2. Übungsblatt (mit Lösungen)

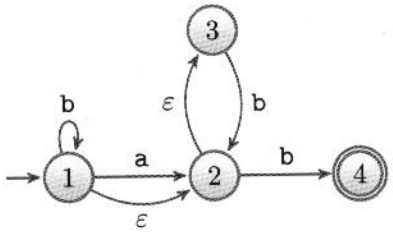
3.0 VU Formale Modellierung

Marion Brandsteidl, Gernot Salzer

5. Jänner 2014

Aufgabe 1 (0.3 Punkte)

Sei $\mathcal{A}$ der folgende Automat:



Konstruieren Sie mit Hilfe des in der Vorlesung besprochenen Determinisierungsverfahrens einen deterministischen Automaten $\mathcal{A}'$, der äquivalent zu $\mathcal{A}$ ist.
Hinweis: Beginnen Sie mit einer Tabelle für die Werte $\delta^*(q, s)$, wobei $q \in \{1, 2, 3, 4\}$ und $s \in \{\mathtt{a}, \mathtt{b}\}$ gilt. Beachten Sie, dass man etwa vom Zustand 3 mit dem Symbol b in einem oder mehreren Schritten nicht nur in den Zustand 2, sondern auch zurück in den Zustand 3 gelangen kann, d.h., $\delta^*(3, \mathtt{b}) = \{2, 3\}$.

Lösung

Wir konstruieren als Zwischenschritt eine Tabelle mit den Werten von $\delta^*(q, s)$ für alle Zustände q und alle Symbole s. ε-Übergänge werden dadurch berücksichtigt, dass vor und nach einem Symbol beliebig viele Leerwörter auftreten können. Etwa sind vom Zustand 1 aus alle Zustände mit dem Symbol b erreichbar (d.h., $\delta^*(1, \mathtt{b}) = \{1, 2, 3, 4\}$), obwohl der einzige b-Übergang bei 1 nur nach 1 führt:

$1 \xrightarrow{\mathtt{b}} 1$

$1 \xrightarrow{\mathtt{b}} 1 \xrightarrow{\varepsilon} 2$ oder $1 \xrightarrow{\varepsilon} 2 \xrightarrow{\varepsilon} 3 \xrightarrow{\mathtt{b}} 2$

$1 \xrightarrow{\mathtt{b}} 1 \xrightarrow{\varepsilon} 2 \xrightarrow{\varepsilon} 3$ oder $1 \xrightarrow{\varepsilon} 2 \xrightarrow{\varepsilon} 3 \xrightarrow{\mathtt{b}} 2 \xrightarrow{\varepsilon} 3$

$1 \xrightarrow{\varepsilon} 2 \xrightarrow{\mathtt{b}} 4$

Aus der Tabelle für $\delta^*(q,s)$ ergibt sich jene für die Übergangsfunktion des deterministischen Automaten, $\hat{\delta}$, durch Vereinigung der entsprechenden Zeilen.

$\delta^*(q,s)$	a	b
1	$\{2,3\}$	$\{1,2,3,4\}$
2	$\{\}$	$\{2,3,4\}$
3	$\{\}$	$\{2,3\}$
4	$\{\}$	$\{\}$

$\hat{\delta}$	a	b
$\{1\}$	$\{2,3\}$	$\{1,2,3,4\}$
$\{2,3\}$	$\{\}$	$\{2,3,4\}$
$\{1,2,3,4\}$	$\{2,3\}$	$\{1,2,3,4\}$
$\{\}$	$\{\}$	$\{\}$
$\{2,3,4\}$	$\{\}$	$\{2,3,4\}$

Startzustand des deterministischen Automaten ist $\{1\}$, Endzustände sind alle Zustände, die den ursprünglichen Endzustand 4 enthalten. Der gesuchte deterministische Automat ist somit gegeben durch

$$\mathcal{A}' = \langle \{\{\},\{1\},\{2,3\},\{2,3,4\},\{1,2,3,4\}\},\quad \{\mathtt{a},\mathtt{b}\},\quad \hat{\delta},\quad \{1\},\{\{2,3,4\},\{1,2,3,4\}\}\rangle\ .$$

Aufgabe 2 (0.3 Punkte)

Sei L die Sprache

$$\{\, w \in \{1,2,3,4,5,6,7,8,9\}^* \mid w \text{ enthält die Ziffernfolge } 125,\ 175 \text{ oder } 287 \,\}\ .$$

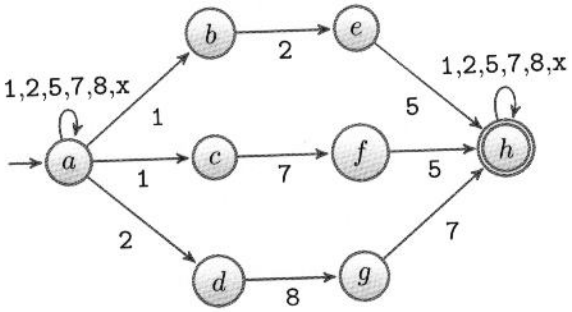
Geben Sie einen *deterministischen* Automaten für L an. Gehen Sie folgendermaßen vor:

1. Konstruieren Sie einen indeterministischen Automaten für diese Sprache.
2. Wandeln Sie den indeterministischen Automaten mit Hilfe des in der Vorlesung besprochenen Verfahrens in einen deterministischen um.

Ist der so gewonnene Automat minimal, d.h., ist er unter den deterministischen Automaten für L jener mit der kleinstmöglichen Zahl von Zuständen?

Lösung

Wir konstruieren zunächst auf möglichst direktem Weg einen beliebigen Automaten für die gesuchte Sprache. Dieser ist im Zustand a indeterministisch: Für das Symbol 1 gibt es drei und für das Symbol 2 zwei mögliche Folgezustände. Zusätzlich zur graphischen Darstellung geben wir die Übergangsfunktion auch als Tabelle an, da diese bei der Determinisierung hilft, wobei x stellvertretend für die Ziffern 3, 4, 6 und 9 steht.



δ^*	1	2	5	7	8	x
a	$\{a,b,c\}$	$\{a,d\}$	$\{a\}$	$\{a\}$	$\{a\}$	$\{a\}$
b	$\{\}$	$\{e\}$	$\{\}$	$\{\}$	$\{\}$	$\{\}$
c	$\{\}$	$\{\}$	$\{\}$	$\{f\}$	$\{\}$	$\{\}$
d	$\{\}$	$\{\}$	$\{\}$	$\{\}$	$\{g\}$	$\{\}$
e	$\{\}$	$\{\}$	$\{h\}$	$\{\}$	$\{\}$	$\{\}$
f	$\{\}$	$\{\}$	$\{h\}$	$\{\}$	$\{\}$	$\{\}$
g	$\{\}$	$\{\}$	$\{\}$	$\{h\}$	$\{\}$	$\{\}$
h	$\{h\}$	$\{h\}$	$\{h\}$	$\{h\}$	$\{h\}$	$\{h\}$

Einen deterministischen Automaten erhalten wir, indem wir den indeterministischen Automaten simulieren. Ein Zustand des deterministischen Automaten repräsentiert dabei jene Zustände des indeterministischen, in denen sich dieser zu diesem Zeitpunkt befinden kann. Der Startzustand wird mit $\{a\}$ bezeichnet, da sich der indeterministische Automat zu Beginn im Zustand a (und nur in diesem) befindet. Von diesem Zustand ausgehend erstellen wir zeilenweise die Tabelle für die Übergangsfunktion des deterministischen Automaten.

δ^*	1	2	5	7	8	x
$\{a\}$	$\{a,b,c\}$	$\{a,d\}$	$\{a\}$	$\{a\}$	$\{a\}$	$\{a\}$
$\{a,b,c\}$	$\{a,b,c\}$	$\{a,d,e\}$	$\{a\}$	$\{a,f\}$	$\{a\}$	$\{a\}$
$\{a,d\}$	$\{a,b,c\}$	$\{a,d\}$	$\{a\}$	$\{a\}$	$\{a,g\}$	$\{a\}$
$\{a,d,e\}$	$\{a,b,c\}$	$\{a,d\}$	$\{a,h\}$	$\{a\}$	$\{a,g\}$	$\{a\}$
$\{a,f\}$	$\{a,b,c\}$	$\{a,d\}$	$\{a,h\}$	$\{a\}$	$\{a\}$	$\{a\}$
$\{a,g\}$	$\{a,b,c\}$	$\{a,d\}$	$\{a\}$	$\{a,h\}$	$\{a\}$	$\{a\}$
$\{a,h\}$	$\{a,b,c,h\}$	$\{a,d,h\}$	$\{a,h\}$	$\{a,h\}$	$\{a,h\}$	$\{a,h\}$
$\{a,b,c,h\}$	$\{a,b,c,h\}$	$\{a,d,e,h\}$	$\{a,h\}$	$\{a,f,h\}$	$\{a,h\}$	$\{a,h\}$
$\{a,d,h\}$	$\{a,b,c,h\}$	$\{a,d,h\}$	$\{a,h\}$	$\{a,h\}$	$\{a,g,h\}$	$\{a,h\}$
$\{a,d,e,h\}$	$\{a,b,c,h\}$	$\{a,d,h\}$	$\{a,h\}$	$\{a,h\}$	$\{a,g,h\}$	$\{a,h\}$
$\{a,f,h\}$	$\{a,b,c,h\}$	$\{a,d,h\}$	$\{a,h\}$	$\{a,h\}$	$\{a,h\}$	$\{a,h\}$
$\{a,g,h\}$	$\{a,b,c,h\}$	$\{a,d,h\}$	$\{a,h\}$	$\{a,h\}$	$\{a,h\}$	$\{a,h\}$

Jene Zustände, die einer Situation entsprechen, in der der indeterministische Automat einen Endzustand erreicht hat, sind die Endzustände des deterministischen Automaten; in diesem Beispiel sind das alle Zustände, deren Bezeichnung h enthält. Dieser wird somit

1-9

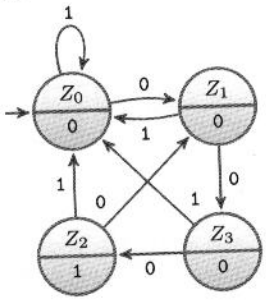
durch das Tupel $\langle \widehat{Q}, \{0,1\}, \hat{\delta}, \{a\}, \widehat{F} \rangle$ beschrieben, wobei

$$\widehat{F} = \{\{a,h\},\{a,d,h\},\{a,f,h\},\{a,g,h\},\{a,b,c,h\},\{a,d,e,h\}\} \quad \text{und}$$
$$\widehat{Q} = \{\{a\},\{a,b,c\},\{a,d\},\{a,g\},\{a,d,e\},\{a,f\}\} \cup \widehat{F} \ .$$

Dieser Automat ist nicht der kleinstmögliche deterministische Automat. In der Vorlesung wurde zwar kein Minimierungsverfahren besprochen, man sieht aber, dass von den Endzuständen mit jeder der Ziffern 1 bis 9 wieder nur Endzustände erreichbar sind. Man kann diese daher zu einem einzigen Endzustand zusammenfassen, ohne die akzeptierte Sprache zu verändern.

Aufgabe 3 (0.3 Punkte)

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



(a) Geben Sie die Ausgaben zu folgenden Eingaben an: 10010, 00011, 10101.

(b) Berechnen Sie für $w = 101000011$ schrittweise $\delta^*(Z_0, w)$ und $\gamma^*(Z_0, w)$.

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

Lösung

(a)

w:	10010	00011	10101
$[\mathcal{A}](w)$:	00000	00100	00000

(b) Übergangsfunktion $\delta^*(Z_0, 101000011)$:

$$\begin{aligned}
\delta^*(Z_0, 101000011) &= \delta^*(\delta(Z_0, 1), 01000011) &&= \delta^*(Z_0, 01000011)\\
&= \delta^*(\delta(Z_0, 0), 1000011) &&= \delta^*(Z_1, 1000011)\\
&= \delta^*(\delta(Z_1, 1), 000011) &&= \delta^*(Z_0, 000011)\\
&= \delta^*(\delta(Z_0, 0), 00011) &&= \delta^*(Z_1, 00011)\\
&= \delta^*(\delta(Z_1, 0), 0011) &&= \delta^*(Z_3, 0011)\\
&= \delta^*(\delta(Z_3, 0), 011) &&= \delta^*(Z_2, 011)\\
&= \delta^*(\delta(Z_2, 0), 11) &&= \delta^*(Z_1, 11)\\
&= \delta^*(\delta(Z_1, 1), 1) &&= \delta^*(Z_0, 1)\\
&= \delta^*(\delta(Z_0, 1), \varepsilon) &&= \delta^*(Z_0, \varepsilon)\\
&= Z_0
\end{aligned}$$

Ausgabefunktion $\gamma^*(Z_0, 101000011)$:

$$\begin{aligned}
\gamma^*(Z_0, 101000011) &= \gamma(\delta(Z_0, 1)) \cdot \gamma^*(\delta(Z_0, 1), 01000011)\\
&= \gamma(Z_0) \cdot \gamma^*(Z_0, 01000011)\\
&= 0 \cdot \gamma(\delta(Z_0, 0)) \cdot \gamma^*(\delta(Z_0, 0), 1000011)\\
&= 0 \cdot \gamma(Z_1) \cdot \gamma^*(Z_1, 1000011)\\
&= 00 \cdot \gamma(\delta(Z_1, 1)) \cdot \gamma^*(\delta(Z_1, 1), 000011)\\
&= 00 \cdot \gamma(Z_0) \cdot \gamma^*(Z_0, 000011)\\
&= 000 \cdot \gamma(\delta(Z_0, 0)) \cdot \gamma^*(\delta(Z_0, 0), 00011)\\
&= 000 \cdot \gamma(Z_1) \cdot \gamma^*(Z_1, 00011)\\
&= 0000 \cdot \gamma(\delta(Z_1, 0)) \cdot \gamma^*(\delta(Z_1, 0), 0011)\\
&= 0000 \cdot \gamma(Z_3) \cdot \gamma^*(Z_3, 0011)\\
&= 00000 \cdot \gamma(\delta(Z_3, 0)) \cdot \gamma^*(\delta(Z_3, 0), 011)\\
&= 00000 \cdot \gamma(Z_2) \cdot \gamma^*(Z_2, 011)\\
&= 000001 \cdot \gamma(\delta(Z_2, 0)) \cdot \gamma^*(\delta(Z_2, 0), 11)\\
&= 000001 \cdot \gamma(Z_1) \cdot \gamma^*(Z_1, 11)\\
&= 0000010 \cdot \gamma(\delta(Z_1, 1)) \cdot \gamma^*(\delta(Z_1, 1), 1)\\
&= 0000010 \cdot \gamma(Z_0) \cdot \gamma^*(Z_0, 1)\\
&= 00000100 \cdot \gamma(\delta(Z_0, 1)) \cdot \gamma^*(\delta(Z_0, 1), \varepsilon)\\
&= 00000100 \cdot \gamma(Z_0) \cdot \gamma^*(Z_0, \varepsilon)\\
&= 000001000 \cdot \gamma^*(Z_0, \varepsilon)\\
&= 000001000 \cdot \varepsilon\\
&= 000001000
\end{aligned}$$

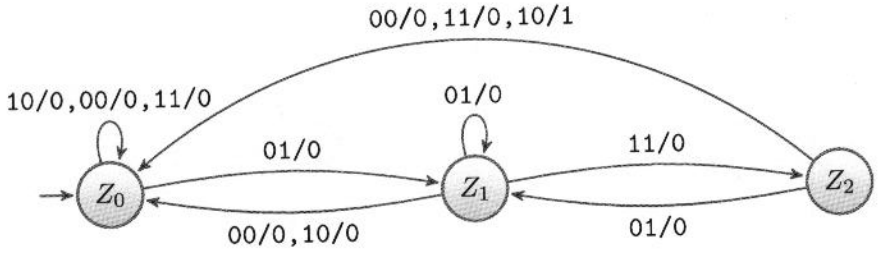
(c) Der Automat ist in 000-Detektor: Nach jedem dritten 0er in Folge wird 1 ausgegeben, dazwischen 0.

Aufgabe 4 (0.3 Punkte)

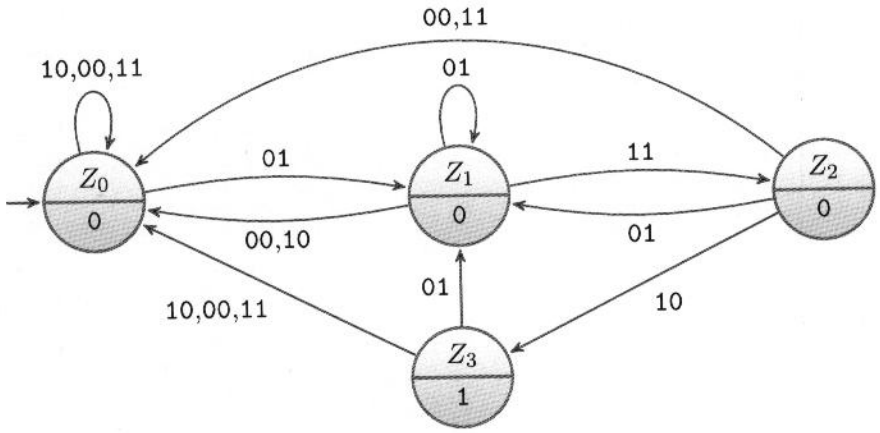
Sei $\Sigma = \{00, 01, 10, 11\}$ das Eingabe- und $\Gamma = \{0, 1\}$ das Ausgabealphabet. Geben Sie sowohl einen Mealy- als auch einen Moore-Automaten an, der 1 ausgibt, wenn er in der Eingabe die Sequenz 01 11 10 erkennt, und 0 sonst. Beispielsweise soll die Eingabe 00 01 11 10 10 zur Ausgabe 00010 führen.

Lösung

Mealy-Automat:



Moore-Automaten:



Aufgabe 5 (0.3 Punkte)

Geben Sie ein Verfahren an, mit dem sich zu jedem indeterministischen endlichen Automaten $\mathcal{A}$ eine äquivalente Grammatik G konstruieren lässt, d.h., es soll $\mathcal{L}(\mathcal{A}) = \mathcal{L}(G)$ gelten. Wenden Sie Ihr Verfahren auf den Automaten aus Aufgabe 1 an.

Lösung

Sei $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$ ein indeterministischer endlicher Automat. Wir führen für jeden Zustand q ein Nonterminal V_q ein und sehen für jeden Übergang $(q, s, q') \in \delta$ eine Produktion $V_q \to s\, V_{q'}$ vor. Jeder Pfad durch den Automaten entspricht dann einer Ableitung, und umgekehrt. Um im Fall eines Endzustandes die Ableitung beenden zu können, müssen wir zusätzlich alle Nonterminale, die Endzuständen entsprechen, durch ε ersetzen können.
Formal lässt sich die dem Automaten $\mathcal{A}$ entsprechende Grammatik beschreiben durch $G_{\mathcal{A}} = \langle V, \Sigma, P, q_0 \rangle$, wobei $V = \{ V_q \mid q \in Q \}$ und die Produktionen definiert sind durch

$$P = \{ V_q \to s\, V_{q'} \mid (q, s, q') \in \delta \} \cup \{ V_q \to \varepsilon \mid q \in F \} \ .$$

Für den Automaten aus Aufgabe 1 erhalten wir die Grammatik

$$\langle \{V_1, V_2, V_3, V_4\}, \{\mathtt{a}, \mathtt{b}\}, P, V_1 \rangle \ ,$$

wobei P die Menge der folgenden Produktionen ist:

$$\begin{aligned} V_1 &\to \mathtt{a}V_2 \mid \mathtt{b}V_1 \mid V_2 \\ V_2 &\to V_3 \mid \mathtt{b}V_4 \\ V_3 &\to \mathtt{b}V_2 \\ V_4 &\to \varepsilon \end{aligned}$$

Aufgabe 6 (0.3 Punkte)

Welche der folgenden Gleichungen sind für alle Sprachen L_1, L_2, L_3 gültig? Begründen Sie Ihre Antwort bzw. geben Sie ein Gegenbeispiel, wenn die Gleichung nicht gültig ist.

(a) $L_1 \cdot L_2 = L_2 \cdot L_1$

(b) $L_1 \cup L_2 = L_2 \cup L_1$

(c) $L_1 \cdot (L_2 \cup L_3) = (L_1 \cdot L_2) \cup (L_1 \cdot L_3)$

(d) $(L_1 \cup L_2)^+ = L_1^+ \cup L_2^+$

Lösung

(a) $L_1 \cdot L_2 = L_2 \cdot L_1$ ist nicht allgemein gültig, da die Konkatenation von Wörtern und daher auch von Sprachen nicht kommutativ ist. Für $L_1 = \{\mathtt{a}\}$ und $L_2 = \{\mathtt{b}\}$ etwa erhalten wir $L_1 \cdot L_2 = \{\mathtt{ab}\} \neq \{\mathtt{ba}\} = L_2 \cdot L_1$.

(b) $L_1 \cup L_2 = L_2 \cup L_1$ ist für alle Sprachen L_1, L_2 gültig, da Mengenvereinigung kommutativ ist.

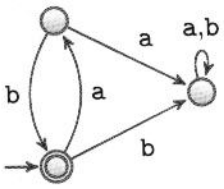
(c) $L_1 \cdot (L_2 \cup L_3) = (L_1 \cdot L_2) \cup (L_1 \cdot L_3)$ gilt für alle Sprachen L_1, L_2, L_3 (Distributivgesetz).

(d) $(L_1 \cup L_2)^+ = L_1^+ \cup L_2^+$ ist nicht allgemein gültig. Für $L_1 = \{\mathtt{0}\}$ und $L_2 = \{\mathtt{1}\}$ etwa erhalten wir $(L_1 \cup L_2)^+ = \{\mathtt{0}, \mathtt{1}\}^+$, was verschieden von $\{\mathtt{0}\}^+ \cup \{\mathtt{1}\}^+$ ist.

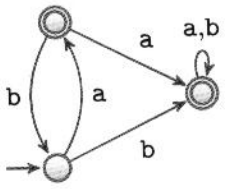
Aufgabe 7 (0.4 Punkte)

Sei $\mathcal{A} = \langle Q, \Sigma, \delta, q_0, F \rangle$ ein beliebiger deterministischer Automat und $L = \mathcal{L}(\mathcal{A})$ die von ihm akzeptierte Sprache. Geben Sie ein Verfahren an, um daraus einen Automaten $\mathcal{A}'$ für das Komplement dieser Sprache zu erhalten; es soll also $\mathcal{L}(\mathcal{A}') = \Sigma^* \setminus L$ gelten.[1] Der Automat $\mathcal{A}'$ akzeptiert somit jene Wörter, die $\mathcal{A}$ nicht akzeptiert, und akzeptiert

[1] Differenz zweier Mengen A und B: $A \setminus B = \{ x \in A \mid x \notin B \}$. Anstelle des Schrägstrichs wird auch das normale Minuszeichen verwendet.



(a) Automat für die Sprache $\{\mathtt{ab}\}^*$



(b) Automat für das Komplement $\Sigma^* \setminus \{\mathtt{ab}\}^*$

Abbildung 1: Beispiel für die Komplementbildung bei Automaten (Aufgabe 7)

jene Wörter nicht, die $\mathcal{A}$ akzeptiert. Welche Eigenschaft regulärer Sprachen ergibt sich daraus? Lässt sich das Verfahren auch auf nicht-deterministische Automaten anwenden? Wenden Sie Ihr Verfahren auf einen selbst gewählten Automaten an und konstruieren Sie den Automaten für die Komplementärsprache.

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

Lösung

In einem deterministischen Automaten ist die Übergangsfunktion δ total, d.h., es gibt zu jedem Zustand q und jedem Symbol s einen eindeutig definierten Folgezustand $\delta(q, s)$. Daher ist auch die erweiterte Übergangsfunktion δ^* total, und der vom Startzustand q_0 mit einem Wort w erreichbare Zustand $q = \delta^*(q_0, w)$ ist eindeutig festgelegt. Der Automat akzeptiert das Wort w genau dann, wenn q ein Endzustand ist. Es genügt daher, den Status der Zustände zu vertauschen und jeden Nicht-Endzustand zu einem Endzustand zu machen und umgekehrt. Ein Automat für das Komplement der Sprache $\mathcal{L}(\mathcal{A})$ lässt sich somit durch $\mathcal{A}' = \langle Q, \Sigma, \delta, q_0, Q \setminus F \rangle$ definieren. Ein Beispiel für diese Konstruktion ist in Abbildung 1 angegeben.
Diese Überlegung zeigt, dass das Komplement jeder durch einen Automaten akzeptierten Sprache wieder durch einen Automaten beschrieben werden kann. Die Familie der Sprachen, die von endlichen Automaten akzeptiert werden, ist daher abgeschlossen gegenüber Komplementbildung. In der Vorlesung haben wir gezeigt, dass diese Sprachfamilie genau die regulären Sprachen sind. Daher sind auch die regulären Sprachen abgeschlossen gegenüber Komplementbildung: Das Komplement einer regulären Sprache ist wieder regulär.
Das angegebene Verfahren lässt sich *nicht* auf indeterministische Automaten anwenden. In der Regel sind mit einem bestimmten Wort vom Startzustand aus mehrere Zustände erreichbar. Wenn mindestens einer dieser Zustände ein Endzustand ist, wird das Wort akzeptiert. Nehmen wir nun an, dass sich unter den erreichbaren Zuständen zusätzlich ein Nicht-Endzustand befindet. Dreht man wie oben beschrieben den Status aller Zustände um, wird aus dem vormaligen Nicht-Endzustand ein Endzustand. Somit wird das Wort immer noch akzeptiert, obwohl es nicht im Komplement der Sprache liegt. Um einen Automat für das Komplement zu finden, muss man zuerst den indeterministischen

Automaten determinisieren (das entsprechende Verfahren wurde nicht in der Vorlesung besprochen, existiert aber) und dann erst die obige Methode anwenden.

Aufgabe 8 (0.3 Punkte)

Die Inventarnummern[2] der TU Wien sind wie in der Abbildung dargestellt aufgebaut:

- Der Inventartyp besteht aus drei Ziffern, einem Bindestrich und zwei weiteren Ziffern.
- Nach einem Schrägstrich folgt zweistellig das Anschaffungsjahr.
- Nach einem weiteren Schrägstrich folgt eine vier- sowie eine zweistellige Identifikationsnummer, die ebenfalls durch einen Schrägstrich getrennt sind.
- Nach einem Leerzeichen folgt die Institutskennung, die aus einem Großbuchstaben und einer dreistelligen Zahl besteht. Die Institutskennung kann auch entfallen.

(a) Geben Sie einen regulären Ausdruck in algebraischer Notation an, der die Menge aller Inventarnummern beschreibt.

(b) Geben Sie einen POSIX Extended Regular Expression an, der alle Zeilen beschreibt, die ausschließlich eine Inventarnummer enthalten.

(c) Geben Sie das Syntaxdiagramm an, das Ihrem regulären Ausdruck aus Teilaufgabe a entspricht.

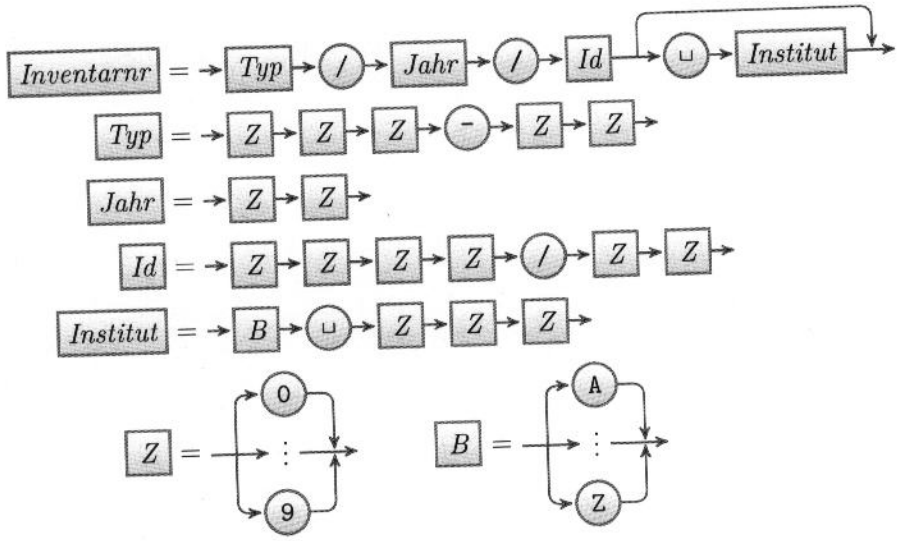
Lösung

(a) $nnn\text{-}nn/nn/nnnn/nn(\varepsilon + \text{␣}l\text{␣}nnn)$ wobei n eine Abkürzung ist für $(0 + \cdots + 9)$ und l eine Abkürzung ist für $(A + \cdots + Z)$.

(b) `^[0-9]{3}-[0-9]{2}/[0-9]{2}/[0-9]{4}/[0-9]{2}(␣[A-Z]␣[0-9]{3})?$`

[2] Allen Anschaffungen ab einem gewissen Wert, etwa Computern und Schränken, wird eine eindeutige Nummer zugewiesen, die sogenannte Inventarnummer. Diese wird auf dem Gegenstand angebracht und dient als Zugriffsschlüssel für die Datenbank, mit der die Anlagegüter verwaltet werden.

(c)

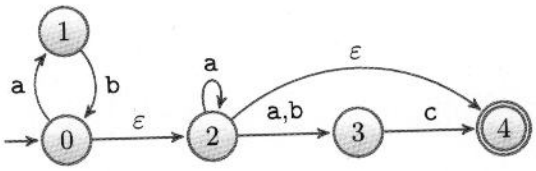


Aufgabe 9 (0.3 Punkte)

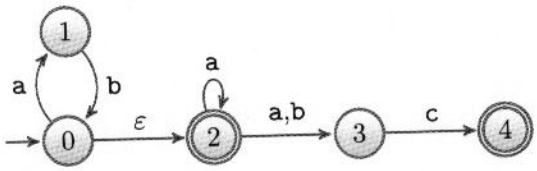
Konstruieren Sie einen endlichen Automaten, der dieselbe Sprache beschreibt wie der reguläre Ausdruck $(\mathtt{ab})^*\mathtt{a}^*(\mathtt{ac} + \mathtt{bc} + \varepsilon)$.

Lösung

Der Automat kann mit dem allgemeinen Verfahren konstruiert werden, enthält dann aber wesentlich mehr Zustände und ε-Kanten als notwendig. Der folgende Automat wurde bereits teilweise vereinfacht.



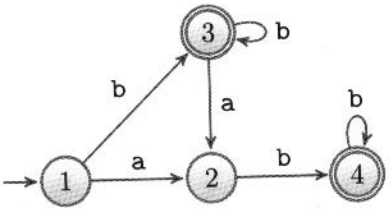
Der zweite ε-Übergang lässt sich einsparen, wenn man einen weiteren Endzustand einführt.



Um einen deterministischen Automaten zu erhalten, sollte man sich aber nicht auf seine Intuition verlassen, sondern den Determinisierungsalgorithmus anwenden.

Aufgabe 10 (0.3 Punkte)

Sei $\mathcal{B}$ der folgende Automat:

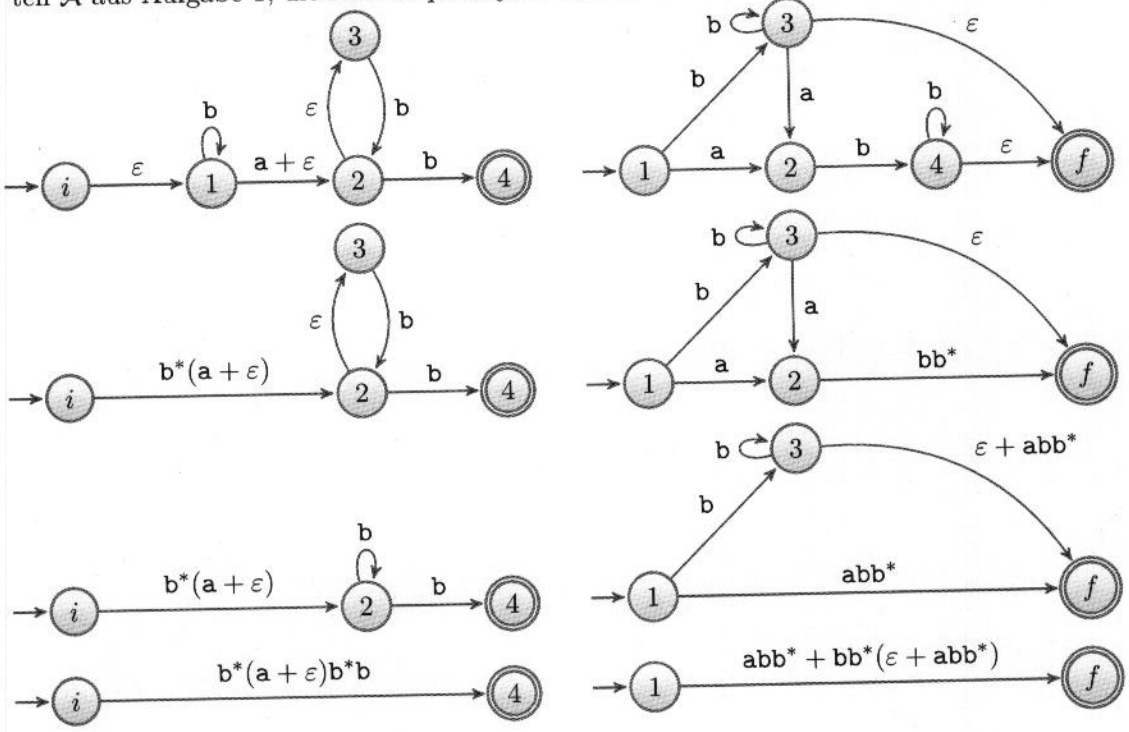


Zeigen Sie, dass $\mathcal{B}$ äquivalent zum Automaten $\mathcal{A}$ aus Aufgabe 1 ist. Gehen Sie folgendermaßen vor:

1. Verwenden Sie das in der Vorlesung besprochene Verfahren, um die von den beiden Automaten akzeptieren Sprachen $\mathcal{L}(\mathcal{A})$ und $\mathcal{L}(\mathcal{B})$ jeweils durch einen regulären Ausdruck zu beschreiben.
2. Zeigen Sie die Äquivalenz der beiden Ausdrücke, indem Sie sie algebraisch ineinander umformen.

Lösung

Wir wandeln beide Automaten in verallgemeinerte endliche Automaten um, indem wir zusätzliche Zustände einführen. Die linke Spalte zeigt die Veränderungen des Automaten $\mathcal{A}$ aus Aufgabe 1, die rechte Spalte jene des Automaten $\mathcal{B}$.



Durch Umformung der beiden Ausdrücke sehen wir, dass die von den beiden Automaten akzeptierten Sprachen identisch sind:

$$\begin{aligned}\mathcal{L}(\mathcal{A}) &= \mathtt{b}^*(\mathtt{a}+\varepsilon)\mathtt{b}^*\mathtt{b} \\ &= \mathtt{b}^*\mathtt{a}\mathtt{b}^*\mathtt{b} + \mathtt{b}^*\mathtt{b}^*\mathtt{b} \\ &= \mathtt{b}^*\mathtt{a}\mathtt{b}^*\mathtt{b} + \mathtt{b}^*\mathtt{b} \quad \text{da } L^* \cdot L^* = L^* \\ &= \mathtt{b}^*\mathtt{a}\mathtt{b}\mathtt{b}^* + \mathtt{b}\mathtt{b}^* \quad \text{da } L^* \cdot L = L \cdot L^*\end{aligned}$$

$$\begin{aligned}\mathcal{L}(\mathcal{B}) &= \mathtt{a}\mathtt{b}\mathtt{b}^* + \mathtt{b}\mathtt{b}^*(\varepsilon + \mathtt{a}\mathtt{b}\mathtt{b}^*) \\ &= \mathtt{a}\mathtt{b}\mathtt{b}^* + \mathtt{b}\mathtt{b}^* + \mathtt{b}\mathtt{b}^*\mathtt{a}\mathtt{b}\mathtt{b}^* \\ &= \mathtt{a}\mathtt{b}\mathtt{b}^* + \mathtt{b}\mathtt{b}^*\mathtt{a}\mathtt{b}\mathtt{b}^* + \mathtt{b}\mathtt{b}^* \\ &= (\varepsilon + \mathtt{b}\mathtt{b}^*)\mathtt{a}\mathtt{b}\mathtt{b}^* + \mathtt{b}\mathtt{b}^* \\ &= \mathtt{b}^*\mathtt{a}\mathtt{b}\mathtt{b}^* + \mathtt{b}\mathtt{b}^* \quad \text{da } \{\varepsilon\} \cup L^+ = L^*\end{aligned}$$

Aufgabe 11 (0.3 Punkte)

Sei G die Grammatik $\langle N, T, P, S\rangle$, wobei gilt:

$$\begin{aligned}N &= \{S, T, U, V, W\} \\ T &= \{\mathtt{a}, \mathtt{e}, \mathtt{h}, \mathtt{H}, \mathtt{i}, \mathtt{l}, \mathtt{o}, \mathtt{ö}\} \\ P &= \{\ S \to \mathtt{Hal}\,T \\ &\qquad T \to \mathtt{li}\,U \mid \mathtt{lö}\,V \mid \mathtt{lo}\,W \\ &\qquad U \to \mathtt{hal}\,T \\ &\qquad V \to \mathtt{lö}\,V \mid \mathtt{le} \\ &\qquad W \to \mathtt{lo}\,T \mid U \mid \varepsilon\}\end{aligned}$$

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

(1) `Hallo`

(2) `Hallilöle`

(3) `Hallololihallihallölöle`

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

(1) Die Anzahl der `lo` ist ungerade.

(2) Jedes Wort enthält mindestens ein `lo` oder `le`.

(3) Man kann beliebig lange Wörter bilden, in denen keine zwei gleichen Silben aufeinander folgen.

(4) In jedem Wort ist die Anzahl der `li` kleiner oder gleich der Anzahl der `lo`.

(5) In jedem Wort ist die Anzahl der `lö` größer oder gleich der Anzahl der `le`.

(c) Konstruieren Sie einen endlichen Automat, der die Sprache $\mathcal{L}(G)$ akzeptiert.

Lösung

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

$$S \Rightarrow_P \texttt{Hal}\,T \Rightarrow_P \texttt{Hal lo}\,W \Rightarrow_P \texttt{Hal lo}\,\varepsilon \Rightarrow_P \texttt{Hal lo}$$

(2) Das Wort `Hallilöle` ist nicht Teil der Sprache $\mathcal{L}(G)$, da jedem `li` ein `hal` folgen muss.

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

$$\begin{aligned}
S &\Rightarrow \texttt{Hal}\,T && \Rightarrow \texttt{Hal lo}\,W \\
&\Rightarrow \texttt{Hal lo lo}\,T && \Rightarrow \texttt{Hal lo lo li}\,U \\
&\Rightarrow \texttt{Hal lo lo li hal}\,T && \Rightarrow \texttt{Hal lo lo li hal li}\,U \\
&\Rightarrow \texttt{Hal lo lo li hal li hal}\,T && \Rightarrow \texttt{Hal lo lo li hal li hal lö}\,V \\
&\Rightarrow \texttt{Hal lo lo li hal li hal lö lö}\,V && \Rightarrow \texttt{Hal lo lo li hal li hal lö lö le}
\end{aligned}$$

(b) (1) Falsch. Z.B. ist `Hallohallo` ein Wort der Sprache mit einer geraden Zahl von `lo`s.

(2) Richtig. Ableitungen können nur über eines der Nonterminale V oder W beendet werden. Von V aus wird die Sprache $(\texttt{lö})^*\texttt{le}$ generiert, d.h., alle derartigen Wörter enden mit `le`. Die Variable W kann nur durch Anwendung der Produktion $T \to \texttt{lo}W$ gleichzeitig mit der Silbe `lo` eingeführt werden.

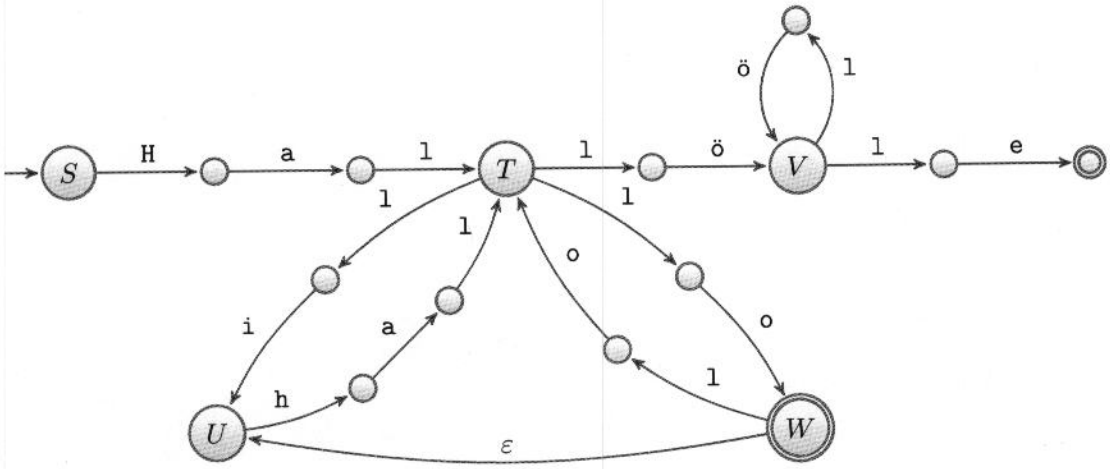
(3) Richtig, da etwa das Wort $\texttt{Hallo}(\texttt{hallo})^n$ für alle $n \geq 0$ in der Sprache $\mathcal{L}(G)$ liegt:

$$\begin{aligned}
S \Rightarrow \texttt{Hal}T &\Rightarrow \texttt{Hallo}W \Rightarrow \texttt{Hallo}U \Rightarrow \texttt{Hallohal}T \\
&\Rightarrow \texttt{Hallohallo}W \Rightarrow \cdots \\
&\Rightarrow \texttt{Hallo}(\texttt{hallo})^n W \Rightarrow \texttt{Hallo}(\texttt{hallo})^n
\end{aligned}$$

(4) Falsch. Z.B. ist `Hallihallihallo` ein Wort der Sprache $\mathcal{L}(G)$ mit zwei `li`s, aber nur einem `lo`.

(5) Richtig. Die Silbe `le` kann nur mithilfe des Nonterminals V, und zwar nur ein einziges Mal, erzeugt werden. (Danach endet die Ableitung.) Da die Variable V ausschließlich durch Anwendung der Produktion $T \to \texttt{lö}V$ oder $V \to \texttt{lö}V$ eingeführt werden kann, wird mindestens ein `lö` erzeugt.

(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 12 (0.4 Punkte)

Queries (Anfragen) an ein PROLOG-Programm beginnen mit den Zeichen `?-`, denen eine Liste von Goals (Zielen) folgt. Zwischen je zwei Goals steht ein Komma. Die Liste besteht aus mindestens einem Goal und wird mit einem Punkt beendet.
Ein Goal besteht aus einem Atom, dem optional eine Argumentliste folgen kann. Eine Argumentliste ist eine in runden Klammern eingeschlossene Liste von einem oder mehreren Termen, die voneinander durch Kommas getrennt werden.
Ein Term ist entweder eine Variable, ein Atom, oder ein Atom gefolgt von einer Argumentliste. Eine Variable ist eine beliebige nicht-leere Folge alphanumerischer Zeichen inklusive dem Unterstreichungszeichen (_), die mit einem Großbuchstaben oder einem Unterstreichungszeichen beginnt.
Ein Atom kann entweder eine nicht-leere Folge alphanumerischer Zeichen (inklusive _) sein, die mit einem Kleinbuchstaben beginnt, oder es kann eine unter einfachen Hochkommas (Apostrophen) eingeschlossene Kette beliebiger ASCII-Zeichen sein. Soll das Hochkomma selber in der ASCII-Zeichenkette vorkommen, muss es verdoppelt werden.
Beispiele:

```
?- atom('Ich bin auch ein Atom', Variable), 'rock''n''roll'(_,_).
?- ist_groesser(s(s(a)),s(a)), write('Ja').
?- ein_komplexer_Term(f(g(X,Y),f(a))).
```

Beschreiben Sie die Menge der Queries mit Hilfe einer kontextfreien Grammatik. Verwenden Sie EBNF-Notationen, um die Grammatik übersichtlich zu strukturieren.

Lösung

$\langle V, T, P, Query \rangle$ mit

$$
\begin{array}{lcll}
V & = & \{Query, Goal, Term, Nonvar, Var, Atom, Alphanum, Upper, Lower, Digit, Ascii\} & , \\
T & = & \{\cdots \text{(alle ASCII-Zeichen)} \cdots\} & , \\
P & = & \{Query & \rightarrow \texttt{"?-"}\ Goal\ \{\texttt{","}\ Goal\}\ \texttt{"."} \quad , \\
 & & Goal & \rightarrow Nonvar \quad , \\
 & & Term & \rightarrow Var \mid Nonvar \quad , \\
 & & Nonvar & \rightarrow Atom\ [\texttt{"("}\ Term\ \{\texttt{","}\ Term\}\ \texttt{")"}] \quad , \\
 & & Var & \rightarrow Upper\ \{Alphanum\} \quad , \\
 & & Atom & \rightarrow Lower\ \{Alphanum\} \mid \texttt{"'"}\ \{Ascii\}\ \texttt{"'"} \quad , \\
 & & Alphanum & \rightarrow Upper \mid Lower \mid Digit \quad , \\
 & & Upper & \rightarrow \texttt{"A"} \mid \cdots \mid \texttt{"Z"} \mid \texttt{"\_"} \quad , \\
 & & Lower & \rightarrow \texttt{"a"} \mid \cdots \mid \texttt{"z"} \quad , \\
 & & Digit & \rightarrow \texttt{"0"} \mid \cdots \mid \texttt{"9"} \quad , \\
 & & Ascii & \rightarrow Alphanum \mid \texttt{"''"} \mid \cdots \text{(Sonderzeichen ohne Hochkomma)} \cdots\ \} \quad .
\end{array}
$$

Aufgabe 13 (0.3 Punkte)

Formalisieren Sie die beiden nachfolgenden Sätze als prädikatenlogische Formeln. Verwenden Sie dabei die Prädikatensymbole *Gallier*, *Römer*, *Verprügelt* und *Angriffslustig* sowie die Konstantensymbole *obelix*, *lacmus* und *cäsar* 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		

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

(b) Manche Gallier verprügeln alle angriffslustigen Römer.

(c) Bestimmen Sie unter Verwendung der Evaluierungsfunktion den Wahrheitswert der Formel

$$\forall x(Gallier(x) \supset Verprügelt(x, lacmus))$$

in folgender Interpretation:

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

Lösung

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

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

(c) $\mathsf{val}_{I,\sigma}(\forall x(\mathit{Gallier}(x) \supset \mathit{Verprügelt}(x, \mathit{lacmus}))) = 1$

$\iff$ 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: $\mathsf{val}_{I,\sigma'}(\mathit{Gallier}(x))$ implies $\mathsf{val}_{I,\sigma'}(\mathit{Verprügelt}(x, \mathit{lacmus})) = 1$

$\iff$ für alle $\sigma' \overset{x}{\sim} \sigma$ gilt:
$(\mathsf{val}_{I,\sigma'}(x) \in I(\mathit{Gallier})$ implies $(\mathsf{val}_{I,\sigma'}(x), \mathsf{val}_{I,\sigma'}(\mathit{lacmus})) \in I(\mathit{Verprügelt})) = 1$

$\iff$ für alle $\sigma' \overset{x}{\sim} \sigma$ gilt:
$(\sigma'(x) \in I(\mathit{Gallier})$ implies $(\sigma'(x), I(\mathit{lacmus})) \in I(\mathit{Verprügelt})) = 1$

$\iff$ für alle $\sigma' \overset{x}{\sim} \sigma$ gilt:
$(\sigma'(x) \in I(\mathit{Gallier})$ implies $(\sigma'(x), \mathrm{Lacmus}) \in I(\mathit{Verprügelt})) = 1$

$\iff$ für alle $\sigma'(x) \in \mathcal{U}$ gilt:
$(\sigma'(x) \in I(\mathit{Gallier})$ implies $(\sigma'(x), \mathrm{Lacmus}) \in I(\mathit{Verprügelt})) = 1$

Verifikation der Aussage:

$\sigma'(x)$	$\sigma'(x) \in I(\mathit{Gallier})$	implies	$(\sigma'(x), \mathrm{Lacmus}) \in I(\mathit{Verprügelt})$
Aerobus	0	**1**	0
Asterix	1	**1**	1
Bonus	0	**1**	0
Cäsar	0	**1**	0
Lacmus	0	**1**	0
Obelix	1	**1**	1
Troubadix	0	**1**	0
Verleihnix	1	**1**	1

Die Formel ist daher wahr in der angegebenen Interpretation.

Aufgabe 14 (0.3 Punkte)

Seien *Isst*/2, *Mädchen*/1, *Speise*/1 und *Gesund*/1 Prädikatensymbole sowie *salat* und *steak* Konstantensymbole mit folgender Bedeutung:

$\mathit{Isst}(x, y)$	... x isst y	$\mathit{Gesund}(x)$	... x ist gesund
$\mathit{Mädchen}(x)$	... x ist ein Mädchen	*salat*	... Salat
$\mathit{Speise}(x)$	... x ist eine Speise	*steak*	... Steak

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

(a) Manche Mädchen essen nur dann Steak, wenn sie auch gesunde Speisen essen.

(b) Gesunde Mädchen essen manche Speisen, aber keinen Salat.

Sei weiters folgende Interpretation gegeben:

$$
\begin{aligned}
\mathcal{U} &= \{\text{Anna, Tina, Mia, Sophie, Salat, Schnitzel, Spätzle, Suppe,} \\
&\qquad \text{Steak, Spaghetti, Pizza}\} \\
I(\mathit{Mädchen}) &= \{\text{Anna, Mia, Sophie}\} \\
I(\mathit{Speise}) &= \{\text{Salat, Schnitzel, Suppe, Steak}\} \\
I(\mathit{Gesund}) &= \{\text{Suppe, Salat, Pizza, Steak}\} \\
I(\mathit{Isst}) &= \{(\text{Anna, Suppe}), (\text{Anna, Salat}), (\text{Anna, Spätzle}), \\
&\qquad (\text{Mia, Schnitzel}), (\text{Mia, Salat}), (\text{Mia, Suppe}), \\
&\qquad (\text{Tina, Pizza}), (\text{Tina, Salat}), (\text{Tina, Spaghetti}), \\
&\qquad (\text{Sophie, Steak}), (\text{Sophie, Spaghetti})\} \\
I(\mathit{salat}) &= \text{Salat} \\
I(\mathit{steak}) &= \text{Steak}
\end{aligned}
$$

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

(c) $\exists x\,(\mathit{Isst}(x, \mathit{salat}) \supset \mathit{Isst}(x, \mathit{steak}))$

(d) $\forall x(\mathit{Isst}(x, \mathit{salat}) \not\equiv \mathit{Isst}(x, \mathit{steak}))$

(e) $\exists x \exists y\,(\mathit{Speise}(y) \wedge \mathit{Mädchen}(x) \wedge \mathit{Isst}(x, y))$

(f) $\forall x \exists y\,(\mathit{Gesund}(x) \supset (\mathit{Mädchen}(y) \wedge \mathit{Isst}(y, x)))$

Lösung

(a) $\exists x(\mathit{Mädchen}(x) \wedge (\mathit{Isst}(x, \mathit{steak}) \supset \exists y(\mathit{Gesund}(y) \wedge \mathit{Speise}(y) \wedge \mathit{Isst}(x, y))))$ oder
$\exists x \exists y(\mathit{Mädchen}(x) \wedge (\mathit{Isst}(x, \mathit{steak}) \supset (\mathit{Gesund}(y) \wedge \mathit{Speise}(y) \wedge \mathit{Isst}(x, y))))$

(b) $\forall x((\mathit{Mädchen}(x) \wedge \mathit{Gesund}(x)) \supset (\exists y(\mathit{Speise}(y) \wedge \mathit{Isst}(x, y)) \wedge \neg \mathit{Isst}(x, \mathit{salat})))$ oder
$\forall x \exists y((\mathit{Mädchen}(x) \wedge \mathit{Gesund}(x)) \supset (\mathit{Speise}(y) \wedge \mathit{Isst}(x, y) \wedge \neg \mathit{Isst}(x, \mathit{salat})))$

(c) Wahr, da (Sophie, Steak) $\in I(\mathit{Isst})$.

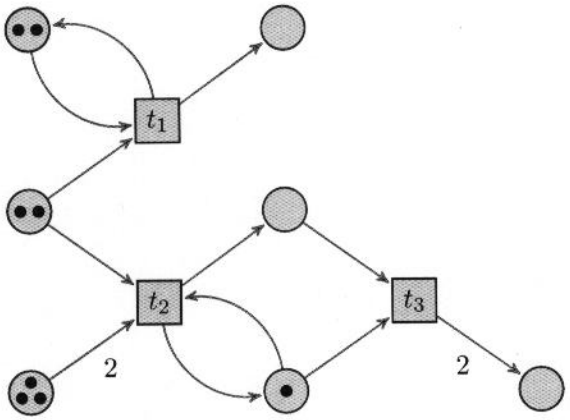
(d) Falsch, da sowohl (Salat, Salat) $\notin I(\mathit{Isst})$ als auch (Salat, Steak) $\notin I(\mathit{Isst})$.

(e) Wahr, da zum Beispiel Suppe $\in I(\mathit{Speise})$, Anna $\in I(\mathit{Mädchen})$ und (Anna, Suppe) $\in I(\mathit{Isst})$.

(f) Falsch, da Pizza $\in I(\mathit{Gesund})$, aber keines der Mädchen isst Pizza: (Anna, Pizza) $\notin I(\mathit{Isst})$, (Mia, Pizza) $\notin I(\mathit{Isst})$ und (Sophie, Pizza) $\notin I(\mathit{Isst})$.

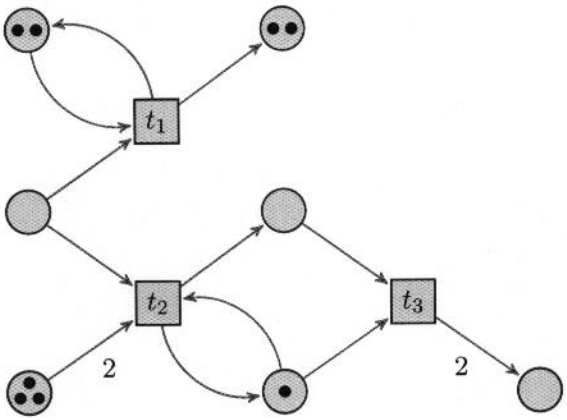
Aufgabe 15 (0.3 Punkte)

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.

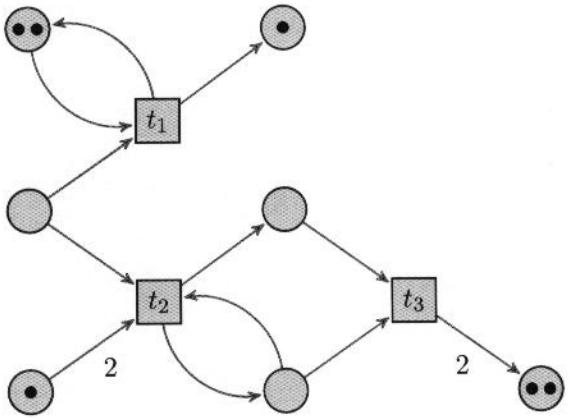


Lösung

Die Transitionsfolge t_1–t_1 liefert:



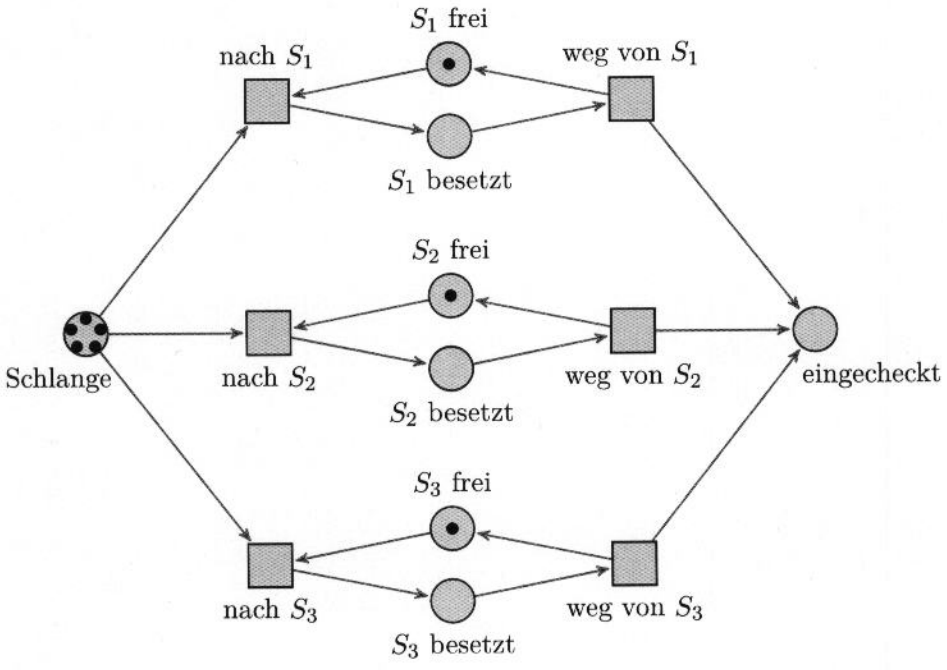
Die Transitionsfolgen t_1–t_2–t_3, t_2–t_1–t_3 und t_2–t_3–t_1 liefern:



Aufgabe 16 (0.3 Punkte)

Am Flughafen gibt es eine Fluglinie mit drei Check-in Schaltern, bei denen die Fluggäste parallel einchecken können. Dazu müssen sie sich in einer gemeinsamen Schlange anstellen. Sobald ein Gast an der Reihe ist, geht er zum nächsten freien Check-in Schalter. Modellieren Sie dieses System mit Hilfe eines Petri-Netzes. Geben Sie eine geeignete Anfangsmarkierung an.

Lösung



Da die Verteilung auf die drei Schalter von einer einzigen Warteschlange aus erfolgt, befinden sich alle Token, die Kunden repräsentieren, zunächst an einer Stelle, der „Schlange“. Kann eine der Transitionen „nach S_1“, „nach S_2“ bzw. „nach S_3“ schalten, heißt das, dass einer der drei Schalter frei ist und der Kunde zu diesem Schalter gehen kann. Dabei ist nicht definiert, zu welchem Schalter er geht, falls gleichzeitig mehrere Schalter frei sind – also welche Transition schaltet, falls mehrere Tranisitonen aktiviert sind. Wenn der Check-in eines Gastes beendet ist, so wird dieser Schalter für den nächsten Gast frei.