# Суффиксные деревья

## Дискретный анализ 2012/13

Андрей Калинин, Татьяна Романова

26 ноября 2012 г.

## Суффиксные деревья

- Определение
- Возможные приложения

## Алгоритм Укконена

- Общее описание
- Ускорение до $O(m^2)$
- Ускорение до $O(m)$

# Литература


- Дэн Гасфилд, «Строки деревья и последовательности в алгоритмах: Информатика и вычислительная биология», 2003. Главы 5-6, «Введение в суффиксные деревья» и «Построение суффиксных деревьев за линейное время», стр. 119–141.
- E. Ukkonen. (1995). On-line construction of suffix trees. http://www.cs.helsinki.fi/u/ukkonen/SuffixT1withFigs.pdf

# Раздел

# Определение

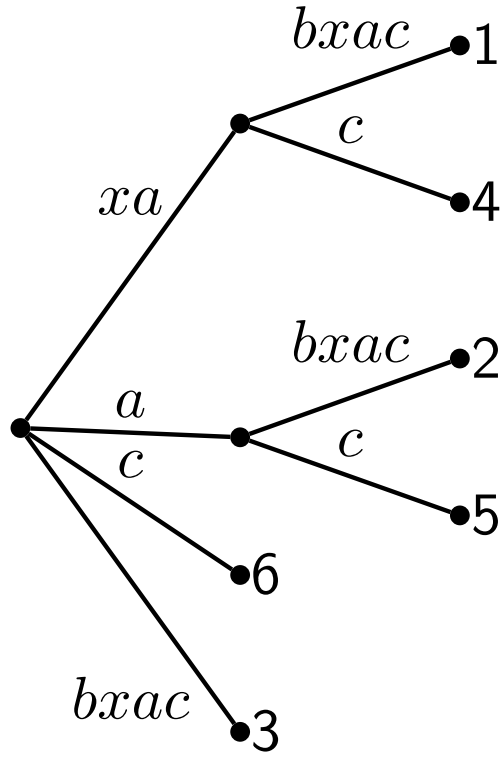
Суффиксное дерево $\mathbb{T}$ для $m$-символьной строки $S$:

1. Ориентированное дерево, имеющее ровно $m$ листьев, пронумерованных от $1$ до $m$.
2. Каждая внутренняя вершина, отличная от корня, имеет не меньше двух детей.
3. Каждая дуга помечена непустой подстрокой строки $S$ (дуговая метка).
4. Никакие две дуги, выходящие из одной вершины, не могут иметь меток, начинающихся с одинаковых символов.
5. Для каждого листа $i$ конкатенация меток от корня составляет $S[i..m]$.

# Определение

Суффиксное дерево $\mathbb{T}$ для $m$-символьной строки $S$:

1. Ориентированное дерево, имеющее ровно $m$ листьев, пронумерованных от $1$ до $m$.
2. Каждая внутренняя вершина, отличная от корня, имеет не меньше двух детей.
3. Каждая дуга помечена непустой подстрокой строки $S$ (дуговая метка).
4. Никакие две дуги, выходящие из одной вершины, не могут иметь меток, начинающихся с одинаковых символов.
5. Для каждого листа $i$ конкатенация меток от корня составляет $S[i..m]$.




Дерево для строки $xabxac$.

# Терминальный символ

- Суффиксное дерево нельзя построить для любой строки: если существует суффикс, совпадающий с префиксом другого суффикса, то не будет выполнено условие о количестве листьев.
- Добавляется терминальный символ, который больше нигде в строке $S$ не содержится (будет обозначаться как $\$$).

# Термины

- Путевая метка вершины: конкатенация подстрок от корня до этой вершины в порядке прохождения соответствующих рёбер.
- Строковая глубина вершины: количество символов в её путевой метке.
- Если некоторый путь заканчивается внутри дуги $\langle u, v \rangle$, то путевая метка этого пути — путевая метка $u$ с добавлением символов дуги $\langle u, v \rangle$ до места назначения.

# Наивный алгоритм

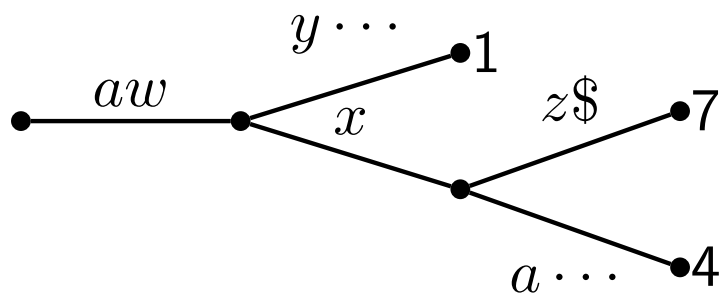
- Начиная со всей строки, последовательно вносить каждый суффикс в дерево.
- Время работы: для строки $S$ длиной $m$ — $O(m^2)$.
- Существуют алгоритмы построения суффиксного дерева за $O(m)$ в предположении ограниченного алфавита.

# Раздел

# Поиск образца в тексте

- Строится суффиксное дерево для текста.
- Ищется путь, совпадающий с образцом. Если такого пути нет, то образец в текст не входит.
- Если путь есть, то все листья поддерева — вхождения.

Поиск $aw$ в $awyawxawxz$:



# Свойства

- Если суффиксное дерево строится за линейное время, то время поиска $O(m+n)$, как и для ранее рассмотренных алгоритмов.
- Но: предварительная обработка $O(m)$, время поиска $O(n)$ (в других алгоритмах было наоборот).
- При обработке большого количества образцов (заранее неизвестного количества) можно выполнять поиск каждого из них в заранее известном тексте за время, зависящее только от длины образца!

# Ещё приложения

- Нахождение общих подстрок для двух и более строк.
- Компрессия данных.
- Нечёткий поиск.
- Выделение повторяющихся фрагментов.

# Раздел

# Неявные суффиксные деревья

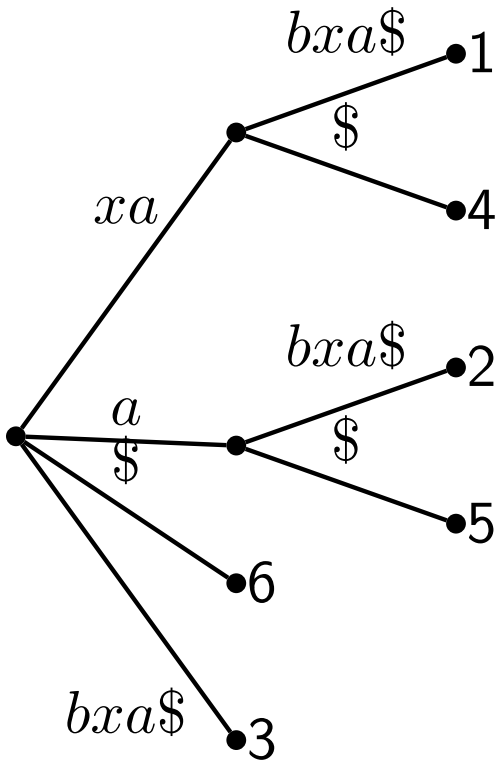
Неявное суффиксное дерево может быть получено из суффиксного дерева строки $S$:

1. удалением всех вхождений терминального символа;
2. затем удалением всех дуг без меток;
3. затем удалением всех вершин, имеющих меньше двух детей (кроме корня).

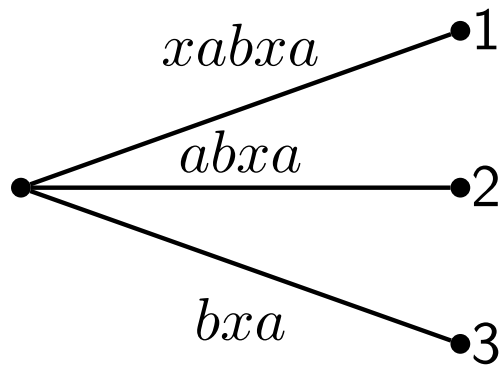
$\mathbb{T}_i$ — неявное суффиксное дерево для строки $S[1..i]$.

# Неявное суффиксное дерево

Суффиксное дерево для $xabxa\$$:




Неявное суффиксное дерево для $xabxa$:



# Алгоритм Укконена

- Последовательно строит неявные деревья $\mathbb{T}_i$ для каждого префикса $S[1..i]$.
- Настоящее суффиксное дерево $\mathbb{T}$ можно получить из $\mathbb{T}_m$ построив следующее неявное дерево для строки с терминальным символом.
- Сначала рассмотрим метод построения дерева за $O(m^3)$, потом улучшим.

# Общий вид алгоритма

1 Построить дерево $\mathbb{T}_1$ (одна дуга с $S(1)$).
2 **for** $i \leftarrow 1$ **to** $m-1$ // Фаза $i+1$
3 **for** $j \leftarrow 1$ **to** $i+1$ // Продолжение $j$
4 Найти в $\mathbb{T}_i$ путь с меткой $S[j..i]$.
5 Если нужно, продолжить путь, добавив символ $S(i+1)$.
// Тогда строка $S[j..i+1]$ будет содержаться в дереве.
6 // В результате получится дерево $\mathbb{T}_{i+1}$

# Правила продолжения суффиксов

$\beta = S[j..i]$ — суффикс $S[1..i]$. Алгоритм находит конец пути $\beta$ и продолжает его так, чтобы $\beta S(i+1)$ так же входил в дерево.

# Правила продолжения суффиксов

$\beta = S[j..i]$ — суффикс $S[1..i]$. Алгоритм находит конец пути $\beta$ и продолжает его так, чтобы $\beta S(i+1)$ так же входил в дерево.

1. Путь $\beta$ кончается в листе: нужно добавить $S(i+1)$ в хвост листовой дуги этого пути.

# Правила продолжения суффиксов

$\beta = S[j..i]$ — суффикс $S[1..i]$. Алгоритм находит конец пути $\beta$ и продолжает его так, чтобы $\beta S(i+1)$ так же входил в дерево.

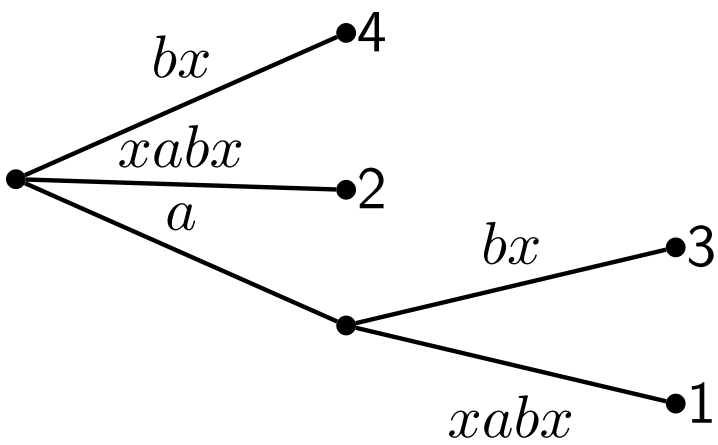
1. Путь $\beta$ кончается в листе: нужно добавить $S(i+1)$ в хвост листовой дуги этого пути.
2. Ни один путь из конца строки $\beta$ не начинается символом $S(i+1)$, но хотя бы один путь оттуда имеется: нужно создать новую листовую дугу, помеченную $S(i+1)$ и указать новому листу номер $j$.

# Правила продолжения суффиксов

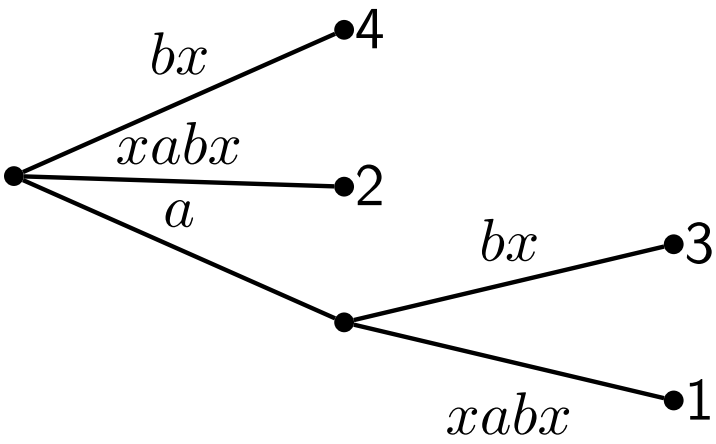
$\beta = S[j..i]$ — суффикс $S[1..i]$. Алгоритм находит конец пути $\beta$ и продолжает его так, чтобы $\beta S(i+1)$ так же входил в дерево.

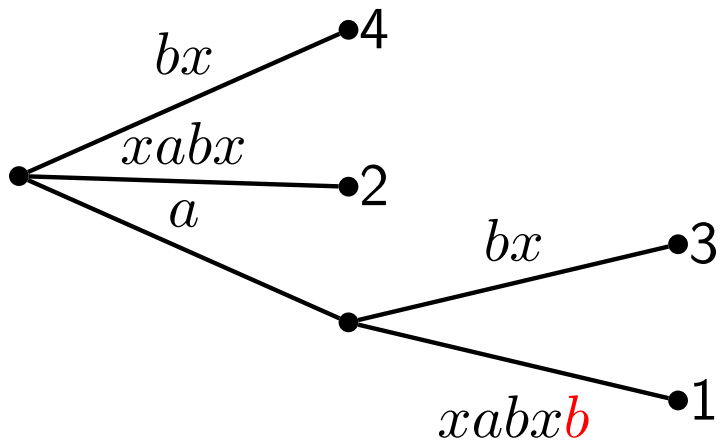
1. Путь $\beta$ кончается в листе: нужно добавить $S(i+1)$ в хвост листовой дуги этого пути.
2. Ни один путь из конца строки $\beta$ не начинается символом $S(i+1)$, но хотя бы один путь оттуда имеется: нужно создать новую листовую дугу, помеченную $S(i+1)$ и указать новому листу номер $j$.
3. Есть некоторый путь от конца строки $\beta$, начинающийся символом $S(i+1)$: ничего делать не надо, строка $\beta S(i+1)$ уже есть в дереве.

# Добавление $b$ в неявное дерево для строки $axabx$



# Добавление $b$ в неявное дерево для строки $axabx$

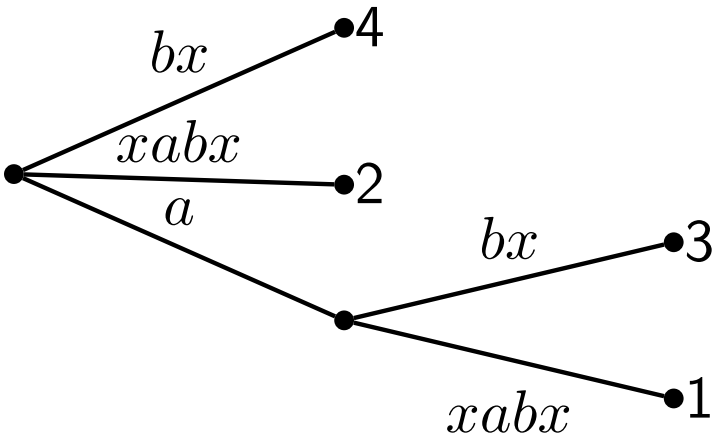


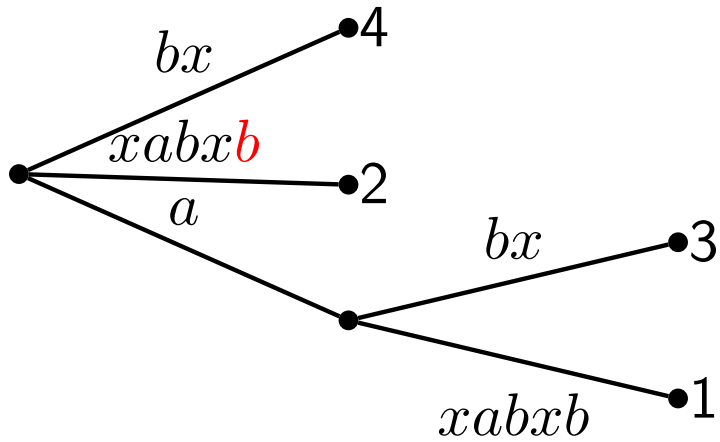




Суффикс `axabxb`

Правило №1, путь $\beta$ заканчивается в листе: добавляем $S(i+1)$ к метке последней дуги.

# Добавление $b$ в неявное дерево для строки $axabx$

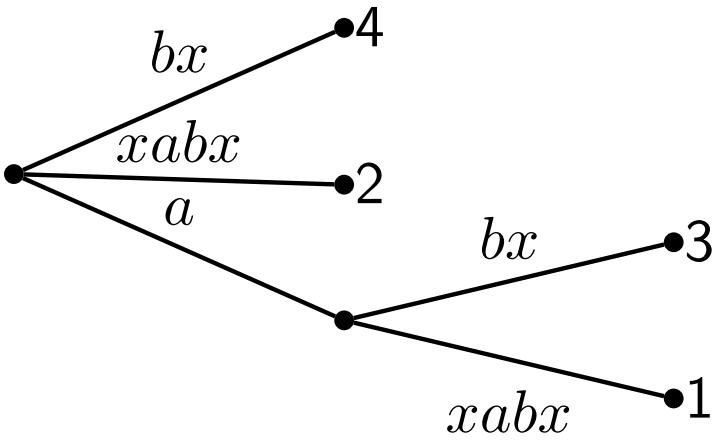


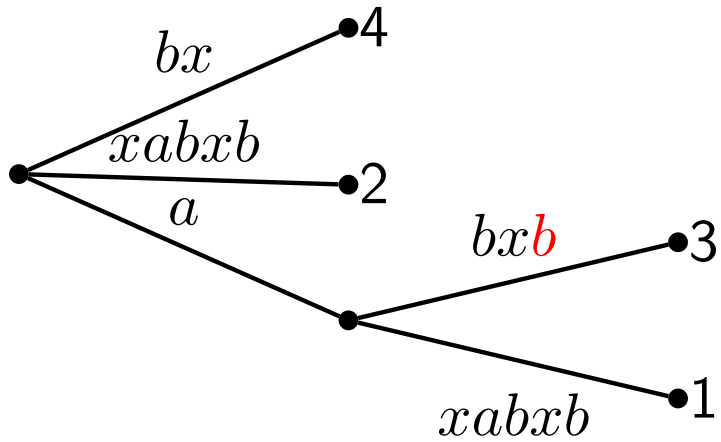




Суффикс `xabxb`

Правило №1, путь $\beta$ заканчивается в листе: добавляем $S(i+1)$ к метке последней дуги.

# Добавление $b$ в неявное дерево для строки $axabx$

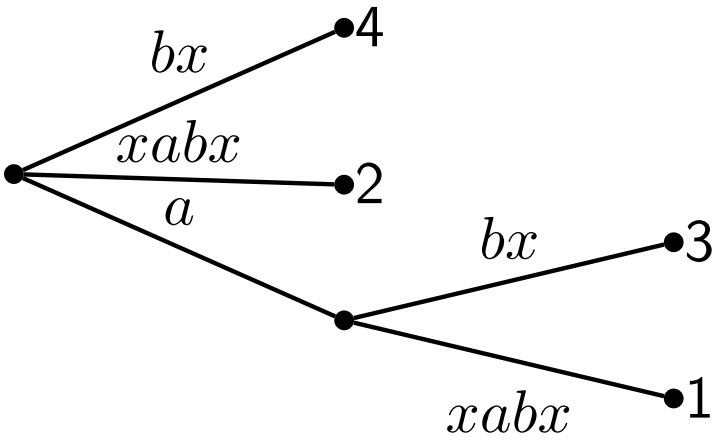


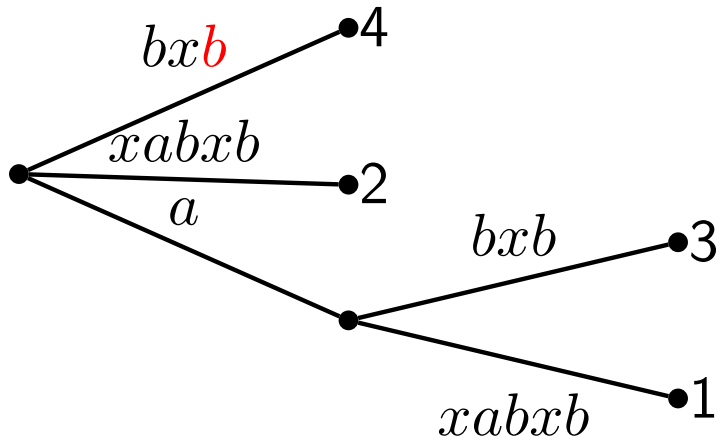




Суффикс `abxb`

Правило №1, путь $\beta$ заканчивается в листе: добавляем $S(i+1)$ к метке последней дуги.

# Добавление $b$ в неявное дерево для строки $axabx$

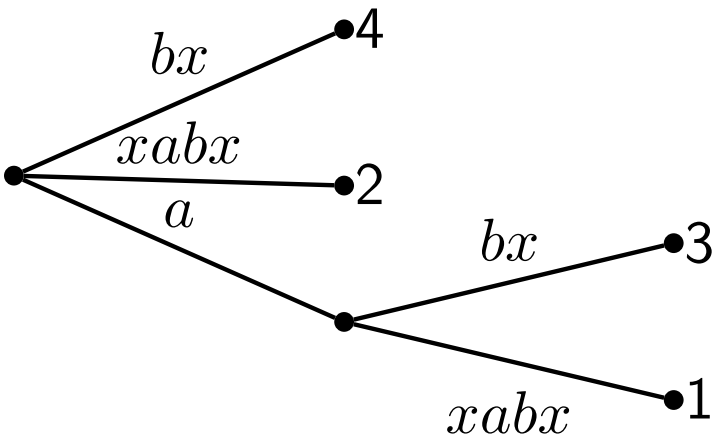


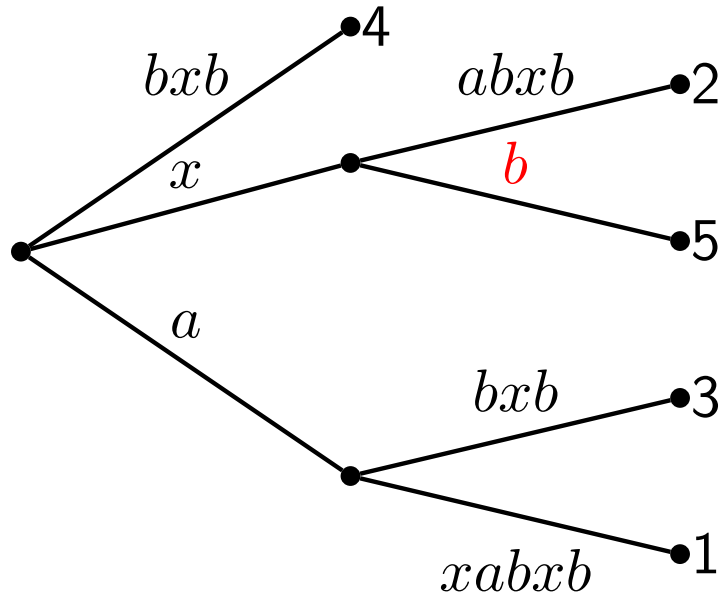




Суффикс bxb

Правило №1, путь $\beta$ заканчивается в листе: добавляем $S(i+1)$ к метке последней дуги.

# Добавление $b$ в неявное дерево для строки $axabx$

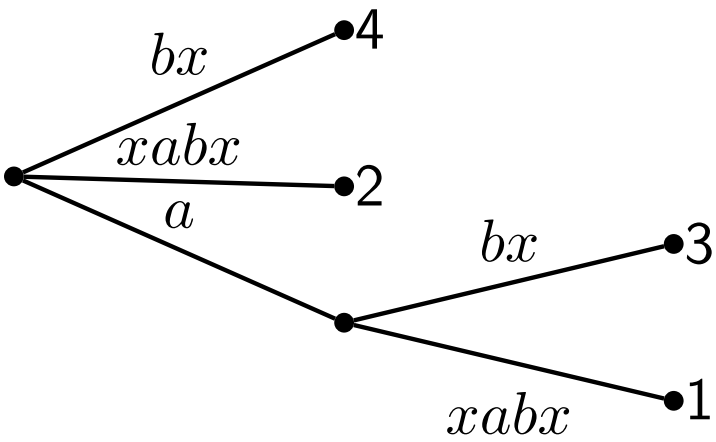


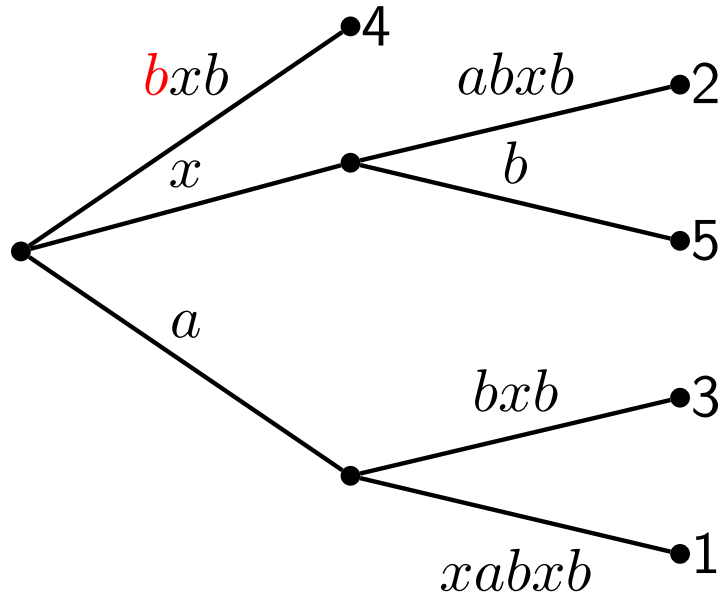




Суффикс `xb`

Правило №2, путь $\beta$ продолжается символами, отличными от $S(i+1)$: создаём новую вершину.

# Добавление $b$ в неявное дерево для строки $axabx$




Суффикс b

Правило №3, путь $\beta S(i+1)$ уже существует: ничего не делаем.

# Время работы

- $m$ фаз.
- В каждой $i$-й фазе $i + 1$ продолжение.
- Каждое продолжение — поиск от корня окончания пути $\beta$, максимум $|\beta|$ операций.
- Само продолжение — константное время.
- Следовательно, время работы $O(m^3)$.

# Время работы

- $m$ фаз.
- В каждой $i$-й фазе $i+1$ продолжение.
- Каждое продолжение — поиск от корня окончания пути $\beta$, максимум $|\beta|$ операций.
- Само продолжение — константное время.
- Следовательно, время работы $O(m^3)$.
- Нужно найти более быстрый метод определения места следующего продолжения (переход от $S[j..i+1]$ к $S[j+1..i+1]$).

# Раздел

# Создание суффиксных связей

## Теорема

*Если в продолжении $j$ фазы $i+1$ добавляется новая внутренняя вершина $v$ с путевой меткой $x\alpha$, то путь с меткой $\alpha$ либо уже заканчивается в какой-то внутренней вершине дерева, либо новая вершина в конце $\alpha$ будет создана в продолжении $j+1$ той же фазы $i+1$.*

## Доказательство.

- Выполняется правило 2, следовательно существует путь $x\alpha c$, где $c \neq S(i+1)$.
- Следовательно, уже существует путь $\alpha c$.
- Если существует путь $\alpha d, d \neq c$, то нужная вершина в дереве уже есть. Иначе она создастся при добавлении $\alpha S(i+1)$, т.е. на следующем продолжении.

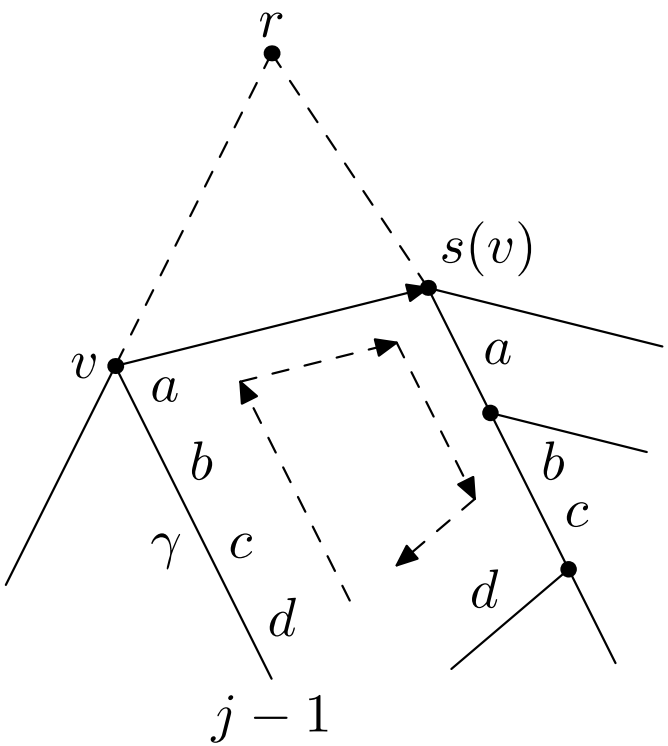
# Следствия

1. В алгоритме Укконена любая вновь созданная вершина получит суффиксную связь при выполнении следующего продолжения.
2. В любом неявном суффиксном дереве $\mathbb{T}_i$ если внутренняя вершина $v$ имеет путевую метку $x\alpha$, то найдётся вершина $s(v)$ дерева $\mathbb{T}_i$ с путевой меткой $\alpha$.

# Переходы по суффиксным связям при построении $\mathbb{T}_{i+1}$

- При последовательном выполнении алгоритма в конце $S[j..i]$ выполняем продолжение и затем нужно попасть в конец $S[j+1..i]$.
- Конец полной строки $S[1..i]$ можно всегда хранить (это лист, соответствующий самому длинному пути).
- Допустим, $S[1..i] = x\alpha$ и $\langle v, 1\rangle$ — дуга дерева, входящая в лист 1; нужно найти конец $S[2..i] = \alpha$.
- Если $v$ — корень, то нужно выполнить поиск прямым способом.
- Если же $v$ — внутренняя вершина, и путь $\langle v, 1\rangle$ помечен $\gamma$, то нужно пройти к $s(v)$ и оттуда проследовать вдоль $\gamma$, конец этого пути будет концом и $\alpha$.

# $j$-ое продолжение




1. Поднимаемся вверх не более чем на одну дугу (с меткой $\gamma$) к вершине $v$.
2. Переходим по суффиксной связи в $s(v)$.
3. Опускаемся по пути, определяемому подстрокой $\gamma$.

# Алгоритм отдельного продолжения

Продолжение $j \geq 2$ фазы $i + 1$:

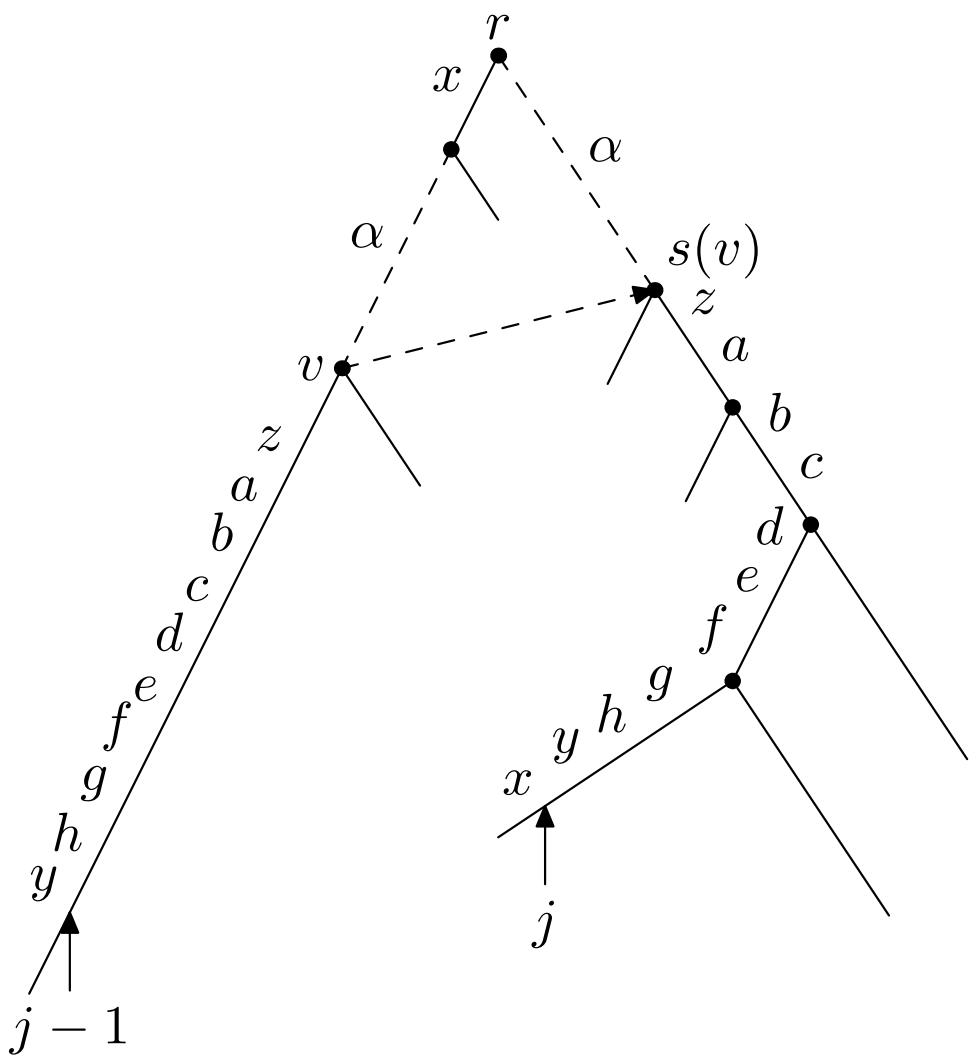
1. Найти в конце строки $S[j-1..i]$ или выше первую вершину $v$, которая либо имеет суффиксную связь, либо является корнем. $\gamma$ — строка между $v$ и концом $S[j-1..i]$, возможно, пустая.
2. Если $v$ — не корень, пройти в $s(v)$ и спуститься оттуда по пути $\gamma$. Если $v$ — корень, пройти по пути $S[j..i]$.
3. Выполнить правила продолжения (обеспечить вхождение $S[j..i+1]$).
4. Если в продолжении $j-1$ была создана вершина $w$ для $x\alpha$, то связать её суффиксной связью с концом строки $\alpha$, найденном в текущем продолжении.

Первое продолжение всегда начинается с сохранённого окончания $S[1..i]$.

# Что дало использование суффиксных связей?

- Явное практическое улучшение.
- Однако, оценка худшего случая не изменилась — одна фаза за $O(m^2)$, полное выполнение — $O(m^3)$.
- Продолжаем дальше: избавляемся от ненужных сравнений символов, для поиска пути можно перейти от времени пропроциональному $|\gamma|$ к времени, пропорциональному количеству вершин на пути.

# Прыжок по счётчику

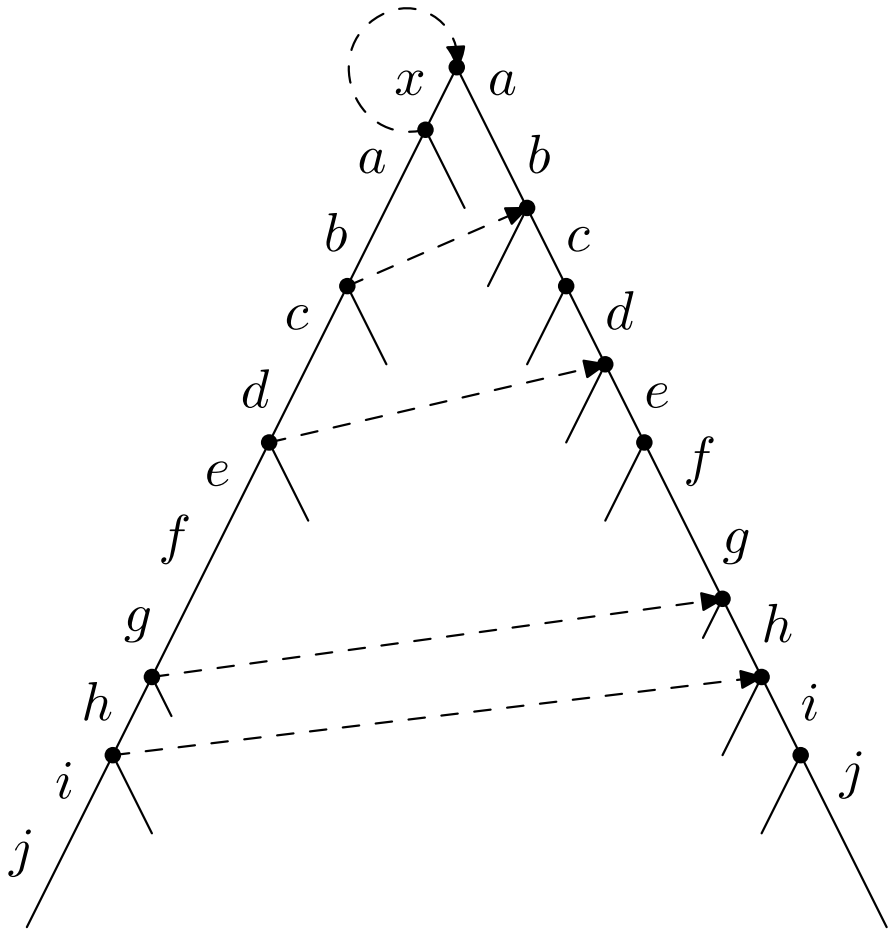



- Пусть $g = |\gamma|$ и $g'$ — количество символов на дуге выходящей из $s(v)$ и помеченной первым символом $\gamma$.
- Если $g' < g$, то остальные символы дуги просматривать не надо, можно сразу перейти к следующей вершине.

# Вершинная глубина

- Полное время прохода по пути пропрционально числу вершин, а не символов.
- Вершинная глубина узла $v$ — число вершин на пути до неё от корня, $h(v)$.
- Текущая вершинная глубина — глубина последней по времени вершины, посещённой алгоритмом.

# Вершинная глубина суффиксной связи




## Теорема

*Пусть* $\langle v, s(v)\rangle$ — *суффиксная связь, проходимая при выполнении алгоритма. В этот момент* $h(v) \leq h(s(v)) + 1$.

# Время выполнения одной фазы

## Теорема

*При использовании прыжков по счётчику любая фаза алгоритма Укконена занимает время $O(m)$.*

## Доказательство.

1. Подъём по дуге может уменьшить текущую вершинную глубину на 1, проход по суффиксной связи так же может уменьшить её не более чем на 1, а каждая дуга при спуске увеличивает вершинную глубину.
2. Тем самым, за всю фазу текущая глубина уменьшается не более $2m$ раз.
3. Отсюда, т.к. глубина любой вершины меньше $m$, приращение текущей глубины не превосходит $3m$.

# Время выполнения алгоритма

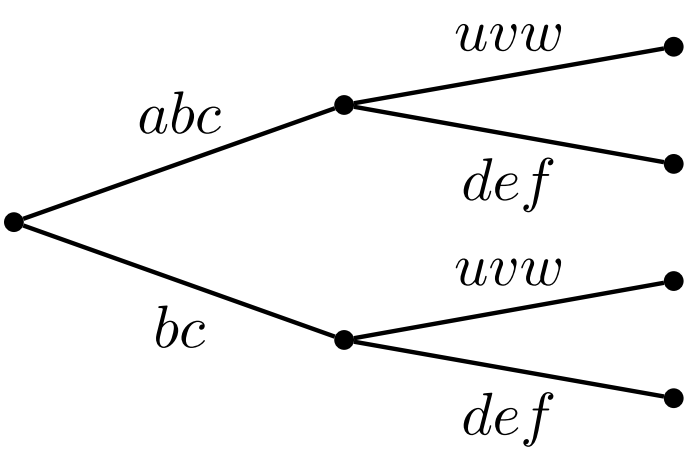
- Суффиксные связи в алгоритме Укконена обеспечивают время работы $O(m^2)$.
- Если хранить строки на дугах, то требуется объём памяти $\Theta(m^2)$.
- Для перехода к линейному алгоритму требуется иное представление данных в дереве.

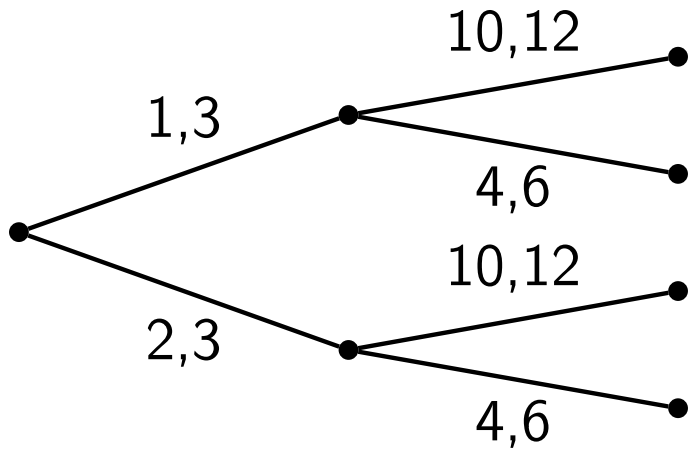
# Раздел

# Сжатие дуговых меток

Вместо выписывания подстрок достаточно хранить пару индексов, определяющую начальную и конечную позиции этой подстроки в $S$, откуда, имея доступ к $S$, всегда можно получить нужные символы.

Например, $S = abcdefabcuvw$



# Третье правило завершает фазу

- Правило №3: если уже существует путь $S[j..i+1]$, то ничего делать не надо.
- Однако, если есть путь $S[j..i+1]$, то есть $S[j+1..i+1]$ и т.п.
- Следовательно, для всех следующих продолжений будет выполняться третье правило.
- Таким образом, фазу можно заканчивать при первом выполнении третьего правила.

# Листья остаются листьями

- Если был создан лист с меткой $j$, то он останется листом вплоть до окончания алгоритма.
- То есть, во всех следующих фазах для этого продолжения будет применяться первое правило (дописать символ в конец листовой дуги).
- Тем самым, можно помечать все листовые дуги не конкретным индексом, а глобальным индексом «текущего окончания строки» $e$, который увеличивать в начале фазы, выполняя неявно все продолжения по первому правилу.

# Общая идея линейного алгоритма

- Запоминается число $l$ — последний лист, созданный на $i$-й фазе.
- Выполнение всех первых правил происходит неявно, увеличивая $e$.
- Каждая фаза: последовательное применение второго правила (увеличивающее $l$) до первого срабатывания третьего правила.
- Тем самым алгоритм превращается в последовательное выполнение правил № 2.

# Алгоритм одной фазы $i + 1$

1. $e \leftarrow i + 1$
2. Вычислить все последовательные продолжения от $l$ до $r$, где применяется третье правило или до конца фазы.
3. $l \leftarrow r$

# Линейность алгоритма

## Теорема

*Используя суффиксные связи и все улучшения, алгоритм Укконена строит неявные суффиксные деревья от $\mathbb{T}_1$ до $\mathbb{T}_m$ за полное время $O(m)$.*

## Доказательство.

1. $j'$ — продолжение, явно выполняемое алгоритмом. $j'$ не убывает и не изменяется при переходе от фазы к фазе, фаз $m$ и $j' \leq m$, следовательно количество продолжений не более $2m$.
2. При этом текущая вершинная глубина не изменяется при переходе от фазы к фазе, то и максимальное количество прыжков для всех фаз имеет порядок $O(m)$.

# Создание настоящего суффиксного дерева

- Нужно добавить терминальный символ (выполнить ещё одну фазу).
- Заменить глобальный индекс $e$ числом $m$.
- Тем самым, алгоритм Укконена строит настоящее суффиксное дерево для $S$ и всего его суффиксные связи за время $O(m)$.