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

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Параллельные алгоритмы»

Тема: Основы работы с процессами и потоками

Студент гр. 9303 _____ Куршев Е.О.

Преподаватель _____ Сергеева Е.И.

Санкт-Петербург

2022

Цель работы.

Изучить работу процессов и потоков в языке C++.

Задание.

Выполнить поэлементное сложение 2х матриц M*N

Входные матрицы вводятся из файла (или генерируются).

Результат записывается в файл

1.1.

Выполнить задачу, разбив её на 3 процесса. Выбрать механизм обмена данными между процессами.

Процесс 1: заполняет данными входные матрицы (читает из файла или генерирует их некоторым образом).

Процесс 2: выполняет сложение

Процесс 3: выводит результат

1.2.1

Аналогично 1.1, используя потоки (threads)

1.2.2

Разбить сложение на P потоков.

Исследовать зависимость между количеством потоков, размерами входных данных и параметрами целевой вычислительной системы.

Выполнение работы.

Для генерации входных данных был написан файл matrix_generate.cpp, в котором происходит генерация матрицы числами из промежутка [0; 10000] с количеством строк и столбцов, которые задаёт пользователь. После выполнения получаются два файла — Matrix1.txt и Matrix2.txt, которые содержат сгенерированные матрицы.

1.1 Реализация с помощью процессов.

Реализация сложения двух матриц с помощью процессов выполнена в файле processes.cpp. Разбиение на три процесса происходит с помощью функции fork(), которая создаёт процессы-потомки. Для того, чтобы определить, какой процесс что выполняет используется идентификатор PID: если значение равно 0, то это потомок, и он выполняет свою часть, иначе, с помощью функции wait() происходит ожидание выполнения кода потомком.

1.2.1 Реализация с помощью потоков.

Реализация сложения двух матриц с помощью потоков выполнена в файле threads.cpp. Поток создаётся с помощью конструктора thread(), который принимает ссылки на выполняемую функцию и ссылки на аргументы для выполнения функции. Ожидание исполнения потока для продолжения исполнения программы выполняется с помощью метода join().

1.2.2 Разбиение операции сложения на P потоков.

Реализация сложения двух матриц с помощью нескольких потоков выполнена в файле Pthreads.cpp. На вход программе подается потоков сложения. Считанные матрицы переводятся из двумерного вектора в одномерный, далее происходит разбиение на потоки оптимальным образом, чтобы каждый поток выполнял примерно одинаковое число операций. Все созданные потоки хранятся в векторе, и после их инициализации для каждого потока вызывается метод join().

1.3 Исследование зависимости между количеством потоков, размерами входных данных и параметрами целевой вычислительной системы.

Зависимость времени сложения от количества потоков P при сложении матриц размером 1000x1000 представлена в табл. 1.

Таблица 1 — Зависимость времени сложения от количества потоков P при сложении матриц размером 1000x1000.

Количество потоков P	Время сложения, с
1	0.013053
2	0.010224
3	0.012193
4	0.011362
6	0.010556
23	0.010758

Зависимость времени сложения от размера матриц при сложении с помощью двух потоков представлена в табл. 2.

Таблица 2 — Зависимость времени сложения от размера матриц при сложении с помощью двух потоков.

Размер матриц	Время сложения, с
100x100	0.000368
500x500	0.002925
1000x1000	0.012531
5000x5000	0.141109
10000x10000	0.59985

Выводы.

В процессе выполнения лабораторной работы была изучена работа процессов и потоков в языке C++. Были решены три задачи, для каждой из которых написан код:

1) Разбиение чтения, сложения и вывода матриц на 3 процесса;
2) Разбиение чтения, сложения и вывода матриц на 3 потока;
3) Аналогично 2, сложение разбивается на P потоков.

Было проведено исследование зависимости времени выполнения программы от различных параметров, и было выявлено:

1) Размер матриц влияет на время выполнения операции сложения: чем больше элементов в матрице, тем дольше осуществляется сложение;
2) Не всегда увеличение числа потоков ведёт к уменьшению времени выполнения программы.