Université de Pau et des Pays de l'Adour
Département de Mathématiques
2022-2023

M2-BigData : GPGPU
Chapter 4 – Exercice 2

Objectives

Implement a basic dense matrix multiplication kernel with a tiled algorithm using device's shared memory.

Instructions

Create a second version of your program from exercise 1 and adjust to perform a tiled algorithm with shared memory. In this exercice, we use a tile size of 8x8.

To simplify, we focus on square matrices of size equal to a multiple of 8 (tile size).

Once the program is correct, produce a final version that handle square matrices of any size.

Questions

1. How many floating operations are being performed in your matrix multiply kernel ? explain.
2. How many global memory reads are being performed by your kernel ? explain.
3. How many global memory writes are being performed by your kernel ? explain.
4. Compute the arithmetic intensity of your kernel. The arithmetic intensity is a FLOP/Byte number standing for the number of floating point operations performed per byte of global memory access.
5. Compare with the arithmetic intensity of the `basicMatMul` kernel. Explain why, at same matrices dimensions, the tiled version is better than the basic one. Get the kernels execution time using NVIDIA profiler `nsys`.
6. Use the occupancy calculator to compute the occupancy for `TILE_SIZE` equals to 8, 16 and 32. Which size gives the best computational time ?
7. Suppose you have matrices with dimensions bigger than the max thread dimensions. Sketch an algorithm that would perform matrix multiplication algorithm that would perform the multiplication in this case.
8. Suppose you have matrices that would not fit in global memory. Sketch an algorithm that would perform matrix multiplication algorithm that would perform the multiplication out of place.