HPC CUDA - Hands on

Marcello Capasso

May 2019

Abstract

This practicum was meant to give a hands on on the CUDA programming. Two algorithms were implemented, matrix transposition and matrix-matrix multiplication. Moreover, a comparison of the running time between the naive and the optimized versions was made.

1 Results

1.1 Matrix Transpose

The matrix transpose algorithm was implemented in two versions: one in which the matrix element indeces are swapped in a naive way, whereas the second one includes the creation of a shared memory tile in order to exploit memory coalescence and data locality. The running time is measured for different combinations of Threads per blocks and the results are plotted in figure 1. The matrix dimensions was set to 8192x8192 doubles.

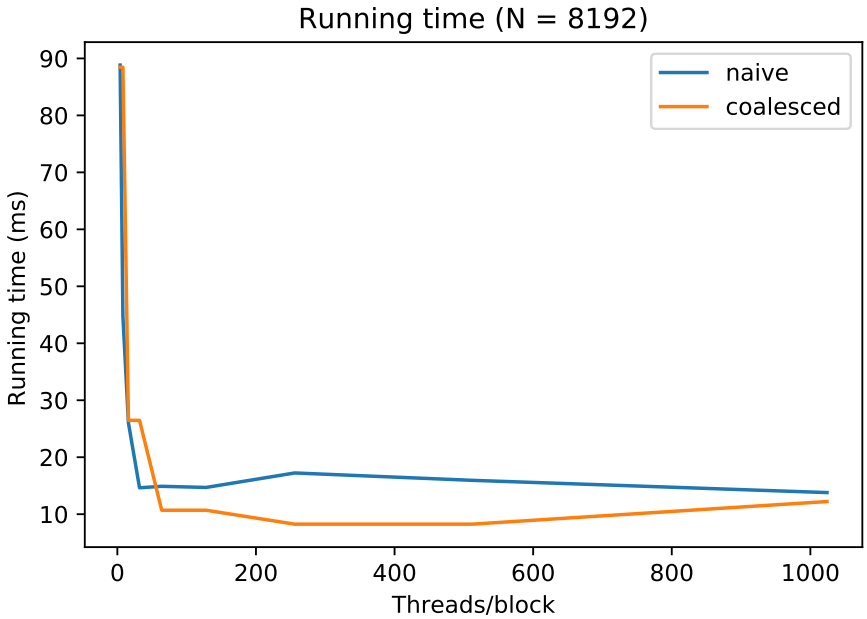

Figure 1: Comparison between naive and coalesced memory versions.

As it can be seen, the coalesced versions runs approximately 1.5 times faster than the naive one. The best tile dimension was found to be 16x16 = 256 threads/block. In this way, the contiguous memory needed by a thread warp is loaded into memory once per warp. Additional speed up is given by the exploitation of data locality during read and write operations on the global memory.

1.2 Matrix-Matrix multiplication

The matrix-matrix multiplication algorithm was implemented in two versions: the first one was based on row-columns multiplication (regular), while the second one was based on row-row multiplication (transposed), in order to exploit data locality during the read operation of both input matrices. Clearly, the second matrix was transposed before multiplication. Running times were taken and plotted in figure 2.

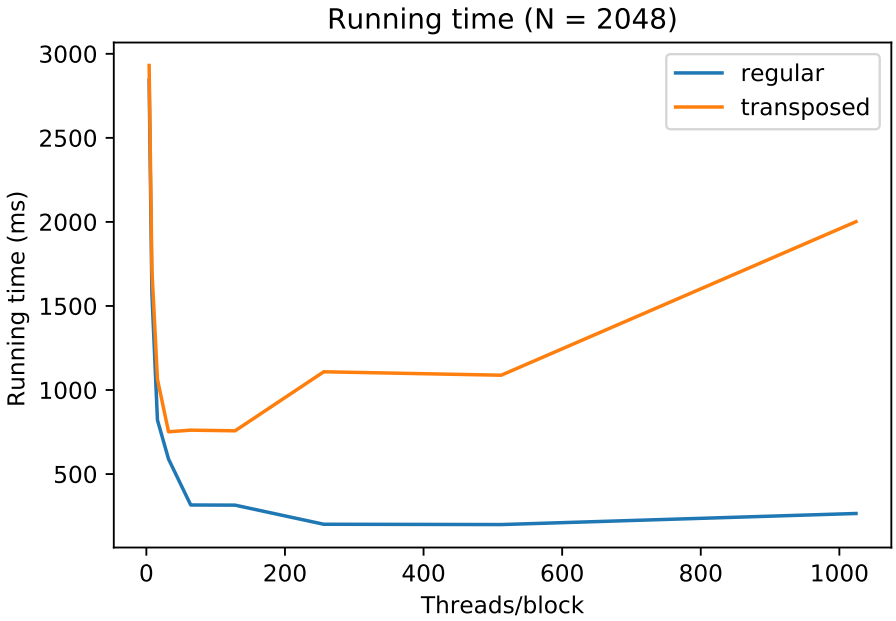

Figure 2: Comparison between the regular and the transposed versions.

Once again the minimum running time for the regular version was found to be around 256 threads per block, again because of memory coalescence. The result obtained for the transposed version however, was not expected at all. This odd behaviour could be given by erroneous coding or by the way memory is loaded into GPU multiprocessors, which causes memory conflicts.

A serial version of the matrix-matrix multiplication was implemented for regular CPU processor and the transposed versions ran 6 times faster than the regular one.