Programming Project #1

CS4379: Parallel and Concurrent Programming
CS5379: Parallel Processing
Spring 2020

Due 2/27 Thur., 12:30 p.m. Please submit a soft copy on Blackboard. Late submissions are accepted till 3/5, Thur., 12:30 p.m. with 10% penalty each day. No submissions accepted after that.

Please submit a single tarball/zipped file containing all your compiled programs, output files, the modified mpi.sh file, and your job's output/error files. Please name your submission file starting as "LastName_FirstName_PP1".

Please note that we may request an in-person, 5-10 mins quick demo for grading.

Job Submission on HPCC Clusters

Q1 (50%). In this part of the project, you are asked to experiment compiling given MPI, Pthreads, and OpenMP programs with different available compilers on HPCC clusters:

1. First, login to Quanah cluster and start an interactive session by using "`qlogin`" command and request for following resources:
   a. Queue: `omni`
   b. Project: `quanah`
   c. Parallel Environment: Shared Memory with 2 CPU cores

   Please keep in mind that due to the high usage of cluster resources, interactive sessions on Quanah are quite competitive to be established, and you may face several failures before you obtain your requested interactive job. Therefore, it is highly recommended to avoid postponing your programming projects to the very last days.

2. After the interactive session is established and your command prompt changed to "`compute-#-#`", then copy the MPI source code sample directory from the following location into your home directory:

   ```
   /lustre/work/examples/quanah/hello-world/
   ```

3. Inside the "mpi" directory there is a "`mpi_hello_world.c`" file which contains a sample code of MPI programming in C language. Compile the code by using `mpicc` command once with GNU and once with Intel version of OpenMPI and IntelMPI (IMPI) and name the executable outputs as following:

   ```
   $ mpicc -o output_name program_name.c
   ```

Compiler	**MPI Compiler**	**Output Name**
intel/18.0.3.222	• openmpi/1.10.7 • impi/2018.3.222	• mpi_hello_world-intel-ompi • mpi_hello_world-intel-impi
gnu7/7.3.0	• openmpi/1.10.7 • impi/2018.3.222	• mpi_hello_world-gnu-ompi • mpi_hello_world-gnu-impi

In this part of the project, it is important to have a good grasp of how to use "`module`" command along with `load`, `unload`, `swap`, and `purge` options in order to load a particular compiler into your environment.

4. Now try to execute any of the compiled outputs by calling `mpirun` command and using 2 CPU cores. (Please keep in mind that for each executable file, the right modules should be loaded into your environment.):

```
$ module load intel openmpi
$ mpirun -np 2 ./mpi_hello_world-gnu-impi
```

5. Inside the `hello-world` directory you will see `openmp` and `pthreads` directories as well. Try to compile the C codes inside these directories by using **Intel** compiler only:

   a. Please make sure `intel/18.0.3.222` module is already loaded.
   b. Intel uses `icc`, `icpc`, and `ifort` to compile C, C++, and Fortran codes.
   c. For intel compilers, `-lpthreads` and `-qopenmp` flags should be used in order to compile pthreads and OpenMP programs.
   d. At the end, try to run your codes and see if the output looks good.

*** Once all your programs executed correctly with no error **exit** the interactive session.

Q2 (50%). In this part of the project, you are asked to submit a job to Quanah cluster by requesting for specific resources, then executing one of your compiled `mpi_hello_world` program on one of the Quanah cluster machines and observe the content of the job's output file:

1. Make sure you've already exited the interactive session and your module list is empty. (you may use "`module purge`" to remove all modules from your environment)

2. Open the "`mpi.sh`" file which is located inside the `mpi` directory on your home directory. (You can use vi or any available text editor on Linux for this purpose). Modify the `mpi.sh` file and request for following resources from UGE job scheduler:

   a. Job Name: *your_name*_MPI_Test
   b. Queue: omni
   c. Project: quanah

d. Memory: 2G per core
e. Run time: 5 mins (hh:mm:ss)

3. Choose one of your compiled `mpi_hello_world` programs and modify the mpi.sh file to load the corresponding compiler modules for your program. Then, make sure the `mpirun` command is correctly set to execute your chosen `mpi_hello_world`.

   (Do not change the `--machinefile machinefile.$JOB_ID -np $NSLOTS` flags for mpirun)

4. Submit you job script by using "`qsub`" command and try to observe the status of your job by using "`qstat`". Once the job finished, two files will be created for the output and errors of your job. Make sure there is no error in the $JOB_NAME.e$JOB_ID file. If so, try to debug your error. Then check the $JOB_NAME.o$JOB_ID output file. You should see your job's output similar to the following:

```
Hello world from processor compute-5-22, rank 0 out of 18 processors
Hello world from processor compute-5-22, rank 1 out of 18 processors
Hello world from processor compute-5-22, rank 2 out of 18 processors
Hello world from processor compute-5-22, rank 3 out of 18 processors
Hello world from processor compute-5-22, rank 4 out of 18 processors
Hello world from processor compute-5-22, rank 5 out of 18 processors
```

   The order of ranks and compute-#-# may vary in your output file, which is totally fine.

~~Please make a single tarball/zipped file from your `hello-world` directory and include all compiled codes, the modified `mpi.sh` file and two job's output files inside the `mpi` directory, then submit your project.~~

Grading Criteria:

Please note that we may request an in-person, 5-10 mins quick demo for grading.

Q1	**Percentage %**	**Criteria**
50%	50	Executables of MPI, Pthreads, and OpenMP versions of hello-world programs, which are compiled correctly with Intel/GNU version of OpenMPI/IMPI compilers. (This part requires correct modules to be loaded before compilation)
	50	Correct execution and output files of MPI, Pthreads, and OpenMP versions of hello-world programs.
Q2	**Percentage %**	**Criteria**
50%	50	Correct mpi.sh file including all resource allocation requests that are asked for in Q2-2.

	50	Correct $JOB_NAME.o$JOB_ID file contains similar output lines as what was mentioned in Q2-4.

Note on cheating: We have zero-tolerance policy for cheating. Working in groups is fine for discussing approaches and techniques. Copying problem solutions or code is cheating. Both the person copying and the person giving the answers will be equally penalized. Make sure you do your own work.