CS6023: GPU Programming

Assignment-1 (10 marks)

Deadline: February 12th, 2023, 23:55 on Moodle

1 Problem Statement

Write three separate CUDA C++ kernels to compute the Kronecker Product of matrices (**A** and $\mathbf{B}^T$) to produce an output matrix C. Launch the kernels as directed in the code and also perform relevant memory allocations and memory transfers. In the first kernel, **per_row_AB_kernel** each thread should process a complete row of both matrices **A** and **B**. In the second kernel, **per_column_AB_kernel** each thread should process a complete column of both matrices **A** and **B**. In the third kernel, **per_element_kernel** each thread should process exactly one element from both the input matrices. For this assignment, **per_row_AB_kernel** is to be launched with **1D grid and 1D block** while the **per_column_AB_kernel** is to be launched with **1D grid and 2D block**. The launch configuration of the **per_element_kernel** has already been set to **2D grid and 2D block** for evaluation purposes, please note this shouldn't be changed.

2 Input and Output

2.1 Input

- Matrix A of size m x n
- Matrix B of size m x n

2.2 Output

- Matrix C of size mn x mn
- Output is computed as C = A $\otimes$ B^T, where X^T refers to the transpose of matrix X, and X $\otimes$ Y is the Kronecker Product of matrices A and B. Please refer to the following article on Kronecker Product for further details.

2.3 Constraints

- $2 \leq m \leq 100$
- $2 \leq n \leq 100$

3 Sample Testcase

A: $\begin{bmatrix} 4 & -3 \\ 1 & 2 \\ 6 & -8 \end{bmatrix}$ B: $\begin{bmatrix} -5 & 0 \\ 4 & 6 \\ 9 & -2 \end{bmatrix}$ B^T: $\begin{bmatrix} -5 & 4 & 9 \\ 0 & 6 & -2 \end{bmatrix}$ C: $\begin{bmatrix} -20 & 16 & 36 & 15 & -12 & -27 \\ 0 & 24 & -8 & 0 & -18 & 6 \\ -5 & 4 & 9 & -10 & 8 & 18 \\ 0 & 6 & -2 & 0 & 12 & -4 \\ -30 & 24 & 54 & 40 & -32 & -72 \\ 0 & 36 & -12 & 0 & -48 & 16 \end{bmatrix}$

4 Points to be Noted

- The file 'main.cu' contains the starter code for reading inputs and writing output to files. You are required to complete the three kernels, allocate memory for A, B, C on the GPU, perform data-transfers, launch kernel with the specified configuration. The launch configuration of the third kernel has already been set and shouldn't be modified, you are required to set the launch configuration for first two kernels.
- The number of threads launched for each of the three kernels will be more than or equal to the number of threads required to do the computation.
- Do not write any print statements inside the kernel and do not remove any part of the starter-code apart from adding custom code.
- Test your code on large input graphs.

5 Submission Guidelines

- Use the file 'main.cu' provided by us.
- Compress the file 'main.cu', which contains the implementation of the above-described functionality to ROLL NUMBER.zip.
- Submit only the ROLL NUMBER.zip file on Moodle.
- After submission, download the file and make sure it was the one you intended to submit.
- Kindly adhere strictly to the above guidelines.

6 Learning Suggestions

Write a CPU-version of code achieving the same functionality. Time the CPU code and GPU code separately for large matrices and compare the performances.