НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ УКРАИНЫ
“КИЕВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ”
ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

ЛАБОРАТОРНАЯ РАБОТА №5

Дисциплина: «Специальные разделы программирования»

Тема: «Параллельное программирование средствами OpenMP»

Выполнила

студентка 2 курса группы ФИ-41

Лавягина Ольга Алексеевна

Проверил

Колотий Андрей Всеволодович

Киев — 2016 года

ЗМІСТ

1 ЗАДАНИЕ

В лабораторном задании необходимо:

1) выполнить параллелизацию известного алгоритма средствами OpenMP. Для этого сначала необходимо реализовать этот алгоритм в последовательном (не параллельном) виде, а потом добавить директивы OpenMP, которые позволяют программе выполнятся параллельно;
2) построить зависимость общего времени выполнения вычислений в программе в зависимости от количества потоков, которое контролируется переменной окружения OMP_NUM_THREADS. Максимальное количество потоков, которые используются для построения зависимости, может быть разным, но не меньшим, чем $MIN\,(8, 4 \cdot N)$, где N — общее количество вычислительных ядер в системе. Для того, чтобы уменьшить влияние побочных факторов на измерение времени выполнения программы, в качестве тестовых данных следует использовать векторы достаточно большого размера. Время, которое измеряется, не должно включать в себя время, которое тратится на генерацию этих векторов;
3) исследовать зависимость времени выполнения от использования разных стратегий распределения итераций среди рабочих потоков. Необходимо исследовать стратегии STATIC, DYNAMIC и GUIDED с разными значениями параметра количества итераций. Значение параметра количества итераций следует брать из следующих рядов: {1, 2, 4, 6, 16, 32, 64} и {10% от M, 10% от M, ..., 100% от M}, где M — общее количество итераций на поток, которое определяется как полное количество итераций цикла делённое на значени OMP_NUM_THREADS.

Порядок выполнения работы:

1) проанализировать условие задачи;
2) реализовать программу согласно с заданием;

3) построить зависимость времени выполнения программы от количества рабочих потоков;
4) измерить время выполнения программы для разных стратегий распределения итераций в параллельном цикле. Объяснить результаты;
5) результаты работы оформить протоколом. Зависимость времени выполнения от количества потоков и стратегийраспределения итераций следует представить в графическом виде (график и столбиковая диаграмма, соответственно).

Задание: необходимо провести параллелизацию алгоритма из индивидуального задания.

Вариант индивидуального задания:

1. скалярное произведение векторов, которые состоят из элементов типа double.

2 ЛИСТИНГ КОДА

Файл main.h

```
struct Array {
  double *data;
  unsigned int size;
};

typedef struct Array Array;

Array* createArray(unsigned int size);
int destroyArray(Array *array);
int fillArray(Array *array);
int displayArray(Array *array);
double multiplyArrays(Array *a, Array *b);
```

Файл main.c

```
#include <stdlib.h>
#include <stdio.h>
#include <omp.h>

#include "main.h"

#define ARRAY_SIZE ((size_t)(1E8))
double omp_get_wtime(void);

Array *createArray(unsigned int size) {
  Array *a = (Array*)malloc(sizeof(Array));
  a->size = size;
  a->data = (double*)malloc(sizeof(double)*size);
  return a;
}

int destroyArray(Array *array) {
  free(array->data);
  free(array);
  return 0;
}

int fillArray(Array *array) {
  int i = array->size;
  while (i-- > 0) {
    array->data[i] = (double)rand()/RAND_MAX;
  }
}

int displayArray(Array *array) {
```

```
  unsigned int i = array->size;
  while (i-- > 0) {
    printf("%f\n", array->data[i]);
  }
}

double multiplyArrays(Array *a, Array *b) {
  if (a == NULL || b == NULL) {
    return 0;
  }

  if (a->size != b->size) {
    return 0;
  }
  unsigned int i = a->size;
  double c = 0;

  #pragma omp parallel for schedule(guided, 12500000)
  for (i = 0; i < a->size; i++) {
    c += (a->data[i])*(b->data[i]);
  }
  return c;
}

int main() {
  double t1, t2;

  Array *array_1 = createArray(ARRAY_SIZE);
  Array *array_2 = createArray(ARRAY_SIZE);
  fillArray(array_1);
  fillArray(array_2);
  //printf("%s\n", "First array");
  //displayArray(array_1);
  //printf("%s\n", "Second array");
  //displayArray(array_2);
  double result;
  t1 = omp_get_wtime();
  result = multiplyArrays(array_1, array_2);
  t2 = omp_get_wtime();
  printf("%s\n", "Result");
  printf("%f\n", result);
  destroyArray(array_1);
  destroyArray(array_2);
  printf("%e\n", t2-t1);
}
```

3 ЗАВИСИМОСТЬ ВРЕМЕНИ ВЫЧИСЛЕНИЯ ОТ КОЛИЧЕСТВА ПОТОКОВ

Для построения зависимости времени вычислений от количества потоков, которые контролируются переменной окружения OMP_NUM_THREADS (3.1), было найдено время для количества потоков от одного до восьми. Максимальное количество потоков, которые используются для постороения зависимости равно $MIN\,(8, 4 \cdot N)$, где N — общее количество вычислительный ядер в системе (равно двум).

Для того, чтобы уменьшить влияние побочных факторов на измерение времени выполнения программы, в качестве тестовых данных были использованы векторы с размером 10^8. Время, которое измеряется, включает в себя только время, которое тратится на вычисление скалярного произведения векторов.

Количество потоков	Время
1	1.35E-01
2	8.48E-02
3	9.91E-02
4	9.03E-02
5	9.10E-02
6	9.09E-02
7	1.01E-01
8	9.95E-02

Таблиця 3.1 — Время выполнения вычислений для разного количества потоков

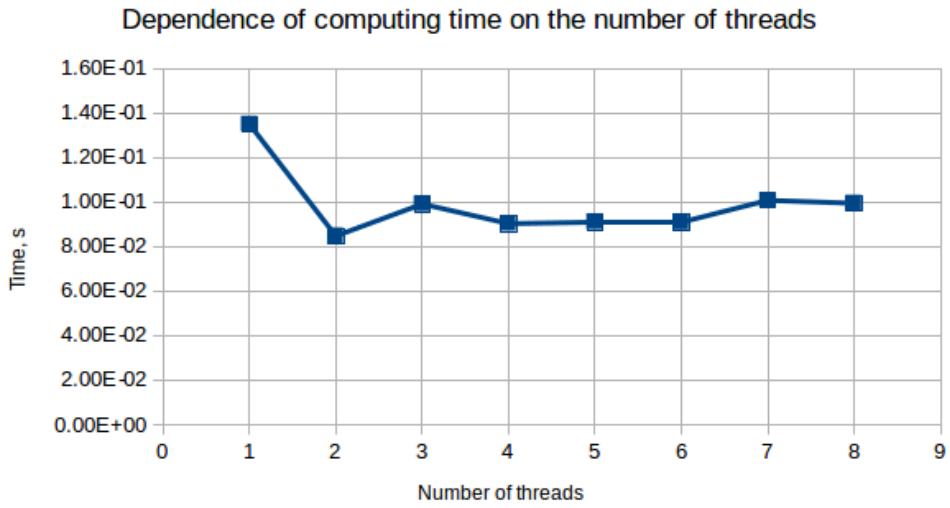

Рисунок 3.1 — Зависимость времени вычисления от количества потоков

4 СТРАТЕГИЯ STATIC

`schedule(static, block_size)`

В данной стратегии все шаги цикла делятся на блоки указанного размера. Каждой задаче назначается приблизительно равное количество блоков. Один блок назначается одной задаче. Таким образом, планировщик static назначает приблизительно равное количество шагов цикла каждой задаче.

Значение параметра количества итераций бралось из ряда 1, 2, 4, 8, 16, 32, 64 (4.1).

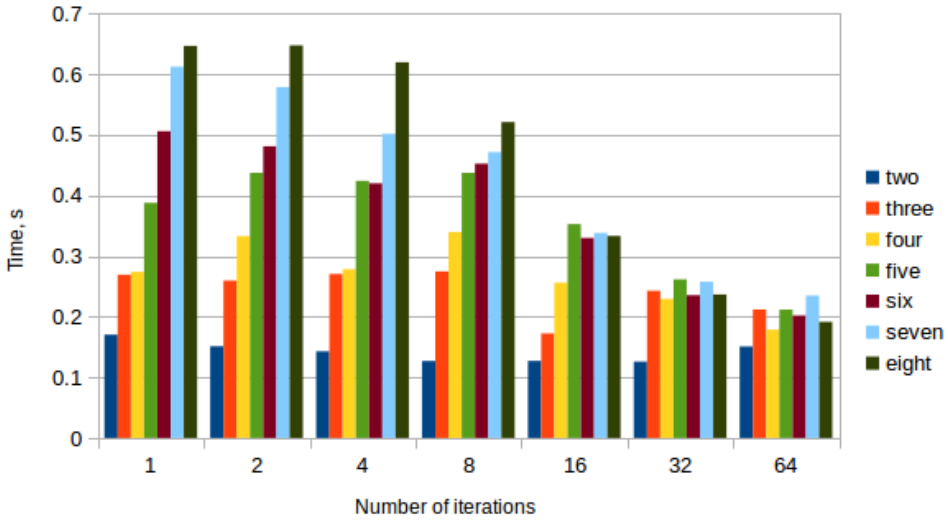

Рисунок 4.1 — Зависимость времени вычисления от количества итераций при разном количестве потоков для стратегии STATIC

В следующей зависимости (4.2) параметр количества итераций брался из ряда {10% от M, 10% от M, ..., 100% от M}, где M — общее количество итераций на поток, которое определяется как полное количество итераций цикла (размер вектора, то есть 10^8) делённое на значение OMP_NUM_THREADS (8), следовательно $M = \frac{10^8}{8} = 12500000$.

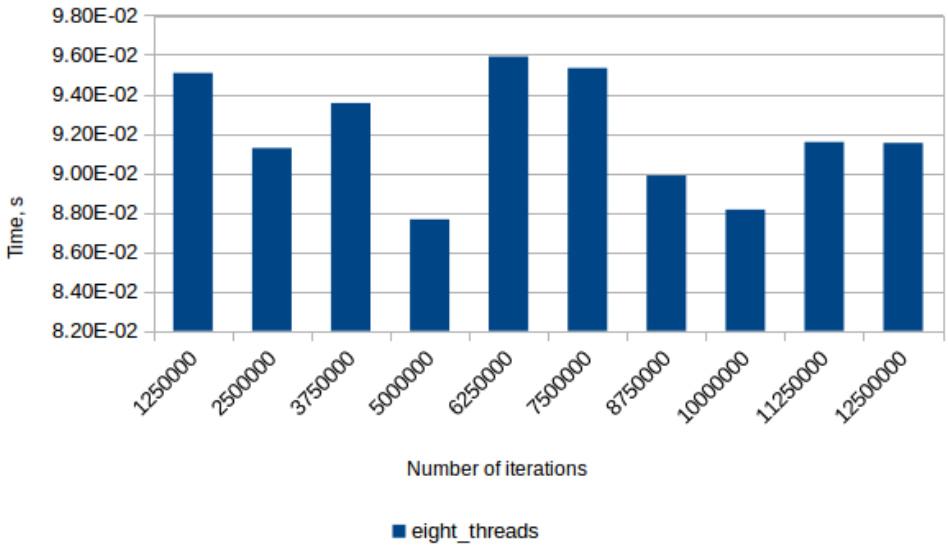

Рисунок 4.2 — Зависимость времени вычисления от количества итераций при восьми потоках для стратегии STATIC

Количество итераций	2 потока	3 потока	4 потока	5 потоков	6 потоков	7 потоков	8 потоков
1	5.59E+00	3.91E+00	3.51E+00	3.49E+00	3.56E+00	3.50E+00	3.58E+00
2	3.15E+00	2.07E+00	1.71E+00	1.67E+00	1.82E+00	1.80E+00	1.67E+00
4	1.28E+00	1.02E+00	8.53E-01	9.08E-01	8.84E-01	8.96E-01	8.52E-01
8	7.64E-01	5.47E-01	4.60E-01	4.68E-01	4.44E-01	4.63E-01	4.55E-01
16	3.26E-01	2.72E-01	2.67E-01	2.60E-01	2.65E-01	2.61E-01	2.65E-01
32	2.40E-01	1.97E-01	1.87E-01	1.66E-01	1.90E-01	1.85E-01	1.85E-01
64	1.59E-01	1.52E-01	1.43E-01	1.45E-01	1.43E-01	1.41E-01	1.41E-01

Таблиця 4.1 — Время выполнения вычислений для разного количества потоков для разного количества итераций при стратегии STATIC

Количество итераций	Время
1250000	9.51E-02
2500000	9.13E-02
3750000	9.35E-02
5000000	8.76E-02
6250000	9.59E-02
7500000	9.53E-02
8750000	8.99E-02
10000000	8.81E-02
11250000	9.16E-02
12500000	9.15E-02

Таблиця 4.2 — Время выполнения вычислений для восьми потоков для разного количества итераций для стратегии STATIC

5 СТРАТЕГИЯ DYNAMIC

`schedule(dynamic, block_size)`

Все шаги цикла делятся на блоки указанного размера. Каждой задаче назначается по одному блоку. Когда задача закончила выполнение блока, ей назначается ещё не назначенный блок, и так до того, пока ещё есть не назначенные блоки.

Таким образом, планировщик dynamic назначает шаги цикла задачам так, чтобы задачи выполняли директиву for приблизительно равное время.

Значение параметра количества итераций бралось из ряда 1, 2, 4, 8, 16, 32, 64 (5.1).

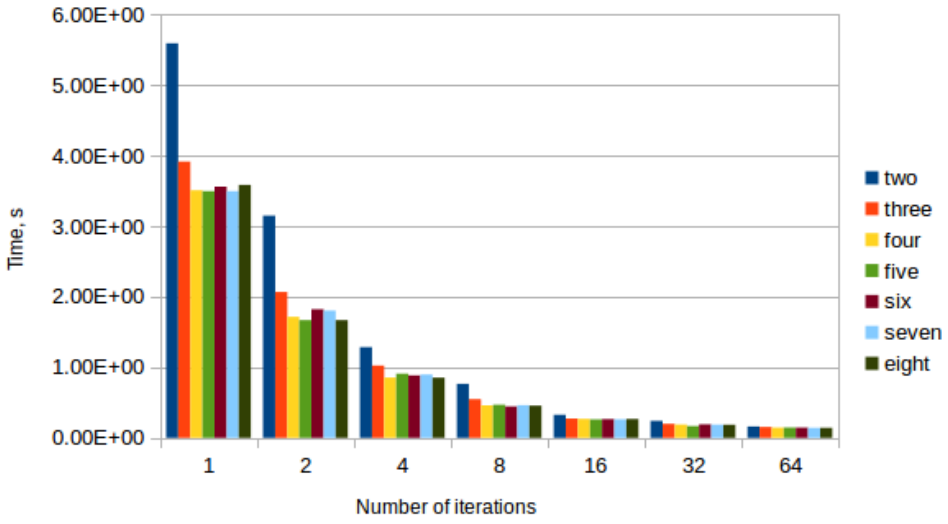

Рисунок 5.1 — Зависимость времени вычисления от количества итераций при разном количестве потоков для стратегии DYNAMIC

В следующей зависимости (5.2) параметр количества итераций брался из ряда {10% от M, 10% от M, ..., 100% от M}, где M — общее количество итераций на поток, которое определяется как полное количество итераций цикла (размер вектора, то есть 10^8) делённое на значение OMP_NUM_THREADS (8), следовательно $M = \frac{10^8}{8} = 12500000$.

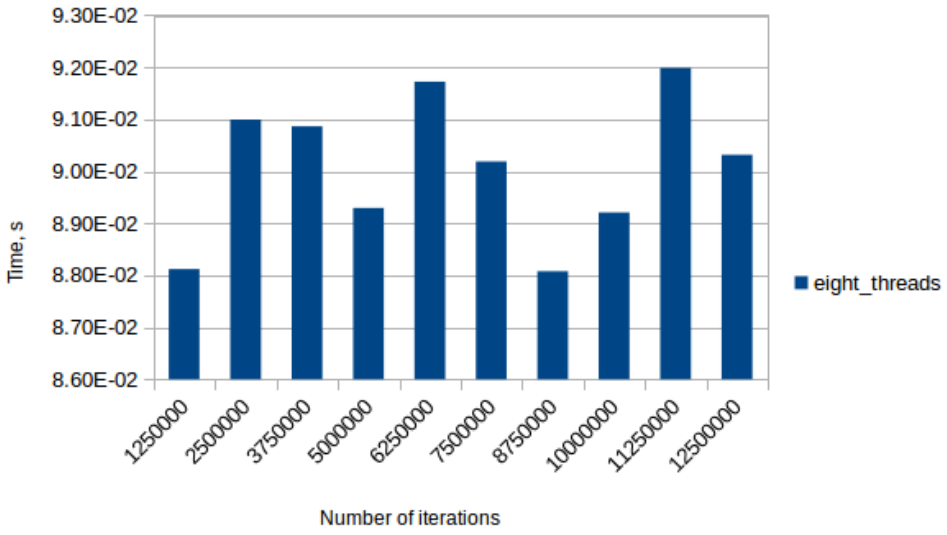

Рисунок 5.2 — Зависимость времени вычисления от количества итераций при восьми потоках для стратегии DYNAMIC

Количество итераций	2 потока	3 потока	4 потока	5 потоков	6 потоков	7 потоков	8 потоков
1	9.00E-02	9.00E-02	9.12E-02	8.80E-02	9.58E-02	9.17E-02	8.93E-02
2	8.37E-02	9.29E-02	9.02E-02	9.23E-02	8.87E-02	9.00E-02	9.05E-02
4	8.41E-02	9.13E-02	9.21E-02	9.56E-02	8.82E-02	9.23E-02	8.90E-02
8	8.55E-02	9.15E-02	9.17E-02	9.32E-02	9.20E-02	9.01E-02	8.99E-02
16	8.58E-02	9.01E-02	9.28E-02	8.94E-02	9.13E-02	9.04E-02	9.11E-02
32	8.72E-02	9.44E-02	9.10E-02	9.02E-02	8.74E-02	9.23E-02	9.19E-02
64							

Таблиця 5.1 — Время выполнения вычислений для разного количества потоков для разного количества итераций при стратегии DYNAMIC

Количество итераций	Время
1250000	8.81E-02
2500000	9.10E-02
3750000	9.09E-02
5000000	8.93E-02
6250000	9.17E-02
7500000	9.02E-02
8750000	8.81E-02
10000000	8.92E-02
11250000	9.20E-02
12500000	9.03E-02

Таблиця 5.2 — Время выполнения вычислений для восьми потоков для разного количества итераций для стратегии DYNAMIC

6 СТРАТЕГИЯ GUIDED

`schedule(guided, block_size)`

Ещё не выполненные шаги цикла делятся на блоки размером, пропорциональным $\frac{N}{P}$ (N — количество шагов цикла), но не меньше, чем размер блока. Каждый из сформированных блоков назначается задачам. Когда задача закончила выполнение блока, для неё формируется новый блок по тому же правилу. Таким образом, размеры блоков со временем уменьшаются до указанного block_size.

Значение параметра количества итераций бралось из ряда 1, 2, 4, 8, 16, 32, 64 (6.1).

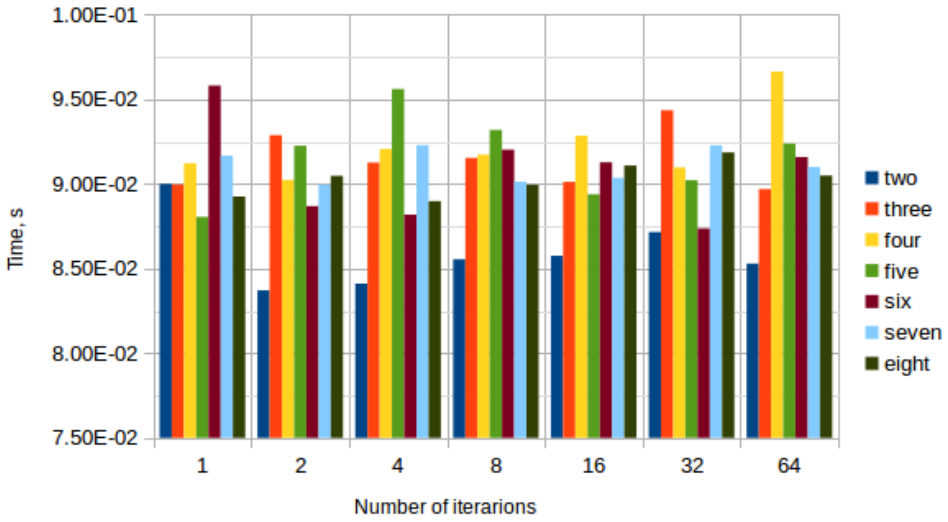

Рисунок 6.1 — Зависимость времени вычисления от количества итераций при разном количестве потоков для стратегии GUIDED

В следующей зависимости (6.2) параметр количества итераций брался из ряда {10% от M, 10% от M, ..., 100% от M}, где M — общее количество итераций на поток, которое определяется как полное количество итераций цикла (размер вектора, то есть 10^8) делённое на значение OMP_NUM_THREADS (8), следовательно $M = \frac{10^8}{8} = 12500000$.

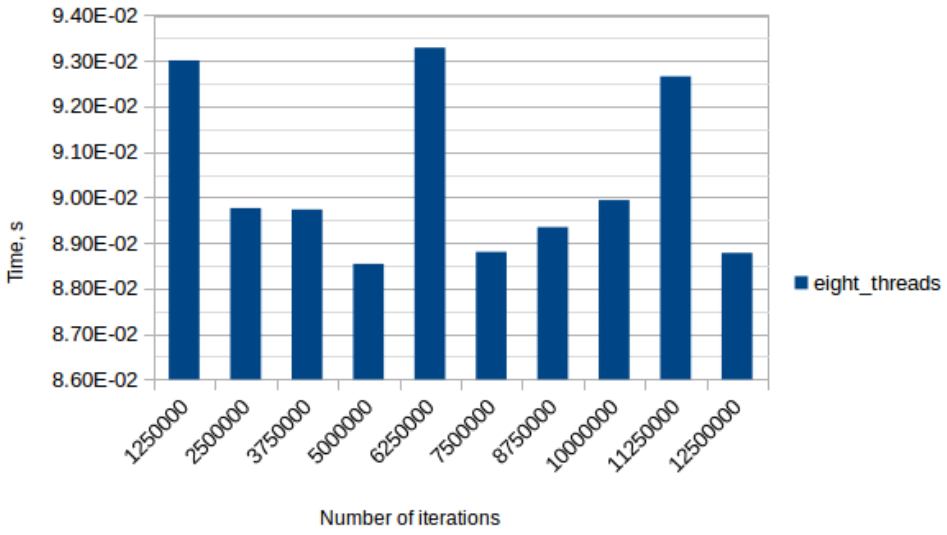

Рисунок 6.2 — Зависимость времени вычисления от количества итераций при восьми потоках для стратегии GUIDED

Количество итераций	2 потока	3 потока	4 потока	5 потоков	6 потоков	7 потоков	8 потоков
1	9.00E-02	9.00E-02	9.12E-02	8.80E-02	9.58E-02	9.17E-02	8.93E-02
2	8.37E-02	9.29E-02	9.02E-02	9.23E-02	8.87E-02	9.00E-02	9.05E-02
4	8.41E-02	9.13E-02	9.21E-02	9.56E-02	8.82E-02	9.23E-02	8.90E-02
8	8.55E-02	9.15E-02	9.17E-02	9.32E-02	9.20E-02	9.01E-02	8.99E-02
16	8.58E-02	9.01E-02	9.28E-02	8.94E-02	9.13E-02	9.04E-02	9.11E-02
32	8.72E-02	9.44E-02	9.10E-02	9.02E-02	8.74E-02	9.23E-02	9.19E-02
64	8.53E-02	8.97E-02	9.66E-02	9.24E-02	9.16E-02	9.10E-02	9.05E-02

Таблиця 6.1 — Время выполнения вычислений для разного количества потоков для разного количества итераций при стратегии GUIDED

Количество итераций	Время
1250000	9.30E-02
2500000	8.98E-02
3750000	8.97E-02
5000000	8.85E-02
6250000	9.33E-02
7500000	8.88E-02
8750000	8.93E-02
10000000	8.99E-02
11250000	9.27E-02
12500000	8.88E-02

Таблиця 6.2 — Время выполнения вычислений для восьми потоков для разного количества итераций для стратегии GUIDED

ВЫВОДЫ

При исследовании зависимости времени вычисления от количества потоков (3.1) было обнаружено, что с увеличением числа потоков время увеличивается. Это можно объяснить тем, что оно тратится на создание потоков.

Лучше всего зависимость времени вычисления от количества итераций прослеживается при использовании стратегии DYNAMIC (5.1). При увеличении количества итераций время вычислений уменишается.