Durham University

Computer Science

Summative Assignment

Module code and title	COMP3467 Advanced Computer Systems
Academic year	2023-24
Coursework title	Parallel programming and system administration
Coursework credits	5
% of module's final mark	50%
Lecturer	Laura Morgenstern
Submission date*	Tuesday, November 21, 2023 14:00
Estimated hours of work	10
Submission method	Gradescope

Additional coursework files	*magic_matrix.cpp* *directory "data_sets"*
Required submission items and formats	*report_<CIS_username>.pdf* *magic_matrix_gpu.cpp* *Makefile* *run_all_magic_matrix.sh* *install_numactl.sh* *numactl.out*

* This is the deadline for all submissions except where an approved extension is in place.

Late submissions received within 5 working days of the deadline will be capped at 40%.
Late submissions received later than 5 days after the deadline will receive a mark of 0.
It is your responsibility to check that your submission has uploaded successfully and obtain a submission receipt.

Your work must be done by yourself (or your group, if there is an assigned groupwork component) and comply with the university rules about plagiarism and collusion. Students suspected of plagiarism, either of published or unpublished sources, including the work of other students, or of collusion will be dealt with according to University guidelines (https://www.dur.ac.uk/learningandteaching.handbook/6/2/4/).

Coursework: Parallel programming and system administration

Module: Advanced computer systems (COMP 3467)
Term: Michaelmas term, 2023
Lecturer: Laura Morgenstern[1]

Submission Please submit this assignment on Gradescope via GitHub Classroom. The steps to submit your coursework are the following.

1. Go to https://classroom.github.com/a/oetpwsgD and accept the assignment. This will create the repository `COMP3467/comp3467-<github_username>`, where `<github_username>` is the username of the GitHub account used to accept the assignment.
2. Add to the repository the file `report_<CIS_username>.pdf`. You can generate this file in any way you wish, but make sure it is a valid PDF document. You can check this by opening it from the GitHub interface. The file should contain only the information required by questions (1.1), (1.3), (2.2) and (2.4).
3. Add all source code, script and output files as specified below to the repository.
4. Follow the instructions in the file `README.md` of your repository to submit your work to Gradescope. Make sure that the latest version of your work is on GitHub—that is, do not forget to commit all of your work before submitting to Gradescope! To check the files currently in the repository, you can use the file explorer at

   https://github.com/COMP3467/comp3467-<github_username>

.

Deadlines Consult the learning and teaching handbook for submission deadlines.

Plagiarism and collusion Students suspected of plagiarism, either of published work or work from unpublished sources, including the work of other students, or of collusion will be dealt with according to departmental and university guidelines.

Preface A magic square is an $N \times N$ matrix of integers with the sums of the elements in each row, each column, the major diagonal (top-left to bottom-right) and the minor diagonal (top-right to bottom-left) being equal. You stumbled across a new algorithm to generate large magic squares from smaller ones [2]. To check whether the algorithm indeed generates magic squares, you implemented it along with a testing framework. Since the sequential implementation takes ages to generate and check large magic squares you aim to parallelize the code and set up a framework for automatic correctness testing and performance measurements on the department's GPU cluster NCC.

Question 1: GPU programming

Please use NCC to implement and test all subsequent tasks.

(1.1) Instrument `magic_matrix.cpp` with OpenMP's time measurement routines to identify the functions that require the most computation time. Neglect the time for I/O operations and memory allocation. State the computation time of each function in `report_<CIS_username>.pdf` in tabular form.

[12 marks]

(1.2) Use OpenMP to develop a GPU implementation of `magic_matrix.cpp`. Save your source code as `magic_matrix_gpu.cpp` and provide a `Makefile` that allows building your GPU

[1] laura.morgenstern@durham.ac.uk
[2] https://www.reddit.com/r/math/comments/b1ktr4/a_simple_method_to_make_large_magic_squares_from/

code via target `mmgpu`. Only implementations that maintain correctness under concurrent execution will receive marks.

[24 marks]

(1.3) In `report_<CIS_username>.pdf`, write a short text (200 words at most) that describes your approach to porting the code to GPUs and justifies the techniques used. Feel free to use plots and graphics as you see fit. In your report, you should:

- justify the applied directives and clauses for parallelization and memory management,
- justify any code restructuring and performance optimization techniques, and
- determine the runtime of your GPU code for an input matrix size of $N = 100$ and compare it to the runtime of the original, sequential CPU code for the same matrix.

[14 marks]

Question 2: Cluster computing

(2.1) Write an `sbatch` script `run_all_magic_matrix.sh` that allows for the parallel execution of `magic_matrix.cpp` on both data sets ($N = 3$ and $N = 10$) located in the directory `data_sets` on NCC.

[12 marks]

(2.2) In `report_<CIS_username>.pdf`, write a short text (200 words at most) that describes how you would parallelize `magic_matrix.cpp` with MPI. Provide a graphic of the data distribution you propose and discuss which MPI functions you would use. You do not need to implement your idea.

[16]

(2.3) `numactl` [3] is a commandline tool that comes in handy when analyzing the memory architecture of a compute node. For instance, to ensure data locality for processes or threads that operate on the same data. However, `numactl` is not necessarily installed on all clusters. Write a shell script `install_numactl.sh` that builds `numactl` from source and installs it locally in your home directory. Document the output of `numactl -H` when executed on the login-node in `numactl.out`.

[10 marks]

(2.4) In `report_<CIS_username>.pdf`, write a short text (200 words at most) that explains how you could make an installation of `magic_matrix.cpp` available to all users on the *PVM*, the virtual machine cluster setup during practicals. Discuss the advantages and disadvantages of your chosen approach.

[12]

[3] https://linux.die.net/man/8/numactl