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

ОТЧЕТ

по лабораторной работе №5
по дисциплине «Построение и анализ алгоритмов»
Тема: Алгоритм Ахо-Корасик

Студент гр. 7383 _____ Васильев А.И.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2019

Цель работы.

Исследовать и реализовывать задачу поиска набора образцов в строке, используя алгоритм Ахо-Корасик.

Формулировка задачи: необходимо разработать программу, решающую задачу точного поиска набора образцов.

Входные данные: Первая строка содержит текст (T,$1 \leq |T| \leq 100000$). Вторая – число n($1 \leq n \leq 3000$), каждая следующая из n строк содержит шаблон из набора P={p1,…,pn}$1 \leq |pi| \leq 75$. Все строки содержат символы из алфавита {A,C,G,T,N}.

Выходные данные: 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.

Также следует разработать программу для решения следующей задачи: реализовать точный поиск для одного образца с джокером.

Входные данные: в первой строке указывается текст (T,$1 \leq |T| \leq 100000$), в котором ищем подстроки, а во второй шаблон (P,$1 \leq |P| \leq 40$), затем идёт символ джокера.

Выходные данные: строки с номерами позиций вхождений шаблона (каждая строка содержит только один номер).

Реализация задачи.

Для решения поставленной задачи были написаны главная функция `main()` и класс `Aho_Korasik`, а также структура `bohr_vertex`, которая используется для реализации бора. Бор – это дерево, в котором каждая вершина обозначает какую-то строку (корень обозначает нулевую строку – ε). На ребрах между вершинами написана одна буква, таким образом, добираясь по ребрам из корня в какую-нибудь вершину и контангенируя буквы из ребер в порядке обхода, получаем строку, соответствующую этой вершине. Из определения

бора как дерева вытекает также единственность пути между корнем и любой вершиной, следовательно – каждой вершине соответствует ровно одна строка.

Поля структуры bohr_vertex:

check – флаг, показывающий является ли вершина конечной;

sample – номер образца;

suffix_link – суффиксная ссылка v, указатель на вершину u, такую что строка u – наибольший собственный суффикс строки v, или, если такой вершины нет в боре, то указатель на корень;

previous_vertex – индекс родителя, индекс вершины, из которой пришли в текущую;

curr_vertex – значение ребра от родителя к текущей вершине;

edge – упорядоченный ассоциативный массив типа map, задающий соседей текущей вершины;

auto_move – переходы автомата;

bohr_vertex(int previous, char vertex) – конструктор, где previous – предок вершины, vertex – значение ребра.

Параметры, хранящиеся в классе:

tree – бор, представляемый в виде вектора структур bohr_vertex;

result – набор строк-шаблонов, хранящихся в векторе;

number_of_tree_nodes – количество узлов дерева, используется как индекс для заполнения бора.

Методы класса:

Aho_Korasik – конструктор по умолчанию, используется для создания корня дерева;

add_pattern – добавление нового образца в набор шаблонов;

get_suffix_link – получение суффиксальной ссылки для заданной вершины;

get_auto_move – выполнение автоматного перехода;

`algorithm` – алгоритм Ахо-Корасик, выводит все вхождения образцов из P в T;

В функции `main()` считывается текст и `n` шаблонов. Заполняется бор, а затем запускается поиск всех вхождений образцов из P в T.

Тестирование.

Тестовые случаи не выявили не правильного поведения программы, следовательно можно судить о том, что поставленная задача была выполнена. Тестовые случаи приведены в табл. 1.

Таблица 1 – Тестовые случаи

Входные данные	Вывод	Верно?
abbaaaba 6 aba ba abba baab bab b	2 6 3 6 1 3 3 2 7 6 6 1 7 2	Да
CCCA 1 CC	11 2 1	Да
ACT A$ $	1	Да
ACTATTCATC A?T ?	1 4	Да

Исследование.

Сложность реализованного алгоритма Ахо-Корасик с использованием структуры данных map – O((|T| + |P|) * log|A| + c), где T – длина текста, в котором осуществляется поиск, |P| – общая длина всех слов в словаре, |A| – размер алфавита и c – общая длина всех совпадений.

По памяти сложность алгоритма составляет O(|T| + |P|), так как память выделяется для всего текста и для вершин шаблонов.

Выводы.

В ходе лабораторной работы был изучен метод поиска подстрок в строке, используя алгоритм Ахо-Корасик. Был написан код на языке программирования C++, который применял этот способ для поставленной задачи. Сложность по количеству просмотренных вершин равна O((|T| + |P|) * log|A| + c), а по памяти – O(|T| + |P|).