2. Übungsblatt (mit Lösungen)

3.0 VU Formale Modellierung SS2020

Marion Scholz, Gernot Salzer

11. Juni 2020

Aufgabe 1 (4 Punkte)

Sei L die Sprache

$$\{\, w \in \{\texttt{F}, \texttt{A}, \texttt{L}, \texttt{O}, \texttt{V}\}^* \mid w \text{ enthält das Teilwort } \texttt{FALL} \text{ oder } \texttt{VOLL} \,\} \ .$$

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

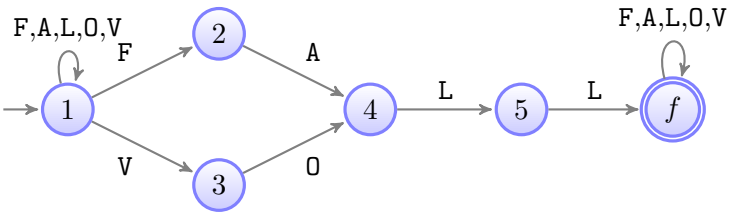
(b) Geben Sie einen nichtdeterministischen Automaten an, der die Sprache L akzeptiert. Der Automat soll der Definition der Sprache direkt entsprechen, sodass die Korrektheit der Modellierung unmittelbar einsichtig ist.

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

Lösung

(a) `[FALOV]*(FA|VO)LL[FALOV]*` oder `^[FALOV]*(FA|VO)LL[FALOV]*$` (falls die Zeichenkette die gesamte Zeile einnehmen soll)

(b) Ein indeterministischer Automat, der diese Sprache darstellt, ist der folgende:



(c) Wir stellen die Übergangsfunktion als Tabelle dar, da diese besser als Ausgangsbasis für die Determinisierung geeignet ist. (Genauer: Wir bestimmen das Ergebnis der erweiterten Übergangsfunktion δ^* für jeden Zustand und jedes Eingabesymbol. Wenn es ε-Übergänge gibt, müssen auch längere Pfade betrachtet werden.)

δ^*	F	A	L	O	V
1	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1\}$	$\{1,3\}$
2	$\{\}$	$\{4\}$	$\{\}$	$\{\}$	$\{\}$
3	$\{\}$	$\{\}$	$\{\}$	$\{4\}$	$\{\}$
4	$\{\}$	$\{\}$	$\{5\}$	$\{\}$	$\{\}$
5	$\{\}$	$\{\}$	$\{f\}$	$\{\}$	$\{\}$
f	$\{f\}$	$\{f\}$	$\{f\}$	$\{f\}$	$\{f\}$

Einen deterministischen Automaten erhalten wir, indem wir den indeterminischen 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 $\{1\}$ bezeichnet, da sich der indeterministische Automat zu Beginn im Zustand 1 (und nur in diesem) befindet. Von diesem Zustand ausgehend erstellen wir zeilenweise die Tabelle für die Übergangsfunktion des deterministischen Automaten.

$\hat{\delta}$	F	A	L	O	V
$\{1\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1\}$	$\{1,3\}$
$\{1,2\}$	$\{1,2\}$	$\{1,4\}$	$\{1\}$	$\{1\}$	$\{1,3\}$
$\{1,3\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1,4\}$	$\{1,3\}$
$\{1,4\}$	$\{1,2\}$	$\{1\}$	$\{1,5\}$	$\{1\}$	$\{1,3\}$
$\{1,5\}$	$\{1,2\}$	$\{1\}$	$\{1,f\}$	$\{1\}$	$\{1,3\}$
$\{1,f\}$	$\{1,2,f\}$	$\{1,f\}$	$\{1,f\}$	$\{1,f\}$	$\{1,3,f\}$
$\{1,2,f\}$	$\{1,2,f\}$	$\{1,4,f\}$	$\{1,f\}$	$\{1,f\}$	$\{1,3,f\}$
$\{1,3,f\}$	$\{1,2,f\}$	$\{1,f\}$	$\{1,f\}$	$\{1,4,f\}$	$\{1,3,f\}$
$\{1,4,f\}$	$\{1,2,f\}$	$\{1,f\}$	$\{1,5,f\}$	$\{1,f\}$	$\{1,3,f\}$
$\{1,5,f\}$	$\{1,2,f\}$	$\{1,f\}$	$\{1,f\}$	$\{1,f\}$	$\{1,3,f\}$

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 f enthält. Dieser wird somit durch das Tupel $\langle \widehat{Q}, \Sigma, \hat{\delta}, \{1\}, \widehat{F} \rangle$ beschrieben, wobei

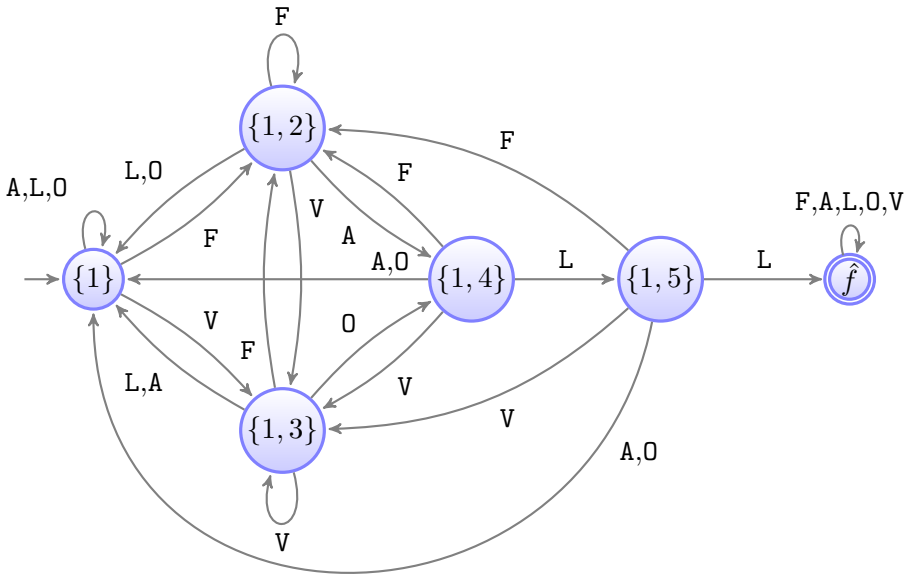
$$\Sigma = \{\mathtt{F}, \mathtt{A}, \mathtt{L}, \mathtt{O}, \mathtt{V}\}$$
$$\widehat{F} = \{\{1,f\}, \{1,2,f\}, \{1,3,f\}, \{1,4,f\}, \{1,5,f\}\}$$
$$\widehat{Q} = \{\{1\}, \{1,2\}, \{1,3\}, \{1,4\}, \{1,5\}\} \cup \widehat{F}$$

Auch ohne Kenntnis des Minimierungsalgorithmus für deterministische Automaten lässt sich erkennen, dass die Endzustände zu einem einzigen zusammengefasst werden können: Einmal in einem solchen angekommen, gelangt man mit jedem weiteren

Symbol wieder in einen Endzustand. Die so vereinfachte Übergangsfunktion sieht folgendermaßen aus:

$\hat{\delta}$	F	A	L	O	V
$\{1\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1\}$	$\{1,3\}$
$\{1,2\}$	$\{1,2\}$	$\{1,4\}$	$\{1\}$	$\{1\}$	$\{1,3\}$
$\{1,3\}$	$\{1,2\}$	$\{1\}$	$\{1\}$	$\{1,4\}$	$\{1,3\}$
$\{1,4\}$	$\{1,2\}$	$\{1\}$	$\{1,5\}$	$\{1\}$	$\{1,3\}$
$\{1,5\}$	$\{1,2\}$	$\{1\}$	$\hat{f}$	$\{1\}$	$\{1,3\}$
$\hat{f}$	$\hat{f}$	$\hat{f}$	$\hat{f}$	$\hat{f}$	$\hat{f}$

Graphische Darstellung dieses vereinfachten deterministischen Automaten:



Aufgabe 2 (2 Punkte)

Geben Sie endliche Automaten an, die dieselbe Sprache beschreiben wie die folgenden regulären Ausdrücke in algebraischer Notation.

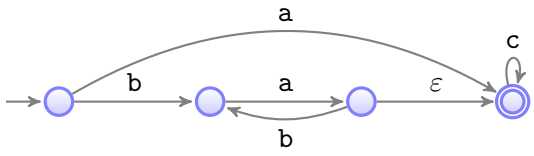
(a) $(\mathtt{a} + (\mathtt{ba})^+)\mathtt{c}^*$

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

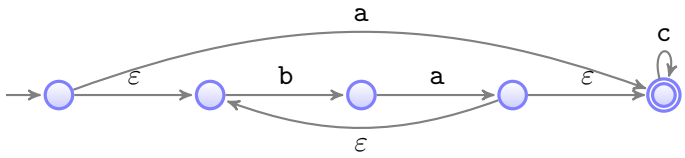
Lösung

Die gesuchten Automaten können mit dem allgemeinen Verfahren konstruiert werden, enthalten dann aber in der Regel viel mehr Zustände und ε-Kanten als notwendig. Die folgenden Automaten wurden bereits vereinfacht.

(a)

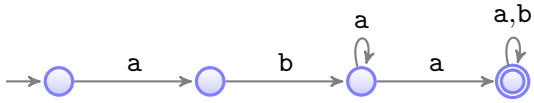


oder



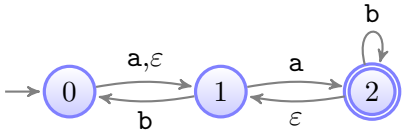
oder ...

(b)



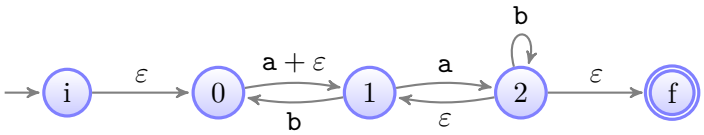
Aufgabe 3 (3 Punkte)

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.



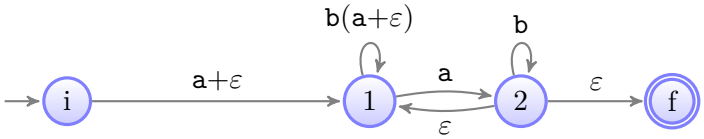
Lösung

Neuer Anfangs- und Endzustand:

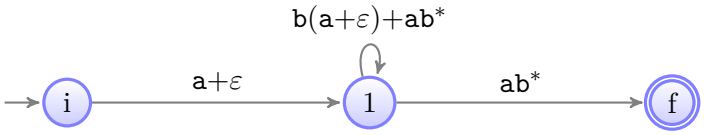


Wir eliminieren die Zustände in der Reihenfolge 0, 2 und 1; die anderen Reihenfolgen sind ebenfalls möglich.

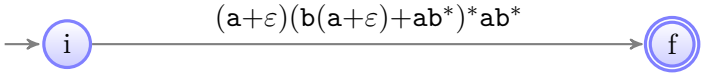
Elimination von Zustand 0:



Elimination von Zustand 2:



Elimination von Zustand 1:



Die Sprache des ursprünglichen Automaten wird also durch den Ausdruck

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

beschrieben.

Aufgabe 4 (3 Punkte)

Auf Wikipedia findet man folgende Beschreibung des Begriffs *Paragraph* (hier stark gekürzt).

> *Der Paragraph dient zur Einteilung in aufzählendem Schrifttum, wie bei Gesetzen, Verträgen oder Lehrbüchern. Im deutschen Sprachraum wird als Paragraphenzeichen „§“ verwendet. Bezieht man sich auf mehr als einen Paragraphen, werden zwei Paragraphenzeichen nebeneinandergestellt: „§§“.*
>
> *Nach dem/den Paragraphenzeichen folgt eine positive Zahl. Falls es sich um mehrere Paragraphen handelt (eingeleitet durch §§) folgt der Zahl die Abkürzung „f.“ (kurz für „und Folgender“) oder „ff.“ (kurz für „und die Folgenden“).*
>
> *In Österreich gliedern sich die Paragraphen in Absätze (Abs.), die Absätze in Ziffern (Z) und/oder in Buchstaben (litera, kurz: lit); diese können weiter durch kleingeschriebene römische Zahlen (sublitera, kurz: sublit) untergliedert werden. Vereinfachend nehmen wir an, dass römische Zahlen beliebige Kombination der Buchstaben i, v und x sind. Anschließend folgt die Abkürzung des Gesetzes, auf das sich die Referenz bezieht. Diese beginnt mit einem Großbuchstaben und wird von beliebig vielen Groß- und Kleinbuchstaben gefolgt.*

Durch Angabe von Paragraph, Absatz und Ziffer/Buchstabe kann auf einzelne Gesetzesstellen Bezug genommen werden. Beispiele:

- §16 Abs. 1 lit a StVO
- §§79 f. Abs. 1 Z 1 GTG
- §§79 ff. Abs. 1 Z 1 GTG
- §16 Abs. 1 lit a sublit ii FBG

Beschreiben Sie den Aufbau solcher Gesetzesbezüge mit den folgenden Methoden. Treffen Sie sinnvolle Annahmen, wenn Ihnen Informationen fehlen. Gehen Sie der Einfachheit halber davon aus, dass sich zwischen den einzelnen Bestandteilen keine Leerzeichen befinden.

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

(b) Geben Sie einen regulären Ausdruck in POSIX-Notation an, der alle Zeilen beschreibt, die *ausschließlich* eine derartige Gesetzesstelle enthalten.

(c) Geben Sie Syntaxdiagramme für Ihren regulären Ausdruck aus Teil a an.

Lösung

Die folgende Lösung geht davon aus, dass eine Gesetzesreferenz auch nur aus der Paragraphenangabe und einem Absatz bestehen kann, also sowohl Ziffer als auch Buchstabe fehlen können.

(a) Die Sprache der Gesetzesreferenzen lässt sich durch den Ausdruck

$$\mathit{Paragraph}\,\mathit{Absatz}\,(\varepsilon + \mathit{Ziffer})\,(\varepsilon + \mathit{Lit}\,(\varepsilon + \mathit{Sublit}))\,\mathit{Name}$$

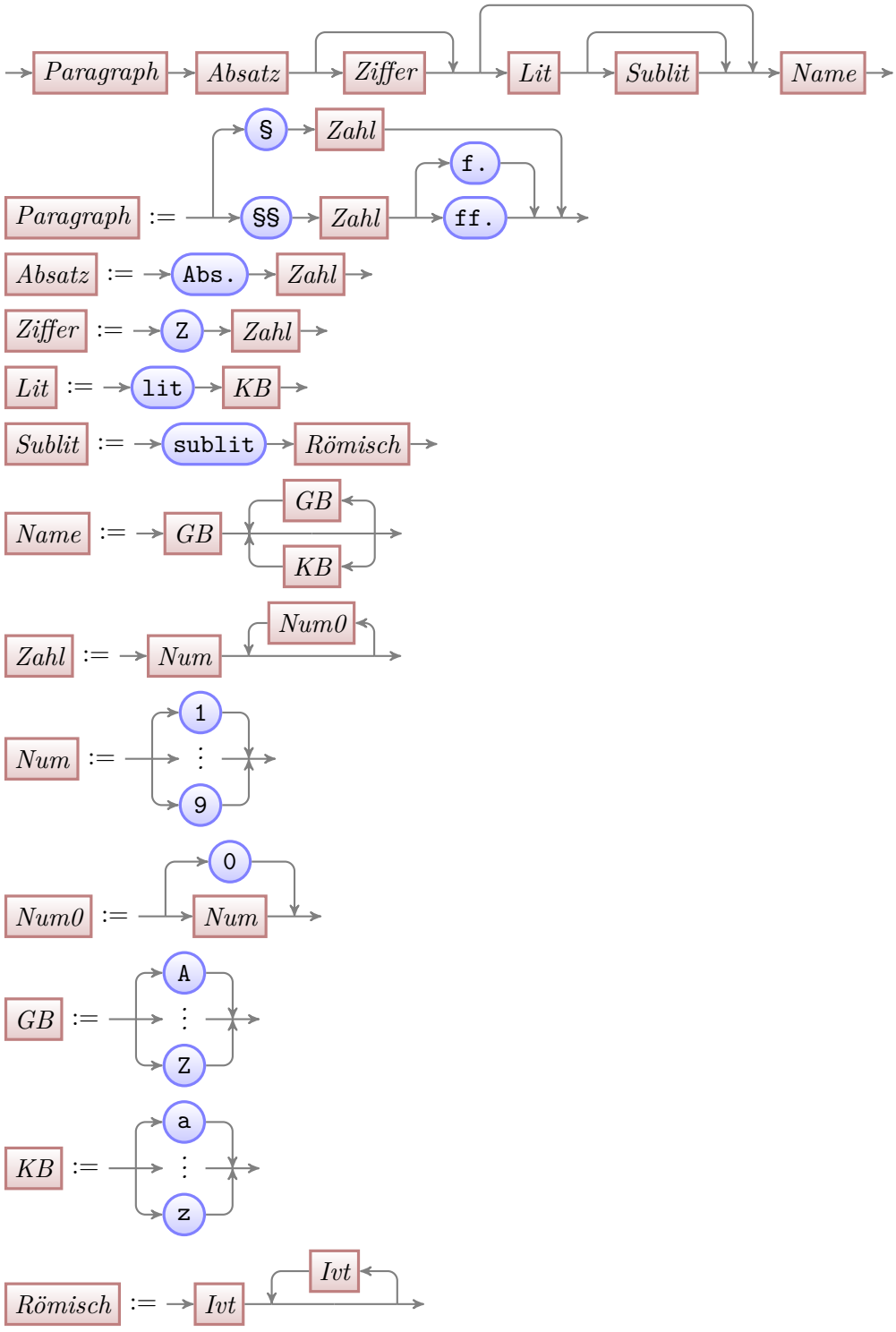
beschreiben, wobei die folgenden Abkürzungen verwendet werden:

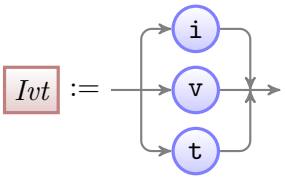
$$\begin{aligned}
\mathit{Paragraph} &:= \texttt{"§"}\,\mathit{Zahl} + \texttt{"§§"}\,\mathit{Zahl}\,(\texttt{"f."} + \texttt{"ff."}) \\
\mathit{Absatz} &:= \texttt{"Abs."}\,\mathit{Zahl} \\
\mathit{Ziffer} &:= \texttt{"Z"}\,\mathit{Zahl} \\
\mathit{Lit} &:= \texttt{"lit"}\,\mathit{Kb} \\
\mathit{Sublit} &:= \texttt{"sublit"}\,\mathit{Römisch} \\
\mathit{Name} &:= \mathit{Gb}\,(\mathit{Gb} + \mathit{Kb})^* \\
\mathit{Zahl} &:= \mathit{Num}\,\mathit{Num0}^* \\
\mathit{Num} &:= \texttt{"1"} + \cdots + \texttt{"9"} \\
\mathit{Num0} &:= \texttt{"0"} + \mathit{Num} \\
\mathit{Gb} &:= \texttt{"A"} + \cdots + \texttt{"Z"} \\
\mathit{Kb} &:= \texttt{"a"} + \cdots + \texttt{"z"} \\
\mathit{Römisch} &:= \mathit{Ivt}\,\mathit{Ivt}^* \\
\mathit{Ivt} &:= \texttt{"i"} + \texttt{"v"} + \texttt{"x"}
\end{aligned}$$

Wir schreiben Symbole des Alphabets unter Anführungszeichen, um sie von algebraischen Symbolen (Metanotation) zu unterscheiden.

(b)
```
^(§[1-9][0-9]*|§§[1-9][0-9]*ff?\.)Abs\.[1-9][0-9]*(Z[1-9][0-9]*)?
                                  (lit[a-z](sublit[ivx]+)?)?[A-Z][a-zA-Z]*$
```

(c) Syntaxdiagramm:





Aufgabe 5 (5 Punkte)

Vereinfachen Sie die folgenden Ausdrücke.

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

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

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

(d) $\{\mathtt{ac}, \mathtt{b}, \mathtt{aba}\} \cdot \{\}$

(e) $\{\mathtt{ad}, \mathtt{bc}, \mathtt{ab}\} \cup \{\}^*$

Lösung

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

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

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

(d) $L_4 = \{\mathtt{ac}, \mathtt{b}, \mathtt{aba}\} \cdot \{\} = \{\}$

(e) $L_5 = \{\mathtt{ad}, \mathtt{bc}, \mathtt{ab}\} \cup \{\}^* = \{\mathtt{ad}, \mathtt{bc}, \mathtt{ab}\} \cup \{\varepsilon\} = \{\mathtt{ad}, \mathtt{bc}, \mathtt{ab}, \varepsilon\}$

Aufgabe 6 (5 Punkte)

Seien L_1 und L_2 Sprachen. Geben Sie für die folgenden Gleichungen/Aussagen an, ob sie gelten oder nicht. Falls die Gleichung/Aussage gilt, geben Sie eine Begründung an. Gilt die Gleichung/Aussage nicht, so geben Sie ein Gegenbeispiel an.

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

(b) $L_1^* \cdot \{\varepsilon\} = \{\}$

(c) $\varepsilon \in L_1 \cdot L_2$ genau dann, wenn $\varepsilon \in L_1$ und $\varepsilon \in L_2$.

(d) Ist L_1 regulär, so ist jede Sprache $L \subseteq L_1$ auch regulär.

(e) Ist die Menge L_1 endlich, so ist L_1 regulär.

Lösung

(a) Die Behauptung gilt nicht. Seien $L_1 = \{\texttt{a}\}$ und $L_2 = \{\texttt{b}\}$. Dann gilt $L_1 \cdot L_2 = \{\texttt{ab}\}$, jedoch $L_2 \cdot L_1 = \{\texttt{ba}\}$.

(b) Die Behauptung gilt nicht. Es gilt ja $L \cdot \{\varepsilon\} = L$ für jede Sprache L. Dies ist unmittelbar aus der Definition der Konkatenation von Sprachen einsichtig. Daher ist $L_1^* \cdot \{\varepsilon\} = L_1^*$. Diese Sprache ist jedoch immer verschieden von $\{\}$, da sie mindestens das Leerwort ε enthält.

(c) Die Behauptung gilt. Wir teilen die Begründung in zwei Teile, die den beiden Richtungen der Äquivalenz entsprechen.

Teil 1: „Wenn $\varepsilon \in L_1 \cdot L_2$, dann $\varepsilon \in L_1$ und $\varepsilon \in L_2$."
Wenn $\varepsilon \in L_1 \cdot L_2$, dann gibt es Wörter $a \in L_1$ und $b \in L_2$, sodass $a \cdot b = \varepsilon$. Die Konkatenation $a \cdot b$ kann jedoch nur dann das Leerwort ergeben, wenn $a = \varepsilon$ und $b = \varepsilon$. Somit gilt $\varepsilon \in L_1$ und $\varepsilon \in L_2$.

Teil 2: „Wenn $\varepsilon \in L_1$ und $\varepsilon \in L_2$, dann $\varepsilon \in L_1 \cdot L_2$."
Die verkettete Sprache $L_1 \cdot L_2$ enthält alle Wörter, die sich aus der Verkettung von Wörtern in L_1 mit Wörtern in L_2 ergeben, also auch $\varepsilon \cdot \varepsilon = \varepsilon$.

(d) Die Behauptung gilt nicht. So ist z.B. $L_1 = \{a, b\}^*$ regulär, jedoch die Sprache $L := \{\texttt{a}^n\texttt{b}^n \mid n \geq 0\}$ nicht. Es gilt aber $L \subseteq L_1$, was die Behauptung widerlegt.

(e) Die Behauptung gilt. Ist die Sprache L_1 endlich, lässt sie sich als $\{w_1, \ldots, w_n\}$ für gewisse Wörter $w_1, \ldots, w_n$ schreiben. Jedes dieser Wörter ist eine endlichen Folge von Symbolen des Alphabets, lässt sich also als $a_1 \cdots a_k$ für gewisse Symbole $a_1, \ldots, a_k$ schreiben. Laut Definition der regulären Sprachen sind $\{a_1\}$, …, $\{a_k\}$ reguläre Sprachen und ebenso deren Verkettung $\{a_1\} \cdot \cdots \cdot \{a_k\} = \{a_1 \cdots a_k\}$. Somit ist $\{w_i\}$ für alle $i = 1, \ldots, n$ eine reguläre Sprache. Laut Definition der regulären Sprachen ist die Vereinigung zweier regulärer Sprachen wieder regulär, daher ist $(\cdots(\{w_1\} \cup \{w_2\}) \cup \cdots \cup \{w_n\}) = \{w_1, \ldots, w_n\} = L_1$ eine reguläre Sprache.

Aufgabe 7 (6 Punkte)

Gegeben sei die Grammatik $G = \langle N, T, P, A\rangle$, wobei

$$
\begin{aligned}
N &= \{A, B, C, D\} \\
T &= \{\texttt{.}, \texttt{a}, \texttt{e}, \texttt{h}, \texttt{j}, \texttt{l}, \texttt{u}\} \\
P &= \{\, A \to B\ \texttt{"."}\ , \\
&\qquad B \to \texttt{"hal"}\ C\ \texttt{"ja"}\ , \\
&\qquad C \to \texttt{"le"}\ C \mid \texttt{"l"}\ D\ , \\
&\qquad D \to \texttt{"u"}\ D \mid \texttt{"u"}\}
\end{aligned}
$$

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, und ändern Sie das Wort möglichst geringfügig ab, sodass es in der Sprache liegt.

(a) `hallelelu uja.`

(b) `hallelu`

Sind folgende Aussagen wahr oder falsch? Begründen Sie Ihre Antwort!

(c) Jedes Wort enthält mindestens zwei `l`.

(d) Jedes Wort enthält mindestens ein `ja`.

Überlegen Sie weiters:

(e) Wie lautet das kürzeste Wort der Sprache?

(f) Ist es möglich, die Sprache $\mathcal{L}(G)$ auch durch einen endlichen Automaten zu beschreiben? Falls ja, geben Sie einen derartigen Automaten an. Falls nein, begründen Sie, warum das nicht geht.

Lösung

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

$$\begin{aligned} A &\Rightarrow B\,\texttt{"."} \\ &\Rightarrow \texttt{"hal"}\,C\,\texttt{"ja."} \\ &\Rightarrow \texttt{"halle"}\,C\,\texttt{"ja."} \\ &\Rightarrow \texttt{"hallele"}\,C\,\texttt{"ja."} \\ &\Rightarrow \texttt{"hallelel"}\,D\,\texttt{"ja."} \\ &\Rightarrow \texttt{"hallelelu"}\,D\,\texttt{"ja."} \\ &\Rightarrow \texttt{"halleleluuja."} \end{aligned}$$

(b) Das Wort liegt nicht in der Sprache $\mathcal{L}(G)$. Die Produktionen sind so aufgebaut, dass jedes Wort mit `ja.` enden muss.
Ein gültiges Wort wäre beispielsweise `halleluja.`

(c) Richtig. Jede Ableitung eines Wortes in dieser Grammatik beginnt mit einer der beiden folgenden Alternativen.

$$A \Rightarrow B\,\texttt{"."} \Rightarrow \texttt{"hal"}\,C\,\texttt{"ja."} \Rightarrow \begin{cases} \texttt{"halle"}\,C\,\texttt{"ja."} \Rightarrow \cdots \\ \texttt{"hall"}\,D\,\texttt{"ja."} \Rightarrow \cdots \end{cases}$$

In beiden Fällen enthält das Wort zumindest zweimal den Buchstaben `l`.

(d) Richtig. Jede Ableitung eines Wortes in dieser Grammatik beginnt mit

$$A \Rightarrow B\,\texttt{"."} \Rightarrow \texttt{"hal"}\,C\,\texttt{"ja."} \Rightarrow \cdots$$

Somit endet jedes Wort mit `"ja."`

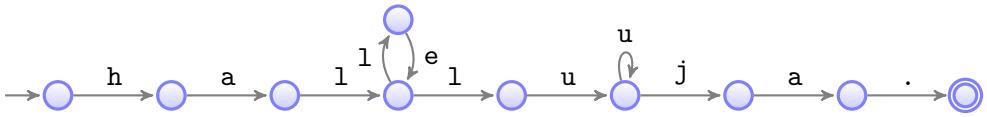
(e) Das kürzeste Wort lautet `halluja.`:

$$A \Rightarrow B\,\texttt{"."} \Rightarrow \texttt{"hal"}\,C\,\texttt{"ja."} \Rightarrow \texttt{"hall"}\,D\,\texttt{"ja."} \Rightarrow \texttt{"halluja."}$$

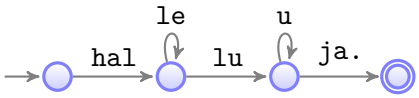
Jede andere Ableitung führt zu längeren Wörtern, da die oben angeführten Ableitungsschritte in jeder Ableitung vorkommen müssen.

(f) Ja, die Sprache lässt sich durch einen endlichen Automaten beschreiben, ist also regulär. Das liegt daran, dass überall dort, wo in einer Schleife beliebig viele Kopien eines Symbols oder Wortes erzeugt werden können, die Anzahl nicht relevant für andere Teile des Wortes ist. Die Sprache enthält keine beliebig tief schachtelbaren Klammerstrukturen.

Legt man das Alphabet $\{\texttt{h}, \texttt{a}, \texttt{l}, \texttt{e}, \texttt{u}, \texttt{j}, \texttt{.}\}$ zugrunde, lässt sich die Sprache durch den folgenden nicht-deterministischen Automaten darstellen.



Legt man hingegen das Alphabet $\{\texttt{hal}, \texttt{le}, \texttt{l}, \texttt{u}, \texttt{ja.}\}$ zugrunde, in dem Buchstabengruppen als einzelne Symbole betrachtet werden, lässt sich die Sprache durch den folgenden deterministischen Automaten beschreiben.



Aufgabe 8 (6 Punkte)

Gegeben seien Rechenaufgaben für die vier Grundrechenarten mit dem folgenden Aufbau:

- Jede Rechenaufgabe besteht aus zwei Zahlen, zwischen denen eine Grundrechenart (`+`, `-`, `*` oder `:`) steht, optional gefolgt von einem Gleichheitzeichen (`=`).
- Vor und nach der Rechenoperation sowie vor dem Gleichheitszeichen befindet sich jeweils ein Leerzeichen.
- Die Zahlen sind ganze positive oder negative Zahlen ohne führende Nullen.

Sei $\mathcal{R}$ die Menge aller derartigen Rechenaufgaben. Beispiele für Wörter in $\mathcal{R}$ sind:

`123␣+␣0`
`-3␣*␣-12␣=`
`4␣:␣-2␣=`

(Das Zeichen ␣ markiert ein Leerzeichen.)

(a) Spezifizieren Sie die Sprache $\mathcal{R}$ mit Hilfe einer kontextfreien Grammatik. Verwenden Sie EBNF-Notationen, um die Grammatik übersichtlich zu strukturieren.

Handelt es sich bei $\mathcal{R}$ um eine reguläre Sprache, d.h., ließe sich $\mathcal{R}$ auch durch einen regulären Ausdruck beschreiben? Begründen Sie Ihre Antwort.

Sei $\mathcal{R}'$ eine Erweiterung der Sprache $\mathcal{R}$, bei der überall dort, wo eine Zahl stehen darf, auch ein Ausdruck vorkommen darf, der aus zwei Zahlen mit einem der vier Operatoren dazwischen besteht, wobei auch hier die Zahlen wieder durch einen Ausdruck ersetzt sein dürfen. Beispiele für Wörter in $\mathcal{R}'$ sind:

`120␣+␣3␣+␣0`
`-3␣*␣-1␣-␣2␣+␣-9␣=`
`4␣:␣-2␣+␣-3␣*␣-12␣=`

(b) Spezifizieren Sie die Sprache $\mathcal{R}'$ mit Hilfe einer kontextfreien Grammatik. Verwenden Sie EBNF-Notationen, um die Grammatik übersichtlich zu strukturieren.

Handelt es sich bei $\mathcal{R}'$ um eine reguläre Sprache? Begründen Sie Ihre Antwort.

Sei $\mathcal{R}''$ eine Erweiterung der Sprache $\mathcal{R}$ ähnlich wie $\mathcal{R}'$, nur dass die statt der Zahlen eingeführten Ausdrücke geklammert sein müssen. Beispiele für Wörter in $\mathcal{R}''$ sind:

`(120␣+␣3)␣+␣0`
`-3␣*␣((-1␣-␣2)␣+␣-9)␣=`
`(4␣:␣-2)␣+␣(-3␣*␣-12)␣=`

(c) Spezifizieren Sie die Sprache $\mathcal{R}''$ mit Hilfe einer kontextfreien Grammatik. Verwenden Sie EBNF-Notationen, um die Grammatik übersichtlich zu strukturieren.

Handelt es sich bei $\mathcal{R}''$ um eine reguläre Sprache? Begründen Sie Ihre Antwort.

Lösung

(a) Die Sprache $\mathcal{R}$ wird durch die Grammatik $\langle N, T, P, \mathit{Aufgabe}\rangle$ beschrieben, wobei

$$
\begin{aligned}
N &= \{\mathit{Aufgabe}, \mathit{Operator}, \mathit{Zahl}, \mathit{Ziffer}, \mathit{Ziffer0}\},\\
T &= \{\cdots \textit{alle Zeichen in den Produktionen zwischen Anführungszeichen} \cdots\},\\
P &= \{\ \ \mathit{Aufgabe} \to \mathit{Zahl}\ \texttt{"␣"}\ \mathit{Operator}\ \texttt{"␣"}\ \mathit{Zahl}\ [\texttt{"␣="}]\ ,\\
&\qquad \mathit{Operator} \to \texttt{"+"} \mid \texttt{"-"} \mid \texttt{"*"} \mid \texttt{":"}\ ,\\
&\qquad\quad \mathit{Zahl} \to \texttt{"0"} \mid [\texttt{"-"}]\ \mathit{Ziffer}\{\mathit{Ziffer0}\}\ ,\\
&\qquad\quad \mathit{Ziffer} \to \texttt{"1"} \mid \cdots \mid \texttt{"9"}\ ,\\
&\qquad \mathit{Ziffer0} \to \texttt{"0"} \mid \mathit{Ziffer}\ \ \}
\end{aligned}
$$

Bei $\mathcal{R}$ handelt es sich um eine reguläre Sprache, da die einzige Art der Schachtelung eine beliebige Hintereinanderreihung von Ziffern ist. Die Sprache $\mathcal{R}$ lässt sich auch durch den folgenden regulären Ausdruck in Posix-Notation darstellen.

```
(0|-?[1-9][0-9]*)␣[-+*:]␣(0|-?[1-9][0-9]*)(␣=)?
```

Beachten Sie, dass das Minuszeichen unmittelbar neben die Klammer gestellt werden muss, damit es vom RegEx-Interpreter nicht als Bereich so wie in `[0-9]` verstanden wird. Beispielsweise liefert `[+-*:]` eine Fehlermeldung, wenn (wie im Ascii-Code) `+` einen höheren Code als `*` hat. Der Ausdruck `[*-+:]` repräsentiert typischerweise die Sprache $\{$`*`, `+`, `:`$\}$ (ohne Minus), da `+` in vielen Codierungen direkt auf `*` folgt.

(b) Die Sprache $\mathcal{R}'$ wird durch die Grammatik $\langle N, T, P, \mathit{Aufgabe}\rangle$ beschrieben, wobei

$$
\begin{aligned}
N &= \{\mathit{Aufgabe}, \mathit{Ausdruck}, \mathit{OpAusdruck}, \mathit{Operator}, \mathit{Zahl}, \mathit{Ziffer}, \mathit{Ziffer0}\},\\
T &= \{\cdots \textit{alle Zeichen in den Produktionen zwischen Anführungszeichen} \cdots\},\\
P &= \{\quad \mathit{Aufgabe} \to \mathit{OpAusdruck}\,[\texttt{"␣="}]\ ,\\
&\qquad \mathit{OpAusdruck} \to \mathit{Ausdruck}\ \texttt{"␣"}\ \mathit{Operator}\ \texttt{"␣"}\ \mathit{Ausdruck}\ ,\\
&\qquad\quad \mathit{Ausdruck} \to \mathit{Zahl} \mid \mathit{OpAusdruck}\ ,\\
&\qquad\quad \mathit{Operator} \to \texttt{"+"} \mid \texttt{"-"} \mid \texttt{"*"} \mid \texttt{":"}\ ,\\
&\qquad\quad\quad \mathit{Zahl} \to \texttt{"0"} \mid [\texttt{"-"}]\ \mathit{Ziffer}\{\mathit{Ziffer0}\}\ ,\\
&\qquad\quad\quad\ \mathit{Ziffer} \to \texttt{"1"} \mid \cdots \mid \texttt{"9"}\ ,\\
&\qquad\quad\quad\ \mathit{Ziffer0} \to \texttt{"0"} \mid \mathit{Ziffer}\quad \}
\end{aligned}
$$

Auch bei $\mathcal{R}'$ handelt es sich um eine reguläre Sprache. Gegenüber können nun zwar Ausdrücke beliebig geschachtelt werden, diese Schachtelung lässt sich aber genauso als eine Folge lesen, in der Zahlen und Operatoren alternieren. Beispielsweise lässt sich $\mathcal{R}'$ durch folgenden regulären Ausdruck in Posix-Notation darstellen.

```
(0|-?[1-9][0-9]*)(␣[-+*:]␣(0|-?[1-9][0-9]*))+(␣=)?
```

Der Unterschied zum Ausdruck in der vorigen Aufgabe sind lediglich die Zeichen

```
(                                   )+
```

die eine beliebig lange Abfolge von Operatoren und Zahlen zulassen.

(c) Die Sprache $\mathcal{R}''$ wird durch die Grammatik $\langle N, T, P, \mathit{Aufgabe}\rangle$ beschrieben, wobei

$$
\begin{aligned}
N &= \{\mathit{Aufgabe}, \mathit{Ausdruck}, \mathit{OpAusdruck}, \mathit{Operator}, \mathit{Zahl}, \mathit{Ziffer}, \mathit{Ziffer0}\},\\
T &= \{\cdots \textit{alle Zeichen in den Produktionen zwischen Anführungszeichen} \cdots\},\\
P &= \{\quad \mathit{Aufgabe} \to \mathit{OpAusdruck}\,[\texttt{"␣="}]\ ,\\
&\qquad \mathit{OpAusdruck} \to \mathit{Ausdruck}\ \texttt{"␣"}\ \mathit{Operator}\ \texttt{"␣"}\ \mathit{Ausdruck}\ ,\\
&\qquad\quad \mathit{Ausdruck} \to \mathit{Zahl} \mid \texttt{"("}\ \mathit{OpAusdruck}\ \texttt{")"}\ ,\\
&\qquad\quad \mathit{Operator} \to \texttt{"+"} \mid \texttt{"-"} \mid \texttt{"*"} \mid \texttt{":"}\ ,\\
&\qquad\quad\quad \mathit{Zahl} \to \texttt{"0"} \mid [\texttt{"-"}]\ \mathit{Ziffer}\{\mathit{Ziffer0}\}\ ,\\
&\qquad\quad\quad\ \mathit{Ziffer} \to \texttt{"1"} \mid \cdots \mid \texttt{"9"}\ ,\\
&\qquad\quad\quad\ \mathit{Ziffer0} \to \texttt{"0"} \mid \mathit{Ziffer}\quad \}
\end{aligned}
$$

Diese Grammatik unterscheidet sich von jener für $\mathcal{R}'$ nur in der zweiten Produktion für *OpAusdruck*: Alle OpAusdrücke ausgenommen jenem auf der obersten Ebene müssen nun in Klammern gesetzt werden.

Genau diese Klammern sind dafür verantwortlich, dass es sich bei $\mathcal{R}''$ um keine reguläre Sprache handelt. OpAusdücke können beliebig tief geschachtelt werden; die Buchführung über die öffnenden und schließenden Klammern erfordert unbegrenzten Speicher, was die Ausdruckskraft von endlichen Automaten übersteigt.

Aufgabe 9 (5 Punkte)

Wählen Sie geeignete Prädikaten- und Konstantensymbole und übersetzen Sie die folgenden Sätze in prädikatenlogische Formeln.

(a) Quaxi ist ein Frosch.

(b) Alle Frösche sind grün.

(c) Manche Frösche fressen Fliegen.

(d) Jeder grüne Frosch frisst Fliegen.

(e) Es gibt einen Frosch, der Fliegen aber keine Spinnen frisst.

Lösung

Seien $\mathit{Frisst}/2$, $\mathit{Frosch}/1$, und $\mathit{Grün}/1$ Prädikatensymbole sowie *quaxi*, *fliegen* und *spinnen* Konstantensymbole mit folgender Bedeutung:

$\mathit{Frisst}(x, y)$	... x frisst y	*quaxi*	... Quaxi
$\mathit{Frosch}(x)$	... x ist ein Frosch	*fliegen*	... Fliegen
$\mathit{Grün}(x)$	... x ist grün	*spinnen*	... Spinnen

(a) $\mathit{Frosch}(\mathit{quaxi})$

(b) $\forall x(\mathit{Frosch}(x) \supset \mathit{Grün}(x))$

(c) $\exists x(\mathit{Frosch}(x) \wedge \mathit{Frisst}(x, \mathit{fliegen}))$

Die Übersetzung von „manche“ mit einem Existenzquantor ist nur eine grobe Annäherung. Der Existenzquantor steht für „mindestens ein“, demnach ist die Formel auch wahr, wenn alle oder fast alle Frösche Fliegen fressen. In so einer Situation würde man aber den natürlichsprachigen Satz „Manche Frösche fressen Fliegen“ als falsch einstufen, da manche meist als „es gibt einige wenige“ interpretiert wird.

(d) $\forall x((\mathit{Frosch}(x) \wedge \mathit{Grün}(x)) \supset \mathit{Frisst}(x, \mathit{fliegen}))$

(e) $\exists x(\mathit{Frosch}(x) \wedge \mathit{Frisst}(x, \mathit{fliegen}) \wedge \neg \mathit{Frisst}(x, \mathit{spinnen}))$

Aufgabe 10 (5 Punkte)

Seien *SchautAn*, *Kind*, *Lustig* und *Sendung* Prädikatensymbole und *teletubbies* und *spongebob* Konstantensymbole mit folgender Bedeutung:

$SchautAn(x, y)$	... x schaut y an	$Sendung(x)$	... x ist eine Fernsehserie
$Kind(x)$	... x ist ein Kind	*teletubbies*	... Teletubbies
$Lustig(x)$	... x ist lustig	*spongebob*	... Spongebob

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

(a) Es gibt lustige Kinder, die dann und nur dann Teletubbies schauen, wenn sie auch Spongebob schauen.

(b) Es gibt lustige Sendungen, die von allen Kinder geschaut werden.

Sei weiters folgende Interpretation I gegeben:

$$\begin{aligned}
\mathcal{U} &= \{\text{Tom}, \text{Anna}, \text{Flo}, \text{Karo}, \text{Spongebob}, \text{Teletubbies}, \text{Barbapapas}, \\
&\quad \text{12oder3}, \text{Niklas}, \text{Heidi}\} \\
I(Kind) &= \{\text{Tom}, \text{Flo}, \text{Anna}, \text{Karo}\} \\
I(Lustig) &= \{\text{Spongebob}, \text{Teletubbies}, \text{Barbapapas}\} \\
I(Sendung) &= \{\text{12oder3}, \text{Teletubbies}, \text{Niklas}, \text{Barbapapas}, \text{Heidi}\} \\
I(SchautAn) &= \{(\text{Tom}, \text{Heidi}), (\text{Tom}, \text{12oder3}), \\
&\quad (\text{Flo}, \text{Barbapapas}), (\text{Flo}, \text{Heidi}), (\text{Flo}, \text{12oder3}), \\
&\quad (\text{Anna}, \text{Barbapapas}), (\text{Anna}, \text{Teletubbies}), (\text{Anna}, \text{Heidi}), \\
&\quad (\text{Karo}, \text{Heidi}), (\text{Karo}, \text{Spongebob})\} \\
I(heidi) &= \text{Heidi} \\
I(niklas) &= \text{Niklas}
\end{aligned}$$

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

(c) $\forall x\,(Kind(x) \wedge SchautAn(x, heidi))$

(d) $\exists x \forall y\,((Sendung(y) \wedge Lustig(y)) \supset SchautAn(x, y))$

(e) $\forall x\,(SchautAn(x, heidi) \not\equiv SchautAn(x, niklas))$

Lösung

(a) $\exists x(Kind(x) \wedge Lustig(x) \wedge (SchautAn(x, teletubbies) \equiv SchautAn(x, spongebob)))$

(b) $\exists y\,(Sendung(y) \wedge Lustig(y) \wedge \forall x\,(Kind(x) \supset SchautAn(x, y)))$

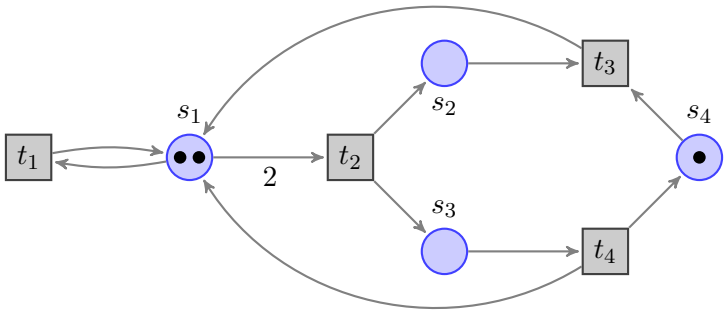
(c) Übersetzung: Alle sind Kinder und schauen Heidi.
Diese Aussage ist falsch; etwa ist Niklas kein Kind

(d) Übersetzung: Es gibt etwas, das alle lustigen Fernsehsendungen anschaut.
Diese Aussage ist richtig, Anna schaut die Teletubbies und die Barbapapas.

(e) Übersetzung: Alles schaut entweder Heidi oder Niklas (aber nicht beides).
Diese Aussage ist falsch, da $(\text{Heidi}, \text{Heidi}) \notin I(SchautAn)$ und $(\text{Heidi}, \text{Niklas}) \notin I(SchautAn)$.

Aufgabe 11 (3 Punkte)

Gegeben sei das folgende Petri-Netz mit Anfangsmarkierung.



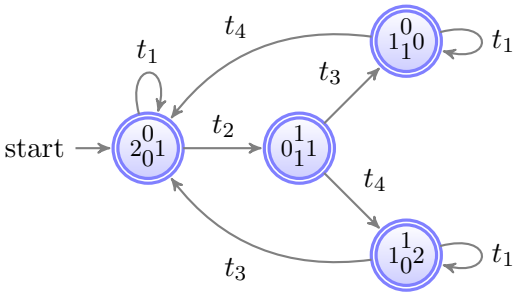
(a) Fassen Sie die Bezeichnungen der Transitionen als Alphabet auf und die Markierungen (also die jeweiligen Belegungen der Stellen mit Marken) als Zustände. Beschreiben Sie die möglichen Reihenfolgen, in der die Transitionen feuern und die Markierungen auftreten können, mit Hilfe eines endlichen Automaten. Der Automat soll also Wörter wie $t_2t_3t_1$ akzeptieren, weil die Transitionen in dieser Reihenfolge feuern können, nicht aber t_3t_1.

(b) Warum lassen sich nicht für alle Petri-Netze die Abläufe durch einen endlichen Automaten beschreiben? Geben Sie ein Petri-Netz an, bei dem das nicht geht.

Lösung

(a) Die Abläufe im Petrinetz lassen sich durch einen Automaten mit (in diesem Fall) endlich vielen Zuständen beschreiben. Die Markierungen bilden die Zustände, die Transitionen das Alphabet. Erhält man aus einer Markierung m durch Feuern einer Transition t eine Markierung m', dann gibt es einen Übergang beschriftet mit t vom Zustand für m zu jenem für m'.

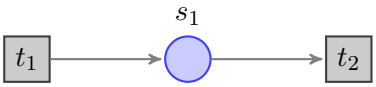
Wir stellen jede Markierung durch vier Zahlen $n_1{}^{n_2}_{n_3}n_4$ dar, wobei n_i die Anzahl der Marken in der Stelle s_i angibt. Die endlichen Reihenfolgen, in denen die Transitionen feuern können, entsprechen der Sprache des folgenden Automaten.



(b) Im Allgemeinen ist die Gesamtzahl der Marken in einem Petri-Netz und damit der Markierungen (also der Verteilung der Marken auf Stellen) nicht begrenzt. Ein einfaches Beispiel dafür ist das folgende Petri-Netz:

Da die Transition t_1 keine Vorbedingung hat, kann sie beliebig oft feuern, dadurch kann die Anzahl der Marken in der Stelle s_1 beliebig groß werden. Allerdings lässt sich die Menge der möglichen Transitionsfolgen, t_1^*, immer noch durch einen endlichen Automaten darstellen:

Die Situation ändert sich, sobald wir eine zweite Transition hinzufügen:



Nun kann zwischendurch auch t_2 immer wieder feuern, allerdings immer nur so oft, wie zuvor t_1 gefeuert hat. Dazu müsste sich ein Automat merken, um wieviel öfter t_1 als t_2 gefeuert hat. Diese Differenz kann beliebig groß werden, die Anzahl der Zustände lässt sich daher nicht beschränken. Die Transitionsfolgen können also nicht mit einem endlichen Automaten beschrieben werden.

Aufgabe 12 (3 Punkte)

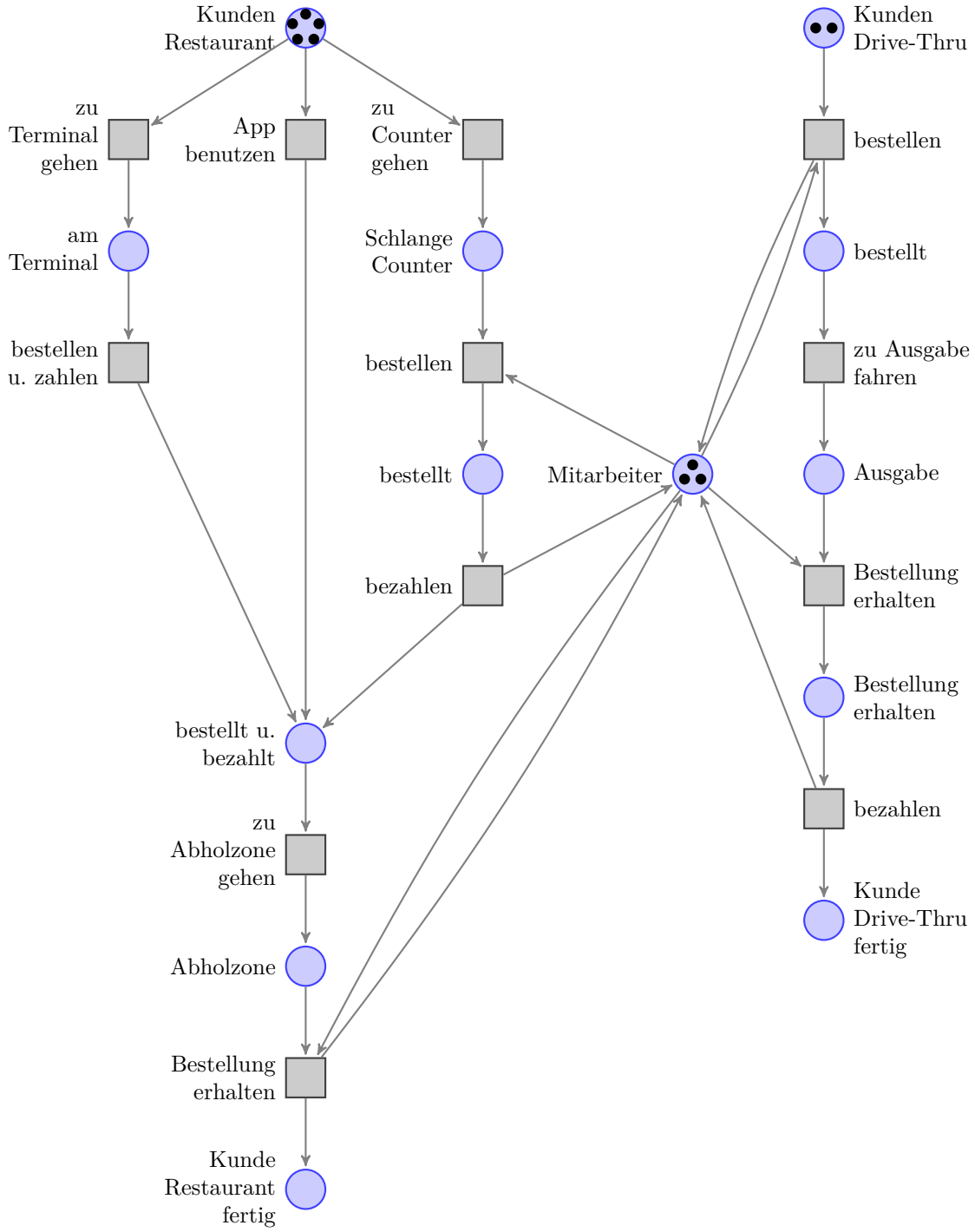
Eine Fastfood-Kette bietet folgende Möglichkeiten um zu bestellen.

- Will man sein Auto nicht verlassen, fährt man zum *Drive-Thru* und gibt dort über eine Gegensprechanlage seine Bestellung bei einem Mitarbeiter auf. Anschließend fährt man zur Ausgabe, wo ein Mitarbeiter zuerst den offenen Betrag kassiert und anschließend die Bestellung aushändigt.

- Im Restaurant selbst gibt es drei weitere Möglichkeiten:
  - Wählt man die klassische Art zu bestellen, geht man zum *Counter* und gibt dort die Bestellung bei einem Mitarbeiter auf. Hat man bezahlt, so erhält man eine Abholnummer und wartet anschließend in der Abholzone auf die fertige Bestellung.
  - Alternativ geht man zu einem *Bestellterminal*, wählt dort am Bildschirm die gewünschten Speisen aus und zahlt anschließend mit Bankomatkarte. Dann erhält man eine Abholnummer und wartet anschließend in der Abholzone auf die fertige Bestellung.
  - Oder man bestellt und bezahlt via *Handy-App*. Auch in diesem Fall erhält man eine Abholnummer für die Abholzone.

Modellieren Sie dieses Bestellkonzept mit Hilfe eines Petri-Netzes mit folgenden Rahmenbedingungen: Es gibt drei Mitarbeiter, die flexibel beim Drive-Thru, am Counter und in der Abholzone eingesetzt werden. Es gibt einen Counter und ein Bestellterminal. Zu Beginn befinden sich zwei Autos am Drive-Thru und fünf Kunden im Eingangsbereich des Restaurants, die sich noch für eine der drei Möglichkeiten, im Restaurant zu bestellen, entscheiden müssen.
Hinweis: Überlegen Sie, wann ein Mitarbeiter in einen Prozess involviert ist und bilden Sie in Ihrem Netz ab, wann ein Mitarbeiter „belegt“ ist.

Lösung


Kunden Restaurant
zu Terminal gehen
App benutzen
zu Counter gehen
am Terminal
Schlange Counter
bestellen u. zahlen
bestellen
bestellt
bezahlen
Mitarbeiter
bestellt u. bezahlt
zu Abholzone gehen
Abholzone
Bestellung erhalten
Kunde Restaurant fertig
Kunden Drive-Thru
bestellen
bestellt
zu Ausgabe fahren
Ausgabe
Bestellung erhalten
Bestellung erhalten
bezahlen
Kunde Drive-Thru fertig