УДК 004.421.2

Б. О. Верещак
Д. В. Власенко

Застосування гібридних алгоритмів сортування для оптимізації агрегації потокових даних моніторингу

Вінницький національний технічний університет

Анотація

Дослідження присвячене аналізу ефективності алгоритмів впорядкування в контексті обробки великих масивів частково впорядкованих даних, характерних для систем логування. Розглянуто конкретний приклад агрегації журналів подій, де записи надходять із часовими мітками, що мають незначні відхилення від хронології. Встановлено, що застосування адаптивних гібридних методів дозволяє скоротити час обробки таких масивів порівняно з класичними алгоритмами швидкого сортування завдяки використанню існуючих впорядкованих підпослідовностей.

Ключові слова: Часова складність, обробка логів, гібридне сортування, стабільність алгоритму.

Abstract

The study is dedicated to analyzing the efficiency of sorting algorithms in the context of processing large arrays of partially ordered data, typical for logging systems. A specific case of event log aggregation is examined, where records arrive with timestamps showing slight deviations from chronology. It is established that the use of adaptive hybrid methods allows for a reduction in processing time compared to classical quicksort algorithms by leveraging existing ordered subsequences.

Keywords: Time complexity, log processing, hybrid sorting, algorithm stability.

Алгоритмічна база обробки даних залишається критично важливим фундаментом продуктивності сучасних інформаційних систем, особливо в контексті експоненційного зростання обсягів телеметричної інформації, що генерується мікросервісними архітектурами. У своїй фундаментальній праці «Мистецтво програмування» Дональд Кнут зазначає, що операції сортування можуть займати більше чверті машинного часу у великих обчислювальних центрах, а вибір методу впорядкування суттєво залежить від апріорних знань про структуру вхідних даних [1, С. 3]. У задачах моніторингу серверної інфраструктури, аналізу трейсів розподілених транзакцій та агрегації системних журналів ключовим етапом є хронологічне впорядкування подій. Традиційно для вирішення цього класу задач застосовують універсальні алгоритми, зокрема швидке сортування. Цей алгоритм, детально проаналізований у підручнику Кормена, у середньому випадку демонструє складність O(n lg n) і вважається стандартом для багатьох системних бібліотек завдяки ефективній роботі з кеш-пам'яттю та відсутності потреби у додатковій пам'яті для злиття [2, С. 186]. Проте класичні академічні дослідження часто фокусуються на поведінці алгоритмів при випадковому рівномірному розподілі вхідних ключів, ігноруючи реальні патерни даних.

У реальних умовах експлуатації високонавантажених розподілених систем вхідні дані рідко бувають повністю хаотичними. Специфіка збору інформації з різних вузлів мережі створює умови, за яких масиви даних є частково впорядкованими. Роберт Седжвік у своїх дослідженнях наголошує, що ігнорування існуючого порядку в даних є нераціональним використанням ресурсів, і пропонує розглядати адаптивні алгоритми, час виконання яких залежить від кількості інверсій у вхідному масиві [3, С. 251]. Це твердження стає особливо актуальним в еру хмарних обчислень, де вартість процесорного часу прямо впливає на операційні витрати бізнесу. Актуальність теми дослідження зумовлена необхідністю оптимізації процесів обробки потокових даних, де класичні методи типу «розділяй і володарюй» часто виконують надлишкову роботу. Більше того, як зазначає Макілрой у статті про вразливості швидкого сортування, певні патерни вхідних даних можуть призвести до деградації продуктивності QuickSort до квадратичної складності O(n^2), що є неприпустимим для систем реального часу [4]. У цьому дослідженні увагу зосереджено на порівняльному аналізі класичних та адаптивних гібридних

алгоритмів на прикладі конкретного виробничого кейсу – модуля агрегації логів централізованої системи моніторингу.

В якості об’єкта для практичного дослідження було обрано підсистему збору логів, яка отримує повідомлення від кількох десятків незалежних сервісів через брокер повідомлень. Кожне повідомлення містить часову мітку з точністю до мілісекунди. Архітектурна специфіка передачі даних мережею призводить до того, що глобальний масив подій формується майже у хронологічному порядку. Проте через асинхронність мережевих запитів, різні політики повторної відправки пакетів та нерівномірне навантаження на канали зв’язку, деякі пакети даних надходять із затримкою, порушуючи ідеальний порядок. Це явище, відоме як network jitter, створює ситуацію, коли 90-95% елементів масиву вже знаходяться на своїх місцях або дуже близько до них, а решта є локальними інверсіями. Для оцінки ефективності алгоритмів у такій ситуації було проведено теоретичне моделювання обробки масивів даних обсягом порядку 10 мільйонів записів, що мають розподіл інверсій, характерний для нормального мережевого джитера (коли 90-95% елементів знаходяться на своїх місцях).

Було здійснено аналітичне порівняння очікуваної продуктивності класичного алгоритму QuickSort та гібридного алгоритму Timsort, описаного Тімом Пітерсом [5]. Асимптотичний аналіз показує, що класичний алгоритм швидкого сортування при роботі з такими даними діє неоптимально, оскільки процедура рекурсивного розбиття масиву виконується у повному обсязі O(n lg n), ігноруючи наявність довгих впорядкованих підпослідовностей. Це призводить до надлишкових операцій порівняння. Натомість аналіз механіки гібридного алгоритму свідчить про принципово іншу динаміку. Алгоритм проектується таким чином, щоб на першому етапі сканувати масив та виявляти вже існуючі послідовності. Для коротких невпорядкованих ділянок використовується бінарне сортування вставками, яке, згідно з дослідженнями Седжвіка, на майже впорядкованих масивах наближається до лінійної складності O(n) [3, С. 245].

Особливу увагу в аналізі було приділено механізму злиття серій. У гібридному алгоритмі Timsort передбачено режим галопування, який активується при виявленні довгих послідовностей елементів з одного масиву, що є меншими за поточний елемент іншого. Теоретична оцінка показує, що на даних, які за структурою схожі на системні логи, цей механізм дозволяє пропускати значні обсяги даних без покрокових порівнянь, що суттєво зменшує загальну кількість атомарних операцій процесора. У той же час, для класичного Merge Sort кількість порівнянь залишалася б стабільно високою незалежно від ступеня впорядкованості, що підтверджує тези Кнута про важливість адаптації алгоритму до природної структури даних (Natural Merge) [1, С. 165].

Розрахунки прогнозованої ефективності демонструють, що на модельованому кейсі обробки логів гібридний підхід теоретично перевершує класичні методи "розділяй і володарюй" на 30-40% за кількістю операцій. Окремим важливим аспектом дослідження став аналіз властивості стабільності. У системах логування часто трапляються події, що відбулися в одну й ту саму мілісекунду. Для коректного відтворення причинно-наслідкових зв’язків критично важливо зберегти первинний порядок надходження таких записів. Кормен наголошує, що стабільність є обов’язковою вимогою для сортування складних об’єктів [2, С. 213]. Використання стандартного QuickSort, який є нестабільним, призвело б до втрати хронології серед одночасних подій. Натомість гібридний алгоритм на базі злиття природним чином гарантує стабільність, оскільки при рівності елементів перевага завжди надається лівому підмасиву [5].

Також було проаналізовано теоретичне споживання пам’яті. Хоча класичний QuickSort працює "на місці", він потребує стекової пам’яті для рекурсії. Гібридний алгоритм потребує буфера для злиття, проте завдяки оптимізаціям, що обмежують злиття лише необхідними ділянками, реальне споживання пам’яті залишається в межах допустимих норм. Це спростовує побоювання щодо надмірної ресурсоємності методів злиття.

Підсумовуючи теоретичне дослідження, можна стверджувати, що еволюція алгоритмів сортування зміщується від пошуку універсального методу до створення адаптивних рішень. Обґрунтовано, що для задач агрегації логів застосування сліпих універсальних алгоритмів є неефективним. Гібридизація підходів дозволяє компенсувати слабкі сторони окремих методів: сортування вставками ефективно обробляє локальні інверсії, а стратегія злиття гарантує передбачувану асимптотику. Аналітичні висновки свідчать про доцільність використання бібліотечних реалізацій на базі адаптивних алгоритмів для розробки високонавантажених систем аналітики, оскільки вони забезпечують оптимальний баланс між швидкодією та стабільністю.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Knuth D. E. The Art of Computer Programming. Vol. 3 : Sorting and Searching. 2nd ed. Reading : Addison-Wesley, 1998. 780 p.
2. Кормен Т., Лейзерсон Ч., Рівест Р., Стайн К. Вступ до алгоритмів. 3-тє вид. Київ: К.І.С., 2019. 1288 с.
3. Sedgewick R., Wayne K. Algorithms. 4th ed. Upper Saddle River : Addison-Wesley Professional, 2011. 992 p.
4. McIlroy M. D. A Killer Adversary for Quicksort. Software–Practice and Experience. 1999. Vol. 29, No. 4. P. 341–344.
5. Peters T. Timsort description. Python Bug Tracker. 2002. URL: https://bugs.python.org/file4451/timsort.txt.

Верещак Богдана Олександрівна – студентка групи 4ПІ-24б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: 04-24-360.stud@vntu.edu.ua

Власенко Данило Володимирович – асистент кафедри програмного забезпечення, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: vlasenkodanylo@vntu.edu.ua

Vereshchak Bohdana Oleksandrivna – student of group 4PI-24b, Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: 04-24-360.stud@vntu.edu.ua

Vlasenko Danylo Volodymyrovych – Assistant Lecturer at the Department of Software, Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: vlasenkodanylo@vntu.edu.ua