Домашнее задание 4

Корнилов Георгий, вариант 11

ФКН ПИ НИУ ВШЭ, группа БПИ-197

1 Текст задачи

Определить индексы $i, j (i \neq j)$, для которых выражение $A[i] - A[i+1] + A[i+2] - A[i+3] + \ldots \pm A[j]$ имеет максимальное значение. Входные данные: массив чисел А, произвольной длины большей 10. Количество потоков не является параметром задачи.

2 Методы решения задачи

Воспользуемся парадигмой итеративного параллелизма, где «цикл» по i идет от 0 до $n-2$.

Внутри каждого потока идет цикл по j от $i+1$ до $n-1$, в котором считается сумма $A[i] - A[i+1] + A[i+2] - A[i+3] + \ldots \pm A[j]$ и обновляется максимальны индекс j для данного индекса i с значением максимальной суммы потока cur_max_sum.

После выполнения цикла обновляется значение максимальной суммы max_sum значением cur_max_sum. Затем выводится информация о результатах вычисления потока.

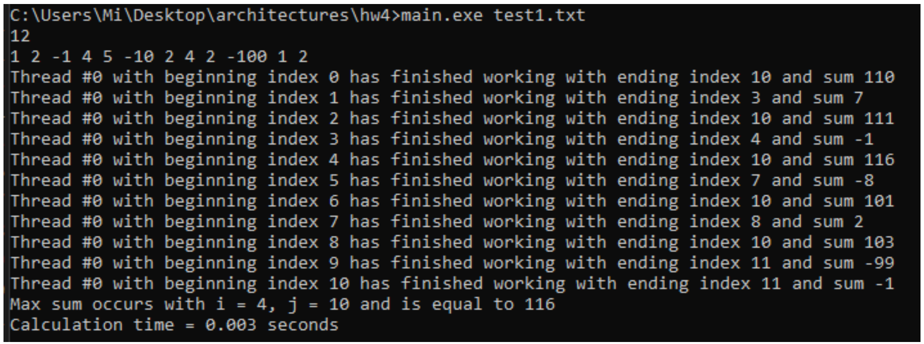
3 Исходный код программы

```
#include <iostream>
#include <ctime>
#include <fstream>
#include <omp.h>

int main(int argc, char **argv) {
    if (argc != 2)
    {
        std::cout << "Wrong usage: main.exe <input_path>" << std::endl;
    }

    size_t arr_size;
    int *arr;
int max_i = -1, max_j;
long long max_sum;

char *file_name = argv[1];
std::ifstream input(file_name);
if (!input.is_open()) {
    std::cout << "wrong file" << std::endl;
    return 1;
}
input >> arr_size;

if (arr_size <= 10) {
    std::cout << "Incorrect size of vector = " << arr_size << std::
        endl;
    return 1;
}

arr = new int[arr_size];

try {
    for (size_t i = 0; i < arr_size; i++) {
        input >> arr[i];
    }
}
catch (...)
{
    std::cout << "wrong content of file" << std::endl;
    input.close();
    return 1;
}
input.close();

std::cout << arr_size << std::endl;
for (size_t i = 0; i < arr_size; i++) {
    std::cout << arr[i] << " ";
}
std::cout << std::endl;

clock_t startTime = clock();

#pragma omp parallel for
#pragma omp shared(arr, arr_size)
for (size_t i = 0; i < arr_size - 1; i++) {
    int max_index = i + 1;

    long long cur_sum = arr[i] - arr[i + 1];
    long long cur_max_sum = cur_sum;

    for (size_t offset = 2; i + offset < arr_size; offset++) {
        int elem = arr[i + offset];
```

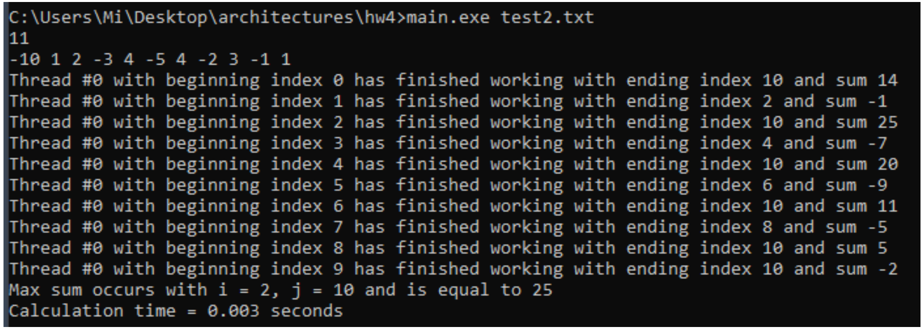
```
            if (offset % 2 == 0) {
                cur_sum += elem;
            }
            else {
                cur_sum -= elem;
            }
            if (cur_sum >= cur_max_sum) {
                cur_max_sum = cur_sum;
                max_index = i + offset;
            }
        }

        #pragma critical
        {
            if (max_i == -1 || cur_max_sum > max_sum) {
                max_i = i;
                max_j = max_index;
                max_sum = cur_max_sum;
            }
        }

        std::cout << "Thread with beginning index " << i << " has
            finished working with ending index " << max_index << " and
            sum " << cur_max_sum << std::endl;
    }

    clock_t endTime = clock();
    std::cout << "Max sum occurs with i = " << max_i << ", j = " << max_j
         << " and is equal to " << max_sum << std::endl;
    std::cout << "Calculation time = " << 1.0 * (endTime - startTime) /
        CLOCKS_PER_SEC << " seconds" << std::endl;

    delete[] arr;
    return 0;
}
```

4 Примеры работы программы

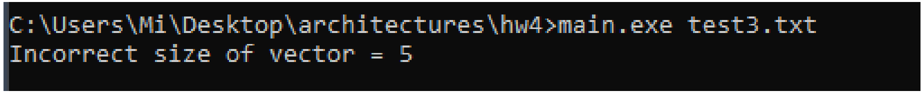
Примеры работы программы на раных тестовых данных (тестовые данные приложены по ссылке):



```
C:\Users\Mi\Desktop\architectures\hw4>main.exe test1.txt
12
1 2 -1 4 5 -10 2 4 2 -100 1 2
Thread #0 with beginning index 0 has finished working with ending index 10 and sum 110
Thread #0 with beginning index 1 has finished working with ending index 3 and sum 7
Thread #0 with beginning index 2 has finished working with ending index 10 and sum 111
Thread #0 with beginning index 3 has finished working with ending index 4 and sum -1
Thread #0 with beginning index 4 has finished working with ending index 10 and sum 116
Thread #0 with beginning index 5 has finished working with ending index 7 and sum -8
Thread #0 with beginning index 6 has finished working with ending index 10 and sum 101
Thread #0 with beginning index 7 has finished working with ending index 8 and sum 2
Thread #0 with beginning index 8 has finished working with ending index 10 and sum 103
Thread #0 with beginning index 9 has finished working with ending index 11 and sum -99
Thread #0 with beginning index 10 has finished working with ending index 11 and sum -1
Max sum occurs with i = 4, j = 10 and is equal to 116
Calculation time = 0.003 seconds
```



```
C:\Users\Mi\Desktop\architectures\hw4>main.exe test2.txt
11
-10 1 2 -3 4 -5 4 -2 3 -1 1
Thread #0 with beginning index 0 has finished working with ending index 10 and sum 14
Thread #0 with beginning index 1 has finished working with ending index 2 and sum -1
Thread #0 with beginning index 2 has finished working with ending index 10 and sum 25
Thread #0 with beginning index 3 has finished working with ending index 4 and sum -7
Thread #0 with beginning index 4 has finished working with ending index 10 and sum 20
Thread #0 with beginning index 5 has finished working with ending index 6 and sum -9
Thread #0 with beginning index 6 has finished working with ending index 10 and sum 11
Thread #0 with beginning index 7 has finished working with ending index 8 and sum -5
Thread #0 with beginning index 8 has finished working with ending index 10 and sum 5
Thread #0 with beginning index 9 has finished working with ending index 10 and sum -2
Max sum occurs with i = 2, j = 10 and is equal to 25
Calculation time = 0.003 seconds
```



```
C:\Users\Mi\Desktop\architectures\hw4>main.exe test3.txt
Incorrect size of vector = 5
```

5 Источники информации

1. http://www.softcraft.ru/edu/comparch/practice/thread/02-sync/

2. https://pro-prof.com/forums/topic/parallel-programming-paradigms

3. http://rsuib.cc.rsu.ru/tutor/high_performance_computing/chapter3/page02.html

4. https://docs.microsoft.com/ru-ru/cpp/parallel/openmp/reference/openmp-clauses?view=msvc-160