Entscheidbarkeit des Wortproblems für Typ 1-Sprachen

Prof. Dr. David Sabel

LFE Theoretische Informatik

Letzte Änderung der Folien: 3. Mai 2022

Vorbemerkungen

- Als Einschub wechseln wir nochmal zu Typ-1-Sprachen
  und holen einen Beweis nach
- Nächstes Mal geht es wieder mit den regulären Sprachen weiter

Das Wortproblem

Wiederholung:

Entscheidbarkeit

Eine Sprache heißt **entscheidbar**, wenn es einen Algorithmus gibt, der bei Eingabe der Grammatik G und einem Wort w in endlicher Zeit feststellt, ob $w \in L(G)$ gilt oder nicht.

Eng verwandt dazu ist:

Definition (Wortproblem für Typ i-Sprachen)

Das **Wortproblem** für Typ i-Sprachen ist die Frage, ob für eine gegebene Typ i-Grammatik $G = (V, \Sigma, P, S)$ und ein Wort $w \in \Sigma^*$ gilt: $w \in L(G)$ oder $w \notin L(G)$.

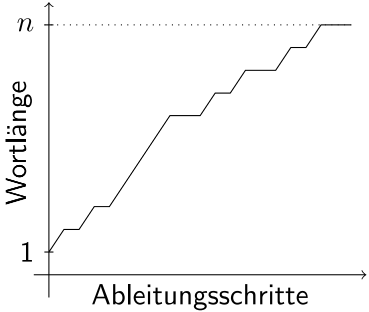
Wortproblem für Typ 1-Sprachen

Satz

Das Wortproblem für Typ 1-Sprachen ist entscheidbar: Es gibt einen Algorithmus, der bei Eingabe von Typ 1-Grammatik G und Wort w nach endlicher Zeit entscheidet, ob $w \in L(G)$ oder $w \notin L(G)$ gilt.

Idee zum Beweis des Satzes

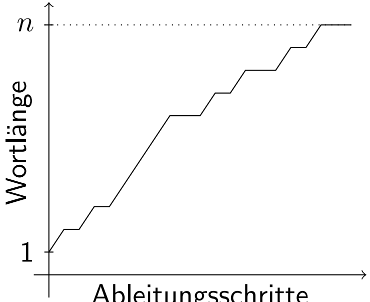
Ableitung eines Wortes der Länge n



mit Typ 1-Grammatik

Idee zum Beweis des Satzes

Ableitung eines Wortes der Länge n



mit Typ 1-Grammatik

Idee zum Entscheiden des Wortproblems:

- Betrachte $S \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_m$ mit $|w_m| = n$
- Da Typ 1-Produktionen nicht verkürzend sind: $\forall 1 \leq i \leq m : |w_i| \leq n$
- Probiere systematisch für alle Satzformen der Länge $\leq n$ durch, ob sie vom Startsymbol aus ableitbar sind
- Es gibt nur endlich viele Satzformen der Länge $\leq n$ und jede Typ 1-Grammatik leitet nur endlich viele Satzformen der Länge $\leq n$ her, **ohne dabei längere Satzformen zwischendrin herzuleiten**
- Herleitbare Satzformen der Länge n können **rekursiv** aus den Satzformen der Länge $< n$ berechnet werden

Beweis: Entscheidbarkeit des Wortproblems für Typ 1-Sprachen

Sei $G = (V, \Sigma, P, S)$ eine Typ 1-Grammatik und $w \in \Sigma^*$.
Für $m, n \in \mathbb{N}$ sei

$$\begin{aligned} L_m^n \quad &= \text{Menge aller Satzformen der Länge höchstens } n, \\ &\quad \text{die in höchstens } m \text{ Schritten von } S \text{ aus ableitbar sind} \\ L_m^n \quad &:= \{w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ und } S \Rightarrow_G^k w \text{ mit } k \leq m\} \end{aligned}$$

Rekursive Berechnung der Mengen (für $n > 0$):

$$\begin{aligned} L_0^n \quad &:= \quad \{S\} \\ L_m^n \quad &:= \quad \mathit{next}(L_{m-1}^n, n) \text{ für } m > 0 \\ &\quad\quad \text{wobei } \mathit{next}(L, n) := L \cup \{w' \mid w \in L, w \Rightarrow_G w', |w'| \leq n\} \end{aligned}$$

Die Berechnung terminiert, da die Mengen endlich sind: $|L_m^n| \leq |\Sigma \cup V|^{n+1}$
Wir können auch iterativ aus L_{i-1}^n die nachfolgende Menge L_i^n berechnen

Beweis: Entscheidbarkeit des Wortproblems für Typ 1-Sprachen (2)

Iterative Berechnung:

- Starte mit $L_0^n = \{S\}$
- Für $i = 1, 2, 3, \ldots$ berechne $L_i^n = next(L_{i-1}^n)$
- Stoppe dabei, wenn $L_i^n = L_{i-1}^n$
- Prüfe, ob $w \in L_i^n$ gilt.

Korrektheit: Wenn $L_i^n = L_{i-1}^n$, dann gilt $L_{i+k}^n = L_{i-1}^n$ für alle $k \in \mathbb{N}$ und daher enthält L_{i-1}^n genau alle aus S ableitbaren Wörter der Länge n.

Terminierung: Beweis durch Widerspruch.

- Nehme an, die Berechnung stoppt nicht.
- Da $L_{i-1}^n \subseteq L_i^n$, muss für alle $i \in \mathbb{N}$ gelten: $L_i^n \supset L_{i-1}^n$
- Daher gilt für alle $i \in \mathbb{N}$: $|L_i^n| > |L_{i-1}^n|$.
- Widerspruch zu $|L_i^n| \leq |V \cup \Sigma|^{n+1}$ für alle $i \in \mathbb{N}$.

Algorithmus 2: Entscheiden des Wortproblems für Typ 1-Grammatiken

Eingabe: Typ 1-Grammatik $G = (V, \Sigma, P, S)$ und ein Wort $w \in \Sigma^*$
Ausgabe: `Ja`, wenn $w \in L(G)$ und `Nein`, wenn $w \notin L(G)$
Beginn

  $n := |w|$;
  $L := \{S\}$;
  wiederhole
    $L_{\text{old}} := L$;
    $L := next(L_{\text{old}}, n)$;
  bis $(w \in L)$ *oder* $(L_{old} = L)$;
  wenn $w \in L$ **dann**
    return $\mathtt{Ja}$;
  sonst
    return $\mathtt{Nein}$;
  Ende

Ende

Beispiel

$G = (\{S, B\}, \{a, b, c\}, P, S)$ mit
$P = \{S \to aSBc, S \to abc, cB \to Bc, bB \to bb\}$

Wir berechnen L_i^6 für alle i:

Beispiel

$G = (\{S, B\}, \{a, b, c\}, P, S)$ mit
$P = \{S \to aSBc, S \to abc, cB \to Bc, bB \to bb\}$

Wir berechnen L_i^6 für alle i:

$L_0^6 = \{S\}$

Beispiel

$G = (\{S, B\}, \{a, b, c\}, P, S)$ mit
$P = \{S \to aSBc, S \to abc, cB \to Bc, bB \to bb\}$

Wir berechnen L_i^6 für alle i:

$L_0^6 = \{S\}$

$L_1^6 = next(L_0^6) = L_0^6 \cup \{w' \mid w \in L_0^6, w \Rightarrow w', |w'| \leq 6\} = \{S, aSBc, abc\}$

da $S \Rightarrow aSBc$ und $S \Rightarrow abc$

Beispiel

$G = (\{S, B\}, \{a, b, c\}, P, S)$ mit
$P = \{S \to aSBc, S \to abc, cB \to Bc, bB \to bb\}$

Wir berechnen L_i^6 für alle i:

$L_0^6 = \{S\}$

$L_1^6 = next(L_0^6) = L_0^6 \cup \{w' \mid w \in L_0^6, w \Rightarrow w', |w'| \leq 6\} = \{S, aSBc, abc\}$

da $S \Rightarrow aSBc$ und $S \Rightarrow abc$

$$L_2^6 = next(L_1^6) = L_1^6 \cup \{w' \mid w \in L_1^6, w \Rightarrow w', |w'| \leq 6\}$$
$$= \{S, aSBc, abc, aabcBc\}$$

da $aSBc \Rightarrow aaSBcBc$ (zu lang) und $aSBc \Rightarrow aabcBc$

Beispiel

$G = (\{S, B\}, \{a, b, c\}, P, S)$ mit
$P = \{S \to aSBc, S \to abc, cB \to Bc, bB \to bb\}$

Wir berechnen L_i^6 für alle i:

$L_0^6 = \{S\}$

$L_1^6 = next(L_0^6) = L_0^6 \cup \{w' \mid w \in L_0^6, w \Rightarrow w', |w'| \leq 6\} = \{S, aSBc, abc\}$

da $S \Rightarrow aSBc$ und $S \Rightarrow abc$

$$L_2^6 = next(L_1^6) = L_1^6 \cup \{w' \mid w \in L_1^6, w \Rightarrow w', |w'| \leq 6\}$$
$$= \{S, aSBc, abc, aabcBc\}$$

da $aSBc \Rightarrow aaSBcBc$ (zu lang) und $aSBc \Rightarrow aabcBc$

$$L_3^6 = next(L_2^6) = L_2^6 \cup \{w' \mid w \in L_2^6, w \Rightarrow w', |w'| \leq 6\}$$
$$= \{S, aSBc, abc, aabcBc, aabBcc\}$$

da $aabcBc \Rightarrow aabBcc$

Beispiel (2)

$$G = (\{S, B\}, \{a, b, c\}, P, S)$$
$$P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}.$$

...

$L_3^6 = \{S, aSBc, abc, aabcBc, aabBcc\}$

Beispiel (2)

$$G = (\{S, B\}, \{a, b, c\}, P, S)$$
$$P = \{S \to aSBc, S \to abc, cB \to Bc, bB \to bb\}.$$

...

$$L_3^6 = \{S, aSBc, abc, aabcBc, aabBcc\}$$

$$L_4^6 = next(L_3^6) = L_3^6 \cup \{w' \mid w \in L_3^6, w \Rightarrow w', |w'| \leq 6\}$$
$$= \{S, aSBc, abc, aabcBc, aabBcc, aabbcc\}$$

da $aabBcc \Rightarrow aabbcc$

Beispiel (2)

$$G = (\{S, B\}, \{a, b, c\}, P, S)$$
$$P = \{S \to aSBc, S \to abc, cB \to Bc, bB \to bb\}.$$

...

$$L_3^6 = \{S, aSBc, abc, aabcBc, aabBcc\}$$

$$L_4^6 = next(L_3^6) = L_3^6 \cup \{w' \mid w \in L_3^6, w \Rightarrow w', |w'| \leq 6\}$$
$$= \{S, aSBc, abc, aabcBc, aabBcc, aabbcc\}$$

da $aabBcc \Rightarrow aabbcc$

$$L_5^6 = next(L_4^6) = L_4^6 \cup \{w' \mid w \in L_4^6, w \Rightarrow w', |w'| \leq 6\}$$
$$= \{S, aSBc, abc, aabcBc, aabBcc, aabbcc\} = L_4^6$$

Beispiel (2)

$$G = (\{S, B\}, \{a, b, c\}, P, S)$$
$$P = \{S \to aSBc, S \to abc, cB \to Bc, bB \to bb\}.$$

...

$$L_3^6 = \{S, aSBc, abc, aabcBc, aabBcc\}$$

$$\begin{aligned} L_4^6 &= next(L_3^6) = L_3^6 \cup \{w' \mid w \in L_3^6, w \Rightarrow w', |w'| \leq 6\} \\ &= \{S, aSBc, abc, aabcBc, aabBcc, aabbcc\} \end{aligned}$$

da $aabBcc \Rightarrow aabbcc$

$$\begin{aligned} L_5^6 &= next(L_4^6) = L_4^6 \cup \{w' \mid w \in L_4^6, w \Rightarrow w', |w'| \leq 6\} \\ &= \{S, aSBc, abc, aabcBc, aabBcc, aabbcc\} = L_4^6 \end{aligned}$$

d.h. $L_i^6 = \{S, aSBc, abc, aabcBc, aabBcc, aabbcc\}$ für alle $i \geq 4$

Wortproblem für Typ 2- und Typ 3-Sprachen

Korollar

Das Wortproblem für Typ 2- und Typ 3-Sprachen ist entscheidbar.

Bemerkungen:

- Der Algorithmus für Typ 1-Sprachen hat exponentielle Laufzeitkomplexität
- Das Wortproblem für Typ 2- und das Wortproblem für Typ 3-Sprachen sind in polynomieller Zeit lösbar.
- Das Wortproblem für Typ 0-Sprachen ist unentscheidbar (aber rekursiv aufzählbar)