МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4
по дисциплине «Параллельные алгоритмы»
Тема: Параллельное умножение матриц

Студент гр. 0303 ____________________ Мыратгелдиев А. М.

Преподаватель ____________________ Сергеева Е.И.

Санкт-Петербург

2023

Цель работы.

Научиться реализовывать алгоритм Штрассена для перемножения матриц.

Задание.

1. Реализовать параллельный алгоритм умножения матриц с масштабируемым разбиением по потокам.

   Исследовать масштабируемость выполненной реализации с реализацией из работы 1.
2. Реализовать параллельный алгоритм “быстрого” умножения матриц (Штрассена или его модификации). Проверить, что результаты вычислений реализаций 4.1 и 4.2 совпадают. Сравнить производительность с реализацией 4.1 на больших размерностях данных (порядка 10^4 – 10^6)

Выполнение работы.

Для выполнения данной лабораторной работы, был использован и расширен класс *Matrix* из предыдущих лабораторных работ. В данном классе были определены операторы суммы и вычитания для удобства дальнейших вычислений.

Был реализован класс *MatrixMultiplier*, который имеет несколько методов:

- *no_parallel* – умножает матрицы по классической формуле умножения за кубическое время;

- *parallel* – умножает, переданные в качестве аргументов матрицы, с масштабируемым разбиением по потокам (количество потоков передается третьим параметром);

- *strassen_alg* – умножает переданные на вход матрицы по алгоритму Штрассена;

Параллельный алгоритм умножения реализован таким образом, что *i*-й поток вычисляет *i+k*n* элементы результирующей матрицы, где *n* – общее количество потоков, которое было передано в качестве 3-го параметра, а k – 1,…, ($k*n < m$^2, m – размерность матрицы).

Алгоритм Штрассена работает только с квадратными матрицами размерности степени 2. Поэтому были реализованы несколько вспомогательных методов (*prepare_matrix, expand_matrix*), чтобы подготовить исходные матрицы для умножения, путем добавления нулевых столбцов и строк.

Алгоритм Штрассена рекурсивно разбивает матрицу на 4 подматрицы размерности *m/2* и вычисляет следующие вспомогательные матрицы:

$$
\begin{aligned}
D &= (A_{11} + A_{22})(B_{11} + B_{22});\\
D_1 &= (A_{12} - A_{22})(B_{21} + B_{22});\\
D_2 &= (A_{21} - A_{11})(B_{11} + B_{12});\\
H_1 &= (A_{11} + A_{12})B_{22};\\
H_2 &= (A_{21} + A_{22})B_{11};\\
V_1 &= A_{22}(B_{21} - B_{11});\\
V_2 &= A_{11}(B_{12} - B_{22});
\end{aligned}
$$

На основе этих вспомогательных матриц, вычисляются элементы результирующей матрицы:

$$\begin{pmatrix} D + D_1 + V_1 - H_1 & V_2 + H_1 \\ V_1 + H_2 & D + D_2 + V_2 - H_2 \end{pmatrix}$$

Сравним эти алгоритмы на тестовых данных:

Размерность матрицы	Масштабируемое параллельное умножение (3 потока)	Алгоритм Штрассена
32 x 32	0 мс	5 мс
256 x 256	36 мс	484 мс
1024 x 1024	3069 мс	15478 мс
4096 x 4096	338607 мс	1088547 мс

Из таблицы видно, что масштабируемое параллельное умножение работает быстрее, чем алгоритм Штрассена. Алгоритм Штрассена даст выигрыш на больших плотных матрицах

Выводы.

В ходе выполнения лабораторной работы были реализованы алгоритмы перемножения матриц – масштабируемое параллельное умножение и алгоритм Штрассена.