Technische Universität München
Fakultät für Informatik
Lehrstuhl Informatik XIV
Prof. Dr. Ernst W. Mayr
Dr. Werner Meixner

Sommersemester 2015
Lösungsblatt 8
22. Juni 2015

Theoretische Informatik

Hausaufgabe 1 (5 Punkte)

Die Produktionen einer Grammatik $G = (V, \{a, b\}, P, S)$ mit $V = \{S, X, A, B\}$ seien wie folgt gegeben.

$$\begin{aligned} S &\rightarrow AXSB \mid AXB\,, \\ X &\rightarrow a \mid SA \mid AX\,. \\ A &\rightarrow a\,,\ B \rightarrow b\,. \end{aligned}$$

1. Alle Nichtterminalen aus V sind nützlich in der Grammatik G. Begründen Sie diesen Sachverhalt.

2. Für $w \in \{a, b\}^*$ bezeichne $|w|_a$ bzw. $|w|_b$ die Anzahl von Vorkommen von a bzw. b in w.

   Zeigen Sie für alle $w \in L(G)$, dass gilt $|w|_a \geq 2 \cdot |w|_b$.

3. Geben Sie eine Linksableitung von $w = aaaabb$ aus S an?

Lösung

1. $S \rightarrow AXB \rightarrow^* aab \in \Sigma^*$. (1P)

2. Beweis durch Induktion über die Erzeugung. A und B durch a bzw. b ersetzen. Eigenschaft $P_S(w)$: $\#_a(w) \geq 2 \cdot \#_b(w)$. Eigenschaft $P_X(w)$: $\#_a(w) \geq 1 + 2 \cdot \#_b(w)$.

   Induktionsanfang: $w = a \in L(X)$. Es gilt $P_X(w)$.

   Induktive Erzeugung:
   $X \rightarrow Sa$: Falls $P_S(w)$ für ein $w \in L(S)$ gilt, dann gilt $P_X(wa)$ für $wa \in L(X)$.
   $X \rightarrow aX$: Falls $P_X(w)$ für $w \in L(X)$ gilt, dann gilt $P_X(aw)$ für $aw \in L(X)$.
   $S \rightarrow aXSb$: Falls $P_X(u)$ bzw. $P_S(v)$ für $u \in L(X)$ bzw. $v \in L(S)$ gilt, dann folgt $P_S(auvb)$ für $auvb \in L(S)$.
   $S \rightarrow aXb$: Falls $P_X(w)$ für $w \in L(X)$ gilt, dann folgt $P_S(awb)$ für $awb \in L(S)$.
   (3P)

3. $S \rightarrow AXSB \rightarrow aXSB \rightarrow aaSB \rightarrow aaAXBB \rightarrow aaaXBB \rightarrow aaaaBB \rightarrow aaaabB \rightarrow aaaabb$. (1P)

Hausaufgabe 2 (5 Punkte)

Sei $K = (\{p, q\}, \{0, 1\}, \{Z_0, X\}, \delta, q, Z_0, \{p\})$ ein Kellerautomat mit folgender Übergangsfunktion

$$\begin{aligned}
\delta(q, 0, Z_0) &= \{(q, XZ_0)\}, & \delta(q, 0, X) &= \{(q, XX)\}, \\
\delta(q, 1, X) &= \{(q, X)\}, & \delta(q, \epsilon, X) &= \{(p, \epsilon)\}, \\
\delta(p, \epsilon, X) &= \{(p, \epsilon)\}, & \delta(p, 1, X) &= \{(p, XX)\}, \\
\delta(p, 1, Z_0) &= \{(p, \epsilon)\}.
\end{aligned}$$

Berechnen Sie, ausgehend von der Anfangskonfiguration (q, w, Z_0), alle mit folgenden Eingaben erreichbaren Konfigurationen:

i) 01, ii) 0011.

Lösung

Wir stellen zunächst fest, dass bei K der Nichtdeterminismus darin besteht, dass mit X als oberstem Kellerzeichen entweder die deterministische Verarbeitung eines Eingabezeichens oder ein ϵ-Übergang zur Auswahl stehen. Damit ist der Konfigurationsbaum ein binärer Baum, den wir zeilen- und spaltenweise notieren können. Die ϵ-Übergänge werden wir in der Zeile nach rechts notieren, wohingegen die Verarbeitung von Eingabezeichen in Spalten nach unten aufgeschrieben wird. Das Zeichen # wird zur Trennung der Konfigurationsübergänge in den Spalten dienen. In den Zeilen verwenden wir Pfeile $\to$ für die einzelnen Übergänge.

i) $(q, 01, Z_0)$
$(q, 1, XZ_0) \to (p, 1, Z_0) \to (p, \epsilon, \epsilon)$
#
$(q, \epsilon, XZ_0) \to (p, \epsilon, Z_0)$

(1P)

ii) $(q, 0011, Z_0)$
$(q, 011, XZ_0) \to (p, 011, Z_0)$
$(q, 11, XXZ_0) \to (p, 11, XZ_0) \to (p, 11, Z_0)$
$(p, 1, \epsilon)$
#
$(p, 1, XXZ_0) \to (p, 1, XZ_0) \to (p, 1, Z_0)$
(p, ϵ, ϵ)
#
$(p, \epsilon, XXZ_0) \to (p, \epsilon, XZ_0) \to (p, \epsilon, Z_0)$
#
(p, ϵ, XZ_0) wie gehabt!
#
$(q, 1, XXZ_0) \to^* (p, \epsilon, XXXZ_0)$
(q, ϵ, XXZ_0) wie gehabt!

(4P)

Hausaufgabe 3 (5 Punkte)

Sei $\Sigma = \{a, b\}$ und

$$L = \{w \in \Sigma^* \,;\, w = uav, u, v \in \Sigma^*, |u|_a = |v|_b\}\,.$$

1. Geben Sie eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ an mit $L_G = L$.
2. Leiten Sie nach Satz der Vorlesung systematisch einen zu G äquivalenten Kellerautomaten K her.

Lösung

1. Seien $V = \{S, X, Y, A, A_0, B, B_0\}$ und P gegeben durch

$$\begin{array}{lcl} S & \to & XSY \mid B_0AA_0\,, \\ X & \to & B_0A\,, \\ Y & \to & BA_0\,, \\ A_0 & \to & aA_0 \mid \epsilon\,, \\ B_0 & \to & bB_0 \mid \epsilon\,, \\ A & \to & a\,, \\ B & \to & b\,. \end{array}$$

   Dann gilt $L_G = L$. Es gibt aber wesentlich kürzere Lösungen. (2P)

2. Wir konstruieren $K = (Q, \Sigma, \Gamma, q_0, S, \delta, \emptyset)$ mit $L_\epsilon(K) = L$ nach Satz der Vorlesung. K wird als nichtdeterministischer Kellerautomat mit nur einem Zustand q_0 konstruiert.

   Seien $Q = \{q_0\}$ und $\Gamma = V$. Alle Produktionen von G haben bereits die in Satz 3.56 verlangte Form $C \to cD_1 \dots D_k$ mit $c \in \Sigma \cup \{\epsilon\}$ und $D_i \in V$, wobei $k = 0$ zugelassen ist und dann für terminale bzw. ϵ-Produktionen steht.

   Die Übergangsfunktion δ wird für alle $c \in \Sigma \cup \{\epsilon\}$ und $D \in V$ definiert durch

$$\delta(q_0, c, D) := \{(q_0, \gamma) \mid (D \to c\gamma) \in P\}\,.$$

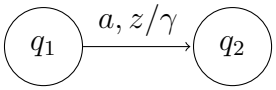
   Wir listen die einzelnen Übergänge wie folgt auf.

$$\begin{array}{llll} \delta(q_0, \epsilon, S) & \ni (q_0, XSY)\,, & \delta(q_0, \epsilon, S) & \ni (q_0, B_0AA_0)\,, \\ \delta(q_0, \epsilon, X) & \ni (q_0, B_0A)\,, & \delta(q_0, \epsilon, Y) & \ni (q_0, BA_0)\,, \\ \delta(q_0, a, A_0) & \ni (q_0, A_0)\,, & \delta(q_0, \epsilon, A_0) & \ni (q_0, \epsilon)\,, \\ \delta(q_0, b, B_0) & \ni (q_0, B_0)\,, & \delta(q_0, \epsilon, B_0) & \ni (q_0, \epsilon)\,, \\ \delta(q_0, a, A) & \ni (q_0, \epsilon)\,, & \delta(q_0, b, B) & \ni (q_0, \epsilon)\,. \end{array}$$

   (3P)

Hausaufgabe 4 (5 Punkte)

Ein Queue-Automat (kurz: QA) ist ein Tupel $A = (Q, \Sigma, \Gamma, q_0, z_0, \delta)$. Dabei sind Q (Zustandsmenge), Σ (Eingabealphabet) und Γ (Queuealphabet) endliche Mengen, $q_0 \in Q$ ist der Startzustand, und $z_0 \in \Gamma$ beschreibt die initiale Queue. Die Übergangsfunktion δ hat den Typ $Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \to \mathcal{P}(Q \times \Gamma^*)$. Graphisch kann eine Transition $(q_2, \gamma) \in \delta(q_1, a, z)$ eines Queue-Automaten wie folgt dargestellt werden:



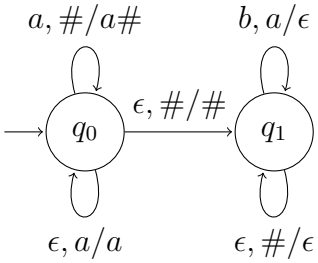
Eine Konfiguration eines Queue-Automaten ist ein Tripel $(q, w, \gamma) \in Q \times \Sigma^* \times \Gamma^*$. Dabei ist q der aktuelle Zustand, w das noch zu lesende Wort und γ der aktuelle Inhalt der Queue. Daraus ergibt sich die Übergangsrelation $\to_A$ zwischen Konfigurationen:

$$(q, bw, z\gamma) \to_A (q', w, \gamma\gamma') \,, \qquad \text{wenn} \qquad (q', \gamma') \in \delta(q, b, z) \,.$$

Die (mit leerer Queue) akzeptierte Sprache eines Queue-Automaten ist

$$L_\epsilon(A) = \{w \in \Sigma^* \,;\, \exists q' \in Q.\ (q_0, w, z_0) \to_A^* (q', \epsilon, \epsilon)\} \,.$$

1. Der Queue-Automat $A_1 = (\{q_0, q_1\}, \{a, b\}, \{a, b, \#\}, q_0, \#, \delta)$ sei wie folgt graphisch dargestellt:



   Geben Sie eine akzeptierende Konfigurationsfolge für das Eingabewort $aabb$ an.

2. Geben Sie einen Queue-Automaten A_2 an mit $L(A_2) = \{ww^R \,;\, w \in \{a, b\}^*\}$.

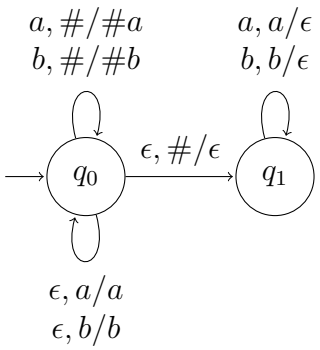
Lösung

1. Die einzig mögliche akzeptierende Folge ist:

$$\begin{aligned}&(q_0, aabb, \#) \to_{A_1} (q_0, abb, a\#) \to_{A_1} (q_0, abb, \#a) \to_{A_1} (q_0, bb, aa\#) \to_{A_1} \\ &(q_0, bb, a\#a) \to_{A_1} (q_0, bb, \#aa) \to_{A_1} (q_1, bb, aa\#) \to_{A_1} (q_1, b, a\#) \to_{A_1} \\ &(q_1, \epsilon, \#) \to_{A_1} (q_1, \epsilon, \epsilon)\end{aligned}$$

(2P)

2. Der Queue-Automat $A_2 = (\{q_0, q_1\}, \{a, b\}, \{a, b, \#\}, q_0, \#, \delta)$ kann wie folgt graphisch dargestellt werden:



(3P)

Zusatzaufgabe 5 (wird nicht korrigiert)

Gegeben sei der Kellerautomat $K = (\{q\}, \Sigma, \Gamma, q, Z_0, \delta)$ mit $\Sigma = \{a, b, \#\}$, $\Gamma = \{Z_0, X, Y, Z\}$ und der Übergangsfunktion

$$\begin{array}{lcllcl} \delta(q, \epsilon, Z_0) & = & \{(q, XZ)\}, & \delta(q, a, X) & = & \{(q, XY)\}, \\ \delta(q, \#, X) & = & \{(q, \epsilon)\}, & \delta(q, b, Y) & = & \{(q, \epsilon)\}, \\ \delta(q, a, Z) & = & \{(q, \epsilon)\}. & & & \end{array}$$

Leiten Sie eine Grammatik G her, die $L_\epsilon(K)$ erzeugt. Die Korrektheit von G muss durch systematische Anwendung einer geeigneten Methode sichergestellt werden.

Hinweis: $K = (\{q\}, \Sigma, \Gamma, q, Z_0, \delta)$ ist eine Kurzschreibweise für $K = (\{q\}, \Sigma, \Gamma, q, Z_0, \delta, \emptyset)$.

Lösung

Anwendung der Methode aus der Vorlesung, allerdings können alle Tripel $[p, A, q]$ vereinfacht werden zu A, weil es nur einen Zustand gibt.

$$\begin{array}{lcllcllcl} S & \to & Z_0, & & & & & & \\ Z_0 & \to & XZ, & X & \to & aXY, & & & \\ X & \to & \#, & Y & \to & b, & Z & \to & a. \end{array}$$

Hinweis: *Die Vorbereitungsaufgaben bereiten die Tutoraufgaben vor und werden in der Zentralübung unterstützt. Tutoraufgaben werden in den Übungsgruppen bearbeitet. Hausaufgaben sollen selbstständig bearbeitet und zur Korrektur und Bewertung abgegeben werden.*

Vorbereitung 1

Seien $L_1, L_2 \subseteq \Sigma^*$. Zeigen Sie:
Wenn L_1 kontextfrei ist und L_2 regulär, dann ist $L_1 \cap L_2$ kontextfrei.
Hinweis: Konstruieren sie aus einem PDA und einem DFA/NFA einen neuen PDA.

Lösung

Seien $M_1 = (Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$ ein PDA mit $L_F(M_1) = L_1$, und $M_2 = (Q', \Sigma, \delta', q_0', F')$ ein DFA mit $L(M_2) = L_2$. Wir konstruieren einen neuen PDA $M'' = (Q \times Q', \Sigma, \Gamma, (q_0, q_0'), Z_0, \delta'', F \times F')$, dessen Übergangsfunktion δ'' wie folgt definiert ist:

$$\delta''((q, q'), a, Z) = \{((p, \delta'(q', a)), \gamma) \mid (p, \gamma) \in \delta(q, a, Z)\} \qquad \forall a \in \Sigma$$
$$\delta''((q, q'), \epsilon, Z) = \{((p, q'), \gamma) \mid (p, \gamma) \in \delta(q, \epsilon, Z)\}$$

Man kann diesen Automaten als das Produkt aus einem DFA und einem Kellerautomaten ansehen. Der neue Automat akzeptiert ein Wort w genau dann, wenn beide Teilautomaten w akzeptieren.

Vorbereitung 2

Die Produktionen einer kontextfreien Grammatik $G = (V, \Sigma, P, S)$ seien gegeben durch

$$S \to A\,, \qquad A \to E \mid E + A \mid E - (A)\,,$$
$$E \to P \mid P \times E \mid E/P\,, \qquad P \to (A) \mid a\,.$$

Zeigen Sie durch Anwendung von Earley's Algorithmus, dass $a \times a - (a + a) \in L(G)$ gilt.

Lösung

Die syntaktischen Objekte $[iAj[\alpha_1 \cdots \alpha_k.\alpha_{k+1} \cdots \alpha_r$ im Earley-Algorithmus bedeuten Aussagen bezüglich einer Sequenz $x = x_1 \cdots x_n \in \Sigma^+$ wie folgt in Worten:
„Der Abschnitt der Länge $j - i$ des Wortes x von x_i ab bis vor x_j ist ableitbar, und zwar aus dem Abschnitt $\alpha_1 \cdots \alpha_k$ der rechten Seite der Produktion $A \to \alpha_1 \cdots \alpha_k \alpha_{k+1} \cdots \alpha_r$."
Man beachte:
Aus $[iAj[\alpha_1 \cdots \alpha_k.B\alpha_{k+2} \cdots \alpha_r$ und $[jBl[\beta_1 \cdots \beta_s.$ folgt $[iAl[\alpha_1 \cdots \alpha_k B.\alpha_{k+2} \cdots \alpha_r\,.$

Die Eingabesequenz ist $a \times a - (a + a) \in L(G)$.
Wir leiten nur die notwendigen Aussagen ab. Zur Orientierung benutzen wir eine Linksableitung von $a \times a - (a + a) \in L(G)$ wie folgt.

$$\begin{aligned} S &\to A \to E - (A) \to P \times E - (A) \to a \times E - (A) \to a \times P - (A) \to a \times a - (A) \\ &\to a \times a - (E + A) \to a \times a - (P + A) \to a \times a - (a + A) \\ &\to a \times a - (a + E) \to a \times a - (a + P) \to a \times a - (a + a) \end{aligned}$$

Initialisierung S_1 : $[1S1[.A;$

S_1 : $[1A1[.E-(A)$ ▮ $[1E1[.P\times E$ ▮ $[1P1[.a$ ▮ $\cdots$

S_2 : $[1P2[a.$ ▮ $[1E2[P.\times E$ ▮ $\cdots$

S_3 : $[1E3[P\times .E$ ▮ $[3E3[.P$ ▮ $[3P3[.a$ ▮ $\cdots$

S_4 : $[3P4[a.$ ▮ $[3E4[P.$ ▮ $[1E4[P\times E.$ ▮ $[1A4[E.-(A)$ ▮ $\cdots$

S_5 : $[1A5[E-.(A)$ ▮ $\cdots$

S_6 : $[1A6[E-(.A)$ ▮ $[6A6[.E+A$ ▮ $[6E6[.P$ ▮ $[6P6[.a$ ▮ $\cdots$

S_7 : $[6P7[a.$ ▮ $[6E7[P.$ ▮ $[6A7[E.+A$ ▮ $\cdots$

S_8 : $[6A8[E+.A$ ▮ $[8A8[.E$ ▮ $[8E8[.P$ ▮ $[8P8[.a$ ▮ $\cdots$

S_9 : $[8P9[a.$ ▮ $[8E9[P.$ ▮ $[8A9[E.$ ▮
$[6A9[E+A.$ ▮ $[1A9[E-(A.)$ ▮ $\cdots$

S_{10} : $[1A10[E-(A).$ ▮ $[1S10[A.$ ▮ $\cdots$ (Rückgabe $x \in L$).

Zweite Version ohne Farbmarkierungen:

Initialisierung S_1 : $[1S1[.A;$

S_1 : $[1A1[.E-(A)$ ▮ $[1E1[.P\times E$ ▮ $[1P1[.a$ ▮ $\cdots$

S_2 : $[1P2[a.$ ▮ $[1E2[P.\times E$ ▮ $\cdots$

S_3 : $[1E3[P\times .E$ ▮ $[3E3[.P$ ▮ $[3P3[.a$ ▮ $\cdots$

S_4 : $[3P4[a.$ ▮ $[3E4[P.$ ▮ $[1E4[P\times E.$ ▮ $[1A4[E.-(A)$ ▮ $\cdots$

S_5 : $[1A5[E-.(A)$ ▮ $\cdots$

S_6 : $[1A6[E-(.A)$ ▮ $[6A6[.E+A$ ▮ $[6E6[.P$ ▮ $[6P6[.a$ ▮ $\cdots$

S_7 : $[6P7[a.$ ▮ $[6E7[P.$ ▮ $[6A7[E.+A$ ▮ $\cdots$

S_8 : $[6A8[E+.A$ ▮ $[8A8[.E$ ▮ $[8E8[.P$ ▮ $[8P8[.a$ ▮ $\cdots$

S_9 : $[8P9[a.$ ▮ $[8E9[P.$ ▮ $[8A9[E.$ ▮
$[6A9[E+A.$ ▮ $[1A9[E-(A.)$ ▮ $\cdots$

S_{10} : $[1A10[E-(A).$ ▮ $[1S10[A.$ ▮ $\cdots$ (Rückgabe $x \in L$).

Vorbereitung 3

Beantworten Sie kurz die folgenden Fragen:

1. Gibt es eine Turingmaschine, die sich nie mehr als vier Schritte vom Startzustand entfernt und eine unendliche Sprache akzeptiert? Begründung!

2. Welche Sprachen lassen sich mit Turingmaschinen, die ihren Kopf immer nur nach rechts bewegen, erkennen?

3. Gibt es für jede Turingmaschine T eine Turingmaschine T' mit nur einem Zustand, die die Sprache von T akzeptiert?

Lösung

1. Ja, die Turingmaschine kann einfach unabhängig von der Eingabe direkt in einen akzeptierenden Zustand wechseln.

2. Die regulären Sprachen. Da der Kopf sich nur nach rechts bewegt, liest eine solche Turingmaschine nur einmal die Eingabe, gegebenenfalls gefolgt von (unendlich) vielen Blanks. Nachdem die Eingabe gelesen wurde, hängt es nur noch von dem aktuellen Zustand der TM ab, ob irgendwann ein Endzustand erreicht werden wird, nicht mehr vom Band.

3. Nein! Die Sprache, die eine TM akzeptiert, ist über Endzustände definiert. Sei also T eine TM mit nur einem Zustand q. Dann gibt es zwei Fälle:

   $q \notin F$. Dann ist $L(T) = \emptyset$, da nie ein Endzustand erreicht werden kann.

   $q \in F$. Dann ist $L(T) = \Sigma^*$, da die TM gleich zu Beginn in einem Endzustand ist und es damit eine akzeptierende Konfigurationsfolge gibt.

   Also braucht man für jede nicht-triviale Sprache mehrere Zustände.

   Bemerkung: Auf ähnliche Weise (aber nicht genauso) kann man zeigen, dass die entsprechende Aussage für Kellerautomaten auch falsch ist, wenn man Akzeptieren mit Endzuständen betrachtet.

Vorbereitung 4

Wir konstruieren eine Turingmaschine $T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$, mit $\Sigma = \{|\}$ wie folgt.

Zu Beginn steht, außer Leerzeichen, nur eine Sequenz von Strichen auf dem Band. Der Schreib-/Lesekopf der Turing-Maschine steht auf dem ersten Strich (von links gesehen). Die Berechnung erfolgt, indem jeweils der erste Strich (von links gesehen) durch ein Hilfszeichen X ersetzt wird und zusätzlich ein Hilfszeichen X an das linke Ende geschrieben wird. Zum Schluß werden alle Hilfszeichen von rechts nach links durch Striche ersetzt.
Sei $T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ mit $Q = \{q_0, q_1, q_2, q_3\}$, $\Sigma = \{|\}$, $\Gamma = \{|, X, \square\}$ und $F = \{q_3\}$. Die Übergangsfunktion δ entnehmen wir der folgenden Tabelle:

Übergang	Kommentar
$\delta(q_0, \|) = (q_1, X, L)$	ersetze erstes | durch Hilfszeichen X
$\delta(q_1, X) = (q_1, X, L)$	gehe nach links zum ersten X
$\delta(q_1, \square) = (q_0, X, R)$	füge zusätzliches Hilfszeichen X am Anfang ein
$\delta(q_0, X) = (q_0, X, R)$	gehe nach rechts zum ersten |
$\delta(q_0, \square) = (q_2, \square, L)$	alle | abgearbeitet
$\delta(q_2, X) = (q_2, \|, L)$	ersetze X durch |
$\delta(q_2, \square) = (q_3, \square, R)$	alle X durch | ersetzt, Stopp

Spezifizieren Sie möglichst knapp den Bandinhalt in Abhängigkeit der Eingabe, wenn die Turingmaschine anhält.

Lösung

T verdoppelt die eingegebene Strichzahl.

Tutoraufgabe 1

1. Zeigen Sie: Die Sprache $L = \{w \in \{0,1\}^* \,;\, w \text{ enthält gleich viele Nullen und Einsen}\}$ ist deterministisch kontextfrei.

2. Die Sprache $L = \{a^n b^n \,;\, n \in \mathbb{N}_0\} \cup \{a^n b^{2n} \,;\, n \in \mathbb{N}_0\}$ ist zwar kontextfrei, sie ist aber keine deterministisch kontextfreie Sprache. Beweisen Sie dies durch Widerspruch, indem Sie annehmen, dass es einen DPDA M gibt, der L erkennt, und dann mit modifizierten Kopien M' und M'' von M einen DPDA $\tilde{M}$ konstruieren, der die Sprache $\{a^n b^n c^n \,;\, n \in \mathbb{N}_0\}$ erkennt.

Lösung

1. Bemerkung: L erfüllt nicht die Präfixbedingung!

   Der folgende deterministische Kellerautomat

   $$M = (\{q_0, q_1\}, \{0,1\}, \{0,1,N,E,Z_0\}, \delta, q_0, Z_0, \{q_0\})$$

   akzeptiert L mit akzeptierendem Zustand q_0.

   $$\begin{array}{lcllcl}
   \delta(q_0,0,Z_0) &=& \{(q_1, NZ_0)\} & \delta(q_0,1,Z_0) &=& \{(q_1, EZ_0)\} \\
   \delta(q_1,0,0) &=& \{(q_1, 00)\} & \delta(q_1,1,1) &=& \{(q_1, 11)\} \\
   \delta(q_1,0,1) &=& \{(q_1, \epsilon)\} & \delta(q_1,1,0) &=& \{(q_1, \epsilon)\} \\
   \delta(q_1,0,N) &=& \{(q_1, 0N)\} & \delta(q_1,1,E) &=& \{(q_1, 1E)\} \\
   \delta(q_1,0,E) &=& \{(q_0, \epsilon)\} & \delta(q_1,1,N) &=& \{(q_0, \epsilon)\}
   \end{array}$$

   Das leere Wort wird akzeptiert, weil der Startzustand auch Endzustand ist.

   Das erste Eingabezeichen muss im Keller besonders gekennzeichnet sein (N bzw. E), damit bei Erfüllung der Gleichheit von eingelesenen Nullen bzw. Einsen in den akzeptierenden Zustand q_0 gegangen werden kann.

2. Bemerkung: L erfüllt nicht die Präfixbedingung!

   Sei $M = (Q, \Sigma, \Delta, q_0, Z_0, \delta, F)$ ein deterministischer Kellerautomat, der L akzeptiert mit $Q = \{q_0, \ldots, q_m\}$, $\Sigma = \{a, b\}$, $\Delta = \{Z_0, \ldots, Z_n\}$ und $F = \{q_k, \ldots, q_m\}$. Seien M' und M'' zwei Kopien von M.

   Wir konstruieren nun einen Automaten $\widetilde{M}$ wie folgt. Zuerst modifizieren wir M'' so, dass $\Sigma'' = \{a, c\}$. Die Übergangsfunktion passen wir an, so dass $\delta''(q_i'', a, X) = (q_j'', \alpha)$ falls $\delta(q_i, a, X) = (q_j, \alpha)$ und $\delta''(q_i'', c, X) = (q_j'', \alpha)$ falls $\delta(q_i, b, X) = (q_j, \alpha)$.

   Die Zustände von $\widetilde{M}$ sind die von M' und M'', also $\tilde{Q} = \{q_0', \ldots, q_m', q_0'', \ldots, q_m''\}$. Das Bandalphabet ist $\tilde{\Sigma} = \{a, b, c\}$. Das Kelleralphabet setzen wir $\tilde{\Delta} = \Delta' = \Delta'' = \Delta$. Das unterste Kellersymbol ist folglich $\tilde{Z}_0 = Z_0$. Der Anfangszustand ist q_0' und die Endzustände sind $\tilde{F} = F''$. Die Übergangsfunktion $\tilde{\delta}$ ist

   $$\begin{aligned}
   \tilde{\delta} &= \delta' \cup \delta'' \\
   &\quad \cup \{(q_i', x, X) \to (q_j'', \alpha) \,;\ \text{falls } \delta''(q_i'', x, X) = (q_j'', \alpha) \text{ für } x \in \Sigma'' \text{ und } q_i' \in F'\}
   \end{aligned}$$

Falls der Automat $\widetilde{M}$ ein Wort akzeptiert, beginnt er im Zustand $q'_0 \in Q'$, macht irgendwann einen Übergang von einem Zustand $q'_i \in F'$ zu einem Zustand $q''_j \in Q''$ und endet in Zustand $q''_i \in F''$. Betrachtet man nur die Indizes der Zustände, so verhält sich $\widetilde{M}$ wie M, außer dass er in der zweiten Phase c für b liest. Daher gilt: $\widetilde{M}$ akzeptiert ein Wort w, genau dann, wenn $w = w_1w_2$ ist, so dass w_1 von M akzeptiert wird und w_1w_2 nach Ersetzen aller c's durch b's von M akzeptiert wird. Um einen akzeptierenden Zustand von M' zu erreichen, muss w_1 entweder a^ib^i oder a^ib^{2i} sein. Da $a^ib^{2i}w'$ für $w' \in \Sigma^+$ nicht von M akzeptiert wird, muss $w_1 = a^ib^i$ sein. Der zweite Teil w_2 kann nicht leer sein, weil sonst kein Übergang zu Zuständen aus Q'' stattfindet. Daher kann w_2 nur c^i sein, da a^ib^i mit einem nicht leeren Suffix nur durch b^i zu einem akzeptierten Wort ergänzt werden kann. Also muss $w = a^ib^ic^i$ sein.

Umgekehrt gilt für ein Wort $w = a^ib^ic^i$ Folgendes.

Da $\widetilde{M}$ deterministisch ist, befindet sich der Automat nach $2i$ Schritten in einem Zustand $q'_i \in F'$. Da a^ib^{2i} auch akzeptiert wird, muss es eine Folge von Zuständen $q'_i = q'_{j_1}, \ldots, q'_{j_t} = q'_j \in F'$ geben, so dass i mal ein b gelesen wird. Also gibt es auch eine Folge $q''_{j_1}, \ldots, q''_{j_t} = q''_j \in F''$, so dass i mal ein b gelesen wird, und einen Übergang $(q'_i, c, X) \to (q''_{j_1}, \alpha)$. Der Stack verhält sich dabei exakt so, als würde M das Wort a^ib^{2i} lesen.

Im Ergebnis akzeptiert $\widetilde{M}$ genau die Sprache $L(\widetilde{M}) = \{a^ib^ic^i \ ; \ n \in N_0\}$. Dies kann aber nicht sein, weil $L(\widetilde{M})$ nicht kontextfrei ist.

Tutoraufgabe 2

Geben Sie eine deterministische Turingmaschine $T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ an, mit $\Sigma = \{|\}$, die eine eingegebene Strichzahl verdoppelt.
Begründen Sie, warum Ihre Turingmaschine die Spezifikation erfüllt.

Lösung

Siehe VA 4.

Begründung: Angenommen der Lesekopf steht auf dem ersten Strich von links gesehen. Wenn er das nächste Mal auf dem ersten Strich steht, dann gibt es zwei X mehr auf dem Band und einen Strich weniger. Rest per Induktion.

Methodisch gesehen beweist man Aussagen über Turingmaschinen häufig auf der Grundlage einer Programmbeschreibung mit Teilmaschinen, für die dann Aussagen bewiesen werden, die zusammengesetzt die zu beweisende Aussage darstellen.

Tutoraufgabe 3

Sei $\Sigma = \{0, 1\}$. Wir bezeichnen mit $\overline{w}$ die Negation eines Wortes $w \in \{0, 1\}^*$, d.h. alle Nullen werden durch Einsen ersetzt und umgekehrt.

1. Geben Sie eine deterministische Turingmaschine an, die für ein Eingabewort $w \in \Sigma^*$ folgende Berechnung durchführt: Am Ende der Berechnung steht auf dem Band das Wort $w\overline{w}$ und der Kopf steht in einem Endzustand auf dem ersten Zeichen dieses Wortes.

   Kommentieren Sie ihre Konstruktion durch eine informelle Beschreibung Ihrer Lösungsidee.

2. Geben Sie nun eine Turingmaschine an, die die Sprache $L = \{w\overline{w}\,;\, w \in \Sigma^*\}$ akzeptiert. Kommentieren Sie wiederum ihre Konstruktion durch eine informelle Beschreibung Ihrer Lösungsidee.

Lösung

1. Wir formulieren die Turingmaschine so, dass sie auf dem am weitesten links liegenden Zeichen der Eingabe startet. Das Bandalphabet ist $\Gamma = \{0, 1, \widehat{0}, \widehat{1}, \square\}$, wobei wir $\widehat{0}$ und $\widehat{1}$ nutzen, um die Zeichen zu markieren, die nicht mehr kopiert werden müssen.

   Wir gehen wie folgt vor:

   (a) Markiere das Zeichen unter dem Cursor als kopiert; dann bewege den Kopf an das rechte Ende des Bandes und füge dort die Negation des Zeichens markiert ein.

   (b) Suche das am weitesten links stehende, nicht markierte Zeichen.

   (c) Gibt es ein solches Zeichen, springe zum ersten Schritt. Ansonsten steht jetzt $w\overline{w}$ auf dem Band, wobei jedes Zeichen markiert ist. Entferne die Markierungen.

   Formal definieren wie die Turingmaschine als $M = (Q, \{0,1\}, \Gamma, \delta, q_0, \square, \{q_5\})$, wobei $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$ und δ wie folgt definiert ist:

	0	1	$\widehat{0}$	$\widehat{1}$	$\square$
q_0	$(q_1, \widehat{0}, R)$	$(q_2, \widehat{1}, R)$	$(q_0, \widehat{0}, R)$	$(q_0, \widehat{1}, R)$	$(q_4, \square, L)$
q_1	$(q_1, 0, R)$	$(q_1, 1, R)$	$(q_1, \widehat{0}, R)$	$(q_1, \widehat{1}, R)$	$(q_3, \widehat{1}, L)$
q_2	$(q_2, 0, R)$	$(q_2, 1, R)$	$(q_2, \widehat{0}, R)$	$(q_2, \widehat{1}, R)$	$(q_3, \widehat{0}, L)$
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	$(q_3, \widehat{0}, L)$	$(q_3, \widehat{1}, L)$	$(q_0, \square, R)$
q_4	$(q_4, 0, L)$	$(q_4, 1, L)$	$(q_4, 0, L)$	$(q_4, 1, L)$	$(q_5, \square, R)$

2. Wir konstruieren eine Turingmaschine, die eine Eingabe genau dann akzeptiert, wenn diese die Form $w\overline{w}$ hat. Diesmal verwenden wir zwei unterschiedliche Markierungen, $\widehat{\ }$ und $\widetilde{\ }$. Die erste verwenden wir, um bereits verarbeitete Zeichen aus w zu markieren, die zweite für bereits verarbeitete Zeichen aus $\overline{w}$. Daher ist $\Sigma = \{0,1\}$ und $\Gamma = \Sigma \cup \{\widehat{0}, \widehat{1}, \widetilde{0}, \widetilde{1}, \square\}$. Wir gehen wiederum davon aus, dass der Kopf anfangs links auf der Eingabe steht.

   Zunächst eine informelle Beschreibung:

   (a) Markiere das erste Zeichen mit $\widehat{\ }$ und rate nicht-deterministisch die Position des ersten Zeichens aus $\overline{w}$. Markiere dieses Zeichen mit $\widetilde{\ }$.

   (b) Bewege den Kopf auf das am weitesten links stehende, unmarkierte Zeichen und markiere es mit $\widehat{\ }$. Gibt es kein solches Zeichen, springe zu Schritt (d).

   Ansonsten vergleiche es mit dem ersten unmarkierten Zeichen nach einer $\widetilde{\ }$-Markierung. Sind die beiden Zeichen ungleich, markiere das zweite mit $\widetilde{\ }$ und wiederhole diesen Schritt. Andernfalls wird das Wort nicht akzeptiert.

   (c) Wiederhole Schritt (b)

(d) Überprüfe, ob es noch ein unmarkiertes Zeichen gibt. Wenn nein, akzeptiere die Eingabe.

Formal definieren wie die Turingmaschine als $M = (Q, \Sigma, \Gamma, \delta, q_0, \square, \{q_f\})$, wobei $Q = \{q_0, \ldots, q_9, q_f\}$ und die partielle Übergangsfunktion δ wie folgt definiert ist.

	0	1	$\widehat{0}$	$\widehat{1}$	$\widetilde{0}$	$\widetilde{1}$	$\square$
q_0	$(q_1, \widehat{0}, R)$	$(q_2, \widehat{1}, R)$					$(q_f, \square, N)$
q_1	$(q_1, 0, R)$	$(q_1, 1, R)$					
		$(q_3, \widetilde{1}, L)$					
q_2	$(q_2, 0, R)$	$(q_2, 1, R)$					
	$(q_3, \widetilde{0}, L)$						
q_3	$(q_3, 0, L)$	$(q_3, 1, L)$	$(q_3, \widehat{0}, L)$	$(q_3, \widehat{1}, L)$	$(q_3, \widetilde{0}, L)$	$(q_3, \widetilde{1}, L)$	$(q_4, \square, R)$
q_4	$(q_5, \widehat{0}, R)$	$(q_7, \widehat{1}, R)$	$(q_4, \widehat{0}, R)$	$(q_4, \widehat{1}, R)$	$(q_9, \widehat{0}, R)$	$(q_9, \widehat{1}, R)$	
q_5	$(q_5, 0, R)$	$(q_5, 1, R)$			$(q_6, \widetilde{0}, R)$	$(q_6, \widetilde{1}, R)$	
q_6		$(q_3, \widetilde{1}, L)$			$(q_6, \widetilde{0}, R)$	$(q_6, \widetilde{1}, R)$	
q_7	$(q_7, 0, R)$	$(q_7, 1, R)$			$(q_8, \widetilde{0}, R)$	$(q_8, \widetilde{1}, R)$	
q_8	$(q_3, \widetilde{0}, L)$				$(q_8, \widetilde{0}, R)$	$(q_8, \widetilde{1}, R)$	
q_9					$(q_9, \widetilde{0}, R)$	$(q_9, \widetilde{1}, R)$	$(q_f, \square, N)$

Die Bedeutungen der Zustände sind dabei wie folgt:

- q_1 bzw. q_2: Rate die Anfangsposition von $\overline{w}$, wenn das erste Zeichen von w 0 bzw. 1 ist.
- q_3: Bewege den Kopf ans linke Ende des beschriebenen Bandes.
- q_4: Finde das erste nicht-markierte Zeichen (in w).
- q_5, q_6 bzw. q_7, q_8: Vergleiche 0 bzw. 1 mit dem ersten, nicht-markierten Zeichen in $\overline{w}$.
- q_9: Stelle sicher, dass alle Zeichen markiert sind.

Tutoraufgabe 4

Sei Σ ein Alphabet. Geben Sie für die kontextsensitive Sprache $L = \{ww\,;\, w \in \Sigma^*\}$ einen linear beschränkten Automaten M an, der L akzeptiert.

Lösung

Wir sind aufgefordert, eine Turingmaschine $T = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$ zu definieren, für deren Übergangsfunktion die Einschränkung gilt, dass das leere Feld, d. h. das Bandzeichen $\square$, nicht überschrieben werden darf.

Wenn man für ein Wort $v \in \Sigma^*$ prüfen will, ob $v \in L$ gilt, wird man zunächst die Mitte von v suchen und anschließend buchstabenweise die Gleichheit der entsprechenden Teilwörter v_1, v_2 von $v = v_1 v_2$ prüfen. Die Mitte eines Wortes v kann man durch wiederholte, unterschiedliche Markierung des ersten bzw. letzten Buchstaben von v erhalten.
Für Markierungen des linken bzw. rechten Teilwortes v_1 bzw. v_2 benützen wird die Alphabete

$$\Sigma_a = \{(x, 1)\,;\, x \in \Sigma\} \quad \text{und} \quad \Sigma_A = \{(x, 2)\,;\, x \in \Sigma\}\,,$$

wobei beide Alphabete als Kopien von Σ aufgefasst werden können. Wir schreiben für Elemente $x \in \Sigma$ das entsprechende Element in Σ_a bzw. Σ_A als x_a bzw. x_A.
Wir benötigen des Weiteren noch ein Hilfszeichen X. Zusammenfassend definieren wir $\Gamma = \Sigma \cup \Sigma_a \cup \Sigma_A \cup \{X, \square\}$ als Bandalphabet des zu konstruierenden LBA.

Der Anfangszustand sei q_0, der Endzustand q_f. Für jedes Symbol $x \in \Sigma$ definieren wir einen Zustand q_x. Die Zustände q_x werden die Aufgabe übernehmen, sich den Buchstaben x zu "merken". Die Übergangsfunktion sei in der folgenden Tabelle gegeben für alle $x \in \Sigma$.

Übergang			Kommentar
$\delta(q_0, \square)$	$\rightarrow$	$(q_f, \square, N)$	das leere Eingabewort wird akzeptiert
$\delta(q_0, x)$	$\rightarrow$	(q_0', x, N)	ein nichtleeres Wort wird dem Zustand q_0' zur Verarbeitung gegeben
$\delta(q_0', x)$	$\rightarrow$	(q_1, x_a, R)	Markierung des ersten Buchstaben des linken Teilwortes
$\delta(q_1, x)$	$\rightarrow$	(q_1, x, R)	die Σ-Eingabe wird nach rechts überlaufen
$\delta(q_1, \square)$	$\rightarrow$	$(q_2, \square, L)$	Stopp erster Rechtsdurchlauf bis zum rechten Rand
$\delta(q_1, x_A)$	$\rightarrow$	(q_2, x_A, L)	Stopp Rechtsdurchlauf bis zum rechten Teilwort
$\delta(q_2, x)$	$\rightarrow$	(q_3, x_A, L)	Markierung x durch x_A und Übergang zum Rücklauf
$\delta(q_3, x)$	$\rightarrow$	(q_4, x, L)	
$\delta(q_3, x_a)$	$\rightarrow$	(q_5, x_a, N)	Stopp Rücklauf und Beginn der 2.Phase
$\delta(q_4, x)$	$\rightarrow$	(q_4, x, L)	
$\delta(q_4, x_a)$	$\rightarrow$	(q_0', x_a, R)	

Mit Erreichen des Zustandes q_5 ist klar, dass das Eingabewort v in gleichlange Wörter v_1, v_2 zerlegbar ist. Der Schreib-/Lesekopf befindet sich auf dem letzten, entsprechend markierten Buchstaben von v_1. v_1 und v_2 sind markiert.

Übergang			Kommentar
$\delta(q_5, x_a)$	$\rightarrow$	(q_x, X, R)	
$\delta(q_x, X)$	$\rightarrow$	(q_x, X, R)	
$\delta(q_x, y_A)$	$\rightarrow$	(q_x, y_A, R)	
$\delta(q_x, \square)$	$\rightarrow$	$(q_x', \square, L)$	
$\delta(q_x', x_A)$	$\rightarrow$	$(q_7, \square, L)$	
$\delta(q_7, x_A)$	$\rightarrow$	(q_7, x_A, L)	
$\delta(q_7, X)$	$\rightarrow$	(q_7, X, L)	
$\delta(q_7, x_a)$	$\rightarrow$	(q_5, x_a, N)	
$\delta(q_7, \square)$	$\rightarrow$	$(q_f, \square, N)$	

Die Möglichkeit, bei Turingmaschinen den Lese-/Schreibkopf nicht zu bewegen, hilft bei der Strukturierung von Turingprogrammen.