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

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Параллельные алгоритмы»

Тема: Параллельное умножение матриц

Студент гр. 9303 _____ Махаличев Н.А.

Преподаватель _____ Сергеева Е.И.

Санкт-Петербург

2022

Цель работы.

Изучение и практическое применение принципов параллельного умножения матриц на языке C++, реализация алгоритма «быстрого» умножения Штрассена.

Задание.

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

Выполнение работы.

Был реализован параллельный алгоритм умножения матриц. При выполнении программы задается число потоков. Индексы результирующей матрицы равномерно распределяются между потоками и в функции vector_mult для каждого элемента [x, y] вычисляется его значение – умножение строки x первого массива на столбец y второго массива. Зависимость времени работы программы от размера массивов и от количества умножающих потоков представлено в табл. 1.

Таблица 1 – Зависимость времени работы параллельного алгоритма умножения матриц от размера массивов и от количества умножающих потоков

	1	2	4	8	16
10x10 и 10x10	0 мс	0 мс	0 мс	1 мс	1 мс
100x10 и 100x100	3 мс	2 мс	2 мс	2 мс	3 мс

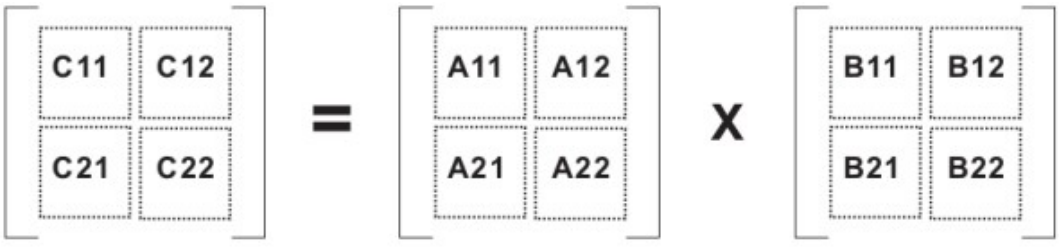
Продолжение таблицы 1.

100x100 и 100x100	20 мс	11 мс	7 мс	9 мс	9 мс
1000x1000 и 1000x1000	16391 мс	8830 мс	5739 мс	4468 мс	4213 мс
3000x3000 и 3000x3000	633268 мс	334322 мс	217727 мс	163818 мс	154067 мс

На основе полученных результатов можно сделать вывод, что операция умножения матриц наиболее зависима от количества потоков, чем операция сложения матриц. Это объясняется тем, что операция умножения для матриц сложнее, чем операция сложения (чем больше строк и столбцов, тем больше необходимо произвести умножений и сложения для получения одного элемента). Получены следующие сложности:

1) Умножение матриц – $O(n^3)$;
2) Сложение матриц – $O(n)$.

Был реализован алгоритм Штрассена умножения матриц, обобщённая схема которого представлена на рис. 1.



$$M_1 = (A_{11} + A_{22})(B_{11} + B_{22}) \qquad C_{11} = M_1 + M_4 - M_5 + M_7$$
$$M_2 = (A_{21} + A_{22})B_{11} \qquad C_{12} = M_3 + M5$$
$$M_3 = A_{11}(B_{12} - B_{22}) \qquad C_{21} = M_2 + M_4$$
$$M_4 = A_{22}(B_{21} - B_{11}) \qquad C_{22} = M_1 - M_2 + M_3 + M_6$$
$$M_5 = (A_{11} + A_{12})B_{22}$$
$$M_6 = (A_{21} - A_{11})(B_{11} + B_{12})$$
$$M_7 = (A_{12} - A_{22})(B_{21} + B_{22})$$

Рисунок 1 – Обобщённая схема алгоритма Штрассена

Так как алгоритм Штрассена работает с квадратными матрицами размера 2^nx2^n, в начале работы программы при необходимости происходит заполнение матриц нулями до необходимого размера. Выполняется рекурсия:

1) На первом уровне вложенности рекурсии создаются семь потоков для вычисления значений M;
2) Спускаясь ниже по рекурсии матрицы A и B каждый раз делятся на 4 части;
3) Достигнув назначенного уровня вложенности рекурсии, разделённые части матриц перемножаются между собой обычным алгоритмом умножения матриц и данное значение возвращается на уровень выше;
4) Поднимаясь выше, вычисляются значения M и объединяются в матрицу C, каждая часть которой тоже вычисляется;
5) В результате выполнения рекурсии возвращена матрица C размера 2^nx2^n, в которой хранится результат умножения двух матриц.

После выполнения рекурсии выводится необходимая часть матрицы C, в которой хранится результат умножения.

Доказательство идентичности результатов рассмотренных алгоритмов представлено на рис. 2.

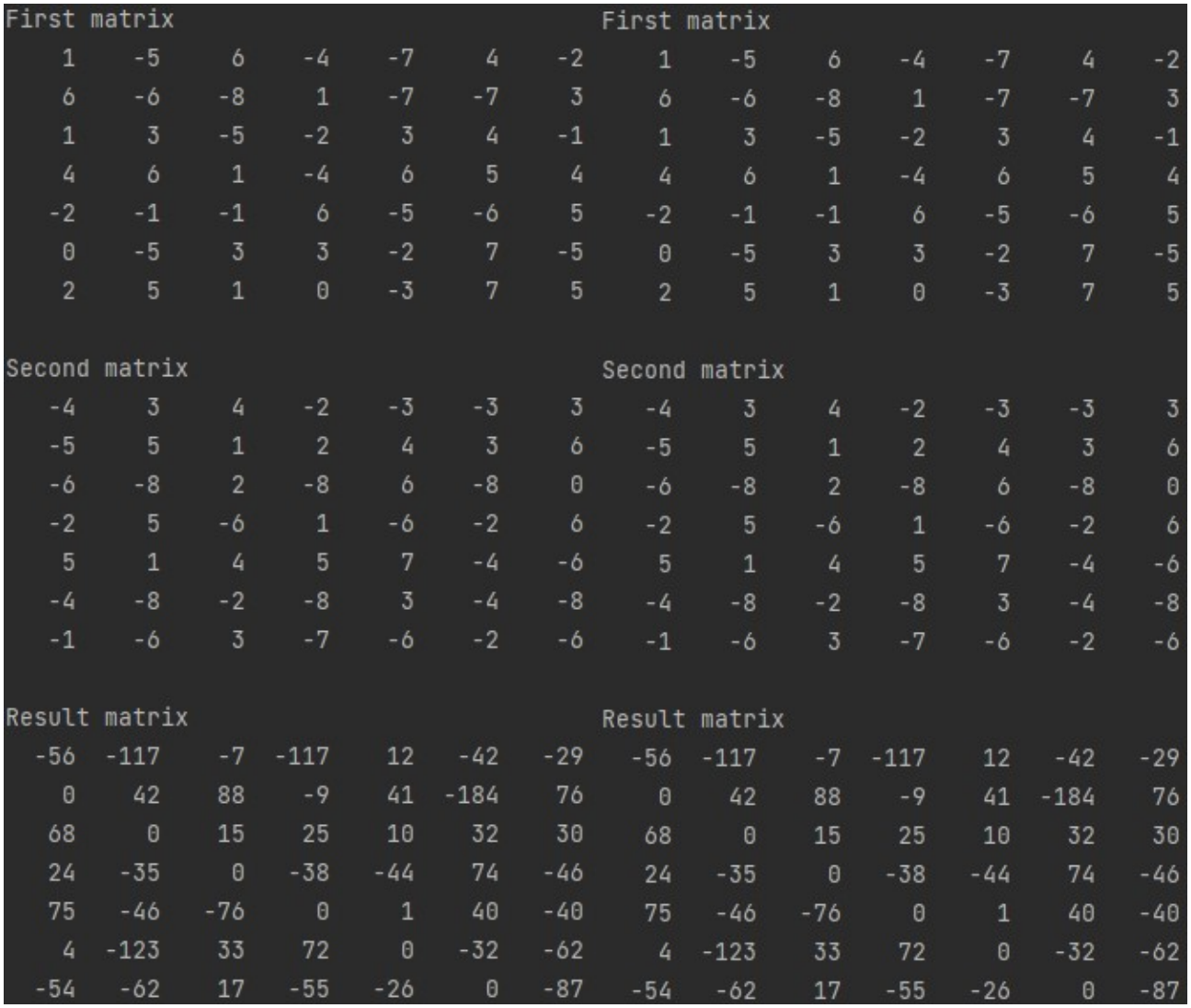


Рисунок 2 – Результаты умножения, полученные двумя алгоритмами (справа – алгоритм Штрассена)

Сравним производительность с реализацией параллельного алгоритма умножения матриц на больших размерностях данных. Результат сравнения представлен в табл. 2.

Таблица 2 – Сравнение производительности алгоритма Штрассена с реализацией параллельного алгоритма умножения матриц на больших размерностях данных

Размер матриц	Глубина рекурсии	Алгоритм Штрассена	Параллельное умножение (16 потоков)
500x500	1	561 мс	538 мс
	2	508 мс	
	4	494 мс	
1000x1000	2	3561 мс	4213 мс
	4	3064 мс	
	6	3468 мс	
2000x2000	2	31752 мс	41484 мс
	4	21326 мс	
	6	21096 мс	
3000x3000	4	182492 мс	154067 мс
	6	144816 мс	
	8	178387 мс	

В результате сравнения был сделан вывод, что алгоритм Штрассена работает быстрее, чем алгоритм параллельного умножения матриц. Причиной уменьшения времени выполнения является уменьшение количества выполняемых операций, что упрощает умножение до сложности $O(n^{2.81})$.

Выводы.

В процессе выполнения лабораторной работы были изучены и применены принципы параллельного умножения матриц на языке C++, реализован алгоритм «быстрого» умножения Штрассена.