CITS3402/5507

Getting started

The purpose of this week's labsheet is to familiarise yourself with developing and running MPI programs on Setonix.

Start by making sure you can execute MPI code running across multiple nodes on Setonix. Here is a small example of MPI code that uses `gethostname(2)` and MPI to print the process number followed by the hostname:

```
#include <string.h>
#include <unistd.h>
#include <stdio.h>
#include <mpi.h>

int main(int argc, char **argv) {
    int rank, size;
    char hostname[256];

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    gethostname(hostname, 255);

    if (rank == 0) {
        printf("%d: %s\n", 0, hostname);
        for (int i = 1; i < size; i++) {
            MPI_Recv(&hostname, 256, MPI_CHAR, i, 99, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
            printf("%d: %s\n", i, hostname);
        }
    } else {
        MPI_Send(&hostname, strlen(hostname)+1, MPI_CHAR, 0, 99, MPI_COMM_WORLD);
    }

    MPI_Finalize();
}
```

Create a new folder in your `/scratch/courses0101/<username>/` and save the above code as `mpi_hostname.c` in your newly created directory.

We have 4 nodes available on Setonix to run our program. Here is a short SLURM script that complies and runs our code on 4 nodes:

```
#!/bin/bash
#SBATCH --nodes=4
#SBATCH --partition=work
#SBATCH --account=courses0101
#SBATCH --mem=4G
#SBATCH --time=00:05:00

cc mpi_hostname.c -o mpi_hostname
srun ./mpi_hostname
```

NOTE: In practice, Setonix may allow you to create jobs using more than 4 nodes. Please do not use more than 4 nodes at a time.

Save the above script as `mpi_hostname.slurm` in your newly created directory, and submit your job to the queue.

```
$ sbatch mpi_hostname.slurm
Submitted batch job 15942735

$ squeue --me
JOBID        USER ACCOUNT                    NAME EXEC_HOST ST     REASON START_TIME ...
15942735 mszikssza courses0101     mpi_hostname.s       n/a PD   Priority N/A
```

The output of your job should look something like this:

```
$ cat slurm-15942735.out
0: nid002796
1: nid002801
2: nid002820
3: nid002822
```

Veryify that all four processes have different hostnames. This confirms our program is being executed accross different nodes.

Tasks

1. Go through the above example, and verify you are able to execute MPI code accross 4 different nodes on Setonix.
2. Extend the above example to also use multiple threads via OpenMP, in addition to multiple processes using MPI. How many physical cores can you now use?
3. Take a look at Exercise 1 from the Lawrence Livermore National Laboratory (LLNL) HPC Tutorial: https://hpc-tutorials.llnl.gov/mpi/exercise_1/
   - Download the serial C code for calculating π from the LLNL website with:
     - `$ wget https://hpc-tutorials.llnl.gov/mpi/examples/ser_pi_calc.c`
   - Parallelise this code using `MPI_Send` and `MPI_Recv`.
   - Once finished, download the solution and compare the performance of your code to the LLNL implementation:
     - `$ wget https://hpc-tutorials.llnl.gov/mpi/examples/mpi_pi_send.c`