

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Дальневосточный федеральный университет»**

**(ДВФУ)**

|  |
| --- |
| **ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**  **Департамент математического и компьютерного моделирования** |

**ДОКЛАД**

о практическом задании по дисциплине

“Алгоритмы и структуры данных”

по теме

“Алгоритм Укконена”.

направление подготовки 09.03.03 «Прикладная информатика»

профиль «Прикладная информатика в компьютерном дизайне»

|  |  |  |
| --- | --- | --- |
|  |  | Выполнил студент  гр. Б9121-09.03.03пикд  Рыжков Данил Максимович \_\_\_\_\_\_\_ |
|  |  | *(Ф.И.О.) (подпись)* |
| Рег. № \_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |  | Руководитель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(должность, уч.звание)*  **\_\_\_**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  *(Ф.И.О.) (подпись)*  «\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022г. |

г. Владивосток

2022

# Глоссарий

# Введение

 Алгоритм Укконена — этоонлайн-алгоритм построения суффиксного дерева за линейным время, который изобрел Эско Укконен в 1995 году.

**Суффикс** — это последние несколько символов строки.

Первым суффиксом строки называется эта строка без первого символа. Например, у строки "семки" есть следующие суффиксы:

|  |  |
| --- | --- |
| Индекс | Суффикс |
| 0 | “семки” |
| 1 | “емки” |
| 2 | “мки” |
| 3 | “ки” |
| 4 | “и” |
| 5 | “” |

**Суффиксный дерево** — дерево, содержащие все суффиксы некоторой строки (и только их). Позволяет выяснять, входит ли строка *w* в исходную строку *t*, за время ***О*** *(|w|)*, где *|w|* — длина подстроки *w*.

Алгоритм, который изобрел Эско Укконен для построения суффиксного дерева за линейное время можно представить сначала как простой, но неэффективный метод, который с помощью нескольких приёмов достигает уровня лучших алгоритмов по времени счёта в наихудших условиях.

Позже были изобретены другие, более эффективные алгоритмы, например обновленный алгоритм Питера Вайнера, создателя суффиксных деревьев.

Оригинальный алгоритм, представленный Питером Вайнером, проходил в обратном порядке от последнего символа к первому, от самого короткого к самому длинному суффиксу. Более простой алгоритм был найден Эдвардом М. Маккрейтом, переходящим от самого длинного к самому короткому суффиксу.

**Основная идея алгоритма заключается в трех модификациях.**

1. Пусть дерево будет сжатым суффиксным деревом.
2. Пусть в ребрах хранятся не строки, а указатели на начало и конец суффикса.
3. Вершины, у которых не более одного ребенка сжимаются в одну.
4. Использование суффиксных ссылок.

**Сжатое суффиксное дерево** — структура данных для хранения набора строк, представляющая из себя дерево с символами на ребрах. Строки получаются последовательной записью всех символов, хранящихся на РЕБРАХ между корнем бора и терминальной вершиной.

**Сжатое суффиксное дерево** — это некоторая оптимизация несжатого дерева, в котором структуру из нескольких последовательных вершин, выглядящих как односвязный список, заменяют на одно ребро

Поиск подстроки P в строке S будет осуществляться за O(P).

Понятие Суффиксной ссылки

* Она определенна для внутренних вершин дерева.
* Из вершины, хранящей в себе подстроку суффиксная ссылка будет вести в вершину, соответствующую той же строке, но без первого символа
* Для корня суффиксной ссылки нет.

**Базовый принцип**

На вход подаётся строка. Выполняется проход по строке с первого до последнего символа, на каждой итерации дерево заполняется новым суффиксом. Этот порядок добавления символов придает алгоритму Укконена свойство "on-line".

Свойство **on-line** означает что, алгоритм читает последовательно поступающие данные и получает готовое решение на каждом шаге. Это свойство основано на том, что данные на вход алгоритму подаются частями, будь то текст или какие-то запросы. Текущая позиция в дереве соответствует максимальному неполному суффиксу уже добавленных букв, который уже встречался где-то раньше.

Применим алгоритм для чрезвычайно простой строки, строки без повторяющихся символов: abc.

Мы начинаем *слева*и сначала вставляем одиночный символ a, создавая ребро из корня к листу и отмечаем его [0, #] - это означает что ребро представляет подстроку начинающуюся с позиции 0 и заканчивающуюся *на текущем конце т.е.* у каждого ребра метка это пара чисел [от, до] - указатели на позиции в тексте. В таком случае, каждое ребро содержит строку произвольной длины, но занимает только O(1) памяти (два указателя).

Что позволяет доказать, память, занимаемая деревом, будет N. нас есть начальное дерево, которое выглядит так:



Теперь мы переходим к позиции 2 (сразу за b). **Наша цель: на каждом шаге** вставлять **все суффиксы до текущей позиции**. Мы делаем это так:

* расширяя существующее a-ребро до ab
* вставляя новое ребро для b

В нашем представлении это выглядит так:

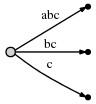


**Наблюдается** два момента:

* Представление ребра для ab такое же, как и было в начальном дереве: [0, #]. Его значение автоматически изменилось, потому что мы обновили текущую позицию # с 1 на 2.
* Каждое ребро занимает ***O***(1) места, потому что состоит только из двух указателей на позиции в тексте, независимо от того, сколько символов оно представляет.

Далее мы снова увеличиваем позицию и обновляем дерево добавлением c ко всем существующим ребрам и вставляя одно новое ребро для нового суффикса c.

В нашем представлении:



**Наблюдения:**

* Дерево является корректным суффиксным деревом *вплоть до текущей позиции* после каждого шага
* Шагов столько же, сколько и символов в тексте
* Объем работы на каждом шаге O(n), потому что все существующие ребра обновляются автоматически приращением #, а вставка одного нового ребра для конечного символа может выполняться за O(1). Следовательно, для строки длинной n, необходимо O(n) времени

Реализация.

Список литературы

* Реализация алгоритмов/Алгоритм Укконена - <https://ru.wikibooks.org/wiki/Реализации_алгоритмов/Алгоритм_Укконена>
* Вики-Итмо <https://neerc.ifmo.ru/wiki/index.php?title=Алгоритм_Укконена>
* Статья об Алгоритме на Хабре <https://habr.com/ru/post/111675/>
* Юрий Лифшиц - Построение суффиксного дерева за линейное время [01ianote.pdf (yury.name)](http://yury.name/internet/01ianote.pdf)
* Укконен Э. (1995)  ["Онлайн-построение деревьев суффиксов"](https://translated.turbopages.org/proxy_u/en-ru.ru.76bae6fa-6330df73-92ce9675-74722d776562/www.cs.helsinki.fi/u/ukkonen/SuffixT1withFigs.pdf).
* Вайнер, Питер (1973). ["Алгоритмы линейного сопоставления с образцом"](https://translated.turbopages.org/proxy_u/en-ru.ru.76bae6fa-6330df73-92ce9675-74722d776562/airelles.i3s.unice.fr/files/Weiner.pdf).
* Реализация Алгоритма Укконена <https://e-maxx.ru/algo/ukkonen>
* Алгоритм Укконена: от простого к сложному <https://habr.com/ru/post/533774/>
* *Гасфилд Д.* Строки, деревья и последовательности в алгоритмах: Информатика и вычислительная биология / Пер. с англ. И. В. Романовского. — 2-е изд. — СПб.: Невский Диалект, 2003. — 654 с.
* Павел Марвин Лекция по суффиксным деревьям ИТМО <https://www.youtube.com/watch?v=WjzR1eFbAeo>
* Алгоритм построения суффиксных деревьев <https://u4isna5.ru/diplom/20-diplominformatika/470-2012-02-13-18-54-40>
* Статья на CodeWork <https://codeforces.com/blog/entry/16780>
* Лекция от Computer Science Centre <https://www.youtube.com/watch?v=wyFvEECgsWs&t=828s>
* Лекция от Технологии в контуре <https://www.youtube.com/watch?v=kxqlaynGgEA&t=3022s>
* Статья на stackoverflow [https://stackoverflow.com/questions/9452701/ukkonens-suffix-tree-algorithm-in-plain-english/9513423#9513423](https://stackoverflow.com/questions/9452701/ukkonens-suffix-tree-algorithm-in-plain-english/9513423%239513423)
* Подробное объяснение алгоритма <https://www.youtube.com/watch?v=aPRqocoBsFQ>
* <https://compscicenter.ru/courses/algorithms-2/nsk/2018-spring/classes/3764/>