Лабораторная работа № 5 по курсу дискретного анализа: «Суффиксные деревья»

Выполнил студент группы М8О-307Б-21 МАИ Дубровин Дмитрий

Условие:

Вариант: 1 -> Поиск в известном тексте неизвестных заранее образцов

Найти в заранее известном тексте поступающие на вход образцы.

Формат ввода

Текст располагается на первой строке, затем, до конца файла, следуют строки с образцами.

Формат вывода

Для каждого образца, найденного в тексте, нужно распечатать строчку, начинающуюся с последовательного номера этого образца и двоеточия, за которым, через запятую, нужно перечислить номера позиций, где встречается образец в порядке возрастания.

Пример

Ввод	**Вывод**
abcdabc	1: 1
abcd	2: 2
bcd	3: 2, 6
bc	

Метод решения

Класс SuffixTreeNode

- Описывает узел дерева суффиксов.
- Содержит start и end для хранения диапазона индексов суффикса.
- Содержит map edges, который отображает символ на соответствующий дочерний узел.
- Содержит suffixNumber для хранения номера суффикса, suffixLink для хранения суффиксной ссылки.

Класс SuffixTree

- Описывает дерево суффиксов.
- Содержит метод Insert для вставки суффикса в дерево.

- Содержит метод Find для поиска подстроки и получения всех индексов суффиксов, начинающихся с этой подстроки.
- Содержит метод DFS для обхода дерева и добавления индексов суффиксов в выходной set.
- Конструктор SuffixTree принимает входной текст и строит дерево суффиксов для данного текста.

Функция main

- Считывает входной текст и строит для него дерево суффиксов.
- Считывает подстроки и для каждой подстроки выводит все индексы, с которых начинается суффикс, содержащий эту подстроку.

Описание работы кода

1. Строительство дерева суффиксов

- Для каждого суффикса в тексте он вставляется в дерево с помощью метода Insert.
- В процессе вставки, для каждого суффикса могут быть созданы новые узлы или подстроки могут быть разделены на два узла.

2. Поиск подстроки

- Используя метод Find, происходит поиск подстроки в дереве суффиксов.
- Если подстрока присутствует в дереве, то выполняется обход поддерева с этой подстрокой, и индексы всех суффиксов, начинающихся с этой подстроки, добавляются в выходной set.

3. Обход дерева

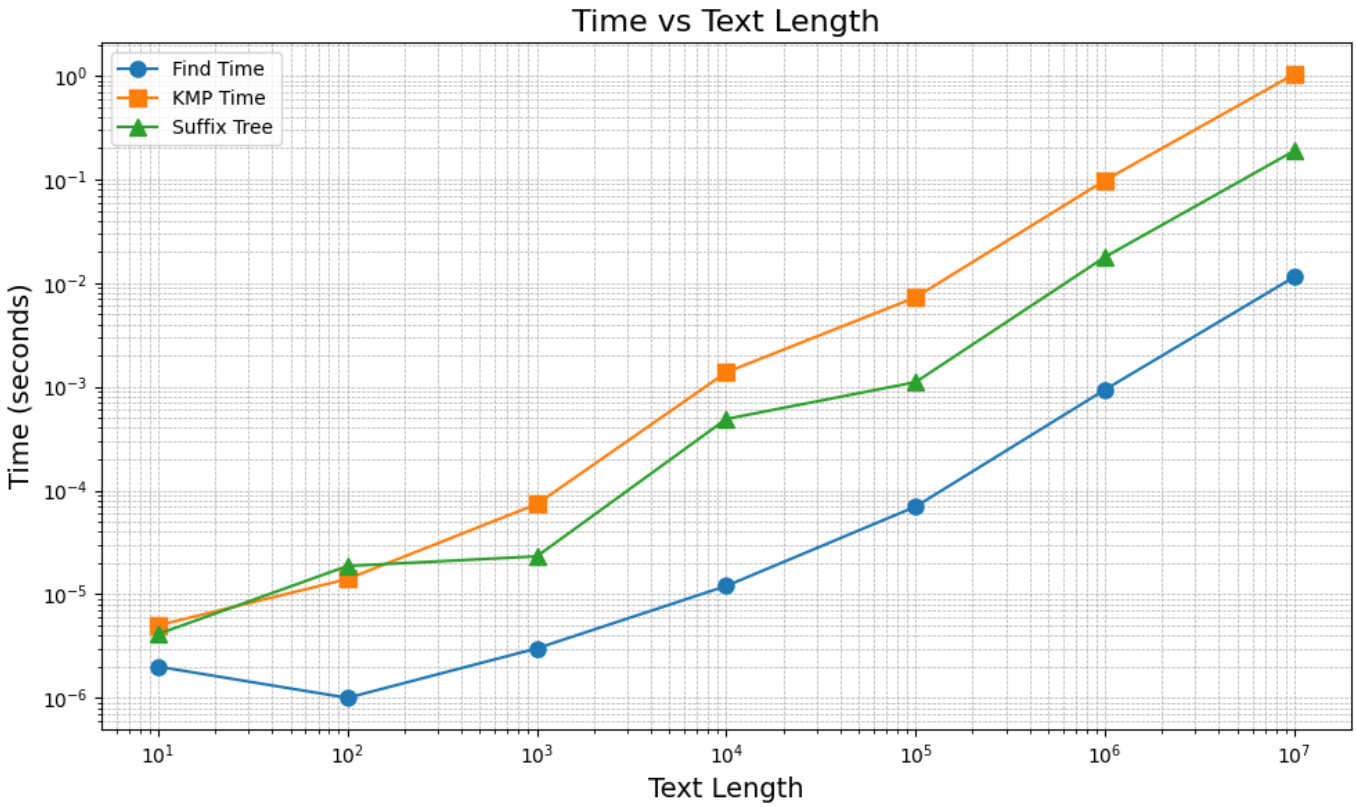
- Метод DFS обходит поддерево, начиная с заданного узла, и добавляет все индексы суффиксов из листьев этого поддерева в выходной set.

4. Вывод результатов

- Функция main выводит результаты поиска подстроки: для каждой подстроки выводит индексы всех суффиксов, начинающихся с этой подстроки.

Тест производительности

Тест производительности из себя представляет собой следующее: я сравниваю время работы разработанного мною алгоритма с КМП и базовым методом find на Python. Код для получения данных для графика, а также тесты представлены в одноименной папке. Также стоит заметить, что тесты получены с помощью библиотеки random на Python:



Как видно из графика, алгоритм быстрее базовой реализации КМП, однако уступает в скорости методу find.

Вывод

Выполнив лабораторную работу №5 по курсу «Дискретный анализ», я изучил суффиксные деревья и вспомнил C++. Я убедился, что использование данной структуры данных позволяет решать задачи поиска подстроки в тексте гораздо более эффективно по сравнению с наивными методами, особенно на больших объёмах данных, что делает данную структуру полезной во многих приложениях. Таким образом, построение и использование дерева суффиксов является важным навыком в области информатики и разработки программного обеспечения, и выполнение данной лабораторной работы позволяет лучше понять принципы работы с текстовыми данными и методы их обработки.