# CS6023 Assignment 2 - Report

E Santhosh Kumar (CS16B107)

## Q1

- Kernel 1 Execution Time: 55155.941406 ms
- Kernel 2 Execution Time:11679.338867 ms

$\implies$ Speed-Up Factor: 4.72
Kernel 2 maps the x dimension (fastest moving dimension for thread-ID calculation) of thread-IDs to the column indices. Thus threads belonging to different columns of the same row get executed as part of the same warp. This requires simultaneous access of contiguous memory segments (elements belonging to one row of the second matrix are accessed in every iteration), which can be done efficiently through burst transfers, etc. On the other hand, Kernel 1 cannot make use of this kind of memory coalescing, and is hence slower.

## Q2

The block and thread dimensions compared are given in Table 1. The figure 1 shows the runtime values for different values of threads per blocks.

| Block Dimension | Thread Dimension |
|---|---|
| (4096, 2048) | (2, 4) |
| (2048, 2048) | (4, 4) |
| (2048, 1024) | (4, 8) |
| (1024, 1024) | (8, 8) |
| (1024, 512) | (8, 16) |
| (512, 512) | (16, 16) |
| (512, 256) | (16, 32) |
| (256, 256) | (32, 32) |

Table 1

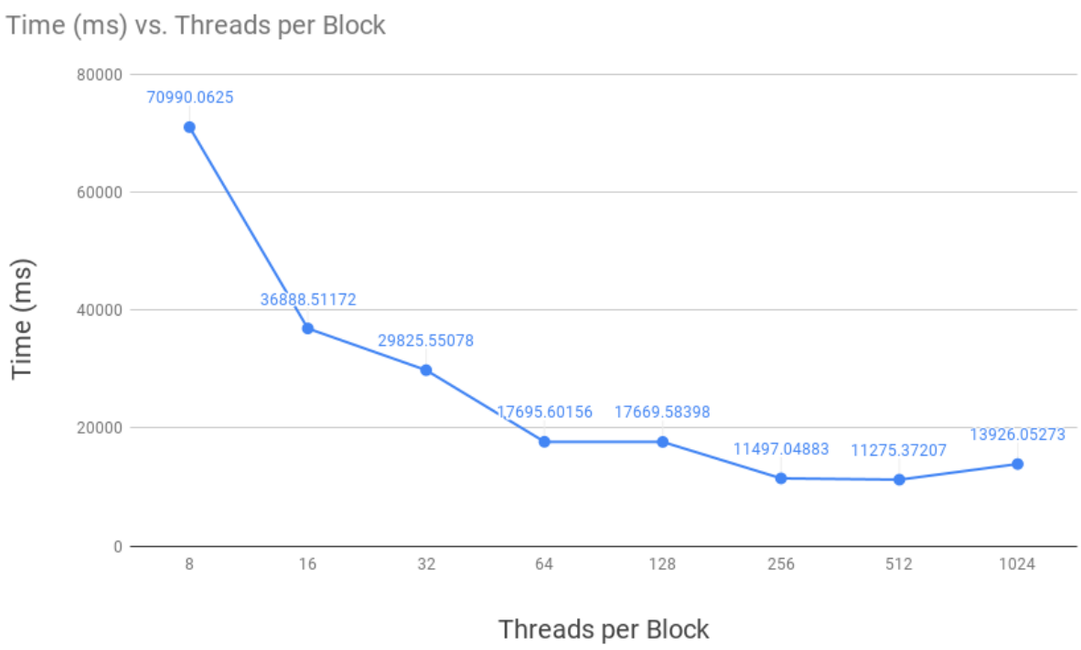


Figure 1

## Q3

- Execution Time for the 32 x 32 dimensional input matrices was observed to be 0.152000 ms.

- Code fails to compile for the case with 8192 x 8192 dimensional input matrices. This is because the required amount of shared memory in this case exceeds the available capacity of the GPU. Each of the 512*512 blocks requires access to 2*16*8192*sizeof(double) = 256 kB of shared memory, which is not available

## Q4

- Execution Time Without Tiling: 11679.338867 ms
- Execution Time With Tiling: 4836.722168 ms

Every memory access made in the non-tiled program is a global memory access (expensive). So, each thread in each block makes 2*8192 global memory accesses.

On the other hand, the tiled program stores specific tiles of memory in the shared memory to reduce the number of global memory accesses. In particular, each thread in each block makes 2*(num of phases) = 2*(8192/16) = 1024 global memory accesses, and, 2*8192 shared memory accesses (relatively faster than global memory accesses). The use of shared memory instead of global memory gives the speedup

## Q5

Figure 2 shows the execution time for different tile sizes. The optimal tile size was observed to be 32 x 32.

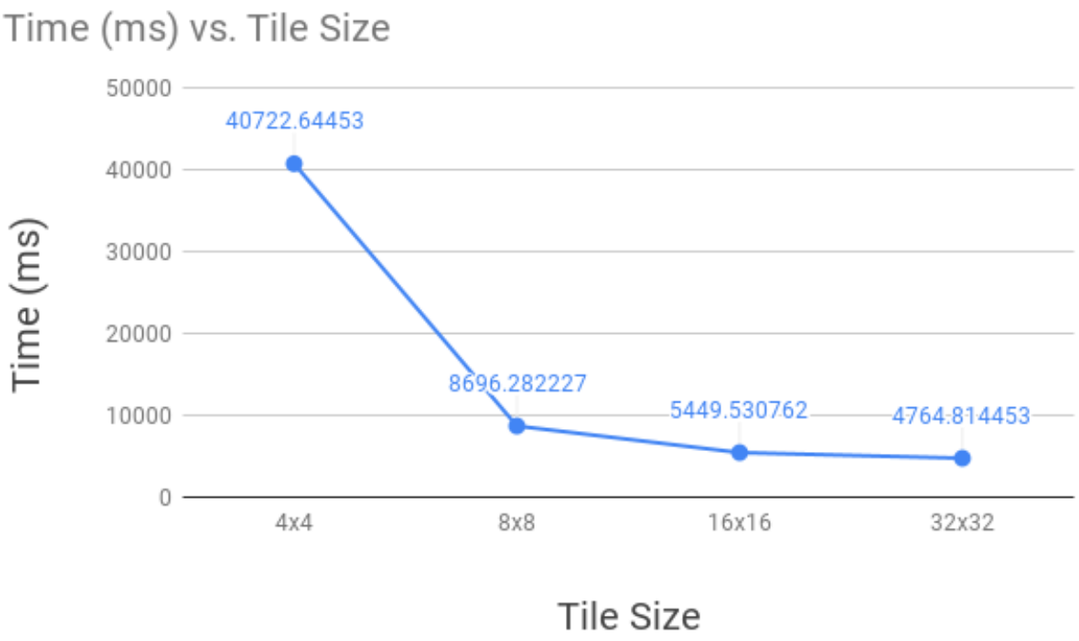


Figure 2

## Q6

Execution Time for rectangular input matrices having sizes 4096 x 8192 and 8192 x 16384 is 11551.587891 ms (tiling not used).