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

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

Условие

Реализовать поиск подстрок в тексте с использованием суффиксного дерева. Суффиксное дерево можно построить за $O(n^2)$.

Метод решения:

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

При этом мы должны сохранять свойства суффиксного дерева такие как:

- такое же кол-во листьев, как и суффиксов в строке;
- trie должен быть compact trie.

В наивном алгоритме сравниваем каждый раз при проходе до определенной позиции, начиная с корня дерева. Если первая буква из паттерна не входит ни в один суффикс из корня, то значит такого паттерна 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 приходим в лист, но паттерн не закончился и следующей буквы нет, то паттерн не входит в текст.

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

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

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

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.

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 main.cpp прописана вся программа, в которой содержится класс вершины дерева, класс самого суффиксного дерева

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 $O(n^2)$.

Выводы

В данной лабораторной работе №5 я вспомнил 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.