

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

Студент гр. 7304

Нгуен К.Х.

Преподаватель

Филатов А.Ю.

Санкт-Петербург

2019

Цель работы.

Исследование алгоритма Форда - Фалкерсона, реализация программы для решения задач с использованием этого алгоритма

Задание

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, 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 шаблону образцу (P) необходимо найти все вхождения (P) в текст (T) .

Например, образец $(ab??c?)$ с джокером $(?)$ встречается дважды в тексте $(xabvccbababcsax)$.

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.

Ход работы

Алгоритм Ахо-Корасик создает конечный автомат для одновременного поиска нескольких строк.

Итак, во-первых, мне нужно было создать класс State для хранения информации о состоянии в автомате состояний, он будет иметь идентификатор, карту для следующих состояний, вектор слов, оканчивающихся на это состояние, указатель на состояние, которое будет использоваться в случае неудачного перехода

Кроме того, хотя и не обязательно, я создал класс `SearchResult`, чтобы упростить хранение результатов поиска.

Шаг 1 алгоритма, включающий создание дерева поиска, поэтому я создал функцию `addWord` для добавления слова к дереву поиска буква за буквой, оно будет идти от корня к следующему узлу, если возможно, в противном случае он создает новый узел. Функция `constructGoto` принимает набор ключевых слов и добавляет их в дерево по одному. В результате мы имеем дерево поиска.

Шаг 2 алгоритма состоит в создании функции отказа и набора выходных данных. функция `constructFailFunction` будет проходить через уровень дерева за уровнем (как при поиске в ширину, используя очередь) и заполнять свойство `failState` для каждого узла, а также вектор `output`.

Функция `AhoCorasick` использует созданный конечный автомат для поиска ключевых слов в тексте.

Я использовал большую часть кода из задачи 1 в задаче 2.

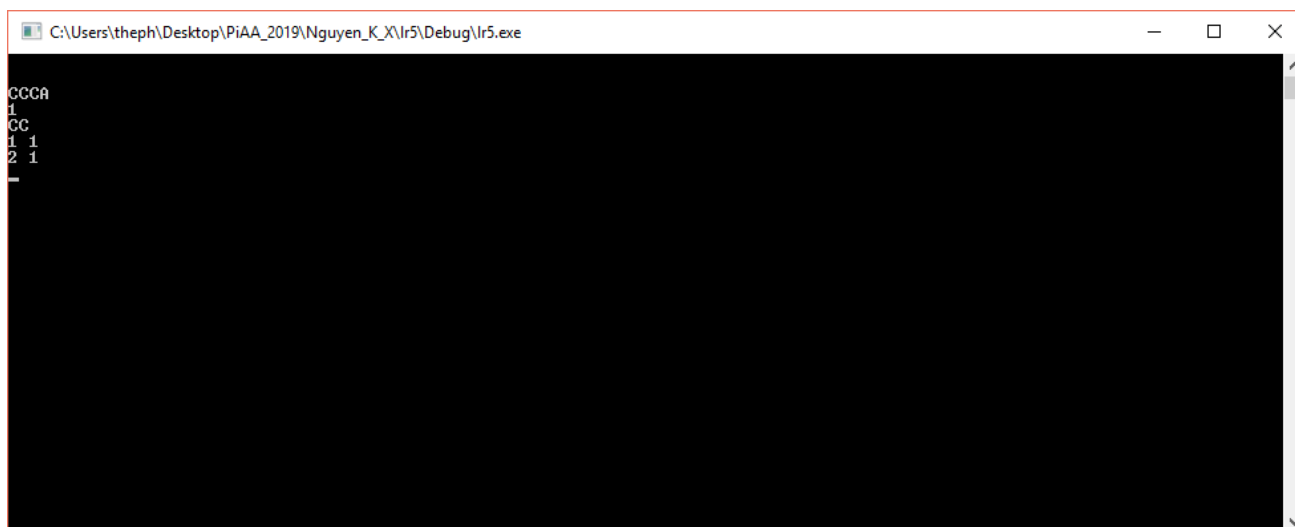
В дополнение к существующему коду я создал функцию `buildDictionary`, которая принимает шаблон ввода и разбивает его на части. Например, шаблон `аас ?? ас? С с джокером -?` будет разделен на 3 шаблона `аас`, `ас` и `с`. В результате получается набор шаблонов и набор `int`, хранящий длины джокеров, стоящих между шаблонами.

Набор шаблонов помещается в функцию из первого задания.

После получения результатов я написал функцию `'match`, чтобы проверить набор результатов и набор длин, полученных ранее, чтобы найти совпадение исходного шаблона.

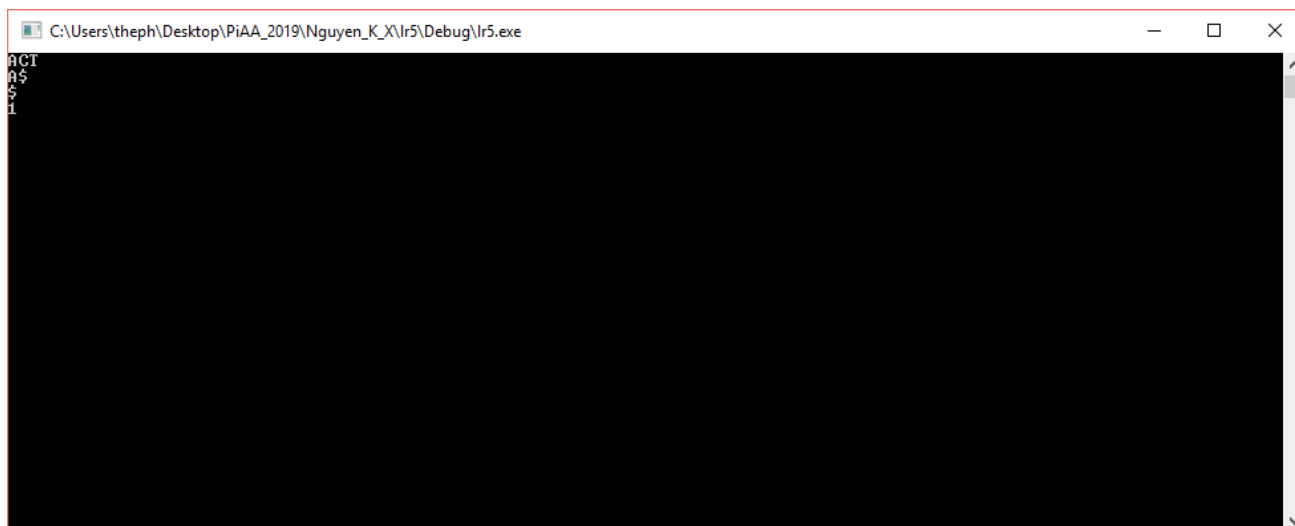
Экспериментальные результаты.

Задание 1



```
C:\Users\theph\Desktop\PiAA_2019\Nguyen_K_X\lr5\Debug\lr5.exe
CCCA
1
CC
1 1
2 1
-
```

Задание 2



```
C:\Users\theph\Desktop\PiAA_2019\Nguyen_K_X\lr5\Debug\lr5.exe
ACT
AS
S
1
```

Выводы.

В результате лабораторной работы я изучил алгоритм ахо-кюразика для поиска набора шаблонов в тексте и использовал его для решения задач.

Алгоритм aho-curasick очень полезен в ситуациях, когда вам нужно найти не 1, а набор шаблонов в тексте. В этом случае это заметно быстрее, чем многократно использовать алгоритм КМР.