Paradigmas de Computação Paralela

20-October-2015

Introduction to MPI

These exercises aim to introduce the basic concept of MPI programing.

Compile and execute the following MPI program (mpicc + mpirun –np 2 a.out):

```
#include <mpi.h>
#include <stdio.h>
int main( int argc, char *argv[]) {
      int rank, msg;
      MPI_Status status;
      MPI_Init(&argc, &argv);
      MPI_Comm_rank( MPI_COMM_WORLD, &rank );
      /* Process 0 sends and Process 1 receives */
      if (rank == 0) {
                  msg = 123456;
                  MPI_Send( &msg, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
      }
      else if (rank == 1) {
                  MPI_Recv( &msg, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, &status );
                  printf( “Received %d\n”, msg);
      }
      MPI_Finalize();
      return 0;
}
```

1) Change the program to implement several types of pipelines:
   a) 4 processes each process receiving 1 message that is processed by every process in the pipeline
   b) same as a) but the number of processes is given in the command line (np parameter).
   c) Same as b) but the pipeline should process n messages

2) Change the program to implement a construct like “work sharing”. A master process has a set of tasks (e.g., numbers) to process. Each worker gets a task (message) to process, sets the contents of the message to its rank and returns the message to the master.
   a) Static scheduling: the number of messages to process is equal to the number of processes
   b) Dynamic scheduling: the number of messages is 10x the number of processes; faster processes will get more messages

3) Change the previous program (2. a) to broadcast a message to every worker and to reduce to a single value a message from each worker containing its rank.