2. Übungsblatt (mit Lösungen)

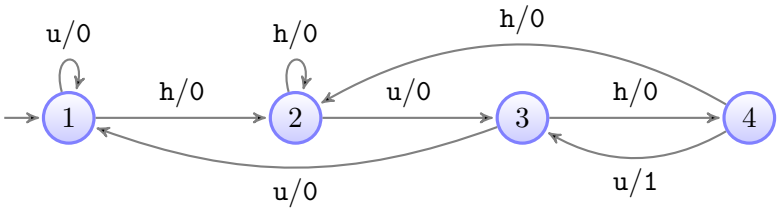
3.0 VU Formale Modellierung

Marion Brandsteidl, Gernot Salzer

30. Mai 2013 (Korrektur 14.6.)

Aufgabe 1 (0.3 Punkte)

Sei $\mathcal{A}$ der folgende Mealy-Automat.



(a) Geben Sie die Ausgabe zur Eingabe `uhuhuhuuhhuhu` an.

(b) Beschreiben Sie $\mathcal{A}$ als 6-Tupel; legen Sie die Übergangsfunktion δ sowie die Ausgabefunktion γ durch eine Tabelle fest.

(c) Berechnen Sie schrittweise $\delta^*(1, \texttt{huhuu})$ und $\gamma^*(1, \texttt{huhuu})$.

(d) Beschreiben Sie die Übersetzungsfunktion $[\mathcal{A}]$.

Lösung

(a) $\gamma^*(1, \texttt{uhuhuhuuhhuhu}) = \texttt{0000101000001}$

(b) $\mathcal{A} = \langle \{1,2,3,4\}, \{\texttt{h}, \texttt{u}\}, \{\texttt{0}, \texttt{1}\}, \delta, \gamma, 1 \rangle$, wobei δ und γ durch die folgenden Tabellen festgelegt werden.

δ	h	u
1	2	1
2	2	3
3	4	1
4	2	3

γ	h	u
1	0	0
2	0	0
3	0	0
4	0	1

(c)
$$\begin{aligned}\delta^*(1,\mathtt{huhuu}) &= \delta^*(\delta(1,\mathtt{h}),\mathtt{uhuu})\\ &= \delta^*(2,\mathtt{uhuu})\\ &= \delta^*(\delta(2,\mathtt{u}),\mathtt{huu})\\ &= \delta^*(3,\mathtt{huu})\\ &= \delta^*(\delta(3,\mathtt{h}),\mathtt{uu})\\ &= \delta^*(4,\mathtt{uu})\\ &= \delta^*(\delta(4,\mathtt{u}),\mathtt{u})\\ &= \delta^*(3,\mathtt{u})\\ &= \delta^*(\delta(3,\mathtt{u}),\varepsilon)\\ &= \delta^*(1,\varepsilon)\\ &= 1\end{aligned}$$

$$\begin{aligned}\gamma^*(1,\mathtt{huhuu}) &= \gamma(1,\mathtt{h})\cdot\gamma^*(\delta(1,\mathtt{h}),\mathtt{uhuu})\\ &= \mathtt{0}\cdot\gamma^*(2,\mathtt{uhuu})\\ &= \mathtt{0}\cdot\gamma(2,\mathtt{u})\cdot\gamma^*(\delta(2,\mathtt{u}),\mathtt{huu})\\ &= \mathtt{0}\cdot\mathtt{0}\cdot\gamma^*(3,\mathtt{huu})\\ &= \mathtt{00}\cdot\gamma(3,\mathtt{h})\cdot\gamma^*(\delta(3,\mathtt{h}),\mathtt{uu})\\ &= \mathtt{00}\cdot\mathtt{0}\cdot\gamma^*(4,\mathtt{uu})\\ &= \mathtt{000}\cdot\gamma(4,\mathtt{u})\cdot\gamma^*(\delta(4,\mathtt{u}),\mathtt{u})\\ &= \mathtt{000}\cdot\mathtt{1}\cdot\gamma^*(3,\mathtt{u})\\ &= \mathtt{0001}\cdot\gamma(3,\mathtt{u})\cdot\gamma^*(\delta(3,\mathtt{u}),\varepsilon)\\ &= \mathtt{0001}\cdot\mathtt{0}\cdot\gamma^*(1,\varepsilon)\\ &= \mathtt{00010}\cdot\varepsilon = \mathtt{00010}\end{aligned}$$

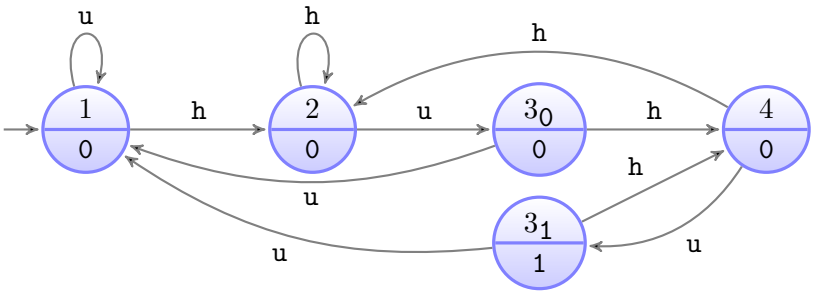
(d) Der Mealy-Automat ist ein `huhu`-Detektor: Immer wenn die letzten vier Eingabesymbole das Wort `huhu` bilden, wird das Symbol `1` ausgegeben, sonst `0`.

Aufgabe 2 (0.3 Punkte)

Finden Sie einen Moore-Automaten, der äquivalent zum Mealy-Automaten aus Aufgabe 1 ist. Geben Sie ein Verfahren an, mit dem sich zu jedem Mealy-Automaten ein äquivalenter Moore-Automat konstruieren lässt.

Lösung

Ein Moore-Automat zeichnet sich dadurch aus, dass sämtliche Übergänge, die zu einem Zustand führen, dieselbe Ausgabe erzeugen. Der Mealy-Automat aus Aufgabe 1 erfüllt diese Bedingung beinahe. Nur zum Zustand 3 führen Übergänge mit den unterschiedlichen Ausgaben `0` und `1`. Wir verdoppeln daher diesen Zustand samt den Übergängen, die vom Zustand wegführen. Die zum Zustand 3 führenden Übergänge teilen wir je nach Ausgabe auf die beiden Zustände auf.



Zur allgemeinen Beschreibung des Verfahrens gehen wir von einem beliebigen Mealy-Automaten $\mathcal{A} = \langle Q, \Sigma, \Gamma, \delta, \gamma, q_0\rangle$ aus. Wir definieren zunächst die Menge Γ_q aller Ausgabesymbole, die beim Übergang in den Zustand q auftreten. Formal:

$$\Gamma_q = \{\, a \in \Gamma \mid \begin{array}{l}\text{Es gibt einen Zustand } p \in Q \text{ und ein Eingabesym-}\\ \text{bol } s \in \Sigma, \text{ sodass } \delta(p,s) = q \text{ und } \gamma(p,s) = a \text{ gilt.}\end{array} \,\}$$

Für den Mealy-Automaten aus Aufgabe 1 gilt $\Gamma_1 = \Gamma_2 = \Gamma_4 = \{\mathtt{0}\}$ und $\Gamma_3 = \{\mathtt{0}, \mathtt{1}\}$.
Wenn die Menge Γ_q für jeden Zustand q nur ein Element enthält ($|\Gamma_q| = 1$ für alle $q \in Q$), dann bedeutet das, dass alle Übergänge in denselben Zustand dieselbe Ausgabe erzeugen. In diesem Fall kann man den Mealy-Automaten $\mathcal{A}$ direkt als Moore-Automaten $\bar{\mathcal{A}} = \langle Q, \Sigma, \Gamma, \delta, \bar{\gamma}, q_0 \rangle$ interpretieren, wobei die Ausgabefunktion $\bar{\gamma}$ festgelegt wird durch $\bar{\gamma}(q) = a$ falls $\Gamma_q = \{a\}$.[1]
Andernfalls wählt man einen Zustand q aus, bei dem mehr als ein Ausgabesymbol auftritt; sei $\Gamma_q = \{a_1, \ldots, a_n\}$, wobei $n > 1$. Wir definieren einen neuen Mealy-Automaten $\mathcal{A}'$, der genauso aussieht wie der ursprüngliche, nur ersetzen wir q durch n neue Zustände $q_{a_1}, \ldots, q_{a_n}$. Die neuen Zustände besitzen dieselben wegführenden Übergänge wie q, die zu q hinführenden Übergänge werden aber abhängig vom Ausgabesymbol aufgeteilt. Formal lässt sich der neue Automat definieren als $\mathcal{A}' = \langle Q', \Sigma, \Gamma, \delta', \gamma', q_0' \rangle$, wobei Q', δ', γ' und q_0' folgendermaßen definiert sind.

- Die neuen Zustände sind die alten, wobei q durch $\{q_{a_1}, \ldots, q_{a_n}\}$ ersetzt wird:

$$Q' = (Q \setminus \{q\}) \cup \{q_{a_1}, \ldots, q_{a_n}\}$$

- Wenn q Startzustand in $\mathcal{A}$ ist, wähle einen beliebigen der neuen Zustände als Startzustand in $\mathcal{A}'$, z.B. q_{a_1}; andernfalls bleibt q_0 Startzustand.

$$q_0' = \begin{cases} q_{a_1} & \text{falls } q_0 = q \\ q_0 & \text{sonst} \end{cases}$$

- Die Ausgabefunktion bleibt unverändert, wobei die neuen Zustände die Ausgabe von q übernehmen.

$$\gamma'(p, s) = \gamma(p, s) \quad \text{und} \quad \gamma'(q_a, s) = \gamma(q, s)$$

  für alle Zustände $p \in Q \setminus \{q\}$, alle Eingaben $s \in \Sigma$ und alle Ausgaben $a \in \Gamma_q$.

- Bei der Definition der neuen Übergangsfunktion müssen wir unterscheiden, ob der Übergang von q weg bzw. nach q hin führt.

$$\delta'(p, s) = \begin{cases} \delta(p, s) & \text{falls } \delta(p, s) \neq q \text{ (von nicht-}q\text{ nach nicht-}q\text{)} \\ q_{\gamma(p,s)} & \text{falls } \delta(p, s) = q \text{ (von nicht-}q\text{ nach }q\text{)} \end{cases}$$

$$\delta'(q_a, s) = \begin{cases} \delta(q, s) & \text{falls } \delta(q, s) \neq q \text{ (von }q\text{ nach nicht-}q\text{)} \\ q_{\gamma(q,s)} & \text{falls } \delta(q, s) = q \text{ (von }q\text{ nach }q\text{)} \end{cases}$$

  für alle Zustände $p \in Q \setminus \{q\}$, alle Eingaben $s \in \Sigma$ und alle Ausgaben $a \in \Gamma_q$.

Diese Ersetzung von Zuständen wiederholen wir solange, bis der Mealy-Automat direkt einem Moore-Automaten entspricht.

[1] Sollten zu einem Zustand q gar keine Übergänge führen, erhalten wir $\Gamma_q = \{\}$. In diesem Fall können wir $\bar{\gamma}(q)$ beliebig definieren, da der Wert mangels Übergang nie zum Tragen kommt. Diese Situation kann praktisch nur beim Anfangszustand auftreten; alle anderen derartigen Zustände könnte man entfernen, da sie nicht erreichbar sind.

Aufgabe 3 (0.3 Punkte)

Sei Σ das Alphabet $\{\texttt{a}, \texttt{f}, \texttt{l}, \texttt{r}, \texttt{x}\}$ und L die Menge aller Wörter über Σ, in denen `ralf` oder `ala` als Teilwort vorkommt.

(a) Geben Sie einen Posix Extended Regular Expression an, der die Sprache L beschreibt.

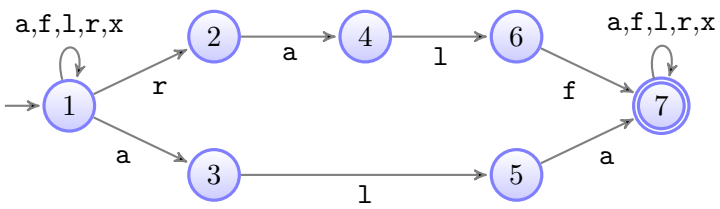
(b) Geben Sie einen nichtdeterministischen Automaten an, der die Sprache L akzeptiert. Der Automat soll der Definition der Sprache direkt entsprechen, sodass die Korrektheit der Modellierung unmittelbar einsichtig ist.

(c) Konstruieren Sie mit Hilfe des in der Vorlesung besprochenen Determinisierungsverfahrens zu Ihrem nichtdeterministischen Automaten einen äquivalenten deterministischen.

Lösung

(a) `[aflrx]*(ralf|ala)[aflrx]*`

(b)



(c) Wir stellen zunächst die Übergangsfunktion des indeterministischen Automaten als Tabelle dar.

δ	a	f	l	r	x
1	$\{1,3\}$	$\{1\}$	$\{1\}$	$\{1,2\}$	$\{1\}$
2	$\{4\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
3	$\{\}$	$\{\}$	$\{5\}$	$\{\}$	$\{\}$
4	$\{\}$	$\{\}$	$\{6\}$	$\{\}$	$\{\}$
5	$\{7\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
6	$\{\}$	$\{7\}$	$\{\}$	$\{\}$	$\{\}$
7	$\{7\}$	$\{7\}$	$\{7\}$	$\{7\}$	$\{7\}$

Damit lässt sich nun systematisch die Übergangsfunktion des deterministischen Automaten konstruieren, indem wir die jeweils relevanten Zeilen vereinigen. Wir wählen $\{1\}$ als Startzustand für den deterministischen Automaten, da 1 der Startzustand

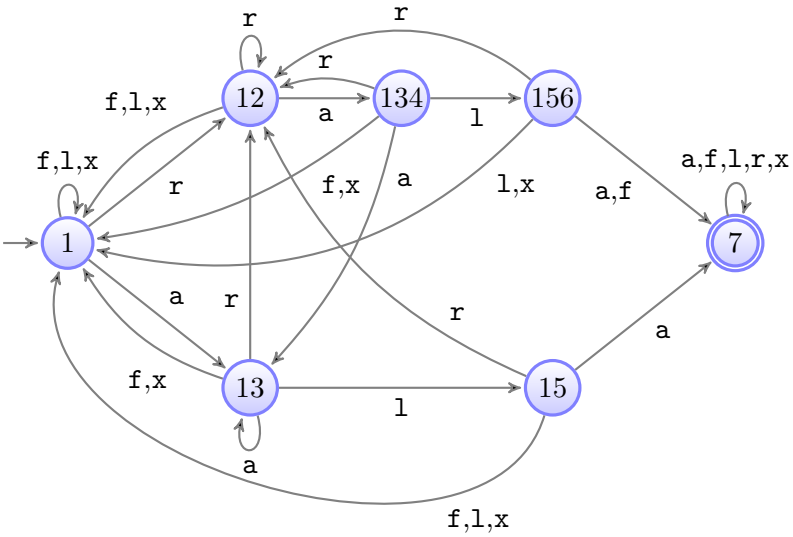


Abbildung 1: Deterministischer Automat für Ralf und Ala (Aufgabe 3)

des nicht-deterministischen ist.

δ	a	f	l	r	x
$\{1\}$	$\{1,3\}$	$\{1\}$	$\{1\}$	$\{1,2\}$	$\{1\}$
$\{1,2\}$	$\{1,3,4\}$	$\{1\}$	$\{1\}$	$\{1,2\}$	$\{1\}$
$\{1,3\}$	$\{1,3\}$	$\{1\}$	$\{1,5\}$	$\{1,2\}$	$\{1\}$
$\{1,3,4\}$	$\{1,3\}$	$\{1\}$	$\{1,5,6\}$	$\{1,2\}$	$\{1\}$
$\{1,5\}$	$\{1,3,7\}$	$\{1\}$	$\{1\}$	$\{1,2\}$	$\{1\}$
$\{1,5,6\}$	$\{1,3,7\}$	$\{1,7\}$	$\{1\}$	$\{1,2\}$	$\{1\}$
$\{1,3,7\}$	$\{1,3,7\}$	$\{1,7\}$	$\{1,5,7\}$	$\{1,2,7\}$	$\{1,7\}$
$\{1,7\}$	$\{1,3,7\}$	$\{1,7\}$	$\{1,7\}$	$\{1,2,7\}$	$\{1,7\}$
$\{1,5,7\}$	$\{1,3,7\}$	$\{1,7\}$	$\{1,7\}$	$\{1,2,7\}$	$\{1,7\}$
$\{1,2,7\}$	$\{1,3,4,7\}$	$\{1,7\}$	$\{1,7\}$	$\{1,2,7\}$	$\{1,7\}$
$\{1,3,4,7\}$	$\{1,3,7\}$	$\{1,7\}$	$\{1,5,6,7\}$	$\{1,2,7\}$	$\{1,7\}$
$\{1,5,6,7\}$	$\{1,3,7\}$	$\{1,7\}$	$\{1,7\}$	$\{1,2,7\}$	$\{1,7\}$

Die Endzustände sind all jene, die den ursprünglichen Endzustand 7 enthalten, also $\{1,3,7\}$, $\{1,7\}$, $\{1,5,7\}$, $\{1,2,7\}$, $\{1,3,4,7\}$ und $\{1,5,6,7\}$.

In diesem Beispiel haben die Endzustände die Eigenschaft, dass man von ihnen ausgehend mit jedem der fünf möglichen Symbole wieder nur in einen Endzustand gelangt. Sobald also ein Endzustand erreicht ist, wird ab hier jedes beliebige Wort akzeptiert. Wir können den Automaten also vereinfachen, indem wir die Endzustände zu einem einzigen zusammenfassen, den wir wieder 7 nennen. Abbildung 1 stellt den vereinfachten deterministischen Automaten graphisch dar.

Aufgabe 4 (0.3 Punkte)

Sei L die Menge aller nicht-leeren Wörter über $\{\texttt{0}, \texttt{1}, \texttt{2}\}$, die als Ternärnumeral (d.h., als Numeral zur Basis 3) interpretiert den Rest 1 oder 3 bei Division durch 5 liefern. Beispielsweise liegt das Wort `210` in L, da es der Zahl $2 \cdot 3^2 + 1 \cdot 3^1 + 0 \cdot 3^0 = 21$ entspricht, die bei Division durch 5 den Rest 1 liefert.

Geben Sie einen endlichen Automaten an, der die Sprache L akzeptiert.

Hinweis: Der Wert eines Numerals $z_1 \cdots z_k$ (z_i sind die einzelnen Ziffern) zur Basis n kann iterativ mittels der Formel

$$((\cdots(\text{wert}(z_1) \cdot n + \text{wert}(z_2)) \cdot n + \cdots) \cdot n + z_{k-1}) \cdot n + z_k$$

berechnet werden, man muss also nicht im Vorhinein die Anzahl der Ziffern kennen, um die Ziffern richtig zu gewichten. (wert(z) bezeichnet die der Ziffer z zugeordnete natürliche Zahl.) Da letztlich nur der Divisionsrest modulo 5 über die Akzeptanz des Wortes entscheidet, kann man sich auch in jedem Zwischenschritt auf den Divisionsrest modulo 5 beschränken (Restklassen-Arithmetik). Dadurch liegen alle Zwischenergebnisse im Bereich 0 bis 4.

Lösung

Da es um Teilbarkeit modulo 5 geht, sehen wir die Zustände 0, 1, 2, 3 und 4 für die fünf möglichen Divisionsreste vor. Befindet sich der Automat im Zustand n, so soll das bedeuten, dass das bisher gelesene Ternärnumeral den Rest n bei Division durch 5 liefert. Zustand 1 und 3 sind Endzustände, da sie dem Divisionsrest 1 bzw. 3 entsprechen.
Zur Festlegung der Übergänge überlegen wir, wie sich der Divsionsrest ändert, wenn auf das bisher verarbeitete Numeral das Symbol `0`, `1` bzw. `2` folgt. War der bisherige Rest n, so ist der Rest nach einem `0`er $(3n+0)$ mod 5, nach einem `1`er $(3n+1)$ mod 5 und nach einem `2`er $(3n+2)$ mod 5. Somit ergibt sich für den Zustand 4 bei Verarbeitung des Symbols `0` der Folgezustand $(3 \cdot 4 + 0) \bmod 5 = 2$, bei `1` der Folgezustand $(3 \cdot 4 + 1) \bmod 5 = 3$ und bei `2` der Folgezustand $(3 \cdot 4 + 2) \bmod 5 = 4$. Insgesamt erhalten wir den in Abbildung 2 dargestellten Automaten.

Aufgabe 5 (0.3 Punkte)

Sei $\Sigma = \{\texttt{S}, \texttt{R}, \texttt{W}, \texttt{B}\}$ das Eingabe- und $\Gamma = \{\texttt{0}, \texttt{1}\}$ das Ausgabealphabet. Die Eingabesymbole stehen für folgende Aktivitäten:

- `S`: Das Symbol `1` wird ausgeben (**S**et)
- `R`: Das Symbol `0` wird ausgeben (**R**eset)
- `W`: Die Ausgabe **W**echselt. War das letzte Symbol eine `1`, wird `0` ausgegeben, sonst `1`.
- `B`: Das zuletzt ausgegebene Symbol wird nochmals ausgegeben (**B**ewahren).

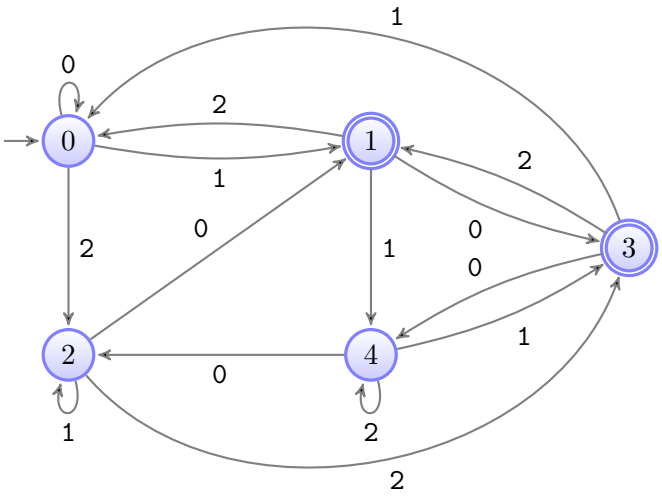
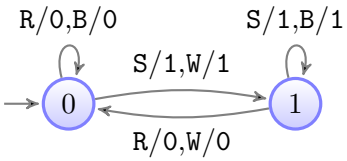


Abbildung 2: Automat für Ternärnumerale (Aufgabe 4)

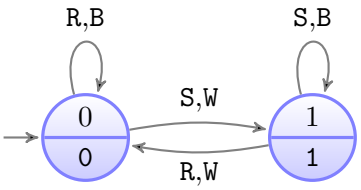
Beispielsweise führt die Eingabe `SBWRRBW` zur Ausgabe `1100001`.
Geben Sie einen Transducer an, der diese Funktion berechnet.

Lösung

Die Aktionen `W` und `B` beziehen sich auf die letzte Ausgabe, daher muss sich der Automat die beiden Möglichkeiten dafür, `0` und `1`, in unterschiedlichen Zuständen „merken“. Wir erhalten den folgenden Mealy-Automaten.



Der Startzustand ist willkürlich gewählt und bewirkt, dass `B` bzw. `W` als erste Aktion die Ausgabe `0` bzw. `1` liefert.
Der angegebene Mealy-Automat besitzt die Eigenschaft, dass alle Übergänge, die in denselben Zustand führen, dieselbe Ausgabe erzeugen. Diese ist also vom Übergang unabhängig und kann dem Zielzustand zugeordnet werden. Der Automat kann daher unverändert auch als Moore-Automat aufgefasst werden.



Aufgabe 6 (0.3 Punkte)

Seien $\mathcal{A}_1 = \langle Q_1, \Sigma, \delta_1, i_1, F_1 \rangle$ und $\mathcal{A}_2 = \langle Q_2, \Sigma, \delta_2, i_2, F_2 \rangle$ zwei beliebige deterministische Automaten über dem Alphabet Σ und $L_1 = \mathcal{L}(\mathcal{A}_1)$ bzw. $L_2 = \mathcal{L}(\mathcal{A}_2)$ die von ihnen akzeptierten Sprachen. Geben Sie ein Verfahren an, um daraus einen Automaten $\mathcal{A}$ für die Differenz dieser Sprachen zu erhalten; es soll also $\mathcal{L}(\mathcal{A}) = L_1 \setminus L_2$ gelten. Der Automat $\mathcal{A}$ akzeptiert somit genau jene Worte, die von $\mathcal{A}_1$ aber nicht von $\mathcal{A}_2$ akzeptiert werden. Welche Eigenschaft regulärer Sprachen ergibt sich daraus? Geben Sie als Beispiel zwei konkrete Automaten und den daraus mit Ihrem Verfahren konstruierten Automaten für die Differenzsprache an.

Hinweis: Überlegen Sie sich die Aufgabenstellung zuerst an Hand einfacher konkreter Automaten und verallgemeinern Sie dann Ihre Beobachtungen.

Lösung

Wir konstruieren einen Automaten $\mathcal{A}$, der die beiden Automaten $\mathcal{A}_1$ und $\mathcal{A}_2$ gleichzeitig ausführt. Als Zustände für $\mathcal{A}$ verwenden wir Paare (q_1, q_2), wobei $q_1 \in Q_1$ ein Zustand des ersten Automaten und $q_2 \in Q_2$ ein Zustand des zweiten Automaten ist. Der neue Automat befindet sich bei Eingabe eines Wortes w im Zustand (q_1, q_2), wenn sich der erste Automat bei diesem Wort im Zustand q_1 und der zweite im Zustand q_2 befinden würde. Der Startzustand (i_1, i_2) entspricht der Situation, in der sich die beiden ursprünglichen Automaten im Startzustand befinden. Ein Übergang mit dem Symbol s von (q_1, q_2) nach (q_1', q_2') existiert genau dann, wenn man mit diesem Symbol in $\mathcal{A}_1$ von q_1 nach $q_1' = \delta_1(q_1, s)$ und in $\mathcal{A}_2$ von q_2 nach $q_2' = \delta_2(q_2, s)$ gelangt.
Uns interessieren nun alle Wörter, die von $\mathcal{A}_1$ aber nicht von $\mathcal{A}_2$ akzeptiert werden, also die Differenz $\mathcal{L}(\mathcal{A}_1) \setminus \mathcal{L}(\mathcal{A}_2)$. Ein derartiges Wort liegt genau dann vor, wenn der neue Automat damit einen Zustand (q_1, q_2) erreicht, bei dem q_1 ein Endzustand des Automaten $\mathcal{A}_1$ aber q_2 kein Endzustand des Automaten $\mathcal{A}_2$ ist. Ein Automat für die Differenz der Sprachen $\mathcal{L}(\mathcal{A}_1)$ und $\mathcal{L}(\mathcal{A}_2)$ lässt sich somit durch

$$\mathcal{A} = \langle Q_1 \times Q_2, \Sigma, \delta, (i_1, i_2), F_1 \times (Q_2 \setminus F_2) \rangle$$

definieren, wobei die Übergangsfunktion festgelegt ist durch

$$\delta((q_1, q_2), s) = (\delta_1(q_1, s), \delta_2(q_2, s))$$

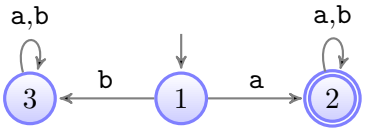
für alle $(q_1, q_2) \in Q_1 \times Q_2$ und alle $s \in \Sigma$.
Als Beispiel betrachten wir die Automaten

$$\mathcal{A}_1 = \langle \{1, 2, 3\}, \{\mathtt{a}, \mathtt{b}\}, \delta_1, 1, \{2\} \rangle$$
$$\mathcal{A}_2 = \langle \{x, y\}, \{\mathtt{a}, \mathtt{b}\}, \delta_2, x, \{y\} \rangle$$

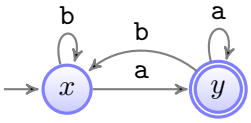
mit den Übergangsfunktionen

δ_1	a	b
1	2	3
2	2	2
3	3	3

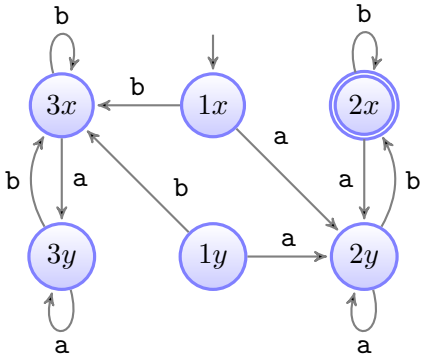
δ_2	a	b
x	y	x
y	y	x

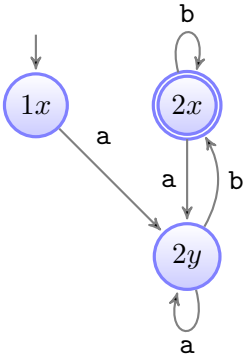


(a) Automat für $L_1 = \{\mathtt{a}\} \cdot \{\mathtt{a}, \mathtt{b}\}^*$



(b) Automat für $L_2 = \{\mathtt{a}, \mathtt{b}\}^* \cdot \{\mathtt{a}\}$





(c) Automat für die Sprache $L_1 \setminus L_2 = \{\mathtt{a}\} \cdot \{\mathtt{a}, \mathtt{b}\}^* \cdot \{\mathtt{b}\}$ (mit bzw. ohne Falle/unerreichbare Zustände)

Abbildung 3: Beispiel für die Differenzbildung bei Automaten (Aufgabe 6)

Automat $\mathcal{A}_1$ akzeptiert alle Worte über $\Sigma = \{\mathtt{a}, \mathtt{b}\}$, die mit $\mathtt{a}$ beginnen, und Automat $\mathcal{A}_2$ jene, die mit $\mathtt{a}$ aufhören, d.h., $\mathcal{L}(\mathcal{A}_1) = \{\mathtt{a}\} \cdot \{\mathtt{a}, \mathtt{b}\}^*$ und $\mathcal{L}(\mathcal{A}_2) = \{\mathtt{a}, \mathtt{b}\}^* \cdot \{\mathtt{a}\}$. Das oben beschriebene Verfahren liefert den Automaten

$$\mathcal{A} = \langle \{1x, 1y, 2x, 2y, 3x, 3y\}, \{\mathtt{a}, \mathtt{b}\}, \delta, 1x, \{2x\} \rangle$$

mit der Übergangsfunktion

δ	$\mathtt{a}$	$\mathtt{b}$
$1x$	$2y$	$3x$
$1y$	$2y$	$3x$
$2x$	$2y$	$2x$
$2y$	$2y$	$2x$
$3x$	$3y$	$3x$
$3y$	$3y$	$3x$

wobei die Zustandsbezeichnugnen (q_1, q_2) auf $q_1 q_2$ verkürzt wurden.[2] Für den Automaten $\mathcal{A}$ gilt $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \setminus \mathcal{L}(\mathcal{A}_2) = \{\mathtt{a}\} \cdot \{\mathtt{a}, \mathtt{b}\}^* \cdot \{\mathtt{b}\}$, d.h., der Automat akzeptiert alle Wörter, die mit $\mathtt{a}$ beginnen und mit $\mathtt{b}$ enden. Abbildung 3 stellt die drei Automaten graphisch dar.

[2] Die konkreten Bezeichnungen der Zustände haben keinen Einfluss auf die akzeptierte Sprache des Automaten. Sie können daher so gewählt werden, dass sie einen Anhaltspunkt für die Rolle bieten, die die Zutände im Automaten spielen.

Diese Konstruktion zeigt, dass die Differenz von Sprachen, die von Automaten akzeptiert werden, wieder durch einen Automaten beschrieben werden kann. Die Familie der Sprachen, die von endlichen Automaten akzeptiert werden, ist daher abgeschlossen gegenüber Mengendifferenz. In der Vorlesung haben wir gezeigt, dass diese Spachfamilie genau den regulären Sprachen entspricht. Daher sind auch die regulären Sprachen abgeschlossen gegenüber Differenzbildung: Die Differenz zweier regulärer Sprachen ist wieder regulär.

Aufgabe 7 (0.2 Punkte)

Vereinfachen Sie die folgenden Ausdrücke.

(a) $((\{\mathtt{a}, \mathtt{ab}, \mathtt{cc}\} \cup \{\varepsilon\}) \cdot (\{\varepsilon, \mathtt{bb}\} \cup \{\mathtt{b}\})) \cup \{\mathtt{c}\}$

(b) $\{\mathtt{0}\}^* \cdot (\{\mathtt{0}\} \cup \{\varepsilon\})$

(c) $(\{\mathtt{1}\}^* \cdot \{\mathtt{1}\}^+) \cdot \{\mathtt{1}\}^*$

(d) $\{\mathtt{a}, \mathtt{b}, \mathtt{ab}\} \cdot \{\}$

(e) $\{\mathtt{a}, \mathtt{b}, \mathtt{ab}\} \cup \{\}$

Lösung

(a) $L_1 = ((\{\mathtt{a}, \mathtt{ab}, \mathtt{cc}\} \cup \{\varepsilon\}) \cdot (\{\varepsilon, \mathtt{bb}\} \cup \{\mathtt{b}\})) \cup \{\mathtt{c}\}$
$= (\{\varepsilon, \mathtt{a}, \mathtt{ab}, \mathtt{cc}\} \cdot \{\varepsilon, \mathtt{b}, \mathtt{bb}\}) \cup \{\mathtt{c}\}$
$= \{\varepsilon, \mathtt{a}, \mathtt{ab}, \mathtt{abb}, \mathtt{abbb}, \mathtt{b}, \mathtt{bb}, \mathtt{cc}, \mathtt{ccb}, \mathtt{ccbb}\} \cup \{\mathtt{c}\}$
$= \{\varepsilon, \mathtt{a}, \mathtt{ab}, \mathtt{abb}, \mathtt{abbb}, \mathtt{b}, \mathtt{bb}, \mathtt{c}, \mathtt{cc}, \mathtt{ccb}, \mathtt{ccbb}\}$

(b) $L_2 = \{\mathtt{0}\}^* \cdot (\{\mathtt{0}\} \cup \{\varepsilon\}) = \{\mathtt{0}\}^* \cdot \{\mathtt{0}\} \cup \{\mathtt{0}\}^* \cdot \{\varepsilon\} = \{\mathtt{0}\}^+ \cup \{\mathtt{0}\}^* = \{\mathtt{0}\}^*$

(c) $L_3 = (\{\mathtt{1}\}^* \cdot \{\mathtt{1}\}^+) \cdot \{\mathtt{1}\}^* = \{\mathtt{1}\}^+ \cdot \{\mathtt{1}\}^* = \{\mathtt{1}\}^+$

(d) $L_4 = \{\mathtt{a}, \mathtt{b}, \mathtt{ab}\} \cdot \{\} = \{\}$

(e) $L_5 = \{\mathtt{a}, \mathtt{b}, \mathtt{ab}\} \cup \{\} = \{\mathtt{a}, \mathtt{b}, \mathtt{ab}\}$

Aufgabe 8 (0.3 Punkte)

Sind folgende Gleichungen für beliebige Sprachen L gültig? Falls ja, begründen Sie warum, falls nein, geben Sie ein Gegenbeispiel an.

(a) $L \cup \{\} = L \cdot \{\varepsilon\}$

(b) $\{\varepsilon\} \cdot L^* = L^+$

(c) $(L \cdot L)^* = L^* \cdot L^*$

(d) $L^+ \cup \{\varepsilon\} = L^* \cdot \{\}$

Lösung

(a) Diese Gleichung gilt für beliebige Sprachen L, da $L \cup \{\} = L = L \cdot \{\varepsilon\}$.

(b) Diese Gleichung gilt nicht allgemein. Wegen $\{\varepsilon\} \cdot L^* = L^*$ ist die Gleichung äquivalent zu $L^* = L^+$. Diese Gleichung ist genau dann erfüllt, wenn L das Leerwort enthält, i.e., wenn $\varepsilon \in L$. Ein Gegenbeispiel wäre $L = \{\texttt{a}\}$.

(c) Diese Gleichung gilt in der Regel nicht. Gegenbeispiel: $L = \{\texttt{a}\}$; wir erhalten $(\{\texttt{a}\} \cdot \{\texttt{a}\})^* = \{\texttt{aa}\}^* \neq \{\texttt{a}\}^* = \{\texttt{a}\}^* \cdot \{\texttt{a}\}^*$.

(d) Diese Gleichung wird von keiner Sprache erfüllt. Die linke Seite vereinfacht sich zu $L^+ \cup \{\varepsilon\} = L^*$; diese Sprache enthält offenbar das Leerwort. Die rechte Seite vereinfacht sich zu $L^* \cdot \{\} = \{\}$; diese Sprache enthält gar kein Wort, insbesondere nicht das Leerwort.

Aufgabe 9 (0.3 Punkte)

Geben Sie einen regulären Ausdruck an, der alle E-Mail-Adressen beschreibt, die die folgenden Bedingungen erfüllen:

- Sie enden auf *@logic.at* .
- Vor dem @-Zeichen steht mindestens ein Buchstabe oder eine Ziffer.
- Der Teil links des @-Zeichens besteht nur aus Buchstaben und Ziffern.
- Das erste Zeichen der E-Mail-Adresse ist ein Buchstabe.

(a) Geben Sie den gesuchten regulären Ausdruck in algebraischer Notation an.

(b) Geben Sie den gesuchten regulären Ausdruck in `egrep`-Notation an. (Gesucht sind alle Zeilen, die *ausschließlich* eine E-Mail-Adresse enthalten.)

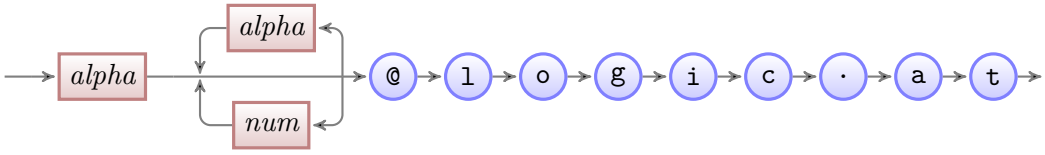
(c) Zeichnen Sie das Syntaxdiagramm, das Ihrem regulären Ausdruck aus Teil a entspricht.

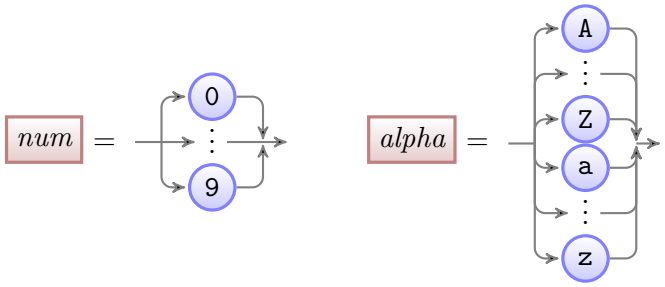
Lösung

(a) $alpha := \texttt{A} + \cdots + \texttt{Z} + \texttt{a} + \cdots + \texttt{z}$
$num := \texttt{0} + \cdots + \texttt{9}$
Regulärer Ausdruck: $alpha(alpha + num)^*\texttt{@logic.at}$

(b) `^[a-zA-Z][a-zA-Z0-9]*@logic\.at$` oder
`^[:alpha:][:alnum:]*@logic\.at$`

(c) Syntaxdiagramm zu $alpha(alpha + num)^*\texttt{@logic.at}$:



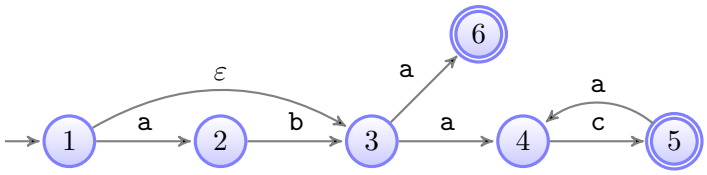


Aufgabe 10 (0.3 Punkte)

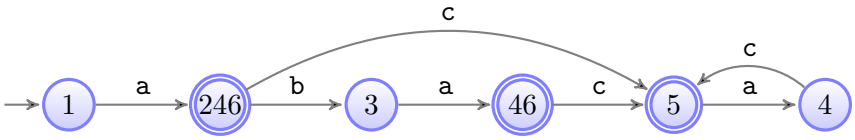
Konstruieren Sie einen endlichen Automaten, der dieselbe Sprache beschreibt wie der reguläre Ausdruck $(\texttt{ab} + \varepsilon)((\texttt{ac})^+ + \texttt{a})$.

Lösung

Konstruiert man den Automaten nach dem allgemeinen Verfahren aus der Vorlesung, benötigt man etwa 20 Zustände und fast ebensoviele ε-Kanten. Der folgende Automat ist dem regulären Ausdruck noch ähnlich, benötigt aber nur 6 Zustände.



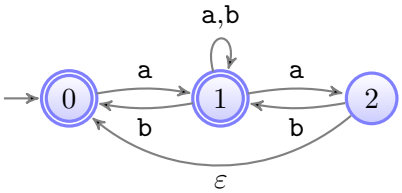
Der Automat ist aufgrund der einen ε-Kante und der beiden a-Kanten bei Zustand 3 indeterministisch. Das Determinisierungsverfahren aus der Vorlesung liefert den folgenden Automaten.



Natürlich gibt es auch noch andere Automaten, die zu diesen beiden äquivalent sind.

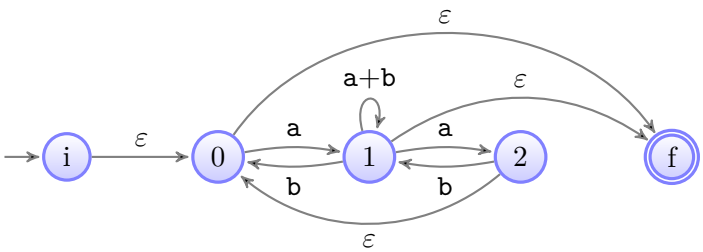
Aufgabe 11 (0.3 Punkte)

Konstruieren Sie zu folgendem endlichen Automaten einen regulären Ausdruck. Orientieren Sie sich am Algorithmus, der in der Vorlesung besprochen wurde und geben Sie den Automaten nach jeder Zustandselimination an!



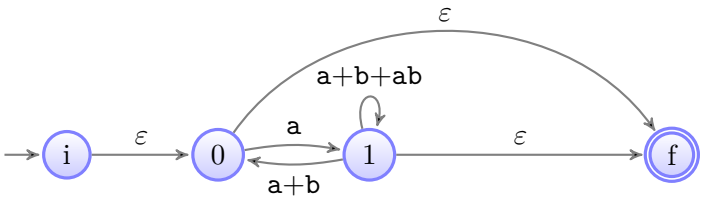
Lösung

Neuer Anfangs- und Endzustand:

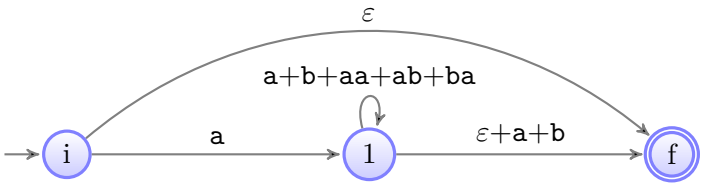


Wir eliminieren die Zustände in der Reihenfolge 2, 0 und 1; andere Reihenfolgen sind ebenfalls möglich.

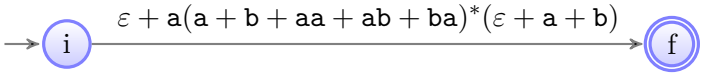
Elimination von Zustand 2:



Elimination von Zustand 0:



Elimination von Zustand 1:



Da $(\mathtt{a}+\mathtt{b})^*$ bereits alle möglichen Wörter über $\{\mathtt{a},\mathtt{b}\}$ enthält, gilt $(\mathtt{a}+\mathtt{b}+\cdots)^* = (\mathtt{a}+\mathtt{b})^*$, der reguläre Ausdruck vereinfacht sich damit zu $\varepsilon + \mathtt{a}(\mathtt{a}+\mathtt{b})^*(\varepsilon+\mathtt{a}+\mathtt{b})$. Mit demselben Argument gilt

$$(\mathtt{a}+\mathtt{b})^*(\varepsilon+\mathtt{a}+\mathtt{b}) = (\mathtt{a}+\mathtt{b})^* + (\mathtt{a}+\mathtt{b})^*(\mathtt{a}+\mathtt{b}) = (\mathtt{a}+\mathtt{b})^*$$

Damit erhalten wir als Ergebnis den regulären Ausdruck $\varepsilon + \mathtt{a}(\mathtt{a}+\mathtt{b})^*$. Der Automat akzeptiert somit alle Wörter über $\{\mathtt{a},\mathtt{b}\}$ außer jenen, die mit dem Symbol $\mathtt{b}$ beginnen.

Aufgabe 12 (0.3 Punkte)

Songtexte enthalten gelegentlich Füllworte wie „Schubbidubbidu“ oder „Schubbidubbiduahhhhh“. Die Grammatik $G = \langle V, T, P, A\rangle$ erzeugt solche Füllwörter, wobei

$$\begin{aligned} V &= \{A, B, C, D\} \\ T &= \{\mathtt{a},\mathtt{b},\mathtt{c},\mathtt{d},\mathtt{h},\mathtt{i},\mathtt{s},\mathtt{u}\} \\ P &= \{\, A \to \mathtt{schub}\,B \ , \\ &\qquad B \to \mathtt{bi}\,C \mid \mathtt{schub}\,B \ , \\ &\qquad C \to \mathtt{dub}\,B \mid D \mid B \ , \\ &\qquad D \to \mathtt{du}\,D \mid \mathtt{ahhhhh} \mid \varepsilon \ \} \end{aligned}$$

(a) Überprüfen Sie für die nachfolgenden Wörter, ob sie in der von der Grammatik G spezifizierten Sprache $\mathcal{L}(G)$ liegen. Falls ja, geben Sie eine Ableitung an. Falls nein, argumentieren Sie, warum nicht.

(1) `schubschubbiduahhhhh`

(2) `schwabschwabbidubbidudu`

(3) `schubbidubschubbischubschubbidubbiduduahhhhh`

(b) Sind folgende Aussagen über die Sprache $\mathcal{L}(G)$ korrekt? Wenn ja, warum? Wenn nein, geben Sie ein Gegenbeispiel!

(1) In jedem Wort der Sprache ist die Anzahl der `schub`-Silben ungerade.

(2) Jedes Wort enthält mindestens ein `bi`.

(3) Man kann beliebig lange Wörter bilden, in denen keine zwei gleichen Silben (`schub`, `bi`, `dub`, `du` oder `ahhhhh`) aufeinander folgen.

(4) In jedem Wort ist die Anzahl der `dub` kleiner als die Anzahl der `bi`.

(c) Geben Sie einen endlichen Automaten für die Sprache $\mathcal{L}(G)$ an.

Lösung

(a) (1) Ja, das Wort liegt in der Sprache $\mathcal{L}(G)$:

$$\begin{aligned} A &\Rightarrow \texttt{schub}\, B \\ &\Rightarrow \texttt{schub schub}\, B \\ &\Rightarrow \texttt{schub schub bi}\, C \\ &\Rightarrow \texttt{schub schub bi}\, D \\ &\Rightarrow \texttt{schub schub bi du}\, D \\ &\Rightarrow \texttt{schub schub bi du ahhhhh} \end{aligned}$$

(2) Das Wort ist nicht Teil der Sprache $\mathcal{L}(G)$, da jedes Wort mit `schub` beginnen muss.

(3) Ja, das Wort liegt in der Sprache $\mathcal{L}(G)$:

$$\begin{aligned} A &\Rightarrow \texttt{schub}\, B \\ &\Rightarrow \texttt{schub bi}\, C \\ &\Rightarrow \texttt{schub bi dub}\, B \\ &\Rightarrow \texttt{schub bi dub schub}\, B \\ &\Rightarrow \texttt{schub bi dub schub bi}\, C \\ &\Rightarrow \texttt{schub bi dub schub bi}\, B \\ &\Rightarrow \texttt{schub bi dub schub bi schub}\, B \\ &\Rightarrow \texttt{schub bi dub schub bi schub schub}\, B \\ &\Rightarrow \texttt{schub bi dub schub bi schub schub bi}\, C \\ &\Rightarrow \texttt{schub bi dub schub bi schub schub bi dub}\, B \\ &\Rightarrow \texttt{schub bi dub schub bi schub schub bi dub bi}\, C \\ &\Rightarrow \texttt{schub bi dub schub bi schub schub bi dub bi}\, D \\ &\Rightarrow \texttt{schub bi dub schub bi schub schub bi dub bi du}\, D \\ &\Rightarrow \texttt{schub bi dub schub bi schub schub bi dub bi du du}\, D \\ &\Rightarrow \texttt{schub bi dub schub bi schub schub bi dub bi du du ahhhhh} \end{aligned}$$

(b) (1) Falsch. Z.B. ist `schubschubbi` ein Wort der Sprache $\mathcal{L}(G)$ (siehe Ableitung unten) mit einer geraden Anzahl an `schub`-Silben.

$$\begin{aligned} A &\Rightarrow \texttt{schub}\, B \\ &\Rightarrow \texttt{schub schub}\, B \\ &\Rightarrow \texttt{schub schub bi}\, C \\ &\Rightarrow \texttt{schub schub bi}\, D \\ &\Rightarrow \texttt{schub schub bi} \end{aligned}$$

(2) Richtig. Jede Ableitung beginnt mit

$$A \Rightarrow \texttt{schub}\, B \Rightarrow \cdots \Rightarrow (\texttt{schub})^n\, B \Rightarrow (\texttt{schub})^n\, \texttt{bi}\, C$$

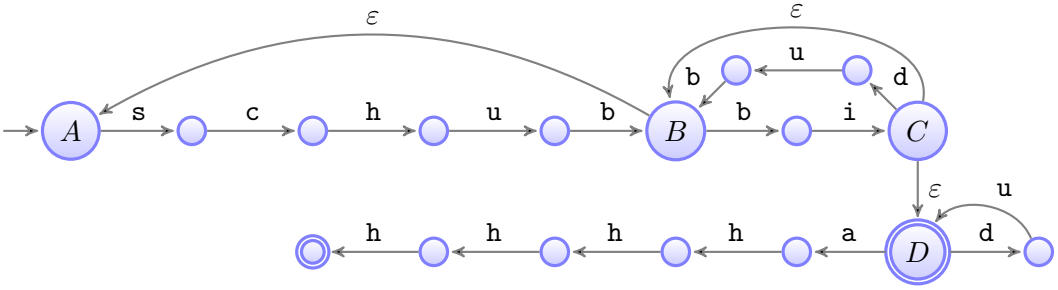
für $n \geq 1$. Somit enthält jedes ableitbare Wort die Silbe `bi`.

(3) Richtig. Z.B. ist das Wort $(\texttt{schub bi})^n$ für jedes $n \geq 1$ Teil der Sprache $\mathcal{L}(G)$:

$$\begin{aligned} A &\Rightarrow \texttt{schub}\, B \Rightarrow \texttt{schub bi}\, C \\ &\Rightarrow \texttt{schub bi}\, B \Rightarrow \texttt{schub bi schub}\, B \Rightarrow \texttt{schub bi schub bi}\, C \\ &\Rightarrow \cdots \Rightarrow (\texttt{schub bi})^n\, C \\ &\Rightarrow (\texttt{schub bi})^n\, D \Rightarrow (\texttt{schub bi})^n\, \varepsilon = (\texttt{schub bi})^n \end{aligned}$$

(4) Richtig. Enthält ein Wort der Sprache die Silbe `dub` n Mal, dann kommt die Silbe `bi` mindestens $n + 1$ Mal vor. Das lässt sich folgendermaßen sehen. Die Silbe `dub` kann nur durch die Produktion $C \to \texttt{dub}\, B$ in ein Wort gelangen. Das Nonterminal C wiederum kann nur durch die Produktion $B \to \texttt{bi}\, C$ entstehen. Somit steht vor `dub` immer die Silbe `bi`. Weiters kann jedes Wort nur über D-Produktionen beendet werden. D kann aber nur über die Ableitungsschritte $w\, B \Rightarrow w\, \texttt{bi}\, C \Rightarrow w\, \texttt{bi}\, D$ erreicht werden, d.h., nach dem letzten `dub` kommt mindestens noch ein weiteres `bi`.

(c) Da das Alphabet laut Angabe aus einzelnen Buchstaben besteht und Übergänge nur für einzelne Symbole definiert sind, müssen wir die Silben mit Hilfe von zusätzlichen Zuständen in einzelne Buchstaben zerlegen.



Aufgabe 13 (0.3 Punkte)

Sei $\mathcal{M}$ folgende Teilmenge der Anweisungen der Programmiersprache Modula.

- Zuweisungen haben die Form b `:=` a
  b ist ein Bezeichner, der mit einem Buchstaben beginnt und auf den beliebig viele Buchstaben und Ziffern folgen können. a steht für einen Ausdruck.

- Blöcke besitzen die Form `BEGIN` m_1`;` $\ldots$`;` m_n `END`
  Dabei können $m_1, \ldots, m_n$ $(n \geq 1)$ beliebige Anweisungen aus $\mathcal{M}$ sein, die durch einen Strichpunkt getrennt werden.

- Konditionale können folgende Formen annehmen:

  IF a_1 THEN f_1 END
  IF a_1 THEN f_1 ELSE f_2 END
  IF a_1 THEN f_1 ELSIF a_2 THEN f_2 END
  IF a_1 THEN f_1 ELSIF a_2 THEN f_2 ELSE f_3 END
  IF a_1 THEN f_1 ELSIF a_2 THEN f_2 ELSIF a_2 THEN f_3 END
  IF a_1 THEN f_1 ELSIF a_2 THEN f_2 ELSIF a_2 THEN f_3 ELSE f_4 END
  $\vdots$

  Das heißt, dem If-Teil folgt immer ein Then-Teil, dann kommt eine beliebige Zahl von Elsif-Teilen, und zuletzt kann optional ein Else-Teil folgen. a_i steht dabei für Ausdrücke, f_i für Anweisungsfolgen der Form m_1 ; … ; m_n, wie sie auch bei Blöcken vorkommen.

- Exit-Anweisungen bestehen nur aus dem Schlüsselwort `EXIT`.

- Schleifen sehen genauso aus wie Blöcke, außer dass `BEGIN` durch das Schlüsselwort `LOOP` ersetzt ist.

Beispiel eines Programms in $\mathcal{M}$:

```
LOOP
  X1 := a1;
  IF a2 THEN EXIT
  ELSIF a3 THEN X2 := a4
  END
END
```

Beschreiben Sie die Sprache $\mathcal{M}$ aller derartigen Anweisungen mit Hilfe einer kontextfreien Grammatik. Verwenden Sie so weit wie möglich EBNF-Notationen, um die Grammatik übersichtlich zu halten und rekursive Regeln zu vermeiden.
Nehmen Sie an, dass es bereits Produktionen gibt, die es ermöglichen, aus dem Nonterminal *Ausdruck* die zulässigen Ausdrücke (wie a_1, a_2, …) zu erzeugen. Es ist nicht notwendig, Leerzeichen und ähnliches (*white space*) zu berücksichtigen.

Lösung

Die Sprache $\mathcal{M}$ wird durch die Grammatik $\langle V, T, P, \mathit{Anweisung}\rangle$ erzeugt, wobei V, T und P die folgenden Mengen sind.

$$
\begin{aligned}
V &= \{\mathit{Anweisung}, \mathit{Zuweisung}, \mathit{Block}, \mathit{Konditional}, \mathit{If}, \mathit{Elsif}, \mathit{Else}, \mathit{Exit}, \mathit{Loop}, \\
&\quad \mathit{Anweisungsfolge}, \mathit{Bezeichner}, \mathit{Buchst}, \mathit{Ziff}, \mathit{Ausdruck}\}\ , \\
T &= \{\texttt{":="}, \texttt{"BEGIN"}, \texttt{"END"}, \texttt{"IF"}, \texttt{"THEN"}, \texttt{"ELSIF"}, \texttt{"ELSE"}, \texttt{"EXIT"}, \texttt{"LOOP"}, \\
&\quad \texttt{";"}, \texttt{"A"}, \ldots, \texttt{"Z"}, \texttt{"a"}, \ldots, \texttt{"z"}, \texttt{"0"}, \ldots, \texttt{"9"}\}\ , \\
P &= \{\mathit{Anweisung} \rightarrow \mathit{Zuweisung} \mid \mathit{Block} \mid \mathit{Konditional} \mid \mathit{Exit} \mid \mathit{Loop}\ , \\
&\quad \mathit{Zuweisung} \rightarrow \mathit{Bezeichner}\ \texttt{":="}\ \mathit{Ausdruck}\ , \\
&\quad \mathit{Block} \rightarrow \texttt{"BEGIN"}\ \mathit{Anweisungsfolge}\ \texttt{"END"}\ , \\
&\quad \mathit{Konditional} \rightarrow \mathit{If}\ \{\ \mathit{Elsif}\ \}\ [\ \mathit{Else}\ ]\ \texttt{"END"}\ , \\
&\quad \mathit{If} \rightarrow \texttt{"IF"}\ \mathit{Ausdruck}\ \texttt{"THEN"}\ \mathit{Anweisungsfolge}\ , \\
&\quad \mathit{Elsif} \rightarrow \texttt{"ELSIF"}\ \mathit{Ausdruck}\ \texttt{"THEN"}\ \mathit{Anweisungsfolge}\ , \\
&\quad \mathit{Else} \rightarrow \texttt{"ELSE"}\ \mathit{Anweisungsfolge}\ , \\
&\quad \mathit{Exit} \rightarrow \texttt{"EXIT"}\ , \\
&\quad \mathit{Loop} \rightarrow \texttt{"LOOP"}\ \mathit{Anweisungsfolge}\ \texttt{"END"}\ , \\
&\quad \mathit{Anweisungsfolge} \rightarrow \mathit{Anweisung}\ \{\ \texttt{";"}\ \mathit{Anweisung}\ \}\ , \\
&\quad \mathit{Bezeichner} \rightarrow \mathit{Buchstabe}\ \{\ \mathit{Buchstabe} \mid \mathit{Ziffer}\ \}\ , \\
&\quad \mathit{Buchstabe} \rightarrow \texttt{"A"} \mid \cdots \mid \texttt{"Z"} \mid \texttt{"a"} \mid \cdots \mid \texttt{"z"}\ , \\
&\quad \mathit{Ziffer} \rightarrow \texttt{"0"} \mid \cdots \mid \texttt{"9"}\ , \\
&\quad \mathit{Ausdruck} \rightarrow \cdots\ \}\ .
\end{aligned}
$$

Aufgabe 14 (0.3 Punkte)

Gegeben seien die folgenden Aussagen. Drücken Sie diese Aussagen als prädikatenlogische Formeln aus. Bestimmen Sie dabei Ihre Prädikate selbst und geben Sie diese an!

(a) Alle rationalen Zahlen sind reelle Zahlen.

(b) Nicht alle reellen Zahlen sind rationale Zahlen.

(c) Manche reellen Zahlen sind keine rationalen Zahlen.

(d) Jede natürliche Zahl ist entweder gerade oder ungerade.

(e) Keine natürliche Zahl ist sowohl gerade als auch ungerade.

Lösung

Wir benutzen die folgenden Prädikate:

$Q(x)$: x ist eine rationale Zahl.
$R(x)$: x ist eine reelle Zahl.
$N(x)$: x ist eine natürliche Zahl.
$G(x)$: x ist eine gerade Zahl.
$U(x)$: x ist eine ungerade Zahl.

(a) $\forall x\,(Q(x) \supset R(x))$

(b) $\neg\forall x\,(R(x) \supset Q(x))$

(c) $\exists x\,(R(x) \wedge \neg Q(x))$

(d) $\forall x\,(N(x) \supset (G(x) \vee U(x)))$ oder $\forall x\,(N(x) \supset (G(x) \not\equiv U(x)))$

(e) $\neg\exists x\,(N(x) \wedge G(x) \wedge U(x))$

Aufgabe 15 (0.3 Punkte)

Seien *Verprügelt*, *Angriffslustig*, *Gallier* und *Römer* Prädikatensymbole und *obelix*, *lacmus* und *cäsar* Konstantensymbole mit folgender Bedeutung:

$Verprügelt(x, y)$	... x verprügelt y	*obelix*	... Obelix
$Angriffslustig(x)$	... x ist angriffslustig	*lacmus*	... Lacmus
$Gallier(x)$	... x ist ein Gallier	*cäsar*	... Cäsar
$Römer(x)$	... x ist ein Römer		

Verwenden Sie diese Symbole, um die beiden nachfolgenden Sätze in prädikatenlogische Formeln zu übersetzen.

(a) Alle Römer werden von Obelix aber nicht von Cäsar verprügelt.

(b) Manche Gallier verprügeln alle angriffslustigen Römer.

Sei weiters folgende Interpretation gegeben:

$$\begin{aligned}
\mathcal{U} &= \{\text{Aerobus, Asterix, Bonus, Cäsar, Lacmus, Obelix, Troubadix, Verleihnix}\}\\
I(Gallier) &= \{\text{Asterix, Obelix, Verleihnix}\}\\
I(Römer) &= \{\text{Aerobus, Bonus, Cäsar}\}\\
I(Verprügelt) &= \{(\text{Asterix, Aerobus}), (\text{Asterix, Lacmus}), (\text{Obelix, Aerobus}), (\text{Obelix, Bonus}),\\
&\qquad (\text{Obelix, Cäsar}), (\text{Obelix, Lacmus}), (\text{Troubadix, Bonus}), (\text{Verleihnix, Lacmus})\}\\
I(cäsar) &= \text{Cäsar}\\
I(lacmus) &= \text{Lacmus}\\
I(obelix) &= \text{Obelix}
\end{aligned}$$

Geben Sie an, ob die nachfolgenden Formeln in dieser Interpretation wahr oder falsch sind. Begründen Sie Ihre Antwort mit einem konkreten Beispiel; es ist keine formale Auswertung erforderlich.

(c) $\forall x\, Verprügelt(x, lacmus)$

(d) $\exists x \forall y(Römer(y) \supset Verprügelt(x, y))$

(e) $\exists x \forall y(Römer(x) \wedge (Gallier(y) \supset Verprügelt(y, x)))$

Bestimmen Sie unter Verwendung der Evaluierungsfunktion den Wahrheitswert der Formel

(f) $\forall x(\mathit{Gallier}(x) \supset \mathit{Verprügelt}(x, \mathit{lacmus}))$

Lösung

(a) $\forall x(\mathit{Römer}(x) \supset (\mathit{Verprügelt}(\mathit{obelix}, x) \wedge \neg \mathit{Verprügelt}(\mathit{cäsar}, x)))$

(b) $\exists x \forall y(\mathit{Gallier}(x) \wedge ((\mathit{Angriffslustig}(y) \wedge \mathit{Römer}(y)) \supset \mathit{Verprügelt}(x, y)))$ oder
$\exists x(\mathit{Gallier}(x) \wedge \forall y((\mathit{Angriffslustig}(y) \wedge \mathit{Römer}(y)) \supset \mathit{Verprügelt}(x, y)))$

(c) Falsch, da z.B. $(\text{Aerobus}, \text{Lacmus}) \notin I(\mathit{Verprügelt})$

(d) Wahr, da Obelix alle Römer verprügelt: Für jeden der drei Römer $r \in I(\mathit{Römer})$ gibt es ein entsprechendes Paar (Obelix, r) in $I(\mathit{Verprügelt})$.

(e) Falsch, da keiner der drei Römer Aerobus, Bonus und Cäsar von jedem der drei Gallier Asterix, Obelix und Verleihnix verprügelt wird. Lacmus ist kein Beleg für die Wahrheit der Formel: Er wird zwar von allen Galliern verprügelt, er ist aber kein Römer in dieser Interpretation.

(f) $\mathsf{val}_{I,\sigma}(\forall x(\mathit{Gallier}(x) \supset \mathit{Verprügelt}(x, \mathit{lacmus}))) = 1$
$\iff$ Für alle $\sigma' \overset{x}{\sim} \sigma$ gilt: $\mathsf{val}_{I,\sigma'}(\mathit{Gallier}(x) \supset \mathit{Verprügelt}(x, \mathit{lacmus})) = 1$
$\iff$ Für alle $\sigma' \overset{x}{\sim} \sigma$ gilt:
Wenn $\mathsf{val}_{I,\sigma'}(\mathit{Gallier}(x)) = 1$, dann $\mathsf{val}_{I,\sigma'}(\mathit{Verprügelt}(x, \mathit{lacmus}) = 1$.
$\iff$ Für alle $\sigma' \overset{x}{\sim} \sigma$ gilt:
Wenn $\sigma'(x) \in I(\mathit{Gallier})$, dann $(\sigma'(x), \text{Lacmus}) \in I(\mathit{Verprügelt})$.

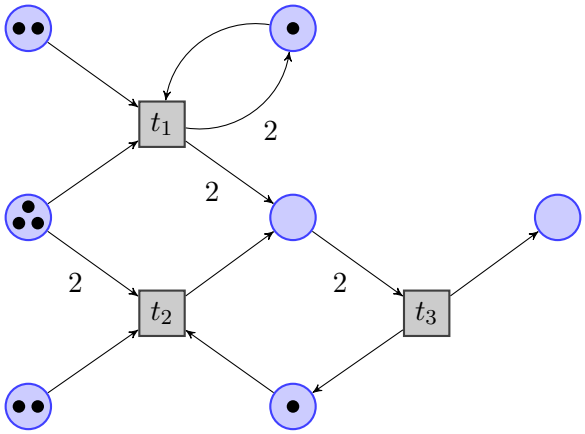
Wegen $I(\mathit{Gallier}) = \{\text{Asterix}, \text{Obelix}, \text{Verleihnix}\}$ müssen wir drei Möglichkeiten für $\sigma'(x)$ untersuchen.

$\sigma'(x)$	$(\sigma'(x), \text{Lacmus}) \in I(\mathit{Verprügelt})$?	
Asterix	$(\text{Asterix}, \text{Lacmus}) \in I(\mathit{Verprügelt})$	✓
Obelix	$(\text{Obelix}, \text{Lacmus}) \in I(\mathit{Verprügelt})$	✓
Verleihnix	$(\text{Verleihnix}, \text{Lacmus}) \in I(\mathit{Verprügelt})$	✓

Die Formel ist daher wahr in der gegebenen Interpretation.

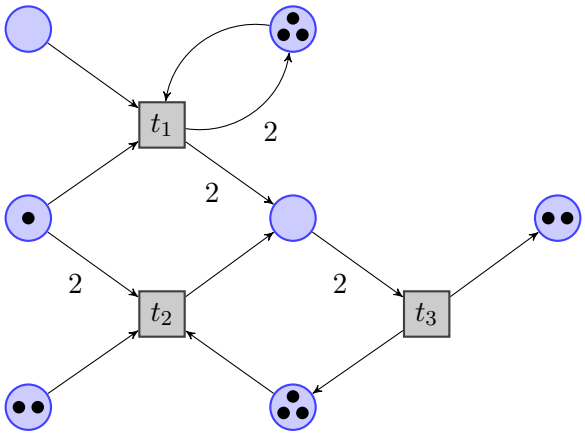
Aufgabe 16 (0.3 Punkte)

Gegeben sei das folgende Petri-Netz mit Anfangsmarkierung. Geben Sie alle möglichen Reihenfolgen an, in denen die Transitionen feuern können. Geben Sie jene erreichbaren Markierungen an, in denen keine Transition aktiviert ist.

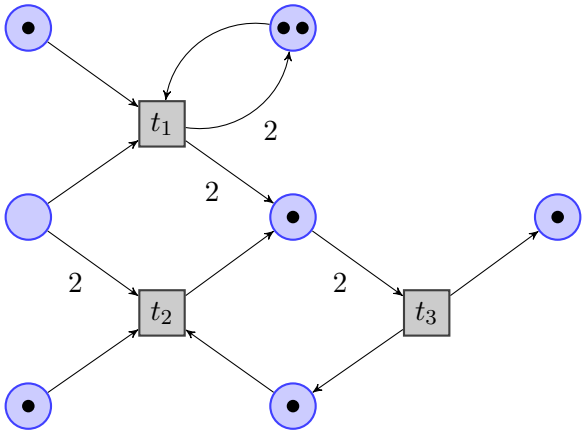


Lösung

Die Transitionsfolgen t_1–t_1–t_3–t_3 und t_1–t_3–t_1–t_3 liefern:



Die Transitionsfolgen t_1–t_2–t_3, t_1–t_3–t_2 und t_2–t_1–t_3 liefern:

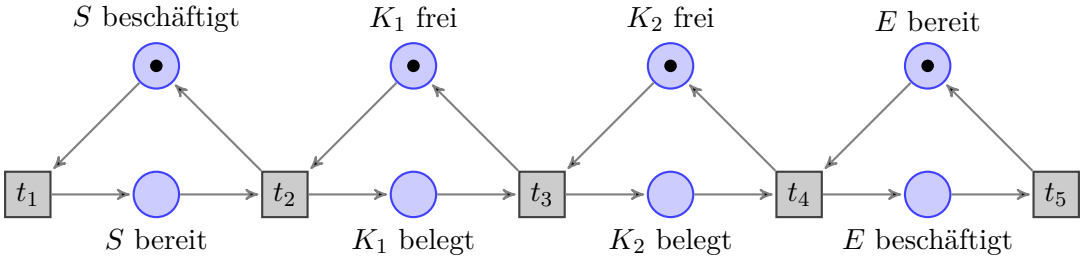


Aufgabe 17 (0.3 Punkte)

Ein Sender und ein Empfänger kommunizieren miteinander über einen Kanal mit Kapazität 2, das heißt, der Kanal ist ein FIFO-Buffer (First-In-First-Out, Queue) mit zwei Plätzen. Wenn der Sender eine Nachricht sendet, belegt diese Platz 1. Nachrichten in Platz 1 werden auf Platz 2 verschoben. Der Empfänger empfängt schließlich die Nachricht, die sich auf Platz 2 befindet.
Der Sender und der Empfänger können sich entweder im Zustand *belegt* oder *bereit* befinden. Der Sender kann nur senden, wenn er sendebereit ist; der Empfänger kann nur empfangen, wenn er empfangsbereit ist und Platz 2 des Kanals eine Nachricht enthält. Nach einer erfolgreichen Sendeoperation (bzw. Empfangsoperation) geht der Sender (bzw.der Empfänger) in den Zustand *belegt* und wechselt nach einer kurzen Zeit wieder in den Zustand *bereit*.

Modellieren Sie das beschriebene System mit Hilfe eines Petri-Netzes.

Lösung



Die beiden Stellen links modellieren den Sender. „S beschäftigt“ bedeutet, dass der Sender die Nachricht erstellt, wohingegen eine Markierung bei „S bereit“ anzeigt, dass die Nachricht sendebereit ist. Die Bereitschaft der Nachricht kann von weiteren Bedingungen abhängig gemacht werden, indem weitere Stellen mit t_1 verbunden werden.
Jeder der beiden Kanal-Plätze wird durch zwei Stellen modelliert. Eine Markierung auf der Stelle „K_i frei“ bzw. „K_i belegt“ zeigt an, dass der Platz i frei bzw. belegt ist.
Analog zeigt eine Markierung bei „E bereit“ bzw. „E beschäftigt“ an, dass der Empfänger bereit zum Nachrichtenempfang bzw. beschäftigt mit der Verarbeitung der letzten Nachricht ist. Die Ankunft einer Nachricht kann weitere Aktionen auslösen, indem t_5 mit weiteren Stellen verbunden wird.
Durch die Art der Verbindungen und die gewählten Markierungen ist sichergestellt, dass eine Nachricht

- ... nur gesendet werden kann, wenn K_1 frei ist;
- ... nur dann von K_1 nach K_2 gelangt, wenn K_2 frei ist;
- ... nur empfangen werden kann, wenn der Empfänger bereit ist.