Programming Assignment III: MPI Programming

The purpose of this assignment is to familiarize yourself with MPI programming.

1 Statement of Problem 1

In this problem, you need to use MPI to parallelize the following serial program (`http://www.cs.nctu.edu.tw/~ypyou/courses/PP-f18/assignments/HW3/prime.c`), which takes a `long long int` argument as an input, finds the largest prime number that is smaller than the input, and counts the prime numbers that are smaller than the input.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int isprime(int n) {
  int i,squareroot;
  if (n>10) {
    squareroot = (int) sqrt(n);
    for (i=3; i<=squareroot; i=i+2)
      if ((n%i)==0)
        return 0;
    return 1;
  }
  else
    return 0;
}

int main(int argc, char *argv[])
{
  int pc,       /* prime counter */
      foundone; /* most recent prime found */
  long long int n, limit;

  sscanf(argv[1],"%llu",&limit);
  printf("Starting. Numbers to be scanned= %lld\n",limit);

  pc=4;     /* Assume (2,3,5,7) are counted here */

  for (n=11; n<=limit; n=n+2) {
    if (isprime(n)) {
      pc++;
      foundone = n;
    }
  }
  printf("Done. Largest prime is %d Total primes %d\n",foundone,
     pc);

  return 0;
}
```

2 Statement of Problem 2

In this problem, you need to use MPI to parallelize the following serial program (`http://www.cs.nctu.edu.tw/~ypyou/courses/PP-f18/assignments/HW3/integrate.c`), which integrates function sin(X) over the range from 0 to π using `N` intervals, where `N` is an argument of the program.

```
#include <stdio.h>
#include <math.h>

#define PI 3.1415926535

int main(int argc, char **argv)
{
  long long i, num_intervals;
  double rect_width, area, sum, x_middle;

  sscanf(argv[1],"%llu",&num_intervals);

  rect_width = PI / num_intervals;

  sum = 0;
  for(i = 1; i < num_intervals + 1; i++) {

    /* find the middle of the interval on the X-axis. */

    x_middle = (i - 0.5) * rect_width;
    area = sin(x_middle) * rect_width;
    sum = sum + area;
  }

  printf("The total area is: %f\n", (float)sum);

  return 0;
}
```

3 Requirement

- Your submitted solution contains two source files: `prime.c` (or `prime.cpp`) for problem 1 and `integrate.c` (or `integrate.cpp`) for problem 2.
- Your programs take one command-line argument.
- You should not modify the output format.

4 Developing and Execution Environment of MPI Programs

4.1 Using the NCTU CS virtual cluster

4.1.1 Login information

IP of the master workstation: pp1.cs.nctu.edu.tw
User name & password: The same account and password as NCTU CSCC account (`https://www.cs.nctu.edu.tw/cscc/account/`)
For more information, please refer to NCTU CS Computer Center (`https://www.cs.nctu.edu.tw/cchonor/`)

4.1.2 You can use SSH to log in to other machines

IP
pp1.cs.nctu.edu.tw
pp2.cs.nctu.edu.tw
pp3.cs.nctu.edu.tw
pp4.cs.nctu.edu.tw

Try the command below to see whether the command exist or the PATH variable is properly set.

```
$mpiexec --version
or
$echo "$PATH"
```

4.2 Executing jobs on other machines without entering a password

To make `mpiexec` work properly, you need to be able to execute jobs on remote nodes without typing a password. You will need to generate an ssh key by yourself.

You can also google “ssh passphrase” for details.

```
//Please create .ssh directory if not exists
user@host:~$ mkdir -p ~/.ssh
user@host:~$ ssh-keygen -t rsa
//Then you will be prompt to enter a passphrase, you can leave it
    empty for convenience.
user@host:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

```
//All nodes share the same file system, so you may done them on
    only the host node.
```

Next, log in to all machines at least one time to record them in `known_hosts` file.

If you were on pp1.cs.nctu.edu.tw, you can try the following command for logging in without password to see whether the configuration works.

```
$ssh pp2.cs.nctu.edu.tw
or
$ssh pp3.cs.nctu.edu.tw
or
$ssh pp4.cs.nctu.edu.tw
```

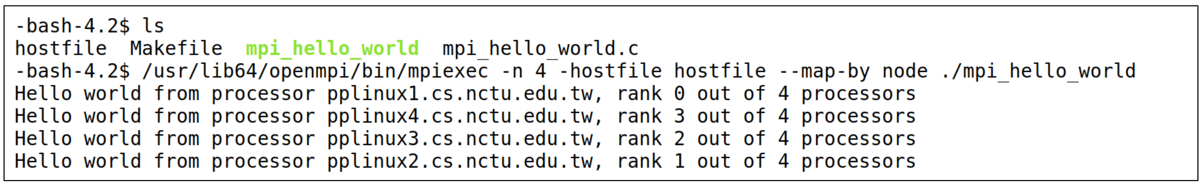
4.2.1 Record machines with host file

An MPI host file records all available machines with its ability. We ignore the `slots` setting here.

```
// hostfile
pp1.cs.nctu.edu.tw
pp2.cs.nctu.edu.tw
pp3.cs.nctu.edu.tw
pp4.cs.nctu.edu.tw
```

4.2.2 Writing your program and running

To run program with MPI, the executable file should exist in the same path on each machines. The NCTU CSCC servers have NFS and NIS system, so you don't need to copy file by yourself. You will see the same file system on the four servers.



```
-bash-4.2$ ls
hostfile  Makefile  mpi_hello_world  mpi_hello_world.c
-bash-4.2$ /usr/lib64/openmpi/bin/mpiexec -n 4 -hostfile hostfile --map-by node ./mpi_hello_world
Hello world from processor pplinux1.cs.nctu.edu.tw, rank 0 out of 4 processors
Hello world from processor pplinux4.cs.nctu.edu.tw, rank 3 out of 4 processors
Hello world from processor pplinux3.cs.nctu.edu.tw, rank 2 out of 4 processors
Hello world from processor pplinux2.cs.nctu.edu.tw, rank 1 out of 4 processors
```

4.3 Compiler and mpiexec flag for the homework

You may need to use `mpicc` with the `-lm` flag to link properly. Here is an example for how to compile and run an MPI program (taking "integrate" as an example):

```
$mpicc -o integrate integrate.c -lm
$mpiexec -n 4 -hostfile hostfile ./integrate 123456
```

5 Submission

Be sure to upload your zipped source codes, which includes no folder, to e-Campus system by the due date and name your file as "HW3_xxxxxxx.zip", where xxxxxxx is your student ID.
Due Date: 23:59, November 23, Friday, 2018

6 References

- https://computing.llnl.gov/tutorials/mpi/