НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із лабораторної роботи №5

із дисципліни «Розподілені і хмарні обчислення»

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-03 | Ліскін В. О. |
| Шаповалов Г. Г. |  |

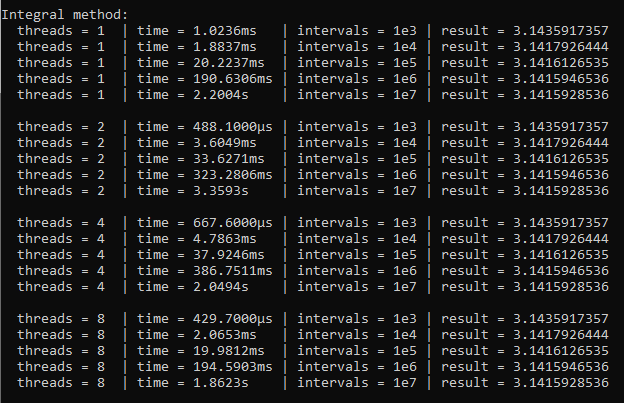
Київ — 2023

Мета роботи

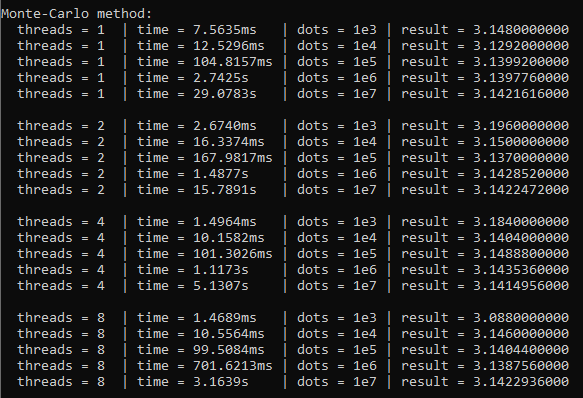
Розпаралелити метод обчислення константи PI

Опис програми

Завдання 1 – Знайти значення PI інтегральним методом:



Завдання 2 - Знайти значення PI методом Монте-Карло:



Висновки: Інтегральний метод виявився точнішим за Монте-Карло

Лістинг програми:

use std::time::Instant;

use lab\_5::constcalc::{picalc, mc\_picalc};

use rayon::ThreadPoolBuilder;

fn main() {

    task\_intergral();

    task\_monte\_carlo();

}

fn format\_dots(n: i32) -> String {

    let mut dots = n.to\_string();

    let len = dots.len();

    if len > 3 {

        let e = len - 1;

        dots = format!("1e{}", e);

    }

    dots

}

fn task\_intergral() {

    println!("\nIntegral method:");

    for &threads in [1, 2, 4, 8].iter() {

        let pool = ThreadPoolBuilder::new().num\_threads(threads).build().unwrap();

        for &n in [1e3 as i32, 1e4 as i32, 1e5 as i32, 1e6 as i32, 1e7 as i32].iter() {

            let start = Instant::now();

            pool.install(|| {

                let pi = picalc(n);

                let duration = start.elapsed();

                println!("  threads = {:<2} | time = {:<10} | intervals = {:<2} | result = {:.10}",

                        threads, format!("{:.4?}", duration), format\_dots(n), pi);

            });

        }

        println!();

    }

}

fn task\_monte\_carlo() {

    println!("\n\n\nMonte-Carlo method:");

    for &threads in [1, 2, 4, 8].iter() {

        let pool = ThreadPoolBuilder::new().num\_threads(threads).build().unwrap();

        for &n in [1e3 as i32, 1e4 as i32, 1e5 as i32, 1e6 as i32, 1e7 as i32].iter() {

            let start = Instant::now();

            pool.install(|| {

                let pi = mc\_picalc(n);

                let duration = start.elapsed();

                println!("  threads = {:<2} | time = {:<10} | dots = {:<2} | result = {:.10}",

                        threads, format!("{:.4?}", duration), format\_dots(n), pi);

            });

        }

        println!();

    }

}