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

ОТЧЕТ

по лабораторной работе №5 (Вар. 1)
по дисциплине «Построение и анализ алгоритмов»
Тема: Ахо-Корасик

Студентrf гр. 3388 _____ Беннер В.А.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2025

Задание (Вариант 1. Выполнение на Stepik двух заданий в разделе 2)

Вход:
Первая строка содержит текст T (1< |T|< 100000).

Вторая строка содержит число n (1< n< 3000). Каждая следующая из n строк содержит шаблон из набора P = { p_1, ldots, p_n } (1< |p_i|< 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 по номеру позиции, затем по номеру шаблона.

Задача:

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?c) с джокером ? встречается дважды в тексте *zabuccbababcax*.

Символ джокер не входит в алфавит, символы которого используются в (T). 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.

Выполнение работы

Для выполнения поиска подстрок в тексте используется алгоритм, основанный на построении бора (префиксного дерева) и применении суффиксных и хороших ссылок.

Построение бора:

- В бор добавляются все подстроки, полученные из шаблона. Сложность добавления подстрок в бор составляет O(M), где *M* — суммарная длина исходного шаблона. Каждый узел содержит словарь next размером *k* (размер алфавита).

Построение суффиксных и хороших ссылок:

- Сложность вычисления суффиксных и хороших ссылок составляет O(M).

Поиск в тексте:

- Для поиска подстрок в тексте длиной *N*, алгоритм проходит по тексту. На каждом шаге происходит переход по бору и, возможно, проход по "хорошим" ссылкам. Сложность поиска в тексте составляет O(N * M), где *N* — длина текста, *M* — суммарная длина исходного шаблона

Итоговая сложность:

- Общая временная сложность алгоритма: O(M + N * M).

Память:

- Пространственная сложность алгоритма: O(M*k), где *k* — размер алфавита.

Тестирование:

Input	Output
NTAG 3 TAGT TAG T	2 2 2 3
ACACACG 2 AC CAC	1 1 2 2 3 1 4 2 5 1

Input	Output
ACTANCA A$$A$ $	1
AGGGGTN A%%%GT% %	1

Input	Output
ACTANCA A$$A$ $ T	
AGGGGTN A%%%GT% % A	1

Выводы:

В ходе работы был разработан и протестирован алгоритм для поиска вхождений шаблона с джокером (или без него, с ограниченным джокером) в тексте. Алгоритм использует автомат Ахо-Корасик, реализованный на основе бора, для эффективного поиска подстрок. "Хорошие" ссылки используются для оптимизации поиска.