CS6023: GPU Programming

Assignment-2 (15 marks)

Deadline: March 5th, 2023, 23:55 on Moodle

1 Problem Statement

Given four integer matrices $A_{p\times q}, B_{q\times r}, C_{p\times q}$, and $D_{r\times q}$, efficiently compute matrix E

$$E = AB + CD^T \tag{1}$$

by considering the aspects of **memory coalescing** and **shared memory**.

2 Input and Output

2.1 Input format

- First line of input contain 3 integers p, q, and r.
- Next p lines contain q space-separated integers representing matrix A.
- Next q lines contain r space-separated integers, representing matrix B.
- Next p lines contain q space-separated integers representing matrix C.
- Next r lines contain q space-separated integers representing matrix D.

2.2 Output format

- Write p lines each containing r space separated integers representing matrix $E_{p\times r}$

2.3 Constraints

- $2 \leq p, q, r \leq 1024$.
- Each integer of matrices A, B, C, D is in range $[-10, 10]$.

3 Sample Testcase

- Let $p = 2, q = 3, r = 2$

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} B = \begin{bmatrix} 1 & -2 \\ -4 & 3 \\ 5 & -6 \end{bmatrix} C = \begin{bmatrix} 4 & 2 & 10 \\ 7 & 5 & -8 \end{bmatrix} D = \begin{bmatrix} 8 & 1 & 5 \\ -3 & 3 & -6 \end{bmatrix}$$

$$\implies AB = \begin{bmatrix} 8 & -14 \\ 14 & -29 \end{bmatrix} \quad CD^T \begin{bmatrix} 84 & -66 \\ 21 & 42 \end{bmatrix}$$

$$\implies E = AB + CD^T = \begin{bmatrix} 92 & -80 \\ 35 & 13 \end{bmatrix}$$

4 Points to be noted

- The file `main.cu` provided by us contains the code, which takes care of taking the input, printing the result, and printing the execution time.
- Don't write any code in the main() function. Do not write any print statements.
- You need to implement the compute() function provided in the `main.cu`.
- You are free to use any number of functions/kernels.
- You can launch the kernels as you wish.
- Test your code on large input matrices.
- **It is compulsory to optimize for coalesced accesses. Also, make use of shared memory.**

5 Submission Guidelines

- Rename `main.cu` to `YourRollNumber.cu`. For example, if your roll number is cs19b100, then rename the file to `cs19b100.cu`.
- **Submit only this `.cu` file on Moodle.**
- After submission, download the file and make sure it was the one you intended to submit.

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.
- Try to use a minimum number of kernel calls.
- Try to use `syncthreads()` and `syncwarps()` as minimum as possible to gain performance benefits.
- Exploit shared memory as much as possible to gain performance benefits.
- Think of an approach or technique that can avoid bank conflicts of shared memory.
- To understand how good coalescing and shared-memory utilization are, use Nvidia profilers to see effective memory bandwidth on large test cases.