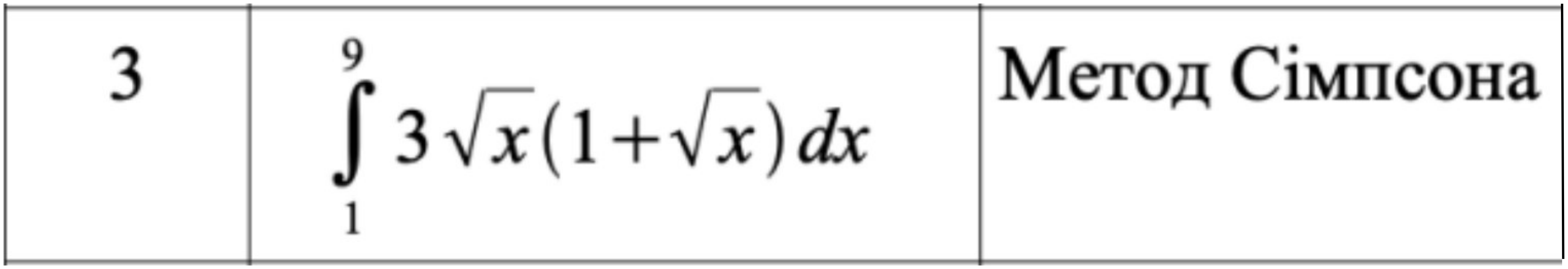
Завдання:

Розпаралелити процес обчислення визначеного інтегралу, використовуючи редукцію.
Обчислити значення визначеного інтеграла відповідно до варіанту.

Варіант 3



3	$\int_1^9 3\sqrt{x}(1+\sqrt{x})\,dx$	Метод Сімпсона

- *Метод Сімпсона* $\int_a^b f(x)\,dx \approx \frac{h}{3}\left(\frac{1}{2}f(x_0)+\sum_{i=1}^{n-1} f(x_i)+2\sum_{i=1}^{n} f\left(\frac{x_{i-1}+x_i}{2}\right)+\frac{1}{2}f(x_n)\right)$

$$h=\frac{b-a}{n},\ x_i=a+i\cdot h$$

Реалізація

Посилання на гітхаб репозиторій: https://github.com/Bohdan628318ylypchenko/parallel-programming-lab3-.git

main.c

```c
#include "integral.h"

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

#define A_TEST     1.0
#define B_TEST     9.0
#define S_EXPECTED 172.0
#define TEST_COUNT 5

#define USAGE_MSG "Usage: [s]ingle-threaded segment_count:int | [m]ulti-threaded segment_count:int thread_count:int\n"

static double f_test(double x);
static void test(double (*simpson)(double(*)(double), double, double, int));

static int n = 0;

int main(int argc, char ** argv)
{
        if (argc < 3)
        {
                puts(USAGE_MSG);
                return 0;
        }

        switch (argv[1][0])
        {
                case 's':

                        if (argc != 3)
                        {
                                puts(USAGE_MSG);
                                return 0;
                        }

                        n = atoi(argv[2]);
                        if (n <= 1)
                        {
                                printf("Invalid segment count: %s\n", argv[2]);
                                return 0;
                        }

                        test(simpson_1t);

                        break;

                case 'm':

                        if (argc != 4)
                        {
                                puts(USAGE_MSG);
                                return 0;
                        }

                        n = atoi(argv[2]);
                        if (n <= 1)
                        {
                                printf("Invalid segment count: %s\n", argv[2]);
                                return 0;
                        }

                        int num_threads = atoi(argv[3]);
                        if (num_threads <= 0)
                        {
                                printf("Invalid thread count: %s\n", argv[3]);
                                return 0;
                        }

                        omp_set_num_threads(num_threads);

                        test(simpson_mt);

                        break;

                default:
                        puts(USAGE_MSG);
                        return 0;
        }

        return 0;
}

static void test(double (*simpson)(double(*)(double), double, double, int))
{
        double s_time, e_time, time, avg_time = 0;
        double integral;
        for (int i = 0; i < TEST_COUNT; i++)
        {
                // Calculating integral
                s_time = omp_get_wtime();
                integral = simpson(f_test, A_TEST, B_TEST, n);
                e_time = omp_get_wtime();

                // Asserting
                time = e_time - s_time;
                printf("s_actual = %lf, time = %lf, diff = %lf\n", integral, time, fabs(integral - S_EXPECTED));

                // Saving
                avg_time += time;
        }
        avg_time /= (double)TEST_COUNT;
        printf("avg_time = %lf\n", avg_time);
}

static double f_test(double x)
{
        double sqrt_x = sqrt(x);
        return 3.0 * sqrt_x * (1 + sqrt_x);
}
```

integral.h

```c
#pragma once

/// <summary>
/// Numeric integration Simpson method
/// single-thread implementation.
/// </summary>
/// <param name="f"> Function to integrate as fpointer. </param>
/// <param name="a"> Integration segment start. </param>
/// <param name="b"> Integration segment end. </param>
/// <param name="n"> Elementary segment count. </param>
/// <returns> Integral value as double. </returns>
double simpson_1t(double (*f)(double x), double a, double b, int n);

/// <summary>
/// Numeric integration Simpson method
/// multi-thread implementation.
/// </summary>
/// <param name="f"> Function to integrate as fpointer. </param>
/// <param name="a"> Integration segment start. </param>
/// <param name="b"> Integration segment end. </param>
/// <param name="n"> Elementary segment count. </param>
/// <returns> Integral value as double. </returns>
double simpson_mt(double (*f)(double x), double a, double b, int n);
```

integral.c

```c
#include "pch.h"

#include "integral.h"

#include <omp.h>

/// <summary>
/// Numeric integration Simpson method
/// single-thread implementation.
/// </summary>
/// <param name="f"> Function to integrate as fpointer. </param>
/// <param name="a"> Integration segment start. </param>
/// <param name="b"> Integration segment end. </param>
/// <param name="n"> Elementary segment count. </param>
/// <returns> Integral value as double. </returns>
double simpson_1t(double (*f)(double x), double a, double b, int n)
{
        // Calculation storage
        double p1 = 0, p2 = 0, p3 = 0, p4 = 0;

        // h value
        double h = (b - a) / (double)(n);

        // 1st part
        p1 = 0.5 * f(a);

        // 2nd part
        for (int i = 1; i <= n - 1; i++)
                p2 += f(a + (double)i * h);

        // 3rd part
        for (int i = 1; i <= n; i++)
                p3 += f((2.0 * a + (2 * (double)i - 1) * h) / 2.0);
        p3 *= 2.0;

        // 4th part
        p4 = 0.5 * f(b);

        // Return
        return (h / 3.0) * (p1 + p2 + p3 + p4);
}

/// <summary>
/// Numeric integration Simpson method
/// multi-thread implementation.
/// </summary>
/// <param name="f"> Function to integrate as fpointer. </param>
/// <param name="a"> Integration segment start. </param>
/// <param name="b"> Integration segment end. </param>
/// <param name="n"> Elementary segment count. </param>
/// <returns> Integral value as double. </returns>
double simpson_mt(double (*f)(double x), double a, double b, int n)
{
        // Calculation storage
        double p1 = 0, p2 = 0, p3 = 0, p4 = 0;
        int i;

        // h value
        double h = (b - a) / (double)(n);

        // 1st part
        p1 = 0.5 * f(a);

        // 2nd part
        #pragma omp parallel for reduction(+:p2) schedule(static)
        for (i = 1; i <= n - 1; i++)
                p2 += f(a + (double)i * h);

        // 3rd part
        #pragma omp parallel for reduction(+:p3) schedule(static)
        for (i = 1; i <= n; i++)
                p3 += f((2.0 * a + (2 * (double)i - 1) * h) / 2.0);
        p3 *= 2.0;

        // 4th part
        p4 = 0.5 * f(b);

        // Return
        return (h / 3.0) * (p1 + p2 + p3 + p4);
}
```

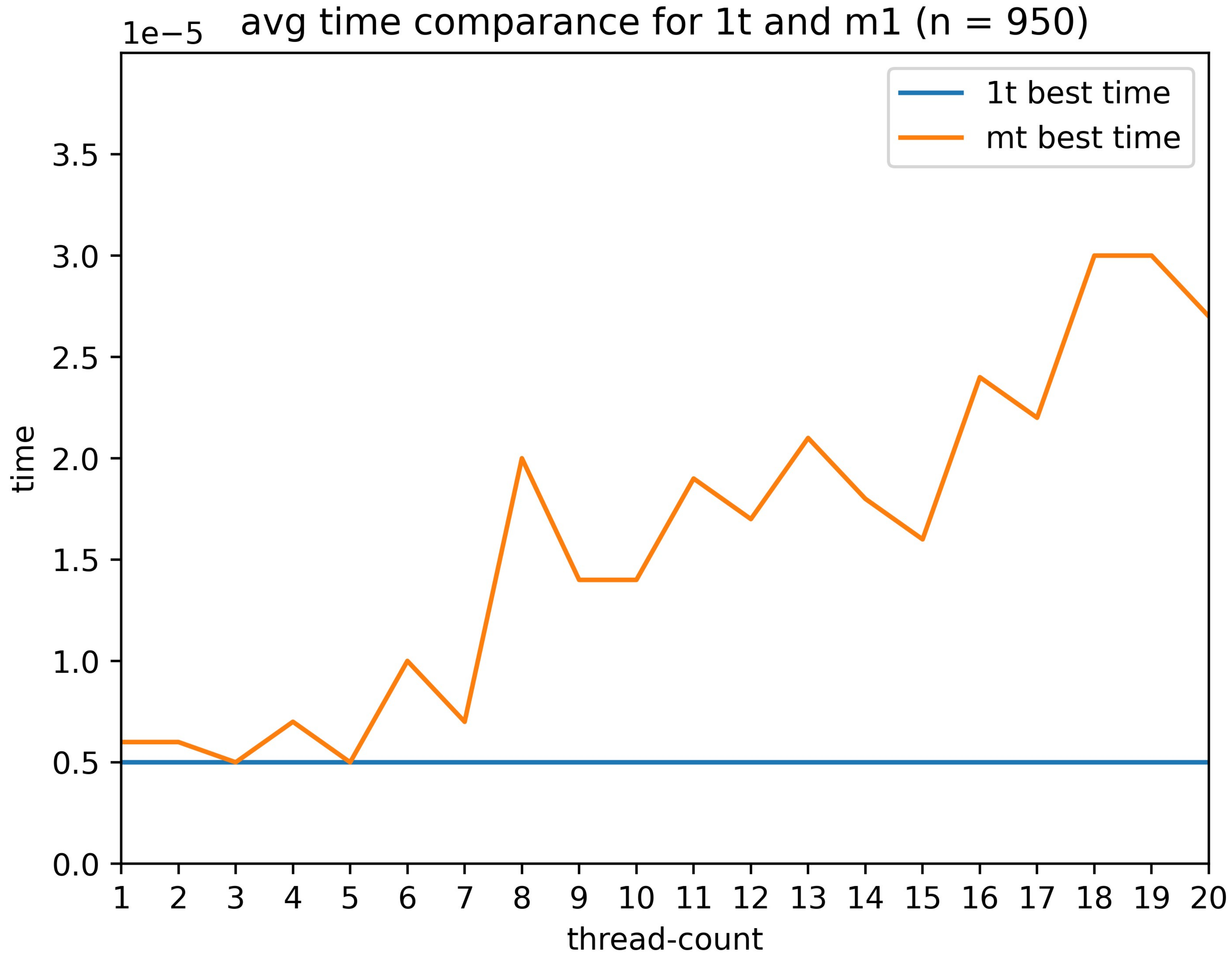
Дослідження швидкодії

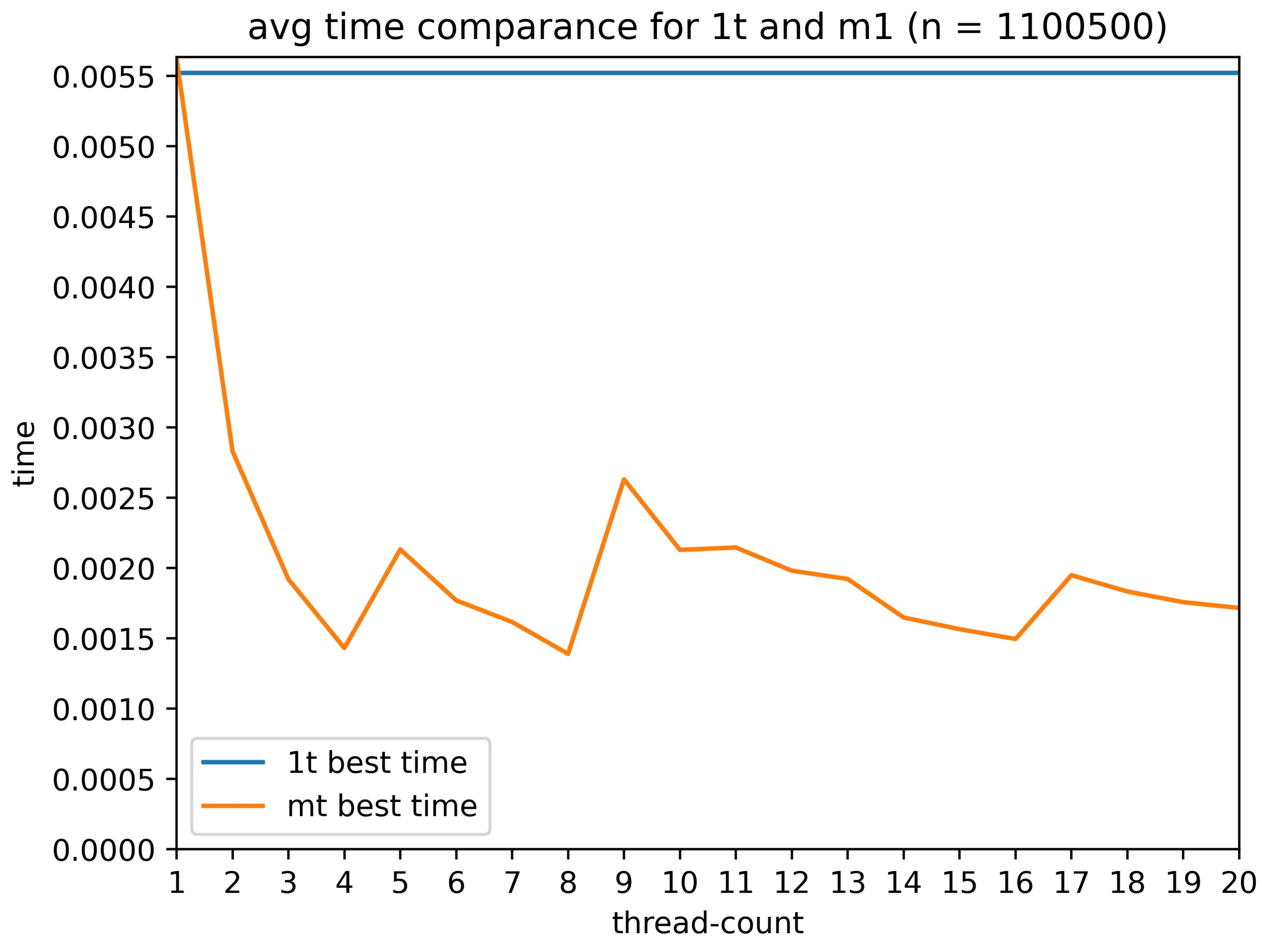
В дослідженні порівнюється швидкодія однопоточної та багатопоточної реалізацій методу для двох розмірностей розбиття: n1 = 950, n2 = 1100500. Кількість потоків для багатопоточної реалізації змінюється у межах [1; 20].

```
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe s 950
s_actual = 172.000000, time = 0.000005, diff = 0.000000
s_actual = 172.000000, time = 0.000005, diff = 0.000000
s_actual = 172.000000, time = 0.000005, diff = 0.000000
s_actual = 172.000000, time = 0.000005, diff = 0.000000
s_actual = 172.000000, time = 0.000005, diff = 0.000000
min time = 0.000005
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe s 1100500
s_actual = 172.000000, time = 0.005521, diff = 0.000000
s_actual = 172.000000, time = 0.005827, diff = 0.000000
s_actual = 172.000000, time = 0.005544, diff = 0.000000
s_actual = 172.000000, time = 0.005753, diff = 0.000000
s_actual = 172.000000, time = 0.005588, diff = 0.000000
min time = 0.005521
PS J:\repos\Parallel computing\Integral\x64\Release>
```

```
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 1
s_actual = 172.000000, time = 0.005714, diff = 0.000000
s_actual = 172.000000, time = 0.005625, diff = 0.000000
s_actual = 172.000000, time = 0.005648, diff = 0.000000
s_actual = 172.000000, time = 0.006128, diff = 0.000000
s_actual = 172.000000, time = 0.005680, diff = 0.000000
min time = 0.005625
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 2
s_actual = 172.000000, time = 0.002988, diff = 0.000000
s_actual = 172.000000, time = 0.002832, diff = 0.000000
s_actual = 172.000000, time = 0.002945, diff = 0.000000
s_actual = 172.000000, time = 0.002892, diff = 0.000000
s_actual = 172.000000, time = 0.002842, diff = 0.000000
min time = 0.002832
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 3
s_actual = 172.000000, time = 0.002304, diff = 0.000000
s_actual = 172.000000, time = 0.001920, diff = 0.000000
s_actual = 172.000000, time = 0.003713, diff = 0.000000
s_actual = 172.000000, time = 0.003526, diff = 0.000000
s_actual = 172.000000, time = 0.003540, diff = 0.000000
min time = 0.001920
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 4
s_actual = 172.000000, time = 0.002465, diff = 0.000000
s_actual = 172.000000, time = 0.001757, diff = 0.000000
s_actual = 172.000000, time = 0.001515, diff = 0.000000
s_actual = 172.000000, time = 0.001563, diff = 0.000000
s_actual = 172.000000, time = 0.001431, diff = 0.000000
min time = 0.001431
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 5
s_actual = 172.000000, time = 0.002986, diff = 0.000000
s_actual = 172.000000, time = 0.002237, diff = 0.000000
s_actual = 172.000000, time = 0.002146, diff = 0.000000
s_actual = 172.000000, time = 0.002132, diff = 0.000000
s_actual = 172.000000, time = 0.002132, diff = 0.000000
min time = 0.002132
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 6
s_actual = 172.000000, time = 0.002709, diff = 0.000000
s_actual = 172.000000, time = 0.001775, diff = 0.000000
s_actual = 172.000000, time = 0.002129, diff = 0.000000
s_actual = 172.000000, time = 0.001770, diff = 0.000000
s_actual = 172.000000, time = 0.001837, diff = 0.000000
min time = 0.001770
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 7
s_actual = 172.000000, time = 0.003040, diff = 0.000000
s_actual = 172.000000, time = 0.001679, diff = 0.000000
s_actual = 172.000000, time = 0.001857, diff = 0.000000
s_actual = 172.000000, time = 0.001616, diff = 0.000000
s_actual = 172.000000, time = 0.001671, diff = 0.000000
min time = 0.001616
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 8
s_actual = 172.000000, time = 0.002403, diff = 0.000000
s_actual = 172.000000, time = 0.001394, diff = 0.000000
s_actual = 172.000000, time = 0.001389, diff = 0.000000
s_actual = 172.000000, time = 0.001415, diff = 0.000000
s_actual = 172.000000, time = 0.002725, diff = 0.000000
min time = 0.001389
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 9
s_actual = 172.000000, time = 0.003458, diff = 0.000000
s_actual = 172.000000, time = 0.002631, diff = 0.000000
s_actual = 172.000000, time = 0.002648, diff = 0.000000
s_actual = 172.000000, time = 0.002686, diff = 0.000000
s_actual = 172.000000, time = 0.002974, diff = 0.000000
min time = 0.002631
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 10
s_actual = 172.000000, time = 0.003445, diff = 0.000000
s_actual = 172.000000, time = 0.002701, diff = 0.000000
s_actual = 172.000000, time = 0.002254, diff = 0.000000
s_actual = 172.000000, time = 0.002503, diff = 0.000000
s_actual = 172.000000, time = 0.002129, diff = 0.000000
min time = 0.002129
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 11
s_actual = 172.000000, time = 0.002680, diff = 0.000000
s_actual = 172.000000, time = 0.002146, diff = 0.000000
s_actual = 172.000000, time = 0.002188, diff = 0.000000
s_actual = 172.000000, time = 0.002726, diff = 0.000000
s_actual = 172.000000, time = 0.002190, diff = 0.000000
min time = 0.002146
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 12
s_actual = 172.000000, time = 0.002488, diff = 0.000000
s_actual = 172.000000, time = 0.001989, diff = 0.000000
s_actual = 172.000000, time = 0.001981, diff = 0.000000
s_actual = 172.000000, time = 0.002100, diff = 0.000000
s_actual = 172.000000, time = 0.002117, diff = 0.000000
min time = 0.001981
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 13
s_actual = 172.000000, time = 0.002706, diff = 0.000000
s_actual = 172.000000, time = 0.001922, diff = 0.000000
s_actual = 172.000000, time = 0.002132, diff = 0.000000
s_actual = 172.000000, time = 0.002793, diff = 0.000000
s_actual = 172.000000, time = 0.002510, diff = 0.000000
min time = 0.001922
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 14
s_actual = 172.000000, time = 0.002395, diff = 0.000000
s_actual = 172.000000, time = 0.001715, diff = 0.000000
s_actual = 172.000000, time = 0.001648, diff = 0.000000
s_actual = 172.000000, time = 0.001874, diff = 0.000000
s_actual = 172.000000, time = 0.001749, diff = 0.000000
min time = 0.001648
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 15
s_actual = 172.000000, time = 0.002636, diff = 0.000000
s_actual = 172.000000, time = 0.001691, diff = 0.000000
s_actual = 172.000000, time = 0.001565, diff = 0.000000
s_actual = 172.000000, time = 0.001589, diff = 0.000000
s_actual = 172.000000, time = 0.001572, diff = 0.000000
min time = 0.001565
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 16
s_actual = 172.000000, time = 0.002284, diff = 0.000000
s_actual = 172.000000, time = 0.001495, diff = 0.000000
s_actual = 172.000000, time = 0.002151, diff = 0.000000
s_actual = 172.000000, time = 0.002187, diff = 0.000000
s_actual = 172.000000, time = 0.001628, diff = 0.000000
min time = 0.001495
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 17
s_actual = 172.000000, time = 0.002716, diff = 0.000000
s_actual = 172.000000, time = 0.002027, diff = 0.000000
s_actual = 172.000000, time = 0.002025, diff = 0.000000
s_actual = 172.000000, time = 0.001951, diff = 0.000000
s_actual = 172.000000, time = 0.001949, diff = 0.000000
min time = 0.001949
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 18
s_actual = 172.000000, time = 0.002724, diff = 0.000000
s_actual = 172.000000, time = 0.001833, diff = 0.000000
s_actual = 172.000000, time = 0.001909, diff = 0.000000
s_actual = 172.000000, time = 0.002013, diff = 0.000000
s_actual = 172.000000, time = 0.001907, diff = 0.000000
min time = 0.001833
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 19
s_actual = 172.000000, time = 0.002554, diff = 0.000000
s_actual = 172.000000, time = 0.001781, diff = 0.000000
s_actual = 172.000000, time = 0.001763, diff = 0.000000
s_actual = 172.000000, time = 0.001757, diff = 0.000000
s_actual = 172.000000, time = 0.002273, diff = 0.000000
min time = 0.001757
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 1100500 20
s_actual = 172.000000, time = 0.003139, diff = 0.000000
s_actual = 172.000000, time = 0.001752, diff = 0.000000
s_actual = 172.000000, time = 0.001716, diff = 0.000000
s_actual = 172.000000, time = 0.001727, diff = 0.000000
s_actual = 172.000000, time = 0.001793, diff = 0.000000
min time = 0.001716
PS J:\repos\Parallel computing\Integral\x64\Release>
```

```
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 1
s_actual = 172.000000, time = 0.000011, diff = 0.000000
s_actual = 172.000000, time = 0.000023, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
min time = 0.000006
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 2
s_actual = 172.000000, time = 0.000265, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
min time = 0.000006
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 3
s_actual = 172.000000, time = 0.000444, diff = 0.000000
s_actual = 172.000000, time = 0.000018, diff = 0.000000
s_actual = 172.000000, time = 0.000024, diff = 0.000000
s_actual = 172.000000, time = 0.000005, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
min time = 0.000005
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 4
s_actual = 172.000000, time = 0.000482, diff = 0.000000
s_actual = 172.000000, time = 0.000025, diff = 0.000000
s_actual = 172.000000, time = 0.000009, diff = 0.000000
s_actual = 172.000000, time = 0.000007, diff = 0.000000
s_actual = 172.000000, time = 0.000007, diff = 0.000000
min time = 0.000007
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 5
s_actual = 172.000000, time = 0.000596, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
s_actual = 172.000000, time = 0.000007, diff = 0.000000
s_actual = 172.000000, time = 0.000006, diff = 0.000000
s_actual = 172.000000, time = 0.000005, diff = 0.000000
min time = 0.000005
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 6
s_actual = 172.000000, time = 0.001268, diff = 0.000000
s_actual = 172.000000, time = 0.000013, diff = 0.000000
s_actual = 172.000000, time = 0.000010, diff = 0.000000
s_actual = 172.000000, time = 0.000027, diff = 0.000000
s_actual = 172.000000, time = 0.000028, diff = 0.000000
min time = 0.000010
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 7
s_actual = 172.000000, time = 0.000732, diff = 0.000000
s_actual = 172.000000, time = 0.000024, diff = 0.000000
s_actual = 172.000000, time = 0.000007, diff = 0.000000
s_actual = 172.000000, time = 0.000027, diff = 0.000000
s_actual = 172.000000, time = 0.000012, diff = 0.000000
min time = 0.000007
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 8
s_actual = 172.000000, time = 0.000915, diff = 0.000000
s_actual = 172.000000, time = 0.000020, diff = 0.000000
s_actual = 172.000000, time = 0.000096, diff = 0.000000
s_actual = 172.000000, time = 0.000021, diff = 0.000000
s_actual = 172.000000, time = 0.000083, diff = 0.000000
min time = 0.000020
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 9
s_actual = 172.000000, time = 0.000847, diff = 0.000000
s_actual = 172.000000, time = 0.000057, diff = 0.000000
s_actual = 172.000000, time = 0.000020, diff = 0.000000
s_actual = 172.000000, time = 0.000014, diff = 0.000000
s_actual = 172.000000, time = 0.000016, diff = 0.000000
min time = 0.000014
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 10
s_actual = 172.000000, time = 0.001107, diff = 0.000000
s_actual = 172.000000, time = 0.000015, diff = 0.000000
s_actual = 172.000000, time = 0.000017, diff = 0.000000
s_actual = 172.000000, time = 0.000014, diff = 0.000000
s_actual = 172.000000, time = 0.000018, diff = 0.000000
min time = 0.000014
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 11
s_actual = 172.000000, time = 0.000930, diff = 0.000000
s_actual = 172.000000, time = 0.000019, diff = 0.000000
s_actual = 172.000000, time = 0.000020, diff = 0.000000
s_actual = 172.000000, time = 0.000059, diff = 0.000000
s_actual = 172.000000, time = 0.000021, diff = 0.000000
min time = 0.000019
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 12
s_actual = 172.000000, time = 0.000817, diff = 0.000000
s_actual = 172.000000, time = 0.000017, diff = 0.000000
s_actual = 172.000000, time = 0.000019, diff = 0.000000
s_actual = 172.000000, time = 0.000025, diff = 0.000000
s_actual = 172.000000, time = 0.000020, diff = 0.000000
min time = 0.000017
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 13
s_actual = 172.000000, time = 0.000859, diff = 0.000000
s_actual = 172.000000, time = 0.000022, diff = 0.000000
s_actual = 172.000000, time = 0.000039, diff = 0.000000
s_actual = 172.000000, time = 0.000021, diff = 0.000000
s_actual = 172.000000, time = 0.000067, diff = 0.000000
min time = 0.000021
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 14
s_actual = 172.000000, time = 0.001804, diff = 0.000000
s_actual = 172.000000, time = 0.000063, diff = 0.000000
s_actual = 172.000000, time = 0.000039, diff = 0.000000
s_actual = 172.000000, time = 0.000021, diff = 0.000000
s_actual = 172.000000, time = 0.000018, diff = 0.000000
min time = 0.000018
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 15
s_actual = 172.000000, time = 0.001163, diff = 0.000000
s_actual = 172.000000, time = 0.000081, diff = 0.000000
s_actual = 172.000000, time = 0.000027, diff = 0.000000
s_actual = 172.000000, time = 0.000016, diff = 0.000000
s_actual = 172.000000, time = 0.000032, diff = 0.000000
min time = 0.000016
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 16
s_actual = 172.000000, time = 0.001089, diff = 0.000000
s_actual = 172.000000, time = 0.000031, diff = 0.000000
s_actual = 172.000000, time = 0.000030, diff = 0.000000
s_actual = 172.000000, time = 0.000085, diff = 0.000000
s_actual = 172.000000, time = 0.000024, diff = 0.000000
min time = 0.000024
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 17
s_actual = 172.000000, time = 0.001066, diff = 0.000000
s_actual = 172.000000, time = 0.000048, diff = 0.000000
s_actual = 172.000000, time = 0.000064, diff = 0.000000
s_actual = 172.000000, time = 0.000022, diff = 0.000000
s_actual = 172.000000, time = 0.000025, diff = 0.000000
min time = 0.000022
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 18
s_actual = 172.000000, time = 0.001118, diff = 0.000000
s_actual = 172.000000, time = 0.000030, diff = 0.000000
s_actual = 172.000000, time = 0.000053, diff = 0.000000
s_actual = 172.000000, time = 0.000084, diff = 0.000000
s_actual = 172.000000, time = 0.000041, diff = 0.000000
min time = 0.000030
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 19
s_actual = 172.000000, time = 0.001409, diff = 0.000000
s_actual = 172.000000, time = 0.000087, diff = 0.000000
s_actual = 172.000000, time = 0.000052, diff = 0.000000
s_actual = 172.000000, time = 0.000030, diff = 0.000000
s_actual = 172.000000, time = 0.000035, diff = 0.000000
min time = 0.000030
PS J:\repos\Parallel computing\Integral\x64\Release> .\IntegralDemo.exe m 950 20
s_actual = 172.000000, time = 0.001924, diff = 0.000000
s_actual = 172.000000, time = 0.000061, diff = 0.000000
s_actual = 172.000000, time = 0.000033, diff = 0.000000
s_actual = 172.000000, time = 0.000034, diff = 0.000000
s_actual = 172.000000, time = 0.000027, diff = 0.000000
min time = 0.000027
PS J:\repos\Parallel computing\Integral\x64\Release>
```





Висновки:

- Паралелізм значно погіршує швидкодію для малої розмірності задачі (n = 950).
- Паралелізм значно покращує швидкодію для великої розмірності задачі (n = 1100500).
- Найшвидше багатопоточна реалізація (n=1100500) працює при thread_count = 8 (0.001389), 4 (0.001431), 16 (0.001495). Процесор, на якому здійснювалось тестування—Intel Core i5-8250U, має 4 ядер, 8 потоків. Таким чином доцільно обирати значення thread_count, кратне кількості фізичних потоків системи.