Федеральное государственное автономное образовательное учреждение высшего образования

«**Университет ИТМО»**

**Факультет ПИиКТ**

**Дисциплина: Параллельные вычисления**

**Лабораторная работа 3**

OpenMP

Выполнил: Гурин Евгений Иванович

Преподаватель: Жданов Андрей Дмитриевич

Группа: Р4116

Санкт-Петербург 2023г.

## Задача

## 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

**Прямая совместимость** была достигнута с помощью проверки

#if defined(\_OPENMP)

omp\_set\_dynamic(0);

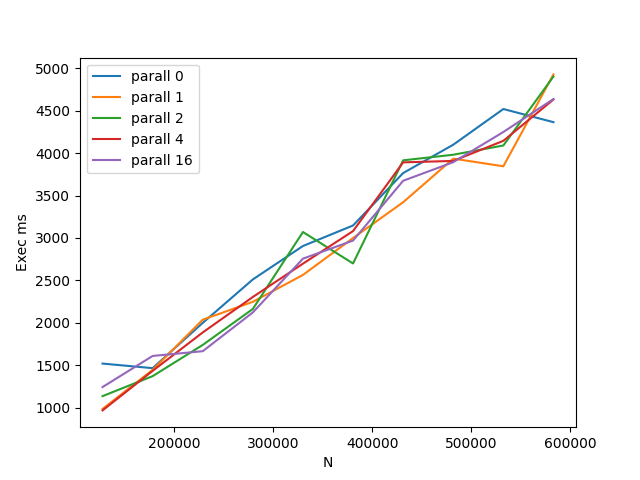
const int M = atoi(argv[2]); /\* M - amount of threads \*/

omp\_set\_num\_threads(M);

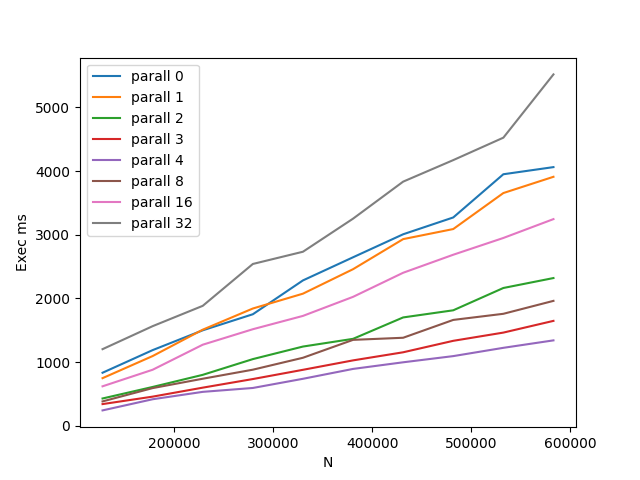
#endif

**Результаты эксперементов**

**CLANG (автоматизированное распараллеливание)**

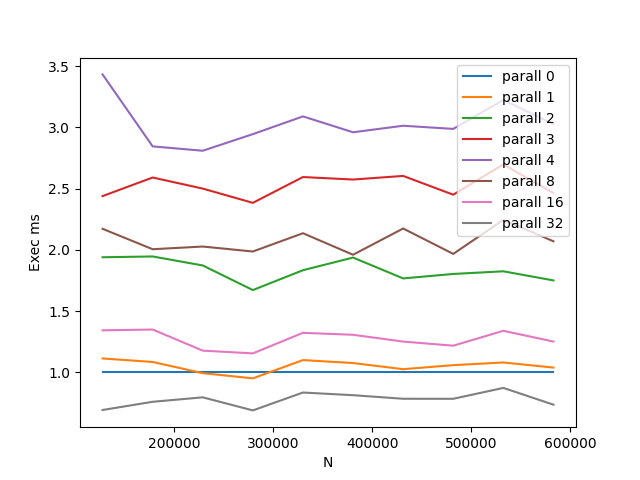


**CLANG (OMP без параметра schedule)**

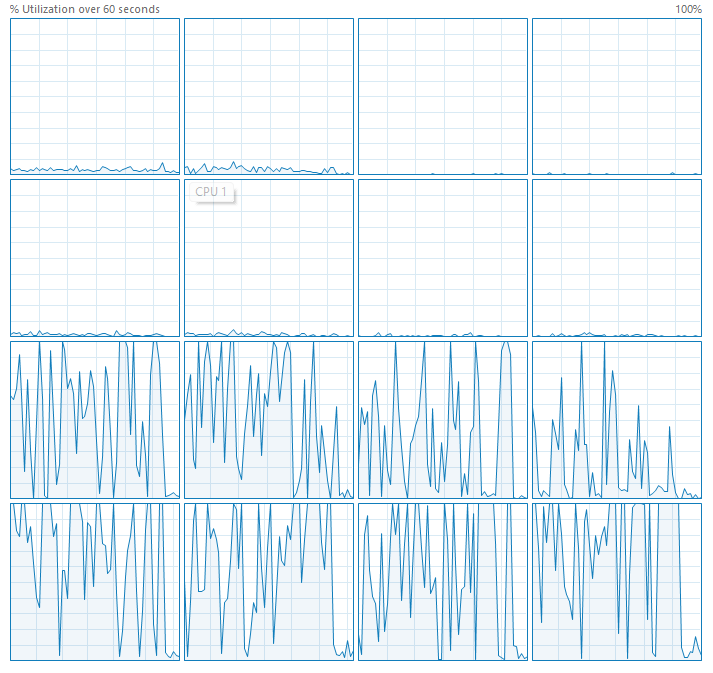


Наблюдается улучшение результатов при увеличении количества потоков до 4, далее ухудшение (хотя процессор 8-ми ядерный)

**Параллельное ускорение**

****

**Загрузка процессора (стандартный диспетчер задач Windows)**

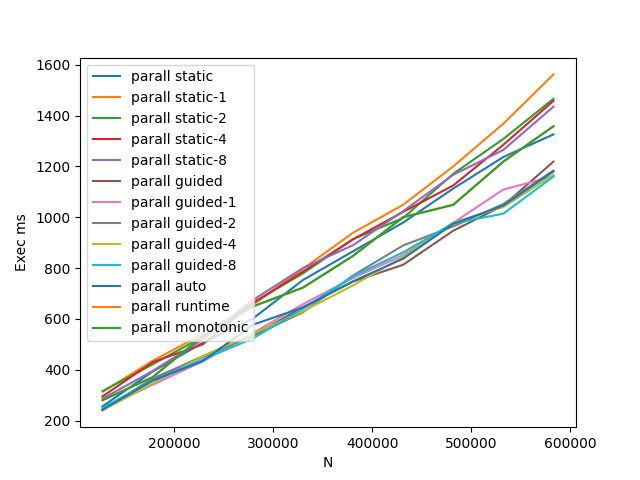


## **Сравнение различных вариантов schedule**

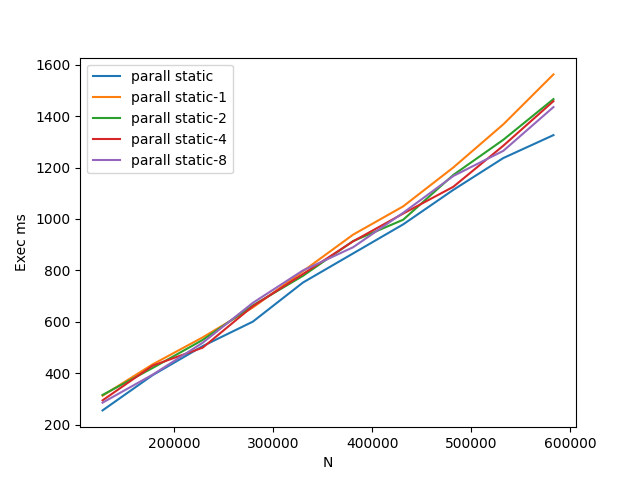
Исследование различных вариаций расписания производились с наилучшим результатом количества потоков (4) на аналогичных значениях параметра N (размерности массива)

**Сравнение static, guided, auto, runtime, monotonic на одном графике**

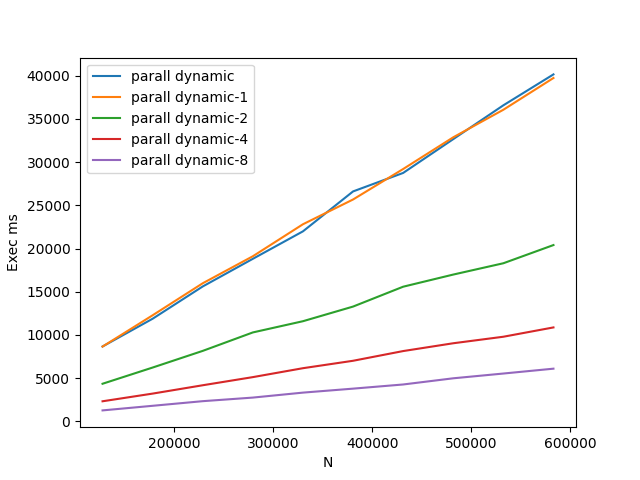
На данном графике не приводится dynamic, так как с ним время работы в разы дольше и различия остальных будут не видны на графике. На графике видно, что наилучшие результаты показывает расписание guided



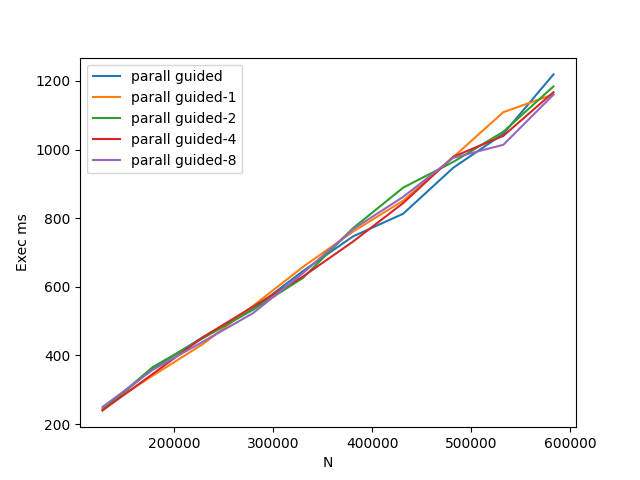
**Сравнение static с различным chunk size**



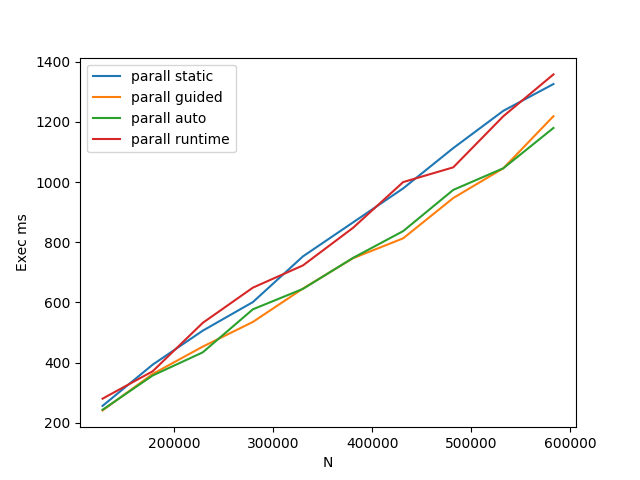
**Сравнение dynamic** **с различным chunk size**



**Сравнение guided** **с различным chunk size**



**Сравнение различных видов расписания с параметром chunk size по умолчанию**



**Расписание по умолчанию**

С помощью метода omp\_get\_schedule было выяснено, что расписание по умолчанию – dynamic с параметром chunk size 1. Хотя при ручном указании соответствующих параметров результат кардинально отличается и результаты с параметрами по умолчанию больше похожи на результат guided.

**Наилучший результат**

M (количество потоков) – 4

Расписание (schedule) – guided (с параметром chunk size 4)

При запуске все вычислители (процессоры) равноценны и размеры массивов не меняются. Для других значений размерности массивов могут быть актуальны другие значения параметров

**Сложность**

Без распараллеливания C1 \* N

С распараллеливанием C1 \* N / M + C2 \* N, где M – количество потоков

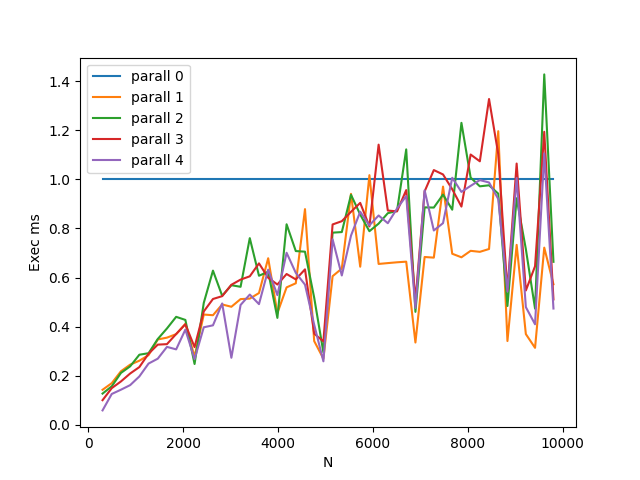
Все циклы проходят по элементам массивов без вложений и часть программы не параллельна (заполнение)

## Границы выигрыша от распараллеливания

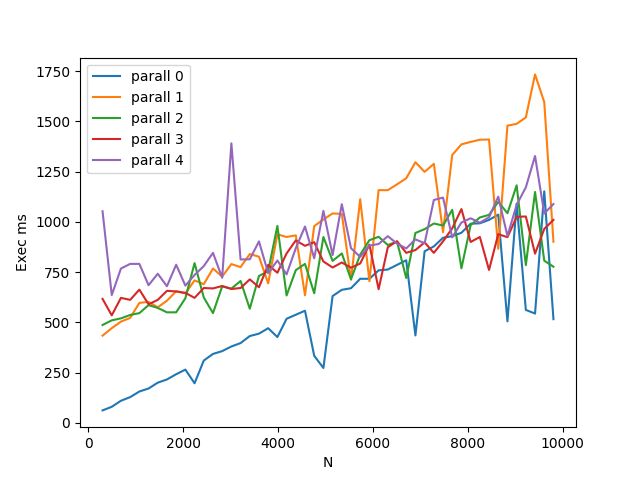
Для поиска соответствующих значений размера массива, когда накладные расходы будут превышать выигрыш от распараллеливания я запускал эксперименты на значениях N < N1 и сужал область поиска.

**Для одного эксперимента**

**График параллельного ускорения для N < 10000**



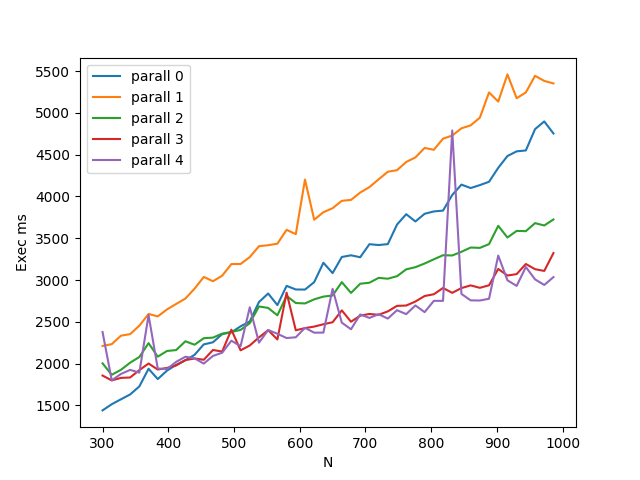
**График времени выполнения программы для N < 10000**



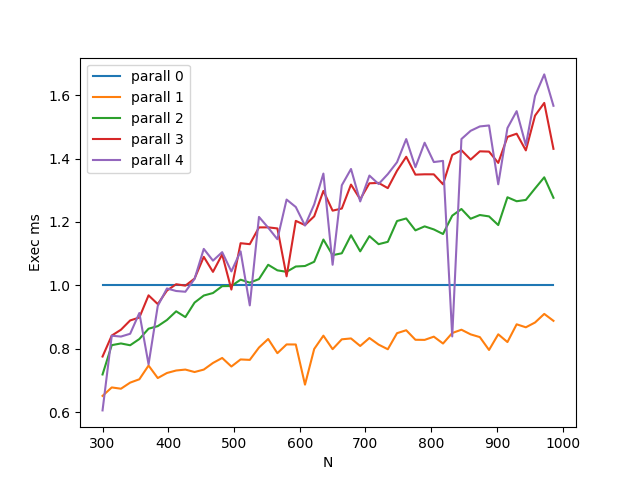
По графикам видно, что для значений размера массива менее 6000 программа без распараллеливания работает быстрее и параллельное ускорение наблюдается только на значениях больше. После проведенных экспериментов было получено, что время на накладные расходы составляет около 500 наносекунд для 4 потоков, а до N = 6000 время выполнения программы меньше времени на накладные расходы.

**Для 100 экспериментов**

**График параллельного ускорения для N < 1000**

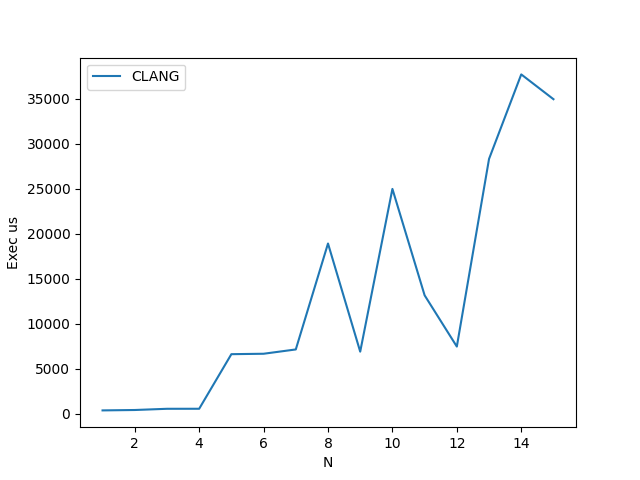
****

**График параллельного ускорения для N < 1000**

****

По графикам видно, что для значений размера массива менее 500 программа без распараллеливания работает быстрее и параллельное ускорение наблюдается только на значениях больше. При использования 1 потока и openMP до N 1000 ускорения в принципе не набюдается из-за накладных расходов по сравнению с программой без распараллеливания.

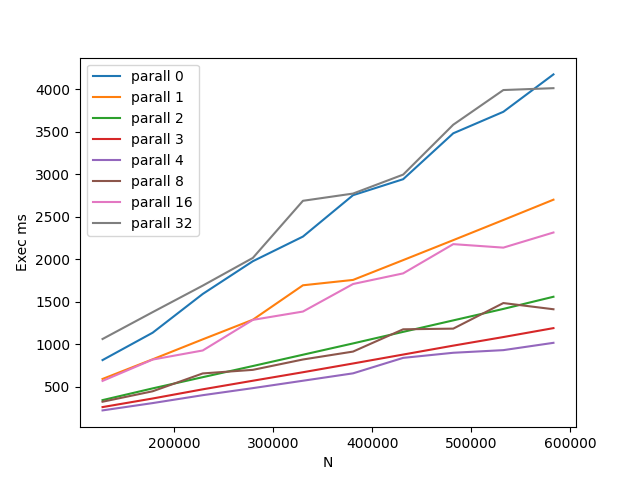
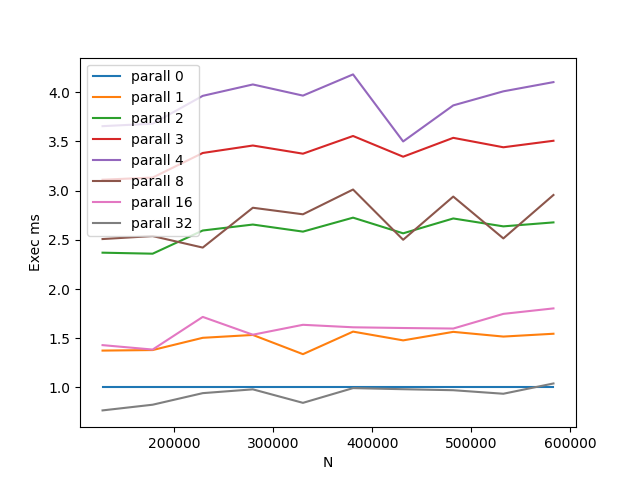
**График времени на накладные расходов**



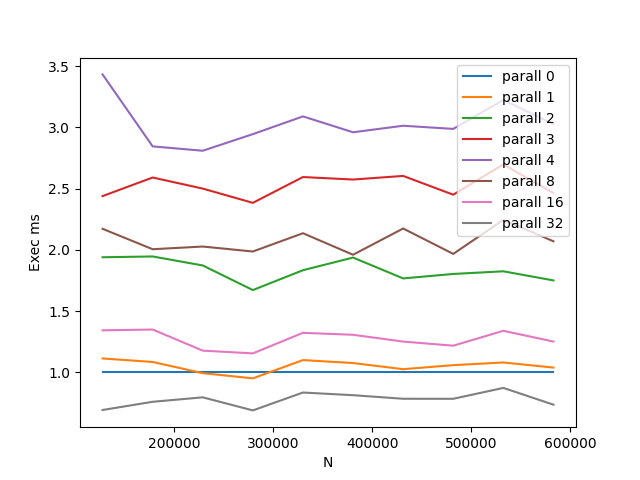
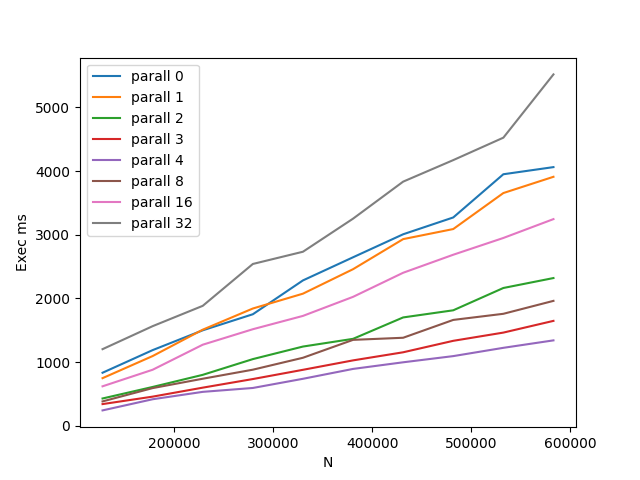
На графике видно, что для количества потоков более 4 время, затрачиваемое на накладные расходы сильно увеличивается (с около 500 наносекунд до 1 мс и больше)

## Эксперименты с параметрами оптимизации

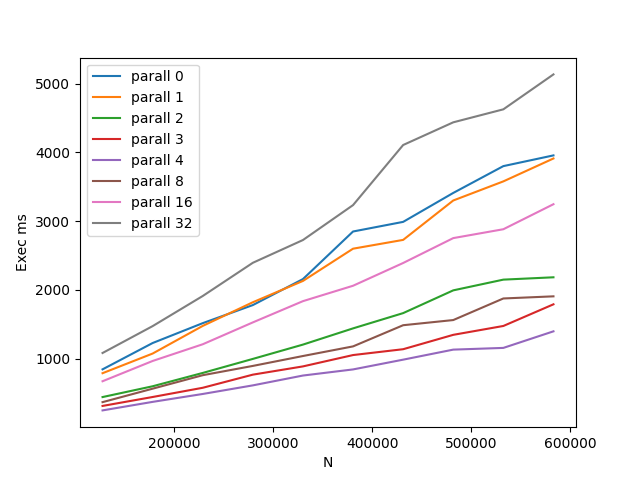
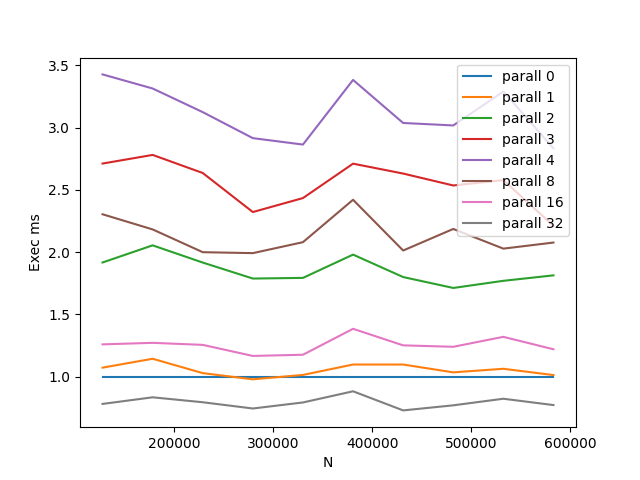
**-Ofast -flto**



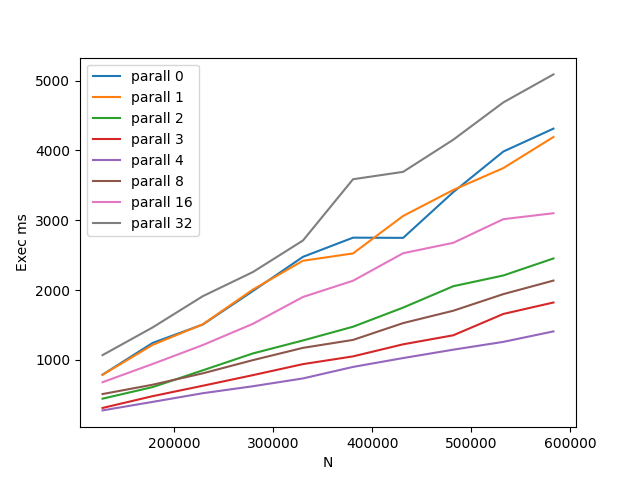
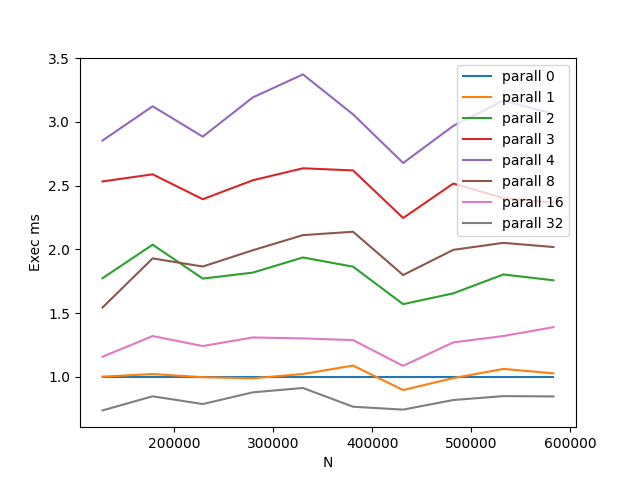
**-O3 (те же результаты что в начале отчета)**

****

**-O2**



**-O1**



Для параметром -O1, -O2, -O3 значения очень похожи, разница во времени выполнения и параллельном ускорении почти не наблюдается. Однако для параметра -Ofast наблюдается прирост в параллельном ускорении и заметно изменение в приросте в для количества параллельных потоков больше 4. Однако при использовании данного флага наблюдалось, что верификация значений не проходила.

# Листинг main.c

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <sys/time.h>

#include <omp.h>

void swap(double \*a, double \*b) {

double t;

t = \*a, \*a = \*b, \*b = t;

}

void sort\_stupid(double \*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(double \*array, int n) {

for (int i = 0; i < n; ++i)

{

printf("%f ", array[i]);

}

printf("\n");

}

void print\_delta(struct timeval T1, struct timeval T2) {

long delta\_ms = 1000 \* (T2.tv\_sec - T1.tv\_sec) + (T2.tv\_usec - T1.tv\_usec) / 1000;

printf("\n%ld\n", delta\_ms);

}

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

struct timeval T1, T2;

gettimeofday(&T1, NULL); /\* запомнить текущее время T1 \*/

const int N = atoi(argv[1]); /\* N - array size, equals first cmd param \*/

const int N\_2 = N / 2;

const int A = 280;

double \* restrict m1 = malloc(N \* sizeof(double));

double \* restrict m2 = malloc(N\_2 \* sizeof(double));

double \* restrict m2\_cpy = malloc(N\_2 \* sizeof(double));

#if defined(\_OPENMP)

omp\_set\_dynamic(0);

const int M = atoi(argv[2]); /\* M - amount of threads \*/

omp\_set\_num\_threads(M);

#endif

for (unsigned int i = 0; i < 1; i++) /\* 100 экспериментов \*/

{

double X = 0;

unsigned int seedp = i;

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);

}

#pragma omp parallel default(none) shared(N, N\_2, A, m1, m2, m2\_cpy, i, X)

{

#pragma omp for

for (int j = 0; j < N\_2; ++j) {

m2\_cpy[j] = m2[j];

}

// map

#pragma omp for

for (int j = 0; j < N; ++j) {

m1[j] = 1 / tanh(sqrt(m1[j]));

}

#pragma omp for

for (int j = 1; j < N\_2; ++j) {

m2[j] = m2[j] + m2\_cpy[j - 1];

}

#pragma omp for

for (int j = 1; j < N\_2; ++j) {

m2[j] = pow(log10(m2[j]), M\_E);

}

#pragma omp for

for (int j = 0; j < N\_2; ++j) {

m2[j] = m2[j] > m1[j] ? m2[j] : m1[j] ;

}

// sort\_stupid(m2, N\_2);

// reduce

int k = 0;

while (m2[k] == 0 && k < N\_2 - 1) k++;

double m2\_min = m2[k];

#pragma omp for

for (int j = 0; j < N\_2; ++j) {

m2\_cpy[j] = 0;

if((int)(m2[j] / m2\_min) % 2 == 0) m2\_cpy[j] = sin(m2[j]);

}

#pragma omp for reduction(+ : X)

for (int j = 0; j < N\_2; ++j) {

X += m2\_cpy[j];

}

}

printf("%f ", X);

}

gettimeofday(&T2, NULL);

print\_delta(T1, T2);

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 <omp.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 M = atoi(argv[1]); /\* M - amount of threads \*/

omp\_set\_dynamic(0);

omp\_set\_num\_threads(M);

int s = 0;

#pragma omp parallel

{

s++;

}

gettimeofday(&T2, NULL);

print\_delta(T1, T2);

return 0;

}

# Вывод

По сравнению с использованием автоматического распараллеливания или использования программы без распараллеливания максимальное наблюдаемое параллельное ускорение составляет около 3.5. При сравнении различных параметров расписание наибольший прирост наблюдается со значением guided. Кроме того получены интересные результаты с параметром dynamic, при котором время выполнения программы существенно увеличивалось.

Были подсчитаны значения времени на накладные расходы. Для количества потоков до 4 все накладные расходы укладывались в 500 наносекунд. При дальнейшем увеличении наблюдается существенное увеличение времени, затрачиваемого на накладные расходы.

Для значений N < N1 было найдено значение, когда использование распараллеливания увеличивает время выполнения. Для 100 экспериментов такое значение было около 500 элементов.

Было проведено 3 дополнительных эксперимента для различных параметров оптимизации. Было обнаружено, что использование параметра -Ofast увеличивает возможный прирост, однако значения перестают проходить валидацию.