МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5 (Вариант 2)
по дисциплине «Построение и анализ алгоритмов»
Тема: Поиск набора подстрок в строке. (Ахо-Корасик).

Студент гр. 3388 _____ Лутфулин Д.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2025

Цель работы.

Реализовать алгоритм Ахо-Корасик для нахождения набора подстрок в строке. Найти вхождения шаблона с джокером.

Задание.

Ахо-Корасик

Разработайте программу, решающую задачу точного поиска набора образцов.

Вход:

Первая строка содержит текст (T,1≤|T|≤100000).

Вторая - число n 1≤n≤3000 каждая следующая из n строк содержит шаблон из набор P={p1,…,pn}1≤|p|≤75

Все строки содержат символы из алфавита {A,C,G,T,N }

Выход:

Все вхождения образцов из P в T. Каждое вхождение образца в текст представить в виде двух чисел - i p, где i - позиция в тексте (нумерация начинается с 1), с которой начинается вхождение образца с номером p (нумерация образцов начинается с 1). Строки выхода должны быть отсортированы по возрастанию, сначала номера позиции, затем номера шаблона.

Sample Input:

NTAG

3

TAGT

TAG

T

Sample Output:

2 2

2 3

Поиск образца с джокером.

Используя реализацию точного множественного поиска, решите задачу точного поиска для одного образца с *джокером*.
В шаблоне встречается специальный символ, именуемый джокером (wild card), который "совпадает" с любым символом. По заданному содержащему шаблоны образцу P необходимо найти все вхождения P в текст T. Например, образец ab??c? с джокером ? встречается дважды в тексте xabvccbababcax.

Символ джокер не входит в алфавит, символы которого используются в TT. Каждый джокер соответствует одному символу, а не подстроке неопределённой длины. В шаблон входит хотя бы один символ не джокер, т.е. шаблоны вида ??? недопустимы.

Все строки содержат символы из алфавита {A,C,G,T,N}

Вход:

Текст (T,1≤|T|≤100000)

Шаблон (P,1≤|P|≤40)

Символ джокера

Выход:

Строки с номерами позиций вхождений шаблона (каждая строка содержит только один номер).

Номера должны выводиться в порядке возрастания.

Sample Input:

ACTANCA

A$$A$

$

Sample Output:

1

Выполнение работы.

Ахо-Корасик

Для решения задачи был использован алгоритм Ахо-Корасик. Из данного набора образцов строится бор – дерево, рёбра которого отвечают за переход по какой-либо букве. Вершины, полученные переходом по буквам из одного из образцов, помечаются как терминальные и хранят в себе индекс этого образца в массиве. Если на текущем этапе прохода по бору не существует ребра, соответствующего символу, по которому мы хотим перейти, то выполняется переход по суффиксной ссылке – мы отправляемся на ту вершину, путь в которую является суффиксом пути в текущую. Например, из hers может быть суффиксная ссылка в ers или rs.Переходы по суффиксным ссылкам выполняются, пока не дойдём до корня или не найдем подходящий суффикс.

Также используются конечные(терминальные) ссылки. Это ближайшая суффиксная ссылка(или суфф. ссылка суфф. ссылки и т.д. до корня), которая ведёт в терминальную вершину. Если при переходе по букве есть терминальный суффикс, то мы идем по этой цепочке терминальных суффиксов и записываем в ответ. Например, в (корень-b-u-r-g-e) переходим r и получилось так, что ger и r– одни из искомые подстрок. Тогда будет выполнен переход по терминальной ссылке в вершину (корень-g-e-r) и оттуда в (корень-r), индексы будут записаны в ответ.

Для подсчёта количества вершин используется рекурсивная функция, вызываемая на корне и возвращающая 1 + результат функции на потомках вершины.

Для нахождения пересекающихся шаблонов выполняется проход по массиву вхождений подстрок. Если индекс текущей подстроки + её длина больше индекса следующей подстроки, то у них есть пересечение

Поиск образца с джокером.

Искомая подстрока разбивается по джокеру на подстроки и выполняется стандартный алгоритм Ахо-Корасик по этому набору подстрок. Но в терминальных вершинах храним пару (подстрока, смещение относительно начала подстроки с джокером). По этому смещению выясняем индекс начала паттерна с джокером, в котором должна оказаться данная подстрока, и её индекс

записывается в словарь по ключу в виде индекса начала паттерна. Если в результате работы алгоритма для какого-то индекса этого словаря собрались все подстроки паттерна с джокером, то мы нашли вхождение.

Оценка сложности.

m – суммарная длина образцов

t – размер алфавита

n – длина текста

Для хранения рёбер используется python-словарь, так что переход занимает O(1), а создание O(t) т.к. словарь основан на хеш-таблице.При построении суффиксного дерева мы за O(mt) строим вершины. Далее нужно расставить суффиксные ссылки. Для вершины глубиной k для установки суффиксной ссылки потребуется не более чем k-1 переходов по суффиксной ссылке (в худшем случае мы каждый раз будем переходить в поддерево высотой на 1 меньше. Но, даже если считать, что с переходов по суффиксным ссылкам из вершины-родителя поднимали нас только на 1 выше, то суффиксная ссылка теперь будет отправлять нас в поддерево высотой (k-c) за один переход. То есть, чем больше переходов мы потратили для нахождения ссылки для родителя, тем сильнее мы будем «срезать дорогу» переходя по ней из детей. Итого количество таких переходов для каждой терминальной вершины будет равно её глубине, а значит всего таких переходов O(m). Тогда сложность построения бора O(mt).

На каждом шаге алгоритма мы находимся в вершине бора, соответствующей последним k символам из текста. Если мы переходим по символу, то становимся в вершине с k+1 символами, а если переходим по суффиксной ссылке, то глубина поддерева уменьшится.

За всё время работы алгоритма нельзя подняться выше по дереву более чем n раз. Значит, переходов по суфф.ссылкам O(n). А так как остальные операции – переходы по рёбрам за O(1), суммарная сложность будет O(n)

Итоговая сложность O(mt + n)

Вывод.

Был написан алгоритм Ахо-Корасик для нахождения набора подстрок и шаблона с джокером, проанализирована сложность.