Київський національний університет імені Тараса Шевченка

факультет комп’ютерних наук та кібернетики

кафедра інформаційних систем

Лабораторна робота №8

Алгоритми пошуку зразка в текстовому рядку

Виконав студент 2 курсу

Групи К-28

Мосьпан Олег Олександрович

2018

Постановка задачі

Реалізувати алгоритми пошуку зразка в текстовому рядку: наївний, Хорспула, Боєра-Мура, КПМ та Рабіна-Карпа і порівняйти їх ефективність. Виконати пошук зразків різної довжини: випадкового бінарного зразка у випадковому бінарному тексті та випадкового слова у природному тексті на цій мові.

Опис алгоритму

Нехай є два рядка a і b довжиною n і m відповідно (n <= m). Необхідно перевірити чи входить a в b.

Наївний алгоритм

Перебираються всі позиції рядка b, в яких може починатися рядок a. Для всіх відповідних позицій i посимвольно порівнюються рядки b[i..i + n – 1] і a. Якщо всі символи співпадають, то ми знайшли входження.

Складність T(n, m) = O(m n).

Алгоритм Хорспула

Спочатку будується таблиця зміщень для шуканого шаблону. Поєднується початок тексту (рядки) і шаблона, перевірка починається з останнього символу шаблону.

Якщо останній символ шаблону і відповідний йому при накладенні символ рядка не збігаються, то зразок зрушується щодо рядка на величину, отриману з таблиці зміщень. Причому символ береться з рядка (а не з шаблону), відповідний зсув знаходиться в таблиці. Проводиться зрушення і знову починається перевірка з останнього символу.

Якщо ж символи збігаються, проводиться порівняння передостаннього символу шаблону і т. д. Якщо всі символи шаблону збіглися з накладеними символами рядки, значить, підрядок знайдений, і пошук закінчений. Якщо ж якийсь (не останній) символ шаблону не збігається з відповідним символом рядка, шаблон зсувається на один символ вправо, і перевірка знову починається з останнього символу. Весь алгоритм виконується до тих пір, поки або не буде знайдено входження шуканого зразка, або не буде досягнуто кінець рядка.

Складність в середньому T(n, m) = (m).

В найгіршому випадку - O(m n).

Алгоритм Бойєра-Мура

Алгоритм порівнює символи шаблону a справа наліво, починаючи з самого правого, один за іншим з символами рядка b. Якщо символи збігаються, проводиться порівняння передостаннього символу шаблону і так до кінця. Якщо всі символи шаблону збіглися з накладеними символами рядка, значить, подрядок знайдений, і пошук закінчено. Що стосується розбіжності будь-якого символу (або повного збігу всього шаблону) алгоритм використовує дві, попередньо обчислені евристичні функції, щоб зрушити позицію для початку порівняння вправо.

Таким чином для зсуву позиції початку порівняння алгоритм Бойера-Мура вибирає між двома функціями, званими евристиками хорошого суфікса і поганого символу (іноді вони називаються евристиками співпалого суфікса і стоп-символу). Так як функції евристичні, то вибір між ними простий - шукається таке підсумкове значення, щоб ми не перевіряли максимальне число позицій і при цьому знайшли всі подрядки рівні шаблону.

Нехай |a| = n, |b| = m. Припустимо, що в процесі порівняння виникає розбіжність між символом a[i] = x шаблону і символом b[i + j] = y вихідного тексту при перевірці в позиції j. Тоді a[i + 1 .. m - 1] = b[i + j + 1 .. j + m-1] = u і a[i] ≠ b[i + j], та m – i - 1 символів шаблону вже збіглося.

Правило зсуву хорошого суфікса:

Якщо при порівнянні тексту і шаблону збіглося один або більше символів, шаблон зсувається в залежності від того, який суфікс збігся.

Якщо існують такі підрядки рівні u, що повністю входять в a і йдуть праворуч від символів, відмінних від a[i], то зсув відбувається до найправішої з них, відмінної від u. Зрозуміло, що таким чином ми не пропустимо жоден рядок, так як зсув проходить на наступний зліва підрядок u від суфікса. Після вирівнювання шаблону по цьому підрядку порівняння шаблону знову почнеться з його останнього символу. На новому етапі алгоритму можна рядок u, по якому був проведений зсув, не порівнювати з текстом.

Якщо не існує таких подрядків, то зміщення складається у вирівнюванні найдовшого суфікса v підрядка b[i + j + 1 .. j + m-1] з відповідним префіксом a. Через те, що ми не змогли знайти такий підрядок, то, очевидно, що жоден суфікс шаблону a вже не буде лежати в підрядку b[i + j + 1 .. j + m - 1], тому єдиний варіант, що в цей підрядок потрапить префікс.

Правило зсуву поганого символу:

У таблиці поганих символів вказується остання позиція в шаблоні (виключаючи останню букву) кожного з символів алфавіту. Для всіх символів, що не увійшли в шаблон, вважаємо це значення рівним m. Припустимо, що у нас не збігся символ z з тексту на черговому кроці з символом з шаблону. Очевидно, що в такому випадку ми можемо зрушити шаблон до першого входження цього символу z в шаблоні, тому що збігів інших символів точно не може бути. Якщо в шаблоні такого символу немає, то можна зрушити весь шаблон повністю.

Якщо символ вихідного тексту b[i + j] зустрічається в шаблоні a, то відбувається його вирівнювання з його найправішим з’явленням в підрядку a[0 ... m - 2].

Якщо b[i + j] не зустрічається в шаблоні a, то жодне входження x в y не може включати в себе b[i + j], і лівий кінець вікна порівняння суміщений з символом безпосередньо йде після b[i + j], тобто символ b[i + j + 1].

Складність T(n, m) = O(m).

Алгоритм Кнута — Морріса — Пратта

Введемо поняття префікс-функції рядка s[0..n-1]. Префікс-функція для s - це масив p[0..n - 1], де p[i] визначається наступним чином: це така найбільша довжина найбільшого власного суфікса підрядка s[0..n - 1], що збігається з її префіксом. Зокрема, значення p[0] вважається рівним нулю.

Існує лінійний алгоритм обчислення префікс-функції. Обчислювати значення префікс-функції p[i] будемо по черзі: від i = 1 до i = n-1 (p[0] = 0).

Для підрахунку поточного значення p[i] ми заводимо змінну j, що позначає довжину поточного розглянутого зразка. Спочатку j = p[i-1].

Тестуємо зразок довжини j, для чого порівнюємо символи s[j] і s[i]. Якщо вони збігаються - то вважаємо p[i] = j + 1 і переходимо до наступного індексу i + 1. Якщо ж символи відрізняються, то зменшуємо довжину j, вважаючи її рівною p[j-1], і повторюємо цей крок алгоритму з початку.

Якщо ми дійшли до довжини j = 0 і так і не знайшли збіги, то зупиняємо процес перебору зразків, вважаємо p[i] = 0 і переходимо до наступного індексу i + 1.

Алгоритм КМП полягає в наступному. Спочатку треба обчислити префікс-функцію для рядка a + “#” + b. Можна побачити, що, якщо в якійсь позиції i=n+1..n+m виявилося p[i] = n, то в позиції i - (n + 1) - n + 1 = i - 2n рядка b починається чергове входження рядка a в рядок b. Тому, якщо існує p[i] = n, то рядок a входить в рядок b.

Складність T(n, m) = O(m).

Алгоритм Рабіна-Карпа

Введемо поняття хеш-функції. Один з кращих способів визначити хеш-функцію від рядка s[0..n-1] наступний:

h(s) = s[0] + s[1] \* P + s[2] \* P^2 + s[3] \* P^3 + ... + s[n] \* P^n

де P - деяке число.

Розумно вибирати для P просте число, яке приблизно дорівнює кількості символів у вхідному алфавіті. Наприклад, якщо рядки складаються тільки з маленьких латинських літер, то хорошим вибором буде P = 31.

Припустимо, нам дано рядок s та індекси i та j. Потрібно знайти хеш від підрядка s[i..j].

За визначенням маємо:

h[i..j] = s[i] + s[i + 1] \* P + s[i + 2] \* P^2 + ... + s[j] \* P^(j - i),

звідки:

h[i..j] \* p[i] = s[i] \* p[i] + ... + s[j] \* p[j],

h[i..j] \* P [i] = h[0..j] - h[0..i - 1],

де p[k] = P^k.

Знаючи тільки хеші від всіх префіксів рядка S, ми можемо за O(1) отримати хеш будь-якого підрядка. У більшості випадків, замість того щоб ділити хеші на степені P, можна, навпаки, множити їх на ці степені. Припустимо, дано два хеші: один помножений на p[i], а інший - на p[j]. Якщо i < j, то помножимо перший хеш на p[j - i], інакше помножимо другий хеш на p[i - j]. Тепер ми привели хеші до одного степеня і можемо їх порівнювати.

В алгоритмі Рабіна-Карпа обчислюються хеш-функції для рядків a і b. Як і в наївному алгоритмі перебираються всі позиції рядка b, в яких може починатися рядок a. Для всіх відповідних позицій i порівнюються рядки b[i..i + n – 1] і a. Але спочатку порівнюються їхні хеші. Лише якщо вони рівні, то виконується посимвольна перевірка рядків. Якщо всі символи співпадають, то ми знайшли входження.

Складність в середньому T(n, m) = (m).

В найгіршому випадку - O(m n).

Інтерфейс

Назва вхідного файлу вводиться з клавіатури.

**Вхідні дані:** Два рядки a і b розмірами n і m відповідно (1 <= n <= m <= 10^5).

**Вихідні дані:** “YES” – рядок a є підрядком b, “NO” - інакше.

Тести

Вхідні дані 1:

cabc

abcdabcabcdabcdab

Вихідні дані 1:

YES

Вхідні дані 2:

dabd

abcdabcdab

Вихідні дані 2:

NO

Вхідні дані 3:

abbad

abeccacbadbabbad

Вихідні дані 3:

YES

Вхідні дані 4:

avbcvbcaba

hjkajhsfavbcvbcabajhkasd

Вихідні дані 4:

YES

Список використаних джерел

1. <https://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%91%D0%BE%D0%B9%D1%94%D1%80%D0%B0_%E2%80%94_%D0%9C%D1%83%D1%80%D0%B0_%E2%80%94_%D0%A5%D0%BE%D1%80%D1%81%D0%BF%D1%83%D0%BB%D0%B0>
2. <https://neerc.ifmo.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%91%D0%BE%D0%B9%D0%B5%D1%80%D0%B0-%D0%9C%D1%83%D1%80%D0%B0>
3. <http://e-maxx.ru/algo/prefix_function>
4. <http://e-maxx.ru/algo/rabin_karp>
5. <http://e-maxx.ru/algo/string_hashes>