

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

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

Студентка гр. 7304

Нгуен Т.Т. Зуен

Преподаватель

Филатов А.Ю.

Санкт-Петербург

2019

Цель работы:

Исследование алгоритма Ахо-Корасик и его реализация на языке C++.

Задание 1:

Разработайте программу, решающую задачу точного поиска набора образцов.

Вход:

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 $P = \{p_1, \dots, p_n\}$ $1 \leq |p_i| \leq 75$

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 и 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.

Задание 2:

Используя реализацию точного множественного поиска, решите задачу точного поиска для одного образца с джокером.

В шаблоне встречается специальный символ, именуемого 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 $xabvccbababсах$.

Символ джокер не входит в алфавит, символы которого используются в T . Каждый джокер соответствует одному символу, а не подстроке неопределенной длины. В шаблоне входит хотя бы один символ не джокер, те шаблоны вида $???$ недопустимы.

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

Вход:

Текст ($T, 1 \leq |T| \leq 100000$)

Шаблон ($P, 1 \leq |P| \leq 40$)

Символ джокера

Выход:

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

Номера должны выводиться в порядке возрастания.

Описание алгоритма

Алгоритм Ахо - Корасик — алгоритм создает конечный автомат для одновременного поиска подстроки.

Алгоритм строит конечный автомат, которому затем передаёт строку поиска. Автомат получает по очереди все символы строки и переходит по соответствующим рёбрам. Если автомат пришёл в конечное состояние, соответствующая строка словаря присутствует в строке поиска.

Описание функций:

- Структур `struct result {int id; int num; };` содержаемый результат поиска, где *id* - позиция в тексте (нумерация начинается с 1), с которой начинается вхождение образца с номером *num* (нумерация образцов начинается с 1).

`g[][];` trie всех слов, где сохраняем следующее состояние для текущего состояния *state* и символа.

`f[];` failure, содержаемый все ребра, которые следуют, когда текущий символ не имеет ребра в Trie.

`out[];` содержаемый индексы всех слов, которые заканчиваются в текущем состоянии.

- `int buildMatching(vector<string> str)` строение машину сопоставления строк .

- `int findNextState(int cur_state, char nextInput)` возвращение следующее состояние *cur_state*, в которое машина перейдет с использованием Trie (*g*) и failure (*f*).

- `void searchWords(vector<string> str, string text, vector<result*> &res)` поиска все вхождения всех слов *str* в тексте и сохранение результатов в векторе *res*.

На втором задании: поиска для одного образца с *джокером*. В шаблоне встречается специальный символ, именуемого джокером (wild card), который "совпадает" с любым символом.

- `bool check(string text, vector<string> &str, vector<int> &j_len, vector<result*> &res, int cur_res, int depth, int cur_pos)`

`str` - вектор, содержаемый подстроки разделены джокером (например A??AC? – A, AC).

`j_len` – вектор, содержаемый количества джокеров соответственно в соответствии с каждым интервалом (например A??AC? – 2,1).

`cur_res` - текущее позиция поиска в векторе `res`

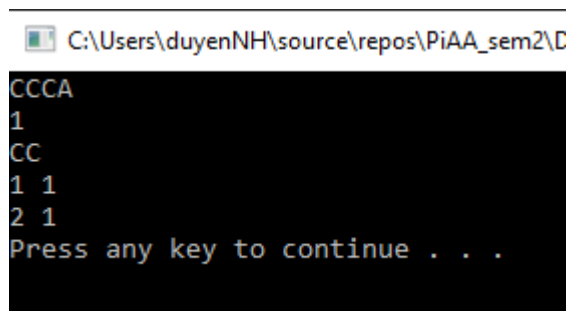
`depth` - соответствует номеру конденсатора образца

`cur_pos` - текущая позиция поиска в тексте

Мы ищем позицию подстроки в тексте, а затем полагаемся на результат и используем рекурсив, чтобы найти позицию шаблона в тексте.

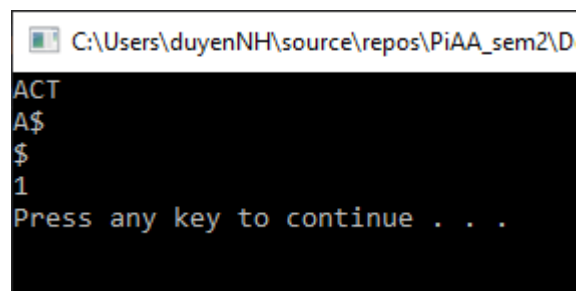
Результаты:

1.

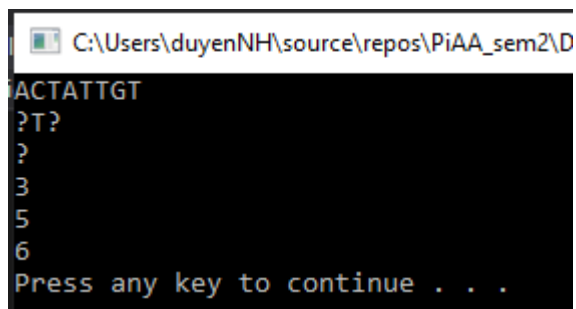


```
C:\Users\duyenNH\source\repos\PiAA_sem2\D
CCA
1
CC
1 1
2 1
Press any key to continue . . .
```

2.



```
C:\Users\duyenNH\source\repos\PiAA_sem2\D
ACT
A$
$
1
Press any key to continue . . .
```



```
C:\Users\duyenNH\source\repos\PiAA_sem2\D
ACTATTGT
?T?
?
3
5
6
Press any key to continue . . .
```

Выводы:

В результате работы программы была реализован алгоритм Ахо - Корасик поиска набора шаблонов в тексте и использовала его для решения задач.

Алгоритм Ахо – Корасик очень удобно и быстро найти позиции в тексте (нумерация начинается с 1), с которой начинается вхождение образца. В этом случае это заметно быстрее, чем алгоритм КМР.