МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 "Компьютерные науки и прикладная математика"

Кафедра 806 "Вычислительная математика и программирование"

Лабораторная работа №2

По курсу «Операционные системы»

Студент: Татаркин И. В.

Группа: М8О-208Б-23

Преподаватель: Живалев Е. А.

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2024

**Тема:** Управление потоками и синхронизация в ОС

**Цель работы:** Целью работы является приобретение практических навыков в:

* Управлении потоками в операционной системе.
* Организации синхронизации между потоками для эффективного использования многопоточности.

**Вариант:** 12. Наложить K раз фильтры эрозии и наращивания на матрицу, состоящую из вещественных чисел. На выходе получается 2 результирующие матрицы

**Задачи:**

1. Разработать программу на языке Си, реализующую многопоточную эрозию и наращивание матрицы.
2. Ограничить максимальное количество одновременно работающих потоков с использованием заданного параметра.
3. Обеспечить корректную синхронизацию потоков с помощью стандартных средств операционной системы.
4. Провести исследование зависимости ускорения и эффективности алгоритма от входных данных и количества потоков.

**Описание решения:** Программное решение представляет собой многопоточную реализацию эрозию и наращивание. Основные компоненты программы:

* **Управление потоками:** Для создания потоков используется библиотека pthread. Максимальное количество одновременно работающих потоков задается пользователем в виде параметра запуска программы.
* **Синхронизация потоков:** Для управления количеством активных потоков используется мьютекс и глобальная переменная countOfActiveThreads. Увеличение и уменьшение счетчика активных потоков синхронизировано с помощью мьютекса.
* **Алгоритм наращивания и эрозии:** Наращивание и эрозия происходит для блоков 3 на 3 внутри матрицы, что позволяет эффективно находить всех соседей конкретного элемента и заменять его на максимальный (минимальный).

Программа функционирует следующим образом:

1. Пользователь задает максимальное количество потоков и тип ввода данных
2. Основной поток вызывает функцию эрозии и наращивания, передавая в нее данные о матрице.
3. В процессе эрозии (наращивания) матрица делится на части, которые обрабатываются либо в новых потоках, либо в текущем потоке, в зависимости от текущей загрузки.
4. После завершения эрозии (наращивания) выводится 2 обработанные матрицы.

**Репозиторий:** https://github.com/IvanTatarkin/OS-Labs/tree/main/LW2

**Исходный код:** Программное обеспечение состоит из следующих файлов:

1. **main.c:** Инициализация программы, ввод данных и запуск эрозии (наращивания).
2. **utils.c:** Реализация функций эрозии (наращивания) и управления потоками.

Пример кода:

// Функция применения фильтров

void applyFilterToSegment(float\*\* matrix, float\*\* matrixCopy, int N, int x, int y, bool isErosion) {

float filterValue = matrix[x][y];

for (int i = -1; i <= 1; i++) {

for (int j = -1; j <= 1; j++) {

int nx = x + i;

int ny = y + j;

if (nx >= 0 && nx < N && ny >= 0 && ny < N) {

if (isErosion) {

filterValue = fminf(filterValue, matrix[nx][ny]);

} else {

filterValue = fmaxf(filterValue, matrix[nx][ny]);

}

}

}

}

matrixCopy[x][y] = filterValue;

}

**Пример работы:**

./lab2

Usage: ./lab2 <размер матрицы> <количество фильтров> <MAX\_THREADS>

Matrix size: 3

Number of filters (K): 2

Max threads: 4

Enter value for matrix[0][0]:

Enter value for matrix[0][0]: 0

Enter value for matrix[0][1]: 1

Enter value for matrix[0][2]: 2

Enter value for matrix[1][0]: 3

Enter value for matrix[1][1]: 4

Enter value for matrix[1][2]: 5

Enter value for matrix[2][0]: 6

Enter value for matrix[2][1]: 7

Enter value for matrix[2][2]: 8

riginal matrix:

0.00 1.00 2.00

3.00 4.00 5.00

6.00 7.00 8.00

Applying Dilation filter:

Matrix after Dilation (After 2 passes):

8.00 8.00 8.00

8.00 8.00 8.00

8.00 8.00 8.00

Applying Erosion filter:

Matrix after Erosion (After 2 passes):

0.00 0.00 0.00

0.00 0.00 0.00

0.00 0.00 0.00

**Вывод:** В ходе выполнения лабораторной работы было реализовано многопоточное наращивание (эрозия) Программа корректно ограничивает количество одновременно работающих потоков, что обеспечивает оптимальное использование многозадачности.

Исследования зависимости ускорения и эффективности алгоритма от числа потоков и объема данных показали, что увеличение числа потоков до определенной точки действительно способствует ускорению выполнения. Однако при слишком большом числе потоков эффективность алгоритма начинает снижаться, что связано с накладными расходами на управление потоками. Полученные результаты соответствуют теоретическим прогнозам.