Розробка сучасної інформаційно-пошукової системи

К. В. ЯШИНА, К. М. ЯЛОВА, В. В. ЗАВГОРОДНІЙ

Дніпродзержинський державний технічний університет

У статті наведено опис нової інформаційно-пошукової системи, що здійснює високоякісний пошук за ключовими словами. Система розроблена за допомогою використання сучасний безкоштовних технологій веб-програмування, трирівневої архітектури. У статті описані пошукова модель, алгоритми індексації та послідовність програмних дій із системою, наведені результати її використання.

В статье приведено описание новой информационно-поисковой системы, осуществляющей высококачественный поиск по ключевым словам. Система разработана с помощью использования современных бесплатных технологий веб-программирования, трехуровневой архитектуры. В статье описаны поисковая модель, алгоритмы индексации и последовательность программных действий с системой, приведены результаты ее использования.

The paper describes a new informational-retrieval system that realizes high-quality keyword search. The system is elaborated through the use of free modern technology of web programming, a three-layer architecture. This article describes the search model, indexing algorithms and sequence of programming actions with the system, the results of its use.

Вступ. Величезна кількість як структурованих, так і неструктурованих даних доступна нам за допомогою мережі Інтернет. Пошукова система – це автоматизована інформаційна система (програмно-апаратний комплекс із веб-інтерфейсом), що надає можливість пошуку інформації в Інтернеті у відповідності до запиту користувача [1]. Не зважаючи на схожість із задачами інформаційного пошуку даних на основі обробки SQL-запитів до бази даних, пошук даних в Інтернеті має наступні особливості:

- на алгоритми пошуку даних суттєво впливає той факт, що структура взаємозв'язку між сторінками не відома заздалегідь, а формат даних, що розміщуються на інтернет-ресурсах, ніким не контролюється і не встановлюється;
- запитом пошуку є текст, написаний користувачем на природній для нього мові;
- вводиться поняття конкурентної інформації, що вимагає встановлення рівня релевантності отриманої відповіді до заданого запиту.

Областю використання пошукових систем є мільярди сторінок Інтернет. При цьому приблизно мільйон сторінок додається щодня і стільки ж оновлюється, а всі дані, що на них містяться, представляються різними цифровими форматами. Стрімкий приріст вмісту глобальної мережі Інтернет ставить лідерів пошукових послуг та розробників програмних технологій перед необхідністю знаходити нові механізми, ресурси та алгоритми, які б забезпечували належний рівень швидкості та якості обробки різноманіття користувацьких запитів.

Постановка задачі. У світі зараз існують сотні пошукових систем, багато з яких є локальними (орієнтованими на певну країну, мову, тощо) або неповними. Незважаючи на існуючу безліч готових рішень для розробки пошукових систем, більшість з них орієнтована на індексацію документів, файлів, HTML-сторінок, тощо. Перетворення цих систем для індексації посилань на файли може зайняти багато часу та коштів (більшість готових рішень потребують отримання ліцензії на використання у комерційних цілях).

Метою даної роботи є представлення результатів розробки інформаційно-пошукової системи, дії якої направлені на пошук за ключовими словами, а саме: ім'я файлу та заголовок сторінки, на якій його було знайдено. Програмні вимоги до інформаційно-пошукової системи наступні:

- використання безкоштовних сучасних програмних засобів;
- можливість застосування системи в рамках корпоративних інформаційних сховищ, файлообмінних сервісах та інформаційних порталах з метою пошуку збережених файлів;
- наявність високого рівня якості результатів пошуку, до яких відноситься: точність – відношення кількості знайдених релевантних файлів до загальної кількості знайдений документів; повнота – відношення кількості знайдених релевантних файлів до загальної кількості релевантних документів в базі.

Результати роботи. Аналіз предметної області показав, що при розробці архітектури пошукової системи доцільно використовувати класичну трирівневу архітектуру інформаційної системи, поділену на: користувацький інтерфейс, бізнес-логіку та базу даних. ***Архітектура розробленої інформаційно-пошукової системи***, яка наведена на *рис. 1*, містить наступні складові частини:

1) інтерфейс користувача, що забезпечує можливість формування користувацького запиту та отримання результатів пошуку;

2) пошукова машина – ядро системи, що представлене набором програмних модулей для реалізації механізмів пошуку даних;

3) база даних або індекс пошукової системи.

Окрім складових частин пошукової системи на *рис. 1* наведено загальну схему пошукового процесу, основними діями якого є пошук нових документів, індексування знайдених документів та виконання запитів користувачів. Особливої уваги заслуговує принципи роботи пошукової машини, яка складається із павука, краулера, індексатора, підсистем ранжування та видачі результатів пошуку.

Павук – це програмний додаток, який виконує завантаження сторінок Інтернет ресурсів.

Після того, як вміст заданої сторінки завантажено, *індексатор* виконує її первинний аналіз, виділяє основні складові частини (назву сторінки, опис, посилання, заголовки, тощо) та розкладує все це по розділам бази даних.

Краулер – програмний додаток, що автоматично проходить по всім посиланням, знайденим на заданій сторінці, він аналізує шляхи, які ведуть з поточної сторінки на інші розділи сайту або на сторінки зовнішніх Інтернет ресурсів, що визначає подальший порядок обходу павука [2]. Дії підсистем ранжування та видачі результатів пошуку направлені на визначення того, які саме сторінки задовольняють запиту користувача та в якому порядку вони повинні бути відсортовані.

Рис. 1. Архітектура розробленої інформаційно-пошукової системи

Не менш важливою частиною пошукової машини інформаційно-пошукової системи є ***модель пошуку інформації***. Вона описує спосіб і критерії порівняння запитів і документів, а також форму представлення результатів цього порівняння. В рамках даної роботи використана булівська модель пошуку, що дозволяє створювати логічні вирази з набору термінів. Нехай мається деякий словник (T) [3]:

$$T = \{t_1, \ldots, t_n\},$$

де t_i – це терми.

Документ, що необхідно знайти – це деяка підмножина словника, що по суті є набором термів:

$$D \in T \text{ , або } D \in \{0,1\}^n$$

на k-ій позиції вектору стоїть одиниця в тому випадку, коли k-е слово зі словника належить документу, і нуль – в іншому випадку.

Запит користувача – це набір ключових слів із булівськими зв'язками, наприклад:

$$t_1 \text{ OR } t_2 \quad (1)$$

$$t_1 \text{ AND } t_2, \quad (2)$$

де t_1 – ім'я файлу, який необхідно знайти; t_2 – заголовок сторінки.

Наведений в (1) вираз означає, що необхідно знайти документи, які містять перший або другий терм. Вираз (2) задає пошук документів, які містять перший і другий терм одночасно. Якщо вирази (1)-(2) виконались на деякому документі – будемо вважати, що даний документ відповідає запиту. Оскільки в булівській моделі відсутній механізм ранжування знайдених документів, то в рамках представленої роботи був задіяний власний механізм, який полягає в сортуванні результатів пошуку за збігом ключових слів та представлення документів із найточнішими результатами на початку списку результатів пошуку. Послідовність дій процесів індексації та пошуку в рамках системи задається розробленими відповідними алгоритмами. ***Алгоритм індексації*** складається з наступних кроків:

Крок 1. Визначити всі посилання на документи.

Крок 2. Кожне з виявлених посилань перевіряється на актуальність (адже файли з часом видаляються з серверів). Для цього:

Крок 2.1. Завантажити цільове посилання. У даному випадку ця подія асинхронна, тому необхідно вказати функцію-колбек, що виконається після завантаження.

Крок 2.2. Підключити бібліотеку jQuery для більш зручних маніпуляцій з документом. Ініціалізувати об'єкт result, який буде використовуватись індексатором.

Крок 2.3. Перевірити документ на наявність деяких контрольних тегів, визначених шляхом порівняння існуючого та неіснуючого файла. Зберегти ім'я файла, розмір (якщо передбачено).

Крок 2.4. Використати об'єкт result для відправки готових упорядкованих даних на сервер.

Крок 3. За допомогою пошукового робота здійснити відправку даних на сервер, а саме:

Крок 3.1. Сформувати пари найменування=значення та відправити їх на сервер для обробки та збереження в базі даних.

Крок 3.2. Завантажити значення з кроку 3.1 та додати їх уміст до бази даних. Одразу після додавання до бази даних, посилання стає доступним для пошуку та займає певне місце при пошукових запитах (відповідно до запиту).

Крок 4. Відмітити поточну веб-сторінку як проіндексовану.

Крок 5. Завантажити наступну веб-сторінку з черги. Повторити дії, починаючи з кроку 1. Індексатор можна безперешкодно запускати в період, коли комп'ютер використовується, тому що він є окремим процесом та не потребує втручань у процес індексації.

Для ***програмної реалізації*** інтерфейсної частини пошукової системи була обрана мова програмування PHP. При розробці інтерфейсу користувача системи були використані механізми зрозумілості, зокрема використання метафори. Тобто, інтерфейс системи розроблено по аналогії з широко відомими пошуковими системами Google, Yandex, тощо. Екранна форма містить поле для формування запиту користувачем та командну кнопку для запуску пошуку (*рис. 2*).

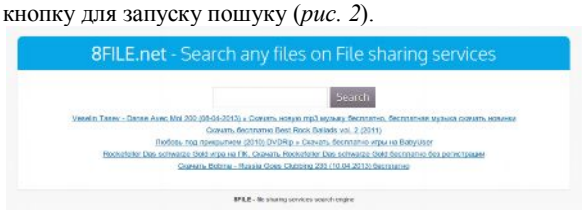

Рис. 2. Головна сторінка пошуку

Для прошарку „База даних" (*рис. 1*) пошукової системи використовується база даних, розроблена засобами СУБД MySQL та SQL-запити для пошуку за ключовими словами. Логічну модель бази даних із визначеними таблицями, їх атрибутами, типами даних та зв'язками представлено засобами діаграми класів уніфікованої мови моделювання UML (*рис. 3*).

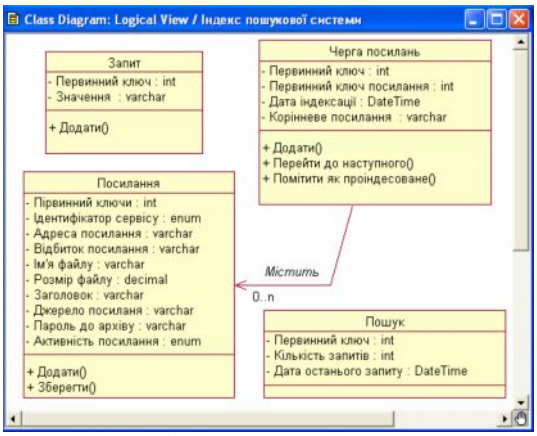

Рис. 3. Логічна структура бази даних

Структура бази даних містить 4 основні таблиці, з яких таблиця „Посилання” призначена для збереження структури посилань та їх основних атрибутів. Для додавання веб-сайту в чергу індексації, використовується таблиця „Черга посилань”.

В якості значення „Відбиток посилання” вказується хеш посилання, що прискорює операції пошуку та зменшує розміри індексу (кожен відбиток використовує 40 байтів).

Таблиця „Запит” містить значення ключових слів, які задаються в якості параметрів запиту користувача. Для забезпечення програмної реалізації підсистеми ранжування знайдених документів використовується таблиця „Пошук”, до складу якої входять відомості стосовно його результатів.

Для реалізації операції повнотекстового пошуку використовуються операції MATCH() та AGAINST(). Функція MATCH() виконує пошук у природній мові, порівнюючи рядок із умістом тексту запиту (сукупність одного і більше стовпчиків, які включені в індекс FULLTEXT). Рядок задається як аргумент у виразі AGAINST(). Пошук виконується без урахування реєстру символів. Для кожного рядка стовпця в заданій таблиці команда MATCH() повертає величину релевантності, тобто ступінь схожості між рядком пошуку і текстом. Пошуковий запит, записаний структурною мовою запитів SQL має наступний вигляд:

SELECT links.* FROM links WHERE MATCH (title,filename) AGAINST(“самоучитель”)

До програмних рішень системи також включена опція використання NoSQL рішень (MongoDB, CouchDB тощо), що дозволить розподіляти навантаження у разі збільшення бази даних до кількох гігабайтів, та для паралельного зберігання реплік (декілька синхронізованих баз даних, що мають однаковий зміст, та можуть бути реставровані у разі повної поломки, видалення, або випадкового знищення за якихось умов однієї з копій). Такі бази даних є документоорієнтованими та використовують об’єкти (масиви) для зберігання та обробки інформації, на відміну від MySQL, що використовує рядки та мову SQL для обробки.

Для реалізації зв’язки Павук-Краулер-Індексатор використовується PhantomJS. PhantomJS – консольний браузер на базі WebKit – безкоштовного двигуна з відкритим кодом для відображення веб-сторінок [4]. За допомогою невеликого програмного коду PhantomJS можна перетворити на пошуковий робот, а його відокремлена реалізація дає змогу зняти навантаження на сервер та запускати його у багатьох екземплярах, обробляючи тільки ту частину посилань, що йому надається. Переваги використання PhantomJS полягають у наступному:

- повністю контрольований браузер;
- використання будь-якої бібліотеки та будь-якого JavaScript;
- відбір елементів CSS селекторами;
- підтримка усіх об’єктів JavaScript: синхронні/асинхронні HTTP запити, Base64 кодування, XML/XHTML парсер і т.д.;
- швидкість розробки: для отримання результатів достатньо підключити будь-яку бібліотеку (jQuery, Prototype і т.п.) та відібрати потрібні елементи. Результат можна відправити на сервер, або вивести у консоль;
- підтримка Proxy-серверів: з’єднання з вебсайтами використовуючі інши сервери;
- прихованість: редагування „підпису” браузера – User-Agent, сервери нездатні розпізнати чи це скрипт, чи користувач зі звичайним браузером;
- скрипт виконання розробляється на JavaScript, та може бути виконаний (підключений) у любий момент без модифікації самого браузера. Тобто, можна переносити скрипт на іншу операційну систему та виконати його без додаткових зусиль, що дає змогу швидко почати роботу, без жодних програмних настроювань.

Після запуску пошукового роботу база даних починає наповнюватись та можна починати користуватись пошуковою системою. Наведемо послідовність програмний дій, що здійснюються в ході роботи системи, наприклад при введені слова для пошуку „самоучитель”:

1. Формується посилання */search?q=самоучитель.*

2. Засобами серверу перевіряється наявність файлу .htaccess, який містить додаткову конфігураційну інформацію.

3. У конфігураційному файлі .htaccess задається:

3.1. Властивість, що дає змогу перенаправляти запити на будь-які шляхи.

3.2. Шлях, який є корінним для всіх перенаправлень.

3.3. Файл із програмним скриптом, яким необхідно опрацювати сформований запит.

4. PHP-скрипт перевіряє чи змінна action дорівнює search, та використовує змінну q для пошуку за цим самим запитом.

5. Використовується функція MATCH() та AGAINST() для повнотекстового пошуку.

6. Результати пошуку представляються на формі у користувача.

Інтерфейс формування запиту до системи та видачі результатів її роботи представлено на *рис. 4.*

Для наведеного прикладу майже серед 50000 варіантів результат пошуку було отримано менш ніж за 0,1 с, що вказує на достатньо високий рівень швидкості обробки даних.

У розробленій пошуковій системі забезпечено функції адміністрування, а саме:

- перевірка бази даних на наявність помилок;
- налагодження індексатора (коли було в останній раз знайдено файл, коректність посилань);

- розширення або чистка списку веб-сайтів для сканування;
- оптимізація хостингу, пошукових запитів та бази даних.

Наведена інформаційно-пошукова система базується на пошуку за ключовими словами, що дозволяє шукати сторінки за їх змістом, формулюючи запит у вигляді набору ключових слів, а саме: ім'я файлу та заголовок сторінки. Теоретично, цього має бути на 99% достатньо для пошуку заданого документу в рамках файлообмінних сервісів, корпоративних інформаційних сховищ або інформаційних порталах.

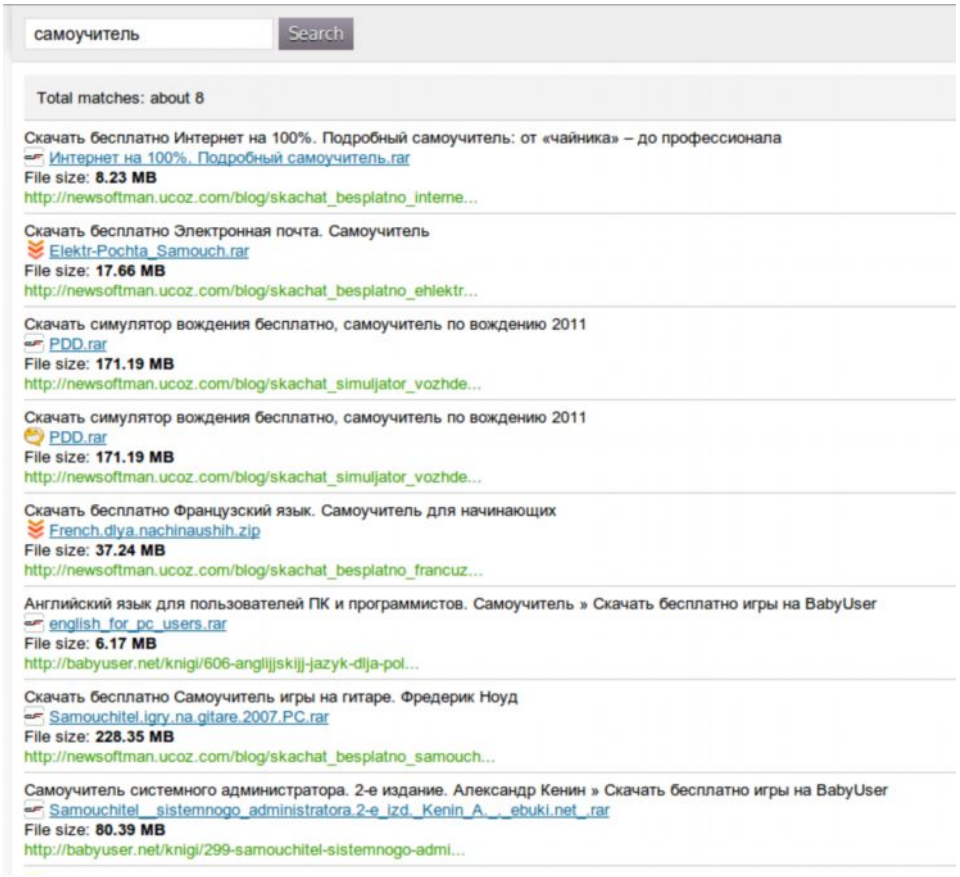

Рис. 4. Форма запиту та видачі результатів пошуку

Висновки

Проблема пошуку і збору інформації – одна з найважливіших проблем інформаційного середовища. Інтернет стає основною формою існування інформації. При появі мережі Інтернет проблема пошуку ставала більш актуальною. В роботі представлено результати розробки інформаційно-пошукової системи із використанням сучасних засобів веб-програмування, яка здійснює повнотекстовий пошук за ключовими словами. Обгрунтовано доцільність використання трирівневої архітектури при розробці системи, причому запропонований підхід відокремленої реалізації пошукового робота дозволив розділити обов'язки різних частин системи та звільнити сервер від навантажень індексатора. Описано пошукова модель, алгоритми індексації та послідовність програмних дій із системою. Розроблена інформаційно-пошукова система може бути ефективно використана для пошуку файлів у рамках корпоративних інформаційних сховищ, файлообмінних сервісах та інформаційних порталах.

ЛІТЕРАТУРА

1. Максимов Н. В. Информационные ресурсы и поисковые системы / Н. В. Максимов, О. Л. Голицына, Г. В. Тихомиров, П. Б. Храмцов, М : МИФИ, 2008. — 400 с.
2. Електронний ресурс: Sergey Brin and Larry Page. The Anatomy of a Search Engine http://www-db.stanford.edu/pub/papers/google.pdf .
3. Сегалович И. В. Как работают поисковые системы // Мир Internet. — 2002. — № 10.
4. Searching the Web. / A. Arasu, [and others] // ACM Trans. on Internet Technology. — 2001. — № 1 (1).

пост.08.10.13