Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 6 з дисципліни
«Технології паралельних обчислень»

«Розробка паралельного алгоритму множення матриць з використанням MPI-методів обміну повідомленнями «один-до-одного» та дослідження його ефективності»

Виконав(ла) *ІП-14 Сергієнко Ю. В.* _____
(шифр, прізвище, ім'я, по батькові)

Перевірив *Дифучина О. Ю.* _____
(прізвище, ім'я, по батькові)

Київ 2024

Комп’ютерний практикум 6

Тема: Розробка паралельного алгоритму множення матриць з використанням MPI-методів обміну повідомленнями «один-до-одного» та дослідження його ефективності

Виконання:

1. Реалізувати алгоритм паралельного множення матриць з використанням розподілених обчислень в MPI з використанням методів блокуючого обміну повідомленнями (лістинг 1). **30 балів**.

Для виконання даного завдання було використано приклад реалізації у лістингу з використанням OpenMPI. Було виконано множення матриць з використанням декількох процесів одночасно, де процес з rank = 0 – майстер. Кількість процесів задається параметром під час запуску програми. За допомогою методів *MPI_Comm_size()* та *MPI_Comm_rank()* будемо визначати к-сть задіяних процесів та поточний процес. У випадку, якщо процесів задіяно менше за 2, будемо закінчувати виконання.

У методі *multiplyBlocking()* після ініціалізації даних будемо перевіряти поточний процес, якщо це майстер – будемо надсилати дані, інакше – обробляти. У випадку майстера, необхідно задати початкові матриці та почастинно надіслати певному воркеру. Воркер буде отримувати частину рядків першої матриці та всю другу матрицю для знаходження результату частини. Оскільки воркер має свій ІД, зможемо отримати унікальну частину для обчислень. Після перемноження матриць, будемо повертати offset (індекс рядка, де ми зараз знаходимось), к-сть оброблених рядків та результуючу матрицю.

Далі майстер буде збирати частини в цілісну матрицю та виводити результат. Щоб заміряти результат виконання, було використано *MPI_Wtime()*. Результат роботи програми можемо бачити на рисунку 1.

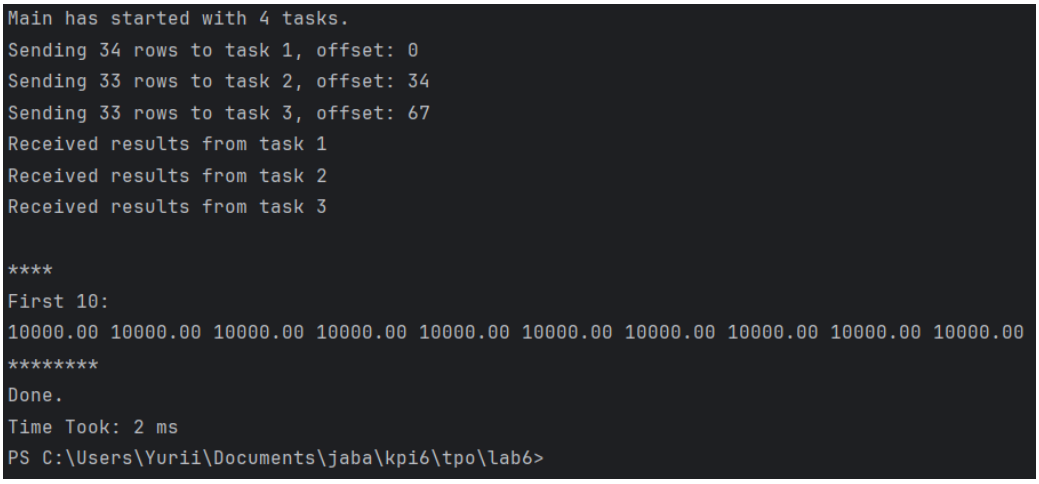

Рисунок 1 – Множення матриці блокуючим обміном повідомлень

2. Реалізувати алгоритм паралельного множення матриць з використанням розподілених обчислень в MPI з використанням методів неблокуючого обміну повідомленнями. **30 балів**.

Для реалізації даного способу було використано основні його методи: *MPI_Isend()*, *MPI_Irecv()*, *MPI_Wait()* та *MPI_Waitall()*. На практиці, методи відправки та отримання не дочікуються виконання своїх дій і йдуть далі. Для коректного отримання даних воркером (у випадку, коли матриця дійшла швидше за rows), будемо використовувати *MPI_Recv()*. Також, будемо дочікуватись к-сть рядків та offset для результуючої матриці у майстері за допомогою MPI_Requests. Дані на обробку будемо відправляти не дочікуючись закінчення дії, адже нам не важливий результат. Результат виконання програми на рисунку 2.

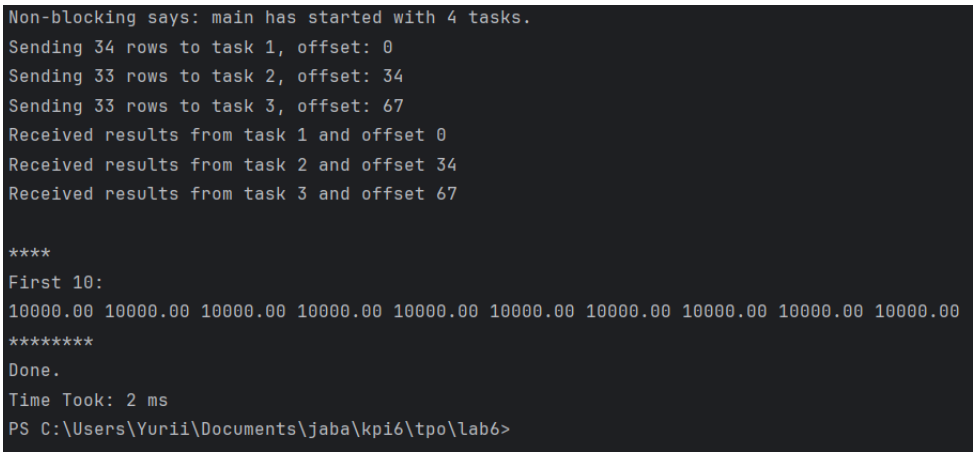

Рисунок 2 – Множення матриці неблокуючим обміном

3. Дослідити ефективність розподіленого обчислення алгоритму множення матриць при збільшенні розміру матриць та при збільшенні кількості вузлів, на яких здійснюється запуск програми. Порівняйте ефективність алгоритму при використанні блокуючих та неблокуючих методів обміну повідомленнями. **40 балів**.

Для порівняння ефективності методів використаємо два основні параметри: кількість елементів матриці та к-сть процесів. Результати дослідежнь зображені у таблиці 1.

Таблиця 1 – Результати досліджень

Size	Threads	Blocking, ms	Non-blocking, ms	Acc, t
500	2	496	489	1.01
	6	146	120	1.2
	12	88	86	1.02
1000	2	3992	3951	1.01
	6	947	902	1.05
	12	703	697	1.01
1500	2	14218	14253	0.998
	6	3497	3191	1.1
	12	2677	2692	0.994

Результати прискорення неблокуючим обміном можемо бачити на рисунку 4.

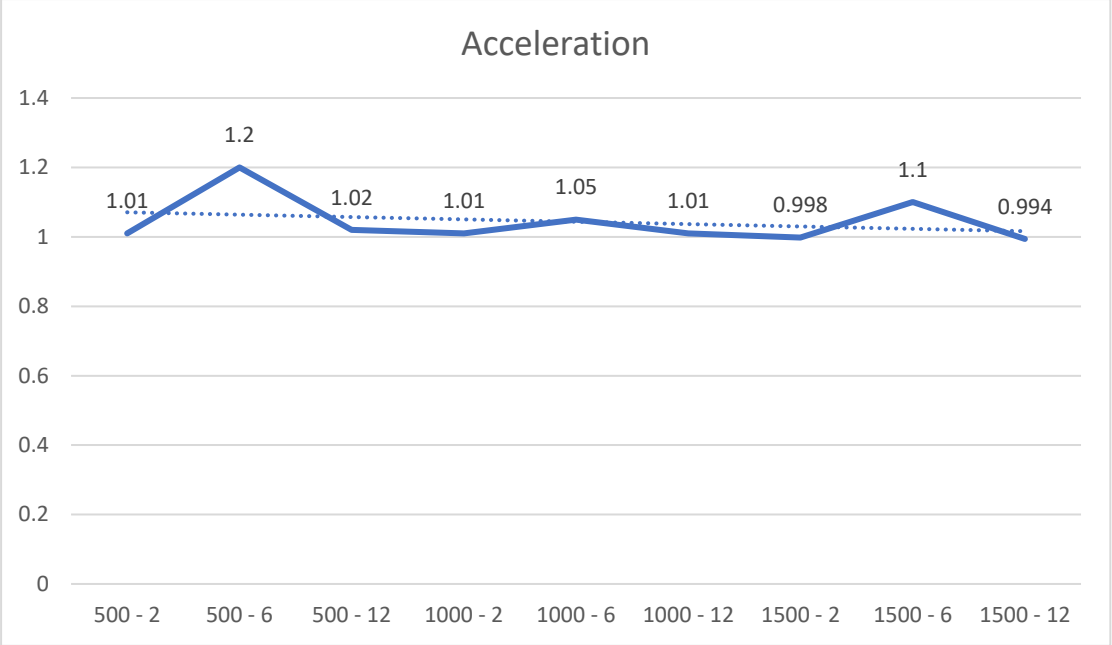

Рисунок 4 – Результати прискорення

Висновок

Під час виконання даного комп’ютерного практикуму я здобув здання паралельного управління процесами комп’ютера за допомогою використання стандарту Message Passing Interface (MPI) мовою C++ (OpenMPI). Було паралельно помножено матриці методами один-до-одного, а саме блокуючі методи обміну повідомлень та неблокуючі. Було досліджено ефективність обох методів, і в результаті неблокуючий метод пересилання повідомлень виявився частково швидшим.

Код програми доступний на Github.