ESERCITAZIONE

Codice parallelo per il prodotto matrice-vettore

1

2

```
#include <stdio.h>
#include "mpi.h"
int main(int argc, char *argv[])
{
  int menum, nproc;
  int n,m; //dimensioni della matrice
  int ncol, mcol; //dimensioni delle sottomatrici
  int col, righe; //dimensioni della griglia
  int coord[2];
  int i,j;
  float *x,*A,*b;
  float *xloc, *bloc, *Aloc, *sumbloc;
  …
  MPI_Status status;
  MPI_Comm *griglia, *grigliar, *grigliac;
  …
  MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &menum);
  MPI_Comm_size(MPI_COMM_WORLD, &nproc);
```


3

```
#include <stdio.h>
#include "mpi.h"
int main(int argc, char *argv[])
{
  int menum, nproc;
  int n,m; //dimensioni della matrice
  int ncol, mcol; //dimensioni delle sottomatrici
  int col, righe; //dimensioni della griglia
  int coord[2];
  int i,j;
  float *x,*A,*b;
  float *xloc, *bloc, *Aloc, *sumbloc;
  …
  MPI_Status status;
  MPI_Comm *griglia, *grigliar, *grigliac;
  …
  MPI_Init(&argc, &argv);
  MPI_Comm_rank(MPI_COMM_WORLD, &menum);
  MPI_Comm_size(MPI_COMM_WORLD, &nproc);
```

Codice puramente indicativo!! Non scritto per funzionare né per essere copiato e funzionare!

4

```
tag = 1;
if(menum==0)
{
     printf("Inserire il numero di righe della matrice:\n");
     scanf("%d",&m);
     printf("Inserire il numero di colonne della matrice:\n");
     scanf("%d",&n);
}
MPI_Bcast(&m,1,MPI_INT, 0, MPI_COMM_WORLD);
MPI_Bcast(&n,1,MPI_INT, 0, MPI_COMM_WORLD);
…
ASSEGNAZIONE col E righe (dimensioni griglia)
…
MPI_Bcast(&col,1,MPI_INT, 0, MPI_COMM_WORLD);
MPI_Bcast(&righe,1,MPI_INT, 0, MPI_COMM_WORLD);
…
ALLOCAZIONI MEMORIA NECESSARIE
…
```


5

```
…
/*CREAZIONE DELLA GRIGLIA DI PROCESSORI*/
crea_griglia(griglia, grigliar, grigliac, menum, nproc,
             righe,col, coord);
```

```
void crea_griglia (MPI_Comm *griglia,
                   MPI_Comm *grigliar,
                   MPI_Comm *grigliac,
                   int menum, int nproc,
                   int riga, int col, int *coordinate)
```

```
…
mloc = m/righe;
nloc = n/col;
…
DISTRIBUZIONE DEI DATI TRA I PROCESSORI
…
```


6

```
…
/*Fase di calcolo*/
for (i=0;i<mloc;i++){
     for (j=0;j<nloc;j++){
           bloc[i] += Aloc[i*nloc+j] * xloc[j];
     }
}
…
…
…
```

```
for (i=0;i<mloc;i++){
          for (j=0;j<nloc;j++){
                bloc[i]+= aloc[i][j] * xloc[j];
           }
}
```


7

```
…
/*Fase di comunicazione e raccolta del risultato*/

MPI_Allreduce(bloc,
              sumbloc,
              mloc,
              MPI_FLOAT,
              MPI_SUM,
              grigliar); //somma sulle righe

MPI_Allgather(sumbloc,
              mloc,
              MPI_FLOAT,
              b,mloc,
              MPI_FLOAT,
              grgliac); //collezione sulle colonne
…
```

```
int MPI_Allreduce(void *sendbuf,
                  void *recvbuf,
                  int count,
                  MPI_Datatype datatype,
                  MPI_Op op,
                  MPI_Comm comm)
```

```
int MPI_Allgather(void *sendbuf,
                  int sendcount,
                  MPI_Datatype sendtype,
                  void *recvbuf,
                  int recvcount,
                  MPI_Datatype recvtype,
                  MPI_Comm comm)
```


8

```
…
/*Output*/
if(menum==0){
          for(i=0;i<m;i++)
                    printf("b(%d) = %f",i,b[i]);
}

…

MPI_Finalize();
return 0;
}
```


9

Esercizi

Scrivere, compilare ed eseguire sul cluster attraverso il PBS i seguenti esercizi. Inviare il codice, il PBS e uno o più esempi di output a valeria.mele@unina.it

1. Sviluppare un algoritmo per il calcolo del **prodotto matrice-vettore**, in ambiente di calcolo parallelo su architettura MIMD a memoria distribuita, che utilizzi la libreria MPI.
   La matrice $A \in \Re^{mxn}$ deve essere distribuita a pxq processi, disposti secondo una topologia a griglia bidimensionale.
   L'algoritmo deve essere organizzato in modo da costruire e utilizzare una griglia bidimensionale dei processi.

   - Utilizzare quanto sviluppato per gli esercizi precedenti!

10

Esercizi

Scrivere, compilare ed eseguire sul cluster attraverso il PBS i seguenti esercizi. Inviare il codice, il PBS e uno o più esempi di output a valeria.mele@unina.it

1. Sviluppare un algoritmo per il calcolo del **prodotto matrice-vettore**, in ambiente di calcolo parallelo su architettura MIMD a memoria distribuita, che utilizzi la libreria MPI.
   La matrice $A \in \Re^{mxn}$ deve essere distribuita a *pxq* processi, disposti secondo una topologia a griglia bidimensionale.
   L'algoritmo deve essere organizzato in modo da costruire e utilizzare una griglia bidimensionale dei processi.

   - Utilizzare quanto sviluppato per gli esercizi precedenti!

Entro la prossima settimana!!