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. 0304 _____ Шквиря Е.В.

Преподаватель _____ Сергеева Е.И.

Санкт-Петербург

2023

Цели работы

Изучение алгоритмов параллельного умножения матриц. Изучение алгоритма Штрассена.

Постановка задачи

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

Исследовать масштабируемость выполненной реализации с реализацией из работы 1.

4.2 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 дальнейшей работы с матрицами был разработан класс `Matrix`, в который были вынесены необходимые в рамках данной работы методы. Класс хранит в себе данные матрицы (элементы и размеры), а также поддерживает операции `slice` (скопировать кусок матрицы), `insert` (заменить кусок матрицы), `fillMatrix` (заполнить матрицу случайными числами), `printMatrix` (вывести элементы матрицы), а также перегружены операторы “+” и “-” для сложения и вычитания матриц. Данные методы нужны, в частности, для более удобной реализации алгоритма Штрассена, в котором нужно копировать и вставлять части матриц, а также производить их сложение и вычитание.

За основу алгоритма для параллельного умножения матриц была взята реализация из лабораторной работы 1, так как она поддерживала разбиение матрицы на отдельные части для дальнейшего умножения. В силу этого

производить сравнение текущей реализации с реализацией из лабораторной работы 1 бессмысленно, так как реализации идентичны.

Алгоритм Штрассена был реализован таким образом, что его распараллеливание происходит только на первом шаге, а после происходит рекурсивный вызов самой же функции, в которой все дальнейшие действия происходят последовательно. Это сделано для оптимизации самого алгоритма, так как в противном случае получалось бы большое количество потоков, из-за чего общая производительность упала. Также дополнительно была предусмотрена ситуация, когда после разбиения матрица имела размеры 8x8, в таком случае умножение производилось в одном потоке. Так как алгоритм Штрассена предназначен для перемножения квадратных матриц, у которых размер является степенью двойки - для анализа брались только такие матрицы. Для случаев матриц с другими параметрами можно бы было дополнять матрицу нулями до необходимых параметров, но в силу упрощения реализации это не было предусмотрено.

Анализ

Перед переходом к вычислениям стоит упомянуть, что работа выполнялась на машине с количеством ядер процессора равным 6 и максимальным числом потоков равным 12.

Для дальнейшего анализа было рассчитано время выполнения задачи с обычным алгоритмом параллельного вычисления и алгоритмом Штрассена. Во всех случаях выполнялся расчёт на 7 потоках. За результат бралось среднее значение из 2 запусков программы в одинаковых условиях. Результат приведён в таблице 1, измерения проводились в микросекундах.

Таблица 1. Результаты запусков программы с разными размерами матрицы.

N	Параллельный алгоритм	Алгоритм Штрассена
8	230	23
16	427	127

32	527	899
64	1680	2451
128	9028	13622
256	46142	54491
512	300057	321282
1024	2145877	1958094
2048	20283651	13933070

Как можно заметить, алгоритм Штрассена превосходит параллельный алгоритм по производительности, когда сторона матрицы не превосходит 16 или когда сторона матрицы превосходит 1024. Между этими размерами преимущество имеет параллельный алгоритм. Это объясняется тем, что когда сторона матрицы не превосходит 16 - алгоритм Штрассена большую часть времени отрабатывает как последовательный алгоритм (так как происходит разбиение до матриц размерности 8x8). Когда сторона матрицы больше 16, но меньше 1024 - в алгоритме Штрассена происходит много копирования и вставки матриц, из-за чего его производительность падает на фоне параллельного алгоритма, которому копирование не требуется. Когда сторона матрицы больше 1024 - алгоритм Штрассена выигрывает за счёт меньшего числа перемножений матриц (7 против 8).

Выводы

В ходе выполнения лабораторной работы были разработаны алгоритмы параллельного умножения матрицы - обычный и Штрассена. Была проанализирована скорость выполнения задачи с помощью этих двух алгоритмов.

Было выявлено, что алгоритм Штрассена работает быстрее, когда размер стороны матрицы либо маленький (когда можно практически сразу задействовать однопоточное умножение), либо большой (когда 7 умножений

против 8 более выгодно, чем отсутствие копирования и вставки матриц). В других случаях производительней оказывается обычный алгоритм перемножения матриц.