Московский Авиационный Институт
(Национальный Исследовательский Университет)
Факультет информационных технологий и прикладной математики
Кафедра вычислительной математики и программирования

Лабораторная работа №3 по курсу
«Операционные системы»

Студент: Соколов Арсений Игоревич
Группа: М8О-207Б-21
Вариант: 15
Преподаватель: Миронов Евгений Сергеевич
Оценка: _____
Дата: _____
Подпись: _____

Москва, 2022

Содержание

Репозиторий

https://github.com/valerasaray/os

Постановка задачи

Цель работы

Приобретение практических навыков в управлении процессами в ОС, обеспечение синхронизации между потоками.

Задание

Составить программу на языке Си, обрабатывающую данные в многопоточном режиме. При обработки использовать стандартные средства создания потоков операционной системы (Windows/Unix). Ограничение потоков должно быть задано ключом запуска вашей программы.
Так же необходимо уметь продемонстрировать количество потоков, используемое вашей программой с помощью стандартных средств операционной системы.
В отчете привести исследование зависимости ускорения и эффективности алгоритма от входящих данных и количества потоков. Получившиеся результаты необходимо объяснить.

Вариант 15

Перемножение полиномов. На вход подается N-полиномов, необходимо их перемножить

Общие сведения о программе

Программа компилируется из файла main.cpp

Общий метод и алгоритм решения

Пользователь вводит в консоль количество полиномов N и количество потоков K. Если пользователь вводит K = -1, то количество потоков устанавливается равным 2 по умолчанию. После чего пользователь вводит степени полиномов и коэффициенты их членов. В основном цикле while, который выполняется, пока не останется лишь один полином, создается массив потоков размером K, в котором каждый элемент представляет собой поток.

Внутри цикла while каждый поток вызывает функцию new_pol, которая производит умножение двух полиномов на определенных индексах массива значений результирующего полинома.

Для избежания одновременного доступа к одному и тому же элементу массива значений результирующего полинома, используется схема "разделяй и властвуй". Каждый поток работает

только со своей частью массива значений результирующего полинома, поэтому нет необходимости использовать мьютексы или другие механизмы синхронизации. После завершения работы всех потоков происходит копирование значений результирующего полинома в массив значений первого множителя с помощью функции free_pol. Кроме того, в цикле while происходит синхронизация потоков с помощью функции join, чтобы дождаться завершения работы всех потоков перед переходом к следующей итерации цикла.

Исходный код

main.cpp

```
#include<iostream>
#include<fstream>
#include<thread>
#include<vector>
#include<malloc.h>
#include<unistd.h>
using namespace std;


void free_pol(int *result, int *poll, int size){
    for (int i = 0; i<size; i++){
        poll[i] = result[i]; // копирование элементов из массива значений произведения в массив значений первого множителя
        result[i] = 0; // освобождение массива значений произведения
    }
}

// функция вывода полинома
void print_pol(int *polynomial, int size){
    for (int i = size-1; i >0; i--){
            cout<<polynomial[i]<<"x^"<<i<<"+";
        }
        cout<<polynomial[0]<<endl;

}

// функция умножения полиномов
```

```
void new_pol(int *result, int *minm, int *maxm, int n, int m,
int K, int np){
    for(int i = 0; i<n; i++) {
        for(int j = (K-(i%K)+np)%K ; j < m; j = j + K){
                result [i+j] = result[i+j] + (minm[i] * maxm[j]);
            }
        }
}

int main(){
    int N; // количество полиномов
    int result_degree=0;  // степень результирующего многочлена
    cout<<"Введите количество полиномов: ";
    cin>>N;
    // проверка на количество полиномов (N >= 2)
    if (N<2){
        cout<<"Количество полиномов меньше двух";
        return 1;
    }

    int K; // количество потоков
    cout<<"\nВведите ограничение на количество потоков.\nЕсли
ограничения нет, то введите -1: ";
    cin>>K;
    // проверка на количество потоков (K >= 0, K == -1)
    if (K<0 && K != -1){
        cout << "Количество потоков отрицательно, и не равно
-1"<<endl;
        return 1;
    }

    if (K == -1) {K = 2;} // количество потоков по умолчанию

    int a, max; max = -1;
    cout<<"\nВведите степень первого полинома: ";
    cin>>a; // степень первого полинома
    int all_degrees[N]; // массив всех степеней полиномов
    all_degrees[0]= a; // добавление степени первого полинома
```

```
    result_degree = a; // приравнивание результирующей степени к
степени первого полинома

    for (int j = 1;j < N; j++){
        cout<<"Введите степень полинома номер " << j+1 << ": ";
        int degree; // степень текущего полинома
        cin>>degree;

        // определение максимальной степени
        if (degree > max) {
            max = degree;
        }

        all_degrees[j] = degree;  // добавление степени текущего
полинома
        result_degree += (degree-1);
    }

    int *pol1 = new int[result_degree]; // массив значений
первого множителя
    int *pol2 = new int [max]; // массив значений второго
множителя

    cout<<"\nВведите коэффициенты первого полинома, начиная с
члена старшей степени: ";
    for (int i = a-1; i>=0; i--){
        cin>>pol1[i]; // ввод коэффициентов первого полинома
    }
        // fflush;
    int *result = new int[result_degree]; // массив
результирующего полинома
    for(int i = 0; i<result_degree; i++){
        result[i] = 0;
    }

    int k = 1; // порядковый номер текущего полинома

    if (N!=1){
```

```
        thread th[K];

    // цикл, выполняемый пока не останется лишь один полином
    while (N > 1){

        int b;  // степень текущего результирующего полинома
        b = all_degrees[k];
        k = k + 1;  // увеличение порядкового номера текущего
полинома

        cout<<"Введите коэффициенты полинома номер " << k <<",
начиная с члена старшей степени: ";
        for (int i = b-1; i>=0; i--){
            cin>>pol2[i];
        }
        cout<<"  "; print_pol(pol1, a);// fflush; // вывод
первого множителя
        cout<<"*"<<endl;
            cout<<"  ";print_pol(pol2, b); // fflush; // вывод
второго множителя
        cout<<"_____"<<endl;

        for (int i = 0; i<K; i++){
            if (a>b){
                 th[i] = thread(new_pol, result, pol1, pol2, a,
b, K, i);
            }
            else{
                  th[i] = thread(new_pol, result, pol2, pol1, b,
a, K, i);
            }
        }
        for(int i = 0; i<K; i++){
            th[i].join(); // синхронизация потоков, чтобы
дождаться завершения каждого
        }

            for (int i = b-1; i>=0; i--){
```

```
            pol2[i] = 0;
        }
        a = a + b - 1;
        N = N - 1; // уменьшение количества полиномов
        print_pol(result, result_degree); // вывод произведения
        free_pol(result, pol1, result_degree); // очистка массива
первого множителя

            }
    }
    // освобождение выделенной памяти
      delete[] result;
      delete[] pol1;
      delete[] pol2;
}
```

Демонстрация работы программы

```
[valerasaray@valerasaray build]$ ./main
Введите количество полиномов: 2

Введите ограничение на количество потоков.
Если ограничения нет, то введите -1: 2

Введите степень первого полинома: 3
Введите степень полинома номер 2: 2

Введите коэффициенты первого полинома, начиная с члена старшей степени: 5 6 2
Введите коэффициенты полинома номер 2, начиная с члена старшей степени: 0 3
  5x^2+6x^1+2
*
  0x^1+3
---------------------------------
0x^3+15x^2+18x^1+6
```

Выводы

В ходе выполнения лабораторной работы №3 я приобрел практические навыки в управлении процессами в ОС,и в обеспечении синхронизации между потоками.