Використання алгоритмів нечіткого пошуку для аналізатору трафіку

Кисляк О.І.
Науковий керівник – канд. техн. наук, доц., Цололо С.О.
Донецький національний технічний університет
(85300 м. Покровськ, вул. Шибанкова, 2, Донецька область, Україна)
e-mail: kissolya13@gmail.com

This work describes several ways of implementation fuzzy string search. Fuzzy algorithms advantages and disadvantages. Ways to improve characteristic features of existing algorithms. Nowadays, fuzzy string search is the most popular way to organize search in developing app. However, most of the specific requirements, that are obligatory for mobile app, cannot be reached and implemented due to only existing methods. This is a reason to find the most appropriate algorithm due to comparing, then investigate ways of its improvement: implement modifications, create indexed tables and save it in the data base.

Keywords: Fuzzy string search, Signature hashing, database, N-gram

Однією з основних задач реалізації аналізатору трафіку для ОС Android є імплементація алгоритму нечіткого пошуку. Аналізатор трафіку використовується для захисту корпоративних та особистих даних, саме тому за відсутності алгоритму зловмисник з легкістю зможе викрасти конфіденційні дані.

Нечіткий пошук (англ. *fuzzy string search*) – це пошук інформації, при якому виконується зіставлення інформації заданому зразку пошуку або близькому до зразку значенню. Використовується у багатьох галузях: для перевірки орфографії, у більшості сучасних пошукових систем та інше. Окрім цього його ефективна реалізація набагато складніше, ніж реалізація простого пошуку за точним збігом [1].

Існує велика кількість різноманітних алгоритмів нечіткого пошуку. Їх можна розділити на дві основні групи: з використанням індексації та без. Алгоритми без використання індексації призначені для пошуку по заздалегідь невідомому тексту, працюють з неперервним потоком даних та не потребують попередньої обробки тексту.

Особливістю алгоритмів з індексацією є те, що індекс будується за словником, що був складений за початковим текстом або за списком записів із будь-якої бази даних. До таких алгоритмів відносяться:

1. *Алгоритм розширення вибірки* – застосовується там, де розмір словника невеликий або швидкість не є основним критерієм. Основний недолік – алгоритм практичне непридатний для пошуку з максимально допустимою кількістю помилок [2];
2. *Алгоритми n-грамної індексації* – мають надзвичайно просту реалізацію. При порівнянні двох строк або слів, з урахуванням максимально можливої кількості розбіжностей, можна виділити у словах (строках) загальні частини. Після чого можна виконати пошук за цими частинами.

Однак, чим менше довжина слова та чим більше у ньому помилок, тим вище шанс, що воно не потрапить у списки, що відповідають N-грамам запиту. Має модифікацію, яка допомагає досягти більшої ефективності – розбиття хеш-таблиці N-грам за довжиною слова та за позицією N-грами у слові [2]. При наявності словників великих розмірів цей метод буде довго виконуватись та потребуватиме велику кількість ресурсів та потужностей, якими йго буде складно забезпечити при роботі з мобільними пристроями.

3. *BK-дерева* – дерева Burkhard-Keller, являють собою метричні дерева, алгоритми побудови таких дерев основані на властивості метрики відповідати нерівності трикутника [1]:

$$\rho(x,y) \le \rho(x,z) + \rho(z,y), \qquad x,y,z \in X$$

Цей метод дозволяє зменшити час виконання пошуку за рахунок більш зручної структури даних, але ускладнює алгоритм пошуку.

4. *Хешування за сигнатурою* – базується на достатньо очевидному представленні «структури» слова у вигляді бітових розрядів, що використовується у якості хешу (сигнатури) у хеш-таблиці [3]. Процес обчислення хешу – кожному біту хешу зіставляється група символів з алфавіту. Біт 1 на позиції *i* в хеші позначає наявність символу з *i*-ої групи алфавіту у початковому слові. Порядок букв у слові не має ніякого значення. Поточний метод потребує значних часових затрат на складання хеш-таблиці, що також ускладнює його використання у мобільному додатку. Однак цей метод достатньо простий при реалізації та дозволяє виконувати швидкий пошук, як по повному збігу слова, так і при наявності однієї або двох помилок у слові. Проблема часових затрат може бути вирішена шляхом створення готових індексів за текстами або словниками.

Ключовими критеріями для додатку, що розроблюється, є час виконання пошуку, а також необхідний обсяг пам’яті, що потребується для обробки даних. Кожний з алгоритмів, що було розглянуто, має ці недоліки. У результаті порівняння було вибрано метод хешування за сигнатурою. Внесення додаткових умов та комбінація з іншими алгоритмами допоможе досягти бажаного результату.

Список використаних джерел:

1. Бойцов Л.М. Использование хеширования по сигнатуре для поиска по сходству. Прикладная математика и информатика / Л.М. Бойцов. – М.: Изд-во факультета ВМиК, МГУ, 2000. – № 7.
2. Классификация и экспериментальное исследование современных алгоритмов нечеткого словарного поиска. Бойцов Л.М. [електронний ресурс]. – Режим доступу: http://rcdl.ru/doc/2004/paper27.pdf
3. Нечеткий поиск в тексте и словаре [електронний ресурс]. – Режим доступу: https://habrahabr.ru/post/114997/