Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики і обчислювальної техніки Кафедра інформаційних систем та технологій

Лабораторна робота №1

З дисципліни «Технології паралельного програмування в умовах великих даних»

Тема: «Паралельні обчислення в моделі зі спільною пам’яттю»

Виконали: Перевірив:

Студент групи ІА-11мн Жереб К. А.

Новиков Д.М.

Студент групи ІС-11мн

Іванов А.І.

Київ 2022

Тема: Паралельні обчислення в моделі зі спільною пам’яттю.

Завдання: Для обраної задачі необхідно реалізувати послідовну (однопоточну) реалізацію, а також мультипоточну реалізацію зі спільною пам’яттю. Бажано реалізувати можливість змінювати кількість потоків, що використовуються для обчислень. Необхідно порівняти швидкодію послідовної та мультипоточної реалізації, а також за можливості продемонструвати залежність часу роботи від розміру задачі, кількості потоків та інших параметрів (залежно від задачі). Реалізована програма може не мати графічного інтерфейсу, але можна також реалізувати графічний інтерфейс і правильну роботу з ним з різних потоків і отримати за це додаткові бали. Результатом виконання даної лабораторної роботи є працююча програма, а також звіт про використані технології та можливості, з результатами вимірів.

Репозиторій: <https://github.com/JokerFunny/parallel_programming>

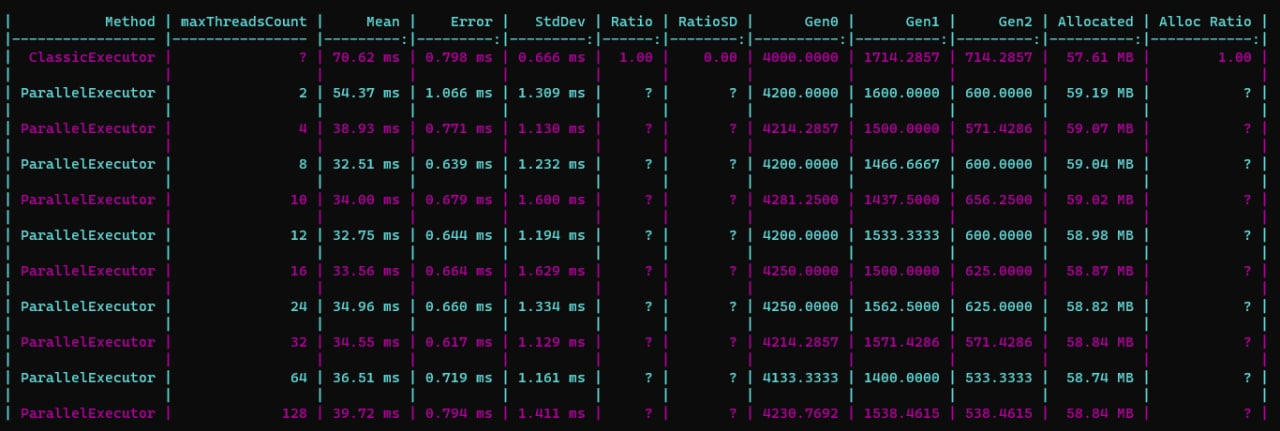
Виконання роботи:

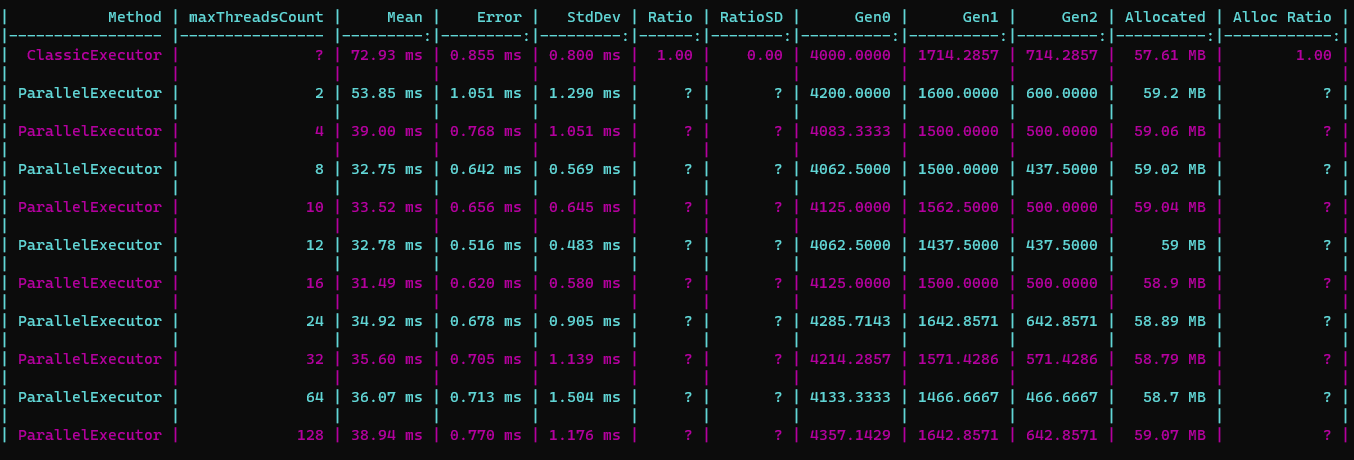
В якості корисного навантаження було обрано задачу: є великий файл, який зчитано рядково. В кожній строчці обчислити математичну формулу над кожним символом.

Реалізовано однопоточну та мультипоточну версії. Для мультипоточної версії було використано Parallel.For з простору імен System.Threading.Tasks. Він виконує цикл for, забезпечуючи можливість паралельного виконання ітерацій, а також контролю стану циклу та управління цим станом.

Для заміру результатів було використано бібліотеку [BenchmarkDotNet](https://github.com/dotnet/BenchmarkDotNet) - допомагає перетворювати методи на еталонні показники, відстежувати їх ефективність і ділитися відтворюваними експериментами вимірювання. Під капотом він виконує багато магії, яка гарантує надійні та точні результати завдяки системі статистики perfolizer. BenchmarkDotNet захищає користувача від популярних помилок порівняльного аналізу та попереджає користувача, якщо щось не так проектом порівняльного аналізу чи отриманими вимірюваннями. Результати представлені в зручній формі, яка висвітлює всі важливі факти про експеримент. Бібліотека використовується в 11700+ проектах, включаючи .NET Runtime, і підтримується .NET Foundation.

Результат роботи:





Як можна побачити, багатопоточний варіант має кращий час виконання, найбільш результативним є при використанні 8/16 потоків (що еквівалентно кількості фізичних/віртуальних ядер мого процесору) та виконується швидше у 2.17 разів. При збільшені кількості потоків, швидкість стає гіршою через затримки на синхронізацію. У цьому конкретному випадку це навантаження на пам’ять; алгоритм потребує більше пам'яті для обчислень і обмежені продуктивністю основної шини пам'яті. Це означає, що поки один апаратний потік очікує пам'яті, інший може продовжувати виконання.

Висновки:

В результаті виконання лабораторної роботи було реалізовано алгоритм для вирішення поставленої задачі за допомогою однопоточного та мультипоточного підходу використовуючи спільну пам'ять. Перевірено ефективність кожного підходу та проведено порівняння результатів.