MPI Programming Lab

About job submission

Jobscript template: *jobscript.sh*
Use your name as *jobname*
Set *-n* in *aprun* statement as appropriate
All output to stdout and stderr from aprun will be in *screen.out*
Submit job (specify queue name for today):

```
$ qsub –q QUEUENAME jobscript.sh
```

MPI Exercises

MPI function syntax available on mpich.org - search MPI_function_name on the web

01-1-hello-world.c
Insert a print statement (ex. hello world) and make it a Hello World C/Fortran program
Compile

```
$ cc 01–1–hello–world.c
```

Submit a job using *jobscript.sh*, with ONE process

```
$ qsub jobscript.sh
```

Check output in *screen.out*

01-2-hello-world.c
Turn your Hello World program, or this one, into an MPI Hello World program
Look for comments and insert appropriate statements

```
MPI_Init(&argc, &argv);
MPI_Finalize();
```

Compile
Submit a job using *jobscript.sh*, with *n* processes (*n <=24*. Your choice)
Check output in *screen.out*

02-hello-world-myrank-size.c
Develop your Hello World MPI program to print MPI Rank and MPI size values
Compile
Submit a job using *jobscript.sh*, with *n* processes (*n <=24*. Your choice)
Check output in *screen.out*

Broadcast
Develop your hello world program to set
sendval = 10 only on Rank 0
Broadcast sendval to all ranks
Print sendval from all ranks, along with myrank.

Run the code with n = 4 processes.

03-prime-serial.c
This code checks if a given number is prime number.
Understand the serial code.
Parallelize with MPI.
Obtain an unambiguous output from parallel code. (Hint: Use some communication).
How is the load balance for an arbitrary test number? Discuss ways to improve.

MPI_Reduce
Write a program to initialize a variable *var* to different integers in different ranks.
Obtain the sum of all such values in *Rank 0* using *MPI_Reduce* and print.
Run the code for *n < 24* and verify the output.

Obtain size of MPI_COMM_WORLD.
Use your hello-world-myrank-size program to obtain the size of MPI_COMM_WORLD, **without** using the MPI function MPI_Comm_size
Run the code for *n < 24* and verify the output.

mpi_hello world two ranks
Write an MPI Hello world program for two ranks.
Print different strings from different ranks.
For example,
“Hello world from Rank 0” from Rank 0
“A separate Hello from Rank 1” from Rank 1
Run the code for *n =2* and verify the output.

MPI_Send and MPI_recv
Use the code from above exercise and set *sendval = 10, recvval = 30*
Send the value *sendval* from Rank 1 and receive it in Rank 0 to *recvval*
Print *recvval* before and after the MPI function calls.
Run the code for *n = 2* and *n > 2.*
Discuss the output

MPI_Sendrecv - circular
Write a program to implement MPI_Sendrecv
a.
set sendval = 1, recvval = 10.
From Rank 0, send sendval to Rank 1 variable recvval
From Rank 1, send sendval to Rank 2 variable recvval and so on.
Rank 9 should send to Rank 0.

b.
Modify the program so that Rank 1 sends *"the value it received"* to Rank 2 and so on
Is it possible to achieve it using Sendrecv?

Calculate the value of Pi
pi_mpi.c is a serial code to calculate the value of pi.
Parallelize the code with MPI. Look for comments and commented lines and update as appropriate.
Compare the parallel result with serial result.

AlltoAll
Examples of MPI_AlltoAll and AlltoAllv are provided in **/mnt/lustre/serc3/secguest*/2020Jan-MPI/Day-2**