Лабораторная работа №5 по курсу дискретного анализа: Суффиксные деревья.

Выполнил студент группы М8О-308Б-22 Н. А. Ширяев

Условие

Необходимо реализовать алгоритм Укконена построения суффиксного дерева за линейное время. Построив такое дерево для некоторых из входных строк, необходимо воспользоваться полученным суффиксным деревом для решения своего варианта задания. Алфавит строк: строчные буквы латинского алфавита (т.е., от a до z).

Упрощенный вариант:

Реализовать поиск подстрок в тексте с использование суффиксного дерева. Суффиксное дерево можно построить за O(n2).

Метод решения:

Наивный способ представляет собой compact trie, которое построено по всем суффиксам текущей строки, начиная от несобственного суффикса и заканчивая суффиксом,состоящий из последней буквы.

При это мы должны сохранять свойства суффиксного дерева такие как:

1) Такое же кол-во листьев как и суффиксов в строке.

2) trie должен быть compact trie.

В наивном алгоритме приходится каждый раз при проходе до определенной позиции, начиная с корня дерева. Если первая буква из паттерна не выходит ни один суффикс из корня, то значит такого паттерн в тексте нет. Если все же начинается, то двигаемся дальше по ребрам деревьев и поиск происходит как в trie. При поиске в trie надо найти вершину путь, у которой от корня совпадает с искомым паттерном, и все следующие вершины следует вывести в возрастающем порядке.

Если приход в лист, но паттерн не закончился и следующей буквы нет, то паттерн не входит в с текст.

Если в ребро или во внутреннюю вершину и следующие буквы паттерна нет в trie, то паттерн не входит в строку.

Если приход куда-то и паттерн закончился, то нашлось вхождение паттерна в тексте:

1) Если в листе, то вхождение одно и номер листа указывает на позицию входа.

2) Если закончился во внутренней вершине, то из этой внутренней вершины обход всех листьев, которые входят в поддерево этой внутренней вершины и со всех этих позиций есть вхождения.

3) Если нахождение на ребре, то с ребра спуск вниз до внутренней вершины и из этой вершины обход всех ее листьев, и со всех этих позиций есть вхождения.

Для вставки очередного суффикса в trie следует найти отличие текущего суффикса от уже существующего суффикса с наибольшим общим префиксом. Когда такой суффикс найден происходит вставка отличающейся части либо в вершину, либо также в вершину, но посредством разделения существующего ребра на две части.

Описание программы:

В файле lab5.cpp прописана вся программа, в которой содержится класс вершины дерева, класс самого суффиксного дерева

Дневник отладки:

Было ограничение по времени. Преодолел с помощью сортировки подсчетом.

Тест производительности

Наивный алгоритм:

Длина теста: 1000; 10000; 100000;

Время выполнения: 1.307; 11.399; 99.098

Из проведенных тестов видно, что наивный алгоритм работает со сложностью O(n2).

Выводы

В данной лабораторной работе №5 я вспомнила тему предыдущего семестра, которая помогла повторить и закрепить полученные знания. Также суффиксные деревья оказываются очень полезны, так как они помогают найти позиции pattern, где его префиксы совпадают со суффиксом текста. Если использовать алгоритм Укконена, то можно добится более эффективной производительности со сложностью O(n).