Лабораторная работа № 4 по курсу дискретного анализа: Строковые алгоритмы

Выполнила студентка группы 08-208 МАИ *Шевлякова София.*

Условие

Необходимо реализовать один из стандартных алгоритмов поиска образцов для указанного алфавита.
Вариант алгоритма: поиск одного образца-маски: в образце может встречаться «джокер» (представляется символом ? — знак вопроса), равный любому другому символу. При реализации следует разбить образец на несколько, не содержащих «джокеров», найти все вхождения при помощи алгоритма Ахо-Корасик и проверить их относительное месторасположение.
Вариант алфавита: слова не более 16 знаков латинского алфавита (регистронезависимые).

Метод решения

Поиск одного образца с джокерами сводится к множественному поиску с помощью алгоритма Ахо-Корасик с некоторой модификацией. В дерево нужно поместить не один образец, а все максимальные подстроки образца, не содержащие джокера. Затем построить для каждой вершины дерева связи неудач, обойдя дерево в ширину. Препроцессинг для построения связей неудач выполняется за O(n), где n — суммарная длина всех подстрок образца без джокеров.
Помимо всего прочего при построении дерева нужно найти длины подстрок образца, не содержащих джокеров, и их позиции начала в образце — L_i. Это понадобится в дальнейшем при поиске образца в тексте.
Непосредственно при поиске нужно пройтись по тексту и найти для каждой подстроки без джокера начальную позицию вхождения в текст — j. Для каждого такого j увеличиваем счетчик в ячейке j - L_i + 1 массива C на единицу. Массив C имеет длину текста и изначально инициализирован нулями. После окончания прохода текста, просматриваем массив C в поисках ячеек со значением k (число подстрок без джокеров). Вхождение паттерна в текст с позиции p, будет только в том случае, если C[p] = k.

Описание программы

В данной программе содержится один файл main.cpp, в котором реализован алгоритм Ахо-Корасик.
Основные этапы работы программы:

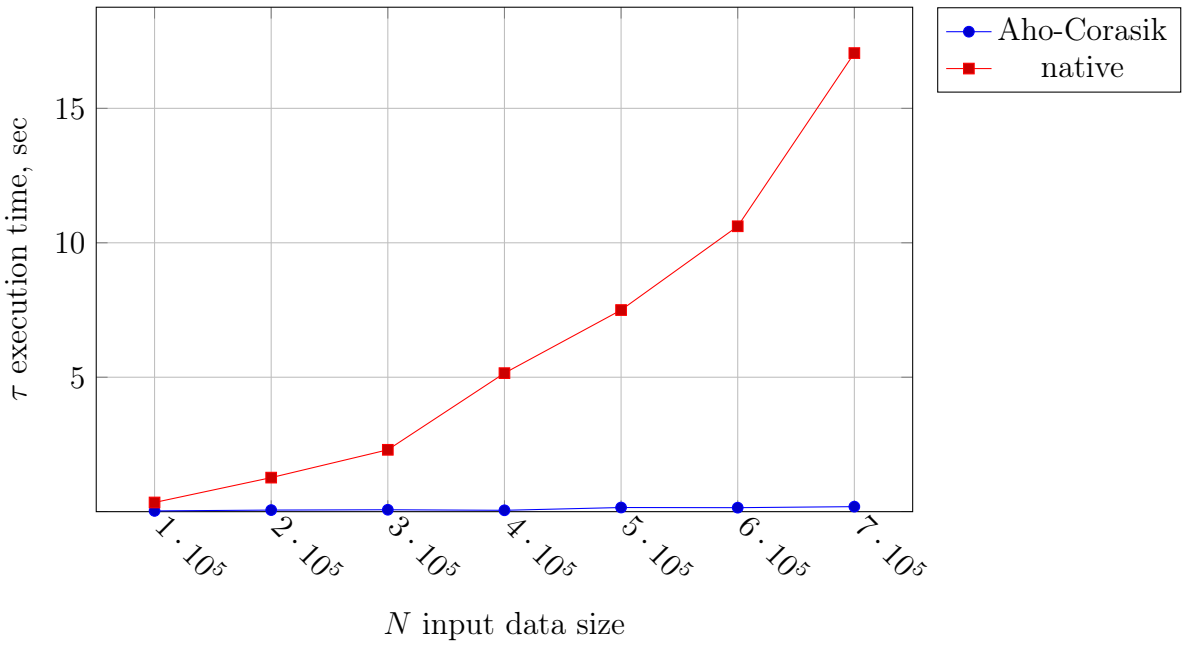
- ReadPattern() — функция считывает часть паттерна до джокера и добавляет в дерево, пока не будут считаны все подстроки максимальной длины без джокера

- Preprocessing() — функция строит связи неудач и связи выхода, обходя дерево в ширину
- ReadText() — функция считывает текст в массив text, учитывая условие регистронезависимости
- Search() — функция поиска вхождения всех подстрок без джокера, согласно алгоритму Ахо-Корасик
- PrintOccurrence() — функция вывода ответа, которая проверяет, что по индексу массива matches лежит число, равное количеству подстрок без джокера

Дневник отладки

№	Описание ошибки	Способ устранения
1	WA9, сначала я хотела проверить правильность работы алгоритма Ахо-Корасик. При нахождении паттерна выводился неправильный номер паттерна.	Ошибка произошло из-за невнимательности, для ее устранения я пересмотрела код и поменяла индекс.

Тест производительности



По результатам тестирования видно, что алгоритм Ахо-Корасик выигрывает по времени у наивного алгоритма, благодаря использованию префиксного дерева и связей неудач. Алгоритм соответствует сложности O(n + m + k), где n — длина текста, m — длина паттерна, не содержащих джокеров, k — количество вхождений всех подстрок в текст.

Выводы

В этой лабораторной мне было необходимо решить задачу поиска с помощью алгоритма Ахо-Корасик для образца с джокерами, который выполняется за время O(n + m + k). Этот алгоритм имеет практическое применение. Он получил широкое распространение в системных программах для потоковой обработки текстов, например, в утилите grep. Также важный случай, где встречаются джокеры, – это факторы транскрипции ДНК.