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

ОТЧЁТ

по лабораторной работе №3
по дисциплине «Параллельные алгоритмы»
Тема: Реализация параллельной структуры данных с тонкой блокировкой

Студент гр. 9303 _____ Эйсвальд М.И.

Преподаватель _____ Сергеева Е.И.

Санкт-Петербург
2022

Цель работы.

Изучить принципы построения структур данных с тонкой блокировкой, lock-free структур данных и алгоритмов работы с ними.

Задача.

Обеспечить структуру данных из лаб.2 как минимум тонкой блокировкой (* сделать lock-free). Протестировать доступ в случае нескольких потоков-производителей и потребителей. Сравнить производительность со структурой с грубой синхронизацией (т.е. с лаб.2). В отчёте сформулировать инвариант структуры данных.

Выполнение работы.

В лабораторной работе использован код предыдущей лабораторной работы: класс `Matrix` и функция предварительной настройки по аргументам командной строки.

Вместо очереди с грубой блокировкой `SyncBuffer` был создан шаблонный класс очереди `Queue<T>`. В основе очереди также лежит массив ограниченной длины с индексами первой занятой и первой свободной ячеек. В массиве хранятся указатели на объекты нужного типа, пустая клетка обозначается нулевым указателем. Указатели в массиве и индексы чтения и записи атомарны.

Процедуры чтения и записи используют, помимо атомарных операций чтения и записи, атомарный примитив `compare_and_swap`: поток сначала запоминает текущую позицию индекса чтения или записи, потом с помощью `compare_and_swap` проверяет, что индекс ещё находится на той же позиции; если это верно, та же атомарная операция заменяет значение индекса очереди на следующее, не давая другим потокам обращаться к этой ячейке. После этого происходит собственно чтение или запись.

Таким образом, инвариант структуры данных выглядит следующим образом: процедуры чтения или записи в одну и ту же клетку производит одновременно не более 1 потока.

Было проведено исследование зависимости времени работы программы от количества конкурирующих потоков. Время работы программы было сопоставлено со временем работы программы, реализованной в рамках второй лабораторной работы (Во второй лабораторной работе с очередью работают только два потока, но каждый из них помещает в очередь или извлекает из неё n задач. Число n было использовано как число конкурирующих потоков). Везде в исследовании размеры складываемых матриц были взяты 500×500. На рисунках ниже приведены построенные зависимости.

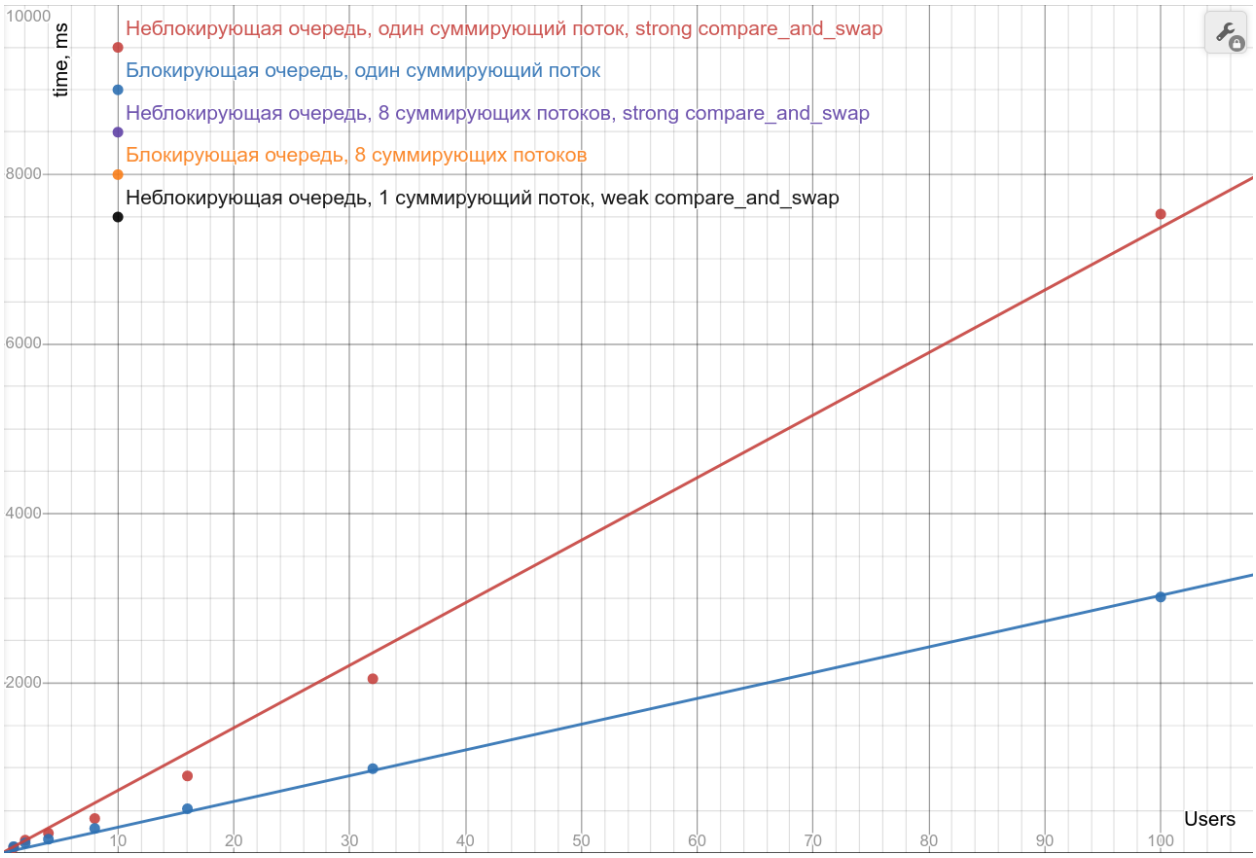

Рисунок 1 – Сравнение времени работы очередей для одного суммирующего потока

Из графиков можно видеть, что блокирующая очередь заметно быстрее неблокирующей при большом числе пользователей. Скорее всего, это объясняется тем, что потоки в неблокирующей очереди находятся в активном

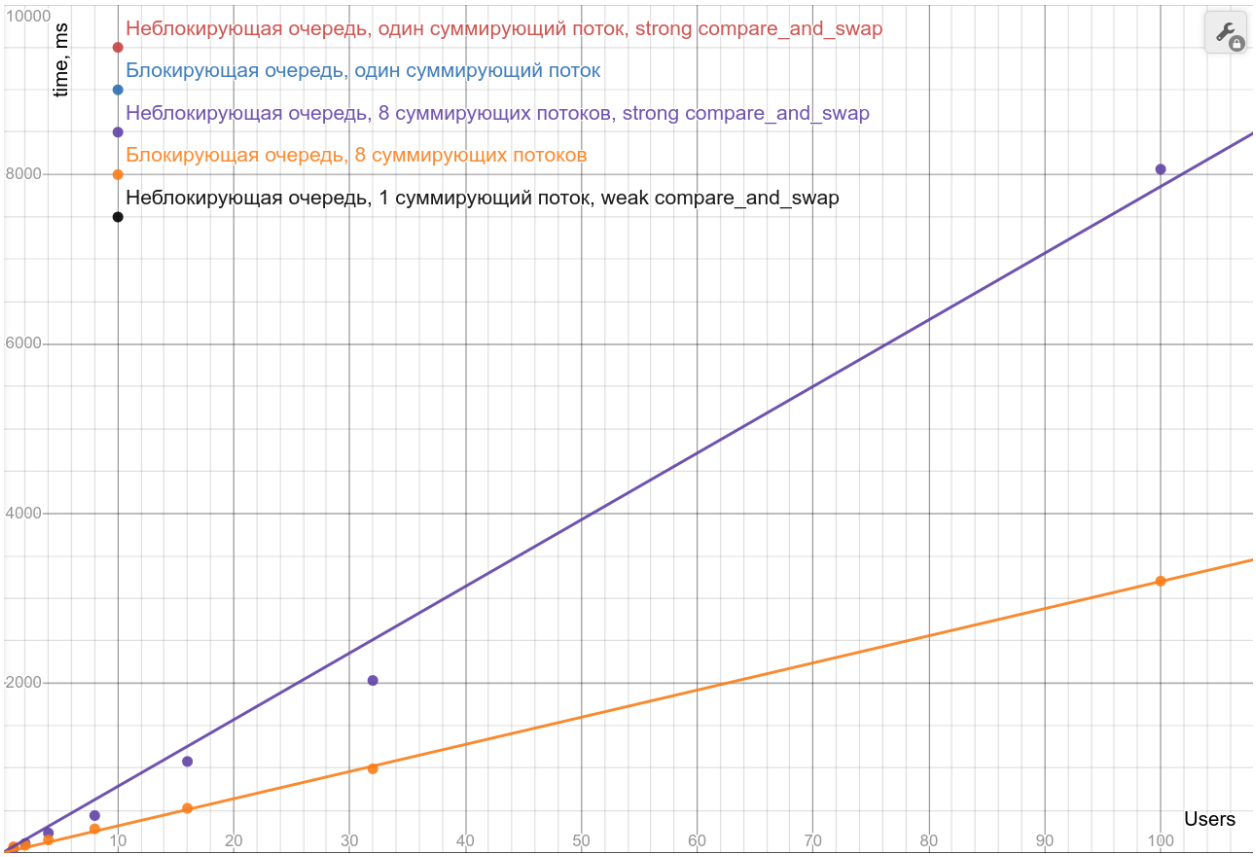

Рисунок 2 – Сравнение времени работы очередей для восьми суммирующих потоков

ожидании.

Также значение времени работы неблокирующей очереди стабильно находится под регрессионной прямой при небольшом количестве пользователей и над регрессионной прямой — при 100 пользователях. Это объясняется большими накладными расходами при переключении между большим количеством потоков.

Эффект от изменения количества суммирующих потоков и от использования другого примитива `compare_and_swap` незначителен на фоне колебаний времени работы программы от измерения к измерению.

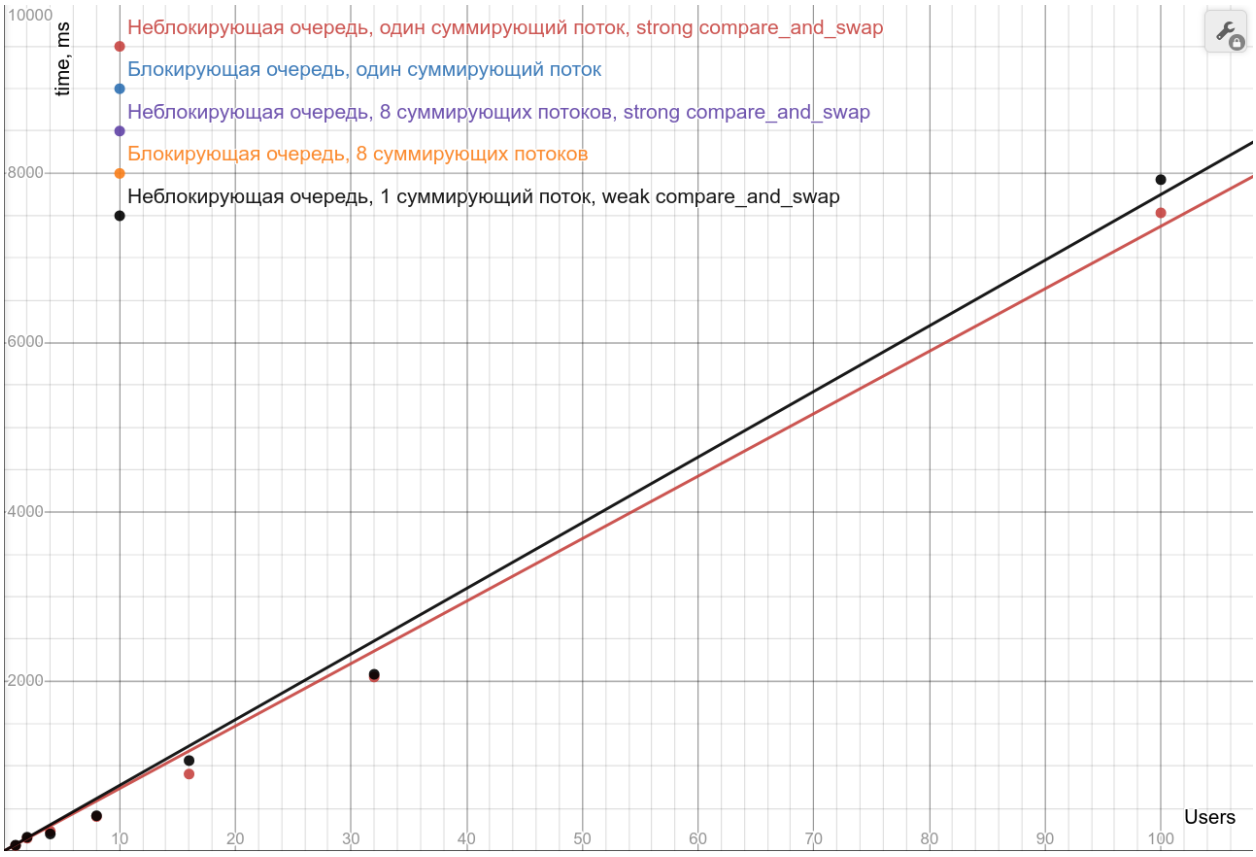

Рисунок 3 – Сравнение работы неблокирующей очереди при использовании разных cas-примитивов

Вывод.

В ходе выполнения работы были изучены принципы построения lock-free структур данных. Результатом работы стала программа, обрабатывающая данные с использованием lock-free очереди.