Programmation parallèle

UFR SAT, CFPP - MaDSI 1

Travaux pratiques 11

————————————

Dans cet exemple, vous allez construire un système maître/esclave d'entrée/sortie. Cela vous permettra d'organiser relativement facilement différents types d'entrées et de sorties de votre programme

Les processus de MPI_COMM_WORLD sont divisés en deux ensembles: le maître (qui exécute toutes les E/S) et les esclaves (qui effectuent toutes leurs E/S en contactant le maître). Les esclaves feront aussi tout autre calcul qu'ils pourraient désirer; par exemple implémenter l'itération Jacobi.

Pour le premier exercice, les processus sont divisés en deux communicateurs, l'un étant le maître et l'autre les esclaves. Le maître doit accepter les messages des esclaves (de type MPI_CHAR) et les imprimer par ordre de rang (c'est-à-dire d'abord l'esclave 0, puis l'esclave 1, etc.). Les esclaves doivent chacun envoyer 2 messages au maître.

Hello from slave i

Goodbye from slave i

(avec **i** la valeur appropriée pour chaque esclave). On suppose une longueur maximale de message de 256 caractères.

Les routines MPI utilisées sont:

MPI_Comm_split MPI_Send MPI_Recv

//————————————————————DEBUT————————————————

```
#include <stdio.h>
#include "mpi.h"

int main( argc, argv )
int argc;
char **argv;
{
    int rank, size;
    MPI_Comm new_comm;
```

```
    MPI_Init( &argc, &argv );
    MPI_Comm_rank( MPI_COMM_WORLD, &rank );
    MPI_Comm_split( MPI_COMM_WORLD, rank == 0, 0, &new_comm );
    if (rank == 0)
     master_io( MPI_COMM_WORLD, new_comm );
    else
     slave_io( MPI_COMM_WORLD, new_comm );

    MPI_Finalize( );
    return 0;
}

/* This is the master */
int master_io( master_comm, comm )
MPI_Comm comm;
{
    int        i,j, size;
    char       buf[256];
    MPI_Status status;

    MPI_Comm_size( master_comm, &size );
    for (j=1; j<=2; j++) {
     for (i=1; i<size; i++) {
         MPI_Recv( buf, 256, MPI_CHAR, i, 0, master_comm,
&status );
         fputs( buf, stdout );
     }
    }
}

/* This is the slave */
int slave_io( master_comm, comm )
MPI_Comm comm;
{
    char buf[256];
    int  rank;

    MPI_Comm_rank( comm, &rank );
    sprintf( buf, "Hello from slave %d\n", rank );
    MPI_Send( buf, strlen(buf) + 1, MPI_CHAR, 0, 0, master_comm );

    sprintf( buf, "Goodbye from slave %d\n", rank );
    MPI_Send( buf, strlen(buf) + 1, MPI_CHAR, 0, 0, master_comm );

    return 0;
}
//-------------------FIN -------------------
```

_ _ _ _ _ _

Questions

1. Quelles sont les fonctions utilisées dans ce programme pour diffuser les sous-tableaux vers les autres processus Esclaves?
2. Exécuter plusieurs fois le code en changeant progressivement le nombre de processus et le nombre d'entiers du tableau.
3. Observer les résultats de la question précédente, et interpréter les résultats obtenus par rapport au temps d'exécution.