(C) Pumping-Lemma

Satz 1.5.3 (Pumping-Lemma für kontextfreie Sprachen) Sei L eine kontextfreie Sprache über Σ. Dann gibt es ein $n \in \mathbb{N}_0$, so dass gilt: Zu jedem $z \in L$ mit $|z| \geq n$ gibt es $u, v, w, x, y \in \Sigma^*$ mit $z = uvwxy$ und

1. $|vx| \geq 1$,
2. $|vwx| \leq n$,
3. Für jedes $k \geq 0$ ist $uv^k wx^k y \in L$.

(D) Abschlusseigenschaften

Satz 1.5.4 Die kontextfreien Sprachen sind abgeschlossen unter den Operationen Vereinigung, Produkt und Kleene-Stern.

Bemerkungen

1. Die kontextfreien Sprachen sind auch abgeschlossen unter Spiegelung.
2. Die kontextfreien Sprachen sind nicht abgeschlossen unter den Operationen Durchschnitt und Komplementbildung.
3. Es gilt aber noch: L_1 kontextfrei, L_2 regulär $\implies L_1 \cap L_2$ kontextfrei.

(E) Der Cocke-Younger-Kasami- (CYK-) Algorithmus

CYK-Algorithmus: zur effizienten Lösung des Wortproblems für kontextfreie Grammatiken (in Chomsky-Normalform).

Algorithmus-Idee

Sei $w = a_1 \ldots a_n$ mit $a_i \in \Sigma$. Für $1 \leq i, l \leq n$ sei $w_{il} = a_i a_{i+1} \ldots a_{i+l-1}$ ($|w_{il}| = l$, $w = w_{1n}$). Für alle i, l wird (iterativ nach l) die Menge V_{il} aller Variablen A bestimmt, für die $A \Rightarrow^* w_{il}$ gilt. (Dann gilt: $w \in \mathcal{L}(G) \iff S \in V_{1n}$.)

- $l = 1$ (d.h. $w_{il} = a_i$): $A \Rightarrow^* a_i \iff A \rightarrow a_i \in P$.
- $l > 1$: $A \Rightarrow^* w_{il} \iff A \Rightarrow BC, B \Rightarrow^* a_i \ldots a_{i+k-1}, C \Rightarrow^* a_{i+k} \ldots a_{i+l-1}$
  $\iff A \rightarrow BC \in P, B \in V_{ik}, C \in V_{i+k,l-k}$

Algorithmus

Eingabe: Grammatik $G = (V, \Sigma, P, S)$, $w = a_1 \ldots a_n \in \Sigma^*$;

1. FOR $i = 1, \ldots, n$ DO $V_{i1} := \{A \mid A \rightarrow a_i \in P\}$ ENDDO $(* \; l = 1 \; *)$
2. FOR $l = 2, \ldots, n$ DO $(*$ Iteration über $l \; *)$
    FOR $i = 1, \ldots, n - l + 1$ DO $(* \; i > n - l + 1 \Longrightarrow i + l - 1 > n \; *)$
        $V_{il} := \emptyset$;
        FOR $k = 1, \ldots, l - 1$ DO
            $V_{il} := V_{il} \cup \{A \mid A \rightarrow BC \in P, B \in V_{ik}, C \in V_{i+k,l-k}\}$
        ENDDO
    ENDDO
ENDDO

Ausgabe: “Ja” (d.h.: $w \in \mathcal{L}(G)$), falls $S \in V_{1n}$, sonst “Nein”.

Bemerkung

Komplexität des CYK-Algorithmus: $O(|P|n^3)$ (für feste Grammatik: $O(n^3)$).

1.6 Kellerautomaten

Definition. Ein (***nichtdeterministischer***) ***Kellerautomat*** (***pushdown automaton***, ***PDA***) $M = (Z, \Sigma, \Gamma, \delta, z_0, \#, E)$ ist gegeben durch:

- eine endliche Menge Z von ***Zuständen***,
- ein ***Eingabealphabet*** Σ,
- ein Alphabet Γ (***Kelleralphabet***),
- eine totale ***Überführungsfunktion*** $\delta : Z \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow \mathfrak{P}_e(Z \times \Gamma^*)$,
- $z_0 \in Z$ (***Startzustand***),
- $\# \in \Gamma$ (***Startsymbol, unterstes Kellerzeichen***),
- $E \subseteq Z$ (Menge der ***Endzustände***).

Eine ***Konfiguration*** von M ist ein Tripel $(z, w, \alpha) \in Z \times \Sigma^* \times \Gamma^*$.

Informelle Bedeutung einer Konfiguration (z, w, α):
z = erreichter Zustand, w = noch zu verarbeitendes Restwort, α = derzeitiger Kellerinhalt.