Algorithmentheorie

15 – Suchen in Texten (1)

Prof. Dr. S. Albers

Suche in Texten

Verschiedene Szenarien:

Statische Texte

- Literaturdatenbanken
- Bibliothekssysteme
- Gen-Datenbanken
- WWW-Verzeichnisse

Dynamische Texte

- Texteditoren
- Symbolmanipulatoren

Eigenschaft von Suffix-Bäumen

Suchindex

zu einem Text σ für Suche nach verschiedenen Mustern α

Eigenschaften:

1. **Teilwortsuche** in Zeit O(|α|).

2. **Anfragen an σ selbst**, z.B.:
   Längstes Teilwort von σ, das an mind. 2 Stellen auftritt.

3. **Präfix-Suche:** Alle Stellen in σ mit Präfix α.

Eigenschaft von Suffix-Bäumen

4. **Bereichssuche:** Alle Stellen in σ im Intervall $[\alpha, \beta]$ mit $\alpha \leq_{lex} \beta$, z.B.

   abrakadabra, acacia $\in$ [abc, acc],
   abacus $\notin$ [abc, acc] .

5. **Lineare Komplexität:**
   Speicherplatzbedarf und Konstruktionszeit $\in O(|\sigma|)$

Tries

Trie: Baum zur Repräsentation von Schlüsseln.

Alphabet Σ, Menge S von Schlüsseln, $S \subset \Sigma^*$

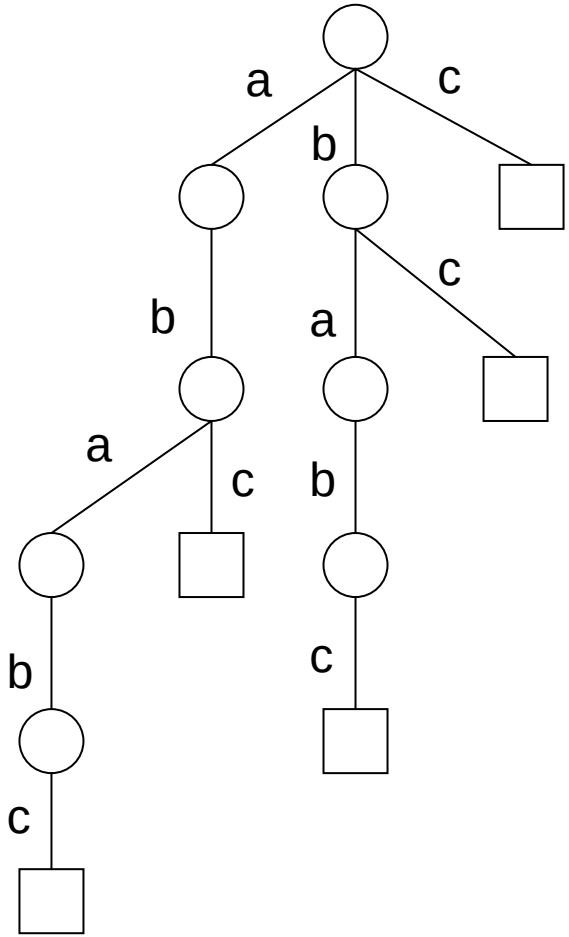
Schlüssel $\hat{=}$ Zeichenkette aus Σ^*

Kante eines Tries T: Beschriftung mit einzelnen Zeichen aus Σ

benachbarte Kanten: verschiedene Zeichen

Tries

Beispiel:



Tries

Blatt repräsentiert Schlüssel:

Entspricht Beschriftung der Kanten des Weges von der Wurzel zum Blatt.

! Schlüssel werden nicht in Knoten gespeichert !

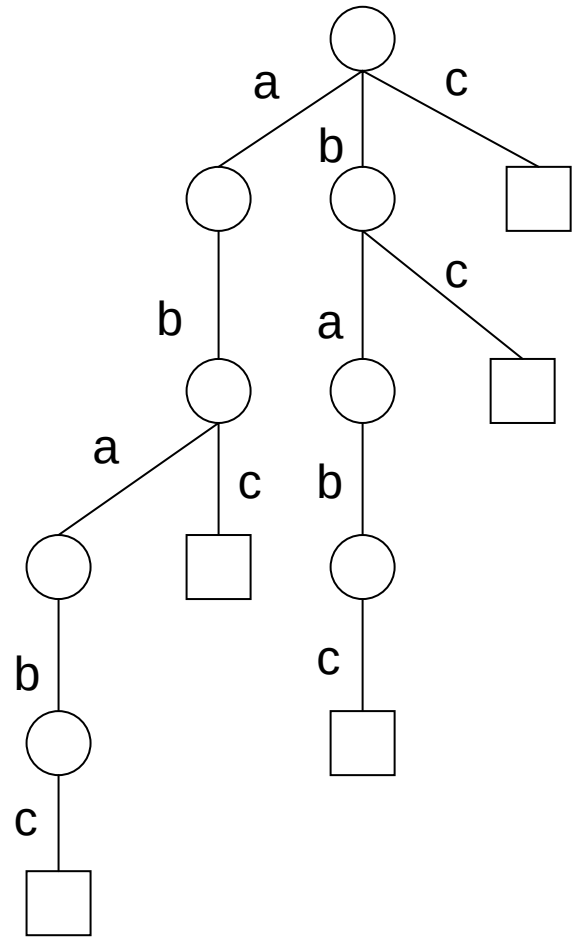
Suffix-Tries

Trie für alle Suffixe eines Wortes

Beispiel: σ = ababc

Suffixe:

ababc = suf_1
babc = suf_2
abc = suf_3
bc = suf_4
c = suf_5



Suffix-Tries

Innere Knoten eines Suffix-Tries = Teilwort von σ.

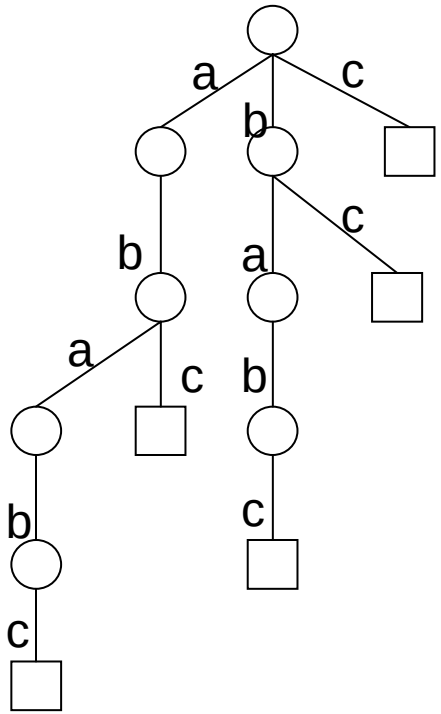
Jedes echte Teilwort von σ ist als innerer Knoten repräsentiert.

Sei $\sigma = a^n b^n$: $\exists$ $n^2 + 2n + 1$ verschied. Teilwörter = innere Knoten

$\Rightarrow$ Speicherplatzbedarf $\in O(n^2)$.

Suffix-Tries

Ein Suffix-Trie T erfüllt bereits einige der geforderten Eigenschaften:



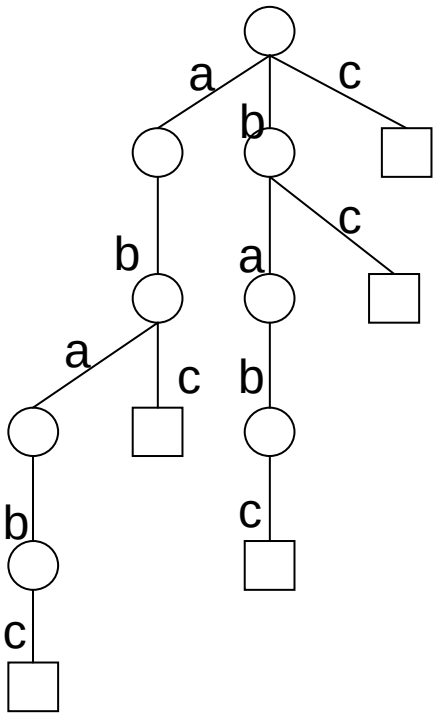
1. Zeichenkettensuche nach α : Folge dem Weg mit Kantenbeschriftung α in T in Zeit $O(|\alpha|)$. Blätter des Teilbaumes $\hat{=}$ Vorkommen von α

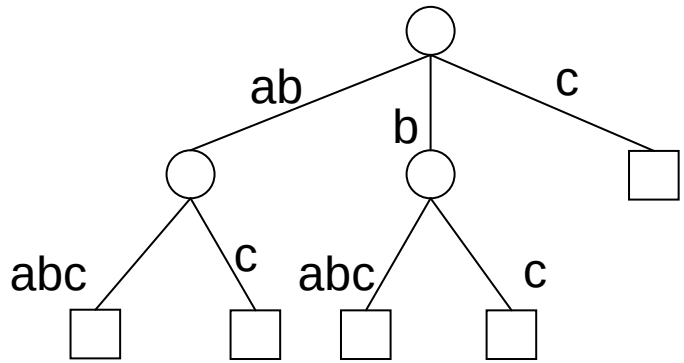
2. Längstes, doppelt auftretendes Wort: Innerer Knoten mit größter Tiefe, der mind. zwei Söhne hat.

3. Präfix-Suche: alle Vorkommen von Zeichenketten mit Präfix α finden sich in dem Teilbaum unterhalb des inneren Knotens von α in T.

Suffix-Bäume

Suffix-Baum entsteht durch Kontraktion von unären Knoten aus Suffix-Trie:





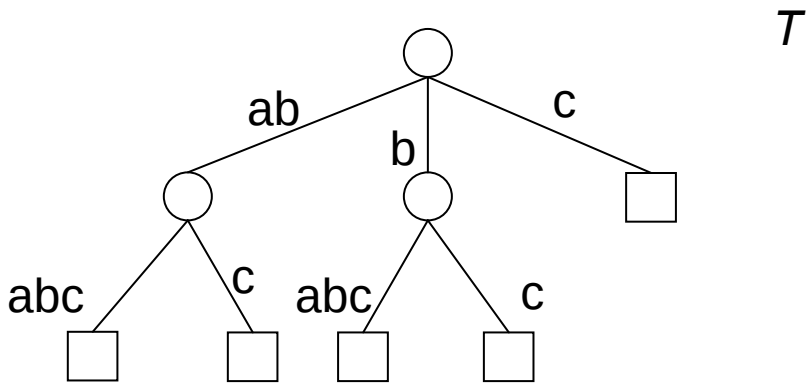
Suffix-Baum = kontraktierter Suffix-Trie

Interne Repräsentation von Suffix-Bäumen

Sohn/Bruder-Repräsentation

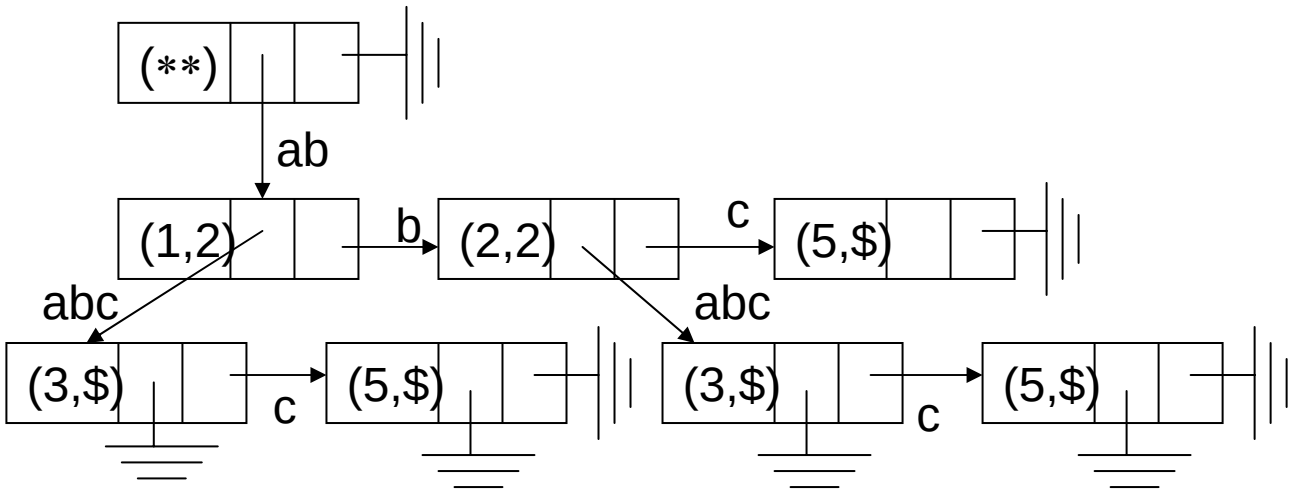
Teilwort: Zahlenpaar (*i,j*)

Beispiel: σ = ababc



Interne Repräsentation von Suffix-Bäumen

Beispiel σ = ababc



Knoten $v = (v.u, v.o, v.sn, v.br)$

Weitere Zeiger (Suffix-Zeiger) kommen später hinzu.

Eigenschaften von Suffix-Bäumen

(S1) Kein Suffix ist Präfix eines anderen Suffixes;
gilt, falls (letztes Zeichen von σ) = \$ $\notin \Sigma$

Suche:

(T1) Kante $\hat{=}$ nichtleeres Teilwort von σ.

(T2) Benachbarte Kanten: zugeordnete Teilworte beginnen mit verschiedenen Zeichen.

Eigenschaften von Suffix-Bäumen

Größe

(T3) Innerer Knoten ($\neq$ Wurzel): mind. zwei Söhne.

(T4) Blatt $\hat{=}$ (nicht-leeres) Suffix von σ.

Sei $n = |\sigma| \neq 1$

$\overset{(T4)}{\Rightarrow}$ Anzahl der Blätter : n

$\overset{(T3)}{\Rightarrow}$ Anzahl der inneren Knoten $\leq n-1$

$\Rightarrow$ Speicherplatz $\in O(n)$

Konstruktion von Suffix-Bäumen

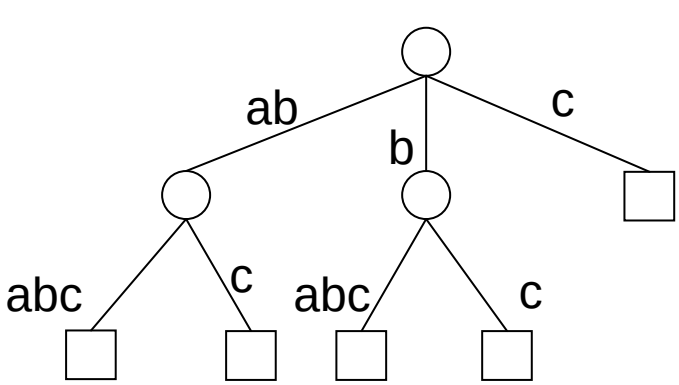
Definition:

partieller Weg: Weg von der Wurzel zu einem Knoten von *T*

Weg: Ein partieller Weg, der bei einem Blatt endet.

Ort einer Zeichenkette α : Knoten am Ende des mit α beschrifteten partiellen Weges (falls er existiert).

T

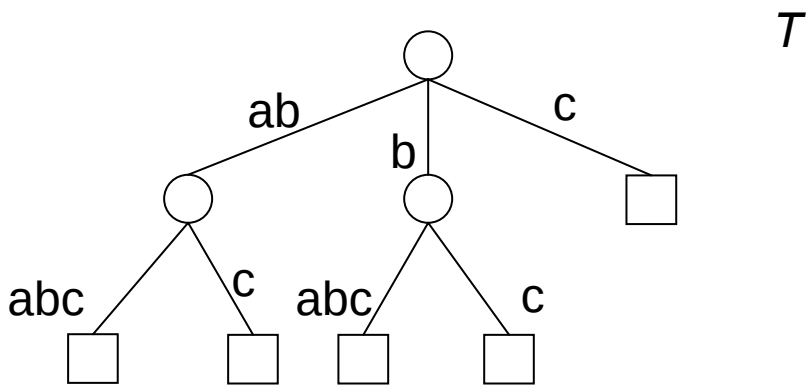


Konstruktion von Suffix-Bäumen

Erweiterung einer Zeichenkette α : Zeichenkette mit Präfix α

erweiterter Ort einer Zeichenkette α : Ort der kürzesten Erweiterung von α, deren Ort definiert ist.

kontrahierter Ort einer Zeichenkette α : Ort des längsten Präfixes von α, dessen Ort definiert ist.



Konstruktion von Suffix-Bäumen

Definitionen:

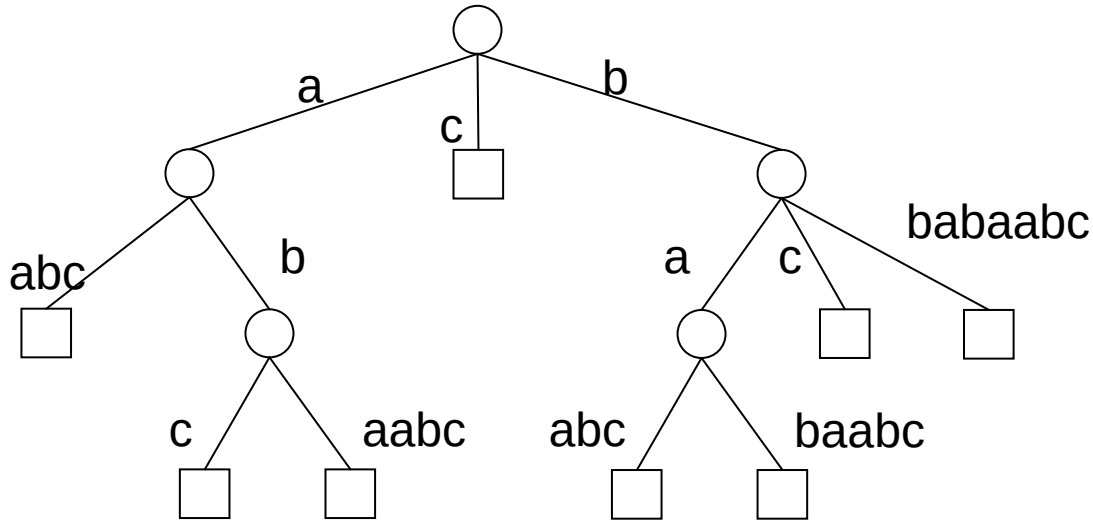
suf_i : an Position i beginnendes Suffix von σ , also z.B.
$suf_1 = \sigma$, suf_n = \$.

$head_i$: längstes Präfix von suf_i , das auch Präfix von suf_j für ein $j < i$ ist.

Beispiel: σ = bbabaabc α = baa (hat keinen Ort)
suf_4 = baabc
$head_4$ = ba

Konstruktion von Suffix-Bäumen

σ = bbabaabc



Naive Suffix-Baum-Konstruktion

Beginne mit dem leeren Baum T_0

Der Baum T_{i+1} entsteht aus T_i durch Einfügen des Suffixes suf_{i+1}.

Algorithmus Suffix-Baum

Input: Eine Zeichenkette σ

Output: Der Suffix-Baum T von σ

1 n := | σ |; T_0 := ∅;

2 **for** i := 0 **to** n – 1**do**

3 füge suf_{i+1} in T_i ein, dies sei T_{i+1} ;

4 **end for**

Naive Suffix-Baum-Konstruktion

In T_i haben alle Suffixe suf_j , $j \leq i$ bereits einen Ort.

→ $head_{i+1}$ = längstes Präfix von suf_{i+1}, dessen erweiterter Ort in T_i existiert.

Definition:

$tail_{i+1} := suf_{i+1} - head_{i+1}$, d.h. also $suf_{i+1} = head_{i+1}\, tail_{i+1}$.

$(S1)$

$\Rightarrow$ $tail_{i+1} \neq \varepsilon$.

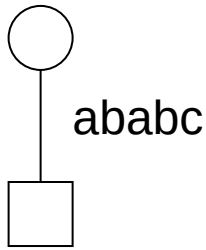
Naive Suffix-Baum-Konstruktion

Beispiel: σ = ababc

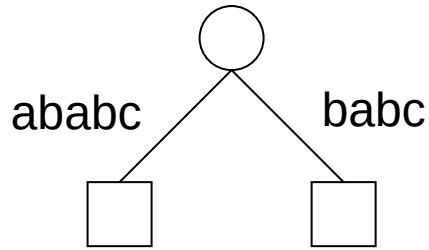
suf_3 = abc
$head_3$ = ab
$tail_3$ = c

T_0 =

T_1 =



T_2 =

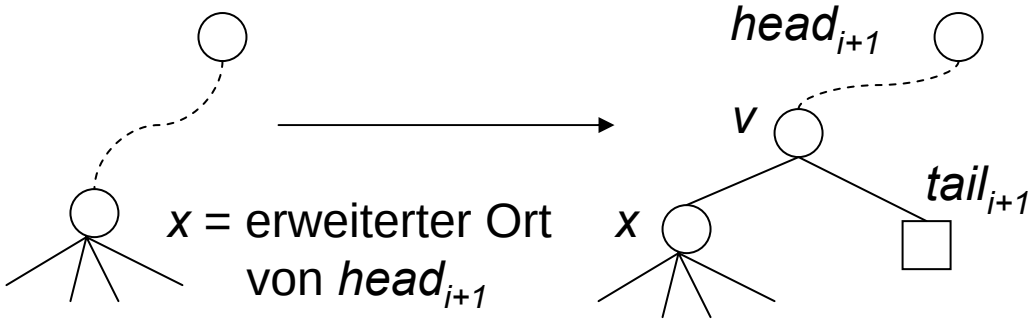


Naive Suffix-Baum-Konstruktion

T_{i+1} kann aus T_i wie folgt konstruiert werden:

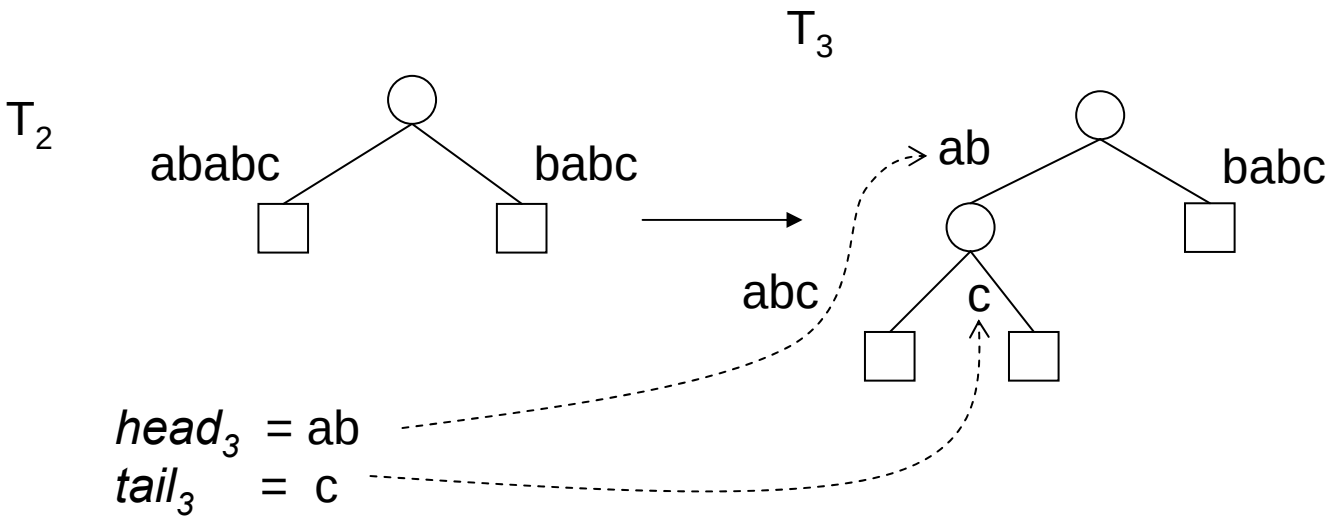
1. Man bestimmt den erweiterten Ort von $head_{i+1}$ in T_i und teilt die letzte zu diesem Ort führende Kante in zwei neue Kanten auf durch Einfügen eines neuen Knotens.

2. Man schaffe ein neues Blatt als Ort für suf_{i+1}



Naive Suffix-Baum-Konstruktion

Beispiel: σ = ababc



Naive Suffix-Baum-Konstruktion

Algorithmus Suffix-Einfügen

Input: Der Baum T_i und der Suffix suf_{i+1}

Output: Der Baum T_{i+1}

1 v := Wurzel von T_i

2 $j := i$

3 **repeat**

4 finde Sohn w von v mit $\sigma_{w.u} = \sigma_{j+1}$

5 $k := w.u - 1$;

6 **while** $k < w.o$ **and** $\sigma_{k+1} = \sigma_{j+1}$ **do**

7 $k := k + 1$; j := $j + 1$

end while

Naive Suffix-Baum-Konstruktion

8 **if** $k = w.o$ **then** $v := w$

9 **until** k $< w.o$ or w = nil

10 /* v ist konktraktierter Ort von $head_{i+1}$ */

11 füge den Ort von $head_{i+1}$ und $tail_{i+1}$ in T_i unter v ein

Laufzeit für Suffix-Einfügen: O( )

Gesamtlaufzeit für naive Suffix-Baum-Konstruktion: O( )

Der Algorithmus M

(Mc Creight, 1976)

Idee: Erweiterter Ort von $head_{i+1}$ wird in konstanter amortisierter Zeit in T_i bestimmt. (Zusatzinformation erforderlich!)

Falls erweiterter Ort von $head_{i+1}$ in T_i gefunden: Erzeugen eines neuen Knotens und Aufspalten einer Kante $\in$ O(1) Zeit.

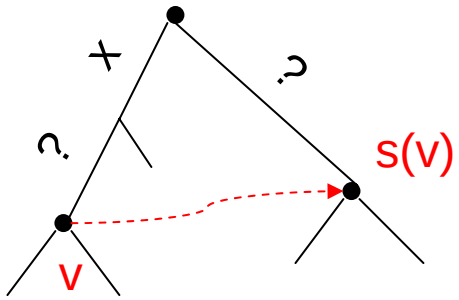
Theorem 1

Algorithmus *M* liefert in Zeit $O(|\sigma|)$ einen Suffix-Baum für σ mit $|\sigma|$ Blättern und höchstens $|\sigma|$ - 1 inneren Knoten.

Suffix-Links

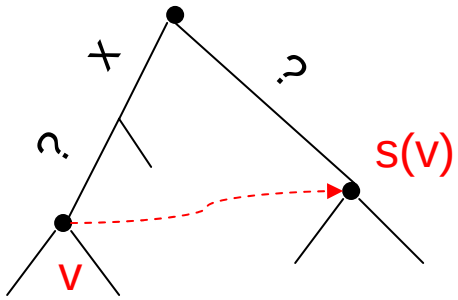
Definition: Sei *x*? ein beliebiger String, wobei *x* ein einzelnes Zeichen darstellt und ? einen (möglicherweise leeren) Teilstring.

Für jeden inneren Knoten *v* mit Kennzeichnung *x*? gilt: Falls es einen weiteren Knoten *s*(*v*) mit Pfad-Markierung ? gibt, so gibt es einen Zeiger von *v* auf *s*(*v*), der als Suffix-Link bezeichnet wird.



Suffix-Links

Die Idee ist, Nutzen aus den Suffix-Links zu ziehen, um die Erweiterungspunkte effizienter, d.h. in amortisiert konstanter Zeit, zu finden, ohne bei jeder expliziten Erweiterung an der Wurzel beginnen zu müssen.



Suffix-Baum Beispiel

T_0 =

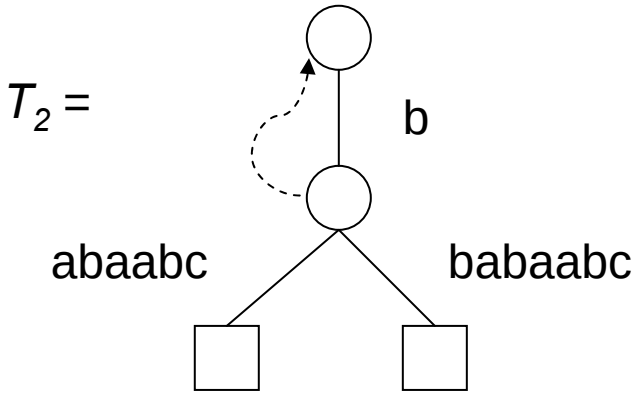
T_1 =

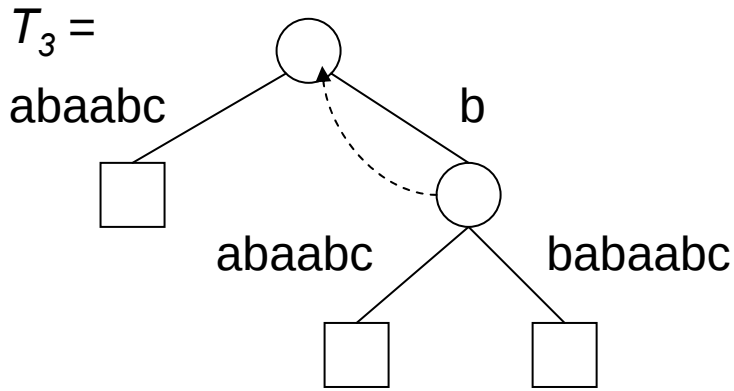
bbabaabc

suf_1 = bbabaabc

suf_2 = babaabc
$head_2$ = b

Suffix-Baum Beispiel

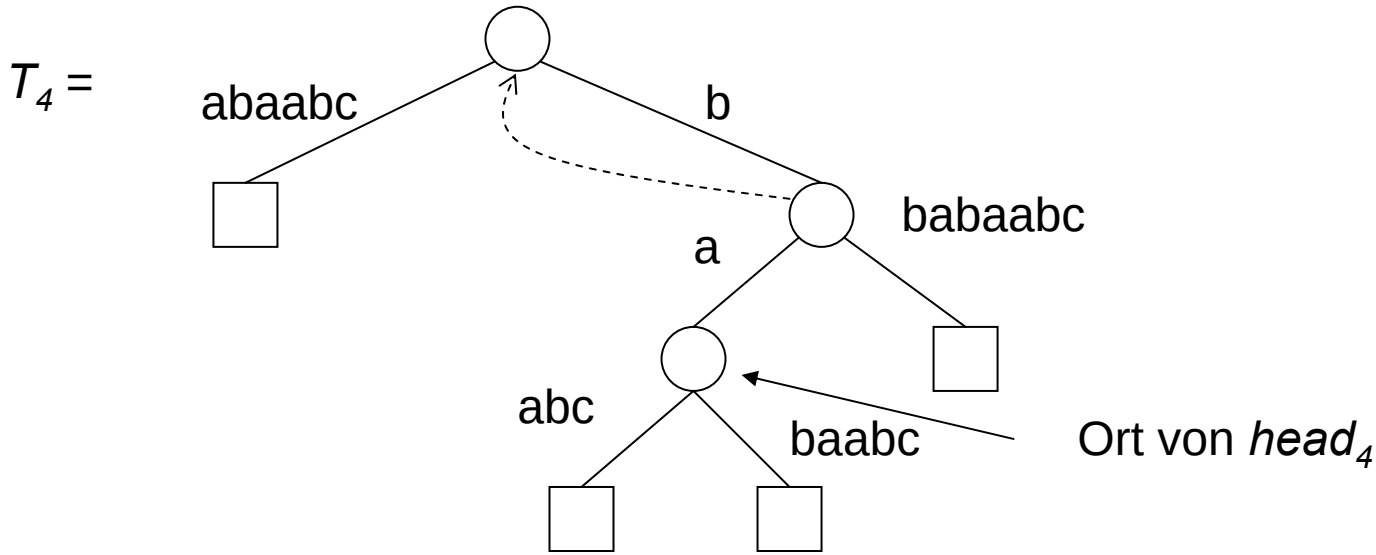




suf_3 = abaabc
$head_3$ = ε

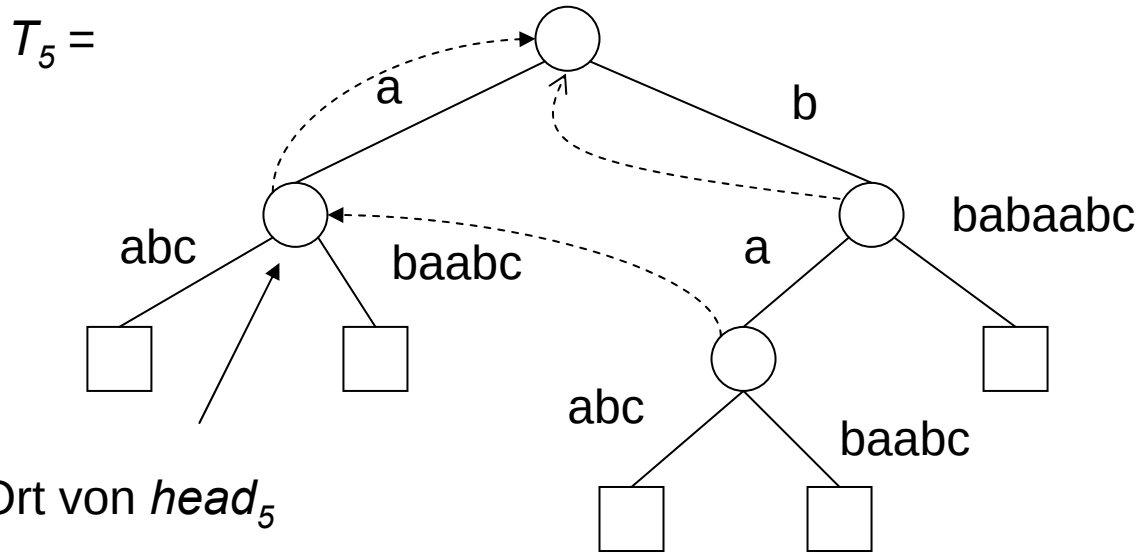
suf_4 = baabc
$head_4$ = ba

Suffix-Baum Beispiel



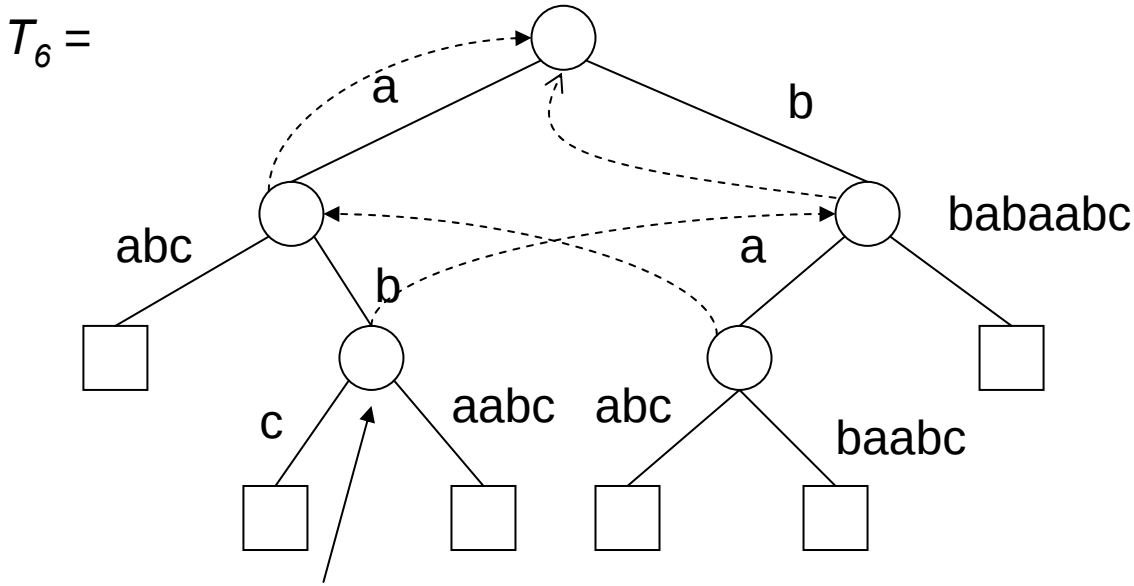
suf_5 = aabc
$head_5$ = a

Suffix-Baum Beispiel



suf_6 = abc
$head_6$ = ab

Suffix-Baum Beispiel

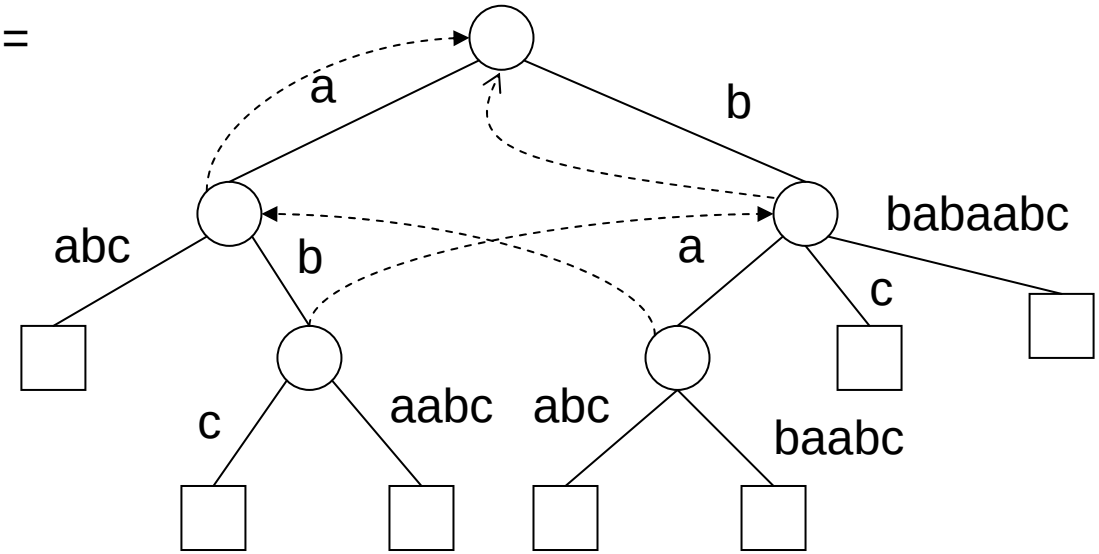


Ort von $head_6$

suf_7 = bc
$head_7$ = b

Suffix-Baum Beispiel

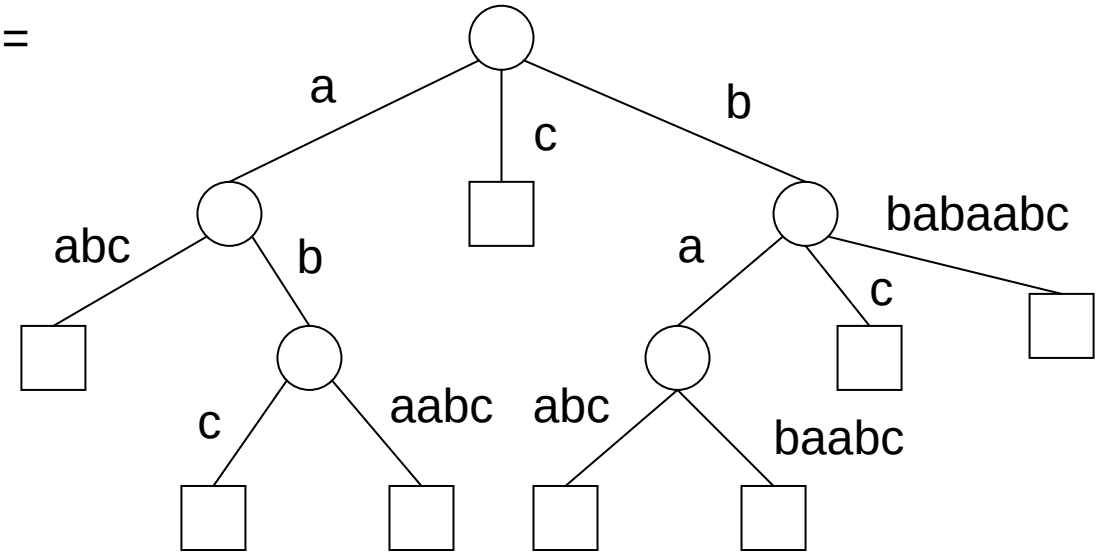
T_7 =



suf_8 = c

Suffix-Baum Beispiel

T_8 =



Suffix-Baum Anwendung

Verwendung von Suffix-Baum *T*:

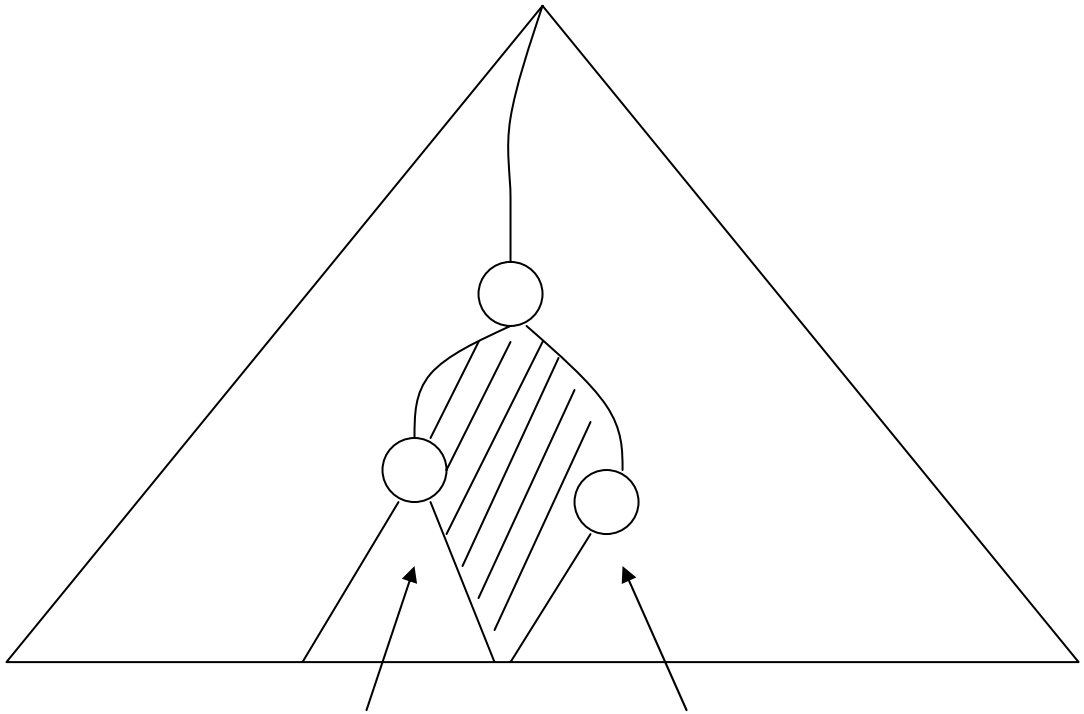
1 Suche nach Zeichenkette α: Folge dem Weg mit Kantenbeschriftung α in *T* in Zeit O(| α |). Blätter des Teilbaumes $\hat{=}$ Vorkommen von α

2 Suche längstes, doppelt auftretendes Wort: Finde Ort eines Wortes mit größter gewichteter Tiefe, der innerer Knoten ist.

3 Suche nach Präfix: Alle Vorkommen von Zeichenketten mit Präfix α finden sich in dem Teilbaum unterhalb des „Ortes“ von α in *T*.

Suffix-Baum Anwendung

4 Bereichssuche nach $[\alpha, \beta]$:



Bereichsgrenzen