УДК 004.8

Д.В. Коломійчук
В.О. Денисюк

ПАРАЛЕЛЬНА РЕАЛІЗАЦІЯ АЛГОРИТМУ СОРТУВАННЯ TREE SORT

Вінницький національний технічний університет

Анотація

Досліджено можливості паралелізації алгоритму Tree Sort для підвищення ефективності сортування великих масивів даних. Основна мета полягала у розробці оптимізованої версії алгоритму, здатної використовувати ресурси багатоядерних процесорів. Паралельна реалізація алгоритму Tree Sort передбачає розподіл вхідних даних між потоками, синхронізоване додавання елементів у дерево та одночасний обхід піддерев. Для реалізації використано мову C# та платформу .NET Framework, що забезпечило інтеграцію з механізмами багатопотоковості Task Parallel Library та ефективне керування пам'яттю.

Ключові слова: *паралельний алгоритм, сортування Tree Sort, C#, оптимізація, багатопотоковість.*

Abstract

The possibilities of parallelization of the Tree Sort algorithm to increase the efficiency of sorting large data sets were investigated. The main goal was to develop an optimized version of the algorithm capable of using the resources of multi-core processors. The parallel implementation of the Tree Sort algorithm involves the distribution of input data between threads, synchronized addition of elements to the tree, and simultaneous traversal of subtrees. The C# language and the .NET Framework platform were used for the implementation, which provided integration with the Task Parallel Library multithreading mechanisms and effective memory management.

Keywords: *parallel algorithm, Tree Sort sorting, C#, optimization, multithreading.*

Вступ

Необхідність розробки паралельних алгоритмів сортування зумовлена зростанням обсягів даних у науці, техніці та фінансах. Класичні методи, такі як Tree Sort, мають обмеження через послідовний характер роботи. Дослідження та реалізація паралельної версії Tree Sort для підвищення швидкості обробки даних є актуальною задачею.

Постановка задачі дослідження

Задачі дослідження включають розгляд паралельних алгоритмів сортування, розробку оптимізованого паралельного алгоритму Tree Sort, його математичне моделювання, розробку та тестування програмного модуля. Паралельна версії Tree Sort повинна задовільняти вимогам:

- рівномірний розподіл навантаження між потоками;
- мінімізація затримок синхронізації;
- ефективне використання ресурсів багатоядерних систем;
- стабільна робота на великих масивах даних.

Виклад основного матеріалу

Алгоритм Tree Sort базується на побудові бінарного дерева пошуку з подальшим обходом у порядку зростання [1, 2]. Паралельна реалізація передбачала розподіл вхідних даних між потоками, синхронізоване додавання елементів у дерево та одночасний обхід піддерев. Алгоритм сортування Tree Sort обрано через можливість паралелізації побудови дерева та обходу піддерев. Складність алгоритму становить $O(n \log n)$ у середньому [3, 4].

Час виконання паралельної версії алгоритму Tree Sort описується формулою [5, 6]:

$$T(p) = \frac{T(1)}{p} + T_{sync}, \quad (1)$$

де $T(1)$ – час виконання послідовного алгоритму Tree Sort;

T_{sync} – час, необхідний для синхронізації між потоками;

p – кількість потоків.

Структурно алгоритм містить етап розподілу даних, етап паралельної вставки елементів у дерево та етап обходу піддерев.

Програмна реалізація алгоритму використовує класи ‘BinaryTree’, ‘TreeNode’, ‘DataPartitioner’ та багатопотоковість C#. Синхронізація реалізована через механізм ‘lock’ [7].

Результати дослідження.

На етапі проектування проведено аналіз існуючих алгоритмів сортування, математичне моделювання паралельного Tree Sort із врахуванням часу синхронізації (T_{sync}), а також синтез потокового графу для візуалізації етапів розподілу даних і обчислень. Програмний модуль включає класи для роботи з деревом (BinaryTree, TreeNode), розподілу даних (DataPartitioner) та керування потоками. Для уникнення конфліктів при доступі до дерева застосовано механізм блокування (lock).

Тестування проводилося на масиві з 100 елементів, що підтвердило коректність сортування та зменшення часу виконання на 30% порівняно з послідовною версією. Аналіз продуктивності показав, що використання 4-х потоків дозволяє ефективно розподілити навантаження між ядрами CPU. Отримані результати вказують на перспективність застосування паралельних підходів для Tree Sort, особливо при роботі з даними великого обсягу. З’ясовано обмеження алгоритму, пов’язані зі складністю балансування дерева в умовах багатопотоковості, та пропонується впровадження lock-free структур.

Тестування розробленого паралельного алгоритму Tree Sort підтвердило:

- коректність сортування - правильний порядок елементів у відсортованому масиві;
- продуктивність - використання 4-х потоків скоротило час виконання порівняно з послідовною реалізацією;
- масштабованість - алгоритм ефективно використовує доступні ядра процесора на різних апаратних конфігураціях.

Висновки

Паралельна реалізація Tree Sort дозволяє значно прискорити сортування великих масивів.

Використання механізмів синхронізації C# забезпечує потокобезпеку роботи зі структурами даних.

Оптимізація розподілу навантаження між потоками є ключовим фактором підвищення ефективності.

Подальші дослідження можуть бути спрямовані на впровадження lock-free структур аби зменшити час синхронізації між потоками (T_{sync}).

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Tree Sort. URL: https://www.geeksforgeeks.org/tree-sort .
2. Дорошенко А.Ю. Паралельні обчислювальні системи: конспект лекцій. К.: Видавничий дім КМ Академія, 2013. 146 с.
3. Методичні вказівки до виконання лабораторних робіт з курсу «Технології розподілених систем і паралельних обчислень» для студентів спеціальності 122 – Комп’ютерні науки/ Уклад. А. А. Яровий, С. В. Барабан, В. С. Озеранський, Є. О. Шемет. Вінниця : ВНТУ, 2019. 56 с.
4. Bentaleb, A.; Yifan, L.; Xin, J. (2016). Parallel and Distributed Algorithms. URL: https://www.comp.nus.edu.sg/~rahul/allfiles/cs6234-16-pds.pdf
5. Samuel Larsen and Saman Amarasinghe. Exploiting Superword Level Parallelism with Multimedia Instruction Sets. URL: http://groups.csail.mit.edu/cag/slp/SLP-PLDI2000.pdf .
6. Using threads and threading. URL: https://learn.microsoft.com/en-us/dotnet/standard/threading/using-threads-and-threading .
7. Troelsen A., Japikse P. Pro C# 7: With .NET and .NET Core. APress. Inc., 2017. 1372 с. URL: https://www.pdfdrive.com/pro-c-7-with-net-and-net-cored183552783.html .

Коломійчук Денис Вадимович – студент кафедри комп`ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: denyskolomiichuk@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.

Kolomiichuk Denys Vadymovych – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: denyskolomiichuk@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