Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО”

Факультет Программной инженерии и компьютерной техники

Направление подготовки (специальность) Системное и прикладное ПО

ОТЧЕТ

Лабораторная работа №2
по предмету «Параллельные вычисления»

Тема проекта: «Исследование эффективности параллельных библиотек для С-программ».

Обучающийся Ткаченко В.В. (**Фамилия И.О.**) Р4114 (**номер группы**)

Преподаватель Жданов А. Д. (**Фамилия И.О.**)

Санкт-Петербург
2023 г.

Содержание

Описание решаемой задачи 3

Краткая характеристика «железа» 4

Листинг программы lab1.c 4

Результаты экспериментов 7

Оценка накладных расходов 10

Оценка коэффициента распараллеливания 12

Вывод 12

Описание решаемой задачи

Вариант: (576)

Map: 6 | 5

Merge: 1

Sort: 6

В исходном коде программы, полученной в результате выполнения лабораторной работы №1, нужно на этапах Map и Merge все циклы с вызовами математических функций заменить их векторными аналогами из библиотеки «AMD Framewave» (http://framewave. sourceforge.net). При выборе конкретной Framewave-функции необходимо убедиться, что она помечена как MT (Multi-Threaded), т.е. распараллеленная. Полный перечень доступных функций находится по ссылке: http://framewave.sourceforge.net/ Manual/fw_section_060.html#fw_section_060. Например, Framewave-функция min в списке поддерживаемых технологий имеет только SSE2, но не MT.

Добавить в начало программы вызов Framewave-функции SetNumThreads(M) для установки количества создаваемых параллельной библиотекой потоков, задействуемых при выполнении распараллеленных Framewave-функций. Нужное число M следует устанавливать из параметра командной строки (argv) для удобства автоматизации экспериментов.

Скомпилировать программу, не применяя опции автоматического распараллеливания, использованные в лабораторной работе №1. Провести эксперименты с полученной программой для тех же значений N1 и N2, которые использовались в лабораторной работе №1, при M = 1, 2, . . . , K, где K – количество процессоров (ядер) на экспериментальном стенде.

Краткая характеристика «железа»

Имя ОС:	Майкрософт Windows 10 Pro
Версия:	10.0.19045 Сборка 19045
Изготовитель:	LENOVO
Модель:	20BE009ART
Тип:	Компьютер на базе x64
SKU системы:	LENOVO_MT_20BE
Процессор:	Intel(R) Core(TM) i7-4710MQ
Версия BIOS:	LENOVO GMET85WW (2.33), 30.05.2018
Версия SMBIOS:	2.7
Версия встроенного контроллера:	1.14
Режим BIOS:	Устаревший
gcc version:	11.3.0 (Ubuntu 11.3.0-1ubuntu1~22.04)

WSL2

Листинг программы lab2.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <sys/time.h>
#include "FW_1.3.1_Lin64/fwBase.h"
#include "FW_1.3.1_Lin64/fwSignal.h"

void insertion_sort(double arr[], int n)
{
    int i, j;
    double key;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i - 1;
        while (j >= 0 && arr[j] > key)
```

```
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}

int main(int argc, char *argv[])
{
    int i, j, N, M;
    double e = exp(1.0);
    struct timeval T1, T2;
    long delta_ms;
    double min = 1;
    double max = 576; // Based on personal data

    N = atoi(argv[1]);  // Размерность массива
    M = atoi(argv[2]);  // кол-во потоков
    fwSetNumThreads(M); // для установки количества создаваемых параллельной
библиотекой потоков, задействуемых при выполнении распараллеленных Framewave-функций

    Fw64f *restrict M1 = (Fw64f *)malloc(N * sizeof(Fw64f));
    Fw64f *restrict M2 = (Fw64f *)malloc(N / 2 * sizeof(Fw64f));
    Fw64f *restrict M2_copy = (Fw64f *)malloc(N / 2 * sizeof(Fw64f));

    unsigned int seed;
    unsigned int *restrict seed1 = &seed;
    unsigned int *restrict seed2 = &seed;

    gettimeofday(&T1, NULL);

    for (i = 0; i < 100; i++)
    {
        seed = i;
        // Generate // Map // Тут при формировании массива было деление на e и
вычисление кубического корня
        for (j = 0; j < N; j++)
        {
            M1[j] = (((double)rand_r(seed1) / (RAND_MAX)) * (max - min) + min) / e;
        }
        for (j = 0; j < N / 2; j++)
        {
            M2[j] = ((double)rand_r(seed2) / (RAND_MAX)) * (max * 10 - max) + max;
            // M2_copy[j] = M2[j];
        }
        fwsCopy_64f(M2, M2_copy, N / 2); // Создаем копию массива M2
        fwsCbrt_64f_A53(M1, M1, N);      // Кубический корень (деление на e при
формировании)

        /*
        for (j = 1; j < N / 2; j++)
        {
            M2[j] = log(fabs(tan(M2[j] + M2_copy[j - 1])));
        }
        M2[0] = log(fabs(tan(M2[0])));
        */
        for (j = 1; j < N / 2; j++)
        {
            M2[j] = M2[j] + M2_copy[j - 1];
        }
```

```
        fwsTan_64f_A53(M2, M2, N / 2);
        fwsAbs_64f_I(M2, N / 2);
        fwsLn_64f_I(M2, N / 2);

        // Merge
        /*
        for (j = 0; j < N / 2; j++)
        {
            M2[j] = pow(M1[j], M2[j]);
        }
        */
        fwsPow_64f_A53(M1, M2, M2, N / 2);

        // Sort
        insertion_sort(M2, N / 2);
        // Reduce
        double min_nonzero = INFINITY;
        double sum_sin = 0.0;
        for (j = 0; j < N / 2; j++)
        {
            if (M2[j] > 0 && M2[j] < min_nonzero)
            {
                min_nonzero = M2[j];
            }
            if ((int)(M2[j] / min_nonzero) % 2 == 0)
            {
                sum_sin += sin(M2[j]);
            }
        }
        printf("%lf\n", sum_sin);
    }
    // printf("%lf\n", sum_sin);
    gettimeofday(&T2, NULL);
    delta_ms = 1000 * (T2.tv_sec - T1.tv_sec) + (T2.tv_usec - T1.tv_usec) / 1000;
    // printf("\nN=%d. Milliseconds passed: %ld\n", N, delta_ms);
    printf("\n%ld\n", delta_ms);

    free(M1);
    free(M2);
    free(M2_copy);

    return 0;
}
```

Выдержка из README дистрибутива библиотеки AMD Framewave для установки библиотеки:

```
Linux(R) Operating System:
--------------------------

Do the following for both 32-bit and 64-bit installations (assuming the installation
directory is "ExampleDir").

1. To use the shared libraries, create the following symbolic links.

        For 64 bit installation
            cd ExampleDir/FW_1.3.1_Lin64/lib
```

```
        For 32 bit installation
            cd ExampleDir/FW_1.3.1_Lin32/lib

        Then create the following soft links using the following commands.

        ln -sf ./libfwBase.so.1.3.1 libfwBase.so.1
        ln -sf ./libfwImage.so.1.3.1 libfwImage.so.1
        ln -sf ./libfwJPEG.so.1.3.1 libfwJPEG.so.1
        ln -sf ./libfwSignal.so.1.3.1 libfwSignal.so.1
        ln -sf ./libfwVideo.so.1.3.1 libfwVideo.so.1

        ln -sf ./libfwBase.so.1 libfwBase.so
        ln -sf ./libfwImage.so.1 libfwImage.so
        ln -sf ./libfwJPEG.so.1 libfwJPEG.so
        ln -sf ./libfwSignal.so.1 libfwSignal.so
        ln -sf ./libfwVideo.so.1 libfwVideo.so

2. Compile a cpp file that uses FW (for example, test.cpp) as follows.

        To create 64 bit binaries:
            g++ -m64 -c -IExampleDir/FW_1.3.1_Lin64 test.cpp

        To create 32 bit binaries:
            g++ -m32 -c -IExampleDir/FW_1.3.1_Lin32 test.cpp

3. All FW libraries have dependency on fwBase.

        For example, link with the Image library as follows.

        To link to 64 bit binaries:
            g++ -m64 -LExampleDir/FW_1.3.1_Lin64/lib test.o -lfwImage -lfwBase

        To link to 32 bit binaries:
            g++ -m32 -LExampleDir/FW_1.3.1_Lin32/lib test.o -lfwImage -lfwBase

4. It may be necessary to explicitly link in the stdc++, pthreads,
   or math libraries if they are not automatically linked in.

5. Before running the application, make sure the ExampleDir/FW_1.3.1_Lin64/lib (for
64-bit installation) or
   ExampleDir/FW_1.3.1_Lin32/lib (for 32-bit installation) is in the environment's
shared library (LD_LIBRARY_PATH) search path.

    The following example shows the correct bash shell command syntax.

            $ export LD_LIBRARY_PATH=ExampleDir/FW_1.3.1_Lin64/lib:$LD_LIBRARY_PATH

        For 32 bit installation:
            $ export LD_LIBRARY_PATH=ExampleDir/FW_1.3.1_Lin32/lib:$LD_LIBRARY_PATH
```

Результаты экспериментов

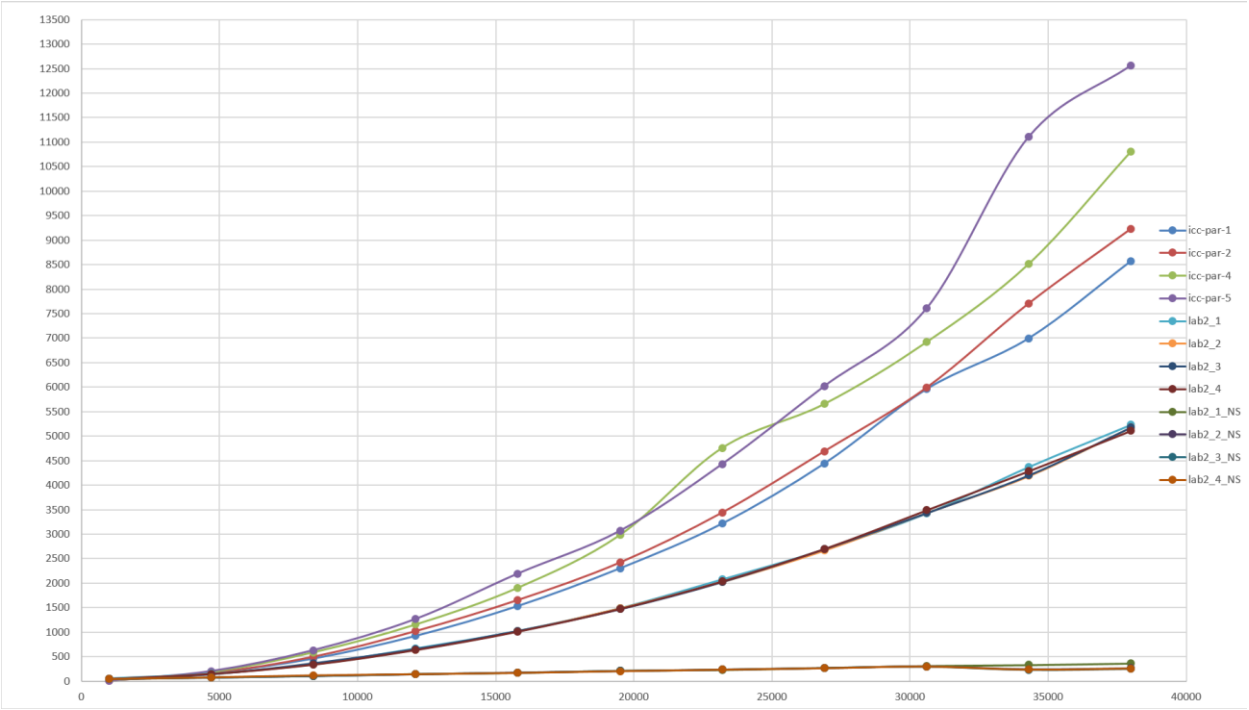
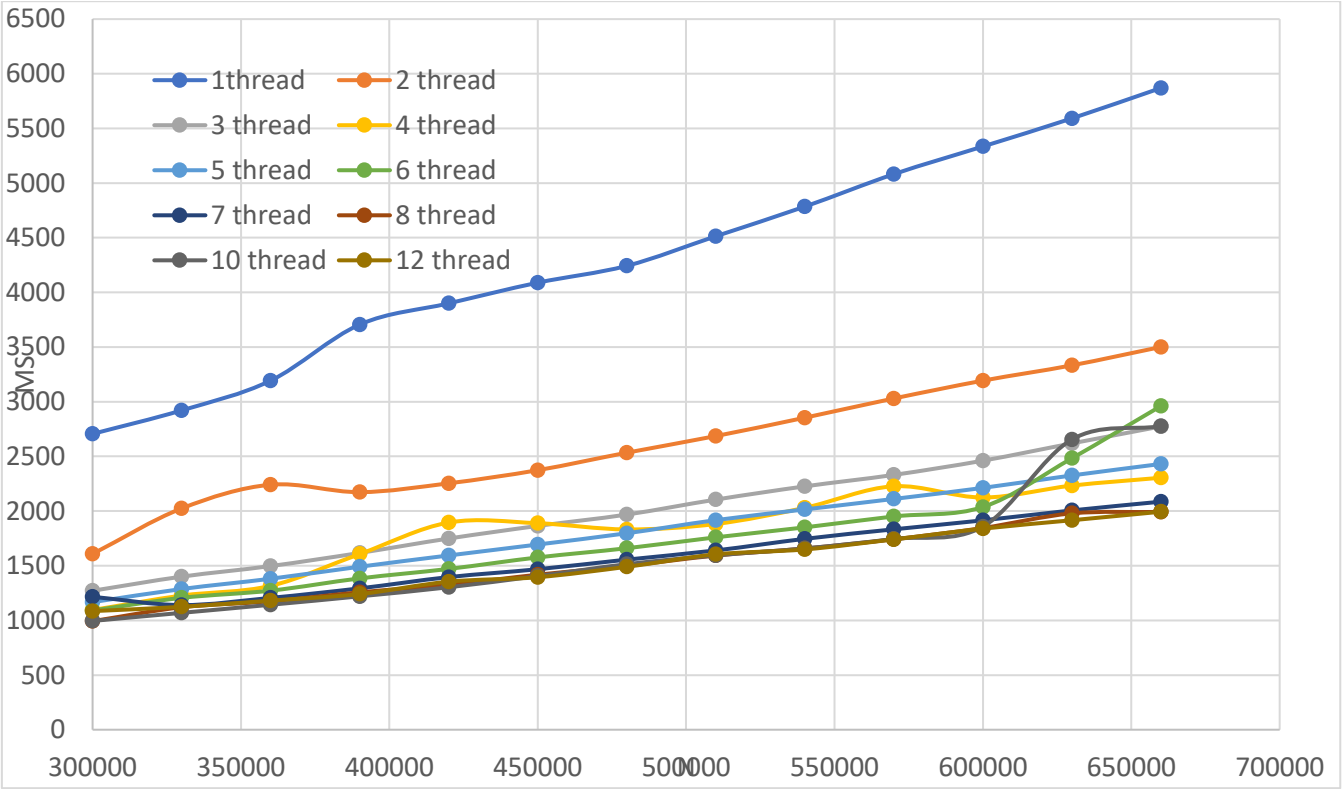


Рисунок 1 – график зависимости времени работы программы от передаваемого аргумента размерности массива.

	WITH SORT				NO SORT			
N	M=1	M=2	M=3	M=4	M=1	M=2	M=3	M=4
1000	55	49	51	49	45	46	48	46
4700	154	152	147	153	78	77	76	81
8400	348	347	372	342	110	107	110	121
12100	666	646	661	639	148	144	147	145
15800	1025	1022	1030	1015	173	178	176	172
19500	1489	1490	1474	1477	216	209	214	208
23200	2077	2027	2029	2038	232	237	240	238
26900	2690	2670	2699	2700	273	269	275	268
30600	3424	3435	3434	3489	315	302	302	302
34300	4374	4187	4204	4286	333	245	235	238
38000	5236	5180	5181	5109	366	264	256	262

Исходные значения N из первой лабораторно работы не могут предоставить нам правдивые результаты т.к. слишком малы, для дальнейших расчетов было принято решение ввести дополнительный диапазон значений для получения корректного результата эксперимента.

В данном случае уже можно наблюдать ускорение работы программы в зависимости от передаваемых значений параметра N



	MS									
N	1	2	3	4	5	6	7	8	10	12
300000	2706	1608	1271	1095	1167	1091	1216	993	995	1085
330000	2921	2024	1400	1227	1287	1206	1140	1120	1069	1124
360000	3193	2241	1497	1316	1381	1272	1206	1175	1143	1181
390000	3706	2173	1617	1607	1491	1384	1293	1257	1220	1241
420000	3900	2254	1749	1895	1594	1471	1396	1325	1303	1354
450000	4089	2374	1864	1889	1694	1576	1468	1418	1405	1395
480000	4243	2534	1968	1832	1797	1660	1556	1499	1514	1490
510000	4514	2686	2105	1882	1917	1760	1640	1593	1595	1604
540000	4785	2854	2226	2030	2015	1851	1746	1658	1658	1649
570000	5082	3030	2331	2227	2111	1951	1833	1742	1745	1741
600000	5335	3193	2461	2124	2212	2037	1915	1845	1858	1839
630000	5592	3333	2619	2233	2326	2483	2006	1979	2655	1916
660000	5870	3501	2774	2305	2431	2961	2086	1992	2776	1994

Дополнительный перечень значений показывает, что распараллеливание происходит и работа проделана корректно.

Оценка накладных расходов

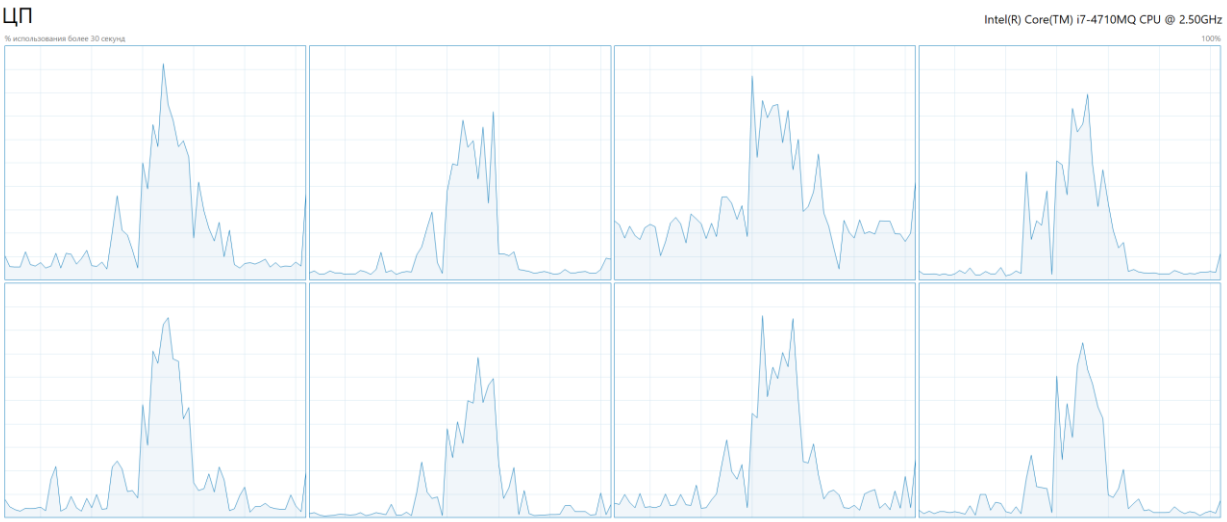


Рисунок 2 Загрузка процессора при M=2

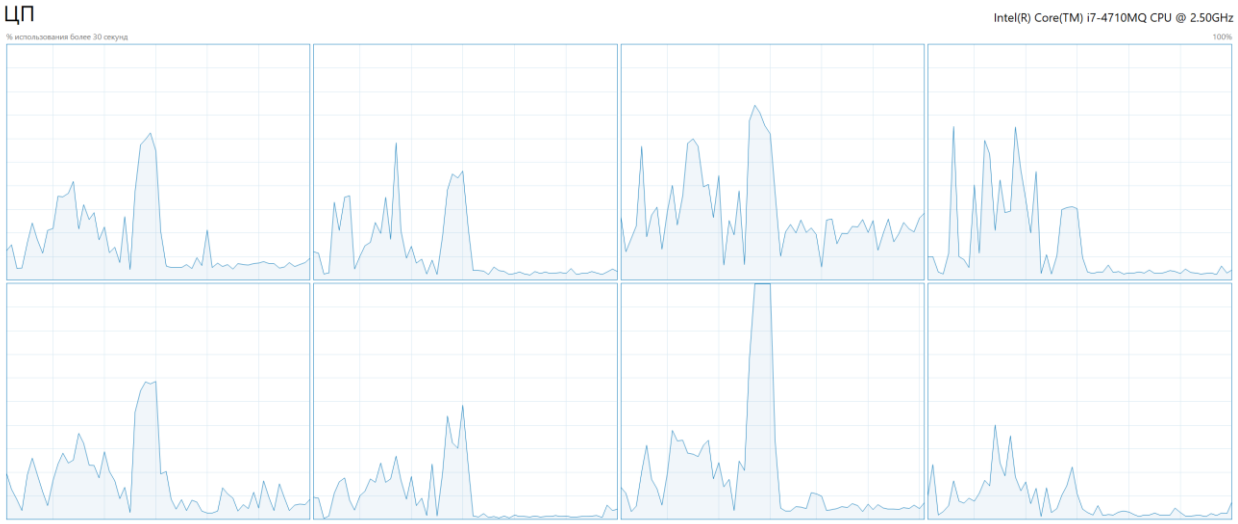


Рисунок 3 Загрузка процессора при M=4

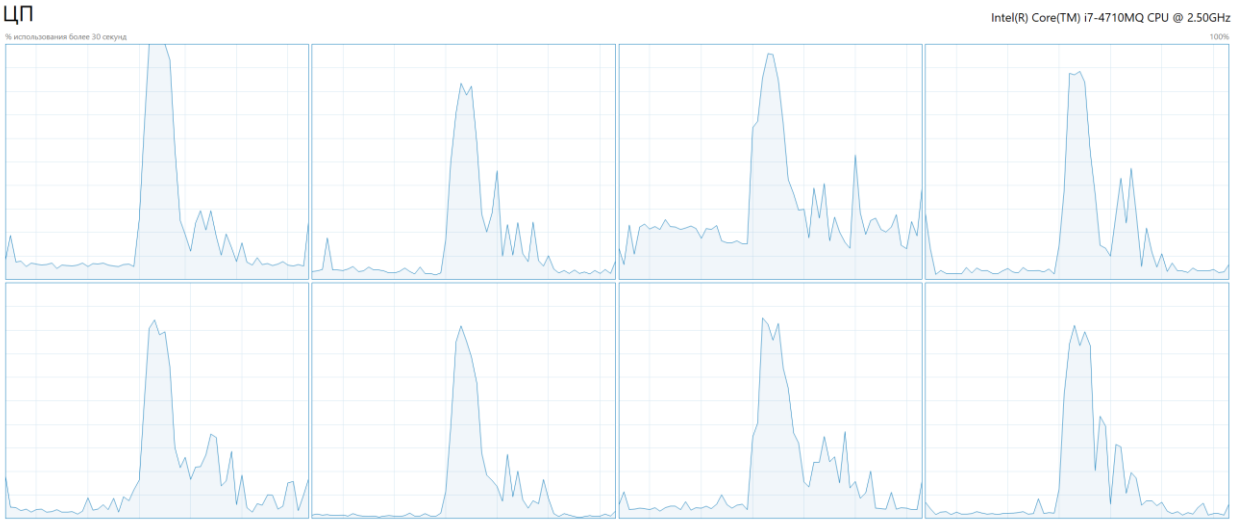


Рисунок 4 Загрузка процессора при M=7

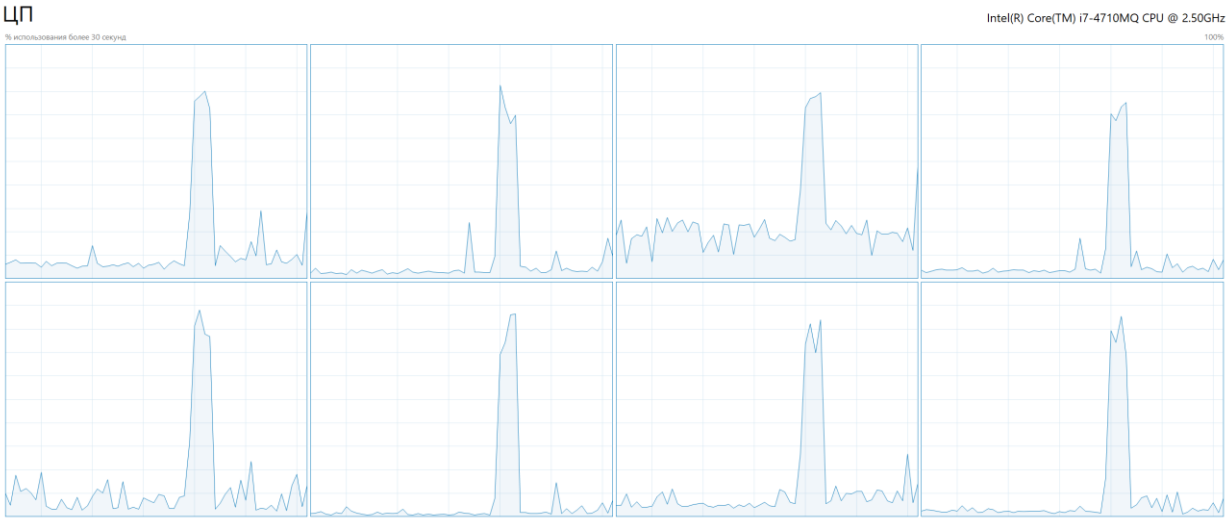


Рисунок 5 Загрузка процессора при M=12

При указании числа потоков, больших количества физических ядер время выполнения сокращается, но не с таким большим шагом. Если указывать число потоков больше 8, то ускорения нет вовсе.

Оценка коэффициента распараллеливания

Рассчитаем коэффициент распараллеливания, он будет равен:

$$t(p) = \frac{k \cdot t(1)}{p} + (1 - k) \cdot t(1).$$

$$k = \frac{\frac{t(p)}{t(1)} - 1}{\frac{1}{p} - 1}$$

где $t(p) -$ время выполнения на p потоках, $t(1)$ –время выполнения на 1 потоке, p –кол-во потоков.

$p = 2$	$p = 3$	$p = 4$	$p = 6$
0,81	0,79	0,81	0,59
$p = 7$	$p = 8$	$p = 10$	$p = 12$
0,75	0,76	0,59	0,72

Теперь подсчитаем получившуюся параллельную эффективность:

$$E_A(p) = (k + p - p * k)^{-1}$$

$E_A(2)$	$E_A(3)$	$E_A(4)$	$E_A(6)$
0.84	0.83	0.84	0.71
$E_A(7)$	$E_A(8)$	$E_A(10)$	$E_A(12)$
0.8	0.81	0,71	0.78

Вывод

В рамках данной лабораторной работы была рассмотрена параллельная библиотека AMD FrameWave. Использование библиотеки AMD Framewave

дает значительный прирост скорости относительно выполнения программы с использованием автоматического распараллеливания. При указании числа потоков, больше количества физических ядер время выполнения сокращается, но не с таким большим шагом как при указывании числа потоков до 4. Если указывать число потоков больше 8, то ускорения нет вовсе.