Лабораторная работа № 5 по курсу дискретного анализа: суффиксные деревья

Выполнил студент группы М08-312Б МАИ *Лобанов Олег.*

Условие

Найти образец в тексте используя статистику совпадений.

Входные данные: На первой строке располагается образец, на второй - текст.

Выходные данные: Последовательность строк содержащих в себе номера позиций, начиная с которых встретился образец. Строки должны быть отсортированы в порядке возрастания номеров.

Метод решения

Строить суффиксное дерево будем при помощи *алгоритма Укконена.* Основной идеей алгоритма является последовательное построение суффиксного дерева для строки S длины $n+1$ в реальном времени, т.е. в каждой $i+1$-ой фазе из $n+1$ фаз алгоритма преобразуется неявное суффиксное дерево, построенное в предыдущей фазе алгоритма для префикса строки длины i, для получения неявного дерева префикса строки длины $i+1$ пр помощи **3-ех привил продолжения суффиксов** (при этом неявное суффискное дерево, построенное в фазе $n+1$ соответсвует искомому суффискному дереву для строки S).

Правила продолжения суффиксов:

- Если путь кончается в листе следует добавить символ в конец листа.
- Если кончается во внутренней вершине и не существует дуг, которые начинаются с добавляемого символа:
  - Если вершина мнимая, тогда создается новая вершина и в ней развилка к которому добавляется листовая дуга, которая начинается с добавляемого символа.
  - Если вершина явная, тогда к ней просто добавляется новая листовая дуга, которая начинается с добавляемого символа.
- Если путь кончается в вершине и существует дуга из вершины, которая начинается с добавляемого символа, ничего делать не надо.

Данный алгоритм имеет сложность $O(n^3)$, но применив **4 приема реализации**, можно сделать линейную сложность $O(n)$ работы *алгоритма Укконена.*

Приемы реализации:

- Переход по суффиксным связям с использованием скачка по счетчику для быстрого поиска продолжений суффиксов в дереве вместо последовательного поиска продолжений от корня.
- Вместо хранения всей подстроки в каждой дуге хранить только два числа, первое из которых обозначает номер первого символа дуги в строке S, а второе обозначает длину этой строки.
- Вместо того, чтобы каждый раз применять *1-ое правило продолжения* ранее созданных листов во всех фазах, сразу при создании устанавливать на дугах длину равную длине всей строки S. При этом хранить указатель на последнюю вершину в которой было применено павило создания листа (*2-ое правило*).
- При первом же использовании *3-его правила* прекращать дальнейшее исполнение текущей фазы и переходить к следующей фазе, так как во всех последующих продолжениях не потребуется никаких действий.

Описание программы

Основные функции и процедуры:

- *Suffix::Suffix(std::istream &check)* - конструктор суффиксного дерева по алгоритму Укконена для образца из потока.
- *std::string &GetStr()* - Функция, возвращающая строку, для которой построено суффиксное дерево.
- *void Suffix::Find(istream &check)* - Процедура, выполненяющая поиск статистики совпадений для текста из потока и выводящая позицию вхождения образца в текст, если статистика совпадает с длиной строки S.
- *Suffix:: Suffix()* - Деструктор суффиксного дерева.

Дневник отладки

После первой отправки выдло ошибку на 3 тесте. Через час выяснилось, что я немного запутался в индексах и начинал обработку данных не с той позиции.

Выводы

После ыполнения пятой лабораторной работы я познакомился с суффиксными деревьями. Я научился строить суффиксные деревья при помощи алгоритма Укконена.