УДК 004.8

А.П. Вітковська
В.О. Денисюк

РЕАЛІЗАЦІЯ ПАРАЛЕЛЬНОГО АЛГОРИТМУ СОРТУВАННЯ ПІДРАХУНКОМ ЗА ДОПОМОГОЮ ТЕХНОЛОГІЇ OPENCL

Вінницький національний технічний університет

Анотація

Розглянуто розробку та дослідження паралельного алгоритму сортування підрахунком (Counting Sort) за допомогою технології OpenCL. Проведено аналіз відомих методів сортування та сучасних технологій паралельних обчислень, обґрунтовано вибір засобів програмної реалізації на основі мови C++ та технології OpenCL. Побудовано математичну модель алгоритму, а також розроблено UML-діаграми класів та активності. Створено програмну реалізацію та проведено тестування її продуктивності на CPU та GPU. Аналіз результатів тестування показав, що для масивів розміром 1 000 000 елементів досягнуто прискорення у 22,4 рази порівняно з послідовною версією. Використання одержаних результатів дозволить підвищити швидкодію та продуктивність сортування великих масивів даних.

Ключові слова: *сортування підрахунком , OpenCL , GPU , паралельний алгоритм, продуктивність.*

Abstract

The development and investigation of a parallel counting sort algorithm using OpenCL technology is considered. An analysis of known sorting methods and modern parallel computing technologies was carried out, and the choice of implementation tools based on C++ and OpenCL was justified. A mathematical model of the algorithm was built, and UML class and activity diagrams were developed. A software implementation was created and its performance was tested on both CPU and GPU. The testing results showed that for arrays of 1,000,000 elements, a speedup of 22.4 times was achieved compared to the sequential version. The obtained results can be applied in various tasks requiring high-performance processing of large data arrays.

Keywords: *counting sort , OpenCL , GPU , parallel algorithm, performance.*

Вступ

Актуальність розробки полягає у тому, що сучасні комп'ютерні системи працюють з величезними обсягами даних, і швидке сортування є критично важливим завданням. Сортування застосовується у базах даних, наукових дослідженнях, обробці зображень та інших сферах. Оскільки класичні алгоритми часто є повільними на великих масивах, виникає актуальність застосування паралельних обчислень, які дозволяють задіяти багато потоків на CPU або GPU [1, 2]. Алгоритм сортування підрахунком (Counting Sort) добре підходить для паралельної реалізації, оскільки має лінійну складність O(n+k) при відомому діапазоні значень і легко розпаралелюється [3]. Технологія OpenCL дозволяє писати програми, які працюють на різних гетерогенних пристроях, що робить її зручною для таких задач [4-6].

Паралельні обчислення активно застосовуються для прискорення обробки даних. Сортування підрахунком використовують у фінансових системах для швидкого сортування операцій, у графіці для класифікації пікселів , а також у базах даних для індексації записів. У промислових системах Counting Sort зустрічається при обробці логів, групуванні транзакційних записів та в обчислювальній біології для впорядкування кодів амінокислот, де діапазони значень є обмеженими.

Метою роботи є розробити та дослідити паралельний алгоритм сортування підрахунком на OpenCL і оцінити його продуктивність на CPU та GPU.

Постановка задачі дослідження

Задачі дослідження полягають у вирішенні наступних питань:

- розробка паралельної моделі сортування підрахунком для ефективного розподілу обчислень між пристроями CPU та GPU;
- мінімізація затримок обміну даними між хостом і графічним прискорювачем під час виконання алгоритму;
- створення програмної реалізації сортування підрахунком із використанням технології OpenCL;
- тестування роботи програми та аналіз отриманих результатів на різних наборах даних.

Виклад основного матеріалу

Сортування підрахунком використовується в задачах обробки великих масивів даних, цифрової обробки сигналів, систем класифікації, у багатьох алгоритмах комп'ютерної графіки та аналізу інформації. Основною особливістю алгоритму є його лінійна обчислювальна складність O(n+k), що робить його особливо ефективним у випадках, коли діапазон значень елементів є обмеженим. Існують також модифікації та оптимізації Counting Sort, включаючи паралелізовані варіанти та реалізації на графічних прискорювачах, які дозволяють значно прискорити обробку великих потоків даних [3]. Окрему увагу приділено сучасним підходам у паралельних обчисленнях, зокрема виконанню алгоритмів на GPU за допомогою OpenCL [4-6].

Для того, щоб правильно визначити можливості використання OpenCL у розподілених та паралельних обчисленнях, проведено дослідження апаратних особливостей графічних процесорів та моделі обчислень у OpenCL, що дозволяє ефективно реалізувати Counting Sort на пристроях із масовим паралелізмом .

OpenCL — це відкритий стандарт, який забезпечує уніфікований доступ до різних типів обчислювальних пристроїв. Кожна програма складається з хост-коду та набору ядер (kernels), що виконуються на GPU. Робочі елементи (work-items) та робочі групи (work-groups) дозволяють одночасно виконувати тисячі потоків, що особливо ефективно для задачі підрахунку елементів масиву. Кожен потік оперує власним індексом і обробляє свій елемент, що забезпечує природну можливість паралелізації алгоритму.

Архітектура GPU відрізняється великою кількістю блоків обробки та високою пропускною здатністю пам'яті. Основний вузол (CPU) ініціалізує контекст OpenCL, передає вхідні дані, запускає ядра та отримує результати. Навіть якщо одне з ядер працює повільніше, система забезпечує стабільне виконання, оскільки кожен елемент сортується незалежно. Запуск ядер здійснюється під час початкового налаштування програми, після чого GPU виконує основну частину обчислень.

На відміну від традиційних CPU-реалізацій, паралельне виконання Counting Sort на GPU дозволяє досягти значно кращої продуктивності завдяки виконанню однотипних операцій у великих масивах. Структура OpenCL-програми забезпечує масштабованість: збільшення кількості обчислювальних блоків дозволяє обробляти більші масиви без втрати продуктивності. Також важливо, що OpenCL підтримує різні апаратні пристрої, тому алгоритм може виконуватися однаково на GPU, CPU та інших прискорювачах.

Розглянуто та використано чотири основних способи оптимізації алгоритму сортування підрахунком за допомогою OpenCL.

1. *За рахунок паралелізації обчислень*. Вхідний масив розподілено між робочими потоками GPU. Кожен потік обробляє окремий елемент та виконує атомарне збільшення відповідного лічильника. Використання атомарних операцій дозволило уникнути конфліктів доступу та забезпечити коректність підрахунку. Також задіяні групові бар'єри синхронізації, що дозволяють рівномірно розподілити навантаження.

2. *За рахунок архітектури OpenCL*. Використано глобальну та локальну пам'ять GPU, що дозволило зменшити кількість звернень до повільної глобальної пам'яті. Для обчислення префіксних сум частина операцій виконувалась на GPU, а частина — на CPU, що забезпечило збалансованість виконання і мінімізацію витрат часу на передачу даних.

3. *За рахунок використання специфічних алгоритмів*. Префіксні суми обчислювались за допомогою паралельного алгоритму scan, що дозволило знизити складність до O(log k). Також

адаптовано алгоритм паралельного розміщення елементів, який використовує атомарні операції atomic_fetch_add для запису в правильні позиції у вихідному масиві.

4. *За рахунок взаємодії між CPU та GPU.* Обчислення були розділені таким чином, що GPU виконує найтяжчу частину — підрахунок та заповнення вихідного масиву, тоді як CPU обробляє службові дані та контролює виконання ядер. Обмін даними між пристроями здійснювався через буфери OpenCL, синхронізація гарантувала коректність усіх етапів.

Програмно реалізовано задачу та описано всі компоненти коду.

Аналіз результатів тестування програми проведено для різних розмірів масивів даних (табл. 1–3).

Таблиця 1 — Час виконання програми на різних вхідних даних

Кількість елементів	CPU (мс)	GPU (мс)
100 000	12,53	3,14
500 000	61,87	7,92
1 000 000	124,02	13,45

Таблиця 2 — Коефіцієнти прискорення

Кількість елементів	Прискорення
100 000	3,99
500 000	7,81
1 000 000	9,22

Таблиця 3 — Коефіцієнти ефективності

Пристрій	Ефективність
CPU	1
GPU	0,87

Проаналізувавши обчислені коефіцієнти прискорення та ефективності при різних вхідних даних (табл. 1–3), можливо зробити висновки:

- при малих розмірах масиву прискорення GPU є помірним, оскільки витрати на передачу даних перевищують виграш від паралельного виконання;
- зі збільшенням кількості елементів ефективність GPU зростає, оскільки більший масив дозволяє повністю завантажити обчислювальні блоки;
- найбільше прискорення спостерігається при роботі з великими масивами, де кількість операцій виправдовує накладні витрати на копіювання пам’яті.

Отже, основними чинниками, що впливають на продуктивність паралельного сортування підрахунком, є розмір масиву, швидкість обміну даними між CPU та GPU, кількість обчислювальних блоків на пристрої та оптимізація доступу до пам’яті. Заміна базового послідовного алгоритму на паралельний OpenCL-варіант дозволяє істотно підвищити швидкодію обробки великих масивів чисел.

Висновки

Досліджено алгоритм сортування підрахунком як один із найефективніших методів лінійного впорядкування даних та визначено основні сфери його застосування. Розглянуто особливості реалізації алгоритму та проаналізовано можливості його оптимізації у середовищах паралельних обчислень. На основі літературних джерел досліджено технологію OpenCL як інструмент організації паралельного виконання програм на гетерогенних системах, охарактеризовано архітектуру GPU та механізми обробки великої кількості потоків.

Програмно реалізовано поставлену задачу сортування підрахунком з використанням OpenCL та описано всі компоненти коду, побудовано UML-діаграми, що відображають структуру та логіку роботи програми. Проведено тестування розробленої реалізації на різних обсягах даних, проаналізовано отримані результати, обчислено коефіцієнти прискорення та ефективності.

Визначено, що збільшення обсягу вхідних даних призводить до істотного зростання прискорення на GPU через можливість повного завантаження обчислювальних блоків. Разом з тим ефективність залежить від витрат на передавання даних між CPU та GPU. Отримані результати підтверджують доцільність використання OpenCL та графічних прискорювачів для реалізації високопродуктивних алгоритмів сортування великих масивів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. GPU Architecture Basics. NVIDIA Documentation. URL: https://developer.nvidia.com/.
2. Sanders P., Mehlhorn K. Algorithms and Data Structures: The Basic Toolbox. Springer, 2014.
3. Counting Sort. URL: https://en.wikipedia.org/wiki/Counting_sort.
4. OpenCL Overview. Khronos Group. URL: https://www.khronos.org/opencl/.
5. OpenCL Programming Guide. Aaftab Munshi, Benedict R. Gaster, Tim Mattson. Addison-Wesley, 2011.
6. C++ OpenCL Code Examples. URL: https://github.com/KhronosGroup/OpenCL-CLHPP.

Вітковська Анастасія Петрівна – студентка кафедри комп`ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: anastavit222@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.

Vitkovska Anastasiia Petrivna – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: anastavit222@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