УДК 004.62

М. А. Благун

ОПТИМІЗАЦІЯ ЧАСУ ОБРОБКИ ПРОСТОРОВИХ ЗАПИТІВ У ВИСОКОНАВАНТАЖЕНИХ СИСТЕМАХ

Вінницький національний технічний університет

Анотація

Розглянуто сучасні підходи до оптимізації часу обробки просторових запитів у високонавантажених інформаційних системах.

Ключові слова: просторові індекси, геопросторові дані, високонавантажені системи, оптимізація запитів, R-дерева, Geohash, H3, Quad-tree.

Abstract

Modern approaches for optimizing spatial query processing time in high-load information systems.

Keywords: spatial indexes, geospatial data, high-load systems, query optimization, R-trees, Geohash, H3, Quad-tree.

Вступ

Високонавантажені системи, що містять велику кількість просторових об’єктів (фізичні симуляції, комп’ютерні ігри, навчальні симулятори, географічні цифрові системи тощо), повинні обробляти просторові запити без великих затримок. Попри потужність сучасного апаратного забезпечення, час обробки просторових запитів стандартним способом зростає в квадратичній залежності від кількості просторових об’єктів, оскільки вимагає обробки кожної можливої пари просторових об’єктів. Тому такий спосіб є неприпустимим для систем, які повинні виконувати запити за дуже короткий проміжок часу.

У зв’язку з цим актуальним є дослідження сучасних рішень для оптимізації просторових запитів, що полягає у зменшенні часу, необхідному для їх обробки.

Метою даного дослідження є аналіз існуючих рішень для оптимізації просторових запитів, що полягає в скороченні часу їх обробки.

Просторові індекси

Просторові індекси – це структури даних та методи організації просторової інформації, що використовуються для прискорення обробки просторових запитів. Їх основне призначення полягає у зменшенні кількості об’єктів, які необхідно перевіряти під час виконання операцій пошуку, перетину, визначення сусідства або належності до певної області. Замість послідовного перебору всіх просторових об’єктів система використовує спеціальні принципи поділу простору або кодування координат, що дозволяє швидко відкидати нерелевантні дані та значно скорочувати час виконання запитів [1].

R-дерева (Rectangle Trees) – це один із найпопулярніших типів просторових індексів для об'єктів складної форми (полігонів, ліній). Принцип полягає в групуванні об'єктів у "мінімальні обмежувальні прямокутники" (MBR – Minimum Bounding Rectangles). В процесі індексування виконується побудова ієрархії прямокутників. В процесі обробки просторового запиту, якщо координати не перетинаються з прямокутником вищого рівня ієрархії, система ігнорує всі об'єкти всередині нього [1].

Quad-trees (Четвертинні дерева) – метод індексування просторових даних, що використовується переважно для точкових об'єктів. Принцип полягає в рекурсивному поділі простору на чотири квадранти, доки кількість точок у кожному не досягне певного ліміту. Таким чином, при обробці просторового запиту, система відкидає три чверті необроблених даних на кожній ітерації [2].

Geohash (Сіткові індекси) – метод просторового індексування, що базується на поділі поверхні Землі на прямокутні комірки різного рівня деталізації. Принцип полягає в перетворенні географічних координат у символьний рядок, де кожен наступний символ уточнює положення точки в межах меншої області. В процесі індексування близькі об’єкти отримують схожі префікси Geohash-кодів. Таким

чином, при виконанні просторового запиту система може швидко знаходити об’єкти в сусідніх областях шляхом пошуку за відповідними префіксами [3].

H3 (Hexagonal Hierarchical Spatial Index) – метод просторового хешування, розроблений компанією Uber Technologies, що використовує ієрархічну сітку шестикутників для представлення просторових даних. Принцип полягає в розбитті поверхні Землі на комірки однакової форми, кожна з яких має унікальний хеш-ідентифікатор. В процесі індексування кожна координата прив’язується до певного шестикутника відповідного рівня деталізації. Таким чином, при обробці просторових запитів система може ефективно визначати сусідні області та виконувати агрегацію даних без складних геометричних обчислень [4].

Оптимізація у високонавантажених системах

Для систем з мільйонами запитів на секунду (високонавантажені), використання лише просторового індексу не завжди достатньо для швидкої обробки запитів. На сьогоднішній день застосовуються різні перспективні стратегії. Наприклад:

1. Гео-шардування (Spatial Sharding): Розподіл даних між різними серверами на основі географічних зон. Наприклад, дані користувачів з Києва зберігаються на одному вузлі, а з Лондона – на іншому [5].
2. In-memory обробка – це стратегія оптимізації, що полягає у зберіганні та обробці просторових даних безпосередньо в оперативній пам’яті (RAM), а не на диску. Такий підхід використовується у високонавантажених системах, де координати об’єктів часто змінюються, а затримка навіть у декілька мілісекунд може бути критичною. На відміну від традиційних баз даних, які виконують читання та запис через файлову систему, in-memory системи забезпечують доступ до даних практично миттєво, що значно прискорює виконання просторових запитів [6].
3. Двоетапна фільтрація (Two-stage filtering) – це стратегія оптимізації просторових запитів, що полягає у послідовному застосуванні двох рівнів перевірки даних: швидкого попереднього відбору та точного геометричного аналізу. Такий підхід використовується для зменшення кількості складних обчислень, необхідних при роботі з великою кількістю просторових об’єктів. Основна ідея полягає в тому, щоб спочатку швидко відкинути більшість нерелевантних об’єктів, а потім виконувати ресурсоємні обчислення лише для невеликої кількості кандидатів [7].

Обмеження та компроміси просторового індексування

Під час проєктування високонавантажених систем із підтримкою просторових запитів важливо враховувати не лише переваги просторових індексів, а й обмеження, пов’язані з їх використанням. Різні типи індексів демонструють різну ефективність залежно від характеру навантаження, частоти оновлення даних та вимог до точності обчислень. У зв’язку з цим вибір конкретного підходу зазвичай є компромісом між швидкістю обробки, точністю результатів та витратами ресурсів системи [8].

•Write-heavy load: Часте оновлення координат створює додаткове навантаження на перебудову або модифікацію просторових індексів. Наприклад, для R-дерев велика кількість операцій вставки та видалення може призводити до зниження продуктивності через необхідність перерахунку ієрархії мінімальних обмежувальних прямокутників. У таких випадках Geohash або H3 є ефективнішими, оскільки оновлення індексу часто зводиться до простої заміни хеш-значення в Key-Value сховищі.

•Точність vs Швидкість: Для підвищення продуктивності деякі системи використовують наближені алгоритми просторових обчислень. Наприклад, обчислення відстаней можуть виконуватися на площині замість сферичної поверхні Землі. Такий підхід значно зменшує обчислювальні витрати, проте може створювати похибки при роботі з великими географічними.

•Споживання пам’яті: Просторові індекси можуть вимагати значного обсягу оперативної пам’яті, особливо у випадку великих наборів даних або використання in-memory систем. Зі збільшенням деталізації індексів зростає і кількість службових структур, необхідних для їх підтримки, що створює додаткове навантаження на апаратні ресурси.

•Нерівномірність розподілу даних: Ефективність деяких просторових індексів залежить від рівномірності розподілу об’єктів у просторі. Наприклад, у густонаселених областях або місцях із великою концентрацією об’єктів окремі комірки Geohash чи H3 можуть містити надмірну кількість елементів, що частково знижує ефективність фільтрації та збільшує кількість перевірок під час виконання запитів.

Висновки

У результаті дослідження було проаналізовано сучасні методи оптимізації просторових запитів у високонавантажених системах та визначено основні підходи до пришвидшення роботи з просторовими даними. Встановлено, що використання просторових індексів дозволяє скоротити кількість перевірок під час виконання запитів, що забезпечує зменшення часу обробки порівняно з повним перебором усіх об’єктів. Розглянуті структури даних демонструють різні переваги залежно від типу просторових об’єктів, характеру навантаження та вимог до швидкодії системи.

Дослідження також показало, що в сучасних високонавантажених системах ефективність просторових індексів значною мірою залежить від використання додаткових стратегій оптимізації, таких як гео-шардування, in-memory обробка та двоетапна фільтрація. Їх поєднання дозволяє забезпечити пришвидшення обробки великої кількості просторових запитів навіть за умов використання просторових індексів.

Крім того, встановлено, що вибір конкретного методу просторового індексування є компромісом між швидкістю обробки, точністю результатів, складністю підтримки індексів та витратами апаратних ресурсів. Тому під час проєктування високонавантажених систем необхідно враховувати специфіку даних, характер запитів і вимоги до масштабованості системи для вибору найбільш ефективного підходу до просторового індексування.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Spatial Databases: A Tour/ Shashi Shekhar, Sanjay Chawla, 2003. URL: https://books.google.com.ua/books/about/Spatial_Databases.html?id=ZOVOAAAAMAAJ&redir_esc=y
2. The Design and Analysis of Spatial Data Structures/ Hanan Samet, 1990. URL: http://lib.ysu.am/disciplines_bk/cc247e24465ba197cb9cdc9d74430272.pdf
3. IBM Knowledge Center Geohashes Guide. URL: https://www.ibm.com/docs/en/streams/4.3.0?topic=334-geohashes
4. H3 Documentation Site. URL: https://h3geo.org/
5. Efficient QoS-Aware Spatial Join Processing for Scalable NoSQL Storage Frameworks/ I. M. Al Jawarneh, P. Bellavista, 2020. URL: https://www.researchgate.net/publication/346435066_Efficient_QoS-Aware_Spatial_Join_Processing_for_Scalable_NoSQL_Storage_Frameworks
6. Processing-in-Memory for AI, Joo-Young Kim, Bongjin Kim, Tony Tae-Hyoung Kim, 2022. URL: https://link.springer.com/book/10.1007/978-3-030-98781-7
7. A Two-stage decision model for information filtering/ Yuefeng Li, Xujuan Zhou, Peter Bruza, Yue Xu, Raymond Y. K. Lau, 2012. URL: https://researchers.mq.edu.au/en/publications/a-two-stage-decision-model-for-information-filtering/
8. Indexing Spatial Data. URL: https://postgis.net/

Благун Михайло Андрійович – студент групи 1ПІ-25м, факультет інформаційних технологій та комп’ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: me.shine.factory@gmail.com

Науковий керівник: ***Ліщинська Людмила Броніславівна*** – д-р техн. наук, професор, професор кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: llb@vntu.edu.ua

Blahun Mykhailo A. – Department of Information Technology and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: me.shine.factory@gmail.com

Supervisor: ***Lishchynska Lyudmyla Bronislavivna*** – Dr. Sc. (Eng.), Full Professor, Professor of Program Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: llb@vntu.edu.ua