УДК 004.8

Т.С. Мельник
В.О. Денисюк

РЕАЛІЗАЦІЯ МАТРИЧНОГО МНОЖЕННЯ З ВИКОРИСТАННЯМ ТОПОЛОГІЇ MESH NETWORK НА ОСНОВІ DIRECTCOMPUTE

Вінницький національний технічний університет

Анотація

У роботі розглянуто реалізацію алгоритму множення матриць із використанням топології Mesh Network та технології DirectCompute. Проаналізовано особливості гібридної архітектури обчислювальної системи, у якій центральний процесор виконує керуючі функції, а графічний процесор застосовується для масово-паралельних обчислень. Обґрунтовано доцільність використання сіткової топології для моделювання потокових обчислень і конвеєрного виконання обчислювальних шейдерів. Розроблено програмну систему у вигляді консольного додатка мовою C# у середовищі .NET, що імітує роботу DirectCompute та взаємодію між CPU і GPU. Запропоноване рішення демонструє ефективність використання паралельної обробки даних для задач матричного множення та може бути використане як навчальна модель для дослідження сучасних підходів до GPU-обчислень.

Ключові слова: *матричне множення, Mesh Network, DirectCompute, GPU, паралельні обчислення.*

Abstract

The paper considers the implementation of a matrix multiplication algorithm using the Mesh Network topology and DirectCompute technology. The features of a hybrid computing architecture are analyzed, where the central processing unit performs control functions, and the graphics processing unit is used for massive parallel computations. The feasibility of using a mesh topology for modeling streaming computations and pipeline execution of compute shaders is substantiated. A software system is developed as a console application in C# using the .NET environment, which simulates DirectCompute mechanisms and CPU–GPU interaction. The proposed solution demonstrates the effectiveness of parallel data processing for matrix multiplication tasks and can be used as a training model for studying modern GPU computing approaches.

Keywords: *matrix multiplication, Mesh Network, DirectCompute, GPU, parallel computing.*

Вступ

Зростання обсягів даних і складності обчислювальних задач у сучасних інформаційних системах зумовлює необхідність використання паралельних методів обробки. Однією з базових операцій чисельних і наукових обчислень є множення матриць, яке широко застосовується у задачах комп'ютерної графіки, машинного навчання, обробки сигналів та моделювання фізичних процесів. Послідовна реалізація цієї операції на центральному процесорі часто не забезпечує необхідної продуктивності, особливо для матриць великого розміру [1-3].

Ефективним способом прискорення таких обчислень є використання графічних процесорів, які характеризуються великою кількістю обчислювальних ядер та високим рівнем паралелізму [4]. Технологія DirectCompute дозволяє задіяти GPU для загальних обчислень, не пов'язаних безпосередньо з візуалізацією, що робить її доцільною для реалізації алгоритмів матричної алгебри.

Постановка задачі дослідження

Задачі дослідження полягають у вирішенні наступних питань:

- проаналізувати особливості паралельного множення матриць і можливості його реалізації на GPU;
- дослідити принципи побудови топології Mesh Network та її застосування для потокових обчислень;

- розробити архітектуру програмної системи з розподілом функцій між CPU і GPU;
- реалізувати програмну модель DirectCompute у середовищі .NET мовою C#;
- забезпечити коректне отримання та аналіз результатів обчислень.

Виклад основного матеріалу

Матричне множення є однією з базових операцій у комп'ютерних науках і широко застосовується в задачах чисельного моделювання, комп'ютерної графіки, обробки сигналів, машинного навчання та аналізу великих даних [1-3]. Продуктивність виконання цієї операції суттєво впливає на загальну ефективність програмних систем, особливо при роботі з матрицями великої розмірності. Класичний алгоритм множення матриць характеризується високою обчислювальною складністю, що обумовлює необхідність застосування паралельних методів обробки даних.

Ефективним апаратним засобом для реалізації масово-паралельних обчислень є графічні процесори, які здатні виконувати велику кількість однотипних операцій одночасно.

Використання GPU дозволяє значно скоротити час виконання арифметичних операцій, характерних для матричної алгебри. Технологія DirectCompute надає можливість застосовувати обчислювальні ресурси графічного процесора для задач загального призначення, реалізуючи паралельні алгоритми у вигляді обчислювальних шейдерів без прив'язки до графічного конвеєра [4].

У роботі для організації паралельних обчислень використано топологію Mesh Network, яка передбачає впорядковане розміщення обчислювальних вузлів у вигляді двовимірної сітки [5, 6]. Кожен вузол взаємодіє лише з найближчими сусідами, що забезпечує локальністьпередачі даних та зменшує накладні витрати на комунікацію. Такий підхід дозволяє ефективно моделювати потокові обчислення та реалізувати алгоритм множення матриць у вигляді конвеєра з передачею часткових результатів між логічними вузлами сітки.

Обчислювальний процес організовано у гібридній архітектурі, де центральний процесор виконує функції керування, а графічний процесор — основні масово-паралельні обчислення. CPU відповідає за ініціалізацію обчислювального середовища, перевірку коректності вхідних даних, управління пам'яттю та координацію фаз виконання. GPU використовується для паралельного виконання операцій множення та додавання елементів матриць у відповідності до обраної топології Mesh Network.

Програмну реалізацію виконано у вигляді консольного додатка мовою C# у середовищі .NET [7]. Клас ComputeController реалізує роль головного диспетчера системи та координує взаємодію між CPU і GPU. Управління обчислювальними ресурсами та буферами даних абстраговано за допомогою класу BufferManager, який імітує механізми створення, прив'язки та передачі даних у відеопам'ять, характерні для DirectCompute. Представлення вхідних і вихідних матриць здійснюється класом Matrix, що забезпечує зручну роботу з даними та контроль коректності отриманих результатів.

Запропонована програмна модель дозволяє наочно продемонструвати принципи побудови паралельних алгоритмів із використанням сіткових топологій та гібридної архітектури CPU–GPU. Отримані результати підтверджують доцільність використання Mesh Network і технології DirectCompute для реалізації задач матричного множення та можуть бути використані як основа для подальших досліджень у галузі високопродуктивних обчислень (табл.1).

Таблиця 1 – Порівняння режимів виконання алгоритму множення матриць

Режим виконання	Обчислювальна платформа	Характер виконання
Послідовний	CPU	Послідовний
Паралельний	GPU (DirectCompute)	Масово-паралельний
Гібридний	CPU + GPU	Керований паралелізм

Аналіз режимів виконання алгоритму множення матриць (табл.1) показує, що гібридний підхід із використанням графічного процесора дозволяє ефективно розподілити обчислювальне навантаження між CPU та GPU. Послідовний режим характеризується простотою реалізації,

проте має обмежену продуктивність при зростанні розмірів матриць. Паралельний режим, реалізований на GPU, забезпечує масово-паралельну обробку елементів матриць, що відповідає природі задачі. Гібридна модель поєднує переваги обох підходів, використовуючи CPU для керування обчислювальним процесом та GPU для виконання ресурсоємних арифметичних операцій (табл.2).

Таблиця 2 – Етапи обчислювального конвеєра множення матриць у топології Mesh Network

Етап конвеєра	Опис операції	Виконавчий компонент	Характер обробки
Ініціалізація	Перевірка розмірів матриць, створення логічних буферів даних	CPU (ComputeController)	Послідовний
Передача даних	Копіювання вхідних матриць у відеопам'ять	CPU → GPU (BufferManager)	Пакетний
Обчислення	Паралельне множення та додавання елементів матриць відповідно до топології Mesh Network	GPU (обчислювальні шейдери)	Масово-паралельний
Агрегація результатів	Формування фінальної матриці з часткових результатів	GPU	Паралельний
Отримання результату	Зчитування результатів з відеопам'яті у системну пам'ять	GPU → CPU	Послідовний

Аналіз етапів обчислювального конвеєра (табл. 2) демонструє чіткий розподіл функцій між центральним і графічним процесорами у гібридній архітектурі. Початкові та завершальні етапи виконуються на CPU та пов'язані з керуванням і передачею даних, тоді як основні обчислювальні операції реалізуються на GPU у масово-паралельному режимі. Така організація дозволяє мінімізувати навантаження на центральний процесор і максимально ефективно використати обчислювальні ресурси графічного процесора, що відповідає концепції Mesh Network та конвеєрного виконання обчислень.

Висновки

У роботі розглянуто задачу множення матриць як одну з базових операцій паралельних обчислень та обґрунтовано доцільність її реалізації з використанням графічних процесорів. Проаналізовано особливості класичного алгоритму матричного множення та визначено обмеження його послідовного виконання на центральному процесорі при зростанні розмірів вхідних даних. Запропоновано підхід до організації обчислювального процесу на основі топології Mesh Network, що дозволяє ефективно розподіляти обчислювальне навантаження між паралельними потоками та забезпечує локальну передачу часткових результатів. Реалізація конвеєрної моделі виконання сприяє підвищенню ступеня паралелізму та узгодженій взаємодії між обчислювальними етапами.

Розроблено програмну систему у вигляді консольного додатка мовою C# у середовищі .NET, що імітує механізми технології DirectCompute та взаємодію між CPU і GPU. У межах гібридної архітектури центральний процесор використовується для керування обчислювальним процесом і підготовки даних, тоді як графічний процесор виконує основні масово-паралельні арифметичні операції.

Проведений структурний аналіз режимів виконання та етапів обчислювального конвеєра підтверджує, що гібридний підхід із використанням GPU є доцільним для задач матричного множення. Запропонована модель може бути використана як навчальна та дослідницька основа для вивчення принципів побудови паралельних алгоритмів і високопродуктивних обчислювальних систем.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Національний технічний університет України «КПІ ім. Ігоря Сікорського». Навчальні матеріали з лінійної алгебри. URL: https://ela.kpi.ua/server/api/core/bitstreams/d605e9e7-d1dc-4ee1-88ad-32a08fc6d570/content
2. Khan Academy. Properties of Matrix Multiplication. URL: https://uk.khanacademy.org/math/precalculus/x9e81a4f98389efdf:matrices/x9e81a4f98389efdf: properties-of-matrix-multiplication/a/properties-of-matrix-multiplication.
3. Wolfram Research. Matrix Multiplication — Technical Background. URL: https://mathworld.wolfram.com/MatrixMultiplication.html.

4. Microsoft Docs. HLSL Shader Model 5.0 Specification. URL: https://learn.microsoft.com/en-us/windows/win32/direct3dhlsl/dx-graphics-hlsl.
5. GeeksforGeeks. What is a Mesh Network? URL: https://www.geeksforgeeks.org/computer-networks/what-is-mesh-network/.
6. Coursera. What is a Mesh Network? URL: https://www.coursera.org/articles/what-is-a-mesh-network.
7. Stephen Cleary. Concurrency in C# Cookbook: Asynchronous, Parallel, and Multithreaded Programming. 2nd Edition. URL:https://dokumen.pub/concurrency-in-c-cookbook-asynchronous-parallel-and-multithreaded-programming-2nbsped-149205450x-9781492054504.html.

Мельник Тетяна Сергіївна – студент кафедри комп`ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м. Вінниця, e-mail: megatetyanamel@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.

Melnyk Tetiana Serhiivna – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e- mail: megatetyanamel@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