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»

Группа: М8О-210Б-23

Студент: Иванченко Макар Дмитриевич

Преподаватель: Бахарев В.Д. (ФИИТ)

Оценка: _____

Дата: 21.11.24

Москва, 2024

Постановка задачи

Вариант 15.

Составить программу на языке Си, обрабатывающую данные в многопоточном режиме. При обработки использовать стандартные средства создания потоков операционной системы (Windows/Unix). Ограничение максимального количества потоков, работающих в один момент времени, должно быть задано ключом запуска вашей программы.

Есть колода из 52 карт, рассчитать экспериментально (метод Монте-Карло) вероятность того, что сверху лежат две одинаковых карты. Количество раундов задаётся ключом программы.

Общий метод и алгоритм решения

Использованные системные вызовы:

- int pthread_create(pthread_t **thread*, const pthread_attr_t **attr*, void *(**start_routine*)(void*), void **arg*);- создание потока.
- int pthread_mutex_init(pthread_mutex_t *mutex, pthread_mutexattr_t *attr); - инициализация мьютекса
- int pthread_mutex_lock(pthread_mutex_t **mutex*); - блокировка мьютекса
- int pthread_mutex_unlock(pthread_mutex_t **mutex*); - разблокировка мьютекса
- int pthread_mutex_destroy(pthread_mutex_t **mutex*); - уничтожение мьютекса.
- Void exit(int code) – завершение программы.

Программа принимает на вход количество раундов и количество потоков. После этого создается необходимое количество потоков, каждый из которых выполняет количество раундов, разделенное на количество потоков. В каждом потоке берутся две верхние карты, после чего происходит проверка, одинаковы ли они по достоинству, и если да, то к результату прибавляется один. После этого результаты всех потоков складываются, полученное число делится на количество операций и выводится.

Код программы

main.c

```
#include "stdlib.h"
#include "pthread.h"
#include <iostream>
#include <atomic>
#include <time.h>
#include <chrono>
#include <unistd.h>
#define COUNT 1

struct input{
   int out;
   int op_count;
};

pthread_mutex_t m;
input s;

void *check(void* ptr){
   input* a = (input*)ptr;
   int temp = 0;
   drand48_data buf;
```

```
    srand48_r(time(0), &buf);
    for (int k = 0; k < a->op_count; k++)
    {
        long card1;
        lrand48_r(&buf, &card1);
        card1 = card1 % 52;
        long card2;
        lrand48_r(&buf, &card2);
        card2 = card2 % 52;

        if (card1 % 14 == card2 % 14){
            temp++;
        }
    }
    pthread_mutex_lock(&m);
    a->out += temp;
    pthread_mutex_unlock(&m);
    return NULL;
}

int main(int argsc, char** args){
    int count = atoi(args[1]);
    int count_threads = atoi(args[2]);
    srand(time(NULL));
    pthread_t* t = (pthread_t*)malloc(count_threads * sizeof(pthread_t));
    if (t == NULL){
        exit(EXIT_FAILURE);
    }

    s.out = 0;
    s.op_count = count / count_threads;

    auto start_time = std::chrono::steady_clock::now();
    int status;
    status = pthread_mutex_init(&m, NULL);
    if (status != 0){
        exit(EXIT_FAILURE);
    }

    for (int i = 0; i < count_threads; i++){
        status = pthread_create(&t[i], NULL, check, &s);
        if (status != 0){
            exit(EXIT_FAILURE);
        }
    }

    for (int i = 0; i < count_threads; i++){
        status = pthread_join(t[i], NULL);
        if (status != 0){
            exit(EXIT_FAILURE);
        }
    }

    pthread_mutex_destroy(&m);
    std::cout << (double)s.out / (double)count << '\n';
    std::chrono::steady_clock::time_point current_time = std::chrono::steady_clock::now();
    std::cout << "Program has been running for " <<
std::chrono::duration_cast<std::chrono::milliseconds>(current_time - start_time).count() << " miliseconds" <<
std::endl;
    free(t);
```

```
    return 0;
}
```

Протокол работы программы

Тестирование:

```
mak@ubuntaa:/media/sf_univer/OSlabs/lab2$ ./a.out 100000000 10
0.072516
Program has been running for 193 miliseconds
mak@ubuntaa:/media/sf_univer/OSlabs/lab2$ ./a.out 100000000 5
0.0725041
Program has been running for 227 miliseconds
mak@ubuntaa:/media/sf_univer/OSlabs/lab2$ ./a.out 100000000 1
0.0724964
Program has been running for 911 miliseconds
mak@ubuntaa:/media/sf_univer/OSlabs/lab2$ ./a.out 200000000 1
0.0724814
Program has been running for 1841 miliseconds
mak@ubuntaa:/media/sf_univer/OSlabs/lab2$ ./a.out 200000000 8
0.0724492
Program has been running for 368 miliseconds
mak@ubuntaa:/media/sf_univer/OSlabs/lab2$ ./a.out 200000000 10
0.072406
Program has been running for 357 miliseconds
```

Strace:

execve("./a.out", ["./a.out", "1000000", "5"], 0x7ffd7ae35d80 /* 57 vars */) = 0

brk(NULL) = 0x591f9195b000

arch_prctl(0x3001 /* ARCH_??? */, 0x7fff9d0a8260) = -1 EINVAL (Недопустимый аргумент)

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7644f139f000

access("/etc/ld.so.preload", R_OK) = -1 ENOENT (Нет такого файла или каталога)

openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=59683, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 59683, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7644f1390000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libstdc++.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=2260296, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 2275520, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7644f1000000

mprotect(0x7644f109a000, 1576960, PROT_NONE) = 0

mmap(0x7644f109a000, 1118208, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x9a000) = 0x7644f109a000

mmap(0x7644f11ab000, 454656, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1ab000) = 0x7644f11ab000

mmap(0x7644f121b000, 57344, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x21a000) = 0x7644f121b000

mmap(0x7644f1229000, 10432, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7644f1229000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0"..., 832) = 832

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48

pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68, 896) = 68

newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) = 0

pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784

mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7644f0c00000

mprotect(0x7644f0c28000, 2023424, PROT_NONE) = 0

mmap(0x7644f0c28000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x7644f0c28000

mmap(0x7644f0dbd000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1bd000) = 0x7644f0dbd000

mmap(0x7644f0e16000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x215000) = 0x7644f0e16000

mmap(0x7644f0e1c000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7644f0e1c000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libm.so.6", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=940560, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 942344, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7644f12a9000

mmap(0x7644f12b7000, 507904, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe000) = 0x7644f12b7000

mmap(0x7644f1333000, 372736, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x8a000) = 0x7644f1333000

mmap(0x7644f138e000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe4000) = 0x7644f138e000

close(3) = 0

openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1", O_RDONLY|O_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0"..., 832) = 832

newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=125488, ...}, AT_EMPTY_PATH) = 0

mmap(NULL, 127720, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7644f1289000

mmap(0x7644f128c000, 94208, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x3000) = 0x7644f128c000

mmap(0x7644f12a3000, 16384, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
0x1a000) = 0x7644f12a3000

mmap(0x7644f12a7000, 8192, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1d000) = 0x7644f12a7000

close(3) = 0

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7644f1287000

arch_prctl(ARCH_SET_FS, 0x7644f12883c0) = 0

set_tid_address(0x7644f1288690) = 7219

set_robust_list(0x7644f12886a0, 24) = 0

rseq(0x7644f1288d60, 0x20, 0, 0x53053053) = 0

mprotect(0x7644f0e16000, 16384, PROT_READ) = 0

mprotect(0x7644f12a7000, 4096, PROT_READ) = 0

mprotect(0x7644f138e000, 4096, PROT_READ) = 0

mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x7644f1285000

mprotect(0x7644f121b000, 45056, PROT_READ) = 0

mprotect(0x591f9087a000, 4096, PROT_READ) = 0

mprotect(0x7644f13d9000, 8192, PROT_READ) = 0

prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0

munmap(0x7644f1390000, 59683) = 0

getrandom("\x5f\xcd\xc3\x81\x06\xe5\x25\xa3", 8, GRND_NONBLOCK) = 8

brk(NULL) = 0x591f9195b000

brk(0x591f9197c000) = 0x591f9197c000

futex(0x7644f122977c, FUTEX_WAKE_PRIVATE, 2147483647) = 0

rt_sigaction(SIGRT_1, {sa_handler=0x7644f0c91870, sa_mask=[],
sa_flags=SA_RESTORER|SA_ONSTACK|SA_RESTART|SA_SIGINFO, sa_restorer=0x7644f0c42520},
NULL, 8) = 0

rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0

mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7644f0200000

mprotect(0x7644f0201000, 8388608, PROT_READ|PROT_WRITE) = 0

rt_sigprocmask(SIG_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE
_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID,
child_tid=0x7644f0a00910, parent_tid=0x7644f0a00910, exit_signal=0, stack=0x7644f0200000,
stack_size=0x7fff00, tls=0x7644f0a00640} => {parent_tid=[7220]}, 88) = 7220

rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7644ef800000

mprotect(0x7644ef801000, 8388608, PROT_READ|PROT_WRITE) = 0

rt_sigprocmask(SIG_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE
_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID,
child_tid=0x7644f0000910, parent_tid=0x7644f0000910, exit_signal=0, stack=0x7644ef800000,
stack_size=0x7fff00, tls=0x7644f0000640} => {parent_tid=[7221]}, 88) = 7221

rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7644eee00000

mprotect(0x7644eee01000, 8388608, PROT_READ|PROT_WRITE) = 0

rt_sigprocmask(SIG_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE
_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID,
child_tid=0x7644ef600910, parent_tid=0x7644ef600910, exit_signal=0, stack=0x7644eee00000,
stack_size=0x7fff00, tls=0x7644ef600640} => {parent_tid=[7222]}, 88) = 7222

rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) =
0x7644ee400000

mprotect(0x7644ee401000, 8388608, PROT_READ|PROT_WRITE) = 0

rt_sigprocmask(SIG_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE
_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID,

child_tid=0x7644eec00910, parent_tid=0x7644eec00910, exit_signal=0, stack=0x7644ee400000, stack_size=0x7fff00, tls=0x7644eec00640} => {parent_tid=[7223]}, 88) = 7223

rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0

mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x7644eda00000

mprotect(0x7644eda01000, 8388608, PROT_READ|PROT_WRITE) = 0

rt_sigprocmask(SIG_BLOCK, ~[], [], 8) = 0

clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT_SETTID|CLONE_CHILD_CLEARTID, child_tid=0x7644ee200910, parent_tid=0x7644ee200910, exit_signal=0, stack=0x7644eda00000, stack_size=0x7fff00, tls=0x7644ee200640} => {parent_tid=[7224]}, 88) = 7224

rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0

futex(0x7644f0a00910, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 7220, NULL, FUTEX_BITSET_MATCH_ANY) = 0

futex(0x7644f0000910, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 7221, NULL, FUTEX_BITSET_MATCH_ANY) = 0

futex(0x7644ee200910, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 7224, NULL, FUTEX_BITSET_MATCH_ANY) = 0

munmap(0x7644f0200000, 8392704)   = 0

newfstatat(1, "", {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}, AT_EMPTY_PATH) = 0

write(1, "0.071875\n", 90.071875

)   = 9

write(1, "Program has been running for 3 m"..., 43Program has been running for 3 miliseconds

) = 43

exit_group(0)   = ?

+++ exited with 0 +++

Число потоков	Время выполнения, мс	Ускорение	Эффективность
1	953	1	1
2	486	1, 96	0,98
5	228	4, 17	0,834
7	201	4, 74	0,677
10(кол-во ядер в системе)	185	5,15	0,515
15	183	5,2	0,34

Объяснение.

Увеличение количества потоков уменьшает время выполнения программы за счёт вычислений, выполняемых параллельно. С каждым новым добавленный потоком, эффективность снижается. Это

связано с тем, что потоки уменьшают время выполнения при помощи дополнительной нагрузки на ЦП, который может производить конечное количество операций в секунду. После превышения числа потоков над количеством ядер прирост эффективности практически отсутствует, так как параллельные потоки фактически выполняются последовательно.

Вывод

В ходе лабораторной работе я приобрел базовые навыки по работе с потоками в си. Помимо этого, я изучил основные принципы параллельного программирования, а также применил эту концепцию на практике. В ходе выполнения лабораторной работы я столкнулся с трудностями, связанными с генераций псевдослучайных чисел, так как использовал непотокобезопасный генератор.