Report

Matrix Multiplication using both naive and shared memory approach.

Paweł Lipiór, Michał Żoczek

November 28, 2019

Multiplication of matrices is one of a more important tasks in Computer Science. It is both common and computationally expensive making good optimization a must. During this lab we tried to compare naive implementation of matrix multiplication and more advanced one utilizing shared memory.

In first version of the program each element of product matrix is calculated independently. The code for this approach is pretty simple, but there are some problems: multiple threads need to read the same data from memory, and one thread need to read the same data multiple times.

The second implementation of the program tries to improve things using shared memory. Shared memory, in accordance to its name is available for all threads within block which leads to faster execution as less time is spent on accessing data. While shared memory makes program faster it also makes things a little more complicated - there isn't that much of this type of memory available so it is necessary to manually split the data into tiles to make sure it will fit. This was done by parting the product matrix into submatrices each size of one block. To calculate the submatrices the two factor matrices were then split into smaller parts, and those were written into shared memory making access to data much faster. As the threads can finish their work in different times it's important to synchronize the threads using `__syncthreads()` command.