Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №3 по курсу**

**«Операционные системы»**

Группа: М80-207Б-20

Студент: Михеева К.О.

Преподаватель: Миронов Е.С.

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

Дата: 26.10.21

Москва, 2021.

# Содержание

1. Постановка задачи.
2. Общие сведения о программе.
3. Общий метод и алгоритм решения.
4. Код программы.
5. Демонстрация работы программы.
6. Вывод.

## Постановка задачи

Составить программу на языке Си, обрабатывающую данные в многопоточном режиме. При обработке использовать стандартные средства создания потоков операционной системы (Windows/Unix). Ограничение потоков должно быть задано ключом запуска программы.  
Необходимо уметь продемонстрировать количество потоков, используемых программой, с помощью стандартных средств операционной системы.

Привести исследование зависимости ускорения и эффективности алгоритма от входящих данных и количества потоков. Объяснить получившиеся результаты.

**Вариант 16:** Задаётся радиус окружности. Необходимо с помощью метода Монте-Карло рассчитать её площадь

## Общие сведения о программе

Для компиляции программы требуется указать ключ -pthread.

Для запуска программы,указываем аргументы в следующей последовательности:

1) В качестве 1 аргумента командной строки указываем радиус окружности,

2) В качестве 2 аргумента указываем количество проверяемых точек,

3) В качестве 3 аргумента указываем количество потоков.

## Общий метод и алгоритм решения

Метод Монте-Карло предполагает генерирование случайных (псевдослучайных) равномерно распределенных чисел. Этим числам сопоставляются координаты точек для рассматриваемой фигуры – квадрата, в которую вписывается круг. Если площадь квадрата *S* подсчитана, то задается число испытаний *N*, из которых *m* исходов могут оказаться внутри круга или на его границе, т.е. на окружности. Тогда площадь круга *S*0 будет определяться выражением



где *N* – число с генерированных случайных чисел, соответствующих количеству точек, находящихся в данном квадрате,*m* — число случайных чисел (точек), которые попали в круг. граница круга – это окружность, уравнение которой имеет вид



где *х0,y0* – координаты центра окружности, *х,y* – текущие значения переменных, *r* – радиус окружности.

Пусть на вход от пользователя поступило m точек и n потоков.Tогда каждый поток будет обрабатывать n/m точек. Будем хранить кол-во

точек, попавших в круг и обрабатываемых i-м потоками,в динам. массиве N.

Каждый поток обрабатывает n/m точек. Для этого генерируется случайная точка с координатами. При этом если точка лежит в пределах круга. В качестве аргумента поток принимает количество обрабатываемых точек и номер i. После завершения работы всех потоков выводится площадь, вычисленная методом Монте-Карло и стандартным методом.

## Код программы

**main.c:**

#include <stdio.h>

#include <math.h>

#include <pthread.h>

#include <stdlib.h>

#include <time.h>

int R;

int \*N;

typedef struct arguments {

int points;

int i;

} Arg;

double get\_rand() { // возвращаем случайное двойное значение от 0 до 1

return ((double) rand()) / RAND\_MAX;

}

double get\_rand\_range(double min, double max) { // возвращаем случайное двойное значение от min до max

return get\_rand() \* (max - min) + min;

}

void \*thread\_function(void \*args) { // создаем n случайных точек с размером квадрата 2 \* R и проверяем, указывает ли точка на круг

Arg \*arg = (Arg \*) args;

int n = arg->points;

int i = arg->i;

for (int j = 0; j < n; j++) {

double x = get\_rand\_range(-R, R);

double y = get\_rand\_range(-R, R);

if (x \* x + y \* y <= R \* R) {

N[i]++;

}

}

return NULL;

}

int main(int argc, char \*argv[]) {

if (argc != 4) {

printf("Syntax: ./\*executable\_file\_name\*Радиуса\_число\_точек Число\_потоков \n");

exit(1);

}

R = atoi(argv[1]);

int points\_num = atoi(argv[2]), threads\_num = atoi(argv[3]);

N = (int \*) calloc(threads\_num, sizeof(int)); // массив чисел в круге

srand(time(NULL));

pthread\_t \*threads = (pthread\_t \*) calloc(threads\_num, sizeof(pthread\_t));

if (threads == NULL) {

printf("Невозможно выделить память для потоков\n");

exit(1);

}

int points\_for\_thread = points\_num / threads\_num;

Arg a;

for (int i = 0; i < threads\_num; i++) {

a.points = points\_for\_thread + (i < (points\_num % threads\_num));

a.i = i;

if (pthread\_create(&threads[i], NULL, thread\_function, &a) != 0) {

printf("Can not create thread\n");

exit(1);

}

}

for (int i = 0; i < threads\_num; i++) {

if (pthread\_join(threads[i], NULL) != 0) {

printf("Еrror\n");

exit(1);

}

}

double n = 0;

for (int i = 0; i < threads\_num; i++) { // вычисляем точки на окружности

n += N[i] / 1.0 / points\_num;

}

printf("Monte-Carlo Circle square is %.5f\n",

(double) 4 \* R \* R \* n); // (Квадрат круга) / (Квадрат квадрата размером 2 \* R) = N / M, где M = points\_num

printf("Real Circle square is %.5f\n", (double) M\_PI \* R \* R);

free(threads);

return 0;

}

## Демонстрация работы программы

Ввод:

gcc main.c -pthread

./a.out 2 10 3

Вывод:

Квадрат круга Монте-Карло равен 11.20000

Действительный квадрат круга равен 12.56637

Ввод:

gcc main.c -pthread

./a.out 2 100 5

Вывод:

Квадрат круга Монте-Карло равен 12.48000

Действительный квадрат круга равен 12.56637

## Вывод

В данной лабораторной работы я научилась распараллеливать выполнение программы. Научилась работать и управлять несколькими потоками в рамках одного процесса, а так же обеспечивать синхронизацию между ними.

Из данной лабораторной работы могу сделать вывод, что использование многосторонности в программе бывает полезно. Такой способ позволяет разбить задачу на несколько более мелких задач, которые будут выполняться параллельно. Так же создать несколько потоков намного эффективнее, чем создать несколько процессов. Потоки делят одно адресное пространство и работают намного быстрее.