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,…, ($k*n < m\hat{}2$, 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 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
64 x 64	1 мс	70 мс
128 x 128	7 мс	64 мс
256 x 256	51 мс	102 мс
512 x 512	480 мс	218 мс
1024 x 1024	4265 мс	1239 мс
2048 x 2048	44940 мс	6275 мс

4096 x 4096	422410 мс	51492 мс

Из таблицы видно, что масштабируемое параллельное умножение работает быстрее, чем алгоритм Штрассена при небольших размерностях. Алгоритм Штрассена дает работает намного быстрее на больших плотных матрицах.

Выводы.

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