Programmation parallèle

UFR SAT, CFPP - MaDSI 1

Travaux pratiques 10

Ce programme permet de recevoir correctement les messages de tous les processus d'envoi. Le programme est organisé pour que tous les processus, à l'exception du processus 0, envoient 100 messages au processus 0. Le processus 0 imprime les messages au fur et à mesure de leur réception. La mise en œuvre du MPI est-elle juste?

Vous aller utiliser les routines MPI suivantes:

MPI_Waitsome MPI_Irecv MPI_Cancel

```
#define large 128
#include "mpi.h"
#include <stdio.h>
int main(argc, argv)
int argc;
char **argv;
{
    int rank, size, i, sbuf = 1, cnt;

    MPI_Init( &argc, &argv );
    MPI_Comm_rank( MPI_COMM_WORLD, &rank );
    MPI_Comm_size( MPI_COMM_WORLD, &size );
     if (rank == 0) {
      MPI_Request requests[large];
      MPI_Status  statuses[large];
      int         indices[large];
      int         buf[large];
      int         j, ndone;

      cnt = (size-1)*100;
      for (i=1; i<size; i++)
          MPI_Irecv( buf+i, 1, MPI_INT, i,
                MPI_ANY_TAG, MPI_COMM_WORLD, &requests[i-1] );
      while(cnt > 0) {
          MPI_Waitsome( size-1, requests, &ndone, indices, statuses );
          for (i=0; i<ndone; i++) {
           j = indices[i];
           printf( "Message du processus %d avec le tag %d\n",
```

```
                statuses[i].MPI_SOURCE,
                statuses[i].MPI_TAG );
            MPI_Irecv( buf+j, 1, MPI_INT, j+1,
                    MPI_ANY_TAG, MPI_COMM_WORLD, &requests[j] );
          }
          cnt -= ndone;
      }
      /* Cancelling the pending receives */
      for (i=0; i<size-1; i++)
          MPI_Cancel( &requests[i] );
    }
    else {
      for (i=0; i<100; i++)
          MPI_Send( &sbuf, 1, MPI_INT, 0, i, MPI_COMM_WORLD );
    }
    MPI_Finalize();
    return 0;
}
```

Sortie

```
www:parallel babacardiop$ mpirun -np 7 ./tag2
Message du processus 1 avec le tag 0
Message du processus 2 avec le tag 0
Message du processus 3 avec le tag 0
Message du processus 4 avec le tag 0
Message du processus 5 avec le tag 0
Message du processus 6 avec le tag 0
Message du processus 1 avec le tag 1
Message du processus 1 avec le tag 2
Message du processus 1 avec le tag 3
Message du processus 1 avec le tag 4
Message du processus 1 avec le tag 5
Message du processus 1 avec le tag 6
Message du processus 1 avec le tag 7
Message du processus 1 avec le tag 8
Message du processus 1 avec le tag 9
Message du processus 1 avec le tag 1
Message du processus 1 avec le tag 11
. . .
```

Questions

1. Quelles sont les routines MPI utilisées dans ce TP ?
2. Que représentent **MPI_Waitsome**, **MPI_Irecv** et **MPI_Cancel** ?
3. Commenter la sortie obtenue à l'écran ?
4. La mise en œuvre du MPI est-elle correcte ?