

Suffixbäume für Muster

Vorverarbeitung des Musters (der Muster)

KMP, Boyer-Moore

Matching Statistiken

Definition

$ms(i)$ = Länge des längsten Teilstrings von T , der an Position i beginnt und **irgendwo** in P vorkommt.

Vorkommen von P in T an Stelle i :

$$ms(i) = |P|$$

Definition

Für jede Position i von T bezeichne $p(i)$ eine Position in P mit $T[i..i+ms(i)] = P[p(i)..p(i)+ms(i)]$

Eine Anwendung

Gegeben:

Eine Menge von kurzen STS-Zeichenketten S_1, \dots, S_k und ein längeres sequenziertes DNA-Stück T .

Frage:

Welche der STS-Zeichenketten enthält T ?

Verfahren:

- generalisierter Suffix-Baum für S_1, \dots, S_k
- Berechnung der Match Statistiken für T und S_1, \dots, S_k zusammen mit $p(i)$
- **if** $p(i)$ gehört zu S_j **and** $ms(i) = |S_j|$
then S_j kommt in T vor

Alle Paare Suffix-Präfix Übereinstimmung (All-pair suffix-prefix matching)

Definition

Für zwei gegebene Zeichenketten S_i und S_j heißt jeder Suffix von S_i , der auch ein Präfix von S_j ist, eine **Suffix-Präfix Übereinstimmung** von S_i, S_j .

Beispiel $((S_i, S_j))$

$S_i = \text{abcxabcd}$ $S_j = \text{cdxabcxabc}$

Für eine gegebene Menge s von Zeichenketten S_1, \dots, S_k ist das **alle Paare Suffix-Präfix Problem** das Problem, für jedes geordnete Paar (S_i, S_j) aus s die **längste Suffix-Präfix Übereinstimmung** von (S_i, S_j) zu finden.

Anwendungen

Kürzeste gemeinsame Überfolge (Shortest common superstring)

Gegeben:

Eine Menge S_1, \dots, S_k von Zeichenketten

Gesucht:

Die kürzeste Zeichenkette S , so daß jede Zeichenkette S_i eine Teilzeichenkette von S ist.

Beispiel:

$S_1 = abcxabcd$ $S_2 = cdxabcxabc$ $S_3 = bcxabcdfa$

$S_4 = abcdxab$

$S = abcdxabcxabcdfa$

Alle Paare Suffix-Präfix Problem

Annahme $|S_1| + \dots + |S_k| = m$

Naiver Ansatz:

- k^2 Paare von Zeichenketten
- Für jedes Paar (S_i, S_j) Lösung des Suffix-Präfix Problems in Zeit $O(|S_i| + |S_j|)$

Zeit $O(km)$

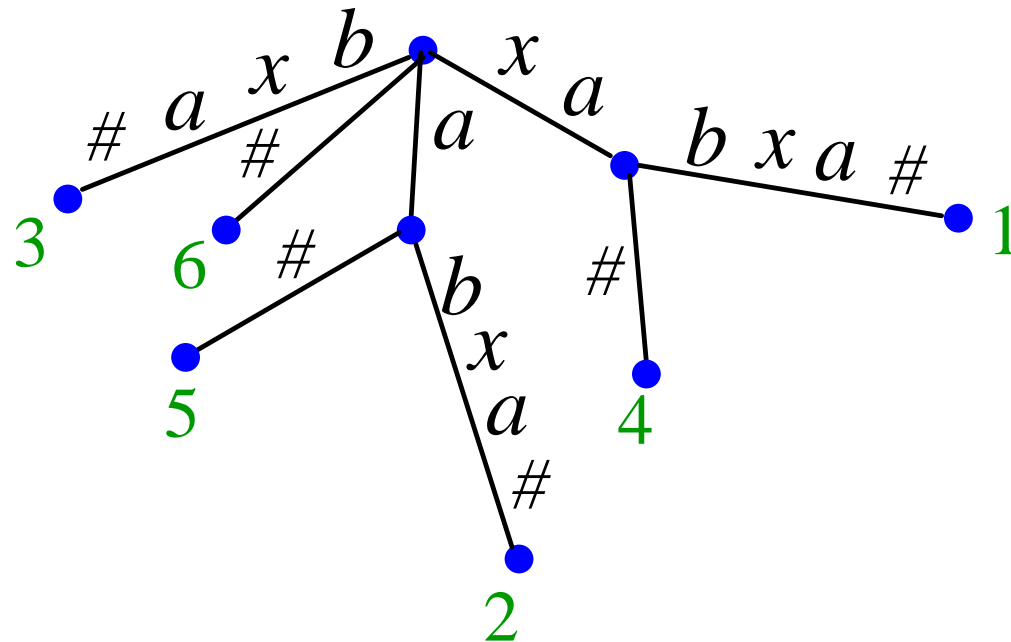
Neuer Ansatz: **Zeit** $O(k^2 + m)$

Definition

Eine Kante eines Suffixbaums heißt eine **Terminale Kante**, wenn sie nur dem einem Endesymbol (#) für eine Zeichenkette beschriftet ist.

Terminale Kante **Blatt**

Beispiel $T = xabxa\#$

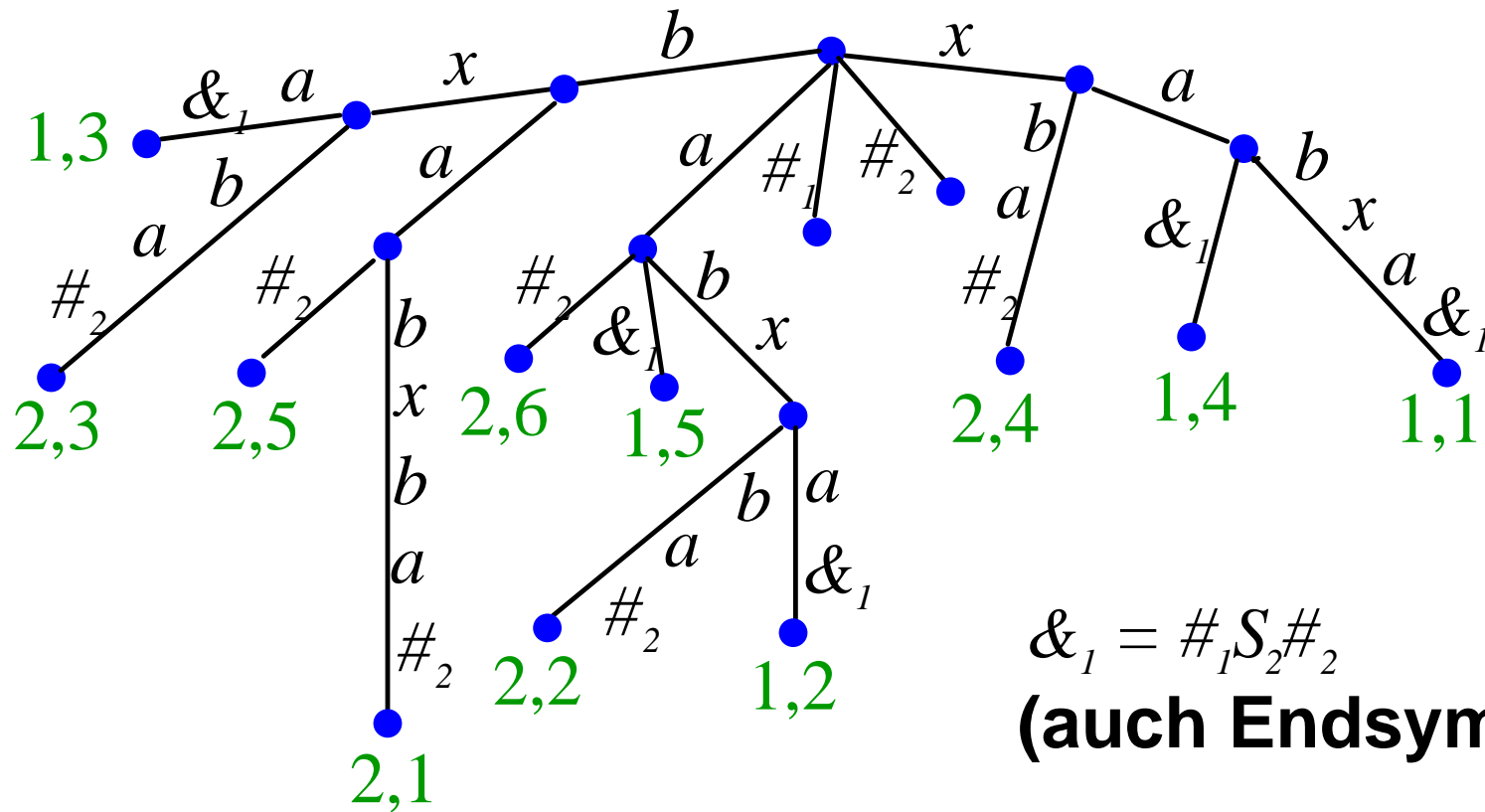


Neues Verfahren:

- generalisierter Suffixbaum $\mathbb{T}(S)$ für die Menge S der Zeichenketten
- für jeden inneren Knoten v :
 - **Liste** $L(v)$: enthält Index i , falls von v eine terminale Kante ausgeht, die in einem Suffix von S_i endet.

Beispiel

$$S_1 = xabxa \quad S_2 = babxba \quad S_1\#_1S_2\#_2 = xabxa\#_1babxba\#_2$$



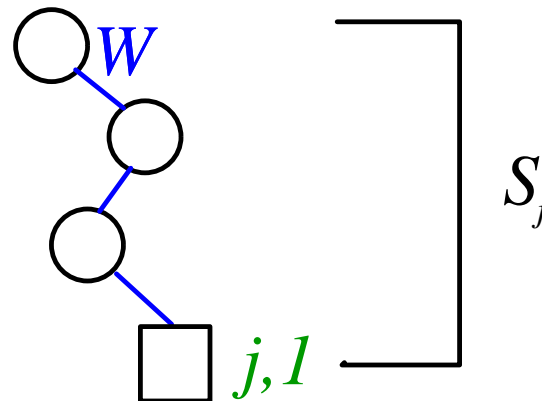
Neues Verfahren:

- generalisierter Suffixbaum $\mathbb{T}(S)$ für die Menge S der Zeichenketten
- für jeden inneren Knoten v :
 - **Liste** $L(v)$: enthält Index i , falls von v eine terminale Kante ausgeht, die in einem Suffix von S_i endet.

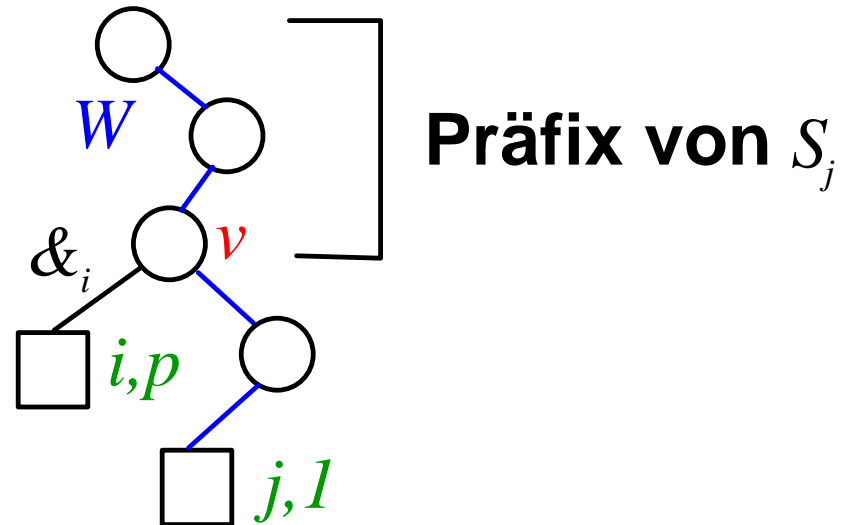
$$\sum_{v \in T(S)} |L(v)| =$$

Neues Verfahren:

- generalisierter Suffixbaum $\mathbb{T}(S)$ für die Menge S der Zeichenketten
- für jeden inneren Knoten v :
 - **Liste $L(v)$** : enthält Index i , falls von v eine terminale Kante ausgeht, die in einem Suffix von S_i endet.
- betrachte Zeichenkette S_j und den mit S_j beschrifteten Weg W von der Wurzel zum Blatt von S_j .



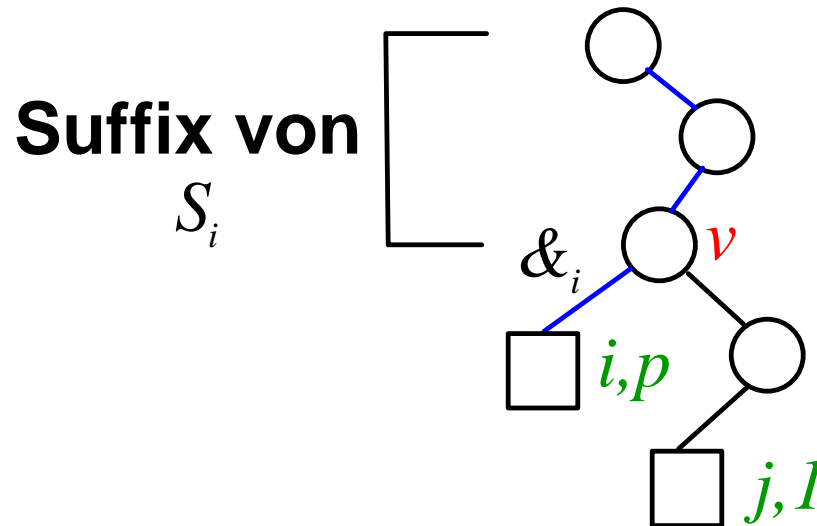
- betrachte einen Knoten v auf W .



Beobachtung:

Falls i in $L(v)$ ist, dann ist die Pfad-Beschriftung von v sowohl ein Präfix von S_j als auch ein Suffix von S_i .

- betrachte einen Knoten v auf W .



Beobachtung:

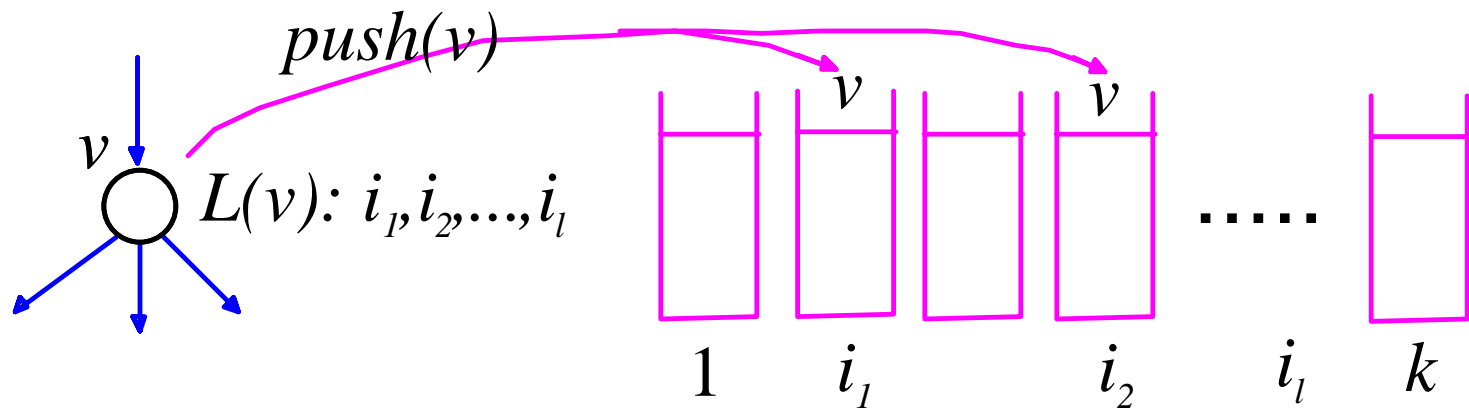
Falls i in $L(v)$ ist, dann ist die Pfad-Beschriftung von v sowohl ein Präfix von S_j als auch ein Suffix von S_i .

tiefster Knoten v auf W , so daß i in $L(v)$ ist, bestimmt die längste Übereinstimmung eines Präfixes von S_j mit einem Suffix von S_i

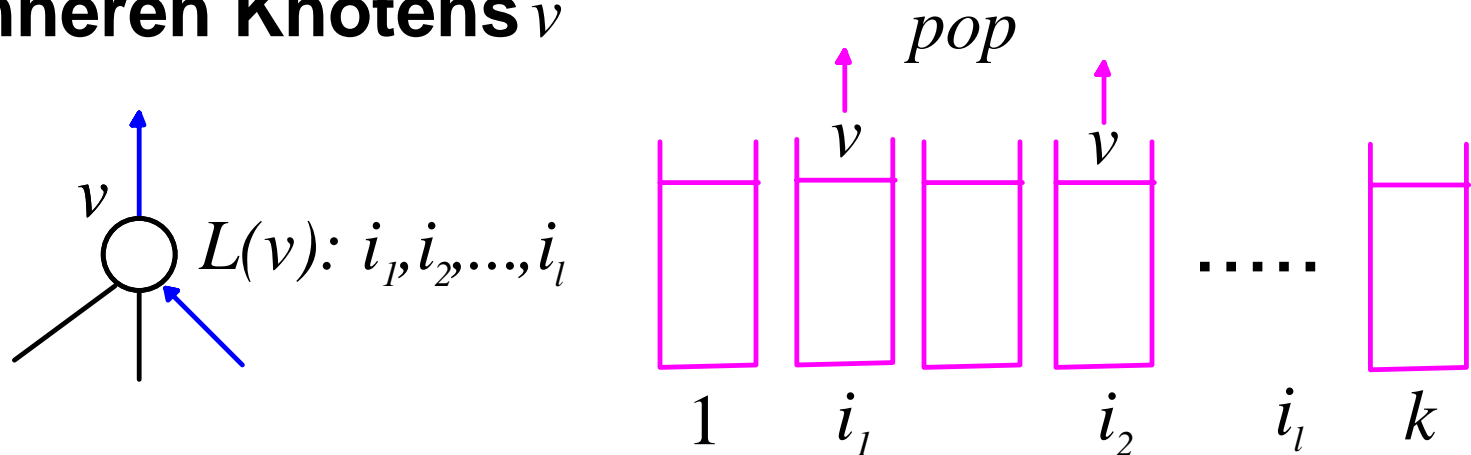
Bestimmung der tiefsten Knoten

Tiefensuche:

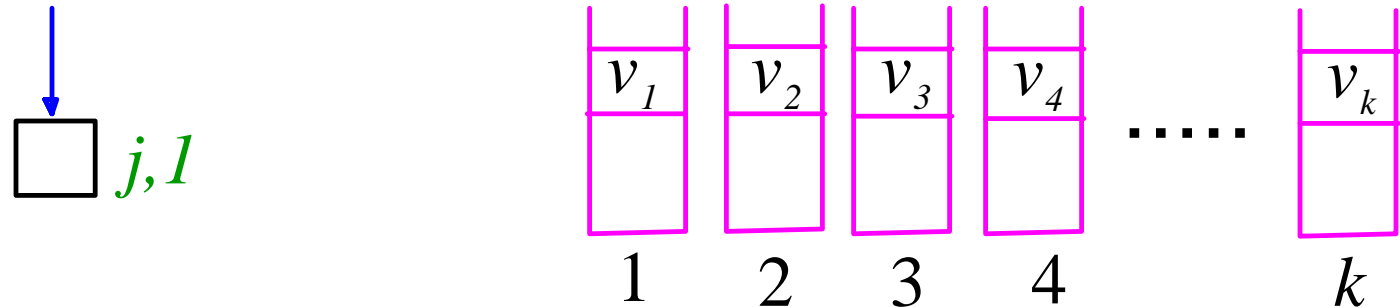
- k Stapel, für jede Zeichenkette einen
- Besuch eines neuen inneren Knotens v



- Letzter Besuch eines bereits besuchten inneren Knotens v



- **Besuch eines Blattes für Zeichenkette j**



Ausgabe (der Tiefen) der obersten Knoten der k Stapel

Invariante: Besuch von v

- **Es befinden sich nur Vorfahren von v auf den Stapeln**
- **Auf dem Stapel i befindet sich der tiefste Vorfahre u von v ganz oben, für den i in $L(u)$ ist.**

Bei Blatt j, l befinden sich für jedes i der tiefste Knoten auf dem Weg W_j oben auf Stapel i .

Analyse

Satz

Alle k^2 längsten Suffix-Präfix Übereinstimmungen werden in Zeit $O(m + k^2)$ berechnet. Da m die Größe der Eingabe ist und k^2 die Größe der Ausgabe ist, ist dies **optimal**.

Beweis

- Die Gesamtanzahl aller Indizes in den Listen ist $O(m)$
- Die Anzahl der Kanten in $\mathbb{T}(s)$ ist auch $O(m)$.
- Die Anzahl der *push* und *pop* Operationen ist proportional zu der Anzahl der Indizes in den Listen.
- Bei dem Besuch eines Blattes wird die Ausgabe in Zeit $O(k)$ erzeugt.