Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

**«Самарский НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**имени академика С.П. Королева»**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

ПАРАЛЛЕЛЬНОЕ ПРОГРАММИРОВАНИЕ

**Отчёт по лабораторной работе №4**

ПРОГРАММА ПЕРЕМНОЖЕНИЯ ДВУХ МАТРИЦ ПО CUDA

Жирнов Владислав Евгеньевич

Группа 6313-100503D

Самара 2025

# **Цель работы**

Модифицировать программу для параллельной работы по технологии CUDA.

# **2 Программа перемножения двух матриц на языке C++**

Программа main.cu, написанная на языке C++ с расширением CUDA, выполняет умножение квадратных матриц с использованием возможностей GPU NVIDIA. Основной акцент был сделан на эффективное распараллеливание вычислений с помощью технологии CUDA, что позволяет значительно ускорить обработку больших матриц.

Создается 11 пар квадратных матриц с размерами от 10x10 до 100x100 с шагом 10, а также матрица с размером 1000x1000. Каждая пара матриц одного размера сохраняется в соответствующих директориях: Matrix\_1 содержит левые множители, а Matrix\_2 - правые множители. Таким образом, после завершения генерации в каждой из этих папок оказывается по 10 матриц соответствующих размеров - от 10x10 до 100x100.

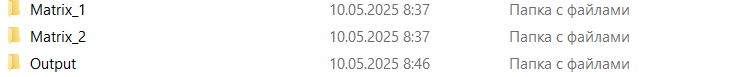


Рисунок 1 – основные каталоги, используемые при работе программы

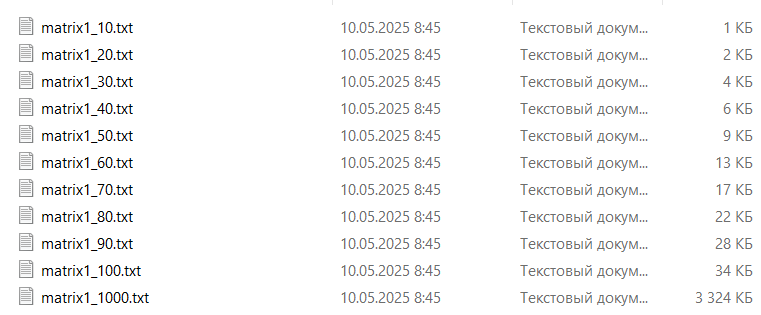


Рисунок 2 – внутренние каталоги каталога Matrix\_1.

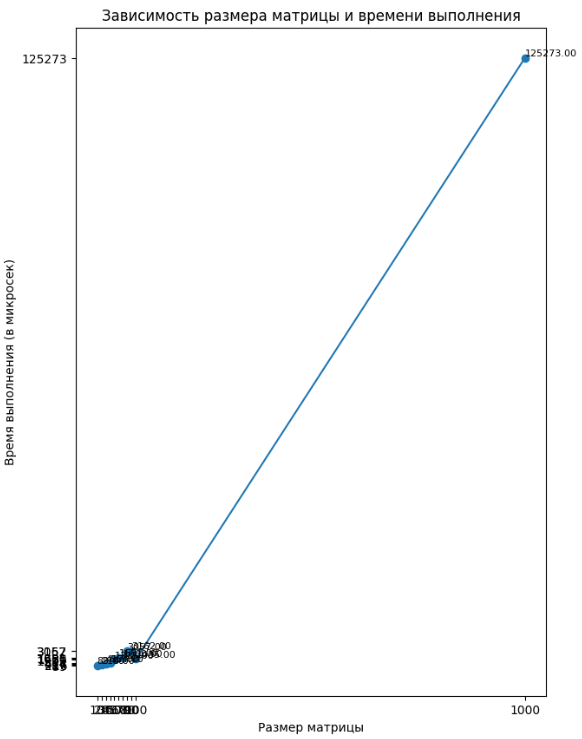
Исходные матрицы считываются из соответствующих файлов в директориях Matrix\_1 и Matrix\_2, после чего производится их поэлементное перемножение. Для каждого размера матрицы фиксируется время умножения, результаты сохраняются в файл times\_CUDA.txt

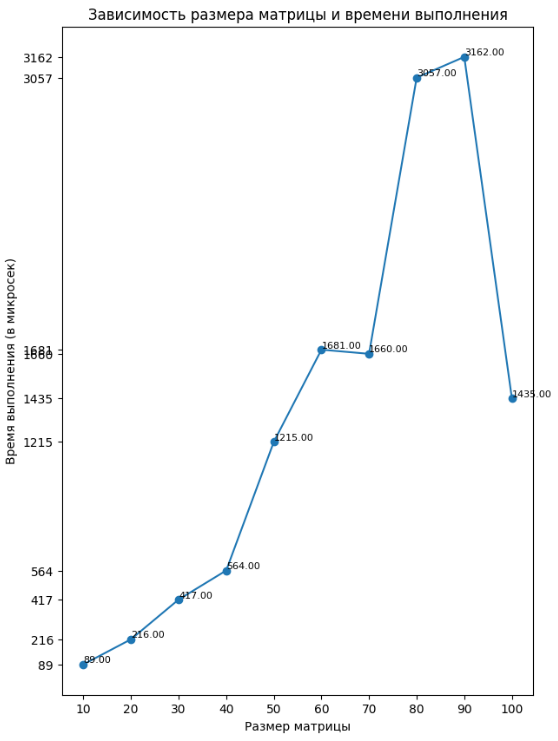
Для верификации результатов работы C++ программы был разработан Python-скрипт check.ipynb. В этом скрипте используется функция dot() из библиотеки NumPy, которая выполняет точное матричное умножение и служит эталоном для сравнения.

Программа успешно адаптирована для работы на GPU с использованием технологии CUDA. Реализация демонстрирует высокую эффективность при обработке больших матриц и может быть использована для решения задач, требующих интенсивных вычислений.

# **Экспериментальные результаты**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Size** | **10** | **20** | **30** | **40** | **50** | **60** | **70** | **80** | **90** | **100** | **1000** |
| **None CUDA** | 9 | 72 | 227 | 536 | 936 | 1638 | 2551 | 3910 | 5344 | 7969 | 9201939 |
| **Using CUDA** | 89 | 216 | 417 | 564 | 1215 | 1681 | 1660 | 3057 | 3162 | 1435 | 125273 |





***Вывод:*** Наблюдаемая линейная зависимость времени выполнения от размера матриц соответствует теоретической оценке сложности алгоритма умножения матриц. Это наглядно демонстрирует, что с увеличением размерности матриц время вычислений растет пропорционально. Однако при использовании CUDA для параллельных вычислений можно добиться значительного ускорения обработки данных.