Prof. J. Esparza
S. Sickert, J. Krämer
Vielen Dank an Jan Wagener für die erweiterten Aufgabenlösungen

Technische Universität München
KEINE ABGABE

Einführung in die theoretische Informatik

Sommersemester 2017 – Übungsblatt 4

Übungsblatt

Wir unterscheiden zwischen Übungs- und Abgabeblättern. Auf diesem *Übungsblatt* finden Sie eine Übersicht über die Kernaspekte, die Sie in Kalenderwoche 21 in den Tutorien diskutieren, üben und vertiefen. Die Aufgaben auf diesem Blatt dienen dem Üben und Verstehen des Vorlesungsstoffes, sowie dem *eigenständigen Erarbeiten* der Kernaspekte. Außerdem sollen Ihnen diese Aufgaben auch helfen, ein Gefühl dafür zu bekommen, was Sie inhaltlich in der Klausur erwartet. Klausuraufgaben können jedoch deutlich von den hier gestellten Aufgaben abweichen. Abschreiben und Auswendiglernen von Lösungen wird Ihnen daher keinen dauerhaften Erfolg in der Vorlesung bringen. Fragen zu den Übungsblättern können Sie montags bis donnerstags von 12 Uhr bis 14 Uhr in der *THEO-Sprechstunde* in Raum 03.11.034 stellen.

Kernaspekte

K4.1 korrektes Wiedergeben der folgenden Definitionen

- Wortproblem
- Leerheitsproblem
- Endlichkeitsproblem
- Äquivalenzproblem

K4.2 korrektes Wiedergeben des Pumping Lemma
K4.3 korrektes Wiedergeben des Ardens Lemma
K4.4 mithilfe des Pumping Lemmas beweisen, dass eine gegebene Sprache nicht regulär ist
K4.5 beweisen, dass eine gegebene Sprache regulär ist
K4.6 mithilfe von Homomorphismen auf Sprachen beweisen, dass eine gegebene Sprache (nicht) regulär ist.
K4.7 das Wortproblem/Leerheitsproblem/Endlichkeitsproblem/Äquivalenzproblem für einen gegebenen NFA/DFA mithilfe des entsprechenden Algorithmus aus der Vorlesung entscheiden
K4.8 das Wortproblem/Leerheitsproblem/Endlichkeitsproblem/Äquivalenzproblem für andere Kodierungen regulärer Sprachen durch passende Reduktion auf NFAs/DFAs entscheiden
K4.9 beurteilen, welche Kodierung zur Entscheidung des Wortproblem/Leerheitsproblem/Endlichkeitsproblem/Äquivalenzproblem am effizientesten ist
K4.10 zu einem gegebenen NFA N einen regulären Ausdruck r mithilfe eines Gleichungssystems (Ardens Lemma) angeben, so dass $\mathsf{L}(\mathsf{N}) = \mathsf{L}(\mathsf{r})$
K4.11 begründet entscheiden, ob gegebene Beispiele neu eingeführte Definitionen erfüllen
K4.12 Aussagen, mit neu eingeführten Definitionen beweisen oder widerlegen

AUFGABE 4.1. (*Produktkonstruktion*) B+C

Sei Σ ein Alphabet und $\mathsf{A}, \mathsf{B}, \mathsf{C} \subseteq \Sigma^*$. Wir definieren den ITE-Operator (*if-then-else*) wie folgt:

$$\mathtt{ITE}(\mathsf{A}, \mathsf{B}, \mathsf{C}) = \{w \in \Sigma^* \mid (w \in \mathsf{A} \wedge w \in \mathsf{B}) \vee (w \notin \mathsf{A} \wedge w \in \mathsf{C})\}$$

Seien jetzt $\mathsf{A}, \mathsf{B}, \mathsf{C} \subseteq \Sigma^*$ reguläre Sprachen, die von den DFAs $\mathsf{D_A}$, $\mathsf{D_B}$ und $\mathsf{D_C}$ erkannt werden.

(a) Erweitern Sie die Produktkonstruktion aus der Vorlesung, so dass aus den Automaten $\mathsf{D_A}$, $\mathsf{D_B}$ und $\mathsf{D_C}$ ein DFA $\mathsf{D}_{\mathtt{ITE}(\mathsf{A},\mathsf{B},\mathsf{C})}$ mit $\mathsf{L}(\mathsf{D}_{\mathtt{ITE}(\mathsf{A},\mathsf{B},\mathsf{C})}) = \mathtt{ITE}(\mathsf{A}, \mathsf{B}, \mathsf{C})$ konstruiert wird und beweisen Sie die Korrektheit Ihrer Konstruktion!

(b) Sei nun $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ und $\mathsf{A} = \{w \in \Sigma^* \mid |w| \text{ ist gerade}\}$, $\mathsf{B} = \{\mathsf{a}^n\mathsf{b}^m \mid n, m \geq 0\}$ und $\mathsf{C} = \{\mathsf{c}^n\mathsf{b}^m \mid n, m \geq 0\}$. Geben Sie einen DFA D für $\mathtt{ITE}(\mathsf{A}, \mathsf{B}, \mathsf{C})$, indem Sie Ihre Konstruktion aus Aufgabenteil (a) verwenden.

Hinweis: Verwenden Sie $\mathsf{Q}_{\mathtt{ITE}(\mathsf{A},\mathsf{B},\mathsf{C})} = \mathsf{Q_A} \times \mathsf{Q_B} \times \mathsf{Q_C}$ als Menge aller Zustände für die erweiterte Produktkonstruktion in Aufgabenteil (a).

Erweiterte Lösungsskizze
Seien Σ ein Alphabet und
$A, B, C \subseteq \Sigma^*$ reguläre Sprachen.

Seien außerdem D_A, D_B, D_C DFAs, die die jeweiligen Sprachen akzeptieren.
Wir definieren:

$$\underbrace{ITE(A,B,C)}_{\substack{\text{Ein Wort} \\ \text{in der Sprache} \\ ITE(A,B,C) \text{ liegt..}}} := \{w \mid \underbrace{(w \in A \wedge w \in B)}_{\substack{\text{..sowohl in } A \\ \text{als auch in } B \\ \text{oder..}}} \vee \underbrace{(w \notin A \wedge w \in C)}_{\substack{\text{..nicht in } A \text{ aber} \\ \text{in } C.}}\}$$

(a) Konstruiere einen DFA $D_{ITE(A,B,C)}$ mit $L(D_{ITE(A,B,C)}) = ITE(A,B,C)$.

Idee: Erweitere die Produktkonstruktion auf drei statt zwei Automaten, dann passe die Endzustände so an, dass der Automat $D_{ITE(A,B,C)}$ genau dann akzeptiert, wenn ein Wort sich in D_A **und** D_B in einem Endzustand befindet **oder** in D_A in keinem Endzustand **und** in D_C schon in einem.

Konstruktion: Die drei Automaten D_A, D_B und D_C sind nach Definition Fünftupel der Form

$$D_A = (Q_A,\ \Sigma,\ \delta_A,\ {q_0}_A,\ F_A)$$
$$D_B = (Q_B,\ \Sigma,\ \delta_B,\ {q_0}_B,\ F_B)$$
$$D_C = (Q_C,\ \Sigma,\ \delta_C,\ {q_0}_C,\ F_C)$$

Und auch der zu konstruierende Automat ist von der Form

$$D_{ITE(A,B,C)} = (Q,\ \Sigma,\ \delta,\ q_0,\ F)$$

Analog zur Produktkonstruktion zweier Automaten wählen wir

$$Q = Q_A \times Q_B \times Q_C$$

$\rightsquigarrow$ Wie bei der Produktkonstruktion zweier Automaten wird hier in einem Zustand von $D_{ITE(A,B,C)}$ gespeichert, in welchen Zuständen man sich in den Automaten D_A, D_B und D_C gerade befindet.
Die Zustände sind also 3-Tupel von Zuständen aus diesen Automaten.

$$q_0 = ({q_0}_A,\ {q_0}_B,\ {q_0}_C)$$

$$F = (F_A \cap F_B) \cup (\overline{F_A} \cap F_C)$$

$\rightsquigarrow$ Nach Definition von $ITE(A,B,C)$

$$\delta : \underbrace{\delta\Big((q_A, q_B, q_C)\ ,\ a\Big)}_{\substack{\text{Durch einlesen eines a's} \\ \text{gelangt man im Automaten..}}} = \Big(\ \underbrace{\delta_A(q_A, a)}_{\substack{..D_A \text{ von } q_A \text{ nach} \\ \delta_A(q_A,a); \\ \text{im Automaten..}}}\ ,\ \underbrace{\delta_B(q_B, a)}_{\substack{..D_B \text{ von } q_B \text{ nach} \\ \delta_B(q_B,a); \\ \text{im Automaten..}}}\ ,\ \underbrace{\delta_C(q_C, a)}_{\substack{..D_C \text{ von } q_C \text{ nach} \\ \delta_C(q_C,a).}}\ \Big)$$

Korrektheitsbeweis: Gefordert war, dass

$$L(D_{ITE(A,B,C)}) = ITE(A,B,C)$$

Diese (Mengen-)Gleichheit ist also noch zu zeigen.
Dafür muss gezeigt werden, dass $L(D_{ITE(A,B,C)}) \subseteq ITE(A,B,C)$
und dass $L(D_{ITE(A,B,C)}) \supseteq ITE(A,B,C)$

I: $L(D_{ITE(A,B,C)}) \subseteq ITE(A,B,C)$: $\left(\begin{smallmatrix}\text{Wenn } w \in L(D_{ITE(A,B,C)}),\\ \text{dann auch } w \in ITE(A,B,C)\end{smallmatrix}\right)$

Sei $w \in L(D_{ITE(A,B,C)})$

$\underset{\text{Akzeptanzbed. von DFAs}}{\Longrightarrow}$ Es existiert ein akzeptierender Lauf

$$(q_{0\,A}, q_{0\,B}, q_{0\,C}) \xrightarrow{w_1} \dots \xrightarrow{w_n} (q_{f_1\,A}, q_{f_2\,B}, q_{f_3\,C}) \in F$$

des Wortes w im DFA $D_{ITE(A,B,C)}$.

$\underset{\text{Def. } D_{ITE(A,B,C)}}{\Longrightarrow}$ In den Automaten D_A, D_B und D_C

existieren die Läufe

$\rightsquigarrow$ Genauso wurde ja der DFA $D_{ITE(A,B,C)}$ definiert.

$$\begin{aligned} &q_{0\,A} \xrightarrow{w_1} \dots \xrightarrow{w_n} q_{f_1\,A}\\ &q_{0\,B} \xrightarrow{w_1} \dots \xrightarrow{w_n} q_{f_2\,B}\\ &q_{0\,C} \xrightarrow{w_1} \dots \xrightarrow{w_n} q_{f_3\,C}\end{aligned}$$

wobei, da ja $F = (F_A \cap F_B) \cup (\overline{F_A} \cap F_C)$,
gelten muss, dass $q_{f_1\,A} \in F_A$ und $q_{f_2\,B} \in F_B$ **oder**
dass $q_{f_1\,A} \in \overline{F_A}$ und $q_{f_3\,C} \in F_C$.

$\underset{\text{Akzeptanzbed. von DFAs}}{\Longrightarrow}$ $w \in L(D_A)$ und $w \in L(D_B)$ **oder**
$w \notin L(D_A)$ und $w \in L(D_C)$.

$\Longrightarrow$ $w \in \Big((L(D_A) \cap L(D_B)) \cup (\overline{L(D_A)} \cap L(D_C)) \Big)$

$\underset{\text{Definitionen von } D_A, D_B \text{ und } D_C}{\Longrightarrow}$ $w \in \Big((A \cap B) \cup (\overline{A} \cap C) \Big)$

$\Longrightarrow$ $w \in \{ w \in \Sigma^* \mid (w \in \mathsf{A} \wedge w \in \mathsf{B}) \vee (w \notin \mathsf{A} \wedge w \in \mathsf{C}) \}$

$\underset{\text{Def. } ITE(A,B,C)}{\Longrightarrow}$ $w \in ITE(A,B,C)$

II: $L(D_{ITE(A,B,C)}) \supseteq ITE(A,B,C)$: $\left(\begin{smallmatrix}\text{Wenn } w \in ITE(A,B,C),\\ \text{dann auch } w \in L(D_{ITE(A,B,C)})\end{smallmatrix}\right)$

Dreht man alle Implikationen im Beweis für '$\subseteq$' um,
so erhält man den Beweis für '$\supseteq$'.
Achtung! Das ist nicht immer möglich!

(b) Seien nun $\Sigma = \{a, b, c\}$ und

$$A = \{ w \in \Sigma^* \,\big|\, |w| \,\%\, 2 = 0 \}$$

$\rightsquigarrow$ Sprache aller Wörter gerader Länge

$$B = \{ a^n b^m \mid n, m \geq 0 \}$$

$\rightsquigarrow \triangleq L(a^* b^*)$

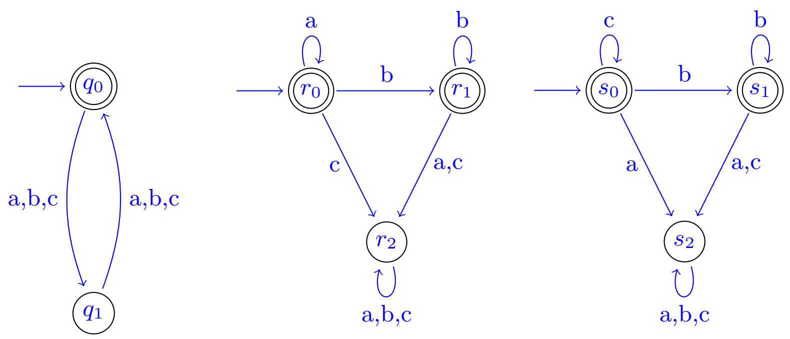
$$C = \{ c^n b^m \mid n, m \geq 0 \}$$

$\rightsquigarrow \triangleq L(c^* b^*)$

Gib mittels der Konstruktion aus (a) einen DFA für die Sprache ITE(A,B,C) an.

DFAs für gegebene Sprachen:

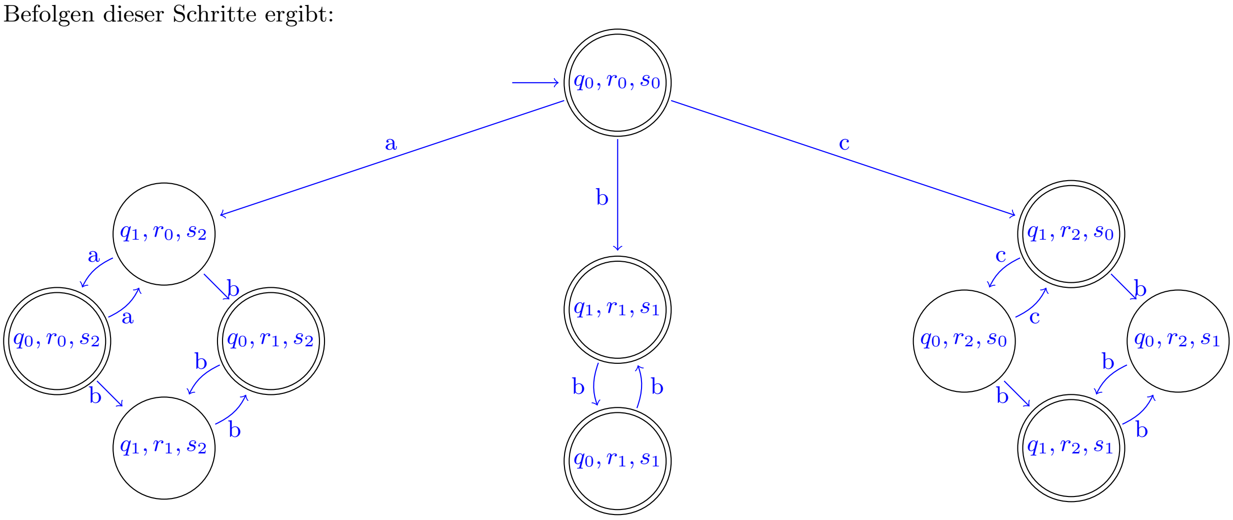
D_A: D_B: D_C:



Vorgehen: (Analog zur bereits bekannten Produktkonstruktion mit zwei Automaten.)

- Beginne mit (q_0, r_0, s_0) als Startzustand.
- Füge nun für jedes Zeichen in Σ eine Transition entsprechend der Konstruktion in (a) hinzu.
- Falls der Zielzustand noch nicht eingezeichnet ist, zeichne ihn ein.
- Wiederhole solange, bis alle bisher gezeichneten Zustände für jedes Zeichen in Σ eine ausgehende Transition haben.
- Markiere Endzustände nach vorgegebenen Regeln.

Befolgen dieser Schritte ergibt:



In dieser Grafik wurden, um die Übersichtlichkeit zu wahren, Fangzustände (Zustände, von denen aus man keinen Endzustand erreichen kann) nicht eingezeichnet. Alle nicht vorhandenen Transitionen gehen also implizit in einen Fangzustand.
In diesem Fall wären die Fangzustände (q_0, r_2, s_2) und (q_1, r_2, s_2).
Die noch übrigen 5 Zustände (z.B. (q_0, r_0, s_1)) sind nicht erreichbar.

AUFGABE 4.2. (*Pumping Lemma*) C

Beweisen Sie für jede der folgenden Sprachen mithilfe des Pumping Lemmas, dass sie *nicht* regulär sind.

(a) $\mathsf{L}_1 = \{w \in \{0,1\}^* \mid w = w^R\}$
(b) $\mathsf{L}_2 = \{w \in \{0,1\}^* \mid |w|_0 \geq |w|_1\}$
(c) $\mathsf{L}_3 = \{\varepsilon, \mathsf{a}, \mathsf{a}^{n \cdot m} \mid m \in \mathsf{N}, n \in \mathsf{N}, m > 1, n > 1\}$
(d) $\mathsf{L}_4 = \{\mathsf{a}^{6i}\mathsf{b}^{6i} \mid i \geq 0\}$
(e) $\mathsf{L}_5 = \{\mathsf{a}^{2^i} \mid i \geq 0\}$

Erweiterte Lösungsskizze
Widerlege Regularität der gegebenen Sprachen mithilfe des Pumping-Lemmas.

Sei $L \subseteq \Sigma^*$ eine reguläre Sprache.
Für jede reguläre Sprache gilt das 'Pumping-Lemma für reguläre Sprachen', welches folgendes besagt:

> **Es existiert** eine sog. Pumping-Lemma-Zahl $n \in \mathbb{N}$, sodass
> **für alle** Wörter $z \in L$ mit $|z| \geq n$
> **eine** Zerlegung **existiert**, die von der Form $z = uvw$ mit $u,\ v,\ w \in \Sigma^*$ ist
> und folgende Bedingungen erfüllt:
>
> - $v \neq \varepsilon$
> - $|uv| \leq n$
>
> und sodass **für alle** Zahlen $i \geq 0$ gilt,
> dass auch $uv^iw \in L$.

Will man nun mithilfe des 'Pumping-Lemmas für reguläre Sprachen' die Regularität einer Sprache widerlegen, so ist die Gültigkeit des Pumping-Lemmas zu widerlegen bzw. man nimmt an, dass das Lemma für gegebene Sprache gilt und führt das dann zu einem Widerspruch indem man den Standardbeweis aufschreibt (Sei $n \in \mathbb{N}$ eine Pumping-Lemma-Zahl ...) und dabei nur

- ein eigenes Wort $z \in L$ mit $|z| \geq n$ wählt, mit dem die Gültigkeit des Pumping-Lemmas dann widerlegt werden soll
- eine allgemeine Form für Zerlegungen von z in uvw findet, welche den Anforderungen entsprechen
- eine Zahl $i \geq 0$ findet, sodass $uv^iw \notin L$ (und Letzteres begründet).

(a) $L_1 = \{w \in \{0,1\}^* \mid w = w^R\}$

Angenommen L_1 wäre regulär.
Dann würde das 'Pumping-Lemma für reguläre Sprachen' gelten.

Sei $n \in \mathbb{N}$ eine Pumping-Lemma-Zahl.
Dann ist $z = 0^n 1 0^n \in L_1$ ein Wort in der Sprache mit $|z| \geq n$.
Jede Zerlegung des Wortes z in uvw ,
die die Bedingungen $v \neq \varepsilon$ und $|uv| \leq n$ erfüllt,
ist von der Form:

$\rightsquigarrow$ Bzw. jede mögliche Pumping-Lemma-Zahl ist von der Form $n \in \mathbb{N}$.

$$u = 0^j$$
$$v = 0^k$$
$$w = 0^{n-j-k} 1 0^n$$

$\rightsquigarrow$ Da das Wort z mit n Nullen beginnt und $|uv| \leq n$ sein muss, bestehen sowohl u als auch v ausschließlich aus Nullen.

($j \geq 0$ und $k > 0$ sodass $v \neq \varepsilon$).

Betrachtet man nun allerdings $i = 2$ und $uv^2w = 0^j 0^{2k} 0^{n-j-k} 1 0^n$
$= 0^{n+k} 1 0^n$
so fällt auf, dass $uv^2w \notin L_1$, da $n + k > n$, weil ja $k > 0$.

↯ Widerspruch! ↯

$\Longrightarrow$ Die Annahme, dass L_1 regulär sei, ist falsch!

(b) $L_2 = \{w \in \{0,1\}^* \mid |w|_0 \geq |w|_1\}$

Angenommen L_2 wäre regulär.
Dann würde das 'Pumping-Lemma für reguläre Sprachen' gelten.

Sei $n \in \mathbb{N}$ eine Pumping-Lemma-Zahl.
Dann ist $z = 1^n 0^n \in L_2$ ein Wort in der Sprache mit $|z| \geq n$.
Jede Zerlegung des Wortes z in uvw ,
die die Bedingungen $v \neq \varepsilon$ und $|uv| \leq n$ erfüllt,
ist von der Form:

$$u = 1^j$$
$$v = 1^k$$
$$w = 1^{n-j-k} 0^n$$

$\rightsquigarrow$ Da das Wort z mit n Einsen beginnt und $|uv| \leq n$ sein muss, bestehen sowohl u als auch v ausschließlich aus Einsen.

($j \geq 0$ und $k > 0$ sodass $v \neq \varepsilon$).

Betrachtet man nun allerdings $i = 2$ und $uv^2w = 1^j 1^{2k} 1^{n-j-k} 0^n$
$= 1^{n+k} 0^n$
so fällt auf, dass $uv^2w \notin L_2$, da $n + k > n$, weil ja $k > 0$.

↯ Widerspruch! ↯

$\Longrightarrow$ Die Annahme, dass L_2 regulär sei, ist falsch!

(c) $L_3 = \{a^{n*m} \mid n, m \in \mathbb{N}, n > 1, m > 1\} \cup \{\varepsilon, a\}$

Angenommen L_3 wäre regulär.
Da die regulären Sprachen unter Komplement abgeschlossen sind, wäre dann auch
$L := \overline{L_3} = \{a^n \mid n \in \mathbb{N}, n \text{ prim}\}$
regulär.
Somit würde für L das 'Pumping-Lemma für reguläre Sprachen' gelten.

Sei $n \in \mathbb{N}$ eine Pumping-Lemma-Zahl.
Wähle $p > n+1$ prim. Das ist möglich, $\rightsquigarrow$ +1 später noch wichtig!
da die Menge der Primzahlen bekanntlich unendlich ist.
Also ist $z = a^p \in L$ ein Wort in der Sprache mit $|z| \geq n$.
Jede Zerlegung des Wortes z in uvw ,
die die Bedingungen $v \neq \varepsilon$ und $|uv| \leq n$ erfüllt,
ist von der Form:

$$u = a^j$$
$$v = a^k$$
$$w = a^{p-j-k}$$

$\rightsquigarrow$ Da das Wort z ausschließlich aus a's besteht, bestehen auch die Teilwörter u und v ausschließlich aus a's.

($j \geq 0$ und $k > 0$ sodass $v \neq \varepsilon$).

Betrachtet man nun allerdings $i = p - k$ und
$$uv^{p-k}w = a^j a^{k*(p-k)} a^{p-j-k} = a^{k*(p-k)} a^j a^{p-j-k} = a^{k*(p-k)} a^{p-k} = a^{(k+1)*(p-k)}$$
so fällt auf, dass $(k+1)*(p-k)$ wegen $1 \leq k \leq n < p-1$
also $k+1 > 1$
und $p-k > 1$
keine Primzahl ist.

$\rightsquigarrow$ Es gilt ja schließlich $k = |v| \leq |uv| \leq n$ und $n+1<p$ also $n<p-1$ Daher das +1 vorher!

Folglich ist $uv^{p-k}w \notin L$.

↯ Widerspruch! ↯

$\Longrightarrow$ Die Annahme, dass $L = \overline{L_3}$ regulär sei, ist falsch!
$\Longrightarrow$ Auch die Annahme, dass L_3 regulär sei, ist falsch, da ja sonst auch deren Komplement regulär sein müsste.

(d) $L_4 = \{a^{6i}b^{6i} \mid i \geq 0\}$

Angenommen L_4 wäre regulär.
Dann würde das 'Pumping-Lemma für reguläre Sprachen' gelten.

Sei $n \in \mathbb{N}$ eine Pumping-Lemma-Zahl.
Dann ist $z = a^{6n}b^{6n} \in L_4$ ein Wort in der Sprache mit $|z| \geq n$.
Jede Zerlegung des Wortes z in uvw ,
die die Bedingungen $v \neq \varepsilon$ und $|uv| \leq n$ erfüllt,
ist von der Form:

$$u = a^j$$
$$v = a^k$$
$$w = a^{6n-j-k}b^{6n}$$

$\rightsquigarrow$ Da das Wort z mit $6n$ a's beginnt und $|uv| \leq n$ sein muss, bestehen sowohl u als auch v ausschließlich aus a's.

($j \geq 0$ und $k > 0$ sodass $v \neq \varepsilon$).

Betrachtet man nun allerdings $i = 0$ und $uv^0w = a^j a^0 a^{6n-j-k}b^{6n}$
$= a^{6n-k}b^{6n}$
so fällt auf, dass $uv^0w \notin L_4$, da $6n - k \neq 6n$, weil ja $k \neq 0$.

$\lightning$ Widerspruch! $\lightning$

$\Longrightarrow$ Die Annahme, dass L_4 regulär sei, ist falsch!

(e) $L_5 = \{a^{2^i} \mid i \geq 0\}$

Angenommen L_5 wäre regulär.
Dann würde das 'Pumping-Lemma für reguläre Sprachen' gelten.

Sei $n \in \mathbb{N}$ eine Pumping-Lemma-Zahl.
Dann ist $z = a^{2^n} \in L_5$ ein Wort in der Sprache mit $|z| \geq n$. $\rightsquigarrow$ Schließlich ist $2^n > n$ für alle $n \in \mathbb{N}$.
Jede Zerlegung des Wortes z in uvw ,
die die Bedingungen $v \neq \varepsilon$ und $|uv| \leq n$ erfüllt,
ist von der Form:

$$u = a^j$$
$$v = a^k$$
$$w = a^{2^n-j-k}$$

$\rightsquigarrow$ Da das Wort z ausschließlich aus a's besteht, bestehen auch die Teilwörter u und v ausschließlich aus a's.

($j \geq 0$ und $k > 0$ sodass $v \neq \varepsilon$).

Betrachtet man nun allerdings $i = 2$ und $uv^2w = a^j a^{2k} a^{2^n-j-k}$
$= a^{2^n+k}$
so fällt auf, dass $2^n < 2^n + k < 2^{n+1}$, da $0 < k \leq n < 2^n$.
'$2^n + k$' ist also keine Zweierpotenz,
woraus folgt, dass $uv^2w \notin L_5$.

$\lightning$ Widerspruch! $\lightning$

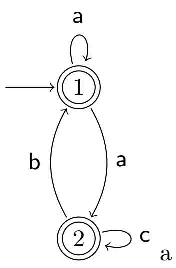
$\Longrightarrow$ Die Annahme, dass L_5 regulär sei, ist falsch!

AUFGABE 4.3. (*Ardens Lemma*) C

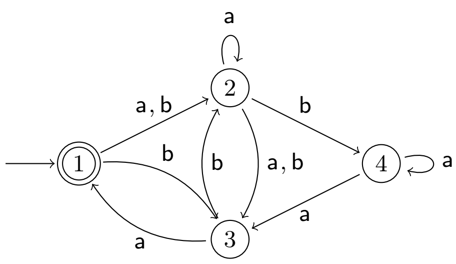
Betrachten Sie die beiden NFAs N_a und N_b.

(a) Geben Sie für beide NFAs das nach Vorlesung dazugehörige Gleichungssystem an.

(b) Geben Sie dann für jeden der beiden NFAs mittels Gauß-Verfahren und Ardens Lemma einen regulären Ausdruck r_a bzw. r_b an, so dass $\mathsf{L}(\mathsf{N}_a) = \mathsf{L}(\mathsf{r}_a)$ bzw. $\mathsf{L}(\mathsf{N}_b) = \mathsf{L}(\mathsf{r}_b)$



(a) NFA N_a



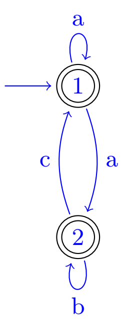
(b) NFA N_b

Erweiterte Lösungsskizze

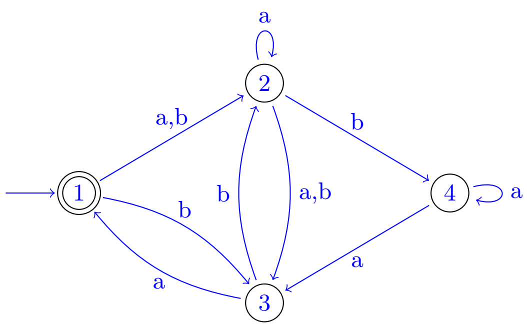
Gegeben die beiden NFAs N_a und N_b .

Konstruiere unter Zuhilfenahme von Ardens Lemma zwei reguläre Ausdrücke für die von den NFAs akzeptierten Sprachen.

N_a :



N_b :



▶ Für den Automaten N_a lautet das Gleichungssystem:

(I) $X_1 \equiv \varepsilon \mid aX_1 \mid aX_2$

(II) $X_2 \equiv \varepsilon \mid bX_1 \mid cX_2$

Dieses lässt sich lösen, wie folgt:

- Umformen von (II): $X_2 \equiv cX_2 \;\Big|\; (\varepsilon \mid bX_1)$
- Mit Ardens Lemma: $X_2 \equiv c^* \;(\varepsilon \mid bX_1)$
- (II) in (I) : $X_1 \equiv \varepsilon \;\Big|\; aX_1 \;\Big|\; a\;(c^*\;(\varepsilon \mid bX_1))$
- Durch Umformen gibt (I) : $X_1 \equiv \varepsilon \;\Big|\; aX_1 \;\Big|\; ac^*bX_1 \mid ac^*$

  $\equiv (a \mid ac^*b)X_1 \;\Big|\; (\varepsilon \mid ac^*)$
- Ardens Lemma liefert: $X_1 \equiv (a \mid ac^*b)^*(\varepsilon \mid ac^*)$

$\Longrightarrow \; L(N_a) = L\Big((a \mid ac^*b)^*(\varepsilon \mid ac^*)\Big)$

▶ Für den Automaten N_b lautet das Gleichungssystem:

(I) $X_1 \equiv \varepsilon \mid (a|b)X_2 \mid bX_3$

(II) $X_2 \equiv aX_2 \mid (a|b)X_3 \mid bX_4$

(III) $X_3 \equiv aX_1 \mid bX_2$

(IV) $X_4 \equiv aX_3 \mid aX_4$

Dieses lässt sich lösen, wie folgt:

- (III) in (I) :
$$\begin{aligned} X_1 &\equiv \varepsilon \,\Big|\, (a|b)X_2 \,\Big|\, b\,(aX_1|bX_2) \\ &\equiv \varepsilon \,\Big|\, aX_2 \mid bX_2 \,\Big|\, baX_1 \mid bbX_2 \\ &\equiv \varepsilon \,\Big|\, baX_1 \,\Big|\, (a|b|bb)X_2 \end{aligned}$$

- (III) in (II) :
$$\begin{aligned} X_2 &\equiv aX_2 \,\Big|\, (a|b)(aX_1|bX_2) \,\Big|\, bX_4 \\ &\equiv aX_2 \,\Big|\, aaX_1 \mid baX_1 \mid abX_2 \mid bbX_2 \,\Big|\, bX_4 \\ &\equiv (aa|ba)X_1 \,\Big|\, (a|ab|bb)X_2 \,\Big|\, bX_4 \end{aligned}$$

- (III) in (IV) :
$$\begin{aligned} X_4 &\equiv a(aX_1|bX_2) \,\Big|\, aX_4 \\ &\equiv aaX_1 \mid abX_2 \,\Big|\, aX_4 \end{aligned}$$

- (IV) mit Ardens Lemma:
$$\begin{aligned} X_4 &\equiv aX_4 \,\Big|\, aaX_1 \mid abX_2 \\ &\equiv a^*(aaX_1 \mid abX_2) \end{aligned}$$

- (IV) in (II) :
$$\begin{aligned} X_2 &\equiv (aa|ba)X_1 \,\Big|\, (a|ab|bb)X_2 \,\Big|\, b(a^*(aaX_1 \mid abX_2)) \\ &\equiv (aa|ba|ba^*aa)X_1 \,\Big|\, (a|ab|bb|ba^*ab)X_2 \\ &\equiv (aa|baa^*)X_1 \,\Big|\, (a|ab|ba^*b)X_2 \end{aligned}$$

- (II) mit Ardens Lemma:
$$\begin{aligned} X_2 &\equiv (a|ab|ba^*b)X_2 \,\Big|\, (aa|baa^*)X_1 \\ &\equiv (a|ab|ba^*b)^*(aa|baa^*)X_1 \end{aligned}$$

- (II) in (I) :
$$\begin{aligned} X_1 &\equiv \varepsilon \,\Big|\, baX_1 \,\Big|\, (a|b|bb)(a|ab|ba^*b)^*(aa|baa^*)X_1 \\ &\equiv \varepsilon \,\Big|\, \Big(ba \mid (a|b|bb)(a|ab|ba^*b)^*(aa|baa^*)\Big)X_1 \end{aligned}$$

- (I) mit Ardens Lemma:
$$\begin{aligned} X_1 &\equiv \Big(ba \mid (a|b|bb)(a|ab|ba^*b)^*(aa|baa^*)\Big)X_1 \,\Big|\, \varepsilon \\ &\equiv \Big(ba \mid (a|b|bb)(a|ab|ba^*b)^*(aa|baa^*)\Big)^*\varepsilon \\ &\equiv \Big(ba \mid (a|b|bb)(a|ab|ba^*b)^*(aa|baa^*)\Big)^* \end{aligned}$$

$$\implies L(N_b) = L\left(\Big(ba \mid (a|b|bb)(a|ab|ba^*b)^*(aa|baa^*)\Big)^*\right)$$

AUFGABE 4.4. (*Umkehrung(Spiegelung)*) D

Ziel dieser Aufgabe ist es Algorithmen anzugeben, die es erlauben, die Umkehrung einer Sprache zu berechnen, d.h. für jede reguläre Sprache L die Sprache L^R anzugeben, so dass $\mathsf{L}^R := \{w^R \mid w \in \mathsf{L}\}$.

(a) Sei D ein DFA. Geben Sie einen Algorithmus an, der den DFA D in einen ε-NFA N übersetzt, so dass $\mathsf{L}(\mathsf{D})^R = \mathsf{L}(\mathsf{N})$. Beweisen Sie auch die Korrektheit Ihres Verfahrens.

(b) Konstruieren Sie für den regulären Ausdruck $\mathsf{r} = 0 \mid 1(0|1)^*0$ einen DFA und wenden Sie Ihr Spiegelungsverfahren aus Aufgabenteil (a) an.

(c) Geben Sie einen regulären Ausdruck r' mit $\mathsf{L}(\mathsf{r}) = \mathsf{L}(\mathsf{r}')^R$ an.

(d) Geben Sie nun ein Verfahren an, welches rekursiv einen regulären Ausdruck r in einen regulären Ausdruck r' umschreibt, so dass $\mathsf{L}(\mathsf{r})^R = \mathsf{L}(\mathsf{r}')$ gilt. Zeigen Sie die Korrektheit Ihres Verfahrens mittels struktureller Induktion und seien Sie besonders ausführlich in den Fällen Konkatenation, Vereinigung und Stern.

Erweiterte Lösungsskizze
Seien Σ ein Alphabet und
$L \subseteq \Sigma^*$ eine reguläre Sprache.

$$L^R = \{w^R \mid w \in L\}$$

(a) Gegeben einen $D = (Q, \Sigma, \delta, q_0, F)$.
Konstruiere einen ε-NFA N' mit $L(N') = L(D)^R$.

Idee: Drehe Läufe im Automaten D um, d.h. beginne bei einem, der Endzustände (nicht-deterministisch über ε-Transitionen wählbar) und drehe im Automaten alle Transitionen um. Der ehemalige Startzustand ist einziger neuer Endzustand.

Konstruktion: Definiere also

$$N' = (Q', \Sigma, \delta', q_0', F')$$

Um den Anforderungen zu entsprechen, wählen wir

$$Q' = Q \cup \{q_0'\}$$

$\leadsto$ Die Menge der Zustände bleibt gleich bis auf den neuen Startzustand, von dem aus man jeden Endzustand allein mit ε-Transitionen erreichen kann.

$$F' = \{q_0\}$$

$\leadsto$ Einziger neuer Endzustand ist der alte Startzustand.

δ' :

$$\delta'(q_0', \varepsilon) = F$$

$$\underbrace{\delta'(q, a) \ni p}_{\substack{\text{Es existiert eine}\\ \text{Transition von } q\\ \text{nach } p \text{ mit } a}} \text{, falls zuvor } \underbrace{\delta(p, a) = q}_{\substack{\text{Von } p \text{ kommt man}\\ \text{durch einlesen}\\ \text{eines } a\text{'s nach } q}}$$

$\leadsto \hat{=}$ Umdrehen aller Transitionen

Korrektheitsbeweis: Gefordert war, dass

$$L(N') = L(D)^R$$

Diese (Mengen-)Gleichheit ist also noch zu zeigen.
Insbesondere heißt das, dass folgende beide Aussagen auf Korrektheit überprüft werden müssen:

$L(N') \subseteq L(D)^R$ (D.h. wenn $w \in L(N')$, dann auch $w \in L(D)^R$.)
$L(N') \supseteq L(D)^R$ (D.h. wenn $w \in L(D)^R$, dann auch $w \in L(N')$.)

I: $L(N') \subseteq L(D)^R$:

Sei $w \in L(N')$

$\underset{\text{Akzeptanzbed. von}\varepsilon\text{-NFAs}}{\Longrightarrow}$ Es existiert ein akzeptierender Lauf

$$q_0' \xrightarrow{\varepsilon} q_f \in F \xrightarrow{w_1} \dots \xrightarrow{w_n} q_0 \in F'$$

$\leadsto$ Jeder Lauf eines Wortes in N' beginnt bei q_0', da dies der Startzustand ist und muss dann erst einmal eine ε-Transition nehmen. Danach kann keine weitere ε-Kante genommen werden, da diese nur vom Startzustand aus gehen. Enden muss der Lauf in q_0, da dies der einzige Endzustand ist.

des Wortes w im ε-NFA N' .

$\underset{\text{Def. N'}}{\Longrightarrow}$ Im DFA D existiert der Lauf

$$q_0 \xrightarrow{w_n} \dots \xrightarrow{w_1} q_f \in F,$$

$\leadsto$ Der ε-NFA N' ist durch umdrehen aller Transitionen in D entstanden. Dabei wurden nur ε-Transitionen hinzugefügt, d.h. alle mit anderen Zeichen beschrifteten Transitionen in N' haben bereits (umgedreht) in D existiert.

der damit akzeptierender Lauf des Wortes w^R in D ist.

$\underset{\text{Akzeptanzbed. von DFAs}}{\Longrightarrow}$ $w^R \in L(D)$

$\underset{\text{Def. der Umkehrung einer Sprache}}{\Longrightarrow}$ $(w^R)^R \in L(D)^R$

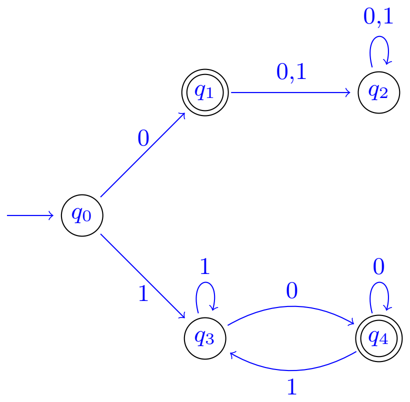
$\underset{(w^R)^R = w}{\Longrightarrow}$ $w \in L(D)^R$

II: $L(N') \supseteq L(D)^R$:

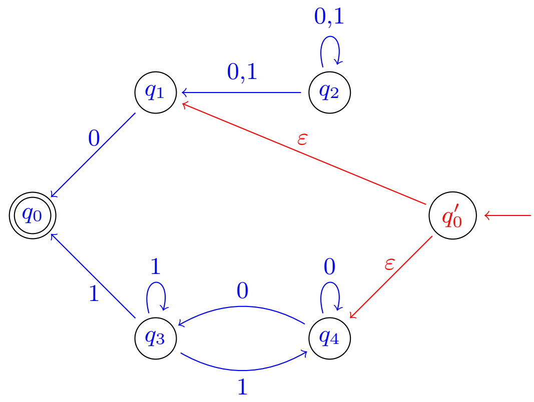
Auch hier erhält man den Beweis für '$\supseteq$' durch umdrehen aller Implikationen im Beweis für '$\subseteq$'.

(b) Sei nun $r = 0 \mid 1\,(0|1)^*\,0$.
Gib einen DFA für r an und wende das in (a) erstellte Spiegelungsverfahren darauf an.

D



N'



(c) Gib für r aus der vorherigen Teilaufgabe einen regulären Ausdruck r' an,
sodass $L(r') = L(r)^R$.

Idee: Betrachte die beiden Teile von r separat:

$$0^R = 0 \text{ und } \Big(\underbrace{1\,(0|1)^*\,0}_{\text{Wörter, die mit 1 beginnen und mit 0 enden}} \Big)^R = \underbrace{0\,(0|1)^*\,1}_{\text{Wörter, die mit 0 beginnen und mit 1 enden}}$$

$$\Longrightarrow \Big(0 \;\Big|\; 1\,(0|1)^*\,0\Big)^R = 0 \;\Big|\; 0\,(0|1)^*\,1$$

(d) Gegeben sei ein regulärer Ausdruck r .
Gib ein Verfahren $\text{rev: RegEx} \to \text{RegEx}$ an, das r in einen regulären Ausdruck r' umwandelt, sodass $L(r') = L(r)^R$.

Vorgehen: Gib für jede der 6 Grundformen eines regulären Ausdrucks eine entsprechende Transformation an und beweise deren Korrektheit.

Für die Korrektheitsbeweise werden folgende Mengengleichungen benötigt:
Für zwei Sprachen A, B gilt

$$(AB)^R = B^R A^R \qquad (1)$$
$$(A \cup B)^R = A^R \cup B^R \qquad (2)$$
$$(A^*)^R = (A^R)^* \qquad (3)$$

Beweis:

$$\begin{aligned}
w \in (AB)^R &\iff w^R \in AB \\
&\iff \exists u \in A, v \in B.\ uv = w^R \\
&\iff \exists u \in A, v \in B.\ (uv)^R = w \\
&\text{Da } (uv)^R = (u_1 ... u_m v_1 ... v_n)^R = v_n ... v_1 u_m ... u_1 = v^R u^R \\
&\iff \exists u \in A, v \in B.\ v^R u^R = w \\
&\iff \exists u^R \in A^R, v^R \in B^R . v^R u^R = w \\
&\iff w \in B^R A^R
\end{aligned}$$

$$
\begin{aligned}
w \in (A \cup B)^R &\iff w^R \in A \cup B\\
&\iff w^R \in A \vee w^R \in B\\
&\iff w \in A^R \vee w \in B^R\\
&\iff w \in A^R \cup B^R
\end{aligned}
$$

$$
\begin{aligned}
w \in (A^*)^R &\iff w^R \in A*\\
&\iff \exists u \in A, i \geq 0.\ w^R = u^i\\
&\iff \exists u \in A, i \geq 0.\ w = (u^i)^R
\end{aligned}
$$

$$\text{Da } (u^i)^R = \underbrace{u_1...u_m \;.......\; u_1...u_m}_{\substack{i \text{ Wiederholungen}\\ \text{des Wortes } u}} = u_m...u_1 \;.......\; u_m...u_1 = (u^R)^i$$

$$
\begin{aligned}
&\iff \exists u \in A, i \geq 0.\ w = (u^R)^i\\
&\iff \exists u^R \in A^R, i \geq 0.\ w = (u^R)^i\\
&\iff w \in (A^R)^*
\end{aligned}
$$

Umwandlung: $\textbf{rev}(r) = r'$	**Korrektheitsbeweis mittels struktureller Induktion:** $L(r') = L(r)^R$
$\mathrm{rev}(\emptyset) = \emptyset$	$L(r') = L(\emptyset) = \emptyset = \emptyset^R = L(\emptyset)^R = L(r)^R$
$\mathrm{rev}(\varepsilon) = \varepsilon$	$L(r') = L(\varepsilon) = \{\varepsilon\} = \{\varepsilon\}^R = L(\varepsilon)^R = L(r)^R$
$\mathrm{rev}(a) = a$	$L(r') = L(a) = \{a\} = \{a\}^R = L(a)^R = L(r)^R$
$\mathrm{rev}(\alpha\beta) = \mathrm{rev}(\beta)\ \mathrm{rev}(\alpha)$	$L(r') = L(\mathrm{rev}(\beta)\ \mathrm{rev}(\alpha)) = L(\mathrm{rev}(\beta))L(\mathrm{rev}(\alpha)) \overset{\text{IH}}{=} L(\beta)^R L(\alpha)^R \overset{(1)}{=} \big(L(\alpha)L(\beta)\big)^R = L(\alpha\beta)^R = L(r)^R$
$\mathrm{rev}(\alpha \mid \beta) = \mathrm{rev}(\alpha) \mid \mathrm{rev}(\beta)$	$L(r') = L(\mathrm{rev}(\alpha) \mid \mathrm{rev}(\beta)) = L(\mathrm{rev}(\alpha)) \cup L(\mathrm{rev}(\beta)) \overset{\text{IH}}{=} L(\alpha)^R \cup L(\beta)^R \overset{(2)}{=} \big(L(\alpha) \cup L(\beta)\big)^R = L(\alpha \mid \beta)^R = L(r)^R$
$\mathrm{rev}(\alpha^*) = \mathrm{rev}(\alpha)^*$	$L(r') = L(\mathrm{rev}(\alpha)^*) \overset{\text{IH}}{=} \big(L(\alpha)^R\big)^* \overset{(3)}{=} \big(L(\alpha)^*\big)^R = L(\alpha^*)^R = L(r)^R$

AUFGABE 4.5. (*Leere Sprachen*) D

Sie haben in der Vorlesung gesehen, wie man entscheiden kann, ob ein endlicher Automat die leere Sprache akzeptiert. Wir entwickeln jetzt ein Verfahren, das direkt und ohne Umwege über endliche Automaten prüft, ob ein regulärer Ausdruck die leere Sprache beschreibt.
Definieren Sie hierzu eine rekursive Funktion $\mathit{iszero} : \mathsf{RE} \mapsto \mathbb{B}$ über den regulären Ausdrücken und beweisen Sie, dass für alle Ausdrücke r gilt:

$$\mathit{iszero}(\mathsf{r}) \Leftrightarrow \mathsf{L}(\mathsf{r}) = \emptyset$$

Erweiterte Lösungsskizze
Gegeben sei ein regulärer Ausdruck r .
Gib ein Verfahren iszero: RegEx $\to \{true, false\}$ an, das sich zu $true$ auswertet, gdw. r die leere Sprache beschreibt:

$$iszero(r) \iff L(r) = \emptyset$$

Vorgehen: Gib für jede der 6 Grundformen eines regulären Ausdrucks eine entsprechende Konstruktion an und beweise deren Korrektheit.

Für die Korrektheitsbeweise werden folgende Implikationen benötigt:
Für zwei Sprachen A, B gilt

$$AB = \emptyset \iff A = \emptyset \lor B = \emptyset \qquad (1)$$
$$A \cup B = \emptyset \iff A = \emptyset \land B = \emptyset \qquad (2)$$

Das ist offensichtlich, da ein Wort w nur dann in AB enthalten sein kann, wenn es ein entsprechendes Wort $u \in A$ und ein Wort $v \in B$ gibt, sodass $w = uv$ die Konkatenation der beiden ist. Ist eine der Sprachen A oder B leer, so gibt es in dieser keine Worte zum konkatenieren.
Die Vereinigung zweier Mengen ist allerdings nur dann leer, wenn beide Mengen leer sind.

Konstruktion: **iszero**(r)

Korrektheitsbeweis mittels struktureller Induktion: **iszero**$(r) \iff L(r) = \emptyset$

$\text{iszero}(\emptyset) \quad = \quad \mathit{true}$

$$\begin{aligned} \text{iszero}(r) &\iff \text{iszero}(\emptyset) \\ &\iff \mathit{true} \\ &\overset{\substack{L(\emptyset)=\emptyset \text{ ist} \\ \text{immer wahr}}}{\iff} L(\emptyset) = \emptyset \\ &\iff L(r) = \emptyset \end{aligned}$$

$\text{iszero}(\varepsilon) \quad = \quad \mathit{false}$

$$\begin{aligned} \text{iszero}(r) &\iff \text{iszero}(\varepsilon) \\ &\iff \mathit{false} \\ &\overset{\substack{\{\varepsilon\}=\emptyset \text{ ist} \\ \text{immer falsch}}}{\iff} L(\varepsilon) = \{\varepsilon\} = \emptyset \\ &\iff L(r) = \emptyset \end{aligned}$$

$\text{iszero}(a) \quad = \quad \mathit{false}$

$$\begin{aligned} \text{iszero}(r) &\iff \text{iszero}(a) \\ &\iff \mathit{false} \\ &\overset{\substack{\{a\}=\emptyset \text{ ist} \\ \text{immer falsch}}}{\iff} L(a) = \{a\} = \emptyset \\ &\iff L(r) = \emptyset \end{aligned}$$

$\text{iszero}(\alpha\beta) = \text{iszero}(\alpha) \vee \text{iszero}(\beta)$

$$\begin{aligned} \text{iszero}(r) &\iff \text{iszero}(\alpha\beta) \\ &\iff \text{iszero}(\alpha) \vee \text{iszero}(\beta) \\ &\overset{\text{IH}}{\iff} L(\alpha) = \emptyset \vee L(\beta) = \emptyset \\ &\overset{(1)}{\iff} L(\alpha)L(\beta) = \emptyset \\ &\overset{\text{Def. Konkat.}}{\iff} L(\alpha\beta) = \emptyset \\ &\iff L(r) = \emptyset \end{aligned}$$

$\text{iszero}(\alpha \mid \beta) = \text{iszero}(\alpha) \wedge \text{iszero}(\beta)$

$$\begin{aligned} \text{iszero}(r) &\iff \text{iszero}(\alpha \mid \beta) \\ &\iff \text{iszero}(\alpha) \wedge \text{iszero}(\beta) \\ &\overset{\text{IH}}{\iff} L(\alpha) = \emptyset \wedge L(\beta) = \emptyset \\ &\overset{(2)}{\iff} L(\alpha) \cup L(\beta) = \emptyset \\ &\overset{\text{Def. Altern.}}{\iff} L(\alpha \mid \beta) = \emptyset \\ &\iff L(r) = \emptyset \end{aligned}$$

$\text{iszero}(\alpha^*) = \mathit{false}$

$$\begin{aligned} \text{iszero}(r) &\iff \text{iszero}(\alpha^*) \\ &\iff \mathit{false} \\ &\overset{\substack{L(\alpha^*)=\emptyset \text{ ist} \\ \text{immer falsch,} \\ \text{da } L(\alpha^*) \\ \text{stets das Wort} \\ \varepsilon \text{ enthält}}}{\iff} L(\alpha^*) = \{\varepsilon\} \cup L(\alpha^+) = \emptyset \\ &\iff L(r) = \emptyset \end{aligned}$$

AUFGABE 4.6. (*Homomorphismen auf regulären Sprachen*) E / C

Sei Σ ein Alphabet und $\mathsf{L} \subseteq \Sigma^*$ eine reguläre Sprache. Weiter sei $h\colon \Sigma \to \Sigma^*$ eine Abbildung, die jedem Zeichen $\mathsf{x} \in \Sigma$ ein Wort $h(\mathsf{x}) \in \Sigma^*$ zuordnet. h erweitert man kanonisch auf Σ^* mittels $h(\varepsilon) = \varepsilon$ und $h(wv) = h(w)h(v)$ für alle $w, v \in \Sigma^*$. Schließlich sei $h(\mathsf{L}) = \{h(w) \mid w \in \mathsf{L}\}$.

(a) Zeigen Sie: Ist L regulär, dann ist auch $h(\mathsf{L})$ regulär.

(b) Zeigen Sie unter Verwendung des Resultats aus Aufgabenteil (a), dass $\mathsf{L}' = \{\mathsf{a}\mathsf{b}^{3i}\mathsf{c}\mathsf{d}^{2i}\mathsf{e} \mid i \in \mathbb{N}_0\}$ nicht regulär ist.