УДК 004.8

Д.О. Загатіна
В.О. Денисюк

ДОСЛІДЖЕННЯ ПАРАЛЕЛЬНОГО ПРОГРАМУВАННЯ З ВИКОРИСТАННЯМ ТЕХНОЛОГІЇ OPENMP

Вінницький національний технічний університет

Анотація

У роботі розглянуто принципи паралельного програмування з використанням технології OpenMP. Проаналізовано особливості багатопотокових обчислень на багатоядерних процесорах, розглянуто основні директиви OpenMP та їх застосування для паралелізації обчислювальних алгоритмів. Реалізовано програмний модуль для паралельної обробки даних та проведено експериментальне дослідження продуктивності для різної кількості потоків і розмірів вхідних даних. Отримані результати дозволяють оцінити ефективність використання OpenMP у задачах паралельних обчислень..

Ключові слова: *паралельне програмування, OpenMP, багатопотоковість, продуктивність, масштабованість.*

Abstract

The paper considers the principles of parallel programming using OpenMP technology. The features of multithreaded computing on multicore processors are analyzed, and the main OpenMP directives and their application for parallelizing computational algorithms are considered. A software module for parallel data processing is implemented, and an experimental study of performance is conducted for different numbers of threads and input data sizes. The obtained results allow evaluating the efficiency of using OpenMP in parallel computing tasks.

Keywords: *parallel programming, OpenMP, multithreading, performance, scalability.*

Вступ

Сучасні обчислювальні системи здебільшого базуються на багатоядерних процесорах, що робить паралельне програмування важливою складовою підвищення продуктивності програмного забезпечення. Послідовні алгоритми не дозволяють повною мірою використовувати обчислювальні ресурси сучасних процесорів, що призводить до неефективного використання апаратних можливостей [1, 2].

OpenMP (Open Multi-Processing) є стандартом для створення паралельних програм зі спільною пам’яттю та широко використовується для реалізації багатопотокових обчислень [3]. Основною перевагою OpenMP є простота використання, яка досягається завдяки директивам компілятора, що дозволяють паралелізувати програму без значних змін у структурі коду.

Метою роботи є дослідження можливостей паралельного програмування з використанням технології OpenMP та аналіз ефективності її застосування.

Постановка задачі дослідження

Для досягнення поставленої мети необхідно виконати такі завдання: дослідити основні принципи паралельного програмування; розглянути архітектуру OpenMP та модель спільної пам’яті; реалізувати послідовну та паралельну версії алгоритму з використанням технології OpenMP; провести тестування продуктивності для різної кількості потоків; проаналізувати коефіцієнти прискорення та ефективності.

Виклад основного матеріалу

Реалізовано оптимізований паралельний метод розв’язання заповненої системи лінійних рівнянь, який виконує прямий та зворотний хід методу Гаусса з частковим розпаралелюванням. Розпаралелювання застосовано лише до тих частин методу Гаусса, де це безпечно: обробка рядків під

опорним елементом; оновлення коефіцієнтів матриці. Паралельне виконання дозволяє скоротити загальний час виконання алгоритму, особливо для матриць великої розмірності, де кількість операцій зростає квадратично. Отже, алгоритм є оптимізованим у частині прямого ходу, де найбільше незалежних операцій, що дає можливість отримати суттєве прискорення при роботі з великими заповненими системами [4-6].

Технологія OpenMP базується на моделі спільної пам'яті, де всі потоки мають доступ до однієї адресної області. Паралелізація в OpenMP здійснюється за допомогою директив, функцій бібліотеки та змінних середовища. Найчастіше використовується директива parallel for, яка дозволяє розподіляти ітерації циклу між кількома потоками.

У рамках дослідження було реалізовано програму для виконання обчислювального алгоритму з використанням OpenMP. Кількість потоків задавалась за допомогою змінної середовища або функції omp_set_num_threads(). Дослідження проводилися для різних розмірів вхідних даних та різної кількості потоків з метою оцінки впливу паралелізації на час виконання програми [3, 7, 8].

Для оцінки масштабованості та прискорення алгоритму було проведено тестування на випадкових щільних системах розміром N = {50, 100, 200, 300, 500, 1000}. Коефіцієнти матриці генерувалися випадковим чином, після чого головна діагональ модифікувалася таким чином, щоб матриця була діагонально домінуючою. Це дозволяє уникнути вироджених випадків і забезпечити збіжність методу Гаусса. Для кожного розміру системи вимірювався час роботи послідовного та паралельного алгоритмів, після чого обчислювалося прискорення як відношення часу послідовного методу до часу паралельного методу (табл.1, 2).

Таблиця 1 – Час виконання програми (с)

Розмір вхідних даних, n	1 потік	2 потоки	4 потоки	8 потоків
50	0.14	0.09	0.06	0.05
200	0.87	0.49	0.31	0.27
1 000	3.10	2.18	1.36	0.58

Таблиця 2 – Коефіцієнт прискорення (n=1000)

Кількість потоків	Прискорення
2	1.42
4	2.28
8	5.34

Результати вимірювань часу виконання та отриманого прискорення для N = {500, 1000, 2000, 3000, 5000} наведено в таблиці 3.

Таблиця 3 – Результати тестування (8 потоків)

Розмір вхідних даних, n	Послідовний метод, с	Паралельний метод, с	Прискорення
500	0.36206	0.11446	3.16
1000	3.09656	0.58336	5.31
2 000	21.94916	3.05084	7.19
3 000	75.44329	9.83279	7.67
5 000	351.15733	44.81691	7.84

Як видно з наведених у таблиці результатів, зі зростанням розмірності системи лінійних рівнянь спостерігається зміна співвідношення часів виконання послідовного та паралельного варіантів методу Гаусса. Для менших розмірів системи накладні витрати, пов'язані з організацією паралельних

обчислень і синхронізацією потоків, частково нівелюють виграш від розпаралелювання, унаслідок чого ефективність паралельної реалізації є обмеженою. Однак із подальшим збільшенням розмірності задачі час роботи паралельного алгоритму зростає значно повільніше, ніж час виконання послідовного варіанта. Це призводить до поступового зростання прискорення та свідчить про ефективніше використання обчислювальних ресурсів багатоядерної обчислювальної системи. Отримані залежності підтверджують, що паралельний підхід є доцільним саме для великих заповнених систем лінійних рівнянь, де обсяг обчислень є достатньо значним. Для всіх досліджених розмірів систем максимальна різниця між розв'язками, отриманими послідовним і паралельним алгоритмами, перебуває в межах машинної точності та фактично дорівнює нулю. Це підтверджує числову еквівалентність двох реалізацій і правильність використаної схеми паралелізації. Таким чином, результати стрес-тестування демонструють коректність, стійкість і ефективність розробленого паралельного методу та підтверджують доцільність його застосування для розв'язування заповнених систем лінійних рівнянь великої розмірності.

Висновки

У роботі досліджено принципи паралельного програмування з використанням технології OpenMP. Реалізовано паралельний алгоритм розв'язання заповненої системи лінійних рівнянь методом Гаусса для різних значень кількості рівнянь у системі. Проведено експериментальне дослідження його продуктивності.

Встановлено, що OpenMP є ефективним засобом для паралелізації обчислювальних задач на багатоядерних процесорах, особливо для задач з великим обсягом даних.

Отримані результати свідчать, що використання OpenMP дозволяє значно скоротити час виконання програми. Водночас зі збільшенням кількості потоків спостерігається зниження швидкості зростання коефіцієнта ефективності при його загальному зростанню, що пов'язано з накладними витратами на керування потоками та синхронізацію.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Грищенко В. М. Паралельні обчислення в сучасних системах моделювання. Вінниця: ВНТУ, 2021. 310 с.
2. Семеренко, В. П. Технології паралельних обчислень : навч. посіб. Вінниця: ВНТУ, 2018. – 104 с. URL: https://www.researchgate.net/publication/334710599_V_P_Semerenko_TEHNOLOGII_PARALELNIH_OBCISLEN_Ministerstvo_osviti_i_nauki_Ukraini_Vinnickij_nacionalnij_tehnicnij_universitet /
3. The OpenMP API specification for parallel programming. URL: https://www.openmp.org /
4. Кривецький О. В. Паралельні алгоритми для нелінійних задач. Вінниця: ВНТУ, 2018. 280 с.
5. Методи оптимізації та паралельного розв'язання нелінійних рівнянь. Комп'ютерні науки. 2020. Вип. 3(21). С. 87-95.
6. Гладкий М. І. Алгоритми розв'язання нелінійних систем рівнянь з використанням паралельного програмування. Київ: Технічна література, 2019. 220 с.
7. Parallel Scaling Guide. URL: https://rcdocs.mines.edu/pages/user_guides/Parallel_Scaling_Guide.html?utm_source=chatgpt.com.
8. Asad A., Hamza A. The C# Programmer's Study Guide (MCSD). Apress, 2017. 475 с. URL: https://www.pdfdrive.com/the-c-programmers-study-guide-mcsd-exam-70-483-e181118218.html .

Загатіна Дар'я Олександрівна – студентка кафедри комп`ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: dzahatina@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 .

Zahatina Daria Oleksandrivna – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: dzahatina@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 .