УДК 004.8

М.С. Рудковський
В.О. Денисюк

РЕАЛІЗАЦІЯ ПАРАЛЕЛЬНОГО АЛГОРИТМУ СОРТУВАННЯ INSERTION SORT

Вінницький національний технічний університет

Анотація

У роботі досліджено алгоритми сортування та можливості їх паралельної реалізації з використанням технології OpenMP. Проведено теоретичний аналіз класичних алгоритмів сортування, зокрема методу вставки (Insertion Sort), а також принципів паралельних обчислень у моделях спільної пам'яті. Описано особливості алгоритму Insertion Sort, його складність, поведінку на різних типах вхідних даних та причини обмежених можливостей прямої паралелізації. На основі проведеного аналізу реалізовано послідовну та паралельну модифікації алгоритму Insertion Sort. Паралельна версія ґрунтується на блочному розбитті масиву та подальшій незалежній обробці підмасивів із використанням директив OpenMP. У ході експериментальної частини виконано тестування коректності сортування та проведено вимірювання продуктивності при різній кількості потоків. Результати показали, що паралельна обробка забезпечує прискорення під час роботи з великими масивами, проте ефективність паралельного алгоритму суттєво залежить від структури вхідних даних та способу розбиття масиву. Отримані дані підтверджують можливість прискорення класичного алгоритму Insert Sort за рахунок паралельної обробки блоків, а також демонструють обмеження та потенційні напрями оптимізації паралельних алгоритмів сортування.

Ключові слова: *сортування вставками, паралельне сортування, C++, OpenMP, багатопоточність, паралельні алгоритми.*

Abstract

The work investigates sorting algorithms and the possibilities of their parallel implementation using OpenMP technology. A theoretical analysis of classical sorting algorithms, in particular the Insertion Sort method, as well as the principles of parallel computing in shared memory models, is carried out. The features of the Insertion Sort algorithm, its complexity, behavior on different types of input data and the reasons for the limited possibilities of direct parallelization are described. Based on the analysis, a sequential and parallel modification of the Insertion Sort algorithm is implemented. The parallel version is based on block partitioning of the array and subsequent independent processing of subarrays using OpenMP directives. During the experimental part, sorting correctness testing was performed and performance measurements were carried out with different numbers of threads. The results showed that parallel processing provides acceleration when working with large arrays, but the efficiency of the parallel algorithm significantly depends on the structure of the input data and the method of partitioning the array. The obtained data confirm the possibility of accelerating the classical Insert Sort algorithm by parallel processing of blocks, and also demonstrate the limitations and potential directions for optimizing parallel sorting algorithms.

Keywords: *insertion sort, parallel sort, C++, OpenMP, multithreading, parallel algorithms.*

Вступ

В сучасних інформаційних системах обробка великих обсягів даних є одним із ключових завдань, що впливають на продуктивність програмних комплексів. Однією з найпоширеніших операцій при роботі з даними є сортування, яке використовується у пошукових системах, базах даних, мультимедіа, наукових розрахунках та інших сферах. Від ефективності алгоритму сортування залежить загальна швидкодія програмних рішень, особливо коли йдеться про масиви даних значного розміру.

Алгоритм сортування вставками (Insertion Sort) є одним із базових та найбільш зрозумілих алгоритмів сортування [1-5]. Його перевагами є простота реалізації, стабільність сортування та ефективність у випадку частково впорядкованих даних. Проте для великих наборів даних алгоритм має квадратичну складність $O(n^2)$, що істотно обмежує його застосування у послідовному вигляді.

З появою багатоядерних процесорів та розвитку технологій паралельного програмування виникла можливість прискорювати класичні алгоритми за рахунок розподілу обчислень між кількома потоками. Паралельні обчислення дозволяють суттєво зменшити час виконання завдань, що містять незалежні або частково незалежні операції. Використання інструментарію OpenMP забезпечує зручний спосіб створення паралельних програм, дозволяючи інтегрувати багатопотоковість у традиційні алгоритми з мінімальною зміною їх структури.

Актуальність дослідження полягає у можливості підвищення продуктивності алгоритму Insertion Sort за рахунок побудови його паралельної модифікації та аналізу впливу кількості потоків на ефективність сортування [6, 7]. Попри те, що класичний Insertion Sort містить сильні залежності між ітераціями, існують підходи, які дозволяють розподілити частину обчислень між потоками, зокрема завдяки блочному розбиттю даних та організації частково незалежних процесів вставки.

Постановка задачі дослідження

Задачі дослідження полягають у вирішенні наступних питань:

- провести аналіз алгоритмів сортування та особливостей класичного Insertion Sort;
- дослідити принципи організації паралельних обчислень і технологію OpenMP;
- розробити послідовну реалізацію алгоритму;
- побудувати паралельну модифікацію Insertion Sort;
- здійснити тестування та порівняння результатів для різної кількості потоків;
- визначити ефективність, недоліки та можливі напрямки оптимізації паралельного алгоритму.

Виклад основного матеріалу

Алгоритми сортування є одними з ключових процедур у комп’ютерній науці, оскільки вони широко застосовуються у базах даних, інформаційних системах, програмному моделюванні, пошукових системах та задачах обробки великих масивів даних. Серед класичних методів сортування виділяють алгоритми, що базуються на порівняннях, такі як Quick Sort, Merge Sort та Heap Sort, для яких характерна середня часова складність порядку $O(n \log n)$, а також прості алгоритми зі складністю $O(n^2)$, зокрема сортування вставками (Insertion Sort), яке є ефективним для малих або частково впорядкованих масивів даних.

На основі аналізу літературних джерел було досліджено особливості алгоритму Insertion Sort, зокрема принцип поступового формування впорядкованої частини масиву шляхом вставлення елементів у відповідні позиції, стабільність сортування та залежність продуктивності від початкового стану вхідних даних. Окрему увагу приділено питанням паралельної реалізації алгоритмів сортування та оцінці їх часової складності в умовах багатопотокового виконання.

У роботі розглянуто підхід до часткової паралельної реалізації алгоритму сортування вставками з використанням технології OpenMP у середовищі спільної пам’яті [8-11]. Основною проблемою прямої паралелізації Insertion Sort є наявність сильних послідовних залежностей між ітераціями алгоритму, що унеможливлює коректне застосування директиви parallel for до основного циклу без порушення логіки сортування.

Для подолання цього обмеження застосовано метод блочного розбиття масиву, при якому вхідні дані поділяються на кілька підмасивів. Кожен підмасив сортується незалежно в окремому потоці за допомогою алгоритму Insertion Sort. Такий підхід дозволяє задіяти паралельні обчислювальні ресурси процесора на етапі локального сортування та зменшити загальний час виконання для великих масивів даних.

Паралельну частину алгоритму реалізовано мовою програмування C++ із використанням директив OpenMP. Кожен потік працює з окремою ділянкою пам’яті, що виключає конфлікти доступу до даних і не потребує складних механізмів синхронізації. Після завершення сортування підмасивів виконується етап злиття відсортованих блоків у єдиний впорядкований масив, який реалізується послідовно.

Для оцінки продуктивності паралельної реалізації алгоритму сортування вставками було проведено серію експериментів із масивами випадкових чисел різного розміру. У всіх тестах використовувалась фіксована кількість потоків, рівна чотирьом, а значення елементів масиву належали відповідному діапазону. Вимірювався час виконання послідовної та паралельної версій алгоритму, а також обчислювався коефіцієнт прискорення.

Узагальнені результати вимірювань подано у таблиці 1.

Таблиця 1 – Час роботи паралельного та послідовного сортувань при різних обсягах даних

Розмір масиву	Потоки	Діапазон значень	Послідовне сортування, с	Паралельне сортування, с	Прискорення
10 000	4	[1; 10 000]	0.2419	0.021482	11.2602
100 000	4	[1; 100 000]	26.5704	2.590560	10.2566
1 000 000	4	[1; 1 000 000]	2 914.7100	261.703000	11.1375

Аналіз отриманих результатів показує, що паралельна реалізація алгоритму Insertion Sort забезпечує суттєве скорочення часу виконання порівняно з послідовною версією для всіх досліджених розмірів масивів. Коефіцієнт прискорення у всіх експериментах перевищує значення 10, що свідчить про ефективне використання багатопотокових обчислювальних ресурсів.

Зі збільшенням розміру масиву спостерігається зростання абсолютного часу виконання алгоритму, що відповідає квадратичній часовій складності Insertion Sort. Водночас відносне прискорення залишається стабільним, що пояснюється рівномірним розподілом обчислювального навантаження між потоками та достатньо великим обсягом роботи для кожного з них. Накладні витрати на створення потоків і синхронізацію в даному випадку компенсуються значним обсягом обчислень.

Отримані експериментальні дані підтверджують, що часткова паралелізація алгоритму сортування вставками з використанням технології OpenMP є доцільною при роботі з великими масивами даних. Разом з тим результати демонструють обмеження такого підходу, зумовлені послідовною природою алгоритму та необхідністю виконання етапів, які не піддаються паралельному виконанню.

Висновки

Досліджено алгоритм сортування вставками (Insertion Sort) як один із базових порівняльних методів упорядкування даних, розглянуто його місце серед інших алгоритмів сортування, зокрема методів зі складністю O(n log n), та проаналізовано теоретичні оцінки часової складності й області практичного застосування. На основі аналізу літературних джерел опрацьовано питання побудови паралельних алгоритмів сортування, а також використання технологій багатопотокового програмування у середовищі спільної пам’яті. У роботі розроблено та програмно реалізовано послідовну і паралельну версії алгоритму сортування вставками з використанням технології OpenMP. Запропонований підхід до часткової паралелізації ґрунтується на блочному розбитті вхідного масиву та незалежному сортуванні підмасивів у декількох потоках. Описано загальну структуру програмного модуля, принципи організації паралельних обчислень і основні елементи реалізації мовою програмування C++.

Проведено експериментальне тестування розробленого алгоритму на масивах різного розміру (10 000, 100 000 та 1 000 000 елементів). Результати експериментів показали, що паралельна реалізація забезпечує коректне сортування вхідних даних і суттєве скорочення часу виконання порівняно з послідовною версією. Встановлено, що зі збільшенням обсягу вхідних даних ефективність паралельного виконання зростає, оскільки накладні витрати на створення та синхронізацію потоків компенсуються значним обсягом обчислень. Отже, реалізований підхід до паралельної реалізації алгоритму Insertion Sort із використанням OpenMP може бути рекомендований для застосування в програмних системах, де необхідне впорядкування великих масивів даних у середовищі багатоядерних процесорів, а також може слугувати основою для подальших досліджень і побудови гібридних паралельних алгоритмів сортування.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Павлов О. П., Костенко О. В. Алгоритми та структури даних. – К.: КПІ ім. Ігоря Сікорського, 2018.
2. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C. Introduction to Algorithms. MIT Press, 2022.
3. Knuth D. E. The Art of Computer Programming. Volume 3: Sorting and Searching. Addison-Wesley, 2011.
4. Sedgewick R., Wayne K. Algorithms. Addison-Wesley Professional, 2011.
5. GeeksforGeeks. Insertion Sort. URL: https://www.geeksforgeeks.org/insertion-sort/
6. Жуков І. А., Поліщук В. В. Паралельні та розподілені обчислення. К.: Видавництво НТУУ «КПІ», 2016.
7. Коваленко О. М. Основи паралельного програмування. Харків: ХНУРЕ, 2017.
8. Chapman B., Jost G., van der Pas R. Using OpenMP: Portable Shared Memory Parallel Programming. MIT Press, 2008.
9. Pacheco P. S. An Introduction to Parallel Programming. – Morgan Kaufmann, 2011.
10. Stroustrup B. The C++ Programming Language. Addison-Wesley, 2013.
11. OpenMP Architecture Review Board. OpenMP Application Programming Interface Specification [Електронний ресурс]. – Режим доступу: https://www.openmp.org

Рудковський Михайло Сергійович – студент кафедри комп`ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: misharudkovskiy1@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.

Rudkovskiy Mykhailo Serhiyovych – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: misharudkovskiy1@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