УДК 004.658

## ПРОГРАМНИЙ КОМПЛЕКС ДЛЯ ЕКСПЕРИМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ ЕФЕКТИВНОСТІ ПОШУКУ ДАНИХ У БАЗАХ ДАНИХ ІЗ ЗАСТОСУВАННЯМ KMF–ДЕРЕВ

**Я. І. Корнага**
Національний технічний університет України «Київський політехнічний інститут»
пров. Ковальського, 22а, м. Київ, 03056, Україна. E–mail: slovyan_k@ukr.net

Розроблено узагальнену структуру програмного комплексу для оцінки ефективності пошуку даних у базах даних на основі KMF-дерев. Створений програмний комплекс дозволяє застосувати запропоновані KMF-дерева для побудови індексів таблиць бази даних. Описано особливості програмної реалізації даного комплексу та побудовано діаграму класів для нього. Проведено експериментальне дослідження параметрів пошуку даних на основі механізмів B+-дерев і KMF-дерев, які показали, що останні забезпечують зменшення часу пошуку даних у базах даних.

**Ключові слова:**пошук даних, індекси, B+-дерева, KMF-дерева.

## ПРОГРАММНЫЙ КОМПЛЕКС ДЛЯ ЭКСПЕРИМЕНТАЛЬНЫХ ИССЛЕДОВАНИЙ ЭФФЕКТИВНОСТИ ПОИСКА ДАННЫХ В БАЗАХ ДАННЫХ С ПРИМЕНЕНИЕМ KMF-ДЕРЕВЬЕВ

**Я. И. Корнага**
Национальный технический университет Украины «Киевский политехнический институт»
пер. Ковальського, 22а, г. Киев, 03056, Украина. E–mail: slovyan_k@ukr.net

Разработана обобщенная структура программного комплекса для оценки эффективности поиска данных в базах данных на основе KMF-деревьев. Созданный программный комплекс позволяет применить предложенные KMF-дерева для построения индексов таблиц базы данных. Описаны особенности программной реализации данного комплекса и построена диаграмма классов для него. Проведено экспериментальное исследование параметров поиска данных на основе механизмов B+-деревьев и KMF-деревьев, которые показали, что последние обеспечивают уменьшение времени поиска данных в базах данных.

**Ключевые слова:** поиск данных, индексы, B+-деревья, KMF-деревья.

АКТУАЛЬНІСТЬ РОБОТИ. У базах даних (БД) важливим елементом є пошук даних в таблицях, які можуть мати велику кількість рядків, що зберігаються в довільному порядку, та їх пошук за заданим критерієм шляхом послідовного перегляду таблиці рядок за рядком може займати багато часу. Індекс формується зі значень одного чи декількох стовпців таблиці і вказівників на відповідні рядки таблиці та дозволяє шукати рядки, що задовольняють критерію пошуку. Прискорення роботи з використанням індексів досягається в першу чергу за рахунок того, що індекс має структуру, оптимізовану під пошук, наприклад, збалансованого дерева [1].

Базовим апаратом для пошуку даних у БД є B-дерева. В основі цього механізму лежать такі ідеї.

*По-перше*, оскільки мова йде про структури даних у зовнішній пам'яті, загальний час доступу до якої визначається, в основному, не обсягом послідовно розташованих даних, а часом підведення магнітних головок, то вигідно отримувати за одне звернення до зовнішньої пам'яті як можна більше інформації, враховуючи при цьому необхідність економного використання основної пам'яті.

При сформованому підході до організації основної пам'яті у вигляді набору сторінок рівного розміру природно вважати саме сторінку одиницею обміну із зовнішньою пам'яттю.

*По-друге*, бажано забезпечити таку пошукову структуру у зовнішній пам'яті, при використанні якої пошук інформації за будь-яким ключем вимагає заздалегідь відомого числа обмінів із зовнішньою пам'яттю [2].

Для створення індексів у сучасних базах даних використовуються B+-дерева, але при роботі з ними виникають проблеми з компактністю, адже вузли у них заповнені не менше ніж на 1/2. Потрібно вирішувати проблему з більш компактним заповненням та відповідно з меншою кількістю рівнів дерева [3].

На основі цього було запропоновано новий вид дерев KMF-дерева, відмінність яких від B+-деревом полягає в тому, у KMF-дереві більша кількість елементів у корені вузла, що потрібно для проведення операцій з'єднання та роз'єднання вузлів, у KMF-дереві на відміну від B+-дерева більша степінь наповненості вузла – вона складає 3/4 вузла та змінені принципи розбиття (з трьох чотири) при переповненні та з'єднання, при досягненні мінімальної наповненості, (з трьох в два) вузлів [4]. Це дозволяє економити місце на жорсткому диску та збільшити швидкість доступу до інформації за рахунок того, що зчитування вузлів KMF-дерева, як і B+-дерева, відбувається по блоках, і відповідно до наповненості блока залежить кількість інформації, яку ми отримаємо.

Метою даної роботи є створення програмного комплексу для порівняння оцінки часу пошуку в таблицях бази даних за індексами, основаними на B+- та KMF-дереві.

МАТЕРІАЛ І РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ. *Узагальнена структура та модулі засобів пошуку в БД*. Одним з ефективних механізмів опису функціонування складних програмних комплексів, яким виступає спеціалізоване середовище для пошуку даних у БД, є мова UML.

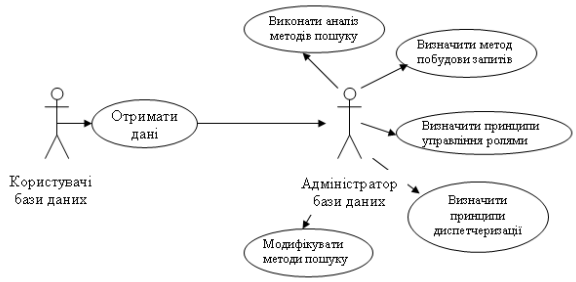


Рисунок 1 – UML діаграма варіантів використання при побудові засобів пошуку в БД

Як показує рис. 1, користувачі бази даних у клієнт-серверних системах проводять пошук даних за відповідними параметрами, причому СУБД за допомогою адміністратора БД повинна виконати дії з аутентифікації користувача, провести аналіз запиту, визначити методи пошуку, якщо потрібно модифікувати їх і перевірити на сумісність з іншими схемами баз серверу БД, тобто провести аналіз за принципам диспетчеризації.

Архітектура системи роботи з базою даних включає в себе 3 наступних рівня (рис. 2):

1. Рівень сховища даних, на якому знаходяться внутрішня робоча база даних, а також фрагменти зовнішніх джерел інформації.

2. Рівень серверу додатків, який має модульну структуру, що забезпечує ефективне управління роботою СУБД.

3. Рівень графічного інтерфейсу користувача (GUI).

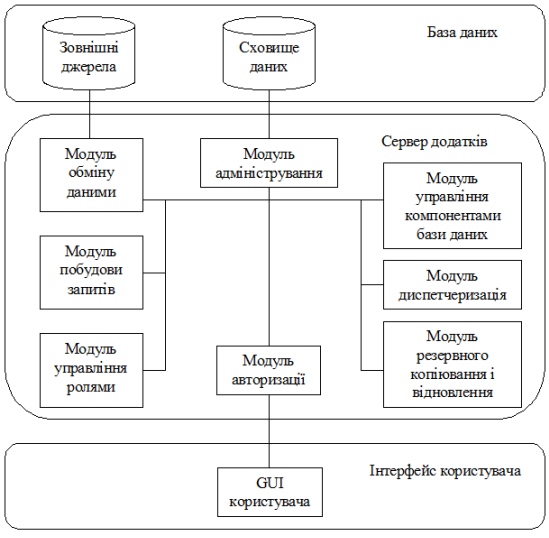


Рисунок 2 – Архітектура системи роботи з БД

Сервер додатків включає в себе наступні модулі:

– *модуль обміну даними* забезпечує ефективний обмін інформаційних потоків між СУБД і зовнішніми підключеннями користувачів бази даних;

– *модуль адміністрування* є основним модулем у роботі СУБД, який забезпечує роботу всієї системи управління базою даних, і керує ним адміністратор серверу баз даних, а саме: керує та налаштовує обмін даними з зовнішніми ресурсами, управляє та налаштовує роботу по роботі з даними у оперативній пам'яті з використання дискового простору, налаштовує роботу із журналізації, резервного копіювання та відновлення під час збоїв у роботі СУБД, забезпечує управління ядром БД і низкою сервісних програм (зовнішніх утиліт), налаштовує розміщення файлів серверу БД на дисках і виділення оперативної пам'яті користувачам, управляє системними таблицями та службами підтримки нормальної роботи серверу, управляє всіма схемами БД та їхніми користувачами;

– *модуль управління ролями* забезпечує налагодження ефективної роботи користувачів БД, через надання відповідних ролей кожному користувачеві для доступу до таблиць, уявлень, функцій та процедур (зареєстрований користувач), схеми бази даних (адміністратор схеми бази серверу БД), серверу бази даних і подій на сервері БД (адміністратор сервера баз даних, який і управляє модулем адміністрування);

– *модуль управління компонентами БД* управляє роботою таблиць, процедур, функцій, тригерів, представлень і пакетів для кращого використання в СУБД і застосування їх у створені різноманітних схем БД;

– *модуль диспетчеризації* управляє потоками даних, між різними схемами в СУБД через створення процедур, які будуть оновлювати дані у відповідних таблицях відповідних схем у відповідний час;

– *модуль авторизації* забезпечує надання користувачам прав доступу на виконання певних дій з елементами схеми бази даних після проходження користувачем аутентифікації з використанням перевірки паролю та цифрового підпису користувача;

– *модуль побудови запитів* використовує спеціальні програми для забезпечення швидкої побудови SQL–запитів для управління та роботи з СУБД;

– *модуль резервного копіювання і відновлення* забезпечує виконання механізмів для створення «гарячих» резервних копій з робочої БД у відповідні періоди доби (в моменти мінімальних навантажень на сервер) у відновлення роботи бази даних із резервної копії при аварійній зупинці сервера.

*Особливості програмної реалізації спеціалізованого середовища*. Для написання програмної реалізації було вибрано мову програмування Java Eclipse під управлінням операційної системи Linux Ubuntu, у зв'язку з тим, що програмні реалізації задач на платформі сімейства Windows не показують достатньої швидкості в обробці даних і точного часу пошуку даних у базах даних через особливості закритості системи для користувачів [3, 5, 6].

На рис. 3 показана UML діаграма класів, що описує типи об'єктів середовища і статичні відносини, які існують між ними. На даній діаграмі відображені лише основні (без деталізації) атрибути класів, операції класів і обмеження, які накладаються на зв'язок між об'єктами.

Клас ***Main*** є центральним класом системи пошуку даних, що управляє класами, запускає та викори-

стовує функції з інших класів через свою основну функцію *main*. До нього додатково входять наступні функції:

– *doctxt_read* – використовується для зчитування слів з текстового файлу, основна змінна *read_list,* в яку записуються слова;

– *crc32gen* – використовується для шифрування слів, що були зчитані за допомогою *doctxt_read***,** змінна *crc32_ret*, в яку записуються контрольні суми слів у десятковій системі числення;

– *doctxt_write* **–** функція для запису до файлу шифрованих слів із функції *crc32gen***,** змінна *white_list* передає шифровані слова для запису;

– *hash_search* – функція пошуку потрібного ключа в масиві.

Клас ***GUI*** призначений для формування інтерфейсу користувача та використовує *SWT* бібліотеку з її класами: ***Display***, – який створює форму на якій відображаються елементи інтерфейсу, ***Button*** – клас, за допомогою якого створюються кнопки до яких прикріплюються функції вводу–виводу даних, ***Text*** – клас для виведення текстової інформації.

Клас ***BPlusTree*** слугує для побудови B+-дерева, додавання нових елементів та пошуку в ньому ключа, змінними класу є: *nodebplus* – вузол пошуку, *keybplus* – ключ пошуку. Основними функціями є:

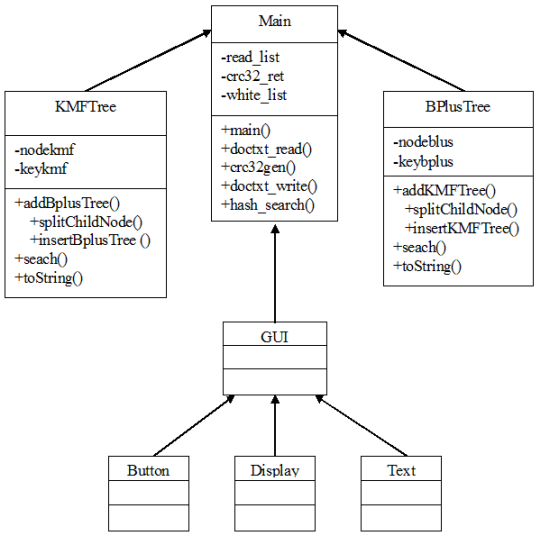


Рисунок 3 – UML діаграма класів спеціалізованого середовища для моделювання і аналізу системи пошуку даних у БД

– *addBPlusTree* **-** функція створення B+-дерева, використовує підфункції *splitChildNode* (зчитування хеш-ключів в пам'ять), *insertBplusTree* (створення дерева та добавлення нового елемента в вузол);

– *seach* – функція рекурсивного методу пошуку в вузлі дерева;

– *toString* – функція проходу по дереву до потрібного вузла.

Клас ***KMFTree*** слугує для побудови KMF–дерева, додавання нових елементів та пошуку в ньому ключа, змінними класу є: *nodekmf* – вузол пошуку, *keykmf* – ключ пошуку. Основними функціями є: *addKMFTree* **–** функція створення KMF-дерева, використовує підфункції *splitChildNode* (зчитування хеш–ключів у пам'ять), *insertKMFTree* (створення дерева та добавлення нового елемента в вузол), а також використовуються функції *seach* та *toString*, які виконують такі ж функції, як у класі ***BPlusTree***.

У процесі експериментальних досліджень використовувалася локальна комп'ютерна система з наступними характеристиками: процесор Intel Core i7-263QM 2Gz, оперативна пам'ять – 4Гб, яка функціонує під управлінням операційної системи Linux Ubuntu.

Програма використовувала реляційну базу даних, що була розроблена на мові програмування Java Eclipse, на основі таблиць, які записані в кожному окремому файлі та індексних файлів, в які зберігалися індекси, що були сформовані на основі стандартних методів побудови – B+-дерев та розроблених нових – KMF-дерева. Оцінювався час пошуку інформації по відповідному індексу в кожній таблиці, який був зафіксований функціями фіксації часу Java (nanotime).

Для проведення експерименту з оцінки часу пошуку даних розглядалися таблиці з такою кількістю записів, як: 5000, 10000, 50000, 100000, 500000, 1000000 і 5000000 записів. Для кожної таблиці кількість проведених експериментальних досліджень складала 30 пошуків. Результати, що були отримані, звелися в спільну таблицю, за якою проводилася оцінка ефективності шляхом розрахунку середнього арифметичного за кожним видом пошуку.

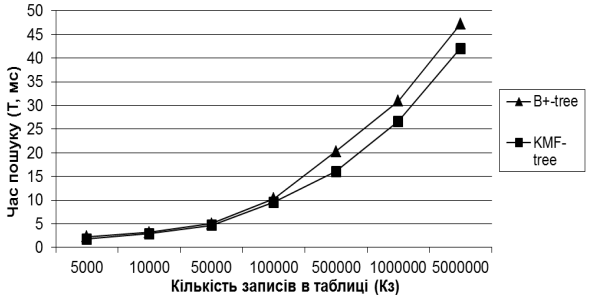


Рисунок 4 – Залежність часу пошуку від кількості записів для різних методів формування індексів у БД

Для оцінки точності проведення експериментів та виведення адекватних результатів було прораховано середньоквадратичне відхилення. Воно склало для B+-дерев – 0,91 %, а для KMF-дерев – 0,89 %, що є підставою для позитивного прийняття результатів досліджень.

У цілому ефективність використання KMF-дерев на заданому проміжку кількості записів в таблицях бази даних, як показано на рис. 4, складає 13 % порівняно з B+-деревами, що дає значне пришвидшення в роботі бази даних і дозволяє витрачати її ресурси на інші потреби.

ВИСНОВКИ. Програмний комплекс для порівняльної оцінки часу пошуку інформації в базах даних за індексами на основі B+- і KMF-дерев показав наступний результат: на невеликій кількості записів KMF-дерева дають зменшення часу пошуку на відповідних проміжках, яке становить в середньому 13 %, але зі збільшенням кількості записів, час пошуку за допомогою індексів, основаних на KMF-деревах значно зменшуватиметься порівняно з B+-деревами, що прискорить роботу бази даних і дозволить витрачати ресурси серверів, на яких розміщені БД, з більшою ефективністю.

Такі дерева, в майбутньому, можуть бути застосовані в програмних продуктах (не тільки у базах даних), що використовують індексацію будь-якої інформації. Це призведе до пришвидшення часу пошуку та зменшить кількість звертань до жорсткого диску, що позволить збільшити його ресурс використання.

ЛІТЕРАТУРА

1. Маркин А.В. Построение запросов и программирования на SQL. – Рязань: РГРТУ, 2008. – 312 с.

2. Райордан Р. Основы реляционных баз данных. – Москва: Русская Редакция, 2009. – 384 с.

3. Ульман Д. Введение в системы баз данных. – Москва: Лори, 2008. – 374 с.

4. Мухін В.Є., Корнага Я.І., Снєгірєв Л. Підвищення ефективності механізмів пошуку в базах даних на основі KMF-дерев // Вісник Національного університету «Львівська політехніка». Серія Компютерні науки та інформаційні технології. – 2012. – № 744. – 423 с.

5. Бек К. Расширения Eclipse. Принципы, шаблоны и подключаемые модули. – Москва: Образ, 2005. – 384 с.

6. Mcaffer J. Eclipse Rich Client Platform: Designing, Coding, and Packaging Jav Applications. – Addison Wesley Professional, 2005. – 384 с.

## SOFTWARE COMPLEX FOR EXPERIMENTAL RESEARCH OF THE DATA RETRIEVAL EFFECTIVENESS IN DATABASE USING KMF-TREES

**Ya. Kornaga**

National Technical University of Ukraine “Kyiv Polytechnic Institute”

per. Kovalskogo, 22a, ap. 806, Kyiv, 03056, Ukraine. E-mail: slovyan_k@ukr.net

The generilized structure of software complex for effectiveness evaluation of the data retrieval in databases on the basis of suggested KMF-trees. This software allows applying of suggested KMF-tree for indexes computation of database tables. The features of the designed software complex are described and its class diagram is charted. The experimental study of parameters of data retrieval mechanisms based on B +-trees and KMF-trees is performed, and the experimental results indicated that the KMF-trees suggested provide the data retrieval time saving.

**Key words:** data search, indexes, B+-trees, KMF-trees.

REFERENCES

1. Myhin V., Kornaga Y., Snegirev L. Increase of efficiency mechanisms for database search based on the KMF-trees //*Journal of National university “Lviv Polytechnic” “Computer science and information technology”*. – 2012. – № 744. – 423 p. [in Ukrainian]

2. Markin A.V. *Postroenie zaprosov I programmirovaniya na SQL.* [Building queries and SQL programming]. – Ryazan: RGRTY, 2008. – 312 p. [in Russian]

3. Rayordan R. *Osnovy relyatsionnykh baz dannykh* [Fundamentals of Relational Databases]. – Moscow: Ryskaya Redakciya, 2009. – 384 p. [in Russion]

4. Ulman D. *Vvedenie v sistemy baz dannykh* [Introduction to database systems]. – Moscow: Lory, 2008. – 374 p. [in Russian]

5. Beck K. Rasshireniya Eclipse. Printsipy, shablony I podklyuchaemye moduli [Eclipse Extentions. Principles, patterns, and plugins]. – Moscow: Obraz, 2005. – 384 p. [in Russian]

6. Mcaffer J. *Eclipse Rich Client Platform: Designing, Coding, and Packaging Jav Applications*. – Addison Wesley Professional, 2005. – 384 p.

Стаття надійшла 06.12.2012.