УДК 004.8

О.А.Кушнір
В.О. Денисюк

РЕАЛІЗАЦІЯ МАТРИЧНОГО МНОЖЕННЯ ЗА ДОПОМОГОЮ MESH NETWORK

Вінницький національний технічний університет

Анотація

Розглянуто розробку алгоритму матричного множення за допомогою Mesh Network з використанням технології MPI. Розглянуто питання аналізу існуючих методів розв'язання задачі обчислень для досягнення високої продуктивності, обґрунтовано вибір засобів розробки програмного модуля, розроблено діаграми класів програмного модуля, обґрунтовано вибір програмного середовища реалізації. У роботі створено програмну реалізацію паралельного алгоритму з використанням MPI та проведено тестування його продуктивності на різних наборах даних. Використання результатів дозволить покращити швидкодію і продуктивність програм та алгоритмів, які потребують обробки великих об'ємів даних.

Ключові слова: *матричне множення, MPI, Mesh Network, продуктивність*.

Abstract

The development of a matrix multiplication algorithm using Mesh Network using MPI technology is considered. The issue of analyzing existing methods for solving a computational problem to achieve high performance is considered, the choice of software module development tools is justified, class diagrams of the software module are developed, the choice of software implementation environment is justified. In the work, a software implementation of a parallel algorithm using MPI is created and its performance is tested on various data sets. Using the results will allow improving the speed and performance of programs and algorithms that require processing large amounts of data..

Keywords: *matrix multiplication, MPI, Mesh Network, performance*.

Вступ

Актуальність реалізації матричного множення за допомогою Mesh Network полягає у тому, що особливістю Mesh-Network є значне прискорення обчислень, а це є критично важливим для сучасних задач обробки великих обсягів даних. Таких як - машинне навчання, обробка зображень, наукове моделювання. Використання паралельних обчислювальних архітектур дозволяє одночасно виконувати множення елементів матриць на різних вузлах мережі, що суттєво скорочує час виконання операцій. Це забезпечує масштабованість і високу продуктивність для розв'язання складних математичних задач у таких сферах, як наука, інженерія та бізнес-аналітика.

Мережі типу Mesh можуть використовуватися для розподіленого обчислення, де кожен вузол виконує частину обчислень, і результати об'єднуються в кінцевий результат, що є особливо корисним в сучасних обчислювальних системах, де паралелізм може покращити швидкодію та продуктивність.

Метою роботи є розробка паралельного алгоритму, що реалізує матричне множення за допомогою Mesh Network для покращення швидкодії та продуктивності обчислювальних систем.

Постановка задачі дослідження

Задачі дослідження полягають у вирішенні наступних питань:

- розробка ефективної архітектури Mesh Network для оптимального розподілу обчислень;
- мінімізація затримок передачі даних між вузлами;
- розробка програми для обчислення матричного множення за допомогою Mesh Network;
- тестування програми та аналіз отриманих результатів.

Виклад основного матеріалу

Матричне множення використовується для розв'язання систем лінійних рівнянь, відображення геометричних трансформацій у комп'ютерній графіці, а також у багатьох алгоритмах машинного навчання для оптимізації ваг та здійснення прогнозів [1-3]. Методи розв'язання включають: стандартний алгоритм з тривалою складністю $O(n^3)$; алгоритм Штрассена, складність біля $O(n^{2.08074})$, для полегшення обчислень[4]; а також більш ефективні методи в розподілених та паралельних обчисленнях для швидшого виконання операцій над великими матрицями [5].

Для того, щоб правильно визначити ідею використання Mesh Network в розподілених системах та паралельних обчисленнях, виконано дослідження у рамках комп'ютерних мереж, що допоможе використати Mesh Network для матричного множення [6, 7].

Mesh Network – це бездротова система, що складається з кількох комп'ютерів, об'єднаних мережею. Кожен комп'ютер у мережі посилає власні сигнали та передає інформацію з інших комп'ютерів. Вузли у сітчастій мережі пов'язані один з одним через виділене з'єднання. Поєднання вузлів дозволяє інформації переміщатися від вузла до вузла без затримок або збоїв. Mesh-мережі також називають «самонастроюваними» мережами, оскільки новий вузол автоматично стає частиною існуючої структури мережі [6, 7].

Mesh-мережа може з'єднати кілька пристроїв розумного будинку без шкоди для якості з'єднання. Основний вузол підключається до порту або модему глобальної мережі (WAN) бездротової мережі . Вузли підключаються за допомогою налаштувань Wi-Fi, які включають назву сітчастої мережі. Під час початкового налаштування мережі всі вузли повинні бути підключені до одного джерела живлення. Отже, навіть якщо один вузол виходить з ладу, вся мережа може продовжувати функціонувати.

На відміну від традиційного розширювача Wi-Fi, сітчаста мережа не створює нових точок доступу. Це мережа, де кожен блок повторює сигнал попереднього. Оскільки сітчаста мережа є частиною однієї мережі, вона має набагато ширший діапазон, ніж розширювач Wi-Fi. Мережа Mesh допомагає досягти стабільного з'єднання Wi-Fi.

Mesh-Network можуть бути семи різних типів:

- мережа Wi-Fi: Бездротова сітчаста мережа (WMN);
- дротова мережа;
- мережа з повною сітчастою топологією;
- мережа з частковою сітчастою топологією;
- гібридна сітчаста мережа;
- інфраструктура Mesh-архітектури мережі;
- клієнтська Mesh-архітектура мережі.

Основні переваги та ідеї застосування Mesh Network включають:

- локальний обмін даними, тобто кожен вузол мережі спілкується тільки зі своїми безпосередніми «сусідами», що сприяє локальному обміну даними та зменшенню навантаження на комунікації в мережі;
- завдяки прямим з'єднанням з сусідніми вузлами, можливе паралельне виконання обчислень на різних вузлах, що дозволяє розподіляти завдання та прискорювати обчислення;
- структура Mesh Network легко масштабується, додавання нових вузлів може бути здійснене без значного впливу на існуючу мережу;
- локальний характер обміну даними може робити мережу менш чутливою до великої кількості вузлів та може підвищувати ефективність у випадках обчислень з обмеженими ресурсами;
- можливість локального обміну даними може знижувати час передачі інформації між вузлами, зменшуючи таким чином затримки у паралельних обчисленнях.

Розглянуто та використано чотири основних способи оптимізація алгоритму матричного множення за допомогою Mesh Network.

1. За рахунок паралелізації обчислень. Розподіл даних - розподіл матриць між вузлами мережі, розділено матрицю A на рядки, а матрицю B на стовпці, що дозволяє кожному вузлу обчислювати частину результату. Використання потоків - обчислюються частини матричного добутку на кожному вузлі, що дозволило паралельно виконувати обчислення.
2. За рахунок архітектури Mesh Network. Вузли повинні мати можливість отримувати та надсилати дані до інших вузлів, що забезпечує швидкий обмін даними між вузлами. Зв'язки повинні використовувати протоколи з низькою затримкою для обміну повідомленнями між вузлами: TCP/IP або UDP.
3. За рахунок використання специфічних алгоритмів. Алгоритм блокового множенн, який дозволяє зменшити кількість звернень до пам'яті та підвищити кеш-ефективність. Блоки матриць розподіляються між вузлами для паралельного обчислення. Алгоритм Strasse, який є алгоритмом множення матриць, більш ефективний за традиційний метод, був адаптований для паралельних обчислень у Mesh Network.
4. За рахунок взаємодії між вузлами. Обмін даними відбувається після завершення обчислень надсиланням результатів міх вузлами для об'єднання частин матричного добутку. Синхронізація забезпечує правильність обчислень і уникнення конфліктів при доступі до даних.

Програмно реалізовано задану задачу та описано всі компоненти коду [8].

Аналіз результатів тестування програми виконано для різної кількості елементів масиву (табл.1-3).

Таблиця 1 – Час виконання програми на різних вхідних даних

Кількість процесів	Загальна розмірність матриці		
	500	1000	5000
2	2.88657	27.0654	750.763
4	2.55853	26.1853	729.933
8	2.62002	25.5974	770.197

Таблиця 2 – Коефіцієнти прискорення

Загальна розмірність матриці	500	1000	5000
Коефіцієнт прискорення	0.8863	0.9674	0.9722

Таблиця 3 – Коефіцієнти ефективності

Кількість процесів	2	4	8
Коефіцієнт ефективності	0.4861	0.24305	0.1215

Проаналізувавши обчислені коефіцієнти прискорення та коефіцієнти ефективності при різних вхідних даних і різних кількостях процесів (табл.1-3) можливо зробити висновки:

- при невеликих вхідних даних значення коефіцієнта прискорення та коефіцієнта ефективності є близькими до нуля, тому кількість процесів при виконанні програми з відносно невеликими вхідними даними фактично не має значення;
- збільшення кількості процесів при виконанні задачі з великими вхідними даними призводить до відносно незначного збільшення коефіцієнта прискорення, проте значного зменшення коефіцієнта ефективності.

Отже, основними чинниками, які впливають на зміну розглянутих показників, таких як коефіцієнт прискорення, коефіцієнт ефективності та час виконання програми, є розмір вхідних даних, в даному випадку розмірність досліджуваної матриці, кількість процесів у програмній

реалізації і також метод виконання матричного множення, якщо замінити розглянутий наївний метод на будь-який інший, якість досліджуваних показників може відповідно покращитись або погіршитись.

Висновки

Досліджено матричне множення, як одну із ключових операцій в лінійній алгебрі, визначено основні сфери її застосування. Розглянуто три різні методи виконання операції матричного множення, а саме «наївний» метод, метод Штрассена та метод Коппермана-Віноградa, описано алгоритми їх вирішення та сфери застосування. На основі літературних джерел досліджено Mesh Network, як складову частину комп'ютерних мереж, розглянуто сім типів Mesh мереж та на основі цих досліджень сформульовано ідею Mesh Network відносно розподілених систем та паралельних обчислень.

Програмно реалізовано задану задачу та описано всі компоненти коду. Також наведено UML-діаграми. Проведено тестування розробленої програми, проаналізовано її результати, досліджено коефіцієнти прискорення та ефективності.

Визначено, що збільшення кількості процесів призводить до відносно незначного збільшення коефіцієнта прискорення, та значного зменшення коефіцієнта ефективності.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Літнарович Р. М. Алгебра матриць. Курс лекцій. МЕГУ. Рівне, 2007, 112 с. URL: https://core.ac.uk/download/pdf/14034615.pdf .
2. Савастру О. В. Матриці та системи лінійних рівнянь: навч. посіб. / О. В. Савастру, О. М. Яковлєва, С. В. Драганюк, О. М. Болдарєва, під ред. О. В. Савастру. Одеса: Одес. нац. ун-т ім. І. І. Мечникова, 2019. 120 с. URL: http://dspace.pdpu.edu.ua/jspui/bitstream/123456789/5300/1/2019%20%281%29.pdf .
3. Новотарський М. А. Алгоритми та методи обчислень. Київ: КПІ ім. Ігоря Сікорського, 2019. 407 с. URL: https://ela.kpi.ua/server/api/core/bitstreams/7421218e-d7dd-4e75-aa3e-bd7979db4e6d/content.
4. Множення матриць URL: https://uk.wikipedia.org/wiki/Множення_матриць.
5. Минайленко Р.М. Паралельні та розподілені обчислення: Навчальний посібник.Кропивницький: Видавець Лисенко В. Ф., 2021. 153 с. URL: https://dspace.kntu.kr.ua/server/api/core/bitstreams/396e02d2-725b-47b5-a1c0-ae07a9bec326/content .
6. How Does Mesh Network Allow IoT Devices To Communicate? URL: https://boosthigh.com/mesh-network-for-iot-devices/ .
7. What Is a Mesh Network? Meaning, Types Working, and Applications in 2022 URL: https://www.spiceworks.com/tech/networking/articles/what-is-mesh-network .
8. MPI для C++. URL: https://www.paulnorvig.com/guides/using-mpi-with-c.htm .

Кушнір Олександр Анатолійович – студент кафедри комп`ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: oleksandrkusnir25@gmail.com;

Денисюк Валерій Олександрович – канд. техн. наук, доцент, доцент кафедри комп'ютерних наук, Вінницький національний технічний університет, м.Вінниця, e-mail: vad64@i.ua.

Kushnir Oleksandr Anatoliyovych – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: oleksandrkusnir25@gmail.com;

Denysiuk Valerii Olexandrovich – Ph.D., Assistant Professor, Assistant Professor of the Chair of Computer Science, Vinnytsia National Technical University, Vinnytsia, e-mail: vad64@i.ua