Федеральное государственное автономное образовательное учреждение высшего образования

«Университет ИТМО»

Факультет ПИиКТ

Дисциплина: Параллельные вычисления

Лабораторная работа 2

AMD Framewave

Выполнил: Гурин Евгений Иванович

Преподаватель: Жданов Андрей Дмитриевич

Группа: P4116

Санкт-Петербург 2023г.

Задача

1. 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 ана- логами 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 на- ходится 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`.

Примечание: выбор библиотеки Framewave не является обязатель- ным, можно использовать любую другую параллельную библиоте- ку, если в ней нужные функции распараллелены, так, например, можно использовать ATLAS (для этой библиотеки необходимо вы- ключить троттлинг и энергосбережение, а также разобраться с ме- ханизмом изменения числа потоков) или Intel Integrated Performance Primitives.

2. Добавить в начало программы вызов Framewave-функции `SetNumThreads(M)` для установки количества создаваемых парал- лельной библиотекой потоков, задействуемых при выполнении рас- параллеленных Framewave-функций. Нужное число M следует уста- навливать из параметра командной строки `(argv)` для удобства ав- томатизации экспериментов.

Примечание: При использовании Intel IPP функцию SetNumThreads(M) не нужно использовать. Необходимо компилировать программу под разное количество потоков.

3. 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 No1, при M = 1, 2, . . . , K, где K – количество процессоров (ядер) на экспериментальном стенде.

4. Сравнить полученные результаты с результатами лабораторной ра- боты No1: на графиках показать, как изменилось время выполне- ния программы, параллельное ускорение и параллельная эффективность.

5. Написать отчёт о проделанной работе.

6. Подготовиться к устным вопросам на защите.

7. **Необязательное задание №1** (для получения оценки «четыре» и «пять»). Исследовать параллельное ускорение для различных зна- чений M > K, т.е. оценить накладные расходы при создании чрез- мерного большого количества потоков. Для иллюстрации того, что программа действительно распараллелилась, привести график за- грузки процессора (ядер) во время выполнения программы при N = N2 для всех использованных M. Для получения графика можно как написать скрипт, так и просто сделать скриншот дис- петчера задач, указав на скриншоте моменты начала и окончания эксперимента (в отчёте нужно привести текст скрипта или назва- ние использованного диспетчера).

8. **Необязательное задание №2** (для получения оценки «пять»). Это задание выполняется только после выполнения предыдущего пунк- та. Используя закон Амдала, рассчитать коэффициент распаралле- ливания для всех экспериментов и привести его на графиках. Про- комментировать полученные результаты.

Конфигурация

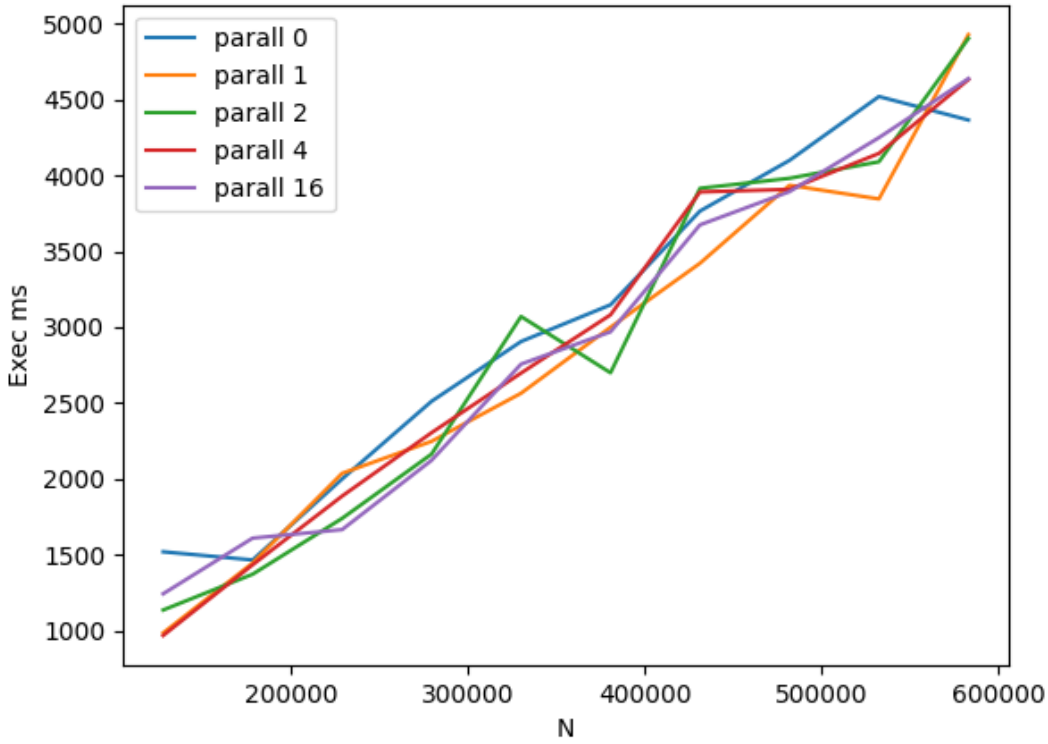
```
Host Name:                 EGURIN-PC
OS Name:                   Microsoft Windows 11 Pro
OS Version:                10.0.22000 N/A Build 22000
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:          user
Registered Organization:   N/A
Product ID:                00331-10000-00001-AA539
Original Install Date:     02.10.2022, 21:59:41
System Boot Time:          20.03.2023, 2:46:00
System Manufacturer:       ASUS
System Model:              System Product Name
System Type:               x64-based PC
Processor(s):              1 Processor(s) Installed.
                           [01]: AMD64 Family 23 Model 113 Stepping 0
AuthenticAMD ~3600 Mhz
BIOS Version:              American Megatrends Inc. 2803, 27.04.2022
Windows Directory:         C:\Windows
System Directory:          C:\Windows\system32
Boot Device:               \Device\HarddiskVolume2
System Locale:             en-us;English (United States)
Input Locale:              en-us;English (United States)
Time Zone:                 (UTC+03:00) Moscow, St. Petersburg
Total Physical Memory:     32 679 MB
Available Physical Memory: 20 506 MB
Virtual Memory: Max Size:  87 975 MB
Virtual Memory: Available: 19 470 MB
Virtual Memory: In Use:    68 505 MB
Page File Location(s):     D:\pagefile.sys
Domain:                    WORKGROUP
Logon Server:              \\EGURIN-PC
Hotfix(s):                 5 Hotfix(s) Installed.
                           [01]: KB5022505
```

```
                           [02]: KB5012170
                           [03]: KB5023698
                           [04]: KB5022369
                           [05]: KB5022925
Network Card(s):           4 NIC(s) Installed.
                           [01]: Realtek PCIe 2.5GbE Family Controller
                                 Connection Name: Ethernet
                                 Status:          Media disconnected
                           [02]: Intel(R) Wi-Fi 6 AX200 160MHz
                                 Connection Name: Wi-Fi
                                 DHCP Enabled:    Yes
                                 DHCP Server:     192.168.1.1
                                 IP address(es)
                                 [01]: 192.168.1.47
                                 [02]: fe80::933b:210e:a9a7:2c6e
                           [03]: Bluetooth Device (Personal Area Network)
                                 Connection Name: Bluetooth Network Connection
                                 Status:          Media disconnected
                           [04]: VirtualBox Host-Only Ethernet Adapter
                                 Connection Name: Ethernet 2
                                 DHCP Enabled:    No
                                 IP address(es)
                                 [01]: 192.168.56.1
                                 [02]: fe80::527e:5766:393d:acc6
Hyper-V Requirements:      A hypervisor has been detected. Features required for
Hyper-V will not be displayed.
```

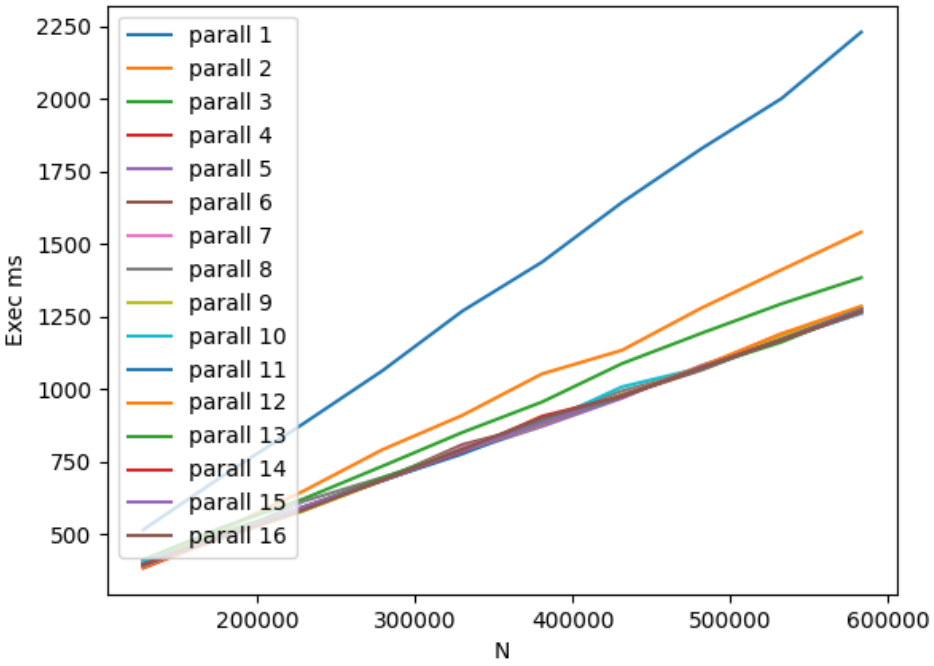
Результаты работы

Для экспериментов был выбран компилятор clang

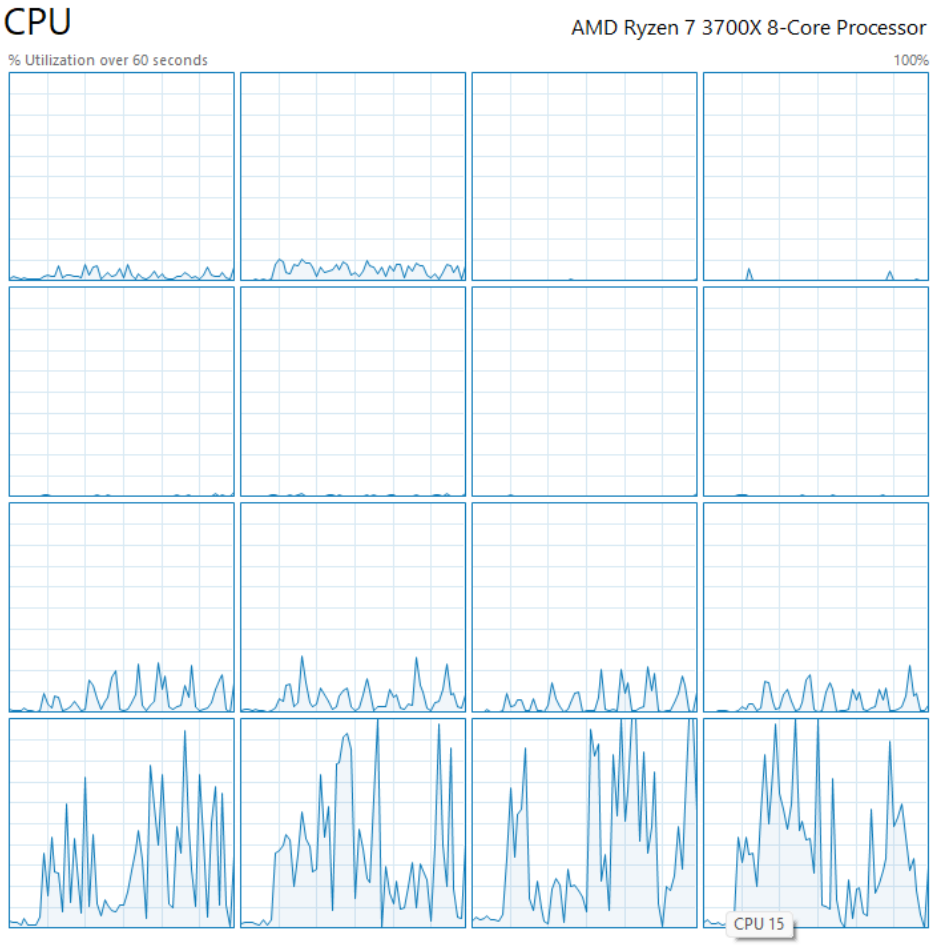
CLANG (автоматизированное распараллеливание)

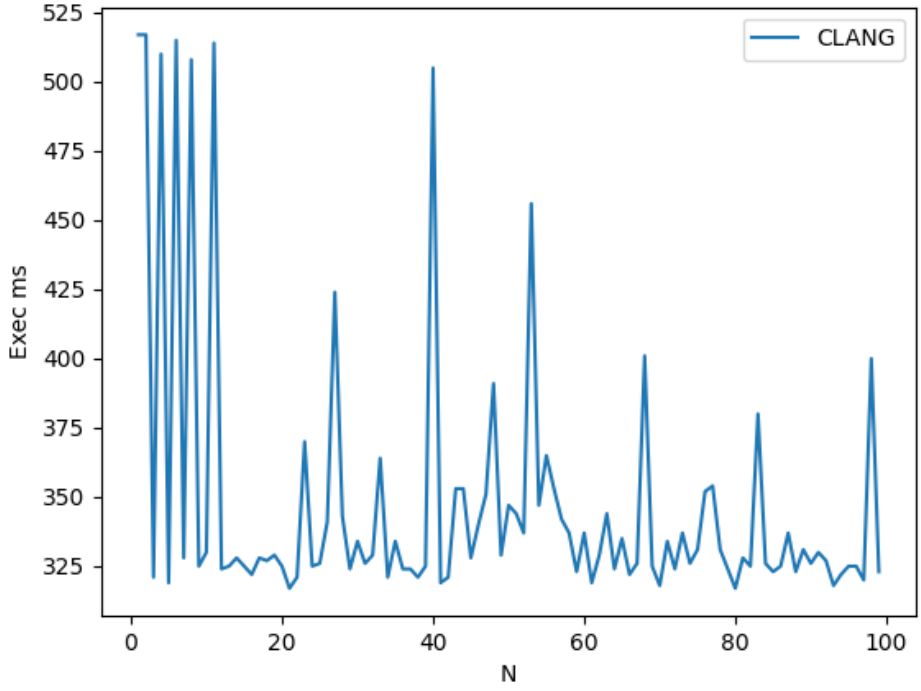


CLANG (Framewave)



Загрузка процессора и накладные расходы





Листинг main.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 swap(Fw32f *a, Fw32f *b) {
    Fw32f * restrict t = NULL;
    *t = *a, *a = *b, *b = *t;
}


void sort_stupid(Fw32f *array, int n) {
    int i = 0;
    while (i < n - 1) {
        if (array[i + 1] < array[i]) swap(array + i, array + i + 1), i = 0;
        else i++;
    }
}

void print_arr(Fw32f *array, int n) {
    for (int i = 0; i < n; ++i)
    {
        printf("%f ", array[i]);
    }
    printf("\n");
}

int main(int argc, char *argv[]) {
    struct timeval T1, T2;
    long delta_ms;
    gettimeofday(&T1, NULL); /* save current time for benchmarking T1 */

    const int N = atoi(argv[1]); /* N - array size, equals first cmd param */
    const int M = atoi(argv[2]); /* M - amount of threads */
    fwSetNumThreads(M);

    const int N_2 = N / 2;
    const int A = 280;

    Fw32f * restrict m1 = fwsMalloc_32f(N);
    Fw32f * restrict m2 = fwsMalloc_32f(N_2);
    Fw32f * restrict m2_cpy = fwsMalloc_32f(N_2);
    Fw64f * restrict m2_cpy_2 = fwsMalloc_64f(N_2);


    for (unsigned int i = 0; i < 100; i++) /* 100 experiments */
    {

        unsigned int seedp = i;
```

```
    // generate 1
    for (int j = 0; j < N; ++j) {
        m1[j] = (rand_r(&seedp) % (A * 100)) / 100.0 + 1;
    }

    // generate 2
    for (int j = 0; j < N_2; ++j) {
        m2[j] = A + rand_r(&seedp) % (A * 9);
    }

    fwsCopy_32f(m2, m2_cpy, N_2);
    // m1[j] = 1 / tanh(sqrt(m1[j]));
    fwsSqrt_32f(m1, m1, N);
    fwsTanh_32f_A24(m1, m1, N);
    fwsDivCRev_32f(m1, 1, m1, N);

    // m2[j] = m2[j] + m2_cpy[j - 1]
    fwsAdd_32f(m2 + 1, m2_cpy, m2 + 1, N_2 - 1);

    // m2[j] = pow(log10(m2[j]), M_E)
    fwsLog10_32f_A24(m2, m2, N_2);
    fwsPowx_32f_A24(m2, M_E, m2, N_2);

    for (int j = 0; j < N_2; ++j) {
        m2[j] = fmax(m2[j], m1[j]);
    }

    // sort_stupid(m2, N_2);

    // reduce
    Fw32f m2_min;
    fwsMin_32f(m2, N_2, &m2_min);

    fwsZero_32f(m2_cpy, N_2);
    fwsZero_64f(m2_cpy_2, N_2);
    fwsDivC_32f(m2, m2_min, m2_cpy, N_2);

    Fw64f X = 0;
    for (int j = 0; j < N_2; ++j) {
        // find where m2/m2_min % 2 == 0 and copy to m2_cpy_2
        if((int) m2_cpy[j] % 2 == 0) m2_cpy_2[j] = m2[j];
    }
    fwsSin_64f_A50(m2_cpy_2, m2_cpy_2, N_2);
    fwsSum_64f(m2_cpy_2, N_2, &X);

    printf("%f ", X);
}

fwsFree(m1);
fwsFree(m2);
fwsFree(m2_cpy);
fwsFree(m2_cpy_2);

gettimeofday(&T2, NULL);
/* запомнить текущее время T2 */
```

```
    delta_ms = 1000 * (T2.tv_sec - T1.tv_sec) + (T2.tv_usec - T1.tv_usec) /
1000;
    printf("\n%ld\n", delta_ms); /* T2 - T1 */
    return 0;
}
```

Программа для поиска времени на накладные расходы

Для поиска накладных расходов я воспользовался способом, описанным в лекции и замерял время исполнения программы, которая вызывает fwSetNumThreads

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


void print_delta(struct timeval T1, struct timeval T2) {
    long delta_us = 1000000 * (T2.tv_sec - T1.tv_sec) + (T2.tv_usec -
T1.tv_usec);
    printf("%ld\n", delta_us);
}

int main(int argc, char *argv[]) {
    struct timeval T1, T2;
    gettimeofday(&T1, NULL);
    const int N = atoi(argv[1]); /* N - max threads amount */
    int s = 0;
    fwSetNumThreads(N);
    s++;
    gettimeofday(&T2, NULL);
    print_delta(T1, T2);
    return 0;
}
```

Вывод

По сравнению с использованием автоматического распараллеливания библиотека Framewave даёт прирост в 2 раза даже с использованием 1 потока. В экспериментах при увеличении количества потоков наблюдается весомый прирост производительности при увеличении количества потоков до 4-х. После этого значения прироста не наблюдается. Кроме того по загрузке процессора видно, что количество использованных физических ядер не превышает 4-х. По сравнению с автоматизированным распараллеливанием результат в разы лучше.

При попытке подсчёта накладных расходов хороших статистических данных получено не было. Все накладные расходы укладывались в 500 наносекунд и из-за того, что более 4 потоков не было использования, то дальнейший подсчёт не представляется возможным и увеличение времени на накладные расходы не наблюдается.