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

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Параллельные алгоритмы»
Тема: Реализация взаимодействия потоков по шаблону
“Производитель-потребитель”

Студентка гр. 9304 _____ Аксёнова Е.А.

Преподаватель _____ Сергеева Е.И.

Санкт-Петербург
2022

Цель работы.

Ознакомиться с использованием примитивов и механизмов синхронизации в языке программирования C++. А также решить проблему взаимодействия потоков по шаблону “Производитель-потребитель”, используя полученные знания.

Задание.

Реализовать итерационное (потенциально бесконечное) выполнение подготовки, обработки и вывода данных.

Обеспечить параллельное выполнение потоков обработки готовой порции данных, подготовки следующей порции данных и вывода предыдущих полученных результатов.

Выполнение работы.

Реализация объекта для буфера.

В классе матрицы содержится три массива размера NxM: две для сложения генерируются рандомно при создании объекта, третья заполняется во время суммирования первых двух.

Реализация буфера для решения проблем.

Класс буфер содержит массив объектов Матиц, размер буфера, индекс для считывания матриц и индекс для записи матриц. Две условные переменные: not_empty (сообщение о отсутствии матриц в буфере для производителя), not_full (сообщение о отсутствии места в буфере для производителя), а так же мьютекс для взаимоисключающего доступа к условной переменной.

Механизм produce:

Получаем замок, передавая в него мьютекс. Это обеспечивает взаимоисключающий доступ к функции. Далее, если буфер полон, то

становимся на ожидание на условной переменной. Если он не переполнен (или нас сообщили, что там появилось место через notify), записываем переданную матрицу в буфер, смещаем циклически индекс вправо для записи и увеличиваем размер буфера. Затем отпускаем замок и сообщаем, что в буфере появилось значение.

Механизм consume:

Получаем замок, передавая в него мьютекс. Это обеспечивает взаимоисключающий доступ к функции. Далее, если буфер пуст, то становимся на ожидание на условной переменной. Если он не пуст (или нас сообщили, что там появилось значение через notify), считываем переданную матрицу в буфер, смещаем циклически индекс влево для чтения и уменьшаем размер буфера. Затем отпускаем замок и сообщаем, что в буфере появилось место.

Работа с файлом.

Для обеспечения взаимно исключающего доступа к файлу был использован std::unique_lock. В него передается std::mutex file_mtx, который нужно получить, чтобы писать в файл. Так как замок может быть единовременно только у одного потока, то только один поток можем писать в файл. Так гарантируется целостность данных при записи в файл.

Выводы.

В ходе выполнения лабораторной работы было написана программа на языке программирования C++ для попарного сложения потока матриц.

С помощью примитивов и механизмов синхронизации в языке программирования C++ обеспечено параллельное выполнение потоков обработки готовой порции данных, подготовки следующей порции данных и вывода предыдущих полученных результатов.