Jun.-Prof. Dr. Bernhard Beckert
Ulrich Koch

UNIVERSITÄT KOBLENZ · LANDAU

Einführung in die Theoretische Informatik I/ Grundlagen der Theoretischen Informatik
Sommersemester 2007
12. Aufgabenblatt

Ausgabe: 10. 07. 2007 Besprechung: 17./18. 07. 2007

---

1 Pumping-Lemma für kontextfreie Sprachen

Lösung:

Annahme: L sei doch kontextfrei. Sei n die Zahl aus dem Pumping-Lemma. Wir untersuchen $z = a^{2^n}$. Sei $uvwxy = z$ eine Zerlegung. Dann gibt es $i, j, k \in \mathbb{N}_0$ mit $uy = a^i, vx = a^j$ und $w = a^k$, und es gilt $1 \leq j \wedge j + k < n$. Wegen $n < 2^n = i + j + k$ gilt $i \neq 0$. Damit hat uv^2wx^2y die Länge $i + 2j + k$, und es gilt:

$$2^n = i + j + k < i + 2j + k < 2i + 2j + 2k = 2(i + j + k) = 2 \cdot 2^n = 2^{n+1}$$

Zusammengefasst: $2^n < |uv^2wx^2y| < 2^{n+1}$, folglich $uv^2wx^2y \notin L$. Damit ist L nicht kontextfrei. □

2 Pushdown-Automaten

Lösung:

$L_l(A) = L'$ gilt für $A = (\{s_0, s_1, s_2, s_3\}, \{a, b, c\}, \{X, Z_0\}, \Delta, s_0, Z_0, \emptyset)$ mit:

$$
\begin{array}{rcll}
\Delta & = & \{((s_0, a, Z_0), & (s_0, XZ_0)), \quad a\text{s zählen (Keller aufbauen)} \\
 & & ((s_0, a, X), & (s_0, XX)), \\
 & & & \\
 & & ((s_0, b, Z_0), & (s_1, Z_0)), \quad b\text{s überlesen} \\
 & & ((s_0, b, X), & (s_1, X)), \\
 & & & \\
 & & ((s_0, c, Z_0), & (s_3, Z_0)), \quad c\text{s zählen (Keller leeren)} \\
 & & ((s_0, c, X), & (s_2, \varepsilon)), \\
 & & & \\
 & & ((s_0, \varepsilon, Z_0), & (s_0, \varepsilon)), \quad \varepsilon \text{ erkennen} \\
 & & & \\
 & & ((s_1, b, Z_0), & (s_1, Z_0)), \quad b\text{s überlesen} \\
 & & ((s_1, b, X), & (s_1, X)), \\
 & & & \\
 & & ((s_1, c, Z_0), & (s_3, Z_0)), \quad c\text{s zählen (Keller leeren)} \\
 & & ((s_1, c, X), & (s_2, \varepsilon)), \\
 & & & \\
 & & ((s_1, \varepsilon, Z_0), & (s_1, \varepsilon)), \quad b^m \text{ erkennen} \\
 & & & \\
 & & ((s_2, c, Z_0), & (s_3, Z_0)), \quad c\text{s zählen (Keller leeren)} \\
 & & ((s_2, c, X), & (s_2, \varepsilon)), \\
 & & ((s_2, \varepsilon, Z_0), & (s_2, \varepsilon)), \\
 & & & \\
 & & ((s_3, c, Z_0), & (s_3, Z_0)), \quad \text{„überschüssige“ } c\text{s überlesen} \\
 & & ((s_3, \varepsilon, Z_0), & (s_3, \varepsilon))\}
\end{array}
$$

3 CYK-Algorithmus

Lösung:

1. Da G weder nutzlose Symbole noch ε-Regeln noch Kettenregeln enthält, kann sie direkt umgeformt werden in $G' = (\{E, B, C, A, M\}, \{+, *, v\}, R', E)$ mit folgenden Regeln in R':

$$
\begin{array}{rcl}
E & \rightarrow & EB \mid EC \mid v \\
B & \rightarrow & AE \\
C & \rightarrow & ME \\
A & \rightarrow & + \\
M & \rightarrow & *
\end{array}
$$

2.

	v	$+$	v	$*$	v
v	$\{E\}$				
$+$	$\emptyset$	$\{A\}$			
v	$\{E\}$	$\{B\}$	$\{E\}$		
$*$	$\emptyset$	$\emptyset$	$\emptyset$	$\{M\}$	
v	$\{E\}$	$\{B\}$	$\{E\}$	$\{C\}$	$\{E\}$

4 Entscheidbarkeit

Lösung:

1. M zählt die Sprache $L'' = \Im((ab)^*)$ auf.
2. $M' = (\{q_0, q_1, q_2, q_3\}, \{a, b, \#\}, \delta', q_0)$ mit:

δ'	a	b	$\#$	
q_0			(q_1, L)	zum letzten Zeichen gehen
q_1		(q_2, L)	(q_3, R)	zum a gehen/rechtes Ende suchen
q_2	(q_1, L)			zum b gehen
q_3	(q_3, R)	(q_3, R)	$(h, \#)$	rechtes Ende suchen/halten

Alle nicht aufgeführten Übergänge sind undefiniert.

Um das Hängen zu vermeiden, kann δ' total gemacht werden: Wir fügen den Zustand q_F hinzu und setzen alle bisher nicht definierten Übergänge für einen Zustand q und ein Eingabezeichen x auf $\delta'(q, x) = (q_F, x)$. Das gilt auch für $q = q_F$.