Assignment 4: Introduction to GPU programming

1 Launching 2D kernels and exploiting the massive parallelism

As the maximum size to test was 1024, and the machine supported maximum of 1024 threads, the number of threads per block was reasonably set to size / 2. There are two blocks per X dimension and N blocks per Y direction.

Thread count could have been increased to achieve even greater performance. Also, thread dimensionality inside blocks could have been exploited too (though it would not have increased the performance).

c)

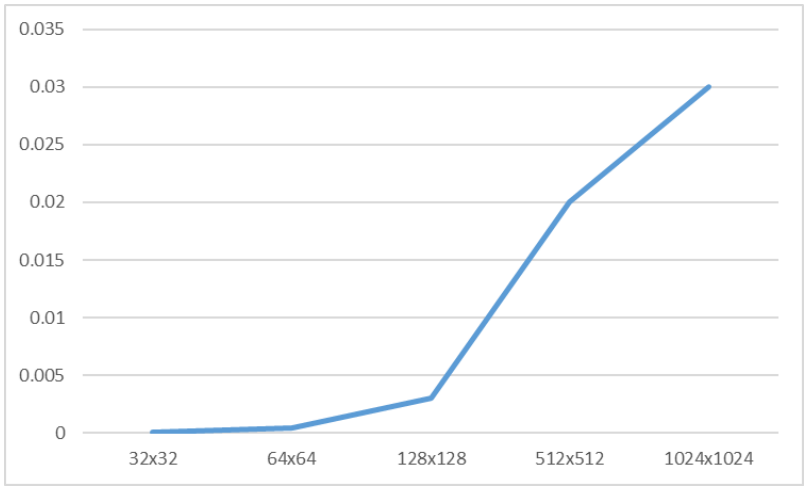

Figure 1: Matrix multiplication GPU multiplication timespan for various sizes. The Y axis is in seconds. X axis represents matrices sizes. It is clear that for small sizes, the multiplication is achieved in miliseconds.

2 Experiencing parallelism using MPI + OpenMP + CUDA

e) Elapsed time: 8.762212s. We can see that the matrix addition part occupies the majority of the runtime (as the program does not finish until addition is finalized),

as we've observed before that GPU matrix multiplication is merely 0.3s for this size.