Суффиксные деревья(СД)

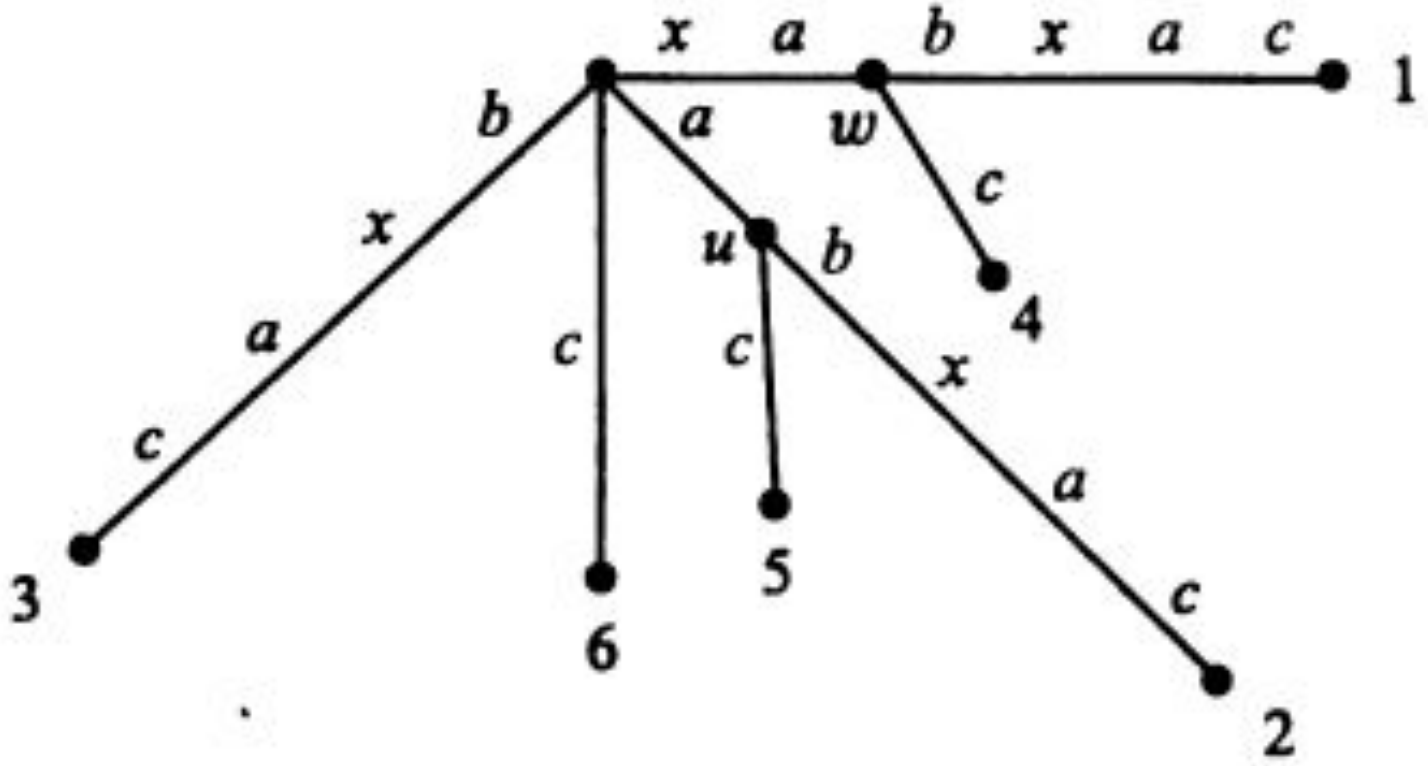
Определение СД

Суффиксное дерево T для произвольной строки из m смволов – это ориентированное дерево с корнем, имеющим ровно m листьев, пронумерованных от 1 до m. Каждая внутренняя вершина, отличная от корня, имеет не менее 2 детей, и каждая дуга помечена непустой подстрокой строки S(дуговой меткой). Никакие 2 дуги, выходящие из одной вершины, не могут иметь пометки, начинающиеся с одного и того же символа.

Суффиксные деревья. Примечание

Для каждого листа l конкатенация дуговых меток пройденных от корня до листа l дуг, т.е. суффикс строки S[i..m].

Пример СД
Строка xabxac



Определения

Определение. *Метка пути* от корня до некоторой вершины — это конкатенация подстрок, помечающих дуги пути в порядке их прохождения. *Путевая метка вершины* — это метка пути от корня $\mathcal{T}$ до этой вершины.

Определение. Для любой вершины v суффиксного дерева *строковой глубиной* v называется число символов в путевой метке v.

Определение. Путь, который кончается в середине дуги (u, v), делит метку (u, v) в своей точке назначения. Определим метку этого пути как путевую метку u с добавлением символов дуги (u, v) до делящей дугу точки назначения.

Определения

Строка xa на 4 слайде помечает внутренн w вершину , , строка a помечает вершину u, а строка xabx помечает путь, который заканчивается внутри дуги
w
(,1), ведущей к листу 1

Использование СД для задачи о точных совпадениях

Заданы образец P длины n, и текст T длины m.Найти все вхождения P в T за время O(n+m).

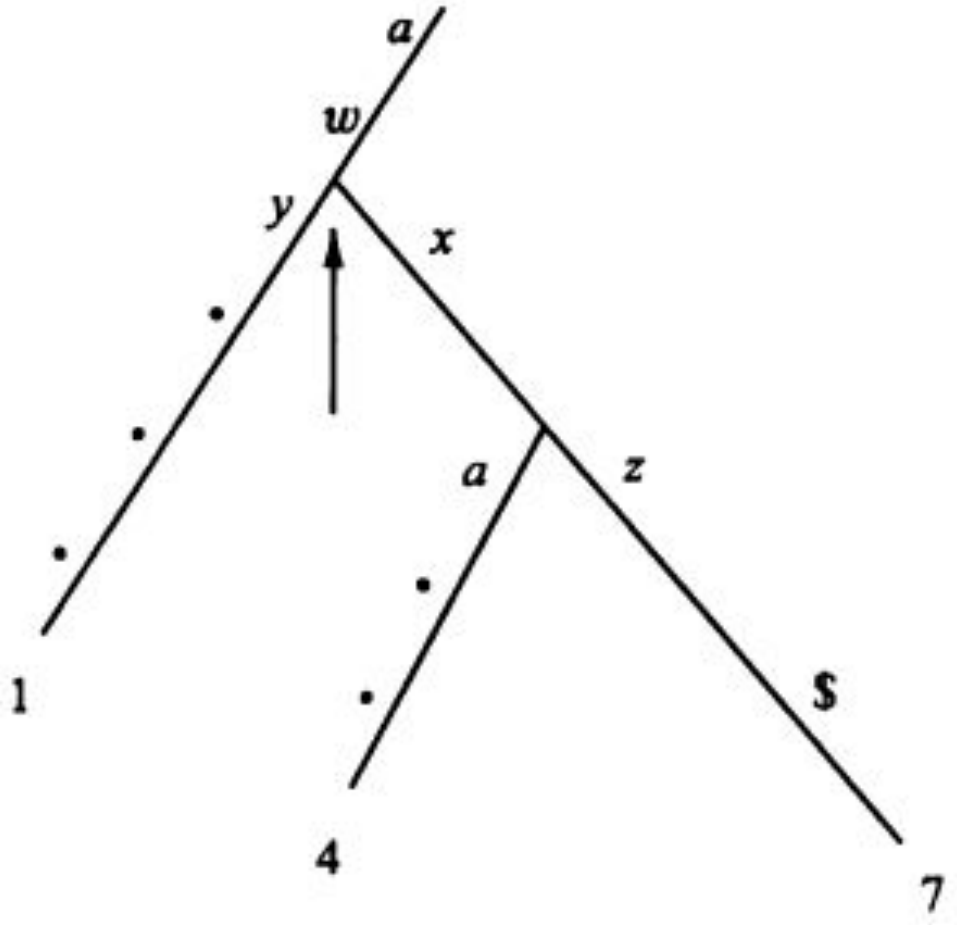
Строим СД для текста T за время O(m). Ищем путь совпадения для P.

Если такого пути нет, нет вхождения.

Если есть, каждый лист в поддереве, корнем которого является вершина окончания поиска, задает номер начальной позиции положения P в T.

Пример определения вхождения строки

a w в *a w y a w x a w x z*.



Наивный алгоритм построения суффиксного дерева

- В дерево включаем простую дугу S$(суффикс S[1..m] $).
- Последовательно включаются суффиксы S[i..m] $ для i от 2 до m.

Обозначим через N[i] промежуточное дерево, в которое входят суффиксы от 1 до i.

- Дерево N[i+1] строится из дерева N[i]
- Начинаем с корня N[i]
- Находим самый длинный путь, метка которого совпадает с префиксом S[i+1 .. m] $. Остановка произойдет либо в вершине дерева, либо в средине какой-нибудь дуги. Если в средине дуги- она разбивается на 2 и вводится новая вершина.
- И т. д.
- Временная сложность O(m*m).

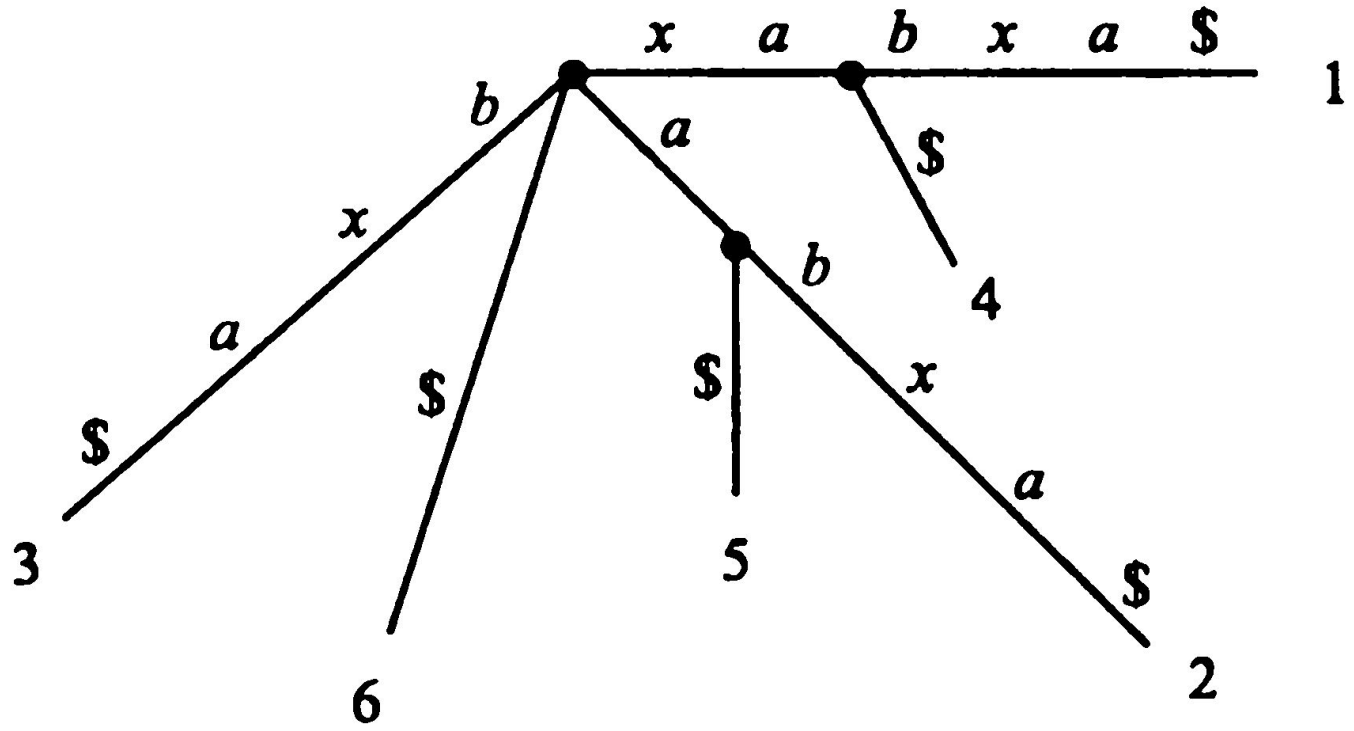
Неявные суффиксные деревья

Определение. *Неявное суффиксное дерево* строки S — это дерево, полученное из суффиксного дерева $S\$$ удалением всех вхождений терминального символа \$ из меток дуг дерева, удалением после этого дуг без меток и удалением затем вершин, имеющих меньше двух детей.

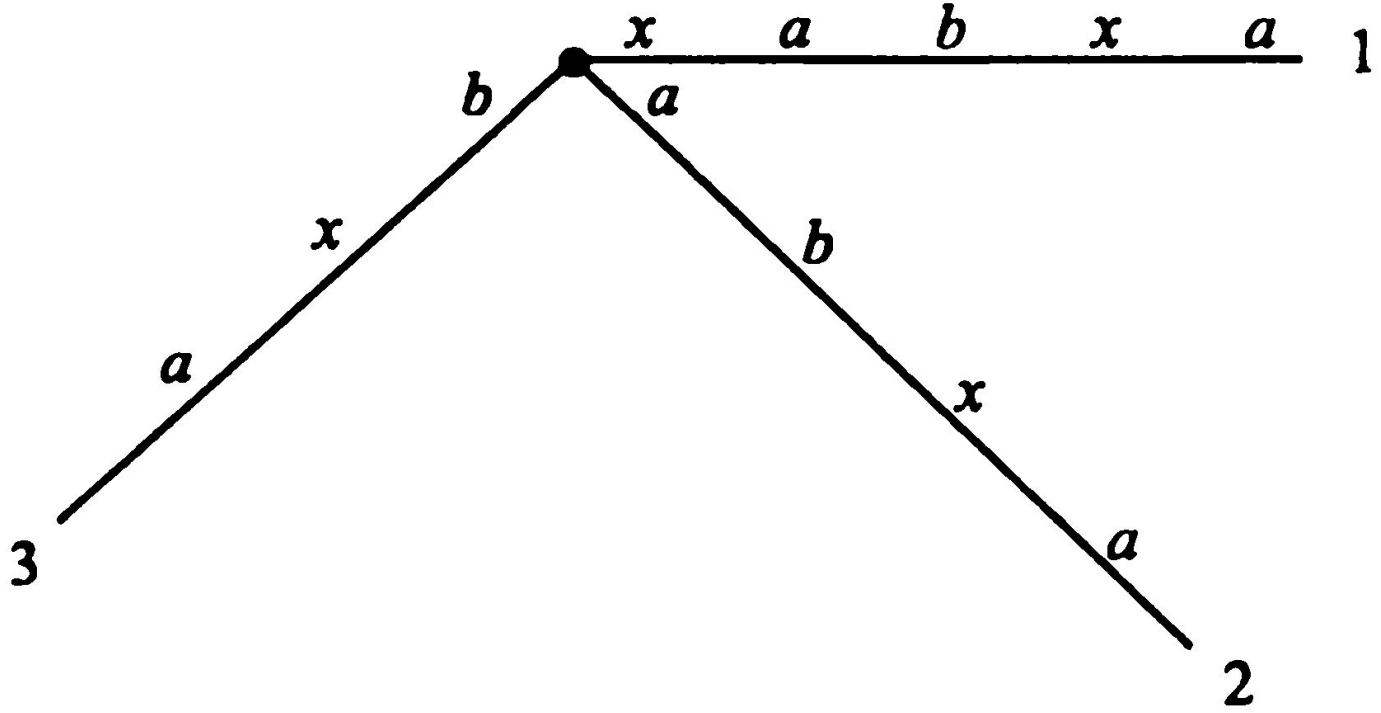
Неявное суффиксное дерево префикса $S[1..i]$ строки S аналогично получается из суффиксного дерева для $S[1..i]\$$ удалением символов \$, дуг и вершин, как описано выше.

Определение. Обозначим через $\mathcal{T}_i$ неявное суффиксное дерево строки $S[1..i]$, где i меняется от 1 до m.

СД строки xabxa$



Неявное суффиксное дерево строки xabxa.



Алгоритм Укконена.

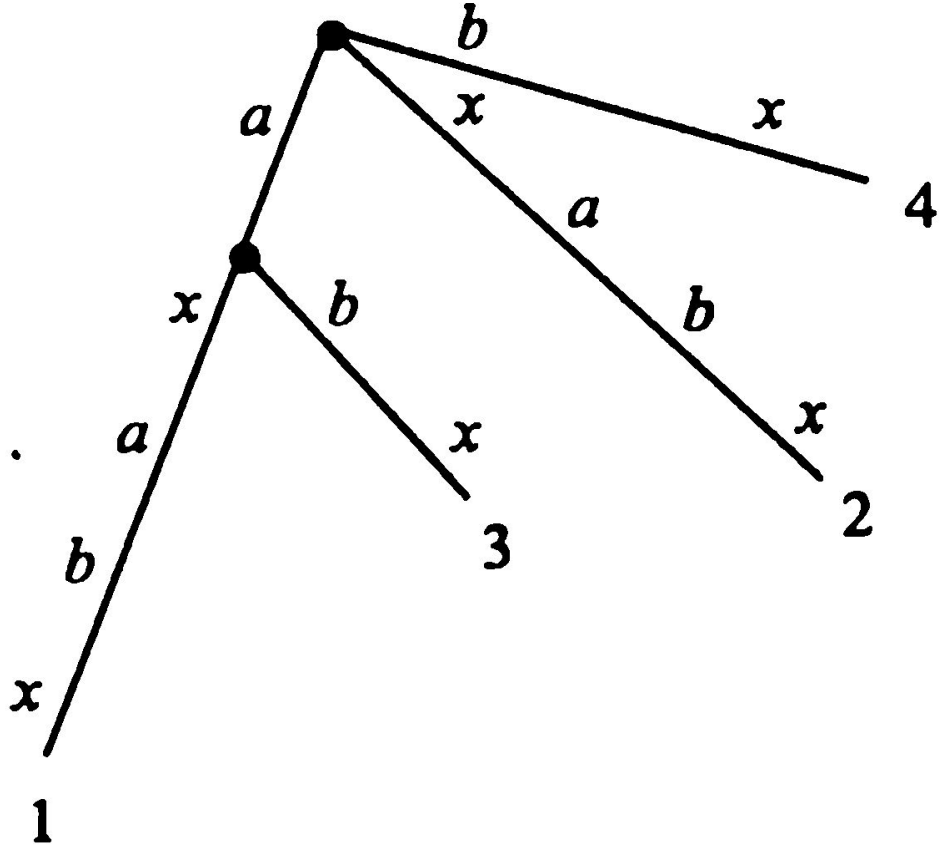
Общее описание алгоритма Укконена

Построить дерево $\mathcal{T}_1$.
for i from 1 to $m - 1$ do begin {фаза $i + 1$}
  for j from 1 to $i + 1$ begin {продолжение j}
    найти в текущем дереве конец пути из корня с меткой $S[j..i]$.
    Если нужно, продолжить путь, добавив символ $S(i + 1)$, обеспечив
      появление строки $S[j..i + 1]$ в дереве.
  end;
end;

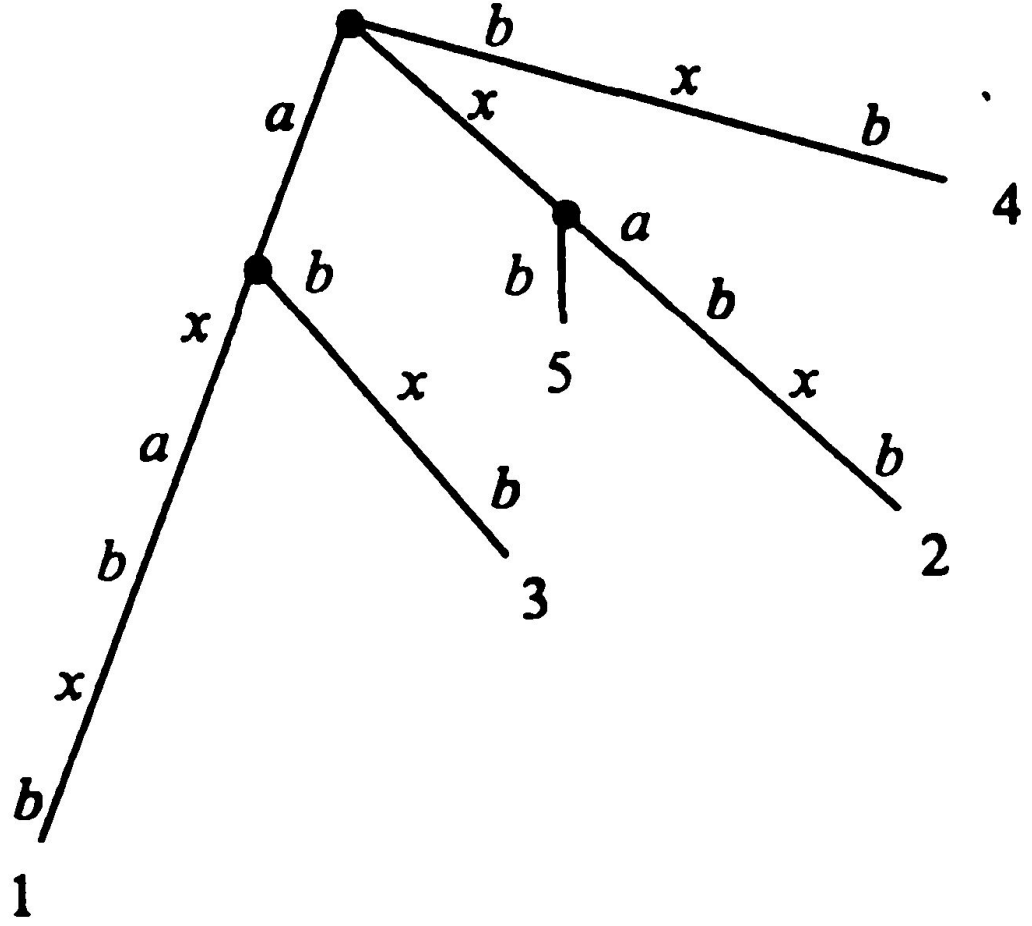
Правила продолжения суффиксов

- Правило 1. В текущем дереве путь β заканчивается в листе. При изменении дерева надо добавить к концу метки этой листовой дуги S[i+1].
- Правило 2. Ни один путь из конца строки β не начинается символом S[i+1], но по крайней мере 1 путь оттуда существует.
- Правило 3. Некоторый путь из конца строки β начинается символом S[i+1], тогда строка βS[i+1]уже существует в текущем дереве. Ничего не делать.

Неявное суффиксное дерево строки axabx до добавления 6 символа b



Расширенное неявное суффиксное дерево строки axabx после добавления b



Реализация и ускорение

Важно в реализации алгоритма Укконена определить концы всех суффиксов S[1..i].

При наивном подходе конец любого суффикса находится за время, порядка его длины.

T[i+1] создается из T[i] за O(i*i), а T[m] – за O(m*m*m).

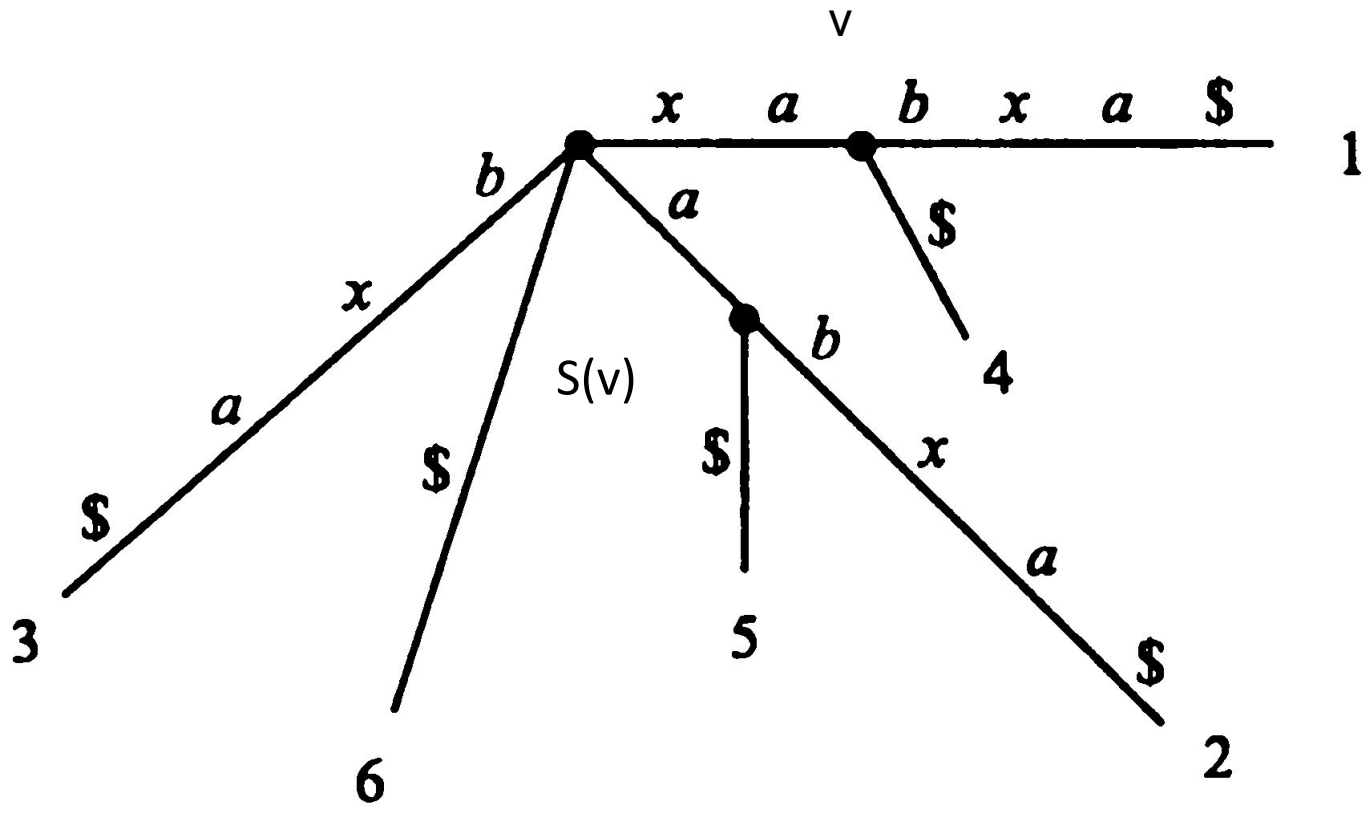
Уменьшим время до O(m).

Суффиксные связи
Первое ускорение реализации

- **Определение.** Пусть xa –произвольная строка. X-первый символ, a - оставшаяся подстрока(м.б. пустой). Если для внутренней вершины v с путевой меткой xa существует другая вершина s(v) с путевой меткой a, то указатель из v в s(v) называется суффиксной связью.

Иногда суффиксная связь из v в s(v) обозначается парой (v,s(v)).

Пример суффиксной связи из v в s(v)



Следствия

Следствие 1. В алгоритме Укконена любая вновь созданная внутренняя вершина будет иметь суффиксную связь с концом следующего предложения.

Следствие 2. В любом неявном суффиксно дереве T[i] , если внутренняя вершина v имеет путевую метку xa, то найдется вершина s(v) дерева T[i] с путевой меткой a

Переходы по суффиксным связям при построении T[i+1]

На фазе i+1 алгоритм находит место суффикса S[j..i] строки S[1..i] в продолжении j и j меняется от 1 до i+1. Суффиксные связи сокращают движение по пути и каждое продолжение.

Переходы по суффиксным связям при построении T[i+1](прод1)

Первое продолжение(j=1) фазы i+1

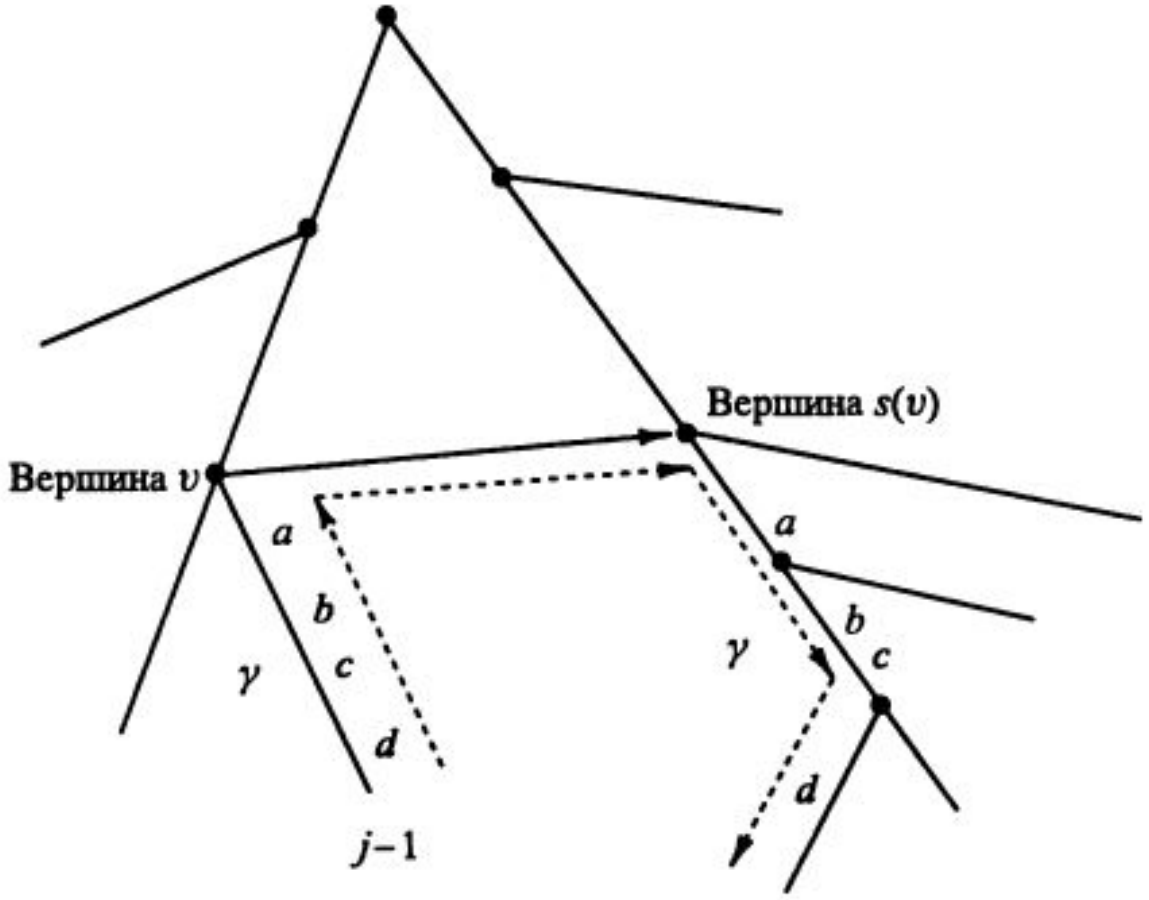
Конец полной строки S[1..i] в листе. Продолжение – просто.

Пусть S[1..i] имеет вид xa, где x -один символ, a - строка, возможно пустая. Пусть (v,1)-дуга, входящая в лист 1. Следующее действие алгоритма – нахождение конца строки S[2..i] =ав текущем дереве.

Переходы по суффиксным связям при построении T[i+1](прод2)

Второе продолжение. Пусть Υ- дуговая метка дуги (v,1). Для нахождения конца a, надо пройти вверх от листа 1 до v, затем по суффиксной связи из v в s(v), затем от s(v) вниз по пути с меткой Υ. Конец пути – конец a.

Пример второго продолжения(в конце пути a дерево изм. по правилам продолжения суффикса)



Переходы по суффиксным связям при построении T[i+1]

Различия между первыми двумя продолжениями и продолжением при j>2.

В общем случае конец S[j-1..i] мб в вершине, из которой выходит уже суффиксная связь. Алгоритм проходит эту суффиксную связь.

Теоретически доказано, что в продолжении j алгоритм никогда не поднимается выше, чем на одну дугу.

Алгоритм отдельного продолжения(SEA). Продолжение j>=2 фазы i+1

begin

1. Найти в конце строки $S[j-1..i]$ или выше его первую вершину v, которая либо имеет исходящую суффиксную связь, либо является корнем. Для этого нужно пройти вверх от конца $S[j-1..i]$ в текущем дереве не более чем на одну дугу. Пусть γ обозначает строку между v и концом $S[j-1..i]$ (возможно, пустую).

Алгоритм отдельного продолжения(SEA). Продолжение j>=2 фазы i+1(1)

2. Если v не корень, пройти суффиксную связь из v в вершину $s(v)$ и спуститься из $s(v)$ по пути строки γ. Если v — корень, пройти по пути для $S[j..i]$ из корня (как в наивном алгоритме).
3. Обеспечить вхождение строки $S[j..i]S(i+1)$ в дерево, используя правила продолжения.
4. Если в продолжении $j-1$ была создана новая внутренняя вершина w (по правилу продолжения 2), то по лемме 1 строка α должна кончаться в вершине $s(w)$, конце суффиксной связи из w. Создать эту суффиксную связь $(w, s(w))$.

end.

Замечание

Результат: Улучшение за счет сокращения перемещений от корня в каждом продолжении.

Для улучшения оценки до O(m*m) введем прием, который будет использоваться при построении и использовании суффиксных деревьев.

Прием №1 – скачок по счетчику

На шаге 2 продолжения j+1 алгоритм идет вниз из вершины s(v) по пути с меткой ϒ. При буквальной реализации прохождение по ϒ имеет сложность, пропорциональную ее длине. При использовании скачка по счетчику временная сложность уменьшается до числа вершин пути.

Временная сложность не превзойдет O (m).

Прием №1

Пусть g –длина ϒ. Пусть g1 – число символов в дуге перехода. Если g1<g, не надо просматривать остальные символы дуги. Просто осуществляется переход к ее концу.

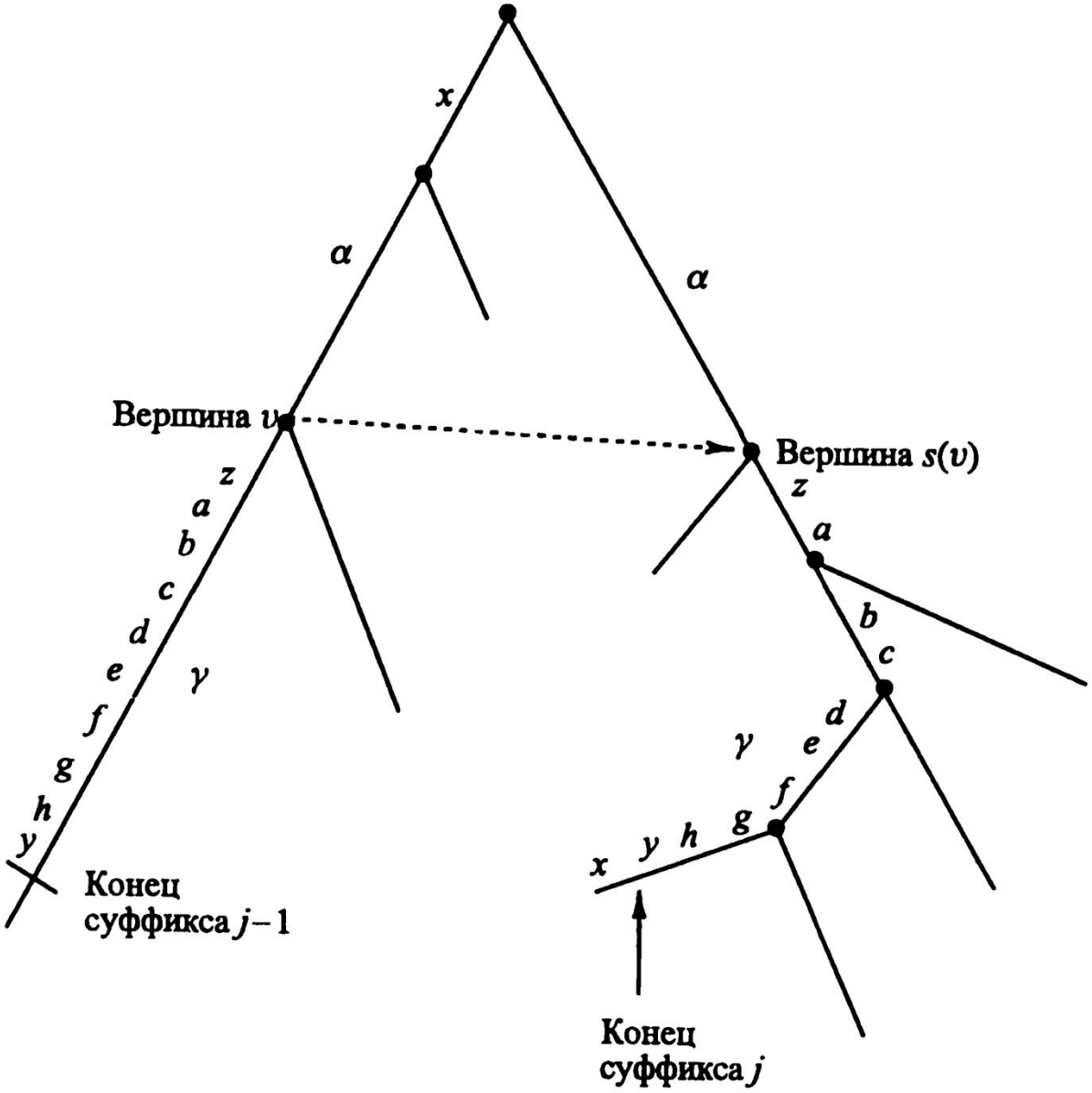
Затем g=g-g1. h=g1+1. Просматриваются выходящие дуги для нахождения правильного продолжения(нач. символ = символу h строки ϒ. Этот шаг повторяется .

При достижении дуги с g<g1, выполняется переход к символу g дуги и завершается, т. к. путь ϒ из s(v)завершился на этой дуге.

Пример

Прием - скачок по счетчику. В фазе i+1 подстрока γ имеет длину 10. Из вершины s(v) выходит копия подстроки γ. Ее конец найден в 3 символах по последней дуге, после того, как алгоритм проскочил 4 вершины.

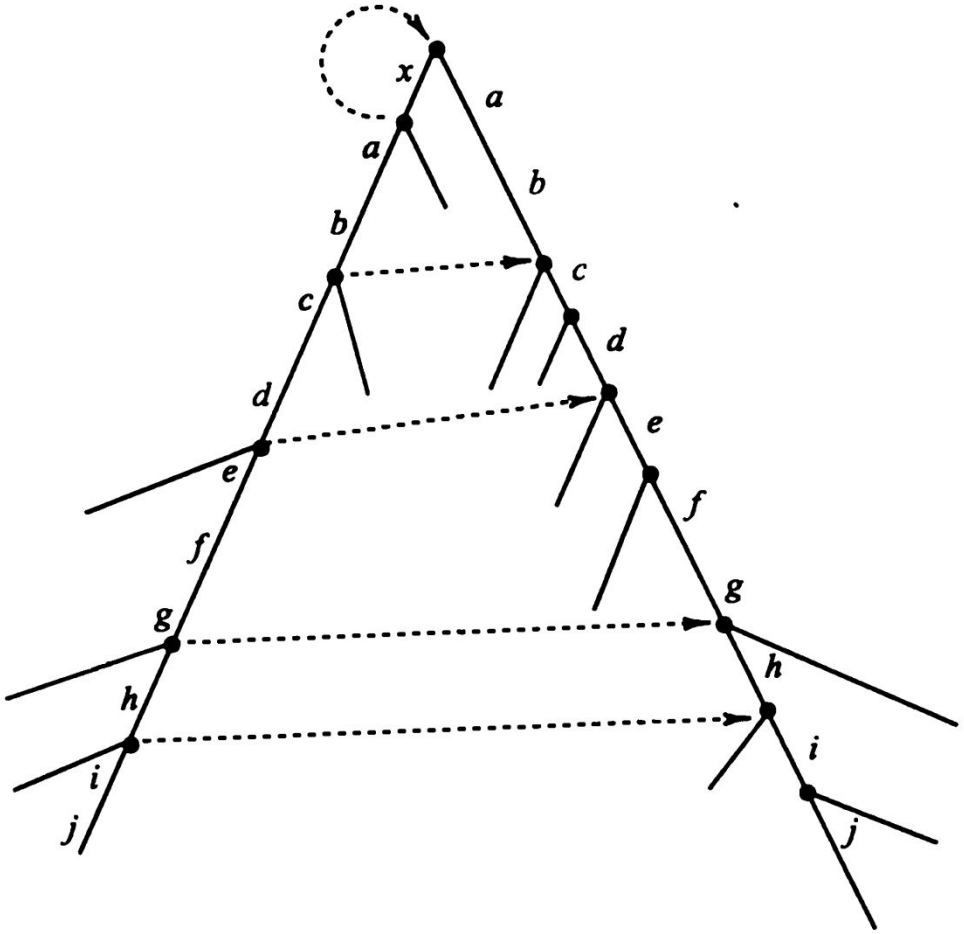
Пример


x
α
α
Вершина v
Вершина s(v)
z
a
b
c
d
e
f
g
h
y
γ
z
a
b
c
d
e
f
g
γ
x
y
h
Конец суффикса j−1
Конец суффикса j

Лемма о суффиксной связи

- Определение. Вершинная глубина вершины – число вершин на пути от нее до корня.
- Лемма 2.Пусть (v,s(v)) – суффиксная связь, проходимая в алгоритме Укконена. В этот момент вершинная глубина v не более чем на единицу превосходит вершинную глубину s(v)).

Соотношение верш глубин: вершина с меткой xab имеет глубину 2, глубина ab равна 1. Глубина вершины с меткой xabcdefg равна 4, а вершины abcdefg -5.



Алгоритм Укконена.

Теорема. При использовании скачков по счетчику любая фаза алгоритма Укконена занимает время O(m).

Всего m фаз. Поэтому справедливо следствие:

Суффиксные связи в алгоритме Укконена обеспечивают время работы O(m*m).

Простая деталь реализации. Сжатие дуговых меток

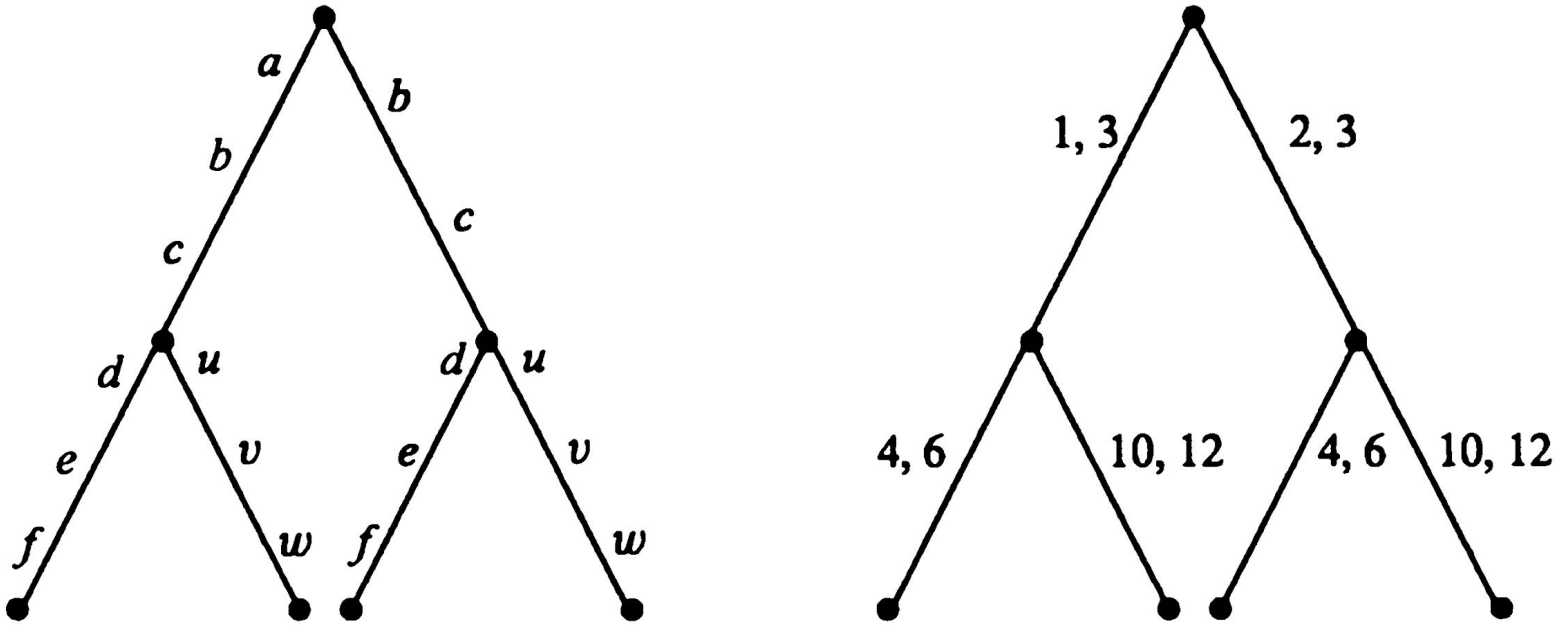
Вместо явной записи подстроки записываем индексную пару: начальная позиция, конечная позиция подстроки в S.

Пусть S=abcdefabcuvw

Деревья представлены на следующем слайде

Деревья с явно заданными дуговыми метками(слева) и сжатыми дуговыми метками (справа)

$S = a\,b\,c\,d\,e\,f\,a\,b\,c\,u\,v\,w$



Дополнительные приемы реализации

Замечание 1. В любой фазе- если правило продолжения 3 применяется в продолжении j, оно будет применяться во всех следующих продолжениях (от j+1 до i+1) до конца фазы.

Прием 2. Заканчивать каждую i+1 после первого использования продолжения 3. Если это произошло в продолжении j, нет необходимости нахождения концов строк S[k..i] с k>j.

Дополнительные приемы реализации

Замечание 2. Стал листом, листом и останешься.

Прием 3. В фазе i+1, когда создается листовая дуга, помечаемая подстрокой S[p..i+1], вместо записи индексов дуги (p,i+1) использовать запись (p,e), где e – глобальный индекс(в начале фазы присваивается значение i+1.

Дополнение

При использовании приемов 2 и 3 явные продолжения в фазе i+1(применяющие алгоритм SEA) требуются только от продолжения j_i +1 до первого продолжения, использующего правило 3(или до продолжения i+1). Все другие продолжения выполняются неявно.

Т.е. фаза i+1 реализуется:

Фаза i+1

Алгоритм одной фазы (single phase algorithm): SPA
begin

1. Увеличить индекс e до $i + 1$. (С помощью приема 3 при этом увеличатся все неявные продолжения от 1 до j_i.)
2. Явно вычислить последовательные продолжения (используя алгоритм SEA), начиная от $j_i + 1$ и до достижения первого продолжения j^*, где применяется правило 3, или до достижения конца этой фазы. (С помощью приема 2 правильно выполняются все продолжения от $j^* + 1$ до $i + 1$.)
3. Приготовляясь к следующей фазе, положить j_{i+1} равным $j^* - 1$.

end.

Теорема 2

Используя суффиксные связи и реализацию приемов 1, 2, 3, алгоритм Укконена строит неявные суффиксные деревья от T1 до Tm за полное время O (m).

Создание настоящего суффиксного дерева

Теорема 3. Алгоритм Укконена строит настоящее суффиксное дерево для S и все его суффиксные связи за время O(m).

Окончательное неявное дерево Tm можно преобразовать в истинное суффиксное дерево за время O(m).

Для этого к S добавляется терминальный символ $ и выполняется шаг алгоритма Укконена с этим символом.