CPSC 524 Assignment 4: CUDA Matrix Multiplication

Rami Pellumbi[*]

December 9, 2023

[*] M.S., Statistics & Data Science

1 Introduction

This assignment explores GPU programming using CUDA, centered around the task of matrix multiplication. The primary challenge involves constructing a CUDA kernel capable of multiplying two random rectangular matrices. This initial task serves as a gateway into the realms of parallel computing and efficient memory management, foundational elements in leveraging GPU architecture for computational tasks. As we progress, the report addresses more advanced techniques, such as the utilization of shared memory and the strategic optimization of thread computations, essential for enhancing the computational performance.

2 Project Organization

The project is laid out as follows:

```
|
├─ docs/
├─ bin/
├─ include/
├─ out/
├─ src/
├─ build-run-task1a.sh
├─ build-run-task1b.sh
└─ Makefile
```

- **docs/**: This folder contains LaTeX files and other documentation materials that pertain to the report.
- **bin/**: The `bin` folder holds compiled objects and executable files, centralizing the output of the compilation process.
- **include/**: Here, all the header files (`.h`) are stored.
- **out/**: The `out` folder stores the outputs from each task. It also houses the csv file containing data generated by the programs.
- **src/**: This directory houses the source files (`.c`) that make up the benchmarks.
- **Shell Scripts**: The shell scripts are used to submit the job for the relevant task to slurm via `sbatch`.

3 Code Explanation, Compilation, and Execution

This section outlines the steps required to build and execute the code. The provided Bash scripts automate the entire process, making it straightforward to compile and run the code. All the below steps assume you are in the root of the project directory.

3.1 Automated Building and Execution

All related code is in the `src/` directory. There are multiple programs:

- `task1.cu`
- `task2.cu`
- `task3.cu`
- `task4.cu`

To run any one expirement, execute the relevant bash script, e.g., `build-run-task1a.sh`. It should be noted that for the `build-run-task1b.sh` the `FP` must be defined to double on line 1.

3.2 Post-Build Objects and Executables

Upon successful compilation and linking, an `obj/` subdirectory will be generated within the directory. This directory will contain the compiled output files. Additionally, the executable files for running each program will be situated in the `bin/` subdirectory.

3.3 Output Files From `sbatch`

The output files generated from running the code by submitting the relevant Bash script via `sbatch` will be stored in the `out` directory.

4 Task 1: CUDA Matrix Multiplication

5 Task 2: CUDA Matrix Multiplication with Shared Memory

6 Task 3: Reducing Tile Loads

7 Task 4: Handling All Tile Sizes

8 Conclusion