**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №6**

**по дисциплине «Параллельные алгоритмы»**

Тема: Оптимизация доступа к памяти в модели OpenCL

|  |  |  |
| --- | --- | --- |
| Студент гр. 0304 |  | Люлин Д.В. |
| Преподаватель |  | Сергеева Е.И. |

Санкт-Петербург

2023

**Цель работы.**

Изучить виды памяти в OpenCL, реализовать умножение матриц с оптимизацией доступа к памяти, сравнить с другими реализациями.

**Здание.**

Реализовать умножение матриц на OpenCL.

В отчете: Произвести сравнение производительности с CPU реализацией из лаб. 4.

**Выполнение работы.**

Умножение матриц выполнено в ядре OpenCL. Была использована рабочая группа размером 16x16 (256). В рамках одного рабочего элемента вычисляется один элемент результирующей матрицы, а в рамках рабочей группы – блок элементов размером 16x16. Схема вычислений представлена на рис. 1.

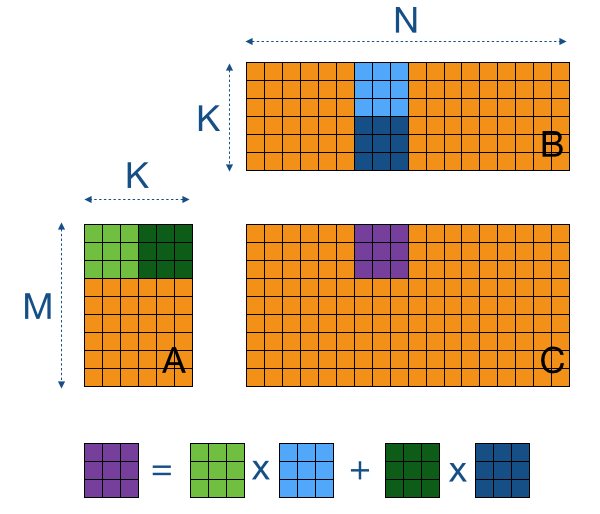


Рисунок 1. Схема умножения матриц с использованием блоков.

Чтобы память блока была разделяемой в рамках своей рабочей группы, для хранения блоков в ядре OpenCL была использована локальная память. Рабочие элементы заполняют блок параллельно, а для синхронизации используется барьер *CLK\_LOCAL\_MEM\_FENCE*.

Каждый элемент представляет собой скалярное произведение строки первой матрицы и столбца второй матрицы. При вычислении блоков создаётся цикл, в каждой итерации которого заполняется два блока, по одному блоку из каждой матрицы. Заполнение происходит параллельно всеми рабочими элементами рабочей группы. Затем вычисляется частичное скалярное произведение элементов блока, в котором учитываются только пройденные блоки. В цикле затем берётся следующая пара блоков, и скалярные произведения дополняются новыми значениями, пока не будут вычислены полностью с использованием всех необходимых строк и столбцов.

Было измерено время вычисления произведения матриц большого размера, аналогичные измерениям в лаб. 4. В таблице 1 приведены результаты измерений.

Таблица 1. Измерения времени умножения матриц.

|  |  |
| --- | --- |
| **Размер матриц** | **Время вычисления, мс** |
| 32x32 | 58.958 |
| 64x64 | 59.7166 |
| 128x128 | 56.624 |
| 256x256 | 60.9101 |
| 512x512 | 84.8798 |
| 1024x1024 | 291.232 |
| 2048x2048 | 2052.25 |
| 4096x4096 | 16500.1 |

По таблице видно, что для небольших матриц время практически одинаково. Это связано с загрузкой данных и созданием контекста OpenCL. По сравнению с этим временем, время самого вычисления пренебрежимо мало.

Для больших матриц (1024x1024 и больше) время вычисления возрастает, но остаётся намного меньшим, чем время вычисления с помощью алгоритма Штрассена из лаб 4. Ускорение составило около 15 раз.

**Выводы.**

В ходе работы был реализован алгоритм умножения матриц с использованием OpenCL. При реализации были использованы оптимизации доступа к памяти.

Полученный алгоритм оказался быстрее параллельного алгоритма Штрассена в 15 раз.

ПРИЛОЖЕНИЕ А

Исходный код программы

Исходный код программы доступен в репозитории <https://github.com/Astana-Mirza/parallel_algo/tree/master>.