BLG562E – Parallel Computing for GPUs using CUDA
Homework #4
Due on: 02/06/2023

1. *Optimized Dense Matrix-Matrix Multiplication using GPUs*

In the first part of this homework, you are expected to implement the dense matrix-matrix multiplication kernel using CUDA using with optimizations (at least implementing tiling using scratchpad memory). Do not use any matrix-multiplication libraries.

a) Measure the running time of your kernel with a matrix size of N=1024x1024, M=1024x1024.
b) Do a performance profiling using NVVP/nvprof/NSight profiling tool and discuss the performance bottlenecks and identify opportunities for performance optimizations. If the occupancy is a problem work on the Block size to make sure you have enough occupancy per SM, then re-do the profiling. Compare your results against the very first version of your dense matrix-matrix multiplication implemented as part of homework 3.
c) Repeat the steps in (a) and (b) with matrix sizes of N=2048x2048, M=2048x2048 and n=4096x4096, M=4096

2. *Sparse Matrix-Matrix Multiplication using GPUs*

In the second part of the homework, you are expected to implement the sparse matrix-matrix multiplication kernel using CUDA with using cuSparse or CUSP library. Make sure you use an appropriate compressed format for storage of sparse matrix (CSR, CSC, COO etc.).

For benchmarking and testing your code, you can use sparse matrixes from https://www.cise.ufl.edu/research/sparse/matrices/list_by_id.html

Choose three different matrix combinations to study performance of your sparse matrix-matrix multiplication kernel. For each matrix set:

a) Measure the running time of your kernel.
b) Do a performance profiling using NVVP/nvprof/NSight profiling tool and discuss the performance bottlenecks and identify opportunities for performance optimizations. If the occupancy is a problem work on the Block size to make sure you have enough occupancy per SM, then re-do the profiling.

In your report, provide details on the GPU model, specs, compiler etc. For both parts, provide the pseudo-code of your kernel implementation. For sparse matrix-matrix multiplication, explain which format that you use for sparse matrix storage and why, explain which matrixes that you use for benchmarking your kernel and why that you have chosen them. Give the final block-size that you use for each part.

Submit your homework as a single zip file. Include your source code and an your report (as a pdf file) in your submission.