# Programming Assignment III: MPI Programming

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

## 1 Problem Statement

In this problem, you need to use MPI to parallelize the following serial program (`http://www.cs.nctu.edu.tw/~ypyou/courses/PP-s19/assignments/HW3/conduction.c`). The program takes two `int` arguments ($N$, which controls the size of an isolated 2D plate, and *seed*, which seeds the random number generator) from command line arguments as inputs and finds the temperature after heat balance, given random temperatures on the 2D plate. For convenience, $N$ is assumed to be divisible by the number of threads.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(int argc, char **argv) {
  int N, seed;
  N = atoi(argv[1]);
  seed = atoi(argv[2]);
  srand(seed);
  int temp[N][N];

  for (int i = 0; i < N; i++) {
    for (int j = 0; j < N; j++) {
      temp[i][j] = random() >> 3; // avoid overflow
    }
  }
  int count = 0, balance = 0;
  int next[N][N];
  while (!balance) {
    count++;
    balance = 1;
    for (int i = 0; i < N; i++) {
      for (int j = 0; j < N; j++) {
        int up = i - 1 < 0 ? 0 : i - 1;
        int down = i + 1 >= N ? i : i + 1;
        int left = j - 1 < 0 ? 0 : j - 1;
        int right = j + 1 >= N ? j : j + 1;
        next[i][j] = (temp[i][j] + temp[up][j] + temp[down][j] +
            temp[i][left] + temp[i][right]) / 5;
        if (next[i][j] != temp[i][j]) {
          balance = 0;
        }
      }
    }
    memcpy(temp, next, N * N * sizeof(int));
  }
```

```
  printf("Size: %d*%d, Seed: %d, ", N, N, seed);
  printf("Iteration: %d, Temp: %d\n", count, temp[0][0]);
  return 0;
}
```

# 2 Requirement

- Your submitted solution contains one source file: `conduction.c` (or `conduction.cpp`).
- Your program takes two command-line arguments.
- You should not modify the output format.

# 3 Developing and Execution Environment of MPI Programs

## 3.1 Using the NCTU CS virtual cluster

### 3.1.1 Login information

Your program will run on these four machines/nodes which have NFS and NIS services that make MPI easy to use.

| IP | Port | User Name | Password |
|---|---|---|---|
| 140.113.215.195 | 37106-37019 | pp<student ID> | <Provided by TA> |

### 3.1.2 SSH from one machine to the others

Handy hostnames corresponding to their private IP are provided in `/etc/hosts`.

| Hostname | Original Port |
|---|---|
| pp1 | 37106 |
| pp2 | 37107 |
| pp3 | 37108 |
| pp4 | 37109 |

For example, use the command below to log in to pp2 with the same user name.

```
$ ssh pp2
```

### 3.1.3 Executing jobs on other machines without entering a password

To make `mpi` 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.

```
$ mkdir -p ~/.ssh
$ ssh-keygen -t rsa
```

```
# Then you will be prompted to enter some infomation, you can leave them empty for
    convenience.
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
# Thanks to NFS, you only need to do it once.
```

Then, log in to all machines at least one time to make sure the configuration works and also record them in `.ssh/known_hosts` file.

```
# Log in without entering password
$ ssh pp1
$ ssh pp2
$ ssh pp3
$ ssh pp4
```

## 3.2 MPI usage

MPI is installed in `/home/PP-s19/MPI`. Check with the command below.

```
$ /home/PP-s19/MPI/bin/mpiexec --version
```

### 3.2.1 Recording machines with host file

An MPI host file records all available machines with its ability. Attribute `slots` indicates at most what number of threads on that machine will be used by MPI.

```
// hostfile
pp1 slots=4
pp2 slots=4
pp3 slots=4
pp4 slots=4
```

### 3.2.2 Compiling and running

You may get an MPI hello world program from `https://reurl.cc/ekGDW` and then compile and run the program on the four machines to make sure that MPI works well.

```
$ ls
hostfile mpi_hello_world.c
$ /home/PP-s19/MPI/bin/mpicc mpi_hello_world.c -o mpi_hello_world
$ /home/PP-s19/MPI/bin/mpiexec -npernode 1 --hostfile hostfile mpi_hello_world
Hello world from processor ec037-106, rank 0 out of 4 processors
Hello world from processor ec037-108, rank 2 out of 4 processors
Hello world from processor ec037-107, rank 1 out of 4 processors
Hello world from processor ec037-109, rank 3 out of 4 processors
```

Then, start to write your own program.

## 4 Submission

Please rename your `conduction.c` to `<your-student-id>.c` and upload it to e-Campus system by the due date.

**Due Date: 23:59, May 6 2019**

## 5 References


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