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

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Параллельные алгоритмы»

Тема: Реализация потокобезопасных структур данных с блокировками

Студент гр. 0304 _____ Максимов Е.А.

Преподаватель _____ Сергеева Е.И.

Санкт-Петербург

2023

Цель работы.

Изучить основные способы работы с блокировками потоков.

Постановка задачи.

Реализовать итерационное (потенциально бесконечное) выполнение подготовки, обработки и вывода данных по шаблону «производитель-потребитель» (потоки на основе лабораторной работы №1).

Обеспечить параллельное выполнение потоков обработки готовой порции данных, подготовки следующей порции данных и вывода предыдущих полученных результатов. Использовать механизм «условных переменных».

1. Использовать очередь с «грубой» блокировкой.

2. Использовать очередь с «тонкой» блокировкой

Сравнить производительность в зависимости от количества производителей и потребителей.

Выполнение работы.

Были написаны функции для ввода/вывода данных `std::vector<int> readInputFile()`, `void writeMatrixToFile(Matrix matrix)`, `bool openOutputFile()`, `void closeOutputFile()`. Чтение параметров программы производится из файла `data/input.txt`. Также из предыдущей лабораторной работы были заимствованы функции, реализующие расчёт матрицы. К этим функциям была добавлена функция `Matrix generateMatrix()` для генерации псевдослучайных квадратных матриц.

В двух случаях для буфера данных была использована очередь `std::queue`. Также в двух случаях количество итераций на поток потребителей было таким, что количество итераций производителей равно количеству итераций потребителя. Размерность матриц составляет 20×20, значения элементов матриц — целые числа в диапазоне от 0 до 999.

1. Очередь с «грубой» блокировкой.

В реализации использовались объекты мьютекса (`std::mutex`) и условной переменной (`std::condition_variable`), общие для производителя и потребителя.

2. Очередь с «тонкой» блокировкой.

В реализации использовались 2 различных объекта мьютекса (`std::mutex`), предназначенные для производителя и потребителя соответственно.

Для измерения времени работы программ была использована утилита `time`. Результаты тестирования представлены в приложении А.

Выводы.

В ходе лабораторной работы были изучены основные способы работы с блокировками потоков. По результатам анализа полученных данных в ходе тестирования выяснилось, что программа в «тонкой» блокировкой работает быстрее, чем программа с «грубой» блокировкой. Практическим результатом лабораторной работы является программный код, реализующий механизмы «грубой» и «тонкой» блокировок.

ПРИЛОЖЕНИЕ А
ТЕСТИРОВАНИЕ

Таблица А1 — Исследование зависимости времени работы программы от количества потоков для «грубой» блокировки

№	Количество производителей	Количество потребителей	Фактическое время (real), мс	Пользовательское время (user), мс	Процессорное время (sys), мс
1	1	1	48	32	20
2	1	2	50	54	21
3	2	1	74	64	12
4	5	5	193	187	21
5	10	10	366	358	58
6	50	50	1758	1617	371
7	10	100	409	366	136
8	50	100	1753	1495	359
9	100	100	3624	3684	516

Таблица А2 — Исследование зависимости времени работы программы от количества потоков для «тонкой» блокировки

№	Количество производителей	Количество потребителей	Фактическое время (real), мс	Пользовательское время (user), мс	Процессорное время (sys), мс
1	1	1	36	19	2
2	1	2	45	36	2
3	2	1	71	50	21
4	5	5	164	144	47
5	10	10	315	293	73
6	50	50	1542	1602	277
7	10	100	325	146	15
8	50	100	1573	1415	206
9	100	100	3082	2954	460