 3 | 90,6872047873 | 84,2746608511 | 83,4927513832 | 7,1944534041 | 92,0667381678 |
| 4 | 123,8512289361 | 117,4355939361 | 116,6510856384 | 7,2001432977 | 94,1864579306 |
| 5 | 93,7737313830 | 87,4217409573 | 86,6392292550 | 7,1345021280 | 92,3917902991 |
| MEDIA | 104,7276284894 | 98,3155481064 | 97,5322585533 | 7,1953699361 | 93,0272697196 |

TEMPI per IMMAGINE di DIMENSIONE 2048 x 1536 su RETE SWITCH

TASK=1 NODI=1 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 7,7346055321 | 1,2206667021 | 0,0215418090 | 7,7130637231 | 0,2785120570 |
| 2 | 7,6955195745 | 1,2176311703 | 0,0212420214 | 7,6742775531 | 0,2760310223 |
| 3 | 7,6264306384 | 1,2182058513 | 0,0211805317 | 7,6052501067 | 0,2777253569 |
| 4 | 7,7429176595 | 1,2178828723 | 0,0212285102 | 7,7216891493 | 0,2741668079 |
| 5 | 7,6662282976 | 1,2198901062 | 0,0216602129 | 7,6445680847 | 0,2825406713 |
| MEDIA | 7,6931403404 | 1,2188553404 | 0,0213706170 | 7,6717697234 | 0,2777951831 |

TASK=10 NODI=10 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 7,2161419149 | 0,8761438299 | 0,0848648937 | 7,1312770212 | 1,1760424711 |
| 2 | 7,2974714891 | 0,8773556387 | 0,0861274467 | 7,2113440424 | 1,1802368377 |
| 3 | 7,3056088297 | 0,8753941488 | 0,0851323404 | 7,2204764893 | 1,1653011052 |
| 4 | 7,3029076594 | 0,8765988296 | 0,0856119148 | 7,2172957446 | 1,1722990176 |
| 5 | 7,3017312766 | 0,8766520212 | 0,0852331915 | 7,2164980851 | 1,1673011272 |
| MEDIA | 7,2847722339 | 0,8764288936 | 0,0853939574 | 7,1993782765 | 1,1722361118 |

TASK=4 NODI=1 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 7,6399335105 | 0,9660548938 | 0,1131508513 | 7,5267826592 | 1,4810449748 |
| 2 | 7,4735532980 | 0,9621856385 | 0,1159918087 | 7,3575614893 | 1,5520302603 |
| 3 | 7,4508602126 | 0,9627557448 | 0,1159554254 | 7,3349047872 | 1,5562689692 |
| 4 | 7,3933476598 | 0,9591895745 | 0,1124701060 | 7,2808775538 | 1,5212338329 |
| 5 | 7,4417046809 | 0,9597104257 | 0,1159252129 | 7,3257794680 | 1,5577776575 |
| MEDIA | 7,4798798724 | 0,9619792555 | 0,1146986809 | 7,3651811915 | 1,5336711389 |

TASK=16 NODI=16 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 7,2585376594 | 0,8729851064 | 0,0900301067 | 7,1685075527 | 1,2403339480 |
| 2 | 7,3423013829 | 0,8747282978 | 0,0918268084 | 7,2504745745 | 1,2506543060 |
| 3 | 7,3945818085 | 0,8705986170 | 0,0889227663 | 7,3056590422 | 1,2025394887 |
| 4 | 7,3627396810 | 0,8719848935 | 0,0901653192 | 7,2725743618 | 1,2246164208 |
| 5 | 7,3368756382 | 0,8727791491 | 0,0909504255 | 7,2459252127 | 1,2396342799 |
| MEDIA | 7,3390072340 | 0,8726152128 | 0,0903790852 | 7,2486281488 | 1,2315556887 |

TASK=4 NODI=1 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 7,5976521277 | 0,9663035106 | 0,1202381914 | 7,4774139363 | 1,5825703708 |
| 2 | 7,4508255320 | 0,9686782977 | 0,1134307447 | 7,3373947873 | 1,5223916363 |
| 3 | 7,4713664895 | 0,9600398939 | 0,1118948935 | 7,3594715960 | 1,4976496417 |
| 4 | 7,4326840425 | 0,9586138299 | 0,1125332979 | 7,3201507446 | 1,5140331166 |
| 5 | 7,7226951062 | 0,9636095746 | 0,1118905321 | 7,6108045741 | 1,4488534189 |
| MEDIA | 7,5350446596 | 0,9634490213 | 0,1139975319 | 7,4210471277 | 1,5130996369 |

TASK=24 NODI=24 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 7,4230201065 | 0,8470088299 | 0,0945780848 | 7,3285059577 | 1,2741186661 |
| 2 | 7,4232058511 | 0,8499181913 | 0,0953696803 | 7,3278932974 | 1,2847505812 |
| 3 | 7,4228351065 | 0,8466867020 | 0,0945315962 | 7,3274897875 | 1,2735241295 |
| 4 | 7,4149378722 | 0,8505408510 | 0,0973484043 | 7,3207803192 | 1,3128687796 |
| 5 | 7,4075115956 | 0,8470677661 | 0,0940990422 | 7,3128907445 | 1,2703192027 |
| MEDIA | 7,4183021064 | 0,8482444681 | 0,0951853616 | 7,3235120213 | 1,2831162718 |

TASK=4 NODI=4 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 7,3488447871 | 0,9277302127 | 0,0706724469 | 7,2781723402 | 0,9616810384 |
| 2 | 7,3797569149 | 0,9269888299 | 0,0702920211 | 7,3094648938 | 0,9524977849 |
| 3 | 7,2961997872 | 0,9348291489 | 0,0734824468 | 7,2227173404 | 1,0071331513 |
| 4 | 7,3624723405 | 0,9300093614 | 0,0708757450 | 7,2915965955 | 0,9626622927 |
| 5 | 7,3753806383 | 0,9262763830 | 0,0709108510 | 7,3044697873 | 0,9614534420 |
| MEDIA | 7,3525308936 | 0,9291667872 | 0,0712467022 | 7,2812841914 | 0,9690855419 |

TASK=24 NODI=6 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 7,4015243617 | 0,8461722343 | 0,0945141488 | 7,3070102129 | 1,2769551805 |
| 2 | 7,3970850001 | 0,8463887235 | 0,0953125537 | 7,3017724464 | 1,2885150528 |
| 3 | 7,3837351063 | 0,8471686170 | 0,0953453190 | 7,2883897873 | 1,2912884555 |
| 4 | 7,2599281915 | 0,8465021278 | 0,0941575530 | 7,1657706385 | 1,2969488204 |
| 5 | 7,2675015959 | 0,8461924470 | 0,0946208511 | 7,1728807448 | 1,3019722095 |
| MEDIA | 7,3419548511 | 0,8464848299 | 0,0947900851 | 7,2471647660 | 1,2911359437 |

TEMPI per IMMAGINE di DIMENSIONE 4000 x 3000 su RETE SWITCH

TASK=1 NODI=1 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 25,8235836171 | 4,6626614896 | 0,0849945745 | 25,7385890426 | 0,0032913547 |
| 2 | 25,1718672342 | 4,6716426595 | 0,0841856385 | 25,0876815957 | 0,0033444336 |
| 3 | 25,2014471276 | 4,6652055320 | 0,0861568083 | 25,1152903193 | 0,0034187246 |
| 4 | 24,9928438298 | 4,6639075528 | 0,0858628717 | 24,9069809581 | 0,0034354983 |
| 5 | 25,4498593616 | 4,6655928723 | 0,0853471274 | 25,3645122342 | 0,0033535402 |
| MEDIA | 25,3279202341 | 4,6658020212 | 0,0853094041 | 25,2426108300 | 0,0033687103 |

TASK=10 NODI=1 NO SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 24,1401178723 | 3,3094505318 | 0,3254217023 | 23,8146961699 | 0,0134805349 |
| 2 | 24,5502688298 | 3,2989692555 | 0,3224965958 | 24,2277722340 | 0,0131361737 |
| 3 | 23,7660335104 | 3,3006574467 | 0,3221339362 | 23,4438995742 | 0,0135543837 |
| 4 | 23,9258993617 | 3,3006963830 | 0,3237476596 | 23,6021517022 | 0,0135312640 |
| 5 | 23,9895471279 | 3,3035844681 | 0,3215517020 | 23,6679954259 | 0,0134038254 |
| MEDIA | 24,0743733404 | 3,3026716170 | 0,3230703192 | 23,7513030212 | 0,0134212363 |

TASK=24 NODI=6 SHARED MEMORY

| PROVE | Tempo esecuzione totale (sec) | Tempo rotazione (sec) | Tempo totale comunicazioni (sec) | Tempo totale senza comunic (sec) | % comunicazioni rispetto al tempo totale |
|--------------|-------------------------------|-----------------------|----------------------------------|----------------------------------|--|
| 1 | 23,6752479789 | 3,2875084043 | 0,3460319149 | 23,3292160640 | 0,0146157673 |
| 2 | 23,5837144684 | 3,2323509576 | 0,3464097872 | 23,2373046812 | 0,0146885168 |
| 3 | 23,8670942553 | 3,2339943617 | 0,3496436169 | 23,5174506383 | 0,0146496098 |
| 4 | 23,5690495747 | 3,2306607447 | 0,3451709573 | 23,2238786174 | 0,0146450945 |
| 5 | 23,5760610639 | 3,2286857446 | 0,3441586171 | 23,2319024468 | 0,0145977997 |
| MEDIA | 23,6542334682 | 3,2426400426 | 0,3462829787 | 23,3079504895 | 0,0146393576 |

Appendice B

Il Sorgente

1 Il sorgente

Listing 1: rotazione.c

```
1  /*****
2  *   PROGETTO DI CALCOLO PARALLELO
3  *   ROTAZIONE PARALLELA di immagini true color 24bpp formato PCX(ZSoft)
4  *
5  *   Marco Cattai
6  *   Valeria Bolzonaro
7  *
8  *****/
9
10 #include <stdio.h>
11 #include <stdlib.h>
12 #include <math.h>
13 #include <mpi.h>
14 /*****
15  *   VARIABILI GLOBALI
16  *****/
17
18 /*
19 HEADER FILE IMMAGINE PCX
20 */
21 typedef struct {
22 char manufacturer; /*FLAG COSTANTE=10 -> ZSoft .pcx*/
23 char version; /* INFORMAZIONI SULLA VERSIONE*/
24 char encoding; /* 1= USA PCX RLE*/
25 char bpp; /* NUMERO DI BIT PER RAPPRESENTARE UN PIXEL PER PIANO 1, 2, 4 o 8*/
26 unsigned short int xmin;
27 unsigned short int ymin;
28 unsigned short int xmax;
29 unsigned short int ymax;
30 unsigned short int hdpi;
31 unsigned short int ydpi;
32 char palette[48];
33 char reserved;
34 char nplanes;
35 unsigned short int bytes_per_line; /* NUMERO DI BYTE DA ALLOCARE PER UNA
SCANLINE (NUMERO PARI)*/
36 unsigned short int palette_info; /*COME INTERPRETARE LA PALETTE*/
37 char unused[58];
38 } PCXHeader;
39
40 PCXHeader hdr; /*Header del file*/
41 unsigned int width; /*larghezza immagine*/
42 unsigned int height; /*altezza immagine*/
43 unsigned short int bytes_per_line; /*bytes da decodificare per canale in una
linea*/
44 unsigned char *screen; /* buffer schermo virtuale */
45 unsigned char *screen2; /* buffer schermo virtuale con immagine filtrata */
46
47 /*
48 VARIABILI PER TENER TRACCIA DEL TEMPO DI ESECUZIONE TOTALE E DELLE
COMUNICAZIONI
49 FA PROCESSI
50 */
51 double time_comm;
52 double time_comm_tmp;
53
54 /*****
55  *   CONVERSIONE ENDIAN
56  *****/
57
58 unsigned short int Endian_Word_Conversion(unsigned short int word) {
59     return ((word>>8)&0x00FF) | ((word<<8)&0xFF00);
```

```

60 }
61
62 /******
63 /*  HEADER                                                    */
64 /******
65 /*-----*/
66 /* FUNZIONE read_header                                       */
67 /* UTILIZZO Legge l'intestazione di un file PCX              */
68 /* INPUT   Nomefile                                          */
69 /* OUTPUT  Width e Height                                    */
70
71 int read_header(char *filename)
72 {
73 FILE *stream;
74     if((stream = fopen(filename, "rb"))==NULL)
75     {
76         printf("Errore: Impossibile trovare il file : %s",filename);
77         return 1;
78     }
79     fread(&hdr, 1, sizeof(hdr), stream);
80
81     if(hdr.manufacturer!=10)
82     {
83         printf("Errore file PCX: produttore non valido : %s",filename);
84         return 1;
85     }
86
87     width=Endian_Word_Conversion(hdr.xmax)-Endian_Word_Conversion(hdr.
88         xmin)+1;
89     height=Endian_Word_Conversion(hdr.ymax)-Endian_Word_Conversion(hdr.
90         ymin)+1;
91     bytes_per_line=Endian_Word_Conversion(hdr.bytes_per_line);
92     fclose(stream);
93 return 1;
94 }
95
96 /*-----*/
97 /* IMMAGINE TRUE COLOR                                       */
98 /******
99 /*-----*/
100 /* FUNZIONE load_pcx                                         */
101 /* UTILIZZO Legge un file PCX                                */
102 /* INPUT   Nomefile                                          */
103 /* OUTPUT  nessuno (1 successo)                              */
104 int pcx(char *nomefile)
105 {
106     FILE *stream;
107     unsigned char *image_buf;
108     unsigned long int filesize;
109     stream=fopen(nomefile,"rb");
110
111     /*PRENDO LA GRANDEZZA DEL FILE*/
112     fseek(stream,0,SEEK_END);
113     filesize=ftell(stream);
114
115
116     if(!(image_buf=(unsigned char *)malloc(filesize-128)))
117     {
118         printf("Errore:\n Memoria RAM insufficiente per continuare.
119             Chiedere le altre applicazioni e riavviare :(\n");
120         fclose(stream);
121         return 0;
122     }
123 /*IN TRUE COLOR IL FILE E' MEMORIZZATO CON I VALORI RGB*/
124

```

```

125     fseek(stream,128,SEEK_SET);
126     fread(image_buf,1,filesize-128,stream); /*BUFFER DELL'IMMAGINE
        COMPRESSA IN RLE*/
127
128     write_imm(image_buf,image_buf+ (filesize-128) );
129
130     free(image_buf);
131     fclose(stream);
132     return 1;
133 }
134
135
136
137 int write_imm(unsigned char *image_buf,unsigned char *finebuff)
138 {
139     unsigned long int offset;
140     unsigned int i=0;
141     unsigned int contatore=1;
142     unsigned char corrente;
143     unsigned int r=2; /*per scrivere in screen*/
144     unsigned int g=1; /*per scrivere in screen*/
145     unsigned int b=0; /*per scrivere in screen*/
146
147     if(!(screen=(unsigned char *)malloc(width*3*height)))
148     {
149         printf("Errore:\n Memoria RAM insufficiente per continuare.
        Chiudere le altre applicazioni e riavviare :(\n");
150         return 0;
151     }
152
153     /*DECOMPRIMO I DATI DELL'IMMAGINE*/
154
155     i=0;
156     contatore=1;
157
158     while(image_buf!=finebuff)
159     {
160
161         i=0;
162         while(i<bytes_per_line)
163         {
164             corrente=*image_buf;
165             if(corrente>191){
166                 contatore=(corrente-192);
167                 image_buf++;
168                 corrente=(image_buf);
169                 while ( (contatore--)&&(i<bytes_per_line) )
170                 {
171                     screen[ r ] = corrente;
172                     r+=3;
173                     i++;
174                 }
175             }
176             else {
177                 i++;
178                 screen[ r ] = corrente;
179                 r+=3;
180             }
181             image_buf++;
182         }
183
184         i=0;
185         while(i<bytes_per_line)
186         {
187             corrente=*image_buf;
188             if(corrente>191){
189                 contatore=(corrente-192);
190                 image_buf++;

```

```

191         corrente=* (image_buf);
192         while ( (contatore--)&&(i<bytes_per_line) )
193         {
194             screen[ g ] = corrente;
195             g+=3;
196             i++;
197         }
198     }
199     else {
200         i++;
201         screen[ g ] = corrente;
202         g+=3;
203     }
204     image_buf++;
205 }
206
207     i=0;
208     while(i<bytes_per_line)
209     {
210         corrente=*image_buf;
211         if(corrente>191){
212             contatore=(corrente-192);
213             image_buf++;
214             corrente=* (image_buf);
215             while ( (contatore--)&&(i<bytes_per_line) )
216             {
217                 screen[ b ] = corrente;
218                 b+=3;
219                 i++;
220             }
221         }
222         else {
223             i++;
224             screen[ b ] = corrente;
225             b+=3;
226         }
227         image_buf++;
228     }
229 }
230
231 }
232
233
234
235
236
237 /*
238 ROUTINE DI CODIFICA FORMATO RLE TRUE COLOR 24 BIT (R-G-B)
239 */
240
241 /*-----*/
242 /* FUNZIONE comprimi_scanline */
243 /* UTILIZZO Decomprime un canale(R G B) alla volta la scanline */
244 /* INPUT File di destinazione, canale da decomprimere, incremento,buffer*/
245 /* OUTPUT scrive nel file il canale della scanline compressa */
246
247 Comprimi_scanline(FILE *filedest, int canale, int y, int incremento, unsigned
    char *buf){
248
249     unsigned char primo, corrente, contatore;
250     unsigned int i=0;
251     unsigned int offset;
252     unsigned int x;
253     x=canale;
254     while(i<(bytes_per_line)){
255         contatore=1;
256         offset=y*(bytes_per_line*3)+x;
257         corrente=buf[offset];

```

```

258         x+=incremento;
259         i++;
260         while ( ( i < bytes_per_line ) && ( corrente == buf[ (y*(
                bytes_per_line*3)+x) ] ) && ( contatore < 63 ) )
261             {
262                 x+=incremento;
263                 ++contatore;
264                 i++;
265             }
266         primo=corrente;
267         if ( contatore > 1 || primo >= 0xc0)
268             {
269                 contatore |= 0xc0;
270                 fwrite(&contatore,1,1,filedest);
271             }
272
273         fwrite(&primo,1,1,filedest);
274     }
275
276 }
277
278
279 /*-----*/
280 /* FUNZIONE Salva_File */
281 /* UTILIZZO Salva l'immagine in Buffer in filedest in formato PCX */
282 /* INPUT Nome File e Buffer Bitmap */
283 /* OUTPUT Salva l'immagine in formato PCX in nomefile */
284
285 Salva_File(char *nomefile, unsigned char *buf){
286
287 FILE *filedest;
288 unsigned char primo, corrente, contatore;
289 int x,y;
290
291     filedest=fopen(nomefile, "wb");
292     fclose(filedest);
293
294     filedest=fopen(nomefile,"ab");
295     fwrite(&hdr,1,sizeof(hdr),filedest);
296
297
298 for(y=0;y<height;y++)
299 {
300     Comprimi_scanline(filedest, 2, y, 3, buf);
301
302     Comprimi_scanline(filedest, 1, y, 3, buf);
303
304     Comprimi_scanline(filedest, 0, y, 3, buf);
305 }
306
307 fclose(filedest);
308 }
309
310
311 /*-----*/
312 /* FUNZIONE Init_screen */
313 /* UTILIZZO Inizializza lo schermo virtuale dove effettuare la rotazione */
314 /* INPUT // */
315 /* OUTPUT schermo virtuale screen2 */
316
317 int Init_screen()
318 {
319     if(!(screen2=(unsigned char *)malloc(width*height*3))
320     {
321         printf("Errore:\n Memoria RAM insufficiente per continuare.
322             Chiudere le altre applicazioni e riavviare :(\n");
323         return 0;
324     }

```

```

324 }
325 }
326
327 long int cos_rad(int angle){
328     return cos((6.28/256)*angle)*1024;
329 }
330 long int sin_rad(int angle){
331     return sin((6.28/256)*angle)*1024;
332 }
333
334
335 /*-----*
336 *      24 bpp Rotazione parallela rispetto al centro dell'immagine *
337 *      *
338 *      i_angle [0..255] *
339 *      r_s - g_s - b_s   colore di sfondo *
340 *-----*/
341 int rotate(unsigned char *source,unsigned char *dest, int i_angle, char r_s,
           int g_s, int b_s)
342 {
343     unsigned char r, g, b;
344     int x, y, xr, yr;
345     int c, s;
346     int h_d2=height/2;
347     int w_d2=width/2;
348     int angle=i_angle;
349     int w_per3=width*3;
350     int numrighe;
351     int resto;
352     unsigned int *i_arr;
353     int start, end;
354     int myrank, ranksize;
355     unsigned long int val;
356
357     MPI_Comm_size(MPI_COMM_WORLD, &ranksize);
358     MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
359
360     if (myrank == 0) /*PROCESSO ROOT HA ACCESSO ALLE TABELLE SIN E COS*/
361     {
362         c = cos_rad(angle);
363         s = sin_rad(angle);
364     }
365 }
366 /*
367 IL PROCESSO ROOT INVIA I DATI COS E SIN A TUTTI I PROCESSORI, NECESSARI PER
    LA ROTAZIONE
368 */
369
370 if(myrank==0)time_comm_tmp=MPI_Wtime();
371
372 MPI_Bcast(&c, 1, MPI_INT, 0, MPI_COMM_WORLD);
373 MPI_Bcast(&s, 1, MPI_INT, 0, MPI_COMM_WORLD);
374
375 if(myrank==0)time_comm+=MPI_Wtime()-time_comm_tmp;
376
377 numrighe=((int)(ceil(height / ranksize)));
378 resto=(int)(ceil(height % ranksize));
379
380
381 if(myrank==0){
382
383     if(!(i_arr=(unsigned int *)malloc((width*(height-resto))*sizeof(int)
    )))
384     {
385         printf("Errore:\n Memoria RAM insufficiente per continuare.
    Chiudere le altre applicazioni e riavviare :(\n");
386
387     return 1;
388 }

```



```

388
389
390 for (y=-h_d2; y<-h_d2+resto; y++) {
391     for (x=-w_d2; x<w_d2; x++) {
392         xr=((x*c) - (y*s))>>10;
393         yr=((x*s) + (y*c))>>10;
394         if (xr>-w_d2 && xr<w_d2 && yr>-h_d2 && yr<h_d2) {
395             r=source[(yr+h_d2)*w_per3+(xr+w_d2)*3];
396             g=source[(yr+h_d2)*w_per3+(xr+w_d2)*3+1];
397             b=source[(yr+h_d2)*w_per3+(xr+w_d2)*3+2];
398         } else {
399             r=r_s;
400             g=g_s;
401             b=b_s;
402         }
403         screen2[(y+h_d2)*w_per3+(x+w_d2)*3]=r;
404         screen2[(y+h_d2)*w_per3+(x+w_d2)*3+1]=g;
405         screen2[(y+h_d2)*w_per3+(x+w_d2)*3+2]=b;
406     }
407 }
408
409 } /*tutti i processori calcolano la loro fetta di matrice*/
410
411 /*2) OGNI PROCESSO ALLOCA UN VETTORE DI WIDTH*NUMRIGHE ELEMENTI*/
412
413 if (myrank!=0) {
414     if (!(i_arr=(unsigned int *)malloc( width*numrighe *sizeof(int) )))
415     {
416         printf("Errore:\n Memoria RAM insufficiente per continuare. Chiudere
417             le altre applicazioni e riavviare :(\n");
418         return 1;
419     }
420 }
421
422 start=-h_d2+(resto)+(numrighe)*(myrank);
423 end=start+numrighe;
424
425     for (y=start; y<end; y++)
426     {
427         for (x=-w_d2; x<w_d2; x++)
428         {
429             xr=((x*c) - (y*s))>>10;
430             yr=((x*s) + (y*c))>>10;
431             if (xr>-w_d2 && xr<w_d2 && yr>-h_d2 && yr<h_d2)
432             {
433                 i_arr[(y-start)*width+(x+w_d2)]=(yr+h_d2)*w_per3+(
434                     xr+w_d2)*3;
435             } else
436             {
437                 i_arr[(y-start)*width+(x+w_d2)]=height*width+2;
438             }
439         }
440     }
441
442 /*printf("PROCESSO %d FINITO E PRONTO A COMUNICARE %d DATI\n", myrank, (end-
443     start)*width);*/
444 /*comunicazione in gather della matrice...poi ci pensa il processo 0 a
445     terminare la rotazione*/
446
447 if (myrank==0) time_comm_tmp=MPI_Wtime();
448
449 MPI_Gather(i_arr, numrighe*width, MPI_INT, i_arr, numrighe*width, MPI_INT, 0,
450     MPI_COMM_WORLD);
451
452 if (myrank==0) time_comm+=MPI_Wtime()-time_comm_tmp;

```

```

451
452
453 if(myrank==0){
454
455 /*
456 width*height
457 nell'immagine ho già ruotato numrighe + resto
458 */
459
460
461 start=resto;
462
463 for(y=start;y<height;y++)
464 {
465     for(x=0;x<width;x++)
466     {
467         val=i_arr[(y-start)*width+x];
468         if(val==width*height+2){
469             dest[(y)*w_per3+x*3]=b_s;
470             dest[(y)*w_per3+x*3+1]=g_s;
471             dest[(y)*w_per3+x*3+2]=r_s;
472         }else {
473             dest[(y)*w_per3+x*3]=source[val];
474             dest[(y)*w_per3+x*3+1]=source[val+1];
475             dest[(y)*w_per3+x*3+2]=source[val+2];
476         }
477     }
478 }
479
480
481 }
482
483 return 1;
484 }
485
486 /*Il processo 0 riceve i dati in gather dagli altri e completa la matrice*/
487
488
489 /*****
490 main(int argc, char *argv[])
491 {
492
493     int ranksize, myrank;
494     int i=0;
495     MPI_Status status;
496     int r_s, g_s, b_s, angle;
497     char *filesource, *filedest;
498
499     double time_rotation=0;
500     double time_total=0;
501     time_comm_tmp=0;
502     time_comm=0;
503
504
505     MPI_Init (&argc, &argv);
506
507     MPI_Comm_size(MPI_COMM_WORLD, &ranksize);
508     MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
509
510     /*PRENDO IL TEMPO INIZIALE*/
511     time_total=MPI_Wtime();
512
513     if(myrank==0)
514     {
515
516         if(argc==4){
517             filesource=argv[1];

```

```

518 filedest=argv[2];
519 angle=atoi(argv[3]);
520 r_s=0;
521 g_s=0;
522 b_s=0;
523
524 }
525 else if(argc==7){
526 /*Ho anche i parametri RGB di sfondo*/
527 filesource=argv[1];
528 filedest=argv[2];
529 angle=atoi(argv[3]);
530 r_s=atoi(argv[4]);
531 g_s=atoi(argv[5]);
532 b_s=atoi(argv[6]);
533 }
534
535 else {
536
537 printf("\nUSO: %s source.pcx dest.pcx angle\n", argv[0]);
538 printf("\nDove:\n\nangle = [0..255]\n");
539 printf("-----\n");
540 printf("\nUSO: %s source.pcx dest.pcx angle r g b\n", argv[0]);
541 printf("\nDove:\n\nr, g, b = [0..255] Definiscono colore di sfondo\n");
542
543 MPI_Finalize();
544 return 0;
545 }
546         read_header(filesource);
547 }
548
549 if(myrank==0)time_comm_tmp=MPI_Wtime();
550
551 /*PASSO L'ENDIAN DI BYTES_PER_LINE LETTO DALL'HEADER IN BROADCAST A TUTTI I
PROCESSORI*/
552 MPI_Bcast(&bytes_per_line, 1, MPI_INT, 0, MPI_COMM_WORLD);
553 /*PASSO WIDTH e HEIGHT IN BROADCAST A TUTTI I PROCESSORI*/
554 MPI_Bcast(&width, 1, MPI_INT, 0, MPI_COMM_WORLD);
555 MPI_Bcast(&height, 1, MPI_INT, 0, MPI_COMM_WORLD);
556 /*
557 SINCRONIZZAZIONE: Quando un processore riceve width ed height (indispensabili
) può proseguire
558 */
559
560 if(myrank==0)time_comm+=MPI_Wtime()-time_comm_tmp;
561
562 if(myrank==0){
563         if((hdr.bpp==8) && (hdr.nplanes==3)){
564 /*
565 Il processore 0 prima di ruotare la sua fetta carica l'immagine in memoria
566 */
567         pcx(filesource);
568
569         Init_screen();
570         } else {
571         printf("Immagine non in true color 24");
572         MPI_Finalize();
573         return 0;
574         }
575 }
576 }
577
578 /*
579 Possibilità di inserimento di filtri vari
580 */
581 if(myrank==0)time_rotation=MPI_Wtime();
582 rotate(screen, screen2, angle, r_s, g_s, b_s);
583 if(myrank==0)time_rotation=MPI_Wtime()-time_rotation;

```

```
584
585 if(myrank==0){
586     Salva_File(filedest,screen2);
587     free(screen);
588
589 time_total=MPI_Wtime()-time_total;
590
591 printf("Durata totale in secondi: %5.10f\n", time_total);
592 printf("tempo di calcolo senza comunicazioni in secondi: %5.10f \n",
593        time_total-time_comm);
594 printf("tempo per comunicazioni in secondi: %5.10f \n", time_comm);
595 printf("Percentuale tempo per comunicazioni: %5.10f %%\n", time_comm*100/
596        time_total);
597
598 }
599
600 MPI_Finalize();
601 return 0;
602 }
```

ESEMPI DI ESECUZIONE DEL PROGRAMMA

File in input:



File in output:



File in input:



File in output:



Bibliografia

- [1] MAZZON PAOLO, *Introduzione all'IBM RS/6000 SP*, Dipartimento di Ingegneria dell'Informazione, Università degli Studi di Padova, 21/04/2004.
- [2] Keigo Matsubara, Edison Kwok, Inge Rodriguez, Murali Paramasivam, *Developing and Porting C and C++ Applications on AIX*, Redbooks, IBM.
- [3] FASANO ANDREA, *Programmazione grafica 3D - Dalle basi alla costruzione di un motore 3D*, Edizioni Infomedia, pp. 75-90.