Wörter

Notation:

Σ	ist ein (meist endliches) Alphabet
Σ^*	Menge aller endlichen Folgen $a_1 \cdots a_n$ mit $a_i \in \Sigma$
Σ^ω	Menge aller unendlichen Folgen $a_1 a_2 \cdots$ mit $a_i \in \Sigma$
$t \in \Sigma^*$	(fester) Text über dem Alphabet
$F(t)$	Menge aller *Faktoren*, d.h. Teilwörter, von t
	w *teilt* t, $w \mid t :\Longleftrightarrow \exists u \in \Sigma^* \exists v \in \Sigma^* (t = uwv)$
$\lvert t \rvert$	Länge n des Texts $t = a_1 \cdots a_n$
$Suf(t)$	Menge aller Suffixe von t,
	w ist ein *Suffix* von t, falls $\exists v \in \Sigma^*\ t = vw$

Graphen und Bäume

Ein endlicher *Graph* $G = (V, E)$ ist eine Menge V von Ecken (vertices) mit einer Relation $R \subseteq V \times V$ von *Kanten* (edges).

Ein *Weg der Länge* n von u nach v in G ist eine Folge $v_1, v_2, \ldots, v_n$ von Ecken, sodaß $u = v_1$, $v = v_n$ und $E(v_i, v_{i+1})$ für alle $i = 1, \ldots, n-1$.

Markierte Bäume

Ein *Baum* ist ein Graph mit einer ausgezeichneten Ecke, der *Wurzel*, von der aus jeder Knoten durch genau einen Weg erreichbar ist.

Ein (kanten-)*markierter* Baum ist ein Baum mit einer Abbildung m der Kanten in eine Menge Σ von *Marken*.

Suchbäume: Tries = re*trie*val trees

Ein Σ-*Trie* ist ein Baum T,

(i) dessen Kanten mit Elementen $a \in \Sigma$ markiert sind,

(ii) in dem von jedem Knoten n höchstens eine mit $a \in \Sigma$ markierte Kante ausgeht,

(iii) in dem eine Menge von *wesentlichen* oder *End*-Knoten ausgezeichnet sind, die ggf. mit Information markiert sind.

Dargestellte Wortmenge

Jedem Knoten n des Tries T entspricht das Wort w_n, das sich aus der Verkettung der Marken der Kanten des Pfads von der Wurzel bis zum Knoten ergibt.

Ein Σ-Trie T stellt die endliche Wortmenge $L(T) \subseteq \Sigma^*$ all der Wörter dar, die den wesentlichen Knoten von T entsprechen.

In T sind genau dann alle Knoten wesentlich, wenn $L(T)$ unter Präfixen abgeschlossen ist.

Suffix-Trie eines Texts

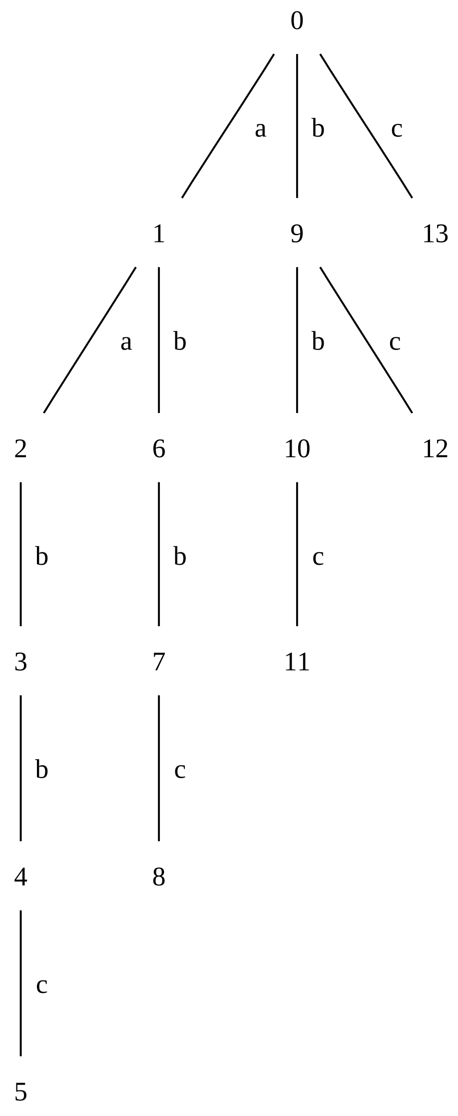
Der *Suffix-Trie* eines Texts t ist der Trie, der die Menge $Suf(t)$ der Suffixe von t darstellt.

Macht man alle Knoten des Suffix-Baums von t zu wesentlichen Knoten, so stellt er die Menge $F(t)$ aller Teilwörter von t dar.

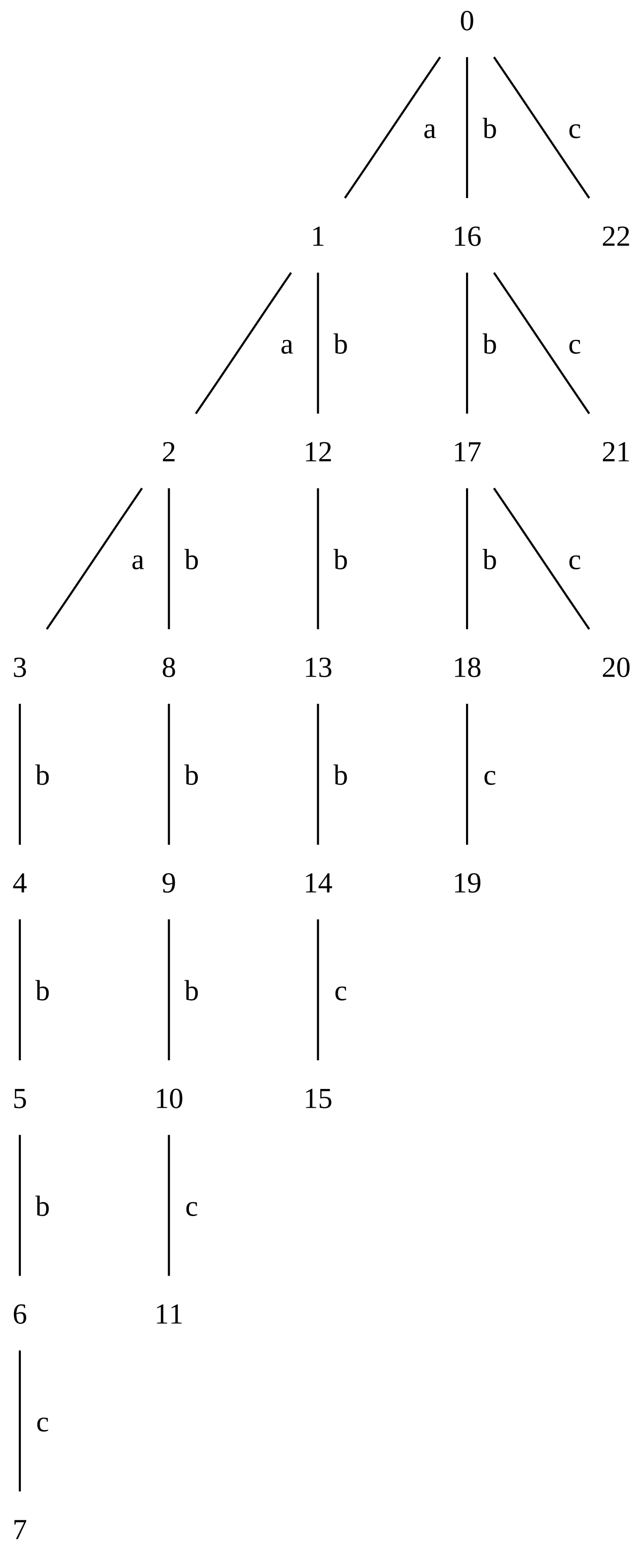
Kommt das letzte Symbol des Texts t sonst nicht in t vor, so sind die wesentlichen Knoten des Suffixbaums genau die Blätter; sie werden dann in der Darstellung nicht ausgezeichnet.

Der Suffixbaum eines Texts t hat i.a. $O(|t|^2)$ Knoten.

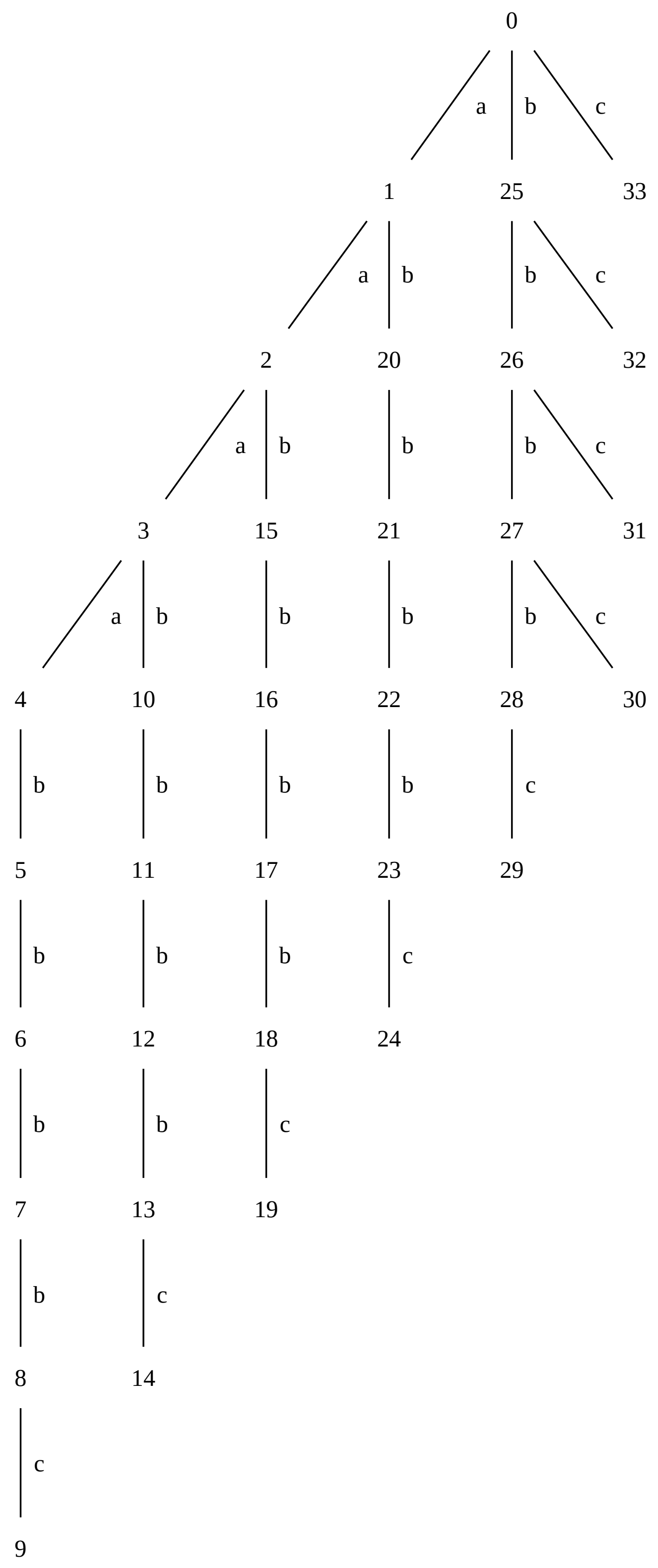
Suffix-Trie für `aabbc`



Suffix-Trie für `aaabbbc`



Suffix-Trie für `aaaabbbbc`



Konstruktion des Suffix-Tries von t

Das Alphabet Σ sei fest, d.h. nicht Teil der Eingabe.

Eingabe: $t = a_1 \cdots a_n \in \Sigma^*$.
Ausgabe: Suffix-Trie $T(t)$.

Naives Vorgehen

(i) Hänge ein *neues* Symbol \$ ans Ende von t an (damit nur die Blätter wesentliche Knoten werden),

(ii) Füge in $T(\epsilon)$ nacheinander die Suffixe von t ein.

Kosten:

(i) Wegen dem Endsymbol gibt es $|t| + 1$ Suffixe $\neq \epsilon$;

(ii) Einfügen des Suffixes $a_{n-k} \cdots a_n\$$ der Länge k braucht pro Buchstaben a höchstens $|\Sigma|$ Vergleiche, um den a-Nachfolger eines Knotens zu finden bzw. zu konstruieren;

Also $O(k)$ Schritte, um das Suffix einzufügen.

(iii) Insgesamt braucht man also $\sum_{k=0}^{|t|+1} O(k) = O(|t|^2)$ Schritte.

Verbesserung durch Suffix-Zeiger

(i) Konstruiere (*on-line*) die Suffix-Tries $T(\epsilon)$, $T(a_1)$, ..., $T(a_1 \cdots a_n)$ jeweils durch Ändern des vorigen Tries.

(Hält man alle Tries getrennt, braucht man $\sum_{k=0}^{n} O(k^2) = O(n^3)$ Platz, also auch mindestens soviel Zeit.)

(ii) In $T_i := T(a_1 \cdots a_i)$ entspreche der (wesentliche) Knoten v_k dem Suffix $suf_k(a_1 \cdots a_i) = a_{i-k+1} \cdots a_i$ der Länge k.

Lege in T_i *Suffix-Zeiger* vom Knoten v_k zum Knoten v_{k-1} an, für $k = 1, \ldots, i$, um v_{k-1} von v_k in $O(1)$ Schritten erreichen zu können.

(iii) Beachte: „neue“ Suffixe von T_i sind die Suffixe aus T_{i-1} um a_i fortgesetzt, die nicht schon in T_{i-1} vorkommen.

Konstruiere T_i aus T_{i-1}: folge den Suffix-Zeigern entlang der sog. *Suffix-Kette* $v_{i-1}, v_{i-2}, v_{i-3}, \ldots, v_0$ von T_{i-1} und

(a) füge an $v_j \in$ *Suffix-Kette* einen a_i-Nachfolger v'_j an, falls es keinen gibt, und einen Suffix-Zeiger vom ggf. vorher eingefügten v'_{j+1} auf v'_j;

(b) falls v_j einen a_i-Nachfolger v'_j hatte, ergänze den Suffix-Zeiger von v'_{j+1} auf v'_j, und wenn v_j die Wurzel ist, auch einen Suffixzeiger von v'_j auf v_j;

(c) falls auch v_0 keinen a_i-Nachfolger hatte, bilde einen Suffixzeiger vom ergänzten v'_0 auf v_0.

Wesentliche Knoten von T_i sind die auf der neuen Suffixkette $v_i = v'_{i-1}, \ldots, v'_0, v_0$.

Satz 1 Der Suffix-Trie $T(t)$ von t kann in $O(|T(t)|)$ Schritten konstruiert werden, wobei $|T(t)|$ die Knotenanzahl ist.

Beweis zum Aufwand

Aufwand zur Konstruktion von T_i aus T_{i-1}:

(i) Merkt man sich den tiefsten Knoten v_{i-1} von T_{i-1} in einer Adresse, so ist man in $O(1)$ Schritten am Anfangsknoten der Suffixkette von T_{i-1}.

(ii) Das Anlegen eines a_i-Nachfolgers v'_j zu v_j der Suffixkette kostet $O(1)$ Schritte (da Σ fest ist).

(iii) Merkt man sich den zuletzt angelegten Knoten v'_{j+1} in einer Adresse, so kann man den Suffixzeiger von v'_{j+1} auf v'_j in $O(1)$ Schritten anlegen.

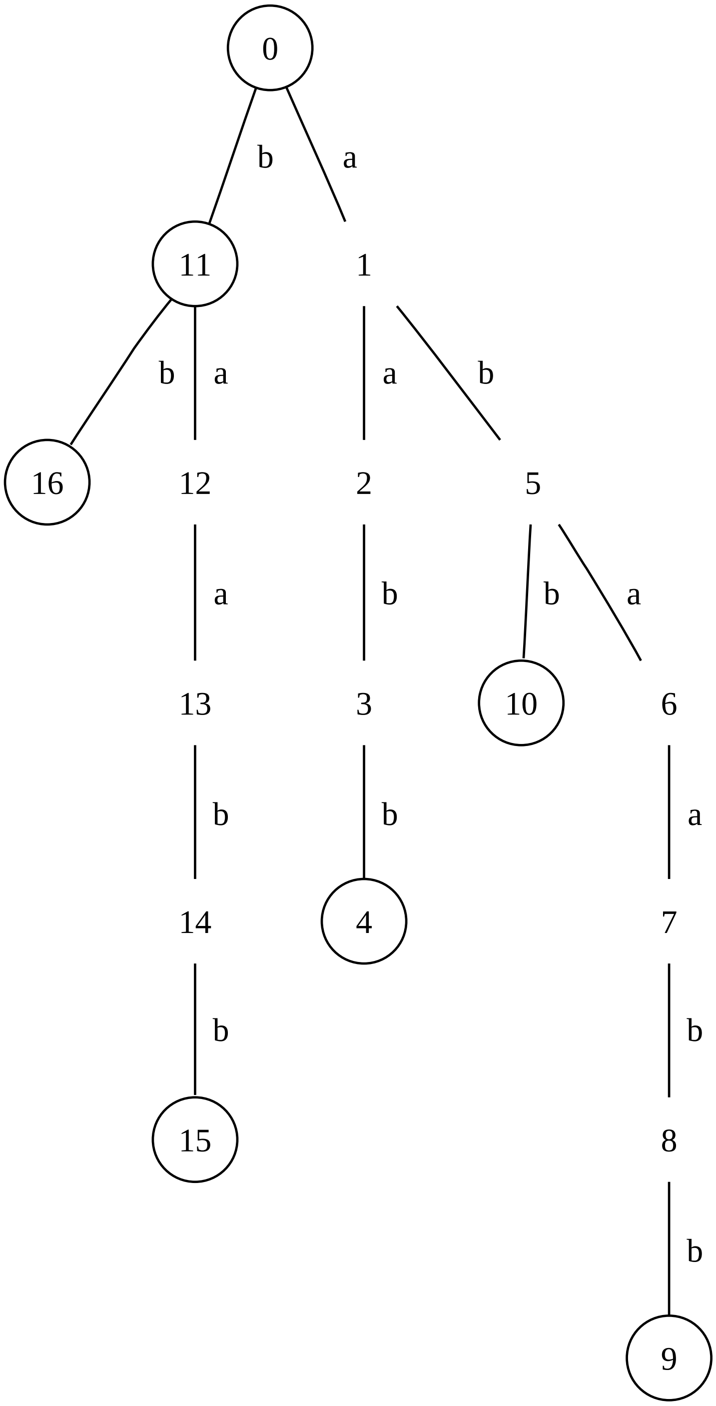
(iv) Man kann die obige Konstruktion abkürzen: falls v_j schon einen a_i-Nachfolger v'_j hatte, braucht man $v_{j-1}, \ldots, v_0$ nicht mehr zu betrachten und kann ab v'_j „alte“ Suffixzeiger von $T_1, \ldots, T_{i-1}$ verwenden.

Gesamtaufwand:

(v) Zum Schluß ($i = n$) durchlaufe in $O(|t|)$ Schritten die Suffixkette von T_n und erkläre die besuchten Knoten zu wesentlichen Knoten. Das macht T_n zu $T(t)$.

(vi) Wegen (iv) wird insgesamt jeder Knoten höchstens $|\Sigma|$ oft durchlaufen, und jedesmal nur $O(1)$ Schritte dabei getan, insgesamt also $O(|T(t)|)$ Schritte. $\square$

Beispiel: Suffix-Trie $T(abaabb)$



(Knotennummern *nicht* die der Einfügungsreihenfolge)