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-214Б-23

Студент: Заваротный А. А.

Преподаватель: Бахарев В. Д.

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

Дата: 28.10.24

Москва, 2024

**Постановка задачи**

**Вариант 20.**

**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. Найти в большом целочисленном массиве минимальный и максимальный элементы.**

**Общий метод и алгоритм решения**

Использованные системные вызовы:

* pthread\_t threads[num\_threads]; - объявляет массив потоков.
* pthread\_create(&threads[i], NULL, Find\_min\_max, &thread\_data\_array[i]); - создаёт новый поток.
* pthread\_join(threads[i], NULL); - ожидание завершения конкретного потока.

Решение:

1. Объявляю функцию, которая будет выполняться внутри потока.
2. Считываю количество потоков и размер массива.
3. Заполняю массив чисел случайным образом, используя функцию rand().
4. Инициализирую массив потоков, массив данных потоков и переменные для максимального и минимального элементов.
5. В цикле заполняю данные потока и запускаю поток.
6. В цикле жду завершения каждого потока и сравнивыаю его ответ с текущим.
7. Вывожу ответ.

**Код программы**

**client.c**

#include <stdint.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <ctype.h>

#include <time.h>

#include <stdio.h>

#include <pthread.h>

**typedef** **struct** {

**int** \*array;

**int** ar\_max;

**int** ar\_min;

**int** start;

**int** end;

} thread\_data;

**void**\* Find\_min\_max(**void**\* arg) {

thread\_data \*data = (thread\_data \*)arg;

**int** \*array = data->array;

**int** start = data->start, end = data->end;

**int** ar\_max = array[start], ar\_min = array[start];

**for** (size\_t i = start; i < end; i++) {

**if** (array[i] > ar\_max) {

ar\_max = array[i];

}

**else** **if** (array[i] < ar\_min) {

ar\_min = array[i];

}

}

data->ar\_max = ar\_max;

data->ar\_min = ar\_min;

pthread\_exit(NULL);

**return** NULL;

}

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

**int** ind\_array = atoi(argv[2]);

**int** \*array = (**int**\*) malloc(ind\_array \* **sizeof** (**int**));

**if** (array == NULL) {

**const** **char** msg[] = "error: failed to allocate memory\n";

write(STDERR\_FILENO, msg, **sizeof**(msg));

exit(EXIT\_FAILURE);

}

srand(time(NULL));

**for** (**int** i = 0; i < ind\_array; i++) {

array[i] = rand() % 2000 - 1000;

//printf("%d ", array[i]);

}

//printf("\n");

**int** num\_of\_threads = atoi(argv[1]);

pthread\_t threads[num\_of\_threads];

thread\_data thread\_data\_array[num\_of\_threads];

**for** (**int** i = 0; i < num\_of\_threads; i++) {

thread\_data\_array[i].array = array;

thread\_data\_array[i].start = ind\_array / num\_of\_threads \* i;

**if** (i == num\_of\_threads - 1){

thread\_data\_array[i].end = ind\_array;

}

**else**{

thread\_data\_array[i].end = thread\_data\_array[i].start + ind\_array / num\_of\_threads;

}

**if** (pthread\_create(&threads[i], NULL, Find\_min\_max, &thread\_data\_array[i]) != 0) {

**const** **char** msg[] = "error: failed to create thread\n";

write(STDERR\_FILENO, msg, **sizeof**(msg));

exit(EXIT\_FAILURE);

}

}

**int** arr\_mx = -40000, arr\_mn = 40000;

**for** (**int** i = 0; i < num\_of\_threads; i++) {

pthread\_join(threads[i], NULL);

**if** (arr\_mx < thread\_data\_array[i].ar\_max) {

arr\_mx = thread\_data\_array[i].ar\_max;

}

**else** **if** (arr\_mn > thread\_data\_array[i].ar\_min) {

arr\_mn = thread\_data\_array[i].ar\_min;

}

}

printf("Min = %d\nMax = %d\n", arr\_mn, arr\_mx);

free(array);

**return** 0;

}

**Протокол работы программы**

**Тестирование:**

$ cc -o client client.c -lm -pthread

$ ./client 2 50

Min = -801

Max = 859

|  |  |  |  |
| --- | --- | --- | --- |
| **Число потоков** | **Время исполнения (мс)** | **Ускорение** | **Эффективность** |
| 1 | 8 | 1 | 1 |
| 2 | 7 | 1,14 | 0,57 |
| 3 | 5 | 1,6 | 0,53 |
| 4 | 8 | 1 | 0,25 |
| 5 | 12 | 0,66 | 0,13 |
| 6 | 8 | 1 | 0,16 |

ind\_array = 50000000

|  |  |  |  |
| --- | --- | --- | --- |
| **Число потоков** | **Время исполнения (мс)** | **Ускорение** | **Эффективность** |
| 1 | 1815 | 1 | 1 |
| 2 | 1490 | 1,21 | 0,6 |
| 3 | 1454 | 1,25 | 0,41 |
| 4 | 1436 | 1,26 | 0,31 |
| 5 | 1429 | 1,27 | 0,25 |
| 6 | 1419 | 1,28 | 0,21 |

Num\_Points = 500000000

|  |  |  |  |
| --- | --- | --- | --- |
| **Число потоков** | **Время исполнения (мс)** | **Ускорение** | **Эффективность** |
| 1 | 16396 | 1 | 1 |
| 2 | 15877 | 1,03 | 0,52 |
| 3 | 15005 | 1,09 | 0,36 |
| 4 | 13751 | 1,19 | 0,29 |
| 5 | 13566 | 1,21 | 0,24 |
| 6 | 13460 | 1,22 | 0,2 |

**Strace:**

$ strace -f ./client 2 50

execve("./client", ["./client", "2", "50"], 0x7fff4a47f088 /\* 26 vars \*/) = 0

brk(NULL) = 0x562b3565f000

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fa9188ad000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, ***{***st\_mode=S\_IFREG|0644, st\_size=20071, ...***}***) = 0

mmap(NULL, 20071, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fa9188a8000

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\0\220\243\2\0\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

fstat(3, ***{***st\_mode=S\_IFREG|0755, st\_size=2125328, ...***}***) = 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, 2170256, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fa918696000

mmap(0x7fa9186be000, 1605632, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7fa9186be000

mmap(0x7fa918846000, 323584, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1b0000) = 0x7fa918846000

mmap(0x7fa918895000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1fe000) = 0x7fa918895000

mmap(0x7fa91889b000, 52624, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fa91889b000

close(3) = 0

mmap(NULL, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fa918693000

arch\_prctl(ARCH\_SET\_FS, 0x7fa918693740) = 0

set\_tid\_address(0x7fa918693a10) = 58965

set\_robust\_list(0x7fa918693a20, 24) = 0

rseq(0x7fa918694060, 0x20, 0, 0x53053053) = 0

mprotect(0x7fa918895000, 16384, PROT\_READ) = 0

mprotect(0x562b1b5ea000, 4096, PROT\_READ) = 0

mprotect(0x7fa9188e5000, 8192, PROT\_READ) = 0

prlimit64(0, RLIMIT\_STACK, NULL, ***{***rlim\_cur=8192\*1024, rlim\_max=RLIM64\_INFINITY***}***) = 0

munmap(0x7fa9188a8000, 20071) = 0

getrandom("\xa9\xae\x34\x4b\x7a\x80\x70\x27", 8, GRND\_NONBLOCK) = 8

brk(NULL) = 0x562b3565f000

brk(0x562b35680000) = 0x562b35680000

rt\_sigaction(SIGRT\_1, ***{***sa\_handler=0x7fa91872f520, sa\_mask=[], sa\_flags=SA\_RESTORER|SA\_ONSTACK|SA\_RESTART|SA\_SIGINFO, sa\_restorer=0x7fa9186db320***}***, 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) = 0x7fa917e92000

mprotect(0x7fa917e93000, 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=0x7fa918692990, parent\_tid=0x7fa918692990, exit\_signal=0, stack=0x7fa917e92000, stack\_size=0x7fff80, tls=0x7fa9186926c0***}***strace: Process 58966 attached

=> ***{***parent\_tid=[58966]***}***, 88) = 58966

[pid 58966] rseq(0x7fa918692fe0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 58965] rt\_sigprocmask(SIG\_SETMASK, [], <unfinished ...>

[pid 58966] <... rseq resumed>) = 0

[pid 58965] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 58966] set\_robust\_list(0x7fa9186929a0, 24 <unfinished ...>

[pid 58965] mmap(NULL, 8392704, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS|MAP\_STACK, -1, 0 <unfinished ...>

[pid 58966] <... set\_robust\_list resumed>) = 0

[pid 58965] <... mmap resumed>) = 0x7fa917691000

[pid 58966] rt\_sigprocmask(SIG\_SETMASK, [], <unfinished ...>

[pid 58965] mprotect(0x7fa917692000, 8388608, PROT\_READ|PROT\_WRITE <unfinished ...>

[pid 58966] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 58965] <... mprotect resumed>) = 0

[pid 58966] openat(AT\_FDCWD, "/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC <unfinished ...>

[pid 58965] futex(0x7fa9188e7a58, FUTEX\_WAIT\_PRIVATE, 2, NULL <unfinished ...>

[pid 58966] <... openat resumed>) = 3

[pid 58966] fstat(3, ***{***st\_mode=S\_IFREG|0644, st\_size=20071, ...***}***) = 0

[pid 58966] mmap(NULL, 20071, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fa9188a8000

[pid 58966] close(3) = 0

[pid 58966] mmap(NULL, 134217728, PROT\_NONE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7fa90f691000

[pid 58966] munmap(0x7fa90f691000, 9891840) = 0

[pid 58966] munmap(0x7fa914000000, 57217024) = 0

[pid 58966] mprotect(0x7fa910000000, 135168, PROT\_READ|PROT\_WRITE) = 0

[pid 58966] openat(AT\_FDCWD, "/lib/x86\_64-linux-gnu/libgcc\_s.so.1", O\_RDONLY|O\_CLOEXEC) = 3

[pid 58966] 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\0"..., 832) = 832

[pid 58966] fstat(3, ***{***st\_mode=S\_IFREG|0644, st\_size=183024, ...***}***) = 0

[pid 58966] mmap(NULL, 185256, PROT\_READ, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fa917663000

[pid 58966] mmap(0x7fa917667000, 147456, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x4000) = 0x7fa917667000

[pid 58966] mmap(0x7fa91768b000, 16384, PROT\_READ, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x28000) = 0x7fa91768b000

[pid 58966] mmap(0x7fa91768f000, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x2b000) = 0x7fa91768f000

[pid 58966] close(3) = 0

[pid 58966] mprotect(0x7fa91768f000, 4096, PROT\_READ) = 0

[pid 58966] futex(0x7fa9188e7a58, FUTEX\_WAKE\_PRIVATE, 1 <unfinished ...>

[pid 58965] <... futex resumed>) = 0

[pid 58966] <... futex resumed>) = 1

[pid 58965] futex(0x7fa9188e7a58, FUTEX\_WAKE\_PRIVATE, 1 <unfinished ...>

[pid 58966] munmap(0x7fa9188a8000, 20071 <unfinished ...>

[pid 58965] <... futex resumed>) = 0

[pid 58966] <... munmap resumed>) = 0

[pid 58965] rt\_sigprocmask(SIG\_BLOCK, ~[], <unfinished ...>

[pid 58966] futex(0x7fa917690230, FUTEX\_WAKE\_PRIVATE, 2147483647 <unfinished ...>

[pid 58965] <... rt\_sigprocmask resumed>[], 8) = 0

[pid 58966] <... futex resumed>) = 0

[pid 58965] clone3(***{***flags=CLONE\_VM|CLONE\_FS|CLONE\_FILES|CLONE\_SIGHAND|CLONE\_THREAD|CLONE\_SYSVSEM|CLONE\_SETTLS|CLONE\_PARENT\_SETTID|CLONE\_CHILD\_CLEARTID, child\_tid=0x7fa917e91990, parent\_tid=0x7fa917e91990, exit\_signal=0, stack=0x7fa917691000, stack\_size=0x7fff80, tls=0x7fa917e916c0***}*** <unfinished ...>

[pid 58966] rt\_sigprocmask(SIG\_BLOCK, ~[RT\_1], NULL, 8) = 0

strace: Process 58967 attached

[pid 58965] <... clone3 resumed> => ***{***parent\_tid=[58967]***}***, 88) = 58967

[pid 58966] madvise(0x7fa917e92000, 8368128, MADV\_DONTNEED <unfinished ...>

[pid 58965] rt\_sigprocmask(SIG\_SETMASK, [], <unfinished ...>

[pid 58967] rseq(0x7fa917e91fe0, 0x20, 0, 0x53053053 <unfinished ...>

[pid 58965] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 58966] <... madvise resumed>) = 0

[pid 58965] futex(0x7fa918692990, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 58966, NULL, FUTEX\_BITSET\_MATCH\_ANY <unfinished ...>

[pid 58967] <... rseq resumed>) = 0

[pid 58966] exit(0 <unfinished ...>

[pid 58967] set\_robust\_list(0x7fa917e919a0, 24 <unfinished ...>

[pid 58966] <... exit resumed>) = ?

[pid 58967] <... set\_robust\_list resumed>) = 0

[pid 58966] +++ exited with 0 +++

[pid 58965] <... futex resumed>) = 0

[pid 58967] rt\_sigprocmask(SIG\_SETMASK, [], <unfinished ...>

[pid 58965] futex(0x7fa917e91990, FUTEX\_WAIT\_BITSET|FUTEX\_CLOCK\_REALTIME, 58967, NULL, FUTEX\_BITSET\_MATCH\_ANY <unfinished ...>

[pid 58967] <... rt\_sigprocmask resumed>NULL, 8) = 0

[pid 58967] rt\_sigprocmask(SIG\_BLOCK, ~[RT\_1], NULL, 8) = 0

[pid 58967] madvise(0x7fa917691000, 8368128, MADV\_DONTNEED) = 0

[pid 58967] exit(0) = ?

[pid 58965] <... futex resumed>) = 0

[pid 58967] +++ exited with 0 +++

fstat(1, ***{***st\_mode=S\_IFCHR|0620, st\_rdev=makedev(0x88, 0), ...***}***) = 0

write(1, "Min = -992\n", ***11Min*** = -992

) = 11

write(1, "Max = 941\n", ***10Max*** = 941

) = 10

exit\_group(0) = ?

+++ exited with 0 +++

**Вывод**

В результате выполнения лабораторной работы удалось познакомиться с многопоточным программированием. Программа успешно реализует задачу обработки данных в многопоточном режиме с использованием стандартных средств операционной системы. Были изучены базовые системные вызовы для создания и управления потоками.