Programming Assignment 5 - Message Passing Interface

Early Due Date (+5%): Wed. 12/2, 11:59pm
Final Due Date: Fri. 12/4, 11:59pm

Background

The concept of "distributed computing" has had remarkable impact upon the computing industry. The motivation to achieve grand orders of scalability by moving application computing outside the bounds of a single computer (or "box") served as motivating factors for the development of both Linux and the internet. MPI provides a portable API that allows both data communications and synchronization events across heterogenous platforms connected via a network.

The purpose of this assignment is to implement a basic MPI program to measure the communcations bandwidth available between nodes on the Owens supercomputer while also observing the effect of changes to the size of the data portion of the message.

Assignment

Determine the time taken and bandwidth achieved while performing repeated point-to-point communication transfers of buffers containing double-precision floating-point numbers between a pair of processors.

Structure the communications section of your program as follows:

process 0

```
barrier
start timer
repeat iterations times
{
   send(A, process1)
   receive(B, process1)
   send(B, process1)
   receive(A, process1)
}
barrier
stop timer
time = totaltime/(4*iterations)
bandwidth = sizeof(double)*msg_size/time
```

process 1

```
barrier
start timer
repeat iterations times
{
   receive(A, process0)
   send(A, process0)
   receive(B, process0)
   send(B, process0)
}
barrier
stop timer
time = totaltime/(4*iterations)
bandwidth = sizeof(double)*msg_size/time
```

Where A and B represent appropriately sized messages composed of arrays of doubles. Report the bandwidth achieved over 1,000,000 iterations for each of the message sizes (number of doubles) 32, 256, 512, 1024 and 2048 when using 2 dedicated nodes on Owens.

Implement your ping-pong code using standard MPI blocking primitives, MPI_Send and MPI_Recv.

Name your executable "a.out" and use no command line parameters. Your program should complete all tests in a single invocation.

Input Data Format

No input nor output files are required for this lab.

Report

Your report should include:

- all requested bandwidth results
- an explanation for the difference in the code (above) for process 0 and process 1.

Testing and Submission

Follow the testing and submission guidelines for previous labs, using assignment “lab5.”’

- Before submitting ensure your executable program can be created with a single invocation of "make".
- Ensure the *mvapich2/2.3.3* modue is loaded prior to compiling or running your programs.
- The MPI compiler is *mpicc* and uses the same options as the gcc C compiler. Compile your programs with optimizer level3 (-O3) option.
- Create a directory "cse5441_lab5". Within this directory, place:
  - all program files (e.g. .c and .h files);
  - makefile
  - your report in .pdf format.
- As before, use the OSC “submit” command to turn-in your work. The submit assignment name is “lab5.”