УДК 004.8

М.В. Білаш
В.О. Денисюк

ДОСЛІДЖЕННЯ ТА РЕАЛІЗАЦІЯ ПАРАЛЕЛЬНОГО АЛГОРИТМУ ПОШУКУ В ГЛИБИНУ ДЛЯ БАГАТОЯДЕРНИХ СИСТЕМ

Вінницький національний технічний університет

Анотація

У роботі розглянуто реалізацію та аналіз паралельного алгоритму пошуку в глибину (Depth-First Search, DFS) для обробки графових структур. Проаналізовано особливості класичного послідовного DFS та визначено основні труднощі його розпаралелювання в багатопотоковому середовищі. Обґрунтовано вибір списку суміжності як основної структури даних для подання графів. Розроблено програмну реалізацію паралельного алгоритму DFS мовою Python із використанням бібліотеки threading. Запропоновано модель розпаралелювання на основі розподілу піддерев графа між потоками з урахуванням синхронізації доступу до спільних даних. Проведено експериментальне дослідження ефективності алгоритму для графів різної розмірності та різної кількості потоків. Отримані результати показують, що паралельна реалізація DFS має обмежену масштабованість, зумовлену послідовною природою алгоритму та накладними витратами на синхронізацію. Разом з тим, запропонований підхід може бути використаний для аналізу ефективності паралельних алгоритмів обходу графів у багатопотокових програмних системах.

Ключові слова: *паралельний алгоритм, пошук у глибину, DFS, графи, багатопотокові обчислення, паралельне програмування, Python.*

Abstract

This paper presents the implementation and analysis of a parallel Depth-First Search (DFS) algorithm for processing graph structures. The features of the classical sequential DFS are analyzed, and the main challenges of its parallelization in a multithreaded environment are identified. The adjacency list is justified as the primary data structure for graph representation.A parallel DFS algorithm is implemented in Python using the threading library. A parallelization model based on distributing graph subtrees among threads with synchronized access to shared data structures is proposed. Experimental evaluation is conducted on graphs of various sizes and different numbers of threads.The results show that the parallel DFS algorithm has limited scalability due to the inherently sequential nature of DFS and synchronization overhead. Nevertheless, the proposed approach can be applied to analyze the performance of parallel graph traversal algorithms in multithreaded software systems.

Keywords: *parallel algorithm, depth-first search, DFS, graphs, multithreading, parallel programming, Python.*

Вступ

Сучасні обчислювальні системи все частіше орієнтуються на використання багатоядерних процесорів і паралельних обчислень з метою підвищення продуктивності програмних рішень. Особливої актуальності це набуває під час обробки великих графових структур, які широко застосовуються в аналізі мереж, інформаційних системах, задачах оптимізації та моделювання.

Одним із базових алгоритмів обходу графів є алгоритм пошуку в глибину (Depth-First Search, DFS). Незважаючи на простоту та ефективність, класичний DFS має виражену послідовну природу, що ускладнює його пряме використання в багатопотокових середовищах. Це зумовлює необхідність розробки та дослідження підходів до його розпаралелювання з урахуванням синхронізації доступу до спільних даних і балансування навантаження між потоками.

У зв’язку з цим актуальним є дослідження можливостей паралельної реалізації DFS, оцінка її ефективності та аналіз обмежень масштабованості при виконанні в багатопотокових програмних системах.

Постановка задачі дослідження

Основні задачі роботи полягають у розв'язанні таких питань:

- аналіз класичного алгоритму пошуку в глибину (Depth-First Search) та можливостей його розпаралелювання;
- дослідження особливостей паралельних і багатопотокових обчислень при обході графових структур;
- обґрунтування вибору структури даних для подання графів у паралельному алгоритмі DFS;
- розробка моделі розпаралелювання алгоритму DFS з використанням багатопотокового виконання;
- реалізація паралельного алгоритму пошуку в глибину мовою програмування Python із використанням бібліотеки threading;
- проведення експериментального дослідження продуктивності алгоритму на графах різної розмірності та при різній кількості потоків;
- аналіз отриманих результатів та оцінка ефективності і масштабованості паралельної реалізації DFS.

Виклад основного матеріалу

Алгоритми обходу графів є базовими процедурами в комп'ютерних науках, оскільки широко застосовуються в аналізі мереж, інформаційних системах, задачах маршрутизації та обробки складних структур даних. Серед класичних методів особливе місце займає алгоритм пошуку в глибину (Depth-First Search, DFS), який характеризується лінійною часовою складністю $O(V + E)$ та простотою реалізації. Проте послідовна природа DFS обмежує його ефективність при використанні сучасних багатоядерних обчислювальних систем.

На основі аналізу літературних джерел досліджено особливості класичного алгоритму DFS, зокрема його стекову модель роботи, залежність порядку обходу від структури графа та проблеми, що виникають при спробі розпаралелювання. Окрему увагу приділено вибору структури даних для подання графа. У роботі обґрунтовано використання списку суміжності як найбільш придатного варіанта для роботи з великими та розрідженими графами, а також для організації паралельного доступу до даних [1-4].

У роботі реалізовано паралельний алгоритм пошуку в глибину з використанням багатопотокового підходу. Реалізація виконана мовою програмування Python із застосуванням бібліотеки threading, що дозволяє створювати та керувати потоками в межах одного процесу [4–6]. Розпаралелювання досягається шляхом розподілу піддерев графа між потоками, при цьому кожен потік виконує обхід власної частини графа з використанням локальної стекової структури. Для забезпечення коректності алгоритму реалізовано механізми синхронізації доступу до спільного масиву відвіданих вершин [5, 7, 8].

Для оцінки ефективності запропонованого підходу проведено серію експериментальних досліджень на графах різної розмірності та з різною кількістю потоків. У ході експериментів вимірювався час виконання алгоритму, а також аналізувалась залежність продуктивності від рівня паралелізму. Отримані результати подано у вигляді таблиць і графіків, що дозволяє наочно оцінити вплив багатопотокового виконання на швидкодію алгоритму DFS [7-9].

Узагальнені результати вимірювань подано в таблиці 1.

Результати демонструють, що паралельний DFS у більшості випадків працює повільніше, ніж послідовний. Це пояснюється особливостями структури DFS:

- стандартний DFS є інтенсивно рекурсивним і майже повністю послідовним за природою;
- паралелізація призводить до синхронізації потоків, які часто очікують доступу до спільних даних;
- потоки можуть дублювати роботу, перевіряючи ті самі вершини повторно, що збільшує час виконання.

Таблиця 1 – Порівняння часу виконання та прискорення

Граф	Послідовний DFS, с	Threads	Паралельний DFS, с	Speedup
SMALL (10)	0.000018	1	0.000238	0.075
SMALL (10)	0.000018	2	0.000350	0.051
SMALL (10)	0.000018	4	0.000632	0.028
SMALL (10)	0.000018	8	0.001235	0.014
MEDIUM (100)	0.000106	1	0.000408	0.259
MEDIUM (100)	0.000106	2	0.000555	0.190
MEDIUM (100)	0.000106	4	0.000856	0.123
MEDIUM (100)	0.000106	8	0.001660	0.064
LARGE (1 000)	0.001280	1	0.004180	0.306
LARGE (1 000)	0.001280	2	0.004902	0.261
LARGE (1 000)	0.001280	4	0.003793	0.337
LARGE (1 000)	0.001280	8	0.004412	0.290
XLARGE (3 000)	0.003884	1	0.009568	0.406
XLARGE (3 000)	0.003884	2	0.010697	0.363
XLARGE (3 000)	0.003884	4	0.010357	0.375
XLARGE (3 000)	0.003884	8	0.010866	0.357

Прискорення для великих графів (1000–3000 вершин) трохи краще, однак всі значення speedup залишаються нижчими за 1, що означає відсутність реального прискорення [9].

Висновки

У роботі розглянуто задачу паралельної реалізації алгоритму пошуку в глибину для обходу графових структур. Проведений аналіз показав, що класичний алгоритм DFS має виражену послідовну природу, що ускладнює його ефективне розпаралелювання в багатопотоковому середовищі [1–4].

Реалізовано паралельний алгоритм DFS мовою програмування Python із використанням бібліотеки threading. Обґрунтовано вибір списку суміжності як основної структури даних для подання графів, що забезпечує ефективну роботу алгоритму на великих та розріджених графах [2, 4]. Запропонований підхід дозволяє коректно виконувати обхід графа з використанням кількох потоків за умови синхронізації доступу до спільних даних [3-6].

Експериментальні дослідження показали, що паралельна реалізація DFS характеризується обмеженою масштабованістю, зумовленою накладними витратами на синхронізацію та залежністю між етапами обходу. Отримані результати підтверджують доцільність використання паралельного DFS для аналізу ефективності багатопотокових алгоритмів та можуть бути використані під час подальших досліджень і розробки програмних систем, що працюють з графовими структурами [7-9].

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Bader, D. A., & Madduri, K. Designing Multithreaded Algorithms for Breadth-First Search and Depth-First Search on Multicore Systems.
2. Georgia Institute of Technology, College of Computing, Technical Report, 2006. Available at: https://www.cc.gatech.edu/~bader/papers/BFS-TR.pdf
3. Ящук С. П., Грицай Я. М. Паралельні обчислення: навчальний посібник. Львів: Вид-во Львівської політехніки, 2020.
4. Васильєв А. М. Паралельні та розподілені обчислення: підручник. Київ: КНУ, 2019.
5. Tarjan R. Depth-First Search and Linear Graph Algorithms. SIAM Journal on Computing, 1972.
6. Bondy J. A., Murty U. S. R. Graph Theory. Springer, 2008.
7. Burtscher M., Pingali K. An Efficient Lock-Free Parallel Depth-First Search. Proceedings of the ACM SIGPLAN PPoPP, 2010.
8. Rauber T., Rünger G. Parallel Programming: for Multicore and Cluster Systems. 2nd ed. Springer, 2013.
9. Grama A., Gupta A., Karypis G., Kumar V. Introduction to Parallel Computing. 2nd ed. Pearson, 2003.

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

Bilash Mykhailo Vasylovych – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: rekocmoc@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