CS6023: GPU Programming

Assignment 1

Deadline: February 11, 2024

1 Problem Statement

In the quiet town of Pallet, there lived a peculiar character named Professor Oak. Professor Oak was a renowned researcher in the field of Pokemon research. During one of his research journeys, he came across a scroll and four matrices. The scroll had three parts, each containing instructions on using these matrices - A, B, C and D - to reveal the final matrix.

Part 1: Finding the Hadamard Product
Excitement twinkled in Professor Oak's eyes as he set out to explore the first part of the matrix puzzle. The first part is to find the Hadamatrix, which is the Hadamard Product of A and $B^\intercal$.

Part 2: Finding the Weight Matrix
With the Hadamatrix in hand, Professor Oak turned his attention to matrix C. The second part was to craft the Weight Matrix, where each element of the weight matrix was the maximum of corresponding elements of C and the Hadamatrix.

Part 3: The Mystery Solved
The last matrix, D, was different from the rest of the matrices Oak had seen so far. While A, B and C were $N \times N$ matrices, D was a $2N \times 2N$ matrix. The challenge was to find the Hadamard Product of each quadrant of the matrix D with the Weight Matrix. Oak quickly realized from Figure 1 given in the scroll that D can be seen as 4 $N \times N$ matrices. The final result was a beautiful $2N \times 2N$ matrix.

Your task is to help Professor Oak solve the mystery of the scroll using CUDA.

2 General Guidelines

- Do not change any other existing pieces of code that are given in the starter code. Please add necessary memory allocation and *memcpy* operations

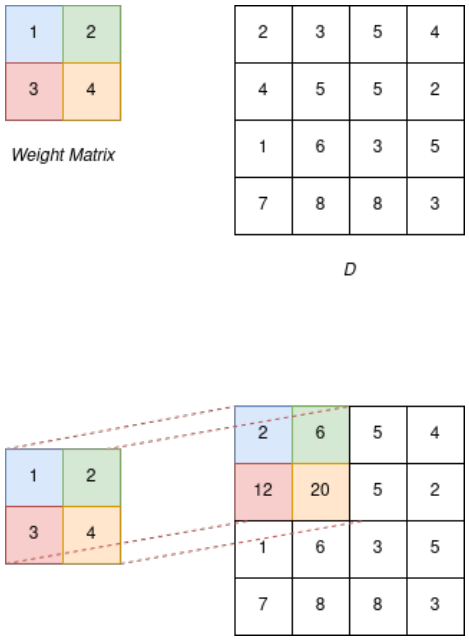

Figure 1: An image showing the Hadamard product of the weight matrix with one of the quadrants of D. This should be done for all four quadrants.

wherever necessary.

- Please use *cudaFree(void* devPtr)* wherever necessary. This will ensure memory doesn't run out, even for large test cases.
- We will time each of the kernel calls to ensure that your implementation is parallel.
- This assignment is an individual effort, and it is expected that all work submitted is your own. Plagiarism, the act of using someone else's work, is a serious academic offence and will not be tolerated. Any instances of plagiarism will result in severe consequences, including receiving a grade of zero for the assignment or even facing academic disciplinary actions.

3 Input and Output Format

3.1 Input Format

Each input is stored in a file. The first line will be the value of N. The next $3N$ lines will contain rows of the matrices A, B, and C, each having N integers. The final $2N$ lines will contain the rows of the matrix D, each having $2N$ elements.

3.2 Output Format

The final output from the device should be stored in the array d_D. This array will be copied to the host and written to a file. This is present in the starter code.

3.3 Constraints

- $2 < N < 2^{14}$
- Each element of all the input arrays will be less than 255.

3.4 Sample Input

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad B = \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 3 \\ 5 & 42 \end{bmatrix} \quad D = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 3 & 2 & 1 \end{bmatrix}$$

3.5 Sample Output

$$Output = \begin{bmatrix} 2 & 24 & 6 & 48 \\ 24 & 126 & 48 & 210 \\ 6 & 48 & 10 & 72 \\ 48 & 126 & 24 & 42 \end{bmatrix}$$

3.6 Testing

- You can use the tester program given along with the starter code to test your program.
- Run *python tester.py* to run the test script. It will show the number of test cases passed.
- You should place your *.cu* file in the *code* folder, given along with tester program.
- In case you are not a Linux user, you may have to change certain function calls in the tester program.

4 Learning Objectives

- To use CUDA Memory Management APIs.
- To learn the usage of multiple kernel configurations.
- To properly utilize the available memory resources.

5 Submission Guidelines

- Fill in your code in the *starter.cu* file. Rename it as your *ROLLNO.cu*. For example, if your roll number is *CS22M056*, your file should be named as *CS22M056.cu*.
- Zip this file as *ROLLNO.zip*. Submit on Moodle.

6 Learning Suggestions (Ungraded)

- Implement the same question serially and run it on the CPU. Compare the time taken to execute the serial and CUDA versions of your program on small and large inputs.
- There are three kernels in the code you implemented. Will there be any significant gain in time if these kernels were merged into a single kernel?
- We saw the usage of three different kernel configurations. Is there any performance gain from using one over the other for the same kernel?
- Is there a performance drop when using column-major access compared to using row-major access on GPUs?

[0]The story of Professor Oak was made with a little help from ChatGPT.