Exercise Sheet 5 for Lecture Parallel Programming

Deadline: 2026-06-07, 23:59

Prof. Dr. Michael Kuhn (michael.kuhn@ovgu.de)
Michael Blesel (michael.blesel@ovgu.de)
Parallel Computing and I/O • Institute for Intelligent Cooperating Systems
Faculty of Computer Science • Otto von Guericke University Magdeburg
https://parcio.ovgu.de

We start again with our serial program for solving the Poisson equation and will again only concern ourself with the Jacobi method. We now want to parallelize it by using POSIX threads.

A tutorial for programming with POSIX threads can be found at:

https://hpc-tutorials.llnl.gov/posix/

1. Parallelization with POSIX Threads (300 Points)

Parallelize the Jacobi method from the serial program with POSIX threads. Again, the parallel version must give the exact same results as the serial one. The termination after iterations and precision must work correctly and deliver the same results and output as the serial version.

From a performance perspective, the parallelization should yield an acceptable speedup. Take a look at Task 2 for how to measure the performance.

The thread count shall be controllable with the already existing first argument of the `partdiff` application.

Please document how much time you needed for this task. How much of it was spent searching for errors?

2. Performance Analysis (120 Points)

Measure the performance of your program and visualize the runtimes for 1, 2, 3, 6, 12, 18 and 24 threads in a diagram. Also remember to compare you program with the original serial version. Use 4,096 interlines for these measurements. The shortest run should take at least 30 seconds. Choose the program arguments accordingly.

Repeat each measurement at least three times to get sensible averages. For this, you can use the `hyperfine` tool which can be loaded with `module load hyperfine`.

For `t` threads, `i` interlines and `n` iterations you can use the following command:

```
./partdiff t 2 i 2 n
```

The measurements should be done with SLURM on one of the compute nodes of the cluster. Document the hardware specifications used for the measurements (processor, core count, available main memory, etc.) and use the same compute node for each batch of measurements.

In the materials, you can find prepared SLURM job scripts within the `slurm` directory, which you can use for your measurements. To run them, your working directory should be the `slurm` directory from which you can start the scripts with `sbatch`:

```
$ cd PP-2026-Exercise-05-Materials
$ make -C pde
$ cd slurm
$ sbatch measurement1.slurm
```

Visualize all results in well-labeled diagrams. Write about a quarter page of interpretation for these results.

Submission

We will count your last commit on the `main` branch of your repository before the exercise deadline as your submission. In the root directory of the repository, we expect a `PP-2026-Exercise-05-Materials` directory with the following contents:

- A file `group.md` with your group members (one per line) in the following format:

  ```
  Erika Musterfrau <erika.musterfrau@example.com>

  Max Mustermann <max.mustermann@example.com>
  ```

- The modified code of the `partdiff` program in the `pde` directory. (Task 1)
- A `performance-analysis.pdf` document with the measured runtimes and your analysis (Task 2)