UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR TECHNISCHE INFORMATIK

Exercise Sheet 4

The Message Passing Interface (MPI)

Lecture *Parallel Computing Systems*, Winter semester 2024/2025

Dr. Javad Ghofrani

A discussion forum for the exercise can be found at: `moodle.uni-luebeck.de`.

In this assignment, a matrix multiplication and distributed search using OpenMPI will be implemented.

Choice of Environment

Students can work on your own virtual machine (Ubuntu 20.04 and 22.04 are recommended) or Google Colab.
If Google Colab is used, this is the trick to execute "mpirun" command.

```
!mpirun --allow-run-as-root --oversubscribe -np 2 ./hello_world
```

Requirements

1. Well explain your code in the exercise sheet or .ipynb file: what it does, expected and actual result. Pseudo code, diagram, flowchart,... anything that makes your ideas easier to understand.
2. Provide plots where applicable (accepted file formats for plots: png and jpg), or code snippets to create the plots on ipynb.
3. If any optimization is made, explain it well.
4. If personal virtual machine is used, each program is written as a single source code file (.c). Exercise sheet, source code files (.c) and build command file (.sh) should be packed into one .zip file.
5. If Google Colab is used, no need to prepare .zip file. After completing the assignment, choose "Edit -> Clear all outputs", then download it as .ipynb file to submit. The .ipynb file should be well explained and organized.
6. Please thoroughly review your submission. Late additional submissions due to corrupted .zip files or non-executable code in both .ipynb and .zip files, ... will not be accepted.

Preparation

In the bash environment of Ubuntu, install OpenMPI to run programs and compile the appropriate library:

```
sudo apt-get install openmpi-bin libopenmpi-dev
```

Hello World

In the lecture an MPMD-Hello-World program was presented. Use it to familiarize yourself with OpenMPI. Test the program on a computer and on distributed structures.

Note: Use the following commands to compile and execute your code (see also lecture slides):

```
mpic++ HelloWorld.cpp -o HelloWorld
mpirun HelloWorld
```

Ping-Pong

Develop an MPI program that measures the transit times between two MPI participants with a ping pong. For this, the master should send a message (Ping) and receive the content of the same in a reply (Pong).

Note: With the command *MPI_Wtime()* you can read out the current system time.

Collective Communication

Develop an MPI program that performs 2 Collective Communication: One-to-All and All-to-One.

Matrix Multiplication

Write an MPI program that works according to the SPMD principle and performs a matrix multiplication. To gain further acceleration through the multi-core systems, you can combine OpenMP and MPI.

Distributed Search

Write an MPI program that search for a given value in an array. Also try it on the worst and the best case. You can combine OpenMP and MPI.

Extension

Choose a topic of your interests which can be implemented in an MPI program. You can combine OpenMP and MPI.