Prof. Dr. J. Giesl

M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Mi., 28.04.2010 im Tutorium abgegeben werden. Alternativ ist es bis 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet.

Tutoraufgabe 1 (Erzeugendensysteme / Monoide):

Beweisen Sie: $(L((ab)^*), \cdot)$ ist ein Monoid. Hierbei bezeichnet $\cdot$ die Aneinanderkettung von Wörtern.
Hinweis: In dieser Vorlesung bezeichnet $\mathbb{N}$ die natürlichen Zahlen inklusive der 0.

Lösung:

Die Aneinanderkettung $\cdot$ ist assoziativ. Somit bleibt noch zu zeigen, dass $L((ab)^*)$ abgeschlossen unter $\cdot$ ist und das neutrale Element enthält. Seien $m, n \in \mathbb{N}$, $u = (ab)^n \in L((ab)^*)$ und $v = (ab)^m \in L((ab)^*)$. Dann gilt $u \cdot v = (ab)^{n+m} \in L((ab)^*)$. Da auch $\varepsilon = (ab)^0 \in L((ab)^*)$, ist $(L((ab)^*), \cdot)$ ein Monoid.

Hausaufgabe 2 (Erzeugendensysteme / Monoide): (2 + 4 = 6 Punkte)

a) Beweisen Sie: $(L((ab)^*), \cdot)$ ist isomorph zu $(\mathbb{N}, +)$.

b)
- Hat das Monoid $(\mathbb{N} \setminus \{0\}, *)$ ein endliches Erzeugendensystem?
- Wir nennen ein Erzeugendensystem E für ein Monoid M *minimal*, falls keine echte Teilmenge von E ein Erzeugendensystem für M ist.
  Geben Sie ein minimales Erzeugendensystem von $(\mathbb{N} \setminus \{0\}, *)$ an. Ist Ihr Erzeugendensystem frei?

Begründen Sie ihre Antworten!

Lösung:

a) Zu zeigen: Es gibt einen bijektiven Homomorphismus von $(L((ab)^*), \cdot)$ nach $(\mathbb{N}, +)$.

Sei $h : L((ab)^*) \to \mathbb{N}$ mit $h((ab)^n) = n$, dann ist h ein Homomorphismus, denn: $h((ab)^n \cdot (ab)^m) = h((ab)^{n+m}) = n+m = h((ab)^n) + h((ab)^m)$. Die Funktion h ist bijektiv, da für jedes $n \in \mathbb{N}$ gilt $h((ab)^n) = n$ und somit die Umkehrfunktion $h^{-1} : \mathbb{N} \to L((ab)^*)$ mit $h^{-1}(n) = (ab)^n$ definiert ist.

b)
- Nein. Jedes Erzeugendensystem muss mindestens alle Primzahlen enthalten und kann daher nicht endlich sein.

- Die Menge der Primzahlen bildet ein minimales Erzeugendensystem, da jede Zahl als Multiplikation ihrer Primfaktoren dargestellt werden kann. Dies ist allerdings kein freies Erzeugendensystem, da die Multiplikation kommutativ ist. So lässt sich 6 als $2 * 3$, aber auch als $3 * 2$ darstellen.

Tutoraufgabe 3 (Homomorphismen / Isomorphismen):

Sei Σ ein beliebiges, endliches Alphabet. Beweisen oder widerlegen Sie die Existenz folgender Homomorphismen:

a) Ein Isomorphismus von $(\mathbb{R}, +)$ nach $(\mathbb{Z}, +)$.

b) Ein Isomorphismus von $(\mathbb{Z}, +)$ nach $(\mathbb{Z}, +)$, der nicht die Identität ist.

Lösung:

a) Da $\mathbb{R}$ überabzählbar unendlich ist und $\mathbb{Z}$ nur abzählbar unendlich ist, kann es keine bijektive Abbildung und damit auch keinen Isomorphismus zwischen den beiden geben.

b) Sei $h(x) = -x$. Die Funktion h ist offensichtlich eine Bijektion und es gilt $h(a + b) = -(a + b) = (-a) + (-b) = h(a) + h(b)$. Somit ist h ein Isomorphismus.

Hausaufgabe 4 (Homomorphismen / Isomorphismen): (1 + 1 + 1 + 2 = 5 Punkte)

Sei Σ ein beliebiges, endliches Alphabet. Beweisen oder widerlegen Sie die Existenz folgender Homomorphismen:

a) Ein Homomorphismus von $(\mathbb{N}, +)$ nach $(\{1\}, *)$.

b) Ein Homomorphismus von $(\Sigma^*, \cdot)$ nach $(\mathbb{N}, +)$.

c) Ein Isomorphismus von $(\{1\}, *)$ nach $(\mathbb{N}, +)$.

d) Ein injektiver Homomorphismus von $(\mathbb{N}, +)$ nach $(\mathbb{N}, *)$.

Lösung:

a) Sei $h : x \mapsto 1$, so ist h ein Homomorphismus, da $h(a + b) = 1 = 1 * 1 = h(a) * h(b)$.

b) Sei $h : x \mapsto |x|$, so ist h ein Homomorphismus, da $h(a \cdot b) = |a \cdot b| = |a| + |b| = h(a) + h(b)$. Hierbei bezeichnet $|w|$ die Länge des Wortes w.

Alternativ: Betrachte $h' : x \mapsto 0$, also die Funktion, die jedes Element von Σ^* auf das neutrale Element des Monoids $(\mathbb{N}, +)$ abbildet. Offensichtlich gilt hier die Homomorphiebedingung: $h'(a \cdot b) = 0 = 0 + 0 = h'(a) + h'(b)$.

Es kann also durchaus unterschiedliche Homomorphismen zwischen zwei Monoiden geben.

c) Es kann keinen Isomorphismus von $\{1\}$ nach $\mathbb{N}$ geben, da $|\{1\}| = 1 < |\mathbb{N}|$.

d) Sei $h : x \mapsto 2^x$. Dann ist h ein Homomorphismus, denn $h(a + b) = 2^{a+b} = 2^a * 2^b = h(a) * h(b)$ für alle $a, b \in \mathbb{N}$. Weiterhin ist $h : \mathbb{N} \to \mathbb{N}$ streng monoton steigend und somit injektiv.

Tutoraufgabe 5 (Reguläre Ausdrücke):

Sei $\Sigma = \{0, 1, \ldots, 9, -\}$ ein Alphabet, mit dem ganze Zahlen beschrieben werden können.

a) Geben Sie für die folgende Sprache über diesem Alphabet einen regulären Ausdruck an. Ganze Zahlen beginnen mit höchstens einem $-$, d.h. die Menge der ganzen Zahlen ist $\{0, 1, -1, 2, -2, \ldots\}$.

Hinweis: Außer 0 beginnt keine ganze Zahl mit der Ziffer 0.

$L_1 = \{w \mid w \text{ ist eine positive ganze Zahl}\}$, d.h. $L_1 = \{1, 2, 3, \ldots\}$.

b) Wir schränken uns nun auf das Alphabet $\Sigma' = \Sigma \setminus \{-\} = \{0, 1, \ldots, 9\}$ ein. Geben Sie für die folgende Sprache einen regulären Ausdruck an, der genau die Wörter über dem Alphabet Σ' beschreibt, die **nicht** in dieser Sprache liegen. Achten Sie darauf, dass $\Sigma' \setminus L$ auch Wörter wie 000 enthalten kann, die keine ganzen Zahlen sind.

$L_2 = \{w \mid w \text{ ist eine gerade natürliche Zahl}\}$, d.h. $L_2 = \{0, 2, 4, 6, \ldots\}$.

Lösung:

a)
- $r_1 = (1 + 2 + \ldots + 9)(0 + 1 + \ldots + 9)^*$

b)
- $r_2 = \varepsilon + 0(\Sigma')^+ + r_1^*(1 + 3 + 5 + 7 + 9)$
  Wie üblich steht Σ' in diesem regulären Ausdruck für die Disjunktion der einzelnen Symbole in der Menge, in diesem Falle also $0 + 1 + \ldots + 9$. Außerdem ist r^+ eine Abkürzung für rr^*.
  Der reguläre Ausdruck deckt drei verschiedene Fälle ab:
  - ε lässt das leere Wort zu, das keine ganze Zahl (und damit auch insbesondere nicht gerade) ist.
  - $0(\Sigma')^+$ erlaubt all die Wörter, die keine ganze Zahl sind, weil sie mit einer 0 beginnen und nicht die 0 sind.
  - $r_1^*(1 + 3 + 5 + 7 + 9)$ erlaubt all die Wörter, die mit einer ganzen Zahl beginnen und deren letzte Ziffer ungerade ist.
    Hier gibt es eine Überschneidung mit dem vorherigen Fall, denn r_1 erlaubt auch das Wort 0. Den Ausdruck r_1 können wir wiederverwenden, weil er das nicht mehr betrachtete Symbol $-$ nicht beinhaltet.

Hausaufgabe 6 (Reguläre Ausdrücke): (2 + 4 = 6 Punkte)

Seien Σ, Σ' die Alphabete aus Tutoraufgabe 5.

a) Geben Sie für die folgenden Sprachen über dem Alphabet Σ wie in Tutoraufgabe 5a) je einen regulären Ausdruck an.

- $L_3 = \{w \mid w \text{ ist eine nicht-positive ganze Zahl}\}$, d.h. $L_3 = \{0, -1, -2, -3, \ldots\}$.
- $L_4 = \{w \mid w \text{ ist eine durch 5 teilbare ganze Zahl}\}$, d.h. $L_4 = \{0, 5, -5, 10, -10, \ldots\}$.
  Hinweis: Für zwei ganze Zahlen $x, y \in \mathbb{Z}$ gilt, dass x durch y teilbar ist genau dann, wenn es ein $z \in \mathbb{Z}$ gibt, so dass $x = y * z$.

b) Geben Sie für die folgenden Sprachen wie in Tutoraufgabe 5b) jeweils einen regulären Ausdruck an, der genau die Wörter über dem Alphabet Σ' beschreibt, die **nicht** in dieser Sprache liegen.

- $L_5 = \{w \mid w \text{ enthält die Ziffer 7}\}$, d.h. $L_5 = \{7, 17, 27, \ldots, 70, \ldots\}$.
- $L_6 = \{w \mid w \text{ enthält die Ziffer 7 genau einmal}\}$, d.h. $L_6 = \{7, 17, 27, \ldots, 70, \ldots, 76, 78, 79, \ldots\}$.

Lösung:

a)
- $r_3 = 0 + -r_1$
- $r_4 = 0 + 5 + -5 + (\varepsilon + -)r_1(0 + 5)$

b)
- Wir definieren $\Sigma'_{\neq 7} := \Sigma' \setminus \{7\}$.
  Dann ist $r_5 = (\Sigma'_{\neq 7})^*$.
- $r_6 = r_5 + (\Sigma')^*7(\Sigma')^*7(\Sigma')^*$
  Hier werden wieder zwei Fälle unterschieden:
  - r_5 erlaubt all die Wörter, die die Ziffer 7 nicht beinhalten.
  - $(\Sigma')^*7(\Sigma')^*7(\Sigma')^*$ erlaubt die Wörter, die die Ziffer 7 mindestens zwei mal enthalten.

Hausaufgabe 7 (Reguläre Ausdrücke): (4 Punkte)

Sei $\Sigma = \{A, \ldots, Z, 0, \ldots, 9, \_, .\}$ ein Alphabet.
Wir definieren die folgenden regulären Ausdrücke

- $r := (A + B + ... + Z)$
- $s := (0 + 1 + ... + 9)$

Damit beschreibt der Ausdruck rs^* also alle Wörter, die aus genau einem Buchstaben und beliebig vielen darauf folgenden Ziffern bestehen.
Wir wollen nun "Wörter" der folgenden Form betrachten:

- $422365\_HAT\_1.7\_PUNKTE$
- $023426\_HAS\_4711\_POINTS$
- $652342\_BEKAM\_13.14\_POINTS$

Dies sind also die Wörter, in denen auf eine (6-stellige) Matrikelnummer ein Verb und dann eine Punktzahl folgt, die beliebige viele Stellen haben kann. Am Ende dieser Wörter steht immer "$PUNKTE$" oder "$POINTS$". Alle diese Teile sind jeweils durch ein "_" getrennt.
Als Punktzahl lassen wir ganze Zahlen wie in Tutoraufgabe 5 zu, erlauben zusätzlich aber auch, dass auf eine ganze Zahl ein "." und dann beliebig viele, aber mindestens eine Ziffer als Nachkommastellen folgen. Das heißt, dass wir .05 und 0.0.0 nicht als Punktzahl zulassen.
Wir wollen der Einfachheit halber die verwendeten Verben nicht näher spezifizieren und wollen daher jede (nicht-leere) Buchstabenfolge als Verb behandeln.
Geben Sie nun einen regulären Ausdruck an, der die Sprache dieser Wörter beschreibt.

Lösung:

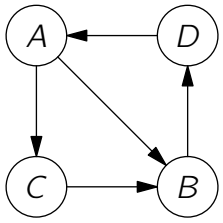
Wir definieren die folgenden regulären Ausdrücke:

$$\begin{aligned} r_{\text{Matrikelnummer}} &:= s^6 \\ r_{\text{Verb}} &:= r^+ \\ r_{\text{Zahl}} &:= (0 + (1 + 2 + 3 + \ldots + 9)s^*)(\varepsilon + .s^+) \end{aligned}$$

Hier beschreibt s^n den regulären Ausdruck $\underbrace{s \ldots s}_{n\text{-mal}}$.
Diese können wir nun zum Ausdruck $r_{\text{Matrikelnummer}}\_r_{\text{Verb}}\_r_{\text{Zahl}}\_(PUNKTE + POINTS)$ zusammensetzen.

Tutoraufgabe 8 (Reguläre Ausdrücke):

Gegeben sei folgender Graph:



Sei L die Sprache der Pfade gerade Länge, die von A nach A führen. Es gilt also beispielsweise $ACBDA \in L$, $ABDABDA \in L$ aber $ABDA \notin L$
Geben Sie einen regulären Ausdruck an, der diese Sprache beschreibt.

Lösung:

Wir erkennen, dass Pfade von A nach A immer aus den Teilpfaden $ABDA$ und $ACBDA$ zusammengesetzt sind. Ersterer hat ungerade Länge, daher muss immer eine gerade Anzahl solcher Stücke in einem Pfad gerader Länge enthalten sein. Wir erhalten den folgenden regulären Ausdruck:

$$r = A((BDA(CBDA)^*BDA) + (CBDA)^*)^*$$

Prof. Dr. J. Giesl

M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Mi., 05.05.2010 im Tutorium abgegeben werden. Alternativ ist es bis 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet.

Tutoraufgabe 1 (Abschlusseigenschaften):

Sei $\Sigma := \{a, b\}$ ein Alphabet. Sei $L := \{a^n b^{2n} \mid n \geq 0\}$ eine Sprache über Σ. Hierbei bezeichnet a^n das Wort $\underbrace{a \ldots a}_{n\text{-mal}}$. Es existiert kein DFA, der L akzeptiert. Zeigen Sie, dass dann auch kein DFA existiert, der die folgenden Sprachen über Σ akzeptiert.

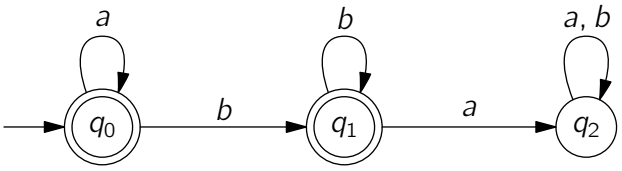
a) $L_1 = \{w \mid w = a^n b^m \text{ für } m, n \geq 0 \text{ und } m \neq 2n\} \cup \Sigma^* ba \Sigma^*$

b) $L_2 = \{a^n b^n a^m b^n \mid n \geq m \geq 0\}$

Lösung:

a) Es gilt $L = \Sigma^* \setminus L_1$. Da kein DFA existiert, der L akzeptiert, folgt aus der Abgeschlossenheit unter Komplement von Sprachen, die von einem DFA akzeptiert werden können, dass L_1 nicht von einem DFA akzeptiert werden kann.

b) Es gilt $L = L_2 \cap L'$ mit $L' = \{a^m b^n \mid m, n \geq 0\}$. Der folgende DFA akzeptiert L'.



Da kein DFA existiert, der L akzeptiert, folgt aus der Abgeschlossenheit unter Schnitt von Sprachen, die von einem DFA akzeptiert werden können, dass L_2 nicht von einem DFA akzeptiert werden kann.

Hausaufgabe 2 (Abschlusseigenschaften): (4 Punkte)

Sei Σ das Alphabet aus der vorherigen Aufgabe. Sei $L_3 := \{a^n b^n \mid n \geq 0\}$ eine Sprache über Σ. Es existiert kein DFA, der L_3 akzeptiert. Zeigen Sie, dass dann auch kein DFA existiert, der die Sprache $L_4 := \{a^m b^n \mid m, n \geq 0, m \neq n\}$ über Σ akzeptiert.

Hinweis: Benutzen Sie die Abschlusseigenschaften für Sprachen, die von einem DFA akzeptiert werden können.

Lösung:

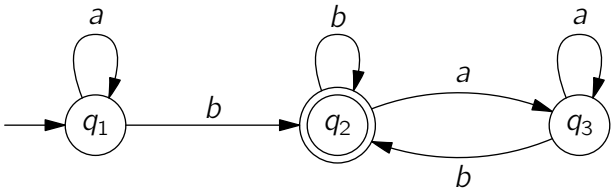
Es gilt $L_3 = (\Sigma^* \setminus L_4) \cap L'$. Da es einen DFA gibt, der L' akzeptiert, es aber keinen DFA gibt, der L_3 akzeptiert, kann aufgrund der Abgeschlossenheit unter Schnitt und Komplement von Sprachen, die von einem DFA akzeptiert werden können, die Sprache L_4 nicht von einem DFA akzeptiert werden.

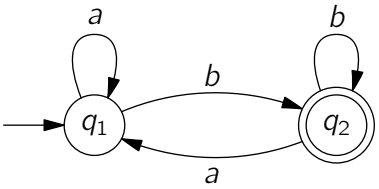
Tutoraufgabe 3 (DFAs aus regulären Ausdrücken):

Sei Σ ein Alphabet. Konstruieren Sie einen Automaten, der genau die Sprache $L(a^*bb^*(aa^*bb^*)^*)$ erkennt.

Lösung:

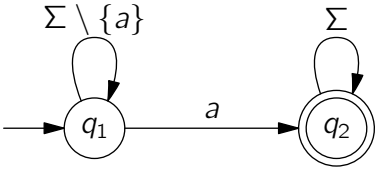
Zwei Lösungen, wobei die rechte minimal ist.





Tutoraufgabe 4 (DFAs aus Wörtern):

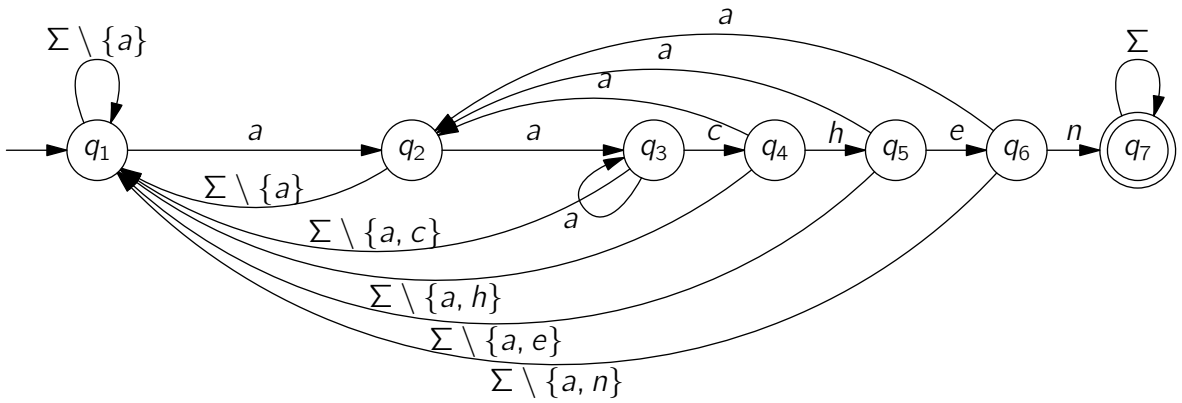
Sei $\Sigma := \{a, \ldots, z\}$ ein Alphabet. Wir führen als verkürzende Schreibweise ein, dass Kanten in einem Automaten mit Mengen (z.B. $\Sigma \setminus \{a\}$) beschriftet werden können und meinen damit, dass die beschriftete Kante einem Übergang entspricht, der für alle Symbole aus der Menge erlaubt ist. Betrachten Sie dazu das folgende Beispiel:



In diesem Automaten ist also ein Übergang von Zustand q_1 zu Zustand q_2 nur mit dem Symbol a möglich. Alle weiteren Buchstaben führen von Zustand q_1 wieder zu sich selbst zurück. Der Automat erkennt also die Sprache $\Sigma^* a \Sigma^*$.

Konstruieren Sie nun einen Automaten, der die Wörter erkennt, die mindestens einmal das Wort *aachen* enthalten, also die Sprache $\Sigma^* aachen \Sigma^*$.

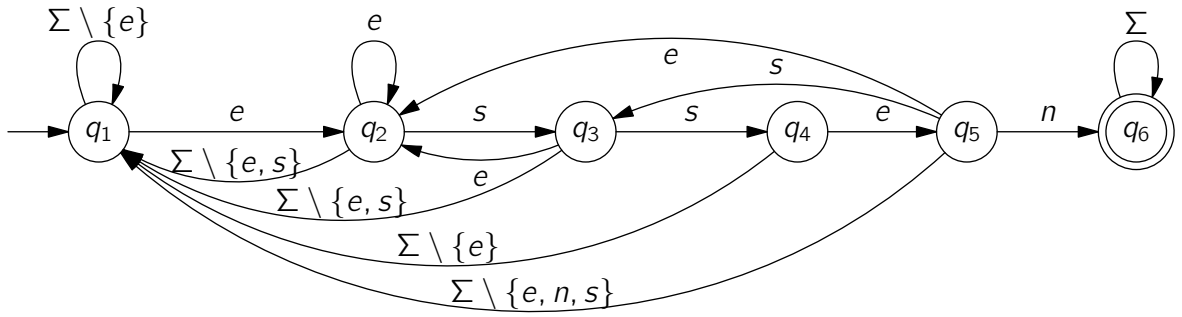
Lösung:



Hausaufgabe 5 (DFAs aus Wörtern): (3 Punkte)

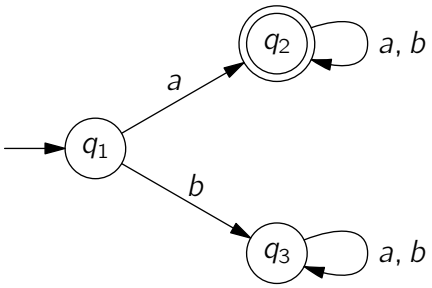
Sei Σ ein Alphabet. Konstruieren Sie einen Automaten, der die Wörter erkennt, die mindestens einmal das Wort *essen* enthalten, also die Sprache $\Sigma^* essen \Sigma^*$.
Hinweis: Um die Darstellung des Automaten zu vereinfachen, nutzen Sie am besten die verkürzende Schreibweise, die in Tutoraufgabe 4 eingeführt wurde.

Lösung:



Tutoraufgabe 6 (L_{ij}^k-Algorithmus):

Bestimmen Sie mit Hilfe des L_{ij}^k-Algorithmus einen regulären Ausdruck, der die Sprache erkennt, die der folgende endliche Automat akzeptiert.



Lösung:

Da es nur einen Endzustand q_3 gibt, reicht es, $L_{1,2}^3$ zu bestimmen.
$L_{1,2}^3 = L_{1,2}^2 + L_{1,3}^2(L_{3,3}^2)^* L_{3,2}^2 = \cdots$

$L_{1,2}^2 = L_{1,2}^1 + L_{1,2}^1(L_{2,2}^1)^* L_{2,2}^1 = \cdots$

$L_{1,2}^1 = L_{1,2}^0 + L_{1,1}^0(L_{1,1}^0)^* L_{1,2}^0 = \cdots$

$L_{1,2}^0 = a$

$L_{1,1}^0 = \varepsilon$

$\cdots = a + \varepsilon\varepsilon^* a = a$

$L_{2,2}^1 = L_{2,2}^0 + L_{2,1}^0(L_{1,1}^0)^* L_{1,2}^0 = \cdots$

$L_{2,2}^0 = \varepsilon + a + b \; L_{2,1}^0 = \emptyset$

$\cdots = (\varepsilon + a + b) + (\emptyset)(\varepsilon)^*(a) = \varepsilon + a + b$

$\cdots = (a) + (a)(\varepsilon + a + b)^*(\varepsilon + a + b) = a(\varepsilon + a + b)^*$

$L_{1,3}^2 = L_{1,3}^1 + L_{1,2}^1(L_{2,2}^1)^* L_{2,3}^1 = \cdots$

$L_{1,3}^1 = L_{1,3}^0 + L_{1,1}^0(L_{1,1}^0)^* L_{1,3}^0 = \cdots$

$L_{1,3}^0 = b$

$\cdots = b + \varepsilon\varepsilon^* b = b$

$L_{2,3}^1 = L_{2,3}^0 + L_{2,1}^0(L_{1,1}^0)^* L_{1,3}^0 = \cdots$

$L_{2,3}^0 = \emptyset \; L_{2,1}^0 = \emptyset$

$\cdots = \emptyset$

$\cdots = b + a(\varepsilon + a + b)^*\emptyset = b$

$L_{3,3}^2 = L_{3,3}^1 + L_{3,2}^1(L_{2,2}^1)^* L_{2,3}^1 = \cdots$

$L_{3,3}^1 = L_{3,3}^0 + L_{3,1}^0(L_{1,1}^0)^* L_{1,3}^0 = \cdots$

$L_{3,3}^0 = \varepsilon + a + b \; L_{3,1}^0 = \emptyset$

$\cdots = \varepsilon + a + b$

$L_{3,2}^1 = L_{3,2}^0 + L_{3,1}^0(L_{1,1}^0)^* L_{1,2}^0 = \cdots$

$L_{3,2}^0 = \emptyset \; L_{3,1}^0 = \emptyset$

$\cdots = \emptyset$

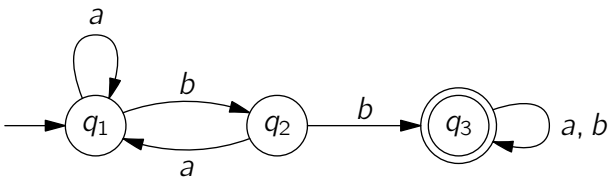
$\cdots = (\varepsilon + a + b) + \emptyset(\varepsilon + a + b)^*\emptyset = \varepsilon + a + b$

$L_{3,2}^2 = L_{3,2}^1 + L_{3,2}^1(L_{2,2}^1)^* L_{2,2}^1 = \emptyset + \emptyset(\varepsilon + a + b)^*(\varepsilon + a + b) = \emptyset$

$\cdots = a(\varepsilon + a + b)^* + b(\varepsilon + a + b)^*\emptyset = a(a + b)^*$

Hausaufgabe 7 (L_{ij}^k-Algorithmus): (4 Punkte)

Bestimmen Sie mit Hilfe des L_{ij}^k-Algorithmus einen regulären Ausdruck, der die Sprache erkennt, die der folgende endliche Automat akzeptiert.



Lösung:

Da es nur einen Endzustand q_3 gibt, reicht es $L_{1,3}^3$ zu bestimmen.
$L_{1,3}^3 = L_{1,3}^2 + L_{1,3}^2(L_{3,3}^2)^*L_{3,3}^2 = \cdots$

$L_{1,3}^2 = L_{1,3}^1 + L_{1,2}^1(L_{2,2}^1)^*L_{2,3}^1 = \cdots$

$L_{1,3}^1 = L_{1,3}^0 + L_{1,1}^0(L_{1,1}^0)^*L_{1,3}^0 = \cdots$

$L_{1,3}^0 = \emptyset$

$L_{1,1}^0 = \varepsilon + a$

$\cdots = \emptyset + (\varepsilon + a)(\varepsilon + a)^*\emptyset = \emptyset$

$L_{1,2}^1 = L_{1,2}^0 + L_{1,1}^0(L_{1,1}^0)^*L_{1,2}^0 = \cdots$

$L_{1,2}^0 = b$

$\cdots = b + (\varepsilon + a)(\varepsilon + a)^*b = a^*b$

$L_{2,2}^1 = L_{2,2}^0 + L_{2,1}^0(L_{1,1}^0)^*L_{1,2}^0 = \cdots$

$L_{2,2}^0 = \varepsilon$

$L_{2,1}^0 = a$

$\cdots = \varepsilon + a(\varepsilon + a)^*b = \varepsilon + a^+b$

$L_{2,3}^1 = L_{2,3}^0 + L_{2,1}^0(L_{1,1}^0)^*L_{1,3}^0 = \cdots$

$L_{2,3}^0 = b$

$\cdots = b + a(\varepsilon + a)^*\emptyset = b$

$\cdots = \emptyset + a^*b(\varepsilon + a^+b)^*b = a^*b(\varepsilon + a^+b)^*b$

$L_{3,3}^2 = L_{3,3}^1 + L_{3,2}^1(L_{2,2}^1)^*L_{2,3}^1 = \cdots$

$L_{3,3}^1 = L_{3,3}^0 + L_{3,1}^0(L_{1,1}^0)^*L_{1,3}^0 = \cdots$

$L_{3,3}^0 = \varepsilon + a + b$

$L_{3,1}^0 = \emptyset$

$\cdots = (\varepsilon + a + b) + \emptyset(\varepsilon + a)^*\emptyset = \varepsilon + a + b$

$L_{3,2}^1 = L_{3,2}^0 + L_{3,1}^0(L_{1,1}^0)^*L_{1,2}^0 = \cdots$

$L_{3,2}^0 = \emptyset$

$\cdots = \emptyset + \emptyset(\varepsilon + a)^*b = \emptyset$

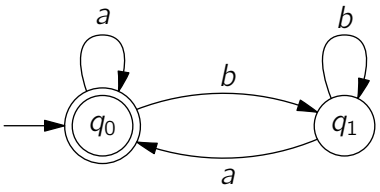
$$\cdots = (\varepsilon + a + b) + (\emptyset)(\varepsilon + a(\varepsilon + a)^* b)^* b = \varepsilon + a + b$$

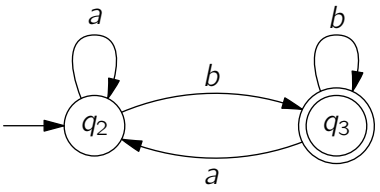
$$\cdots = (a^* b(\varepsilon + a^+ b)^* b) + (a^* b(\varepsilon + a^+ b)^* b)(\varepsilon + a + b)^*(\varepsilon + a + b) = a^* b(\varepsilon + a^+ b)^* b\Sigma^* = (a + b)^* bb(a + b)^*$$

Tutoraufgabe 8 (Produktautomat):

a) In der Vorlesung wurde der Produktautomat $M \times M'$ verwendet, um genau die Wörter zu akzeptieren, die sowohl von M als auch von M' akzeptiert werden. Wie kann man die Konstruktion des Produktautomaten so modifizieren, dass stattdessen die Vereinigung der beiden Sprachen $L(M)$ und $L(M')$ akzeptiert wird?

b) Verwenden Sie die Konstruktion aus der vorherigen Teilaufgabe, um zu zeigen, dass jedes Wort aus Σ^* mit $\Sigma = \{a, b\}$ von einem der folgenden Automaten akzeptiert wird.

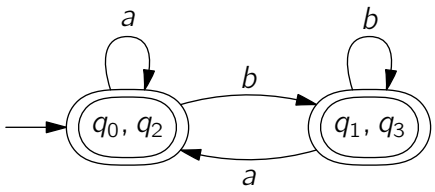




Lösung:

a) Nach wie vor verwendet man das kartesische Produkt der Zustandsmengen beider Ausgangsautomaten als neue Zustandsmenge und benutzt die gleiche Kantenrelation wie beim herkömmlichen Produktautomaten. Als Endzustände definiert man allerdings nicht nur solche Zustände, die Paaren entsprechen, bei denen beide Ausgangsautomaten einen Endzustand haben, sondern auch solche, bei denen nur einer der beiden Ausgangsautomaten einen Endzustand hat.

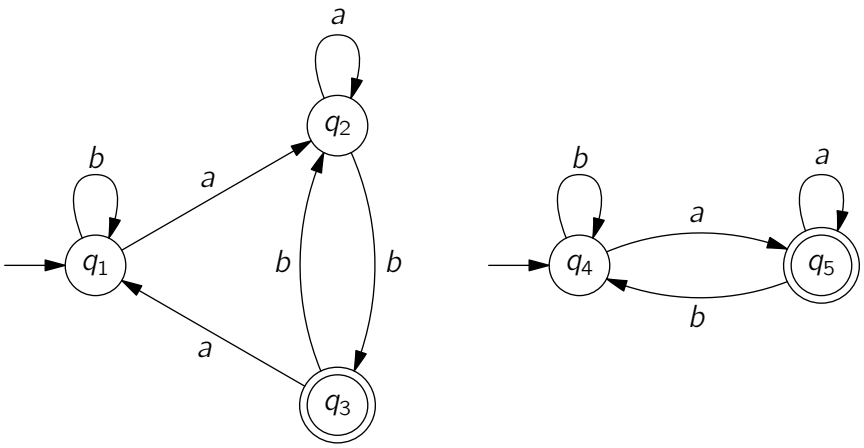
b) Der modifizierte Produktautomat



akzeptiert offensichtlich jedes Wort aus Σ^*, da er nur Endzustände besitzt.

Hausaufgabe 9 (Produktautomat): (3+1 Punkte)

Gegeben seien zwei endliche Automaten.

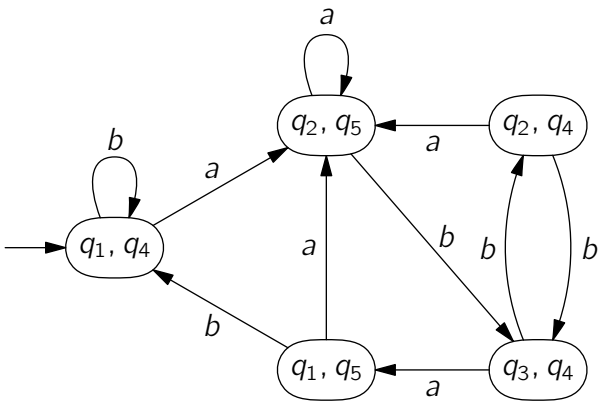


a) Konstruieren Sie den Produktautomaten der beiden Automaten.

b) Beweisen Sie, dass es kein Wort der Sprache $\{a, b\}^*$ gibt, das von beiden Automaten akzeptiert wird.

Lösung:

a)



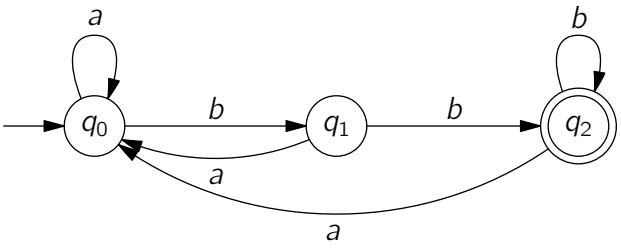
b) Falls es kein Wort gibt, das von beiden Automaten erkannt wird, hat ihr Produkt keinen erreichbaren Endzustand. Dies ist hier der Fall.

Alternative Lösung: Im ersten Automat sind alle eingehenden Kanten des Endzustandes mit b beschriftet und im zweiten Automat mit a. Daher kann es kein Wort geben, das von beiden Automaten erkannt wird.

Hausaufgabe 10 (Verifikation mit Model Checking): (2+4+1 Punkte)

Sei $\Sigma = \{a, b\}$ ein Alphabet. Wir wollen nun ein Verifikations-Problem betrachten und ein Verfahren entwickeln, das automatisch und in endlicher Zeit überprüft, ob jedes Wort, das ein DFA (die "Implementierung") akzeptiert, auch in einer als Sprache gegebenen Spezifikation L liegt.

Sei nun L die Sprache aller Wörter, die mit einer geraden Anzahl von bs enden, d.h. $L := \{w(bb)^n \mid w \in (\Sigma^* a \cup \{\varepsilon\}), n > 0\}$. Als zu überprüfende Implementierung betrachten wir den Automaten M:



a) Konstruieren Sie einen Automaten M', der genau die Sprache L akzeptiert.

b) Geben Sie ein Verfahren an, mit dem automatisch für zwei beliebige, gegebene DFAs M und M' entschieden werden kann, ob jedes von M akzeptierte Wort auch von M' akzeptiert wird. Es soll also ein Algorithmus entwickelt werden, der für eine Eingabe von zwei Automaten M, M' genau dann *true* ausgibt, falls $L(M) \subseteq L(M')$ gilt und sonst *false* ausgibt.

Verwenden Sie (beliebigen) Pseudo-Code, um das Verfahren zu beschreiben.

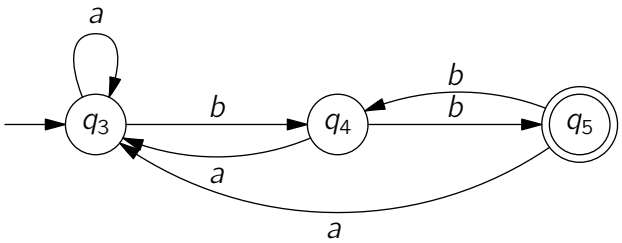
Hinweis: Zur Lösung der Aufgabe hilft es, sich klar zu machen, dass $L(M) \subseteq L(M')$ genau dann gilt, wenn $w \in L(M) \rightarrow w \in L(M')$ für alle $w \in \Sigma^*$ gilt.

Danach sollen Sie die aus der Vorlesung bekannten Verfahren anwenden, mit denen automatisch der **Schnitt** und das **Komplement** von durch DFAs akzeptierten Sprachen ermittelt werden kann. Benutzen Sie auch das Verfahren zum **Leerheitstest** aus Aufgabe 9, mit dem Sie überprüfen können, ob die von einem DFA erkannte Sprache leer ist.

c) Führen Sie das Verfahren exemplarisch durch und geben Sie ein Wort w mit $w \in L(M), w \notin L$ an.

Lösung:

a) Der Automat M':



b) Es gelten die folgenden Äquivalenzen, wenn wir mit $\overline{M}$ und $\overline{M'}$ die Komplementärautomaten von M bzw. M' bezeichnen:

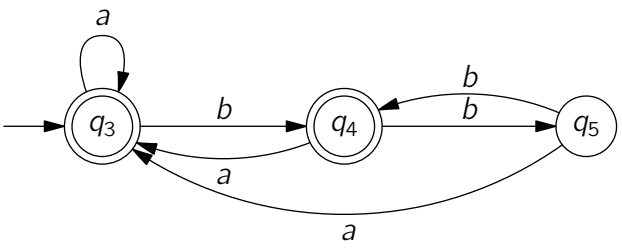
$$
\begin{aligned}
& L(M) \subseteq L(M') && \\
\Leftrightarrow & w \in L(M) \rightarrow w \in L(M') && \forall w \in \Sigma^* \\
\Leftrightarrow & w \notin L(M) \vee w \in L(M') && \forall w \in \Sigma^* \\
\Leftrightarrow & w \in L(\overline{M}) \vee w \in L(M') && \forall w \in \Sigma^* \\
\Leftrightarrow & w \in (L(\overline{M}) \cup L(M')) && \forall w \in \Sigma^* \\
\Leftrightarrow & w \notin (L(M) \cap L(\overline{M'})) && \forall w \in \Sigma^* \\
\Leftrightarrow & w \notin L(M \times \overline{M'}) && \forall w \in \Sigma^*
\end{aligned}
$$

Wir können also $L(M) \subseteq L(M')$ überprüfen, indem wir überprüfen, ob die vom Automaten $M \times \overline{M'}$ erkannte Sprache leer ist. Dies ist genau dann der Fall, wenn es keinen erreichbaren Endzustand gibt.

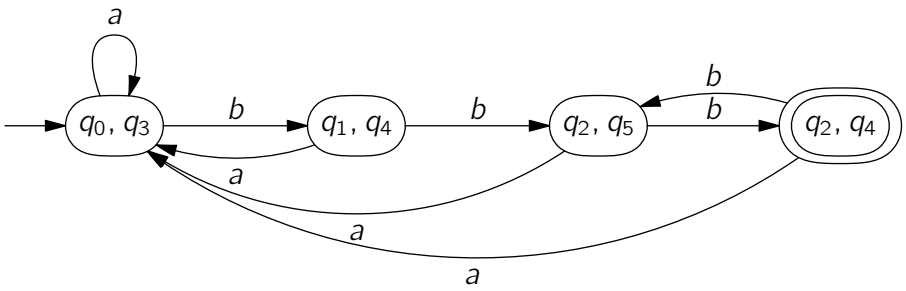
Damit ergibt sich der folgende Pseudocode:

```
langSubset(M, M'):
  CM' = Complement(M')
  PM = ProductDFA(M, CM')
  return PM.hasNoReachableFinalState()
```

c) Zuerst müssen wir den Komplementärautomaten $\overline{M'}$ konstruieren:



Nun können wir den Produktautomaten $M \times \overline{M'}$ konstruieren:



Es gilt z.B. $bbb \in L(M \times \overline{M'})$. Es gilt offensichtlich $w \notin L$ und $w \in L(M)$.

Prof. Dr. J. Giesl

M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Mi., 12.05.2010 im Tutorium abgegeben werden. Alternativ ist es bis 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet.

Tutoraufgabe 1 (Induktionsbeweis):

Sei Σ ein Alphabet. Wir wollen die "Verdoppelungsfunktion" $dup : \Sigma^* \to \Sigma^*$ betrachten, die alle Buchstaben in Wörtern verdoppelt, also z.B. das Wort ab auf $aabb$ und 123 auf 112233 abbildet.

a) Definieren sie die Funktion dup formal auf induktive (rekursive) Weise.

b) Sei M ein DFA. Wie kann man nun **automatisch** (d.h. mit einem allgemeinen Verfahren, das unabhängig vom Automaten M ist) einen DFA M^2 konstruieren, der die Sprache $dup(L(M)) := \{dup(w) \mid w \in L(M)\}$ erkennt?

c) Beweisen Sie, dass $L(M^2) = dup(L(M))$ gilt.

Lösung:

a)

$$dup(\epsilon) := \epsilon$$
$$dup(w \cdot a) := dup(w) \cdot a \cdot a \qquad a \in \Sigma, w \in \Sigma^*$$

b) Die Idee ist hier, jede Transition des Ursprungsautomaten aufzutrennen und einen "Zwischenzustand" einzusetzen, so dass das Zeichen der ursprünglichen Transition zwei mal gelesen wird. Damit der resultierende Automat ein DFA bleibt, müssen für diese neuen Zustände alle Symbole, mit denen die "zerschnittene" Transition nicht beschriftet war, zu einem "Fehlerzustand" führen, von dem aus nie wieder ein Endzustand erreicht werden kann.

Sei $M = (Q, \Sigma, \delta, q_0, F)$. Dann definieren wir $M^2 := (Q^2, \Sigma, \delta^2, q_0, F)$ mit $Q^2 := \{e\} \cup (\bigcup_{q \in Q}(\{q^a \mid a \in \Sigma\} \cup \{q\}))$, wobei wir o.B.d.A. verlangen, dass $q^a \notin Q$ gilt (entsprechendes kann man durch ein Umbenennen der Zustände in Q erreichen). Hier ist e der angesprochene neue "Fehlerzustand".

Zuletzt definieren wir

$$\delta^2(q, a) = \begin{cases} q^a & q \in Q \\ \delta(\tilde{q}, a) & q = \tilde{q}^a \in Q^2, \tilde{q} \in Q \\ e & \text{sonst} \end{cases}$$

c) Wir müssen nun $L(M^2) = dup(L(M))$ zeigen. Wir beweisen dies, indem wir die Aussage (†) $\hat{\delta}(q, w) = q' \Leftrightarrow \hat{\delta}^2(q, dup(w)) = q'$ für alle $q \in Q$ (also nicht die neuen Zustände $Q^2 \setminus Q$) per Induktion über die Wortlänge $|w|$ zeigen.

Zum Induktionsanfang ist $w = \epsilon$ und es gilt $\hat{\delta}(q, \epsilon) = q$ und $\hat{\delta}^2(q, dup(\epsilon)) = \hat{\delta}^2(q, \epsilon) = q$.

Im Induktionsschritt betrachten wir $w = w'a$ für ein $a \in \Sigma$ und setzen voraus, dass die Aussage für w' bereits gilt. Da M ein DFA ist, gibt es genau ein q' mit $\hat{\delta}(q, w' \cdot a) = q'$ und ein $\tilde{q}$ mit $\hat{\delta}(q, w') = \tilde{q}$, so dass $\delta(\tilde{q}, a) = q'$ gilt.

Nach unser Induktionshypothese gilt dies für w' genau dann, wenn auch $\hat{\delta}^2(q, dup(w')) = \tilde{q}$ gilt. Nach Konstruktion gilt aber auch $\delta^2(\tilde{q}, a) = \tilde{q}^a$ und $\delta^2(\tilde{q}^a, a) = q'$. Damit gilt dann auch die Aussage $\hat{\delta}(q, w) = \hat{\delta}(q, w' \cdot a) = q' \Leftrightarrow \hat{\delta}^2(q, dup(w)) = \hat{\delta}^2(q, dup(w' \cdot a)) = \hat{\delta}^2(q, dup(w') \cdot a \cdot a) = q'$.

Nun gilt $w \in L(M) \Leftrightarrow \hat{\delta}(q_0, w) = f \in F$ und nach (†) ist dies äquivalent zu $\hat{\delta}^2(q_0, dup(w)) = f \in F \Leftrightarrow dup(w) \in L(M^2)$. Damit ist die Aussage bewiesen.

Hausaufgabe 2 (Induktionsbeweis): (1 + 3 + 4 = 8 Punkte)

Sei Σ ein Alphabet. Wir wollen die "Spiegelfunktion" $rev : \Sigma^* \to \Sigma^*$ betrachten, die Wörter auf ihr Spiegelbild abbildet, also z.B. das Wort *ab* auf *ba* und 123 auf 321.

a) Definieren sie die Funktion *rev* formal auf induktive (rekursive) Weise.

b) Sei M ein DFA. Wie kann man nun **automatisch** (d.h. mit einem allgemeinen Verfahren, das unabhängig vom Automaten M ist) einen ϵ-NFA M_{rev} konstruieren, der die Sprache $rev(L(M)) := \{rev(w) \mid w \in L(M)\}$ erkennt?

c) Beweisen Sie, dass $L(M_{rev}) = rev(L(M))$ gilt.

Hinweis: Beweisen sie eine **allgemeine** Aussage über den Zusammenhang zwischen der Zustands-Übergangsfunktion $\hat{\delta}$ des Automaten M und der Zustands-Übergangsfunktion $\hat{\delta}_{rev}$ des Automaten M_{rev}. Verwenden Sie dazu wie in der Vorlesung eine Induktion über die Länge der betrachteten Wörter.

Lösung:

a)

$$rev(\epsilon) := \epsilon$$
$$rev(w \cdot a) := a \cdot rev(w) \qquad a \in \Sigma, w \in \Sigma^*$$

b) Die Idee ist hier, die Kanten im Automaten "umzudrehen", also in M_{rev} eine a-Transition von q nach q' zu erlauben, wenn es in M eine a-Transition von q' nach q gab. Der Startzustand in M wird dann zum einzigen Endzustand von M_{rev}. Als Startzustände in M_{rev} wollen wir eigentlich die Endzustände von M zulassen. Da aber auch ϵ-NFAs einen eindeutigen Startzustand haben müssen, führen wir einen neuen, eindeutigen Startzustand ein, der per ϵ-Transition mit den Endzuständen von M verbunden wird.

Sei $M = (Q, \Sigma, \delta, q_0, F)$. Mit einem neuen Zustand q_s (d.h. $q_s \notin Q$) ist $M_{rev} := (Q \dot{\cup} \{q_s\}, \Sigma, \delta_{rev}, q_s, \{q_0\})$. Dabei ist δ_{rev} wie folgt definiert:

$$\delta_{rev}(q, a) = \begin{cases} \{q' \in Q \mid q = \delta(q', a)\} & q \neq q_s \\ F & q = q_s \wedge a = \epsilon \\ \emptyset & q = q_s \wedge a \neq \epsilon \end{cases}$$

c) Wir müssen nun beweisen, dass $L(M_{rev}) = rev(L(M))$ gilt. Wir zeigen dafür die allgemeinere Aussage $\hat{\delta}(q, w) = q' \Leftrightarrow q \in \hat{\delta}_{rev}(q', rev(w))$ für alle $q, q' \in Q$ (d.h. für alle Zustände außer dem neu eingeführten Startzustand q_s) per Induktion über die Wortlänge $|w|$.

Im Induktionsanfang ist die Länge 0, also $w = \epsilon$. Dann gilt $\hat{\delta}(q, \epsilon) = q$ und $\hat{\delta}_{rev}(q, \epsilon) = \{q\}$.

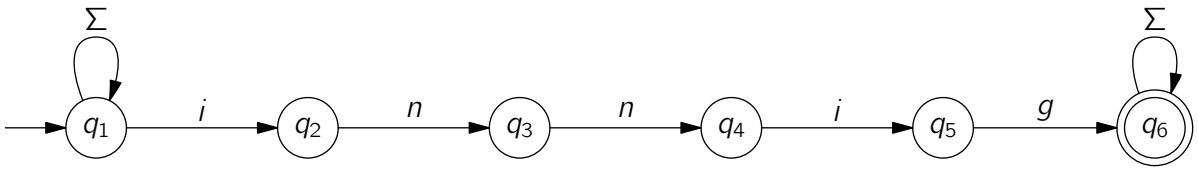
Im Induktionsschluss setzen wir für $w = w' \cdot a$, $a \in \Sigma$ voraus, dass die Aussage für w' gilt. Sei nun $\hat{\delta}(q, w) = q'$ mit $\hat{\delta}(q, w') = \tilde{q}$ und $\delta(\tilde{q}, a) = q'$. Nach Konstruktion von δ_{rev} gilt dann auch $\tilde{q} \in \delta_{rev}(q', a)$.

Nach unserer Induktionshypothese gilt außerdem $\hat{\delta}(q, w') = \tilde{q} \Leftrightarrow q \in \hat{\delta}_{rev}(\tilde{q}, rev(w'))$. Zusammen mit $\tilde{q} \in \delta_{rev}(q', a)$ folgt damit direkt auch $q \in \hat{\delta}_{rev}(q', a \cdot rev(w')) = \hat{\delta}_{rev}(q', rev(w' \cdot a)) = \hat{\delta}_{rev}(q', rev(w)) \Leftrightarrow \hat{\delta}(q, w) = q'$.

Damit gilt $w \in L(M) \Leftrightarrow \hat{\delta}(q_0, w) = f \in F \Leftrightarrow q_0 \in \hat{\delta}_{rev}(f, rev(w))$. Da nach Konstruktion $\delta_{rev}(q_s, \epsilon) = \epsilon\text{-}Huelle(q_s) = F$, gilt dies genau dann wenn $q_0 \in \hat{\delta}_{rev}(q_s, rev(w))$ und damit gilt $rev(w) \in L(M_{rev}) \Leftrightarrow w \in L(M)$.

Tutoraufgabe 3 (Potenzmengenkonstruktion):

Sei $\Sigma := \{a, b, \ldots, z\}$ ein Alphabet. Der folgende NFA akzeptiert alle Wörter, die mindestens einmal das Wort *innig* enthalten, also die Sprache $\Sigma^* innig \Sigma^*$. Um die Darstellung des Automaten zu vereinfachen, nutzen wir die verkürzende Schreibweise, die in Tutoraufgabe 4 auf dem zweiten Übungsblatt eingeführt wurde, und erlauben es, Transitionen mit Mengen von Symbolen zu beschriften.

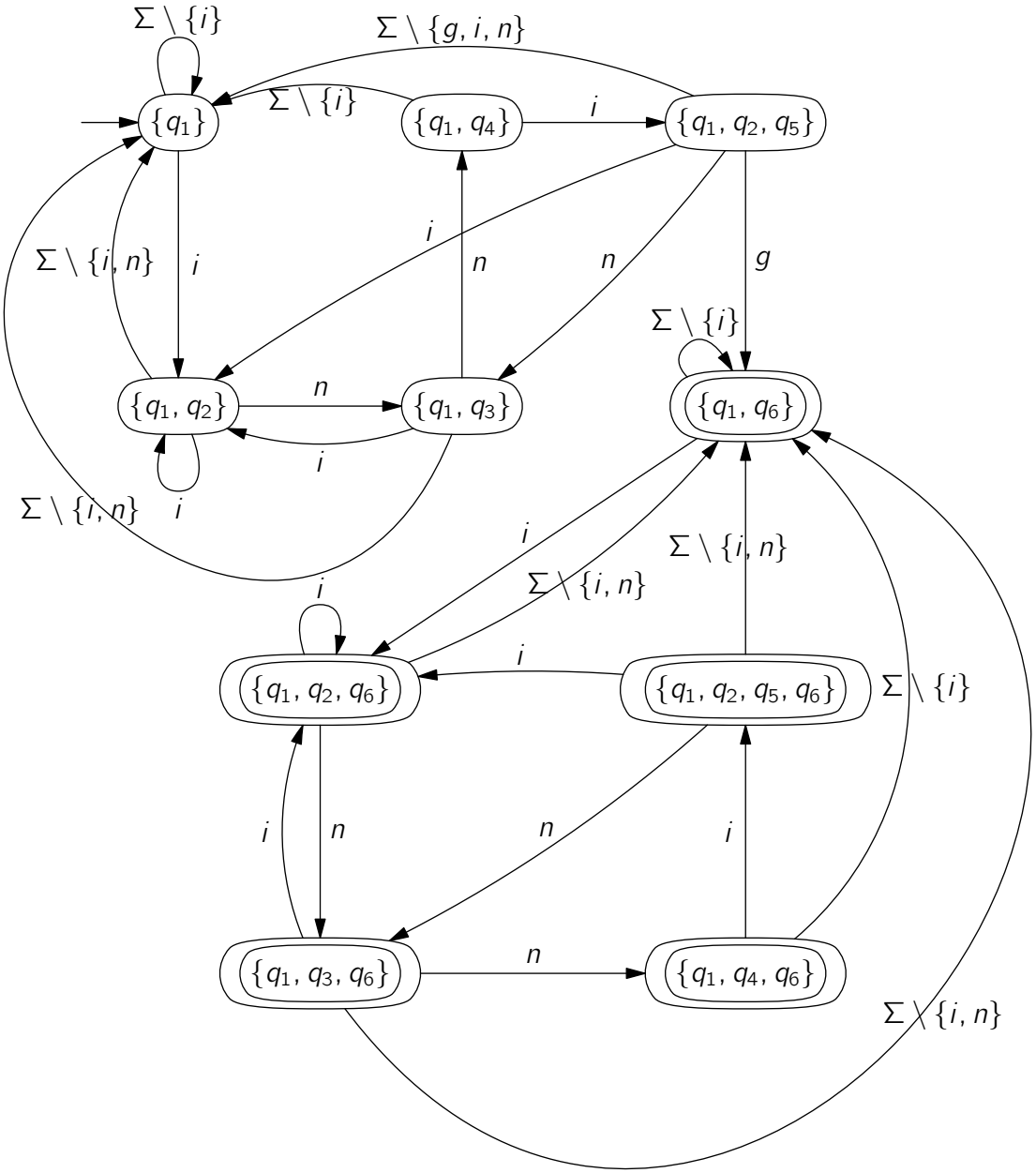


Geben Sie den Potenzautomaten für diesen NFA an, um einen DFA zu erhalten, der dieselbe Sprache akzeptiert. Wie in der Vorlesung dürfen Sie nicht erreichbare Zustände weglassen.

Lösung:

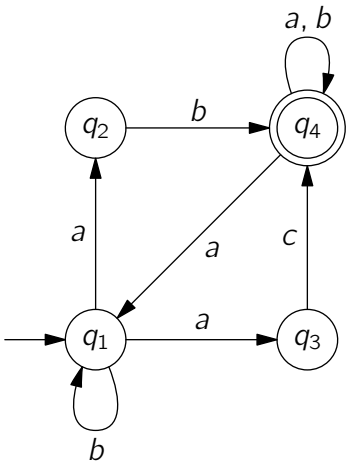
Folgende Tabelle veranschaulicht die Konstruktion des Potenzautomaten.

	g	i	n	$\Sigma \setminus \{g, i, n\}$	$\in F'$?
$\{q_1\}$	$\{q_1\}$	$\{q_1, q_2\}$	$\{q_1\}$	$\{q_1\}$	
$\{q_1, q_2\}$	$\{q_1\}$	$\{q_1, q_2\}$	$\{q_1, q_3\}$	$\{q_1\}$	
$\{q_1, q_3\}$	$\{q_1\}$	$\{q_1, q_2\}$	$\{q_1, q_4\}$	$\{q_1\}$	
$\{q_1, q_4\}$	$\{q_1\}$	$\{q_1, q_2, q_5\}$	$\{q_1\}$	$\{q_1\}$	
$\{q_1, q_2, q_5\}$	$\{q_1, q_6\}$	$\{q_1, q_2\}$	$\{q_1, q_3\}$	$\{q_1\}$	
$\{q_1, q_6\}$	$\{q_1, q_6\}$	$\{q_1, q_2, q_6\}$	$\{q_1, q_6\}$	$\{q_1, q_6\}$	✓
$\{q_1, q_2, q_6\}$	$\{q_1, q_6\}$	$\{q_1, q_2, q_6\}$	$\{q_1, q_3, q_6\}$	$\{q_1, q_6\}$	✓
$\{q_1, q_3, q_6\}$	$\{q_1, q_6\}$	$\{q_1, q_2, q_6\}$	$\{q_1, q_4, q_6\}$	$\{q_1, q_6\}$	✓
$\{q_1, q_4, q_6\}$	$\{q_1, q_6\}$	$\{q_1, q_2, q_5, q_6\}$	$\{q_1, q_6\}$	$\{q_1, q_6\}$	✓
$\{q_1, q_2, q_5, q_6\}$	$\{q_1, q_6\}$	$\{q_1, q_2, q_6\}$	$\{q_1, q_3, q_6\}$	$\{q_1, q_6\}$	✓



Hausaufgabe 4 (Potenzmengenkonstruktion): (3 Punkte)

Sei $\Sigma := \{a, b, c\}$ ein Alphabet. Betrachten Sie den folgenden NFA.

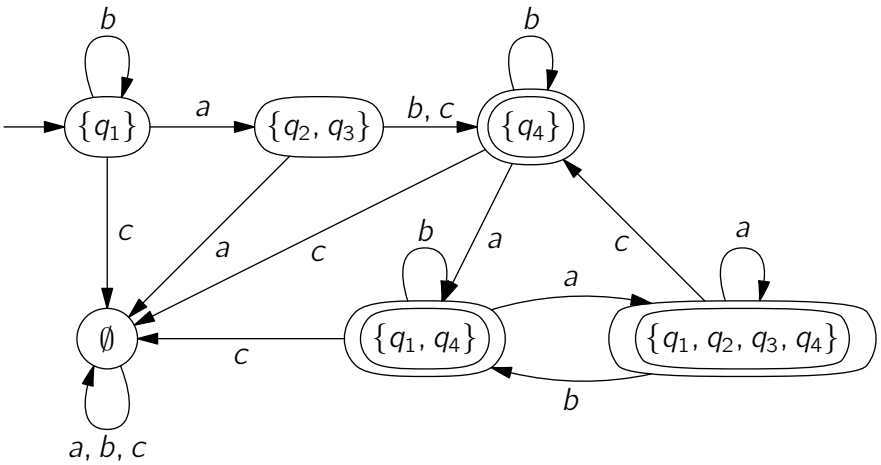


Geben Sie den Potenzautomaten für diesen NFA an, um einen DFA zu erhalten, der dieselbe Sprache akzeptiert. Wie in der Vorlesung dürfen Sie nicht erreichbare Zustände weglassen.

Lösung:

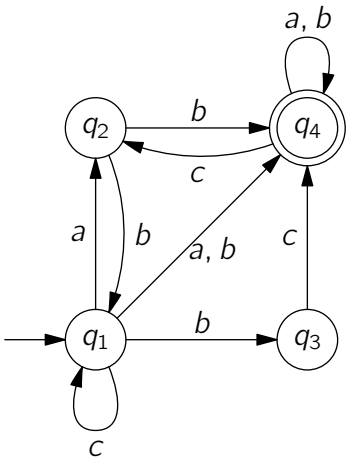
Folgende Tabelle veranschaulicht die Konstruktion des Potenzautomaten.

	a	b	c	$\in F'$?
$\{q_1\}$	$\{q_2, q_3\}$	$\{q_1\}$	$\emptyset$	
$\{q_2, q_3\}$	$\emptyset$	$\{q_4\}$	$\{q_4\}$	
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	
$\{q_4\}$	$\{q_1, q_4\}$	$\{q_4\}$	$\emptyset$	✓
$\{q_1, q_4\}$	$\{q_1, q_2, q_3, q_4\}$	$\{q_1, q_4\}$	$\emptyset$	✓
$\{q_1, q_2, q_3, q_4\}$	$\{q_1, q_2, q_3, q_4\}$	$\{q_1, q_4\}$	$\{q_4\}$	✓



Tutoraufgabe 5 (Simulation):

Gegeben sei folgender NFA über dem Alphabet $\Sigma = \{a, b, c\}$.



Geben Sie an, wie sich der Algorithmus zur Simulation dieses NFAs bei Eingabe der folgenden Wörter verhält. Es genügt dabei, die Menge S nach jeder Iteration des Simulationsverfahrens aus der Vorlesung und den Rückgabewert anzugeben.

a) *ababc*

b) *bcbca*

Lösung:

a)

Iteration	S
0	$\{q_1\}$
1	$\{q_2, q_4\}$
2	$\{q_1, q_4\}$
3	$\{q_2, q_4\}$
4	$\{q_1, q_4\}$
5	$\{q_1, q_2\}$

Rückgabe: false

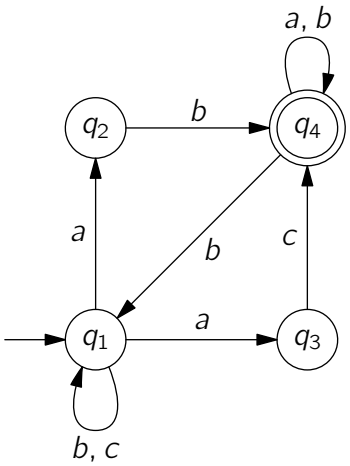
b)

Iteration	S
0	$\{q_1\}$
1	$\{q_3, q_4\}$
2	$\{q_2, q_4\}$
3	$\{q_1, q_4\}$
4	$\{q_1, q_2\}$
5	$\{q_2, q_4\}$

Rückgabe: true

Hausaufgabe 6 (Simulation): (1 + 1 = 2 Punkte)

Gegeben sei folgender NFA über dem Alphabet $\Sigma = \{a, b, c\}$.



Geben Sie an, wie sich der Algorithmus zur Simulation dieses NFAs bei Eingabe der folgenden Wörter verhält. Es genügt dabei, die Menge S nach jeder Iteration des Simulationsverfahrens aus der Vorlesung und den Rückgabewert anzugeben.

a) *bacabbac*

b) *ababca*

Lösung:

a)

Iteration	S
0	$\{q_1\}$
1	$\{q_1\}$
2	$\{q_2, q_3\}$
3	$\{q_4\}$
4	$\{q_4\}$
5	$\{q_1, q_4\}$
6	$\{q_1, q_4\}$
7	$\{q_2, q_3, q_4\}$
8	$\{q_4\}$

Rückgabe: true

b)

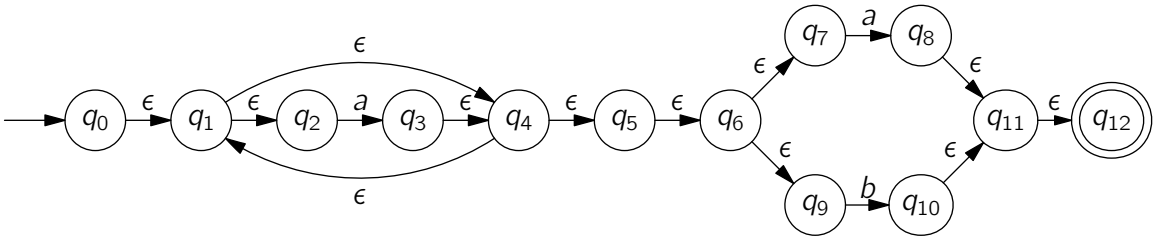
Iteration	S
0	$\{q_1\}$
1	$\{q_2, q_3\}$
2	$\{q_4\}$
3	$\{q_4\}$
4	$\{q_1, q_4\}$
5	$\{q_1\}$
6	$\{q_2, q_3\}$

Rückgabe: false

Tutoraufgabe 7 (Thompson-Konstruktion):

Erzeugen Sie mit der Thompson-Konstruktion einen NFA (mit ϵ-Transitionen) zum regulären Ausdruck $a^*(a+b)$.

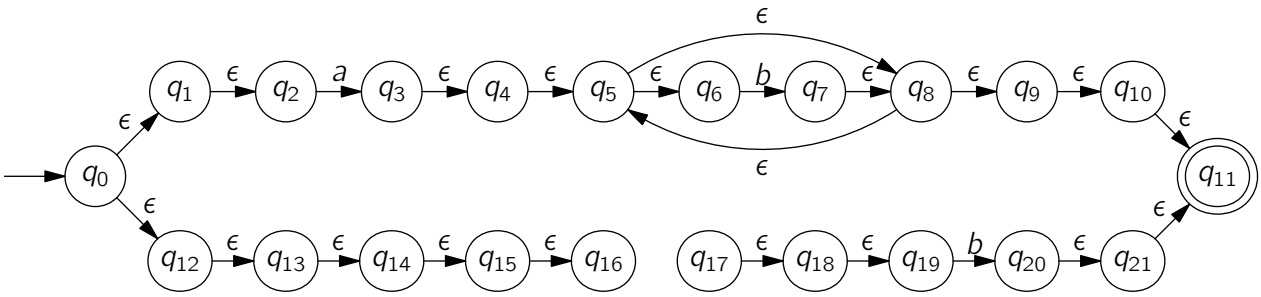
Lösung:



Hausaufgabe 8 (Thompson-Konstruktion): (3 Punkte)

Erzeugen Sie mit der Thompson-Konstruktion einen NFA (mit ϵ-Transitionen) zum regulären Ausdruck $(a \cdot b^*) + ((\epsilon \cdot \emptyset) \cdot b)$.

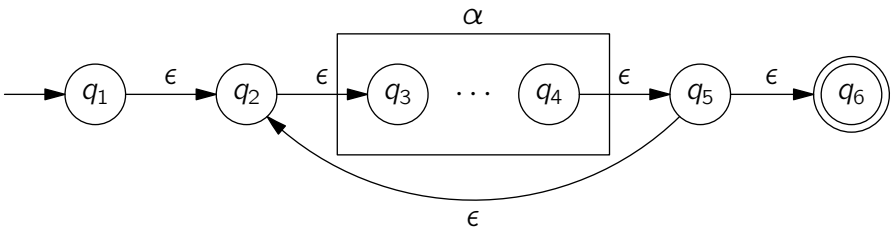
Lösung:



Tutoraufgabe 9 (Thompson-Konstruktion):

Sei α ein beliebiger regulärer Ausdruck. In der Vorlesung wurde die Thompson-Konstruktion für α^*, allerdings nicht für α^+ vorgestellt. Da $\alpha^+ := \alpha \cdot \alpha^*$ definiert wurde, ist das allerdings kein großes Problem. Trotzdem ist es manchmal hilfreich, die Thompson-Konstruktion auch *direkt* für α^+ anwenden zu können.
Erweitern Sie die Thompson-Konstruktion so, dass für jeden Ausdruck α^+ *direkt* ein Automat konstruiert wird, der auf dem Automaten für α basiert. Hierbei sollte der für α^+ erzeugte Automat weniger Transitionen haben als der Automat, der von der Thompson-Konstruktion für α^* erzeugt wird. Der entstehende Automat sollte nur einen Endzustand haben, der Endzustand sollte keine ausgehenden Kanten und der Startzustand sollte keine eingehenden Kanten haben.

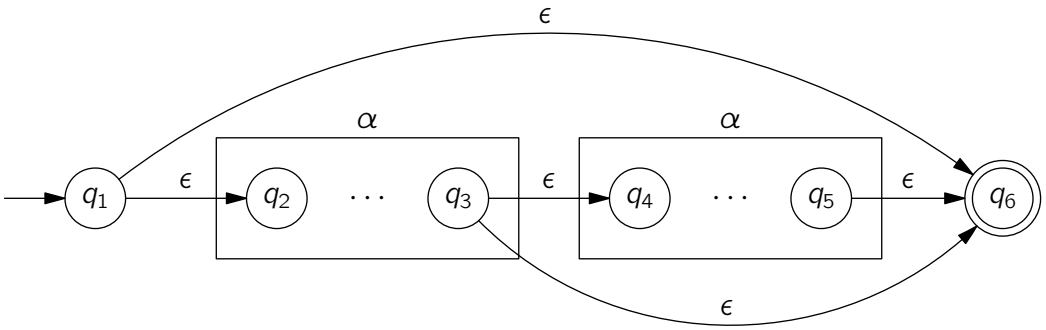
Lösung:



Hausaufgabe 10 (Thompson-Konstruktion): (3 Punkte)

Sei α ein beliebiger regulärer Ausdruck. Um Wiederholungen der von diesem Ausdruck erkannten Wörter auszudrücken, gibt es die regulären Ausdrücke ϵ (α 0-mal), α (α 1-mal), α^+ (α min. 1-mal) und α^* (α beliebig oft).
Unter Umständen ist es aber auch interessant, ein eigenes Konstrukt für die 0-malige bis 2-malige Anwendung von α zu haben. Sei $\alpha^\square := \epsilon + \alpha + \alpha \cdot \alpha$ eine entsprechende Erweiterung der regulären Ausdrücke.
Erweitern Sie die Thompson-Konstruktion so, dass für jeden Ausdruck $\alpha^\square$ **direkt** ein Automat konstruiert wird, der auf dem Automaten für α basiert. Hierbei sollte der für $\alpha^\square$ erzeugte Automat maximal so viele **Zustände** haben wie der Automat, der von der Thompson-Konstruktion für $\alpha \cdot \alpha$ erzeugt wird. Der entstehende Automat sollte nur einen Endzustand haben, der Endzustand sollte keine ausgehenden Kanten und der Startzustand sollte keine eingehenden Kanten haben.
Hinweis: Bis auf den Endzustand darf jeder Zustand beliebig viele ausgehende Transitionen haben. Außerdem darf bis auf den Startzustand jeder Zustand beliebig viele eingehende Transitionen haben.

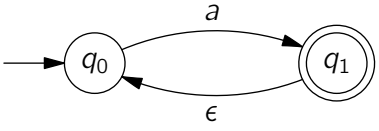
Lösung:



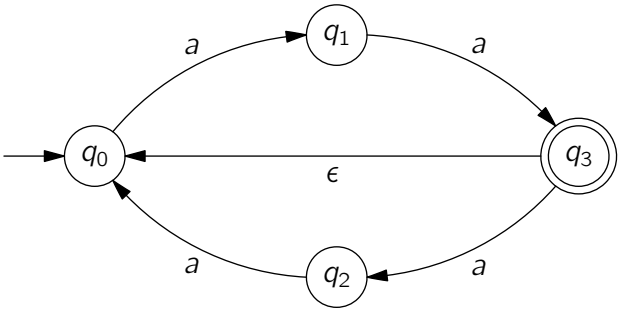
Tutoraufgabe 11 (ϵ-Kanten entfernen):

Sei $\Sigma := \{a, b, c, d\}$ ein Alphabet. Geben sie zu den folgenden ϵ-NFAs jeweils einen NFA ohne ϵ-Transitionen an, der dieselbe Sprache erkennt.

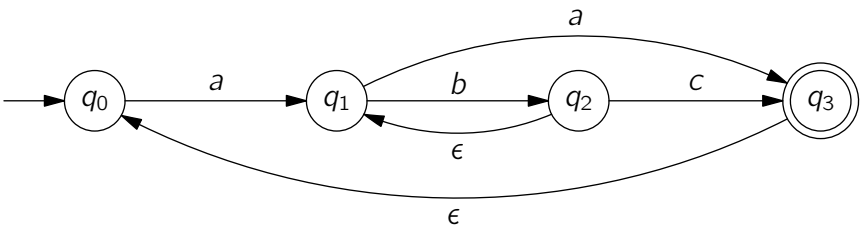
a)



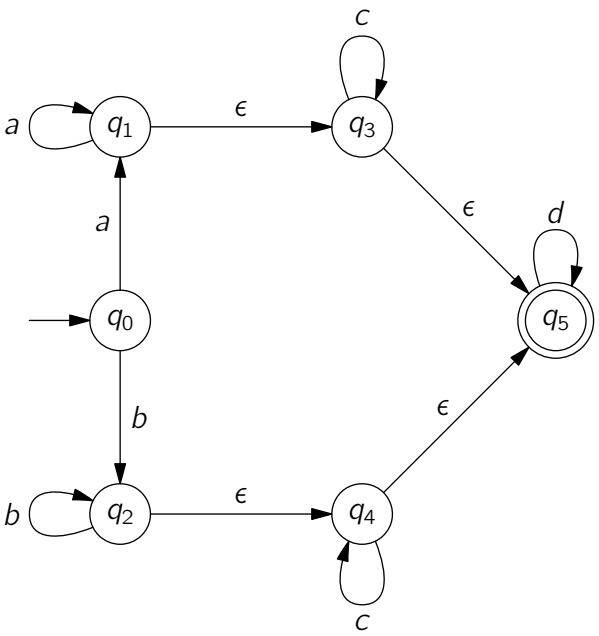
b)


q_1
a
a
q_0
q_3
ϵ
a
q_2
a

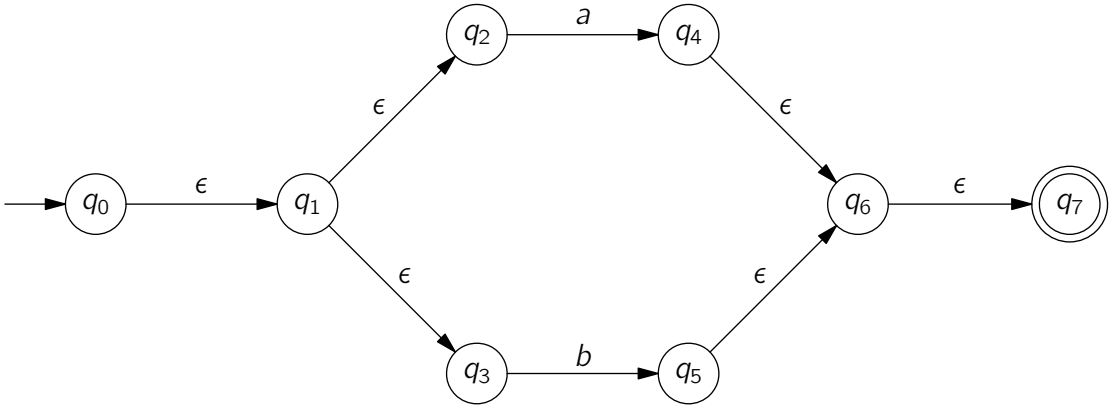
c)


a
q_0
a
q_1
b
q_2
c
q_3
ϵ
ϵ

d)


c
a
q_1
ϵ
q_3
ϵ
d
a
q_0
q_5
b
ϵ
b
q_2
ϵ
q_4
c

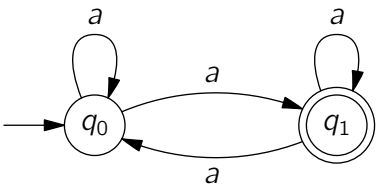
e)



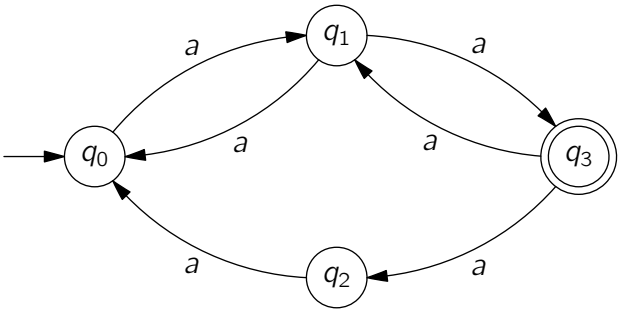
Lösung:

Sei $\Sigma := \{a, b, c, d\}$ ein Alphabet. Geben sie zu den folgenden ϵ-NFAs jeweils einen NFA ohne ϵ-Transitionen an:

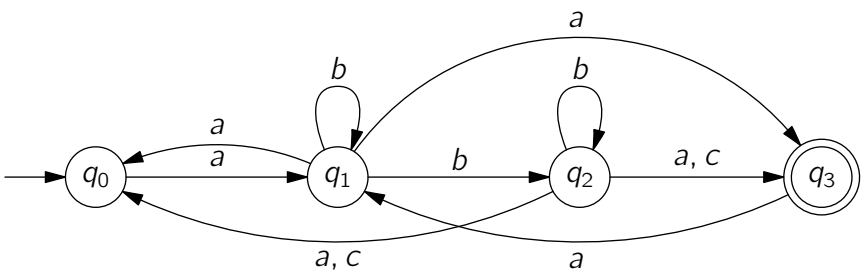
a)



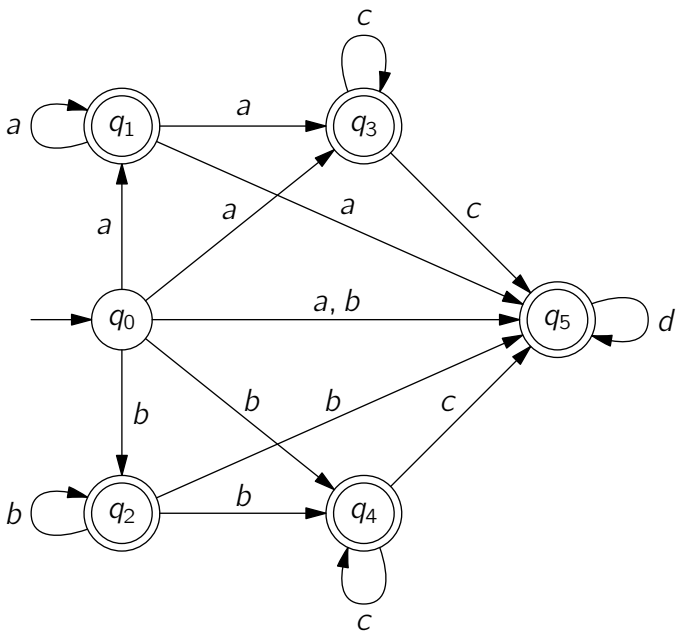
b)



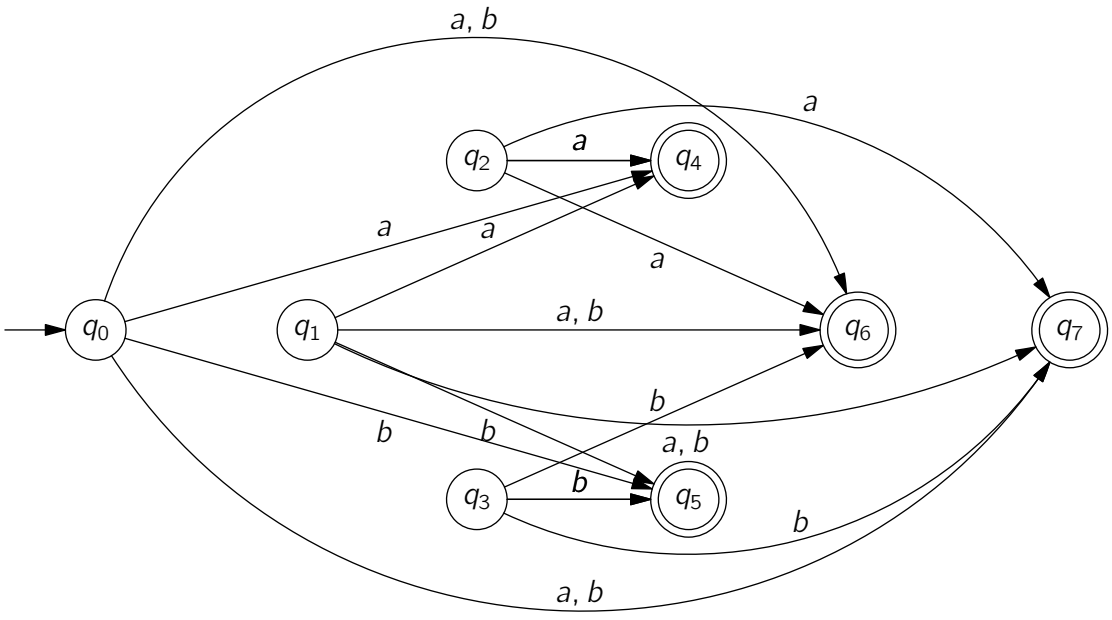
c)



d)



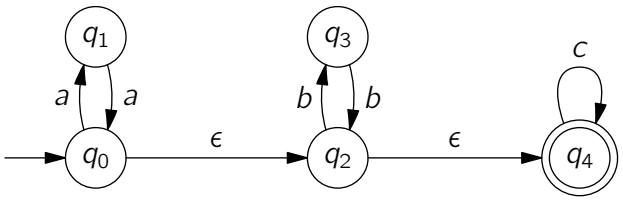
e)



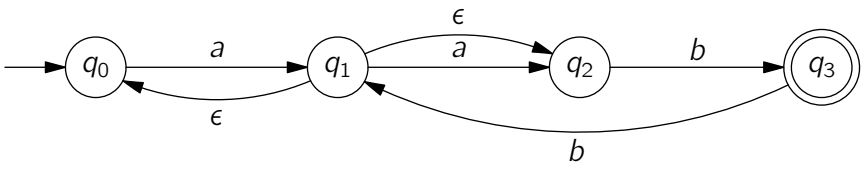
Hausaufgabe 12 (ϵ-Kanten entfernen): (2 + 2 = 4 Punkte)

Sei $\Sigma := \{a, b, c, d\}$ ein Alphabet. Geben sie zu den folgenden ϵ-NFAs jeweils einen NFA ohne ϵ-Transitionen an, der dieselbe Sprache erkennt.

a)

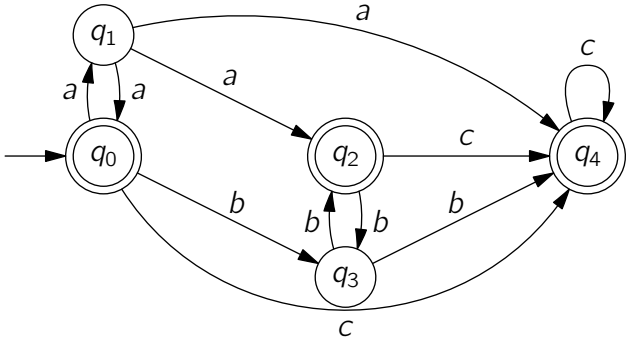

q_1
q_3
q_0
q_2
q_4
a
a
b
b
c
ϵ
ϵ

b)

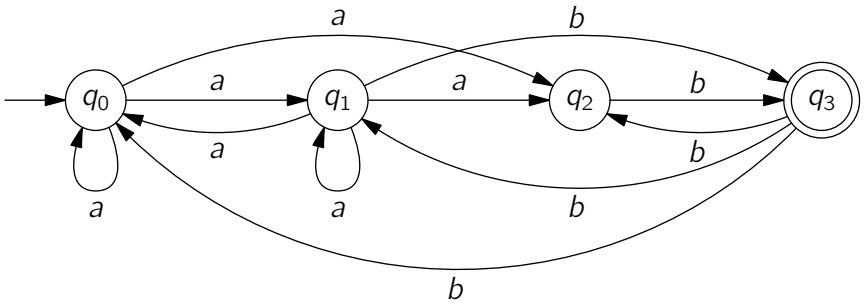

q_0
q_1
q_2
q_3
a
ϵ
ϵ
a
b
b

Lösung:

a)


q_1
q_0
q_2
q_4
q_3
a
a
a
a
c
c
b
b
b
b
c

b)


q_0
q_1
q_2
q_3
a
a
a
a
b
b
b
b
a
a
b
b

Tutoraufgabe 13 (Abschlusseigenschaften):

Die Sprache $L = \{a^n b^n \mid n \geq 0\}$ über dem Alphabet $\Sigma = \{a, b\}$ ist nicht regulär. Benutzen Sie die Abschlusseigenschaften regulärer Sprachen, um zu zeigen, dass folgende Sprachen ebenfalls nicht regulär sind.
Hinweis: Sie dürfen für jede Teilaufgabe benutzen, dass die Sprachen vorheriger Teilaufgaben nicht regulär sind.

a) $L_1 = \{a^m b^n \mid m, n \geq 0, m + n > 0, m \neq n\} \cup \Sigma^* ba \Sigma^*$

b) $L_2 = \{a^m b^n \mid m, n \geq 0, m \neq n\} \cup \Sigma^* ba \Sigma^*$

c) $L_3 = \{b^n a^n \mid n \geq 0\}$

Lösung:

a) $L = \Sigma^* \setminus L_1$.

b) $L_1 = L_2$.

c) $L = h(L_3)$ mit $h(a) = b$ und $h(b) = a$. Die Funktion h ist ein Homomorphismus, da offensichtlich $h(ww') = h(w)h(w')$. In der Tat ist h sogar ein Isomorphismus.

Prof. Dr. J. Giesl

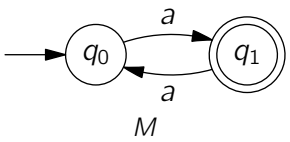
M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

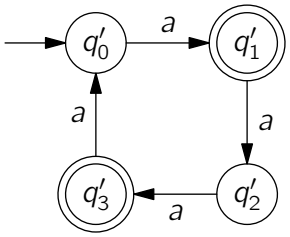
- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Mi., 19.05.2010 im Tutorium abgegeben werden. Alternativ ist es bis 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet.

Tutoraufgabe 1 (Homomorphismen auf DFAs):

Betrachten Sie die beiden folgenden DFAs M und M'.



M



M'

a) Gibt es einen Homomorphismus von M nach M'? Begründen Sie Ihre Antwort.

b) Gibt es einen Homomorphismus von M' nach M? Begründen Sie Ihre Antwort.

Lösung:

a) Nein. Der Zustand q_0 kann nicht gleichzeitig auf die Zustände q'_0 und q'_2 abgebildet werden. Ebenso kann q_1 nicht gleichzeitig auf q'_1 und q'_3 abgebildet werden. Letzteres ist allein schon nötig, um die erste Bedingung für einen Homomorphismus zu erfüllen, während die dritte Bedingung für einen Homomorphismus beide Abbildungen benötigen würde. Denn mit dem Wort aa erreicht man von q'_0 den Zustand q'_2 und umgekehrt, während man von q_0 aus wieder q_0 erreicht. Genauso verhält es sich bei den Zuständen q'_1, q'_3 und q_1.

b) Ja. Die Funktion h mit $h(q'_0) = h(q'_2) = q_0$ und $h(q'_1) = h(q'_3) = q_1$ ist ein solcher Homomorphismus.

Hausaufgabe 2 (Homomorphismen auf DFAs): (4 + 3 = 7 Punkte)

Sei Σ ein Alphabet.

a) Beweisen Sie die folgende Aussage:

Seien $M = (Q, \Sigma, \delta, q_0, F), M' = (Q', \Sigma, \delta', q_0', F')$ zwei beliebige DFAs und sei $h : Q \to Q'$ ein Homomorphismus von M nach M'. Dann gilt $L(M) = L(M')$.

Hinweis: Zeigen Sie zunächst, dass für alle $w \in \Sigma^*$ gilt: $h(\hat{\delta}(q_0, w)) = \hat{\delta}'(h(q_0), w)$. Verwenden Sie hierzu Induktion über die Länge des Wortes w.

b) Geben Sie zwei DFAs M und M' über dem Alphabet $\Sigma = \{a, b\}$ an, die alle folgenden Eigenschaften erfüllen:

- $L(M) = L(M')$
- Es existiert kein Homomorphismus h von M nach M'
- Es existiert kein Homomorphismus h' von M' nach M
- M und M' haben jeweils maximal 4 Zustände

Lösung:

a) Da h ein Homomorphismus ist, gilt:

(1) $h(q) \in F' \Leftrightarrow q \in F$
(2) $h(q_0) = q_0'$
(3) $h(\delta(q, a)) = \delta'(h(q), a)$

Wir zeigen zunächst, dass (†) $h(\hat{\delta}(q_0, w)) = \hat{\delta}'(h(q_0), w)$ per Induktion über $|w|$.

Im Induktionsanfang ist $w = \epsilon$ und $h(\hat{\delta}(q_0, \epsilon)) = h(q_0) \overset{(2)}{=} q_0' = \hat{\delta}'(q_0', \epsilon) \overset{(2)}{=} \hat{\delta}'(h(q_0), \epsilon)$.

Als Induktionshypothese nehmen wir an, dass (†) für $w' \in \Sigma^*$ gilt.

Im Induktionsschritt ist nun $w = w'a$ mit $a \in \Sigma$. Es gilt $h(\hat{\delta}(q_0, w)) = h(\hat{\delta}(q_0, w'a)) = h(\delta(\hat{\delta}(q_0, w'), a)) \overset{(3)}{=} \delta'(h(\hat{\delta}(q_0, w')), a) \overset{IH}{=} \delta'(\hat{\delta}'(h(q_0), w'), a) = \hat{\delta}'(h(q_0), w'a) = \hat{\delta}'(h(q_0), w)$.

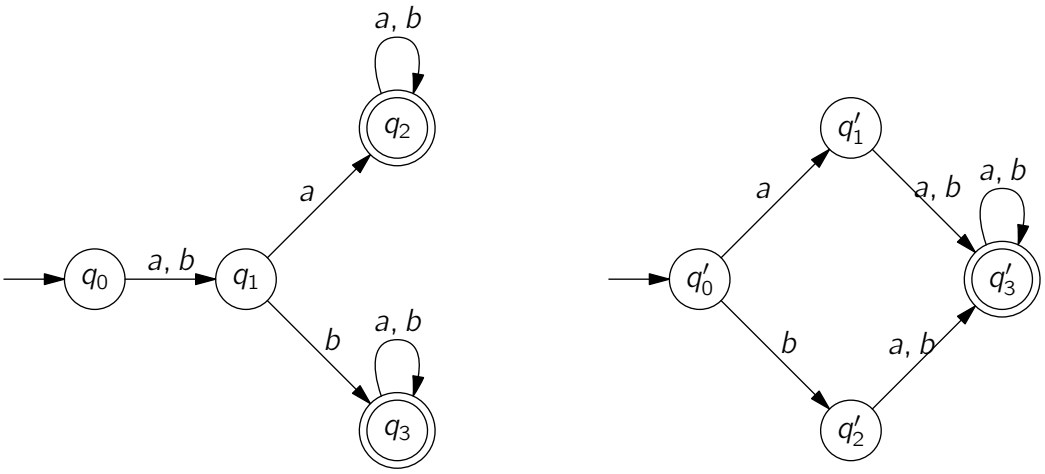
Damit gilt:

$$\begin{aligned} L(M) = L(M') &\Leftrightarrow \forall w \in \Sigma^* : w \in L(M) \Leftrightarrow w \in L(M') \\ &\Leftrightarrow \forall w \in \Sigma^* : \hat{\delta}(q_0, w) \in F \Leftrightarrow \hat{\delta}'(q_0', w) \in F' \\ &\overset{(2)}{\Leftrightarrow} \forall w \in \Sigma^* : \hat{\delta}(q_0, w) \in F \Leftrightarrow \hat{\delta}'(h(q_0), w) \in F' \\ &\overset{(\dagger)}{\Leftrightarrow} \forall w \in \Sigma^* : \hat{\delta}(q_0, w) \in F \Leftrightarrow h(\hat{\delta}(q_0, w)) \in F' \end{aligned}$$

$\forall w \in \Sigma^* : \hat{\delta}(q_0, w) \in F \Leftrightarrow h(\hat{\delta}(q_0, w)) \in F'$ gilt wegen (1).

□

b)



Beide DFAs akzeptieren die Sprache $L = \{w \in \{a,b\}^* \mid |w| \geq 2\}$. Der Zustand q_1 lässt sich nicht gleichzeitig auf q'_1 und q'_2 abbilden, während q'_3 nicht gleichzeitig auf q_2 und q_3 abgebildet werden kann. Ersteres ist nötig, da man von q'_0 aus mit a den Zustand q'_1 erreicht, während man mit b den Zustand q'_2 erreicht. Von q_0 aus erreicht man allerdings mit beiden Worten den Zustand q_1. Letzteres ist nötig, da man von q_0 aus mit aa den Zustand q_2 erreicht, während man mit ab den Zustand q_3 erreicht. Von q'_0 aus erreicht man allerdings mit beiden Worten den Zustand q'_3.

Tutoraufgabe 3 (Myhill-Nerode-Relation):

Bestimmen Sie für die folgenden Sprachen L_i über den Alphabeten Σ_i die Menge der Äquivalenzklassen $\Sigma^*/\equiv_{L_i}$. Geben Sie für die Sprachen L_i, für die diese Menge endlich ist, einen minimalen DFA an, der L_i erkennt. Geben Sie für die **anderen** Sprachen für alle Paare $[w]_{\equiv_{L_i}} \neq [w']_{\equiv_{L_i}}$ ein trennendes Wort u an, so dass $wu \in L_i$ und $w'u \notin L_i$ gilt.

a) Seien $\Sigma_1 := \{a,b\}^*$ und $L_1 := \{w \in \Sigma_1 \mid w \text{ enthält } ab \text{ oder } aa\}$

b) Sei $\Sigma_2 := \{0,1\}^*$.

Wir führen die "Gewichtsfunktion" $weight : \{0,1\}^* \to \mathbb{N}_0$ ein, die die Quersumme einer Binärzahl berechnet:

$$weight(\epsilon) := 0$$
$$weight(w \cdot a) := weight(w) + a \qquad a \in \{0,1\}, w \in \{0,1\}^*$$

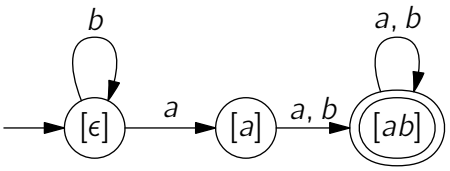
$L_2 := \{w \in \Sigma_2^* \mid \exists k \in \mathbb{N}_0 : weight(w) = 2^k\}$

c) Seien $\Sigma_3 := \{0,1\}^*$ und $L_3 := \{w \in \Sigma_3^* \mid \exists n \in \mathbb{N}_0 : w = 1^n 0 1^n\}$

Lösung:

Wir verwenden im folgenden die Schreibweise $[w]$ statt $[w]_{\equiv_{L_i}}$ für ein $w \in \Sigma_i$, die Sprache L_i ist vom Kontext eindeutig festgelegt.

a) $\Sigma^*/\equiv_{L_1} = \{[\epsilon], [a], [ab]\}$



b) Die Äquivalenzklassen entsprechen jeweils den Mengen aller Wörter, die die gleiche Anzahl von 1en enthalten:

$\Sigma^*/\equiv_{L_2} = \{\{w \in \{0,1\}^* \mid weight(w) = k\} \mid k \in \mathbb{N}_0\}$

Seien nun $[w] \neq [w'] \in \Sigma^*/\equiv_{L_2}$. Wir setzen $m = weight(w)$ und $m' = weight(w')$. Nach Voraussetzung wissen wir, dass $m \neq m'$. Ohne Beschränkung der Allgemeinheit fordern wir $m < m'$ (dies kann durch Vertauschen von w, w' erreicht werden).

Wir wollen ein trennendes Wort u finden, so dass $w \cdot u \in L_2$ und $w' \cdot u \notin L_2$ gilt. Betrachten wir zuerst zwei Spezialfälle:

Fall $m = 0$ und $m' \neq 2^k - 1$ für ein $k \in \mathbb{N}_0$: Dann gilt für $u = 1$, dass $\tilde{w} \cdot 1 \in L_2 \forall \tilde{w} \in [w]$ und $\tilde{w'} \cdot 1 \notin L_2 \forall \tilde{w'} \in [w']$.

Fall $m = 0$ und $m' = 2^k - 1$ für ein $k \in \mathbb{N}_0$: Dann gilt für $u = 11$, dass $\tilde{w} \cdot 11 \in L_2 \forall \tilde{w} \in [w]$ und $\tilde{w'} \cdot 11 \notin L_2 \forall \tilde{w'} \in [w']$.

Wir betrachten nun die verbliebenen Fälle, d.h. $1 \leq m < m'$. Seien $l = \lceil log_2(m) \rceil$ und $l' = \lceil log_2(m') \rceil$, d.h. die kleinsten natürlichen Zahlen mit $2^{l-1} \leq m < 2^l$ und $2^{l'-1} \leq m' < 2^{l'}$. Aus $m < m'$ folgt hier insbesondere auch $l \leq l'$. Mit $\delta = 2^l - m$ (bzw. $\delta' = 2^{l'} - m'$) beschreiben wir nun also die Differenz zwischen m (bzw. m') und der nächstgrößeren Zweierpotenz 2^l (bzw. $2^{l'}$).

Im Fall $\delta \neq \delta'$ ist $u = 1^\delta$ ein trennendes Wort, da dann [1] $weight(\tilde{w} \cdot u) = m + \delta = 2^l$ und $weight(\tilde{w'} \cdot u) = m' + \delta' \neq 2^{l'}$ für alle $\tilde{w} \in [w], \tilde{w'} \in [w']$ ist. Da außerdem $l \leq l'$ gilt, wissen wir, dass $m' + \delta' < 2^{l'+1}$ gilt. Damit ist dann $\tilde{w} \cdot u \in L_2$ und $\tilde{w'} \cdot u \notin L_2$.

Im Fall $\delta = \delta'$ folgt direkt $l < l'$. Dann ist $u = 1^{\delta+2^l}$ ein trenndes Wort, da dann $weight(\tilde{w} \cdot u) = m + \delta + 2^l = 2^l + 2^l = 2^{l+1}$ und $weight(\tilde{w'} \cdot u) = m' + \delta + 2^l = 2^{l'} + 2^l < 2^{l'+1}$ folgt.

c)

$$C_3^n := \{1^n\}$$
$$C_3^{-k} := \{1^n 0 1^{n-k} \mid n \in \mathbb{N}_0\}$$
$$\Sigma^*/\equiv_{L_3} = \bigcup_{i \in \mathbb{N}_0} \{C_3^i, C_3^{-i}\} \cup \{\{1^n 0 1^k \mid k > n\} \cup \{w0w'0w'' \mid w, w', w'' \in \Sigma^*\}$$

Hierbei entsprechen die Elemente von C_3^n den Wörtern, in denen bereits n mal die 1 gelesen wurde; die C_3^{-k} sind die Wörter, in denen noch k 1en "fehlen" und in der letzten Äquivalenzklasse liegen all die Wörter, die nicht Prefix eines Wortes der Sprache sind.

Seien nun $[w], [w']$ zwei Äquivalenzklassen mit $[w] \neq [w']$.

Fall $w = C_3^t$ für ein $t \in \mathbb{N}_0$: Mit $u = 01^t$ gilt $w \cdot u \in L$, aber nicht $w' \cdot u \in L$. Ist w' von der Form $1^{t'}$ mit $t' \neq t$, ist dies klar. Für alle anderen Fälle enthält $w' \cdot u$ mehr als einmal die 0 und kann daher nicht in der Sprache liegen.

Fall $w = C_3^{-t}$ für ein $t \in \mathbb{N}_0$: Mit $u = 1^t$ gilt $w \cdot u = 1^n 0 1^{n-t} \cdot 1^t = 1^n 0 1^n \in L$. Ist w' von der Form $1^{t'}$, enthält $w' \cdot u$ keine 0 und ist daher nicht in der Sprache. Ist w' von der Form $1^l 0 1^k$ mit $t \neq t' = l - k$, ist $w' \cdot u$ ebenfalls nicht in L. Ist w' eines der "ungültigen" Prefixe, gilt generell $w' \cdot u \notin L$.

[1] An dieser Stelle sei bemerkt, dass *weight* ein Monoid-Homomorphismus von $(\{0,1\}^*, \cdot)$ nach $(\mathbb{N}_0, +)$ ist.

Hausaufgabe 4 (Myhill-Nerode-Relation): (2 + 3 + 2 = 7 Punkte)

Bestimmen Sie für die folgenden Sprachen L_i über den Alphabeten Σ_i die Menge der Äquivalenzklassen $\Sigma^*/\equiv_{L_i}$. Geben Sie für die Sprachen L_i, für die diese Menge endlich ist, einen minimalen DFA an, der L_i erkennt. Geben Sie für die **anderen** Sprachen für alle Paare $[w]_{\equiv_{L_i}} \neq [w']_{\equiv_{L_i}}$ ein trennendes Wort u an, so dass $wu \in L_i$ und $w'u \notin L_i$ gilt.

a) Seien $\Sigma_4 := \{a, b\}$ und $L_4 := \{w \in \Sigma_4^* \mid w \text{ enthält } aba \text{ oder } abba\}$

b) Seien $\Sigma_5 := \{0, 1, \ldots, 9\}$ und $L_5 := \{w \in \Sigma_5^* \mid w \text{ durch 3 teilbare Dezimalzahl }\} = \{0, 3, 6, 9, 12, \ldots\}$

Dabei sind Dezimalzahlen all die Ziffernfolgen, die entweder die 0 sind oder nicht mit 0 beginnen.

Hinweis: Um herauszufinden, ob eine Dezimalzahl z durch 3 teilbar ist, kann man die Quersumme q der Zahl bilden und stattdessen überprüfen, ob q durch 3 teilbar ist. Genau dann, wenn dies der Fall ist, ist auch z durch 3 teilbar.

c) Sei $\Sigma_6 := \{0, 1\}$.

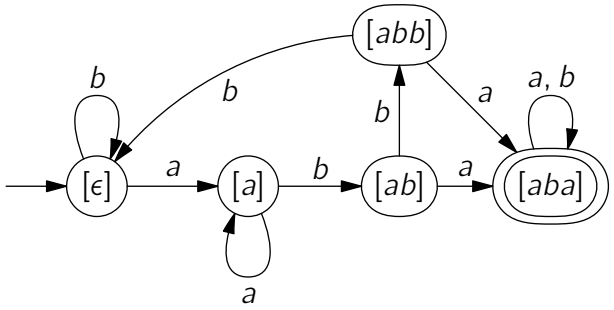
Wir erinnern an $dup : \Sigma^* \to \Sigma^*$ aus Tutoraufgabe 1 von Übungsblatt 3:

$$dup(\epsilon) := \epsilon$$
$$dup(w \cdot a) := dup(w) \cdot a \cdot a \qquad a \in \Sigma, w \in \Sigma^*$$

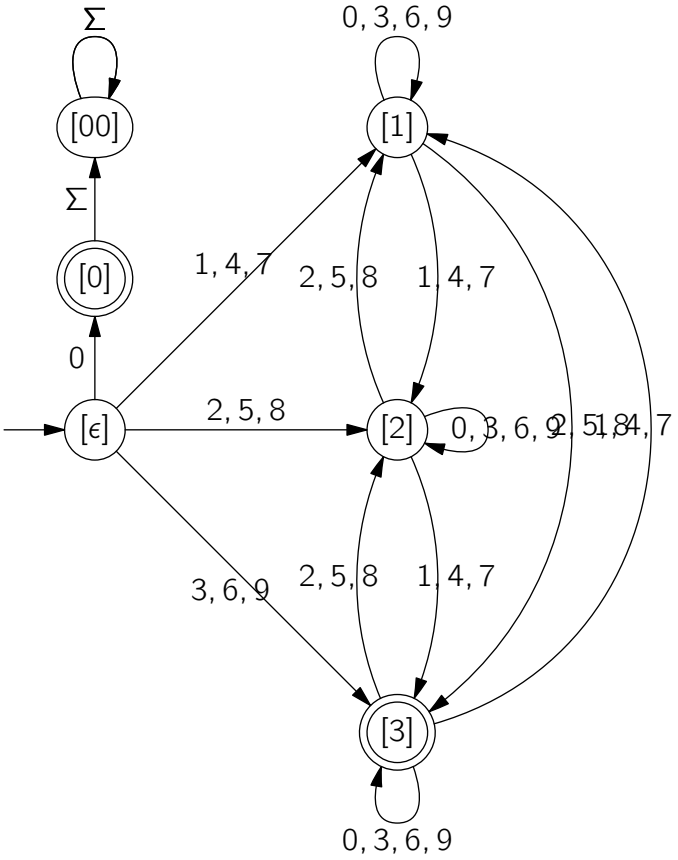
$L_6 := \{w \in \Sigma_6^* \mid \exists w' \in \{1\}^* : w = w' \cdot 0 \cdot dup(w')\}$

Lösung:

a) $\Sigma^*/\equiv_{L_4} = \{[\epsilon], [a], [ab], [abb], [aba]\}$



b) $\Sigma^*/\equiv_{L_5} = \{[\epsilon], [0], [00], [1], [2], [3]\}$



c) Wir können auch eine alternative Schreibweise für L_6 angeben:

$L_6 = \{w \in \{1,0\}^* \mid \exists n \in \mathbb{N}_0 : w = 1^n 0 1^{2n}\}$

Damit können wir analog zur Tutoraufgabe 3(c) die folgenden Äquivalenzklassen benennen:

$$
\begin{aligned}
C_6^n &:= \{1^n\} \\
C_6^{-k} &:= \{1^n 0 1^{2n-k} \mid n \in \mathbb{N}_0\} \\
\Sigma^*/\equiv_{L_6} &= \bigcup_{i \in \mathbb{N}_0} \{C_6^i, C_6^{-i}\} \cup \{\{1^n 0 1^k \mid k > 2n\} \cup \{w0w'0w'' \mid w, w', w'' \in \Sigma^*\}
\end{aligned}
$$

Hierbei entsprechen die Elemente von C_6^n den Wörtern, in denen bereits n mal die 1 gelesen wurde; die C_6^{-k} sind die Wörter, in denen noch k 1en "fehlen" und in der letzten Äquivalenzklasse liegen all die Wörter, die nicht Prefix eines Wortes der Sprache sind.

Seien nun $[w], [w']$ zwei Äquivalenzklassen mit $[w] \neq [w']$.

Fall $w = C_6^t$ für ein $t \in \mathbb{N}_0$: Mit $u = 01^{2t}$ gilt $w \cdot u \in L_6$, aber nicht $w' \cdot u \in L_6$. Ist w' von der Form $1^{t'}$ mit $t' \neq t$, ist dies klar. Für alle anderen Fälle enthält $w' \cdot u$ mehr als einmal die 0 und kann daher nicht in der Sprache liegen.

Fall $w = C_6^{-t}$ für ein $t \in \mathbb{N}_0$: Mit $u = 1^t$ gilt $w \cdot u = 1^n 0 1^{2n-t} \cdot 1^t = 1^n 0 1^{2n} \in L_6$. Ist w' von der Form $1^{t'}$, enthält $w' \cdot u$ keine 0 und ist daher nicht in der Sprache. Ist w' von der Form $1^l 0 1^k$ mit $t \neq t' = 2l - k$, ist $w' \cdot u$ ebenfalls nicht in L_6. Ist w' eines der "ungültigen" Prefixe, gilt generell $w' \cdot u \notin L_6$.

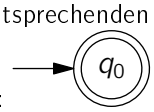
Tutoraufgabe 5 (Sprach-Index):

Kann es einen NFA $M = (Q, \Sigma, \delta, q_0, F)$ geben, bei dem $|Q|$ kleiner ist als der Index von $\equiv_{L(M)}$?

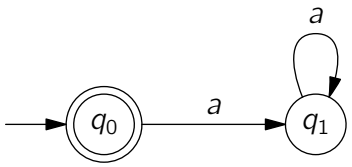
Lösung:

Ja! Es reicht, einen entsprechenden NFA und den passenden minimalen DFA anzugeben:

NFA M über $\Sigma = \{a\}$:



minimaler DFA M' mit $L(M) = L(M')$:



Hausaufgabe 6 (Größenabschätzung für Automaten): (3 Punkte)

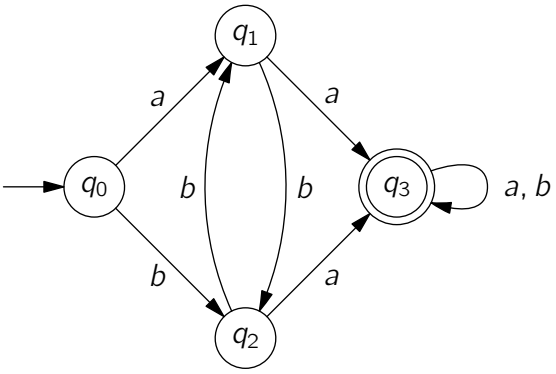
Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein NFA mit $|Q| = 16$ Zuständen. Kann ein Automat $M' = (Q', \Sigma, \delta', q_0', F')$ existieren, der ein minimaler DFA mit $|Q'| = 76543$ und $L(M) = L(M')$ ist? Begründen Sie Ihre Antwort.

Lösung:

Nein. Durch die Potenzmengenkonstruktion für M erzeugt man einen DFA M'' mit $L(M) = L(M'')$. Die Konstruktion erzeugt dabei maximal einen Zustand pro Menge von Zuständen aus Q. Es gibt daher maximal $2^{|Q|} = |\mathcal{P}(Q)| = 2^{16} = 65536$ Zustände in M''. Da $L(M') = L(M) = L(M'')$ gilt und der DFA M'' weniger Zustände als M' hat, kann M' kein minimaler DFA für $L(M)$ sein.

Tutoraufgabe 7 (Vergleich $\sim$ und $\equiv_L$):

Der folgende Automat erkennt die Sprache L. Geben Sie zwei Wörter $u, v \in \{a, b\}^*$ an, für die $u \equiv_L v$ und $u \not\sim v$ gilt. Hierbei bedeutet $u \sim v \Leftrightarrow \hat{\delta}(q_0, u) = \hat{\delta}(q_0, v)$. Geben Sie ausserdem $[u]_\sim$, $[v]_\sim$, $[u]_{\equiv_L}$ und $[v]_{\equiv_L}$ in Form von regulären Ausdrücken an.

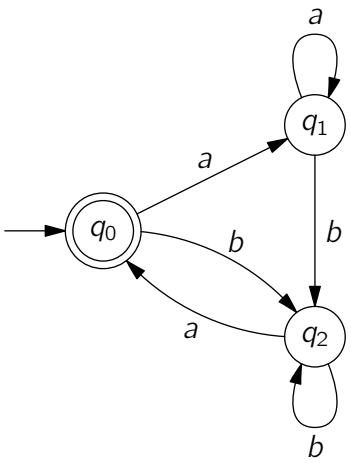


Lösung:

Wir wählen $u = a$ und $v = b$. Aus der Definition von $\sim$ ergibt sich $[w]_\sim = \{ w' \mid \hat{\delta}(q_0, w') = \hat{\delta}(q_0, w) \}$. Es folgt somit $[a]_\sim = (a + bb)(bb)^*$ und $[b]_\sim = (ab + b)(bb)^*$.
Die Äquivalenzklassen von $\equiv_L$ beschreiben die Wörter, die durch Anhängen von Suffixen bezüglich der Sprache L nicht unterschieden werden können. Es gilt $[a]_{\equiv_L} = (a + b)b^* = [b]_{\equiv_L}$.

Hausaufgabe 8 (Minimalität): (3 Punkte)

Beweisen Sie, dass folgender DFA minimal ist. Verwenden Sie hierzu den Satz von Myhill-Nerode.



Lösung:

Da der Automat 3 Zustände besitzt, kann die Nerode-Kongruenz der von ihm erkannten Sprache höchstens 3 Äquivalenzklassen besitzen. Wir betrachten die Äquivalenzklassen von $\sim$, die nach Lemma 2.6.2 eine Verfeinerung der Klassen von $\equiv_L$ darstellen.

- $[\epsilon]_\sim = \{w \in \{a, b\}^* \mid \hat{\delta}(q_0, w) = q_0\}$
- $[a]_\sim = \{w \in \{a, b\}^* \mid \hat{\delta}(q_0, w) = q_1\}$
- $[b]_\sim = \{w \in \{a, b\}^* \mid \hat{\delta}(q_0, w) = q_2\}$

Es bleibt zu zeigen, dass diese Klassen keine echte Verfeinerung bezüglich $\equiv_L$ sind. Dafür geben wir für aus je zwei Klassen Elemente an, die sich bezüglich $\equiv_L$ unterscheiden.

- Für $[\epsilon]_\sim$ und $[a]_\sim$: $\epsilon\epsilon \in L$, aber $a\epsilon \notin L$,
- für $[a]_\sim$ und $[b]_\sim$: $aa \notin L$, aber $ba \in L$ und
- für $[b]_\sim$ und $[\epsilon]_\sim$: $b\epsilon \notin L$, aber $\epsilon\epsilon \in L$.

Somit hat ein minimaler Automat, der die Sprache L erkennt mindestens 3 Zustände.

Tutoraufgabe 9 (Myhill-Nerode-DFA und Äquivalenzklassen):

Sei $[\epsilon]_{\equiv_L} \cap L = \emptyset$. Zeigen Sie, dass der Startzustand im Myhill-Nerode-DFA zu L kein Endzustand ist ($q_0 \notin F$).

Lösung:

Aus der Konstruktion des Myhill-Nerode-DFA folgt, dass $[\epsilon]_{\equiv_L}$ der Startzustand des Automaten ist. Es gilt offensichtlich, dass $\epsilon \in [\epsilon]_{\equiv_L}$ enthalten ist. Mit $[\epsilon]_{\equiv_L} \cap L = \emptyset$ folgt $\epsilon \notin L$. Der im Automaten für ϵ erreichte Zustand ist der Startzustand und darf also kein Endzustand sein.

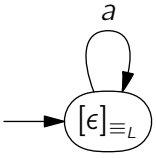
Hausaufgabe 10 (Abschätzung von $\equiv_L$): (4 Punkte)

Sei L eine reguläre Sprache. Für diese Sprache ist $[\epsilon]_{\equiv_L} \cap L = \emptyset$ bekannt (also kein Wort aus dieser Äquivalenzklasse ist in der Sprache L enthalten). Weiterhin gilt $[\epsilon]_{\equiv_L} = [a]_{\equiv_L}$. Zudem ist $[ab]_{\equiv_L} \subseteq L$ bekannt (also alle Wörter in dieser Äquivalenzklasse sind auch in der Sprache L enthalten). Welche der folgenden Aussagen sind korrekt? Begründen Sie Ihre Antworten.
Hinweis: Versuchen Sie, den Myhill-Nerode-DFA für L so weit wie möglich zu konstruieren!

1. $L = \emptyset$
2. $[aaaa]_{\equiv_L} \cap L = \emptyset$
3. $[aaaa]_{\equiv_L} \subseteq L$
4. $b \equiv_L ab$

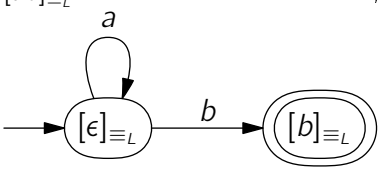
Lösung:

1. Falsch. Offensichtlich gilt $ab \in [ab]_{\equiv_L}$, also $[ab]_{\equiv_L} \neq \emptyset$. Allgemein kann keine Äquivalenzklasse $[w]_{\equiv_L}$ leer sein, da jeweils der Repräsentant w enthalten ist. Mit $[ab]_{\equiv_L} \subseteq L$ folgt $ab \in L$. Die erste Aussage ist also falsch.
2. Wahr. Aus $[\epsilon]_{\equiv_L} \cap L = \emptyset$ folgt, dass der Startzustand des Myhill-Nerode-DFA kein Endzustand ist. Aus $\epsilon \in [\epsilon]_{\equiv_L}$ und $\epsilon \cdot a = a \in [a]_{\equiv_L} = [\epsilon]_{\equiv_L}$ folgt, dass es eine a-Schleife für den $[\epsilon]_{\equiv_L}$-Zustand (also dem Startzustand) geben muss. Bisher ergibt sich also folgender Myhill-Nerode-DFA:

   An diesem einen Zustand kann man schon ablesen, dass die Wörter $\epsilon, a, aa, aaa, aaaa, \ldots$ sowohl in der gleichen Äquivalenzklasse liegen als auch nicht in der Sprache L enthalten sind. Die zweite Aussage ist also wahr.
3. Falsch. Da $aaaa \in [aaaa]_{\equiv_L}$ gilt, ist $[aaaa]_{\equiv_L}$ also nicht leer. Aus der zweiten Aussage folgt dann, dass die dritte Aussage falsch ist.

4. Wahr. Da $[\epsilon]_{\equiv_L} = [a]_{\equiv_L} \cap L = \emptyset$ und $[ab]_{\equiv_L} \subseteq L$ gelten, folgt $[\epsilon]_{\equiv_L} \neq [ab]_{\equiv_L}$. Im Myhill-Nerode-DFA existiert also ein Zustand zu $[ab]_{\equiv_L}$, welcher b-Nachfolger vom Zustand für $[a]_{\equiv_L} = [\epsilon]_{\equiv_L}$ ist. Dieser Zustand für $[ab]_{\equiv_L}$ muss ein Endzustand sein, da $[ab]_{\equiv_L} \subseteq L$ gilt.



Am bisher konstruierten Automaten kann man ablesen, dass $[b]_{\equiv_L} = [ab]_{\equiv_L}$ gelten muss, also auch $b \equiv ab$.

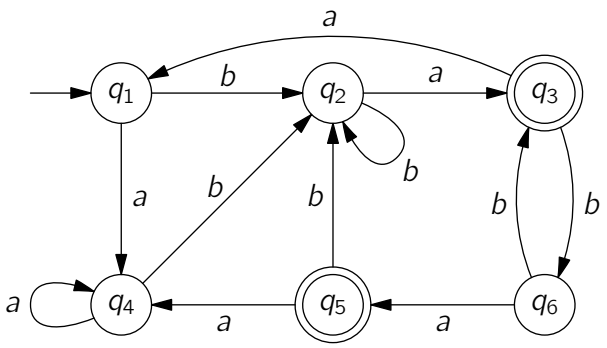
Prof. Dr. J. Giesl
M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Sie können die Lösungen zu diesem Aufgabenblatt bei der Präsenzübung am Di., 01.06.2010 abgeben. Alternativ ist es bis Mi., 02.06.2010 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** 1, 3 und 7 werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet. Die **Tutoraufgabe** 5 zum Pumping Lemma wird in der Globalübung am 21.05.2010 besprochen.
- **Bitte melden Sie sich bis zum 21.05.2010 im Übungssystem für die Präsenzübung an!**
- **Am Mi., 02.06.2010 werden keine Tutorien stattfinden.**

Tutoraufgabe 1 (DFA Minimierung):

Verwenden Sie den schnellen Markierungsalgorithmus aus der Vorlesung um folgenden DFA zu minimieren. Geben Sie in der Lösung die beim Anwenden des Algorithmus entstehende Tabelle und den minimalen Automaten an.



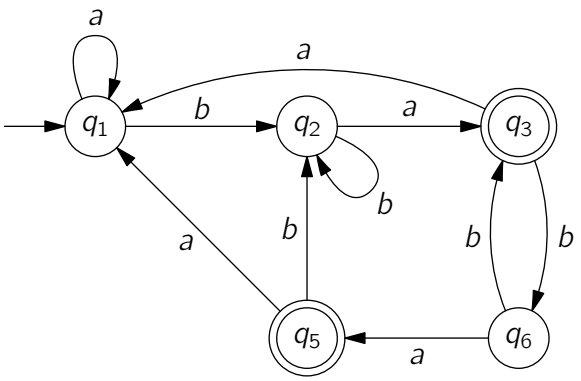
Lösung:

2	$\times_1$				
3	$\times_0$	$\times_0$			
4	$(3,5)$	$\times_3$	$\times_0$		
5	$\times_0$	$\times_0$	$(2,6)$, $\times_6$	$\times_0$	
6	$\times_5$	$(3,5)$, $\times_6$	$\times_0$	$\times_7$	$\times_0$
	1	2	3	4	5

In einem ersten Schritt markieren wir mit $\times_0$ all die Zellen, die Paaren von Zuständen $\{p, q\}$ entsprechen, für die $p \in F$ und $q \notin F$ gilt.
Nun durchlaufen wir alle Paare $\{p, q\}$ von Zuständen, indem wir die Tabelle von oben nach unten und von links nach rechts durchlaufen. Dabei überprüfen wir jeweils, ob $\delta(p, a)$ und $\delta(q, a)$ bereits unterscheidbar sind. Ist dies der Fall, fügen wir ein $\times_i$ ein, ansonsten markieren wir in der Zelle für das Paar $\{\delta(p, a), \delta(q, a)\}$, dass $\{p, q\}$ dann unterscheidbar sind, wenn $\{\delta(p, a), \delta(q, a)\}$ unterscheidbar sind:

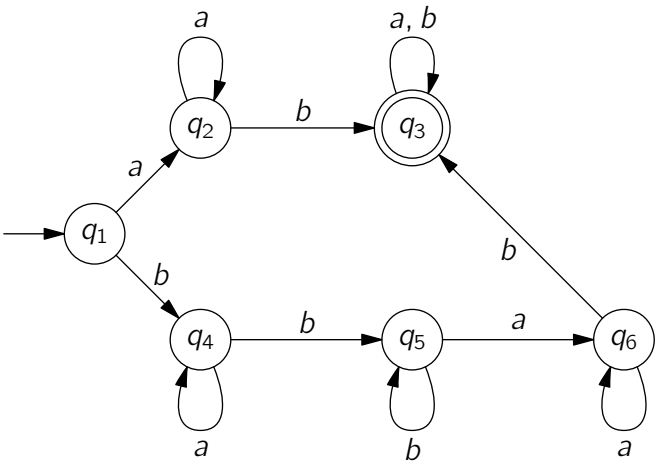
1. Paar $\{q_1, q_2\}$:
   - *a* $\delta(q_1, a) = q_4$, $\delta(q_2, a) = q_3$, $\{q_2, q_3\}$ unterscheidbar $\Rightarrow$ Füge $\times_1$ in Zelle $\{q_1, q_2\}$ ein.
2. Paar $\{q_1, q_4\}$:
   - *a* $\delta(q_1, a) = q_4$, $\delta(q_4, a) = q_4$, $\{q_4, q_4\}$ nicht unterscheidbar.
   - *b* $\delta(q_1, b) = q_2$, $\delta(q_4, b) = q_2$, $\{q_2, q_2\}$ nicht unterscheidbar.
3. Paar $\{q_2, q_4\}$:
   - *a* $\delta(q_2, a) = q_3$, $\delta(q_4, a) = q_4$, $\{q_3, q_4\}$ unterscheidbar $\Rightarrow$ Füge $\times_3$ in Zelle $\{q_2, q_4\}$ ein.
4. Paar $\{q_3, q_5\}$:
   - *a* $\delta(q_3, a) = q_1$, $\delta(q_5, a) = q_4$, $\{q_1, q_4\}$ nicht unterscheidbar. $\Rightarrow$ Füge $(3, 5)$ in Zelle $\{q_1, q_4\}$ ein.
   - *b* $\delta(q_3, b) = q_6$, $\delta(q_5, b) = q_2$, $\{q_2, q_6\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(3, 5)$ in Zelle $\{q_2, q_6\}$ ein.
5. Paar $\{q_1, q_6\}$:
   - *a* $\delta(q_1, a) = q_4$, $\delta(q_6, a) = q_5$, $\{q_1, q_5\}$ unterscheidbar $\Rightarrow$ Füge $\times_5$ in Zelle $\{q_1, q_6\}$ ein.
6. Paar $\{q_2, q_6\}$:
   - *a* $\delta(q_2, a) = q_3$, $\delta(q_6, a) = q_5$, $\{q_3, q_5\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(2, 6)$ in Zelle $\{q_3, q_5\}$ ein.
   - *b* $\delta(q_2, b) = q_2$, $\delta(q_6, b) = q_3$, $\{q_2, q_3\}$ unterscheidbar $\Rightarrow$ Füge $\times_6$ in Zelle $\{q_2, q_6\}$ ein. Folge Verweis auf $\{q_3, q_5\}$ und füge dort ebenfalls $\times_6$ ein.
7. Paar $\{q_4, q_6\}$:
   - *a* $\delta(q_4, a) = q_4$, $\delta(q_6, a) = q_5$, $\{q_4, q_5\}$ unterscheidbar $\Rightarrow$ Füge $\times_7$ in Zelle $\{q_4, q_6\}$ ein.

Insgesamt können wir also schließen, dass nur die Zustände q_1 und q_4 verschmolzen werden können. Es ergibt sich der folgende minimale Automat:



Hausaufgabe 2 (DFA Minimierung): (4 Punkte)

Verwenden Sie den schnellen Markierungsalgorithmus aus der Vorlesung, um folgenden DFA zu minimieren. Geben Sie in der Lösung die beim Anwenden des Algorithmus entstehende Tabelle und den minimalen Automaten an.



Lösung:

2	$\times_1$				
3	$\times_0$	$\times_0$			
4	$\times_3$	$(1,4),\ \times_3$	$\times_0$		
5	$\times_9$	$\times_5$	$\times_0$	$(1,4),\ (1,5),\ \times_9$	
6	$\times_7$	$(1,5),(2,5),(1,6)$	$\times_0$	$(4,5),\ \times_9$	$\times_{10}$
	1	2	3	4	5

In einem ersten Schritt markieren wir mit $\times_0$ all die Zellen, die Paaren von Zuständen $\{p, q\}$ entsprechen, für die $p \in F$ und $q \notin F$ gilt.
Nun durchlaufen wir alle Paare $\{p, q\}$ von Zuständen, indem wir die Tabelle von oben nach unten und von links nach rechts durchlaufen. Dabei überprüfen wir jeweils, ob $\delta(p, a)$ und $\delta(q, a)$ bereits unterscheidbar sind. Ist dies der Fall, fügen wir ein $\times_i$ ein, ansonsten markieren wir in der Zelle für das Paar $\{\delta(p, a), \delta(q, a)\}$, dass $\{p, q\}$ dann unterscheidbar sind, wenn $\{\delta(p, a), \delta(q, a)\}$ unterscheidbar sind:

1. Paar $\{q_1, q_2\}$:
   - *a* $\delta(q_1, a) = q_2$, $\delta(q_2, a) = q_2$, $\{q_2, q_2\}$ nicht unterscheidbar.
   - *b* $\delta(q_1, b) = q_4$, $\delta(q_2, b) = q_3$, $\{q_3, q_4\}$ unterscheidbar $\Rightarrow$ Füge $\times_1$ in Zelle $\{q_1, q_2\}$ ein.
2. Paar $\{q_1, q_4\}$:
   - *a* $\delta(q_1, a) = q_2$, $\delta(q_4, a) = q_4$, $\{q_2, q_4\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(1, 4)$ in Zelle $\{q_2, q_4\}$ ein.
   - *b* $\delta(q_1, b) = q_4$, $\delta(q_4, b) = q_5$, $\{q_4, q_5\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(1, 4)$ in Zelle $\{q_4, q_5\}$ ein.
3. Paar $\{q_2, q_4\}$:
   - *a* $\delta(q_2, a) = q_2$, $\delta(q_4, a) = q_4$, $\{q_2, q_4\}$ noch nicht unterscheidbar.
   - *b* $\delta(q_2, b) = q_3$, $\delta(q_4, b) = q_5$, $\{q_3, q_5\}$ unterscheidbar $\Rightarrow$ Füge $\times_3$ in Zelle $\{q_2, q_4\}$ ein. Folge Verweis auf $\{q_1, q_4\}$ und füge dort ebenfalls $\times_3$ ein.
4. Paar $\{q_1, q_5\}$:
   - *a* $\delta(q_1, a) = q_2$, $\delta(q_5, a) = q_6$, $\{q_2, q_6\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(1, 5)$ in Zelle $\{q_2, q_6\}$ ein.
   - *b* $\delta(q_1, b) = q_4$, $\delta(q_5, b) = q_5$, $\{q_4, q_5\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(1, 5)$ in Zelle $\{q_4, q_5\}$ ein.
5. Paar $\{q_2, q_5\}$:
   - *a* $\delta(q_2, a) = q_2$, $\delta(q_5, a) = q_6$, $\{q_2, q_6\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(2, 5)$ in Zelle $\{q_2, q_6\}$ ein.

   b $\delta(q_2, b) = q_3$, $\delta(q_5, b) = q_5$, $\{q_3, q_5\}$ unterscheidbar $\Rightarrow$ Füge $\times_5$ in Zelle $\{q_2, q_5\}$ ein.

6. Paar $\{q_4, q_5\}$:
   a $\delta(q_4, a) = q_4$, $\delta(q_5, a) = q_6$, $\{q_4, q_6\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(4, 5)$ in Zelle $\{q_4, q_6\}$ ein.
   b $\delta(q_4, b) = q_5$, $\delta(q_5, b) = q_5$, $\{q_5, q_5\}$ nicht unterscheidbar.

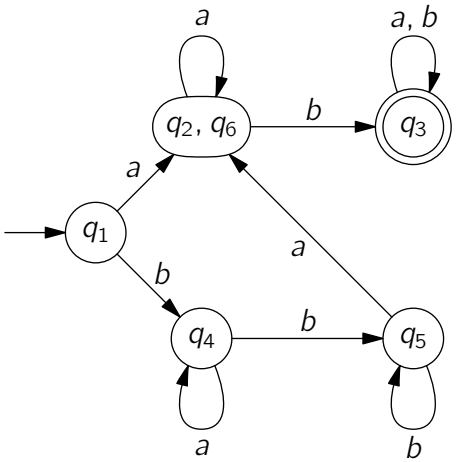
7. Paar $\{q_1, q_6\}$:
   a $\delta(q_1, a) = q_2$, $\delta(q_6, a) = q_6$, $\{q_2, q_6\}$ noch nicht unterscheidbar $\Rightarrow$ Füge $(1, 6)$ in Zelle $\{q_2, q_6\}$ ein.
   b $\delta(q_1, b) = q_4$, $\delta(q_6, b) = q_3$, $\{q_4, q_3\}$ unterscheidbar $\Rightarrow$ Füge $\times_7$ in Zelle $\{q_1, q_6\}$ ein.

8. Paar $\{q_2, q_6\}$:
   a $\delta(q_2, a) = q_2$, $\delta(q_6, a) = q_6$, $\{q_2, q_6\}$ noch nicht unterscheidbar.
   b $\delta(q_2, b) = q_3$, $\delta(q_6, b) = q_3$, $\{q_3, q_3\}$ nicht unterscheidbar.

9. Paar $\{q_4, q_6\}$:
   a $\delta(q_4, a) = q_4$, $\delta(q_6, a) = q_6$, $\{q_4, q_6\}$ noch nicht unterscheidbar.
   b $\delta(q_4, b) = q_5$, $\delta(q_6, b) = q_3$, $\{q_3, q_5\}$ unterscheidbar $\Rightarrow$ Füge $\times_9$ in Zelle $\{q_4, q_6\}$ ein. Folge Verweis auf $\{q_4, q_5\}$ und füge dort ebenfalls $\times_9$ ein. Folge Verweisen auf $\{q_1, q_4\}$ und $\{q_1, q_5\}$ und füge dort ebenfalls $\times_9$ ein (wenn noch kein $\times$ vorhanden).

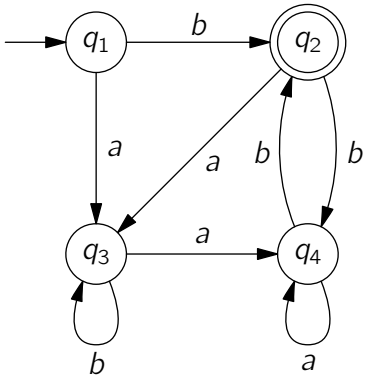
10. Paar $\{q_5, q_6\}$:
    a $\delta(q_5, a) = q_6$, $\delta(q_6, a) = q_6$, $\{q_6, q_6\}$ nicht unterscheidbar.
    b $\delta(q_5, b) = q_5$, $\delta(q_6, b) = q_3$, $\{q_3, q_5\}$ unterscheidbar $\Rightarrow$ Füge $\times_{10}$ in Zelle $\{q_5, q_6\}$ ein.

Insgesamt können wir also schließen, dass nur die Zustände q_2 und q_6 verschmolzen werden können. Es ergibt sich der folgende minimale Automat:



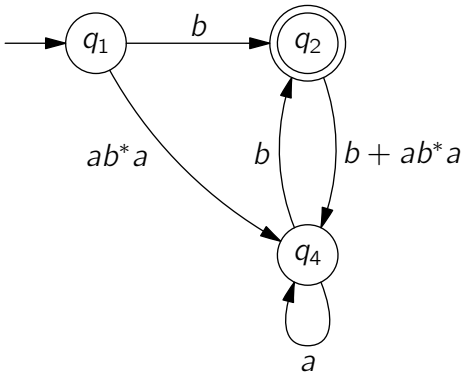
Tutoraufgabe 3 (DFA in regulären Ausdruck umwandeln):

Wandeln Sie folgenden DFA in einen regulären Ausdruck um, indem Sie Zustände schrittweise entfernen und die betroffenen Kanten durch reguläre Ausdrücke ersetzen. Geben Sie nach jedem Schritt den resultierenden Automaten und zum Schluss den abgelesenen regulären Ausdruck an.

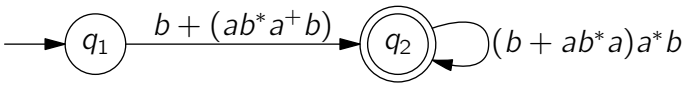


Lösung:

Im ersten Schritt wird der Zustand q_3 entfernt.



Im resultierenden Automaten wird nun der Zustand q_4 entfernt.

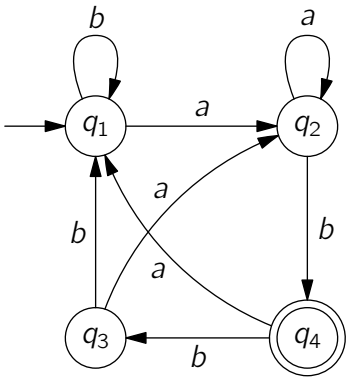


Nun läßt sich das Ergebnis ablesen:

$$(b + (ab^*a^+b))((b + ab^*a)a^*b)^*$$

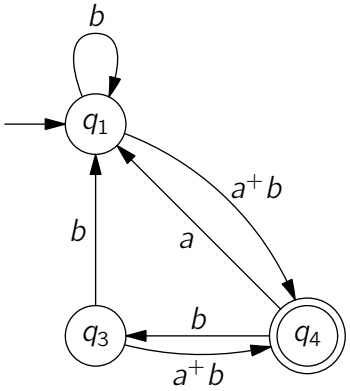
Hausaufgabe 4 (DFA in regulären Ausdruck umwandeln): (4 Punkte)

Wandeln Sie folgenden DFA in einen regulären Ausdruck um, indem Sie Zustände schrittweise entfernen und die betroffenen Kanten durch reguläre Ausdrücke ersetzen. Geben Sie nach jedem Schritt den resultierenden Automaten und zum Schluss den abgelesenen regulären Ausdruck an.

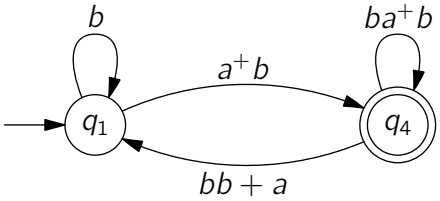


Lösung:

Im ersten Schritt entfernen wir den Zustand q_2.



Nach Entfernen von q_3 ergibt sich folgender Automat:



Der resultierende Ausdruck ist:

$$b^*a^+b(ba^+b + (bb + a)b^*a^+b)^*$$

Tutoraufgabe 5 (Pumping Lemma):

Weisen Sie mit Hilfe des Pumping Lemmas nach, dass die folgenden Sprachen nicht regulär sind:

a) $L_1 := \{0^k 1^{2k} \mid k \in \mathbb{N}_0\}$ über dem Alphabet $\{0, 1\}$.

b) $L_2 := \{a^{k^2} \mid k \in \mathbb{N}_0\}$ über dem Alphabet $\{a\}$.

c) Wir erinnern an die Funktion $weight : \{0,1\}^* \to \mathbb{N}_0$ aus Tutoraufgabe 3(b) auf Übungsblatt 4:

$$\begin{aligned} weight(\epsilon) &:= 0 \\ weight(w \cdot a) &:= weight(w) + a \qquad\qquad a \in \{0,1\}, w \in \{0,1\}^* \end{aligned}$$

$L_3 := \{w \in \{0,1\}^* \mid \exists k \in \mathbb{N}_0 : weight(w) = 2^k\}$ über dem Alphabet $\{0,1\}$.

d) $L_4 := \{w \in \{0,1\}^* \mid w$ enthält gleich viele 0en und 1en$\}$ über dem Alphabet $\{0,1\}$.

Lösung:

Alle folgenden Lösungen sind nach demselben Schema aufgebaut. Wir nehmen jeweils an, dass die Sprache L_j regulär ist. Dann muss es nach dem Pumping Lemma eine Länge $n \in \mathbb{N}_{>0}$ geben, so dass es für jedes Wort $w \in L_j$ mit $|w| \geq n$ eine Zerlegung $w = xyz$ gibt, so dass $|xy| \leq n$, $y \neq \epsilon$ und $xy^iz \in L_j$ für alle $i \in \mathbb{N}_0$ gilt. Wenn wir für jede solche Zerlegung ein i angeben können, so dass $xy^iz \notin L_j$ gilt, muss die Annahme, dass L_j regulär ist, falsch sein.

a) Wir wählen das Wort $w = 0^n1^{2n} \in L_1$. Nach Pumping Lemma muss es also eine Zerlegung xyz von w geben, so dass $xy^iz \in L_1$ für alle $i \in \mathbb{N}_0$. Da $|xy| \leq n$ gelten muss, gibt es ein $\ell \in \mathbb{N}_0$ mit $\ell \leq n$, so dass $y = 0^\ell$ ist.

Es muss auch $xy^0z = xz = 0^{n-\ell}1^{2n} \in L_1$ gelten. Da aber $y \neq \epsilon$ gelten muss, ist $\ell > 0$ und daher $0^{n-\ell}1^{2n} \notin L_1$. Dies ist ein Widerspruch zur Annahme und wir können schließen, dass L_1 nicht regulär ist.

b) Wir wählen das Wort $w = a^{n^2} \in L_2$. Nach Pumping Lemma muss es also eine Zerlegung xyz von w geben, so dass $xy^iz \in L_2$ für alle $i \in \mathbb{N}_0$. Sei $y = a^\ell$ für ein $\ell \in \mathbb{N}_0$ mit $1 \leq \ell \leq n$.

Es muss auch $xy^2z = a^{n^2+\ell} \in L_2$ gelten. Da aber $y \neq \epsilon$ gelten muss, ist $\ell > 0$ und daher $n^2 + \ell > n^2$. Außerdem gilt $\ell \leq n$ (da $|xy| \leq n$), also auch $n^2 + \ell < n^2 + 2n + 1 = (n+1)^2$. Damit gilt $xy^2z \notin L_2$, was ein Widerspruch zur Annahme ist. Wir können daher schließen, dass L_2 nicht regulär ist.

c) Wir wählen das Wort $w = 1^{2^n} \in L_3$. Nach Pumping Lemma muss es also eine Zerlegung xyz von w geben, so dass $xy^iz \in L_3$ für alle $i \in \mathbb{N}_0$. Sei $y = 1^\ell$ für ein $\ell \in \mathbb{N}_0$ mit $1 \leq \ell \leq n$.

Es muss auch $xy^2z = 1^{2^n+\ell} \in L_3$ gelten. Da aber $y \neq \epsilon$ gelten muss, ist $\ell > 0$ und daher $2^n + \ell > 2^n$. Außerdem muss $|xy| \leq n$ gelten. Da $n > 0$ gilt, impliziert das $2^n + \ell \leq 2^n + n < 2^{n+1}$. Daher ist $|xy^2z|$ keine Zweierpotenz und damit gilt $xy^2z \notin L_3$. Dies ist ein Widerspruch zur Annahme und wir können schließen, dass L_3 nicht regulär ist.

d) Wir wählen das Wort $w = 0^n1^n \in L_4$. Nach Pumping Lemma muss es also eine Zerlegung xyz von w geben, so dass $xy^iz \in L_4$ für alle $i \in \mathbb{N}_0$. Sei $y = 0^\ell$ für ein $\ell \in \mathbb{N}_0$ mit $1 \leq \ell \leq n$.

Es muss auch $xy^0z = 0^{n-\ell}1^n \in L_4$ gelten. Da aber $y \neq \epsilon$ gelten muss, ist $\ell > 0$ und daher $n - \ell < n$, also insbesondere $n - \ell \neq n$ und damit gilt $xy^0z \notin L_4$. Dies ist ein Widerspruch zur Annahme und wir können schließen, dass L_4 nicht regulär ist.

Hausaufgabe 6 (Pumping Lemma): (2 + 2 + 2 + 2 = 8 Punkte)

Weisen Sie mit Hilfe des Pumping Lemmas nach, dass die folgenden Sprachen nicht regulär sind:

a) $L_5 := \{a^k b^m c^{k+m} \mid k, m \in \mathbb{N}_0\}$ über dem Alphabet $\{a, b, c\}$.

b) $L_6 := \{1^k 0 1^k \mid k \in \mathbb{N}_0\}$ über dem Alphabet $\{0, 1\}$.

c) $L_7 := \{a^p \mid p \text{ Primzahl}\}$ über dem Alphabet $\{a\}$.

d) Wir erinnern an die Funktion $rev : \Sigma^* \to \Sigma^*$ aus Hausaufgabe 2(a) auf Übungsblatt 3:

$$\begin{aligned} rev(\epsilon) &:= \epsilon \\ rev(w \cdot a) &:= a \cdot rev(w) \qquad\qquad a \in \Sigma, w \in \Sigma^* \end{aligned}$$

$L_8 := \{w \cdot rev(w) \mid w \in \{0, 1\}^*\}$ über dem Alphabet $\{0, 1\}$.

Lösung:

Alle folgenden Lösungen sind nach demselben Schema aufgebaut. Wir nehmen jeweils an, dass die Sprache L_j regulär ist. Dann muss es nach dem Pumping Lemma eine Länge $n \in \mathbb{N}_{>0}$ geben, so dass für jedes Wort $w \in L_j$ mit $|w| \geq n$ eine Zerlegung $w = xyz$ gibt, so dass $|xy| \leq n$, $y \neq \epsilon$ und $xy^i z \in L_j$ für alle $i \in \mathbb{N}_0$ gilt. Wenn wir für jede solche Zerlegung ein i angeben können, so dass $xy^i z \notin L_j$ gilt, muss die Annahme, dass L_j regulär ist, falsch sein.

a) Wir wählen das Wort $w = a^n b^n c^{2n} \in L_5$. Nach Pumping Lemma muss es also eine Zerlegung xyz von w geben, so dass $xy^i z \in L_5$ für alle $i \in \mathbb{N}_0$. Da $|xy| \leq n$ gelten muss, gibt es ein $\ell \in \mathbb{N}$, mit $\ell \leq n$ so dass $y = a^\ell$ ist.

Es muss auch $xy^0 z = xz = a^{n-\ell} b^n c^{2n} \in L_5$ gelten. Da aber $y \neq \epsilon$ gelten muss, ist $\ell > 0$ und daher $a^{n-\ell} b^n c^{2n} \notin L_5$. Dies ist ein Widerspruch zur Annahme und wir können schließen, dass L_5 nicht regulär ist.

b) Wir wählen das Wort $w = 1^n 0 1^n \in L_6$. Nach Pumping Lemma muss es also eine Zerlegung xyz von w geben, so dass $xy^i z \in L_6$ für alle $i \in \mathbb{N}_0$. Sei $y = 1^\ell$ für ein $\ell \in \mathbb{N}_0$ mit $1 \leq \ell \leq n$.

Es muss auch $xy^0 z = 1^{n-\ell} 0 1^n \in L_6$ gelten. Da aber $y \neq \epsilon$ gelten muss, ist $\ell > 0$ und daher $1^{n-\ell} 0 1^n \notin L_6$ Dies ist ein Widerspruch zur Annahme und wir können schließen, dass L_6 nicht regulär ist.

c) Wir wählen das Wort $w = a^p \in L_7$ für eine Primzahl $p > n$. Nach Pumping Lemma muss es also eine Zerlegung xyz von w geben, so dass $xy^i z \in L_7$ für alle $i \in \mathbb{N}_0$. Sei $y = a^\ell$ für ein $\ell \in \mathbb{N}_0$ mit $1 \leq \ell \leq n$.

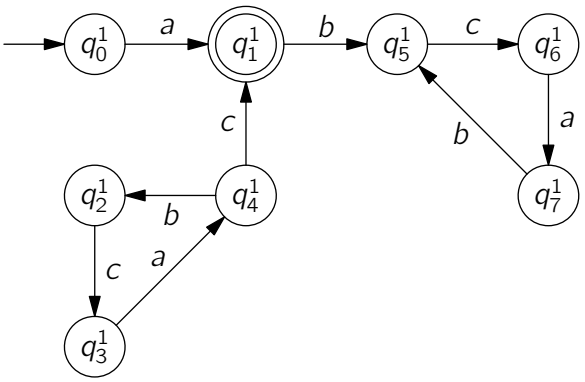
Es muss auch $xy^{p+1} z = a^{|x|+\ell*(p+1)+|z|} = a^{|x|+\ell+|z|+\ell*p} \in L_7$ gelten. Da aber $y \neq \epsilon$ gelten muss, ist $\ell > 0$. Außerdem gilt nach Konstruktion der Zerlegung von w, dass $p = |x|+\ell+|z|$. Also ist $xy^{p+1} z = a^{(\ell+1)*p} \notin L_7$. Dies ist ein Widerspruch zur Annahme und wir können schließen, dass L_7 nicht regulär ist.

d) Wir wählen das Wort $w = 0^n 1^{2n} 0^n \in L_8$. Nach Pumping Lemma muss es also eine Zerlegung xyz von w geben, so dass $xy^i z \in L_8$ für alle $i \in \mathbb{N}_0$. Sei $y = 0^\ell$ für ein $\ell \in \mathbb{N}_0$ mit $1 \leq \ell \leq n$.

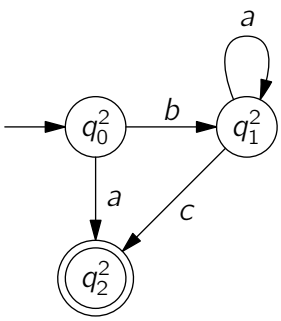
Es muss auch $xy^0 z = 0^{n-\ell} 1^{2n} 0^n \in L_8$ gelten. Da aber $y \neq \epsilon$ gelten muss, ist $\ell > 0$ und daher $n-\ell < n$, also insbesondere $n - \ell \neq n$ und damit gilt $xy^0 z \notin L_8$. Dies ist ein Widerspruch zur Annahme und wir können schließen, dass L_8 nicht regulär ist.

Tutoraufgabe 7 (Entscheidungsverfahren):

a) Sei $\Sigma = \{a, b, c\}$ ein Alphabet. Betrachten Sie die folgenden NFAs M_1 und M_2.



M_1



M_2

Entscheiden Sie mit dem in der Vorlesung vorgestellten Verfahren, welche der zwei Sprachen $L(M_1)$ und $L(M_2)$ endlich sind.

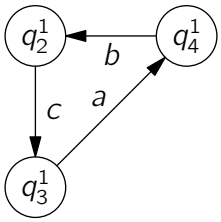
b) Die symmetrische Differenz (bezeichnet mit dem $\oplus$-Operator) zweier Mengen A und B ist definiert als:

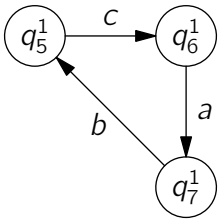
$A \oplus B = (A \cup B) \setminus (A \cap B)$.

Geben Sie ein vollständig automatisierbares Entscheidungsverfahren mithilfe der in der Vorlesung vorgestellten Konzepte an, welches bei Eingabe dreier regulärer Sprachen (spezifiziert durch reguläre Ausdrücke, DFAs, NFAs oder ϵ-NFAs) L_1, L_2, L_3 entscheidet, ob $L_1 \oplus L_2 = L_3$ gilt (d.h. das Verfahren entscheidet das Problem der symmetrischen Differenz regulärer Sprachen).

Lösung:

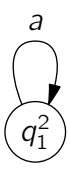
a) M_1 hat die folgenden zwei starken Zusammenhangskomponenten.





Während die erste nicht vom Startzustand aus erreichbar ist, ist von der zweiten aus kein Endzustand erreichbar. Daher ist $L(M_1)$ endlich.

M_2 hat die folgende starke Zusammenhangskomponente.



Diese ist vom Startzustand aus erreichbar und von ihr aus ist ein Endzustand erreichbar. Daher ist $L(M_2)$ nicht endlich.

b) Es gilt:

$$\begin{aligned} L_1 \oplus L_2 = L_3 &\Leftrightarrow (L_1 \cup L_2) \setminus (L_1 \cap L_2) = L_3 \\ &\Leftrightarrow (L_1 \cup L_2) \cap (\Sigma^* \setminus (L_1 \cap L_2)) = L_3 \end{aligned}$$

Falls L_1 und/oder L_2 durch einen regulären Ausdruck spezifiziert sind, wird dieser durch die Thompson-Konstruktion in einen ϵ-NFA überführt. Anschließend haben wir zwei Automaten (DFAs, NFAs oder ϵ-NFAs) M_1 und M_2 mit $L_1 = L(M_1)$ und $L_2 = L(M_2)$.

Wir konstruieren einen ϵ-NFA $M_\cup$ mit $L(M_\cup) = L_1 \cup L_2$ durch Übernahme aller Zustände und Transitionen aus M_1 und M_2 und Einführung eines neuen Zustands als Startzustand, der mit ϵ-Transitionen die ehemaligen Startzustände von M_1 und M_2 erreicht.

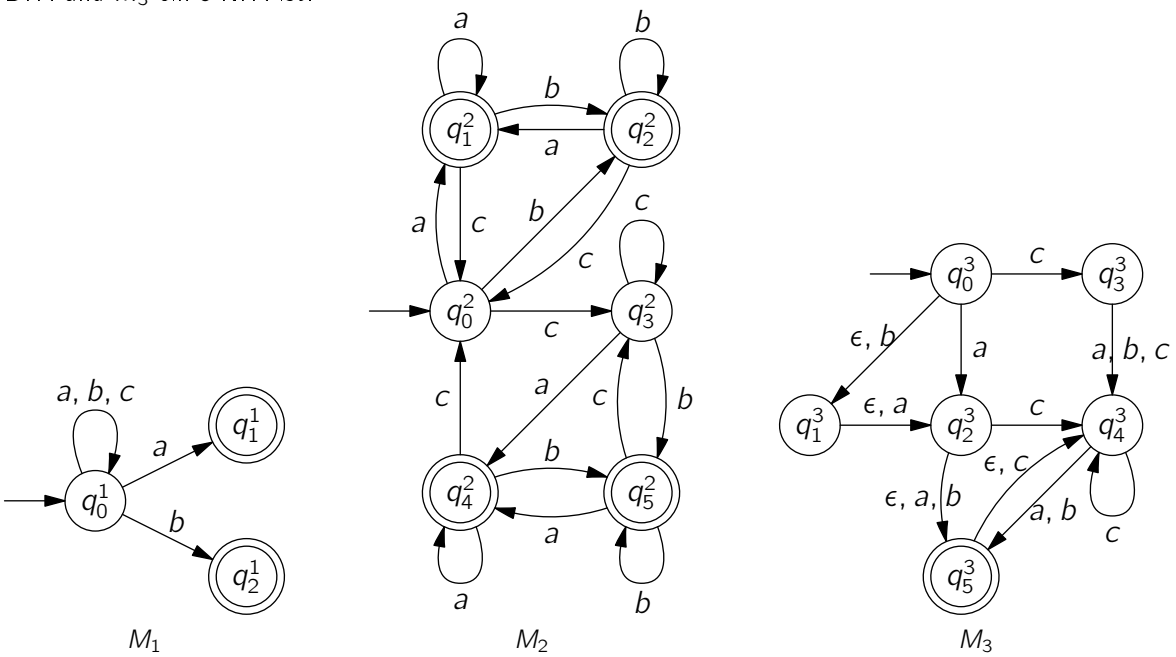
Nun überführen wir mittels ϵ-Kanten-Elimination und Potenzmengenkonstruktion die Automaten M_1, M_2 und $M_\cup$ in DFAs M_1', M_2' und $M_\cup'$.

Wir konstruieren den Produktautomaten $M_\cap$ der beiden Automaten M_1' und M_2', welcher die Sprache $L_1 \cap L_2$ erkennt. Da M_1' und M_2' DFAs sind, ist $M_\cap$ ebenfalls ein DFA. Wir konstruieren den Komplementärautomaten M_C von $M_\cap$, welcher die Sprache $\Sigma^* \setminus (L_1 \cap L_2)$ erkennt. M_C ist ebenfalls ein DFA.

Schließlich bilden wir den Produktautomaten M der beiden DFAs $M_\cup'$ und M_C, welcher die Sprache $(L_1 \cup L_2) \cap (\Sigma^* \setminus (L_1 \cap L_2))$ erkennt. Damit haben wir das Problem der symmetrischen Differenz regulärer Sprachen auf das Äquivalenzproblem reduziert und können anschließend das entsprechende Verfahren aus der Vorlesung benutzen, um $L(M) = L_3$ zu entscheiden.

Hausaufgabe 8 (Entscheidungsverfahren): (10 Punkte)

Sei $\Sigma = \{a, b, c\}$ ein Alphabet. Betrachten Sie die folgenden Automaten M_1, M_2, M_3, wobei M_1 ein NFA, M_2 ein DFA und M_3 ein ϵ-NFA ist.

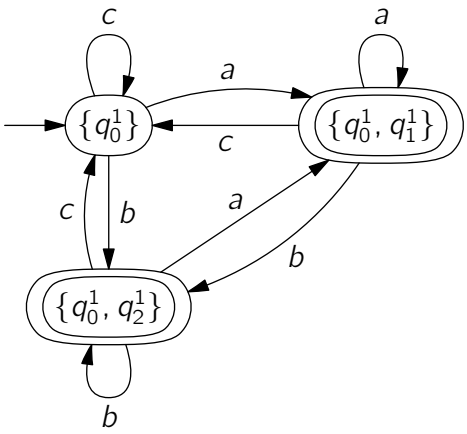


Entscheiden Sie mit dem in der Vorlesung vorgestellten Verfahren zur Lösung des Äquivalenzproblems, welche der drei Sprachen $L(M_1)$, $L(M_2)$ und $L(M_3)$ gleich sind. Geben Sie dabei nach jeder Transformation (wie z.B.

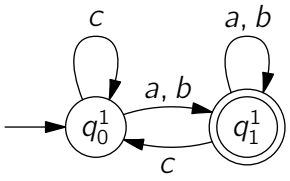
Thompson-Konstruktion, ϵ-Kanten-Elimination, Potenzmengenkonstruktion, Minimierung) den jeweils resultierenden Automaten an.

Lösung:

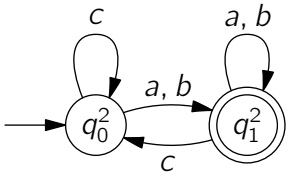
Wir wenden zuerst die Potenzmengenkonstruktion auf M_1 an und erhalten den folgenden Automaten M_1'.



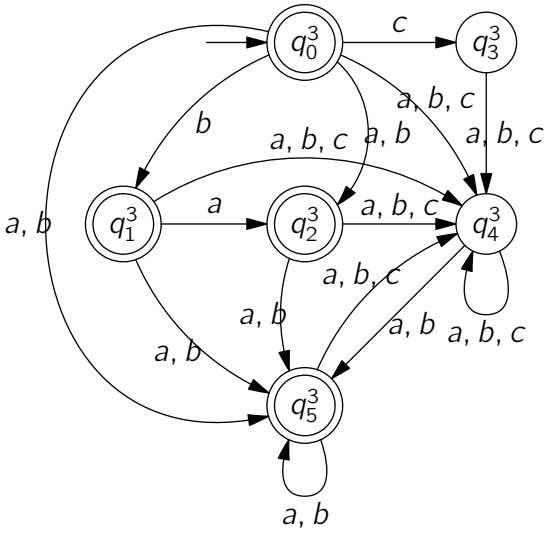
Minimierung von M_1' liefert den folgenden minimalen DFA M_1''.



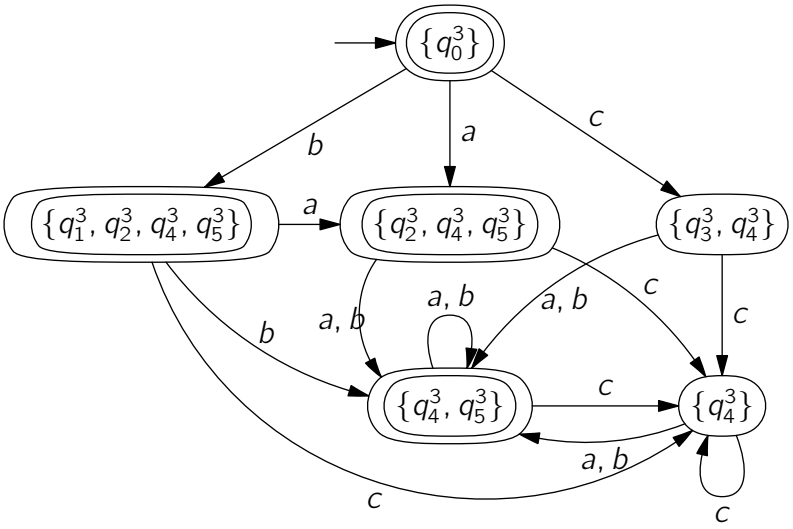
Danach minimieren wir den DFA M_2 und erhalten den folgenden minimalen DFA M_2'.



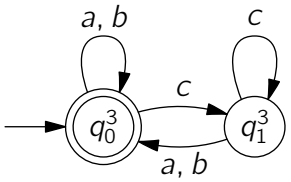
Nun entfernen wir die ϵ-Transitionen aus M_3 und erhalten den folgenden NFA M_3'.



Nach Anwendung der Potenzmengenkonstruktion erhalten wir den folgenden DFA M_3''.



Minimierung von M_3'' führt zu dem folgenden minimalen DFA M_3'''.



Schließlich überprüfen wir die minimalen DFAs auf Isomorphie und erkennen, dass $L(M_1) = L(M_2) \neq L(M_3)$.

Prof. Dr. J. Giesl

M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Mi., 16.06.2010 im Tutorium abgegeben werden. Alternativ ist es bis 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Da am Mi., 9.6.2010 der Studieninformationstag stattfindet, fallen die Tutorien an diesem Tag aus.
- Die **Tutoraufgaben** werden in der kommenden Globalübung am 11.06.2010 vorgerechnet.
- Die **Einsicht** in die Präsenzübungs-Ergebnisse wird am **11.06.2010 zwischen 8:00 und 10:00 in 2356|056** (Ahornstraße 55, Raum 5056) stattfinden. Studenten mit einer Matrikelnummer kleiner als 294000 werden gebeten, zwischen 8:00 und 9:00 zu kommen, alle anderen zwischen 9:00 und 10:00.
- Die **Rückgabe** der korrigierten Präsenzübungen erfolgt in der Einsicht und in den Tutorien am 16.6.2010. Nach der Rückgabe sind **keine Eingaben zur Korrektur möglich**!

Hausaufgabe 1 (Eindeutigkeit): (1 + 2 + 2 = 5 Punkte)

Sei $G = (N, T, P, S)$ eine Grammatik mit $N = \{S, A, B\}, T = \{a, b\}$ und P wie folgt:

$$S \to AB \mid SAB$$
$$A \to Aa \mid \epsilon$$
$$B \to Bb \mid \epsilon$$

a) Geben Sie die von G erzeugte Sprache $L(G)$ an (ohne Beweis).

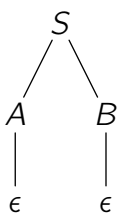
b) Beweisen Sie, dass G nicht eindeutig ist.

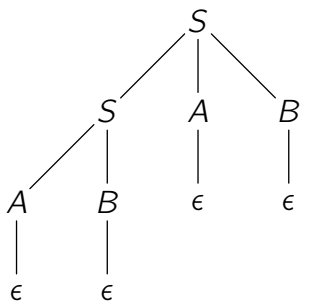
c) Geben Sie eine eindeutige kontextfreie Grammatik G' an mit $L(G') = L(G)$ (ohne Beweis).

Lösung:

a) Die von G erzeugte Sprache ist $L(G) = L((a + b)^*)$.

b) Bereits das Wort ϵ lässt sich mit verschiedenen Ableitungsbäumen erzeugen:





Damit ist bewiesen, dass G nicht eindeutig ist.

c) Wir definieren $G' := (\{S\}, T, P', S)$ mit P' wie folgt:

$$S \to aS \mid bS \mid \epsilon$$

Hausaufgabe 2 (Grammatiken und reguläre Sprachen): (2 + 4 + 2 = 8 Punkte)

Sei $G = (N, T, P, S)$ eine Grammatik. Wenn für G gilt, dass

$$\forall (A \to \alpha) \in P : (A \in N) \wedge (\alpha \in \{\epsilon\} \cup T \cup T \cdot N)$$

dann nennt man G **linkslineare** Grammatik. In einer solchen Grammatik sind also alle Produktionen so gestaltet, dass sie entweder ein Nonterminalsymbol löschen, es in ein Terminalsymbol oder in ein Terminalsymbol und ein weiteres Nonterminalsymbol umwandeln.

a) Betrachten Sie die linkslineare Grammatik $G := (N, T, P, S)$ mit $N := \{S, A, B\}$, $T := \{a, b\}$ und P wie folgt:

$$S \to aA \mid bB$$
$$A \to aA \mid \epsilon$$
$$B \to aB \mid bA$$

Geben Sie Ableitungsbäume für die Worte $w_1 = aa$ und $w_2 = baaba$ an.

b) Nehmen Sie an, dass $G = (N, T, P, S)$ eine beliebige linkslineare Grammatik ist. Zur Vereinfachung fordern wir zusätzlich, dass P keine Regeln der Form $A \to b$ enthält, sondern alle Regeln die Form $A \to \varepsilon$ oder $A \to bC$ für beliebige $A, C \in N, b \in T$ haben. [1]

Definieren Sie nun einen NFA M_G, für den $L(M_G) = L(G)$ gilt. Sie brauchen diese Gleichheit nicht zu beweisen.

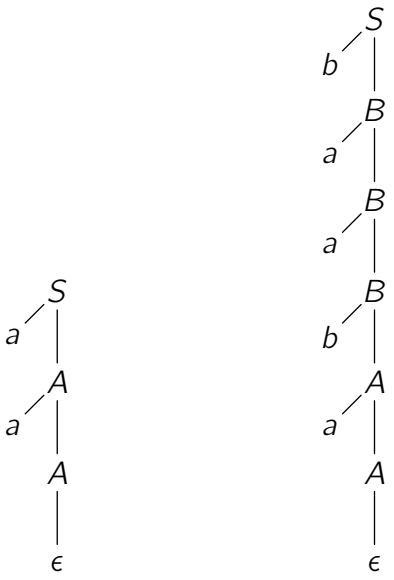
Hinweis: Sie sollten als Zustände für M_G die Nonterminalsymbole N wählen.

c) Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein NFA. Geben Sie eine linkslineare Grammatik G_M an, so dass $L(G_M) = L(M)$. Sie brauchen diese Gleichheit nicht zu beweisen.

Lösung:

a) Die Ableitungsbäume können wie folgt dargestellt werden:

[1] Dies ist keine wesentliche Einschränkung, denn mit Hilfe eines zusätzlichen Nonterminalsymbols E mit der Produktion $E \to \epsilon$ können Regeln der Form $A \to b$ in $A \to bE$ umgewandelt werden. Damit kann die selbe Sprache wie mit Hilfe der ursprünglichen Grammatik erkannt werden.



b) Wir definieren $M_G := (N, T, \delta, S, F)$ mit

$$\delta(A, b) = \{C \mid (A \to bC) \in P\} \qquad A, C \in N, b \in T$$

und $F = \{A \mid (A \to \epsilon) \in P\}$.

c) Wir definieren $G_M := (Q, \Sigma, P, q_0)$ mit

$$\begin{aligned} P := \quad & \{q \to aq' \mid q, q' \in Q, a \in \Sigma, q' \in \delta(q, a)\} \\ & \cup \{q \to \epsilon \mid q \in F\} \end{aligned}$$

Tutoraufgabe 3 (Kontextfreie Grammatiken):

Die Dyck-Sprache ist die Sprache aller korrekt geklammerten Ausdrücke. Z.B. sind "()", "(()())" und "()(())" korrekt geklammerte Ausdrücke, während ")(" und "(()" keine korrekt geklammerten Ausdrücke sind. Zur Vereinfachung der Lesbarkeit folgender Definitionen verwenden wir die Zeichen a und b anstelle von (und). Formal ist die Dyck-Sprache über dem Alphabet $\Sigma = \{a, b\}$ dann folgendermaßen definiert:
$D = \{w \in \Sigma^* \mid \sharp_a(w) = \sharp_b(w) \wedge \forall u, v \in \Sigma^* : u \cdot v = w \Rightarrow \sharp_a(u) \geq \sharp_b(u)\}$, wobei die Funktionen $\sharp_a$ und $\sharp_b$ die Anzahl der vorkommenden Zeichen a bzw. b in einem Wort zählen. Formal:
$\sharp_a : \Sigma^* \to \mathbb{N}_0$ mit $\sharp_a(\epsilon) = 0$, $\sharp_a(w \cdot a) = \sharp_a(w) + 1$ und $\sharp_a(w \cdot b) = \sharp_a(w)$. Die Funktion $\sharp_b$ ist analog definiert.
Geben Sie eine kontextfreie Grammatik an, die D erzeugt (Sie brauchen nicht zu beweisen, dass diese Grammatik D erzeugt).

Lösung:

Wie in der Vorlesung geben wir nur die Produktionen an. Das Startsymbol ist S.

$$\begin{aligned} S &\to SS \\ S &\to aSb \\ S &\to \epsilon \end{aligned}$$

Alternative Lösung:

$$\begin{aligned} S &\rightarrow aSbS \\ S &\rightarrow \epsilon \end{aligned}$$

Hausaufgabe 4 (Kontextfreie Grammatiken): (2 + 2 + 3 = 7 Punkte)

Geben Sie jeweils eine kontextfreie Grammatik an, welche die folgenden Sprachen erzeugt (Sie brauchen nicht zu beweisen, dass Ihre Grammatiken die geforderten Sprachen erzeugen).

a) $\{a^n b^n \mid n \geq 0\}$

b) $\{ab^n c \mid n > 0\}$

c) $\{w \in \{a, b, c\}^* \mid \sharp_a(w) = \sharp_b(w)\}$

Lösung:

Wie in der Vorlesung geben wir nur die Produktionen an. Das Startsymbol ist jeweils S.

a)

$$\begin{aligned} S &\rightarrow aSb \\ S &\rightarrow \epsilon \end{aligned}$$

b)

$$\begin{aligned} S &\rightarrow aBc \\ B &\rightarrow bB \\ B &\rightarrow b \end{aligned}$$

Alternativ als linkslineare Grammatik:

$$\begin{aligned} S &\rightarrow aA \\ A &\rightarrow bB \\ B &\rightarrow bB \\ B &\rightarrow c \end{aligned}$$

c)

$$\begin{aligned} S &\rightarrow SaSbS \\ S &\rightarrow SbSaS \\ S &\rightarrow cS \\ S &\rightarrow \epsilon \end{aligned}$$

Alternative Lösung mit zusätzlichem Nonterminal C:

$$\begin{aligned} S &\to SaCbC \\ S &\to SbCaC \\ S &\to CaSbC \\ S &\to CbSaC \\ S &\to CaCbS \\ S &\to CbCaS \\ S &\to C \\ C &\to cC \\ C &\to \epsilon \end{aligned}$$

Hausaufgabe 5 (Induktion): (6 Punkte)

Betrachten Sie folgende kontextfreie Grammatik $G = (N, T, P, S)$ mit $N = \{S\}$, $T = \{a, b\}$ und $P = \{$

$$\begin{aligned} S &\to abS, \\ S &\to \epsilon \end{aligned}$$

$\}$
und die Sprache $L = \{(ab)^n \mid n \geq 0\}$.

Beweisen Sie, dass $L(G) = L$ gilt.

Hinweis: Zeigen Sie die Inklusionen $L(G) \subseteq L$ und $L \subseteq L(G)$ separat. Verwenden Sie für die erste Inklusion eine Induktion über die Ableitungslänge der von G erzeugten Wörter und für die zweite Inklusion eine Induktion über die Anzahl von ab Teilworten in $w \in L$.

Lösung:

Wir beweisen zunächst $L(G) \subseteq L$. Sei $w \in L(G)$. Wir zeigen $w \in L$ per Induktion über die Ableitungslänge von $\Rightarrow_G$.
Im Induktionsanfang ist die Ableitungslänge 1 und wir haben $S \Rightarrow_G w$. Das einzige Terminalwort, das in einem Schritt von S erzeugt werden kann, ist $\epsilon \in L$.
Als Induktionshypothese setzen wir voraus, dass die Aussage für alle Ableitungslängen $m' < m$ gilt.
Im Induktionsschritt gilt nun $S \Rightarrow_G^m w$ mit $m > 1$. Da $m > 1$ gilt, kann der erste Ableitungsschritt nicht die Produktion $S \to \epsilon$ verwendet haben, da danach kein Nonterminal mehr in der resultierenden Satzform enthalten und die Ableitung beendet wäre. Also lautet der erste Ableitungsschritt $S \Rightarrow_G abS$ und wir haben $abS \Rightarrow_G^{m-1} w$. Da ein Ableitungsschritt einer kontextfreien Grammatik keine Terminale entfernen kann, gilt außerdem $w = abw'$ und $S \Rightarrow_G^{m-1} w'$. Aus der Induktionshypothese folgt nun, dass $w' \in L$ gilt. Damit gilt dann aber auch $abw' = w \in L$.

Nun beweisen wir $L \subseteq L(G)$. Sei $w = (ab)^n \in L$ für ein $n \in \mathbb{N}_0$. Wir zeigen $w \in L(G)$ per Induktion über n.
Im Induktionsanfang ist $n = 0$ und wir haben $w = \epsilon$. Es gilt $S \Rightarrow_G \epsilon$ und damit $\epsilon \in L(G)$.
Als Induktionshypothese nehmen wir an, dass die Aussage für alle $n' < n$ gilt.
Im Induktionsschritt ist nun $w = (ab)^n = ab(ab)^{n-1}$. Es gilt $S \Rightarrow_G abS$. Aus der Induktionshypothese folgt außerdem, dass $(ab)^{n-1} \in L(G)$, d.h. $S \Rightarrow_G^* (ab)^{n-1}$. Zusammen folgt daraus, dass $S \Rightarrow_G abS \Rightarrow_G^* ab(ab)^{n-1} = w$ und damit $w \in L(G)$.

Aus $L(G) \subseteq L$ und $L \subseteq L(G)$ folgt schließlich $L(G) = L$.

$\square$.

Prof. Dr. J. Giesl

M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Mi., 23.06.2010 im Tutorium abgegeben werden. Alternativ ist es bis 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet.

Tutoraufgabe 1 (Die pre^*-Operation):

Sei $G = (N, T, P, S)$ eine CFG mit $N = \{S, C\}, T = \{a, b, c\}$ und P wie folgt.

$$\begin{aligned} S &\rightarrow aSb \mid cC \mid \epsilon \\ C &\rightarrow cS \end{aligned}$$

Konstruieren Sie NFAs, welche für die folgenden Sprachen L_i jeweils die Sprache $pre^*_G(L_i)$ akzeptieren. Geben Sie außerdem an, ob $L_i \cap L(G) = \emptyset$ gilt.

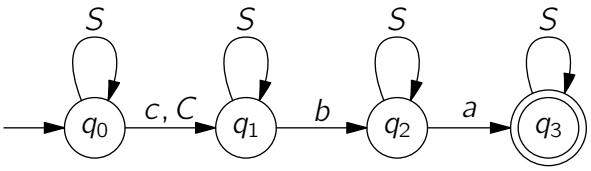
a) $L_1 = \{cba\}$

b) $L_2 = \{ccab\}$

c) $L_3 = L(a(cc)^*b)$

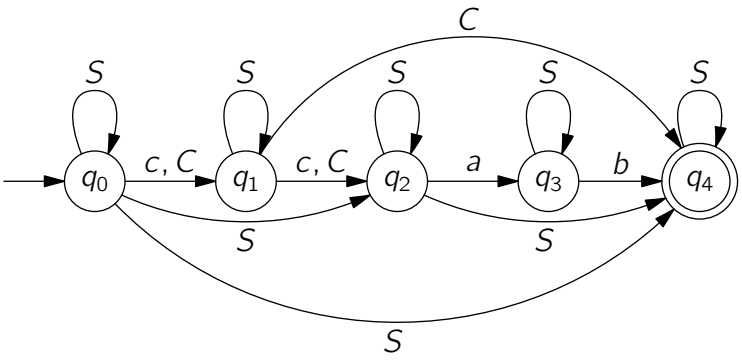
Lösung:

a)



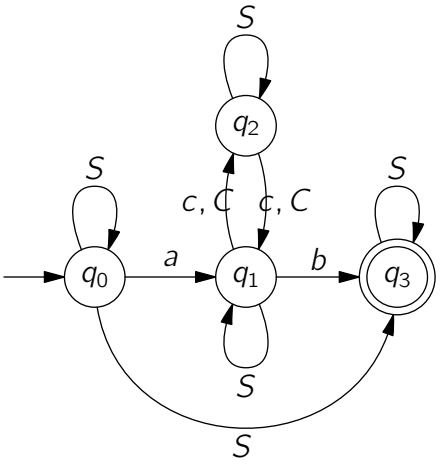
Da dieser Automat S nicht akzeptiert, gilt $L_1 \cap L(G) = \emptyset$.

b)



Da dieser Automat S akzeptiert, gilt $L_2 \cap L(G) \neq \emptyset$.

c)



Da dieser Automat S akzeptiert, gilt $L_3 \cap L(G) \neq \emptyset$.

Hausaufgabe 2 (Die pre^*-Operation): (1 + 3 + 3 = 7 Punkte)

Sei $G' = (N', T, P', S)$ eine CFG mit $N' = \{S\}, T = \{a, b, c\}$ und P' wie folgt.

$$S \quad \rightarrow \quad aSbSc \mid ScbS \mid aa$$

Konstruieren Sie NFAs, welche für die folgenden Sprachen L_i jeweils die Sprache $pre^*_{G'}(L_i)$ akzeptieren. Geben Sie außerdem an, ob $L_i \cap L(G') = \emptyset$ gilt.

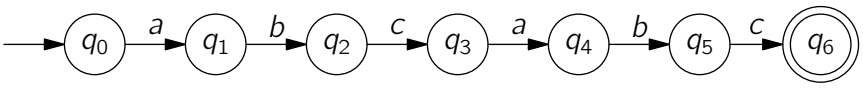
a) $L_4 = \{abcabc\}$

b) $L_5 = \{aaacbaabaac\}$

c) $L_6 = L(aa(cbaa)^*)$

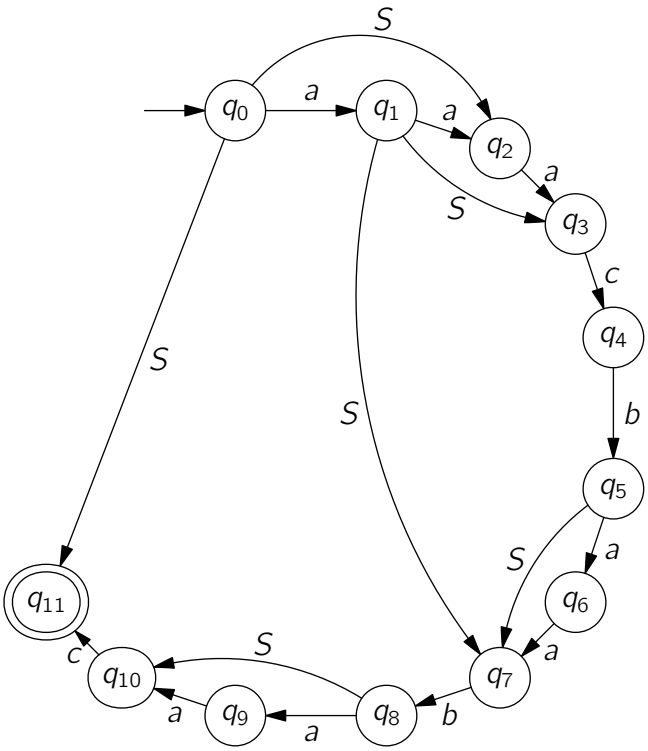
Lösung:

a)



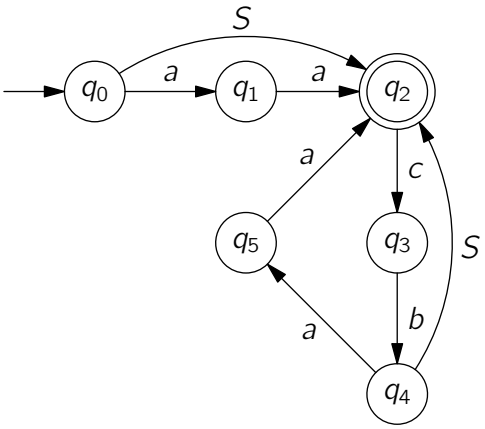
Da dieser Automat S nicht akzeptiert, gilt $L_4 \cap L(G) = \emptyset$.

b)



Da dieser Automat S akzeptiert, gilt $L_5 \cap L(G) \neq \emptyset$.

c)



Da dieser Automat S akzeptiert, gilt $L_6 \cap L(G) \neq \emptyset$.

Tutoraufgabe 3 (Äquivalenz kontextfreier Sprachen):

Sei $G_1 := (N_1, T, P_1, S_1)$ die linkslineare Grammatik aus Hausaufgabe 2 auf Übungsblatt 6 mit $N_1 := \{S_1, A, B\}$, $T := \{a, b\}$ und P_1 wie folgt:

$$S_1 \to aA \mid bB$$
$$A \to aA \mid \epsilon$$
$$B \to aB \mid bA$$

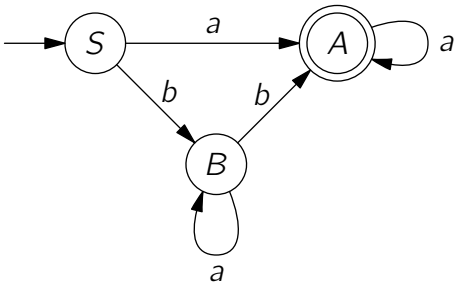
Betrachten Sie nun auch die linkslineare Grammatik $G_2 := (N_2, T, P_2, S_2)$ mit $N_2 := \{S_2, X, Y, Z\}$ und P_2 wie folgt:

$$S_2 \to aX \mid bY$$
$$X \to aZ \mid \epsilon$$
$$Y \to aY \mid bZ$$
$$Z \to aZ \mid \epsilon$$

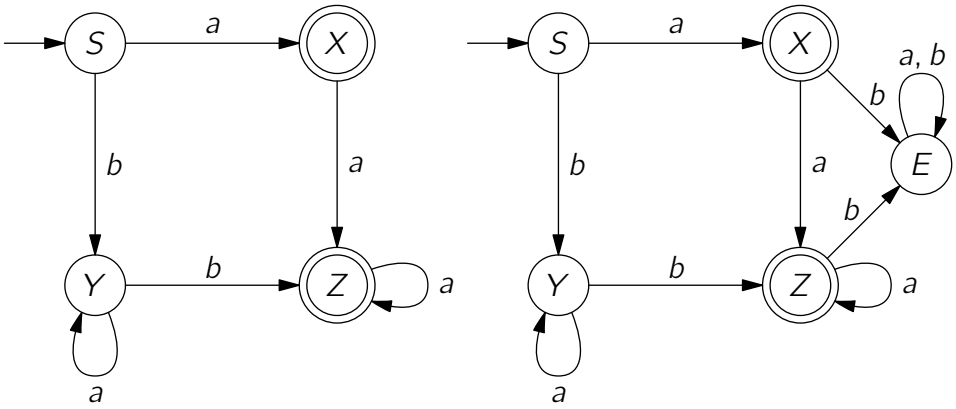
Beweisen oder widerlegen Sie $L(G_1) = L(G_2)$.

Lösung:

Nach der Konstruktion aus Aufgabe 2 auf Übungsblatt 6 können wir für G_1 den entsprechenden Automaten M_1 konstruieren:



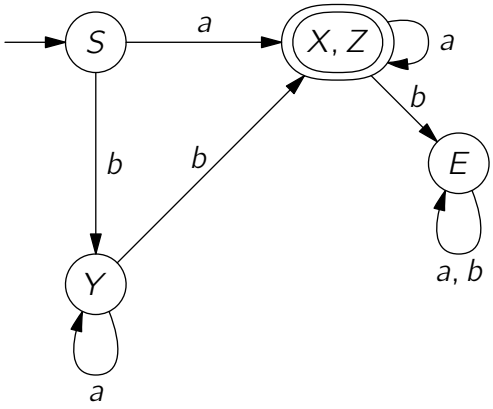
Wir können die gleiche Konstruktion für G_2 durchführen und erhalten M_2. Hier ist M_2 links dargestellt, rechts steht M_2', eine determinisierte Variante von M_2:



Wir minimieren nun den Automaten M_2' mit Hilfe des schnellen Markierungsalgorithmus:

X	$\times_0$		
Y	$\times_1$	$\times_0$	
Z	$\times_0$		$\times_0$
	S	X	Y

Wir sehen, dass Y und S mit Hilfe des Symbols a unterschieden werden können, wohingegen Z und X nicht unterscheidbar sind. Der minimale Automat $\tilde{M}_2$ zu M_2' ist also:



Offensichtlich gilt nun $L(G_1) = L(M_1) = L(\tilde{M}_2) = L(M_2') = L(M_2) = L(G_2)$.

Hausaufgabe 4 (Universalität kontextfreier Sprachen): (4 Punkte)

Sei G die folgende Grammatik. Ist $L(G)$ universell? Begründen Sie ihre Antwort.

$$
\begin{aligned}
S &\to A \mid \epsilon \mid b \mid Ca \\
A &\to aSB \mid bB \\
B &\to bSB \mid aS \mid \epsilon \\
C &\to C \mid DD \\
D &\to C \mid CC
\end{aligned}
$$

Lösung:

Die Grammatik ist universell. Es lassen sich folgende Worte ableiten:

$$
\begin{aligned}
&S \Rightarrow \epsilon \\
&S \Rightarrow A \Rightarrow aSB \Rightarrow aB \Rightarrow a \\
&S \Rightarrow b \\
&S \Rightarrow A \Rightarrow aSB \Rightarrow aS \\
&S \Rightarrow A \Rightarrow bB \Rightarrow baS \\
&S \Rightarrow A \Rightarrow bB \Rightarrow bbSB \Rightarrow bbS
\end{aligned}
$$

Somit läßt sich jedes beliebige Wort aus $\{a, b\}^*$ ableiten.

Tutoraufgabe 5 (Produktivität von Nichtterminalen):

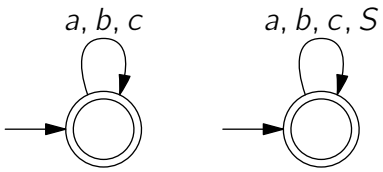
Sei $G := (N, \{a, b, c\}, P, S)$ eine kontextfreie Grammatik mit $N := \{S, A, B, C\}$ und P wie folgt:

$$
\begin{aligned}
S &\to aSb \mid bSa \mid C \mid SS \mid a \mid b \\
C &\to cC
\end{aligned}
$$

Ermitteln sie die Menge $\{A \in N \mid \text{es gibt kein } w \in T^* \text{ mit } A \Rightarrow^* w\}$ der unproduktiven Nichtterminalsymbole mit Hilfe des in der Vorlesung vorgestellten Verfahrens. Geben Sie als Zwischenergebnisse die Automaten an, die bei der Berechnung von pre^* durch Sättigungsschritte entstehen. Geben Sie außerdem eine Grammatik G' an, in der die unproduktiven Nichtterminale entfernt wurden.

Lösung:

Nach Vorlesung reicht es, $N \cap pre^*(T^*)$ zu bestimmen. Im Folgenden ist links ein NFA dargestellt, der T^* erkennt, die weiteren Automaten werden durch Sättigungsschritte erzeugt:



Der rechts stehende Automat erkennt die Sprache $pre^*(T^*) = \{a, b, c, S\}$. Damit sind die produktiven Nichtterminalsymbole $\{a, b, c, S\} \cap \{S, C\} = \{S\}$. Die auf produktive Nichtterminale eingeschränkte Grammatik ist $G' := (\{S\}, \{a, b, c\}, P', S)$ mit P' wie folgt:

$$S \to aSb \mid bSa \mid SS \mid a \mid b$$

Hausaufgabe 6 (Nullierbarkeit von Nichtterminalen): (3 Punkte)

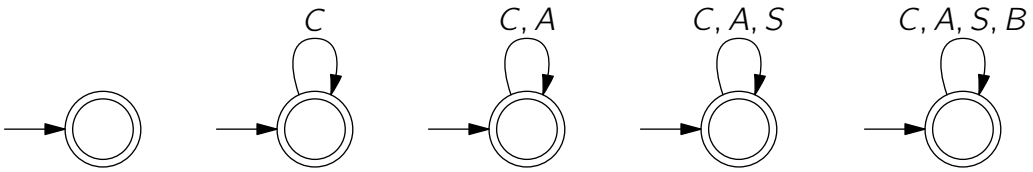
Sei $G := (N, \{a, b, c\}, P, S)$ eine kontextfreie Grammatik mit $N := \{S, A, B, C\}$ und P wie folgt:

$$\begin{aligned} S &\to A \mid B \\ A &\to aAb \mid aBb \mid AA \mid C \\ B &\to bAa \mid bBa \mid BB \mid C \\ C &\to CC \mid c \mid \epsilon \end{aligned}$$

Ermitteln sie die Menge $\{A \in N \mid A \Rightarrow^* \epsilon\}$ der nullierbaren Nichtterminalsymbole mit Hilfe des in der Vorlesung vorgestellten Verfahrens. Geben Sie als Zwischenergebnisse die Automaten an, die bei der Berechnung von pre^* durch Sättigungsschritte entstehen.

Lösung:

Nach Vorlesung reicht es, $N \cap pre^*(\{\epsilon\})$ zu bestimmen. Im Folgenden ist links ein NFA dargestellt, der $\{\epsilon\}$ erkennt, die weiteren Automaten werden durch Sättigungsschritte erzeugt:



Der rechts stehende Automat erkennt die Sprache $pre^*(\{\epsilon\}) = pre^*(\{\epsilon\}) \cap N = \{S, A, B, C\}$.

Tutoraufgabe 7 (Anwendung von pre^*):

Sei $G = (N, T, P, S)$ eine CFG. Geben Sie ein Verfahren an, das für ein $A \in N$ und ein $a \in T$ entscheidet, ob $A \Rightarrow_G^* T^*a$ gilt.

Lösung:

Falls $A \in pre(T^*a)$, so kann A zu einem Wort abgeleitet werden, dass auf a endet.

Hausaufgabe 8 (Anwendung von pre^*): (3 Punkte)

Sei $G = (N, T, P, S)$ eine CFG. Wir sagen $A \in N$ ist ein n-Produktionssymbol gdw. ein Wort $w \in T^*$ mit $|w| = n$ existiert, so dass $A \Rightarrow^+ w$.
Geben Sie ein Verfahren an, das entscheidet ob ein Nichtterminal ein n-Produktionssymbol ist.

Lösung:

Es gilt A ist ein n-Produktionssymbol gdw. $A \in pre(T^n)$.

Tutoraufgabe 9 (Präfixsprachenproblem):

Seien G_1, G_2 zwei kontextfreie Grammatiken über den Terminalsymbolen T. Wir nennen $L(G_1)$ Präfixsprache von $L(G_2)$ wenn für alle $w \in L(G_1)$ und beliebige Wörter $v \in T^*$ das Wort $w \cdot v$ in $L(G_2)$ enthalten ist.
Beweisen Sie, dass die Frage „Ist $L(G_1)$ Präfixsprache von $L(G_2)$?" unentscheidbar ist.

Lösung:

Angenommen, das Präfixsprachenproblem wäre entscheidbar. Man kann G_1 so konstruieren, dass $L(G_1) = \{\epsilon\}$ gilt. Dann ist das Präfixsprachenproblem für $L(G_1)$ und $L(G_2)$ äquivalent zu der Frage, ob für alle Wörter $v \in T^*$ gilt, dass $\epsilon \cdot v = v \in L(G_2)$ ist. Dies ist aber das Universalitätsproblem $L(G_2) = T^*$, das nach Vorlesung nicht entscheidbar ist.

Hausaufgabe 10 (Komplementärsprachenproblem): (4 Punkte)

Seien G_1, G_2 zwei kontextfreie Grammatiken über den Terminalsymbolen T. Beweisen Sie, dass das Entscheidungsproblem $L(G_1) = T^* \setminus L(G_2)$ unentscheidbar ist.

Lösung:

Angenommen, das Komplementärsprachen wäre entscheidbar. Man kann G_2 so konstruieren, dass $L(G_2) = \emptyset$ gilt. Dann ist das Komplementärsprachen für $L(G_1)$ und $L(G_2)$ äquivalent zu der Frage, ob $L(G_1) = T^* \setminus \emptyset = T^*$. Dies ist aber das Universalitätsproblem $L(G_1) = T^*$, das nach Vorlesung nicht entscheidbar ist.

Tutoraufgabe 11 (Epsilon-Schnittproblem):

Das ϵ-Schnittproblem für zwei CFGs G_1 und G_2 ist die Frage, ob $L(G_1) \cap L(G_2) = \{\epsilon\}$ gilt.

Beweisen Sie, dass das ϵ-Schnittproblem nicht entscheidbar ist.

Hinweis: Zeigen Sie, wie man mit einem Entscheidungsverfahren für dieses Problem das Post'sche Korrespondenzproblem entscheiden kann.

Lösung:

Sei $I := \left\{ \begin{bmatrix} u_1 \\ v_1 \end{bmatrix}, \begin{bmatrix} u_2 \\ v_2 \end{bmatrix}, \ldots, \begin{bmatrix} u_n \\ v_n \end{bmatrix} \right\}$ eine beliebige PCP-Instanz.

Konstruiere zwei Grammatiken G_1 und G_2:

$$
\begin{aligned}
&G_1: \\
&\quad S \rightarrow u_1 D v_1^R \mid u_2 D v_2^R \mid \ldots \mid u_n D v_n^R \mid \epsilon \\
&\quad D \rightarrow u_1 D v_1^R \mid u_2 D v_2^R \mid \ldots \mid u_n D v_n^R \mid \$
\end{aligned}
$$

$$
\begin{aligned}
&G_2: \\
&\quad S \rightarrow 0S0 \mid 1S1 \mid \$ \mid \epsilon
\end{aligned}
$$

Nun gilt $L(G_1) \cap L(G_2) = \{\epsilon\}$ gdw. I nicht lösbar ist. Aufgrund der Unentscheidbarkeit des Post'schen Korrespondenzproblems ist damit das ϵ-Schnittproblem nicht entscheidbar.

Hausaufgabe 12 (Palindromproblem): (6 Punkte)

Das Palindromproblem für eine CFG G ist die Frage, ob ein $w \in T^*$ existiert mit $ww^R \in L(G)$.

Beweisen Sie, dass das Palindromproblem nicht entscheidbar ist.

Hinweis: Zeigen Sie, wie man mit einem Entscheidungsverfahren für dieses Problem das Post'sche Korrespondenzproblem entscheiden kann.

Lösung:

Sei $I := \left\{ \boxed{\begin{matrix} u_1 \\ v_1 \end{matrix}}, \boxed{\begin{matrix} u_2 \\ v_2 \end{matrix}}, \ldots, \boxed{\begin{matrix} u_n \\ v_n \end{matrix}} \right\}$ eine beliebige PCP-Instanz.

Konstruiere eine Grammatik G wie folgt:

$$\begin{aligned} S &\rightarrow u_1 E v_1^R \mid u_2 E v_2^R \mid \ldots \mid u_n E v_n^R \\ E &\rightarrow S \mid \$\$ \end{aligned}$$

Es existiert ein $w \in T^*$ mit $ww^R \in L(G)$ gdw. I lösbar ist. Aufgrund der Unentscheidbarkeit des Post'schen Korrespondenzproblems ist damit das Palindromproblem nicht entscheidbar.

Prof. Dr. J. Giesl

M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Mi., 30.06.2010 im Tutorium abgegeben werden. Alternativ ist es bis 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet.

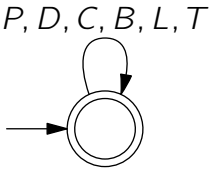
Tutoraufgabe 1 (ϵ-Produktionen):

Überführen Sie die folgende Grammatik mit dem in der Vorlesung vorgestellten Verfahren in eine äquivalente Grammatik ohne ϵ-Produktionen.

$$
\begin{array}{lcl}
S & \to & HB \\
H & \to & xTLy \\
T & \to & t \mid \epsilon \\
L & \to & LL \mid \ell \mid \epsilon \\
B & \to & iCj \mid \epsilon \\
C & \to & CC \mid D \mid P \\
D & \to & DD \mid P \mid \epsilon \\
P & \to & p \mid \epsilon
\end{array}
$$

Lösung:

Der Automat für $pre^*(\{\epsilon\})$:



Es sind also P, D, C, B, L und T nullierbar. Wir fügen die Produktionen $B \to ij$, $L \to L$, $C \to C$, $D \to D$, $H \to xTy$, $H \to xLy$, $H \to xy$ und $S \to H$ hinzu und löschen danach $T \to \epsilon$, $L \to \epsilon$, $B \to \epsilon$, $D \to \epsilon$ und $P \to \epsilon$. Es ergibt sich eine Grammatik mit den folgenden Produktionsregeln:

$$
\begin{aligned}
S &\to HB \mid H \\
H &\to xTLy \mid xTy \mid xLy \mid xy \\
T &\to t \\
L &\to LL \mid \ell \mid L \\
B &\to iCj \mid ij \\
C &\to CC \mid D \mid P \mid C \\
D &\to DD \mid P \mid D \\
P &\to p
\end{aligned}
$$

Hausaufgabe 2 (ϵ-Produktionen): (2 Punkte)

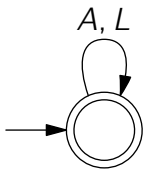
Überführen Sie die folgende Grammatik mit dem in der Vorlesung vorgestellten Verfahren in eine äquivalente Grammatik ohne ϵ-Produktionen.

$$
\begin{aligned}
S &\to (S + S) \mid (L - S) \mid N \\
N &\to DA \\
D &\to 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\
A &\to AA \mid 0 \mid D \mid \epsilon \\
L &\to S \mid \epsilon
\end{aligned}
$$

Hinweis: "(" und ")" sind Terminalsymbole.

Lösung:

Der Automat für $pre^*(\{\epsilon\})$:



Es sind also A und L nullierbar. Wir fügen die Produktionen $S \to (-S)$, $A \to A$ und $N \to D$ hinzu und löschen $A \to \epsilon$ und $L \to \epsilon$. Es ergibt sich eine Grammatik mit den folgenden Produktionsregeln:

$$
\begin{aligned}
S &\to (S + S) \mid (L - S) \mid (-S) \mid N \\
N &\to DA \mid D \\
D &\to 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\
A &\to AA \mid 0 \mid D \mid A \\
L &\to S
\end{aligned}
$$

Tutoraufgabe 3 (Chomsky-Normalform):

Überführen Sie die folgende Grammatik mit dem in der Vorlesung vorgestellten Verfahren in eine äquivalente Grammatik in Chomsky-Normalform.

$$\begin{aligned} S &\rightarrow aB \mid bA \mid ABc \mid B \\ A &\rightarrow SSa \\ B &\rightarrow cS \mid bB \mid b \end{aligned}$$

Lösung:

$$\begin{aligned} S &\rightarrow R_aB \mid R_aR_b \mid R_bA \mid AC \mid R_cS \mid R_cB \mid R_cR_b \mid R_bB \mid R_bR_b \mid b \\ A &\rightarrow SD \mid BD \mid R_bD \\ B &\rightarrow R_cS \mid R_cB \mid R_cR_b \mid R_bB \mid R_bR_b \\ C &\rightarrow BR_c \mid R_bR_c \\ D &\rightarrow SR_a \mid BR_a \mid R_bR_a \\ R_a &\rightarrow a \\ R_b &\rightarrow b \\ R_c &\rightarrow c \end{aligned}$$

Hausaufgabe 4 (Chomsky-Normalform): (4 Punkte)

Überführen Sie die folgende Grammatik mit dem in der Vorlesung vorgestellten Verfahren in eine äquivalente Grammatik in Chomsky-Normalform.

$$\begin{aligned} S &\rightarrow aSbS \mid C \\ C &\rightarrow cC \mid c \end{aligned}$$

Lösung:

$$
\begin{aligned}
S &\rightarrow R_aA \mid R_cC \mid R_cR_c \mid c \\
A &\rightarrow SB \mid CB \mid R_cB \\
B &\rightarrow R_bS \mid R_bC \mid R_bR_c \\
C &\rightarrow R_cC \mid R_cR_c \\
R_a &\rightarrow a \\
R_b &\rightarrow b \\
R_c &\rightarrow c
\end{aligned}
$$

Tutoraufgabe 5 (CYK-Algorithmus):

Gegeben sei die folgende Grammatik G.

$$
\begin{aligned}
S &\rightarrow SS \mid R_aA \mid R_bB \mid R_cC \\
A &\rightarrow R_bR_c \mid R_cR_b \\
B &\rightarrow R_aR_c \mid R_cR_a \\
C &\rightarrow R_aR_b \mid R_bR_a \\
R_a &\rightarrow a \\
R_b &\rightarrow b \\
R_c &\rightarrow c
\end{aligned}
$$

Testen Sie mit dem CYK-Algorithmus, ob die folgenden Wörter von G erzeugt werden.

a) $w_1 = abccab$

b) $w_2 = abcba$

Lösung:

a)

$w =$	a	b	c	c	a	b
1	R_a	R_b	R_c	R_c	R_a	R_b
2	C	A		B	C	
3	S			S		
4						
5						
6	S					

$\Rightarrow w_1 \in L(G)$

b)

$w =$	a	b	c	b	a
1	R_a	R_b	R_c	R_b	R_a
2	C	A	A	C	
3	S		S		
4					
5					

$\Rightarrow w_2 \notin L(G)$

Hausaufgabe 6 (CYK-Algorithmus): (3 + 3 = 6 Punkte)

Gegeben sei die folgende Grammatik G.

$$
\begin{aligned}
S &\rightarrow R_aA \mid R_eB \mid R_gC \mid R_lD \mid R_rE \mid R_aR_a \mid R_eR_e \mid R_gR_g \mid R_lR_l \mid R_rR_r \\
A &\rightarrow SR_a \\
B &\rightarrow SR_e \\
C &\rightarrow SR_g \\
D &\rightarrow SR_l \\
E &\rightarrow SR_r \\
R_a &\rightarrow a \\
R_e &\rightarrow e \\
R_g &\rightarrow g \\
R_l &\rightarrow l \\
R_r &\rightarrow r
\end{aligned}
$$

Testen Sie mit dem CYK-Algorithmus, ob die folgenden Wörter von G erzeugt werden.

a) $w_1 = regallager$

b) $w_2 = allee$

Lösung:

a)

$w =$	r	e	g	a	l	l	a	g	e	r
1	R_r	R_e	R_g	R_a	R_l	R_l	R_a	R_g	R_e	R_r
2					S					
3					A					
4				S						
5				C						
6			S							
7			B							
8		S								
9		E								
10	S									

$\Rightarrow w_1 \in L(G)$

b)

$w =$	a	l	l	e	e
1	R_a	R_l	R_l	R_e	R_e
2		S		S	
3		B			
4					
5					

$\Rightarrow w_2 \notin L(G)$

Tutoraufgabe 7 (Greibach-Normalform):

Verwenden Sie den Algorithmus aus der Vorlesung, um für die folgende Grammatik G eine Grammatik G' in Greibach-Normalform mit $L(G') = L(G)$ zu bestimmen. Verwenden Sie dabei die folgende Ordnung der Nichtterminale: $S < A < B < C$.

$$\begin{aligned} S &\to AS \mid SB \\ A &\to a \mid CB \\ B &\to AC \mid b \\ C &\to AB \end{aligned}$$

Lösung:

Die Regel $B \to AC$ ist die echt verkleinernde Regel mit dem kleinsten linken Nichtterminal und wird ersetzt.

$$\begin{aligned} S &\to AS \mid SB \\ A &\to a \mid CB \\ B &\to aC \mid CBC \mid b \\ C &\to AB \end{aligned}$$

Die Regel $C \to AB$ ist die echt verkleinernde Regel mit dem kleinsten linken Nichtterminal und wird ersetzt.

$$\begin{aligned} S &\to AS \mid SB \\ A &\to a \mid CB \\ B &\to aC \mid CBC \mid b \\ C &\to aB \mid CBB \end{aligned}$$

Nun gibt es keine echt verkleinernden Regeln mehr. Die Linksrekursion in den S-Regeln wird beseitigt. Dabei wird das Nichtterminal Z als neues kleinstes Element eingeführt.

$$\begin{aligned} Z &\to B \mid BZ \\ S &\to AS \mid ASZ \\ A &\to a \mid CB \\ B &\to aC \mid CBC \mid b \\ C &\to aB \mid CBB \end{aligned}$$

Die Linksrekursion in den C-Regeln wird beseitigt. Dabei wird das Nichtterminal Y als neues kleinstes Element eingeführt.

$$\begin{aligned} Y &\to BB \mid BBY \\ Z &\to B \mid BZ \\ S &\to AS \mid ASZ \\ A &\to a \mid CB \\ B &\to aC \mid CBC \mid b \\ C &\to aB \mid aBY \end{aligned}$$

Nun gibt es keine verkleinernde Regel mehr. Es bleibt, die auf manchen rechten Seiten führenden Nichtterminale zu eliminieren. Das größte Nichtterminal (C) ist nach Konstruktion schon in Greibach Form. Wir beginnen also mit B.

$$\begin{aligned}
Y &\to BB \mid BBY \\
Z &\to B \mid BZ \\
S &\to AS \mid ASZ \\
A &\to a \mid CB \\
B &\to aC \mid aBBC \mid aBYBC \mid b \\
C &\to aB \mid aBY
\end{aligned}$$

Nun folgt die Regel $A \to CB$.

$$\begin{aligned}
Y &\to BB \mid BBY \\
Z &\to B \mid BZ \\
S &\to AS \mid ASZ \\
A &\to a \mid aBB \mid aBYB \\
B &\to aC \mid aBBC \mid aBYBC \mid b \\
C &\to aB \mid aBY
\end{aligned}$$

Und nun die S-Regeln.

$$\begin{aligned}
Y &\to BB \mid BBY \\
Z &\to B \mid BZ \\
S &\to aS \mid aBBS \mid aBYBS \mid aSZ \mid aBBSZ \mid aBYBSZ \\
A &\to a \mid aBB \mid aBYB \\
B &\to aC \mid aBBC \mid aBYBC \mid b \\
C &\to aB \mid aBY
\end{aligned}$$

Nach dem Ersetzen in den Z- und Y-Regeln befindet sich die Grammatik in Greibach-Normalform:

$$\begin{aligned}
Y &\to aCB \mid aBBCB \mid aBYBCB \mid bB \mid aCBY \mid aBBCBY \mid aBYBCBY \mid bBY \\
Z &\to aC \mid aBBC \mid aBYBC \mid bB \mid aCZ \mid aBBCZ \mid aBYBCZ \mid bZ \\
S &\to aS \mid aBBS \mid aBYBS \mid aSZ \mid aBBSZ \mid aBYBSZ \\
A &\to a \mid aBB \mid aBYB \\
B &\to aC \mid aBBC \mid aBYBC \mid b \\
C &\to aB \mid aBY
\end{aligned}$$

Hausaufgabe 8 (Greibach-Normalform): (5 Punkte)

Verwenden Sie den Algorithmus aus der Vorlesung, um für die folgende Grammatik G eine Grammatik G' in Greibach-Normalform mit $L(G') = L(G)$ zu bestimmen. Verwenden Sie dabei die folgende Ordnung der Nichtterminale: $S < A < B < C$.

$$\begin{aligned}
S &\to AB \mid a \\
A &\to SA \\
B &\to CA \mid b \\
C &\to BB \mid c
\end{aligned}$$

Lösung:

Die Regel $A \to SA$ ist die echt verkleinernde Regel mit dem kleinsten linken Nichtterminal und wird ersetzt.

$$\begin{aligned} S &\to AB \mid a \\ A &\to ABA \mid aA \\ B &\to CA \mid b \\ C &\to BB \mid c \end{aligned}$$

Die Regel $C \to BB$ ist die echt verkleinernde Regel mit dem kleinsten linken Nichtterminal und wird ersetzt.

$$\begin{aligned} S &\to AB \mid a \\ A &\to ABA \mid aA \\ B &\to CA \mid b \\ C &\to CAB \mid bB \mid c \end{aligned}$$

Nun gibt es keine echt verkleinernden Regeln mehr. Die Linksrekursion in den A-Regeln wird beseitigt. Dabei wird das Nichtterminal Z als neues kleinstes Element eingeführt.

$$\begin{aligned} Z &\to BA \mid BAZ \\ S &\to AB \mid a \\ A &\to aA \mid aAZ \\ B &\to CA \mid b \\ C &\to CAB \mid bB \mid c \end{aligned}$$

Die Linksrekursion in den C-Regeln wird beseitigt. Dabei wird das Nichtterminal Y als neues kleinstes Element eingeführt.

$$\begin{aligned} Y &\to AB \mid ABY \\ Z &\to BA \mid BAZ \\ S &\to AB \mid a \\ A &\to aA \mid aAZ \\ B &\to CA \mid b \\ C &\to bB \mid c \mid bBY \mid cY \end{aligned}$$

Nun gibt es keine verkleinernde Regel mehr. Es bleibt, die auf manchen rechten Seiten führenden Nichtterminale zu eliminieren. Das größte Nichtterminal (C) ist nach Konstruktion schon in Greibach Form. Wir beginnen also mit B.

$$\begin{aligned} Y &\to AB \mid ABY \\ Z &\to BA \mid BAZ \\ S &\to AB \mid a \\ A &\to aA \mid aAZ \\ B &\to bBA \mid cA \mid bBYA \mid cYA \mid b \\ C &\to bB \mid c \mid bBY \mid cY \end{aligned}$$

Nun folgt S, da die A-Regeln ebenfalls schon die gewünschte Form haben.

$$\begin{aligned}
Y &\to AB \mid ABY \\
Z &\to BA \mid BAZ \\
S &\to aAB \mid aAZB \mid a \\
A &\to aA \mid aAZ \\
B &\to bBA \mid cA \mid bBYA \mid cYA \mid b \\
C &\to bB \mid c \mid bBY \mid cY
\end{aligned}$$

Nach dem Ersetzen in den Z- und Y-Regeln befindet sich die Grammatik in Greibach-Normalform:

$$\begin{aligned}
Y &\to aAB \mid aAZB \mid aABY \mid aAZBY \\
Z &\to bBAB \mid cAB \mid bBYAB \mid cYAB \mid bA \mid bBAAZ \mid cAAZ \mid bBYAAZ \mid cYAAZ \mid bAZ \\
S &\to aAB \mid aAZB \mid a \\
A &\to aA \mid aAZ \\
B &\to bBA \mid cA \mid bBYA \mid cYA \mid b \\
C &\to bB \mid c \mid bBY \mid cY
\end{aligned}$$

Tutoraufgabe 9 (Pumping-Lemma):

Prüfen Sie, ob folgende Sprachen über dem Alphabet $\Sigma = \{a, b, c\}$ kontextfrei sind. Falls die Sprache nicht kontextfrei ist, beweisen Sie dies mit Hilfe des Pumping-Lemmas für kontextfreie Sprachen. Falls die Sprache kontextfrei ist, geben Sie eine CFG an, welche die Sprache erzeugt.

a) $L_1 = \{(ab)^n(ca)^n(bc)^n \mid n \geq 0\}$

b) $L_2 = \{a^n b^{2n} c \mid n \geq 0\}$

Lösung:

a) Sei $n \in \mathbb{N}$. Wir wählen das Wort $z = (ab)^n(ca)^n(bc)^n \in L_1$. Sei $uvwxy = z$ eine Zerlegung mit $|vwx| \leq n$ und $|vx| > 0$. Wir wählen nun $i = 0$ und betrachten das Wort $z' = uwy$. Wegen $|vwx| \leq n$ gibt es nur zwei Fälle. Falls $(bc)^n$ ein Teilwort von y ist, muss z' zu wenige ab oder ca Teilworte enthalten und es gilt $z' \notin L_1$. Falls $(ab)^n$ ein Teilwort von u ist, muss z' zu wenige ca oder bc Teilworte enthalten und es gilt wiederum $z' \notin L_1$. Damit ist L_1 nicht kontextfrei.

b) L_2 ist kontextfrei mit $L_2 = L(G)$ für G wie folgt:

$$\begin{aligned}
S &\to Ac \\
A &\to aAbb \mid \epsilon
\end{aligned}$$

Hausaufgabe 10 (Pumping-Lemma): (3 + 3 + 3 = 9 Punkte)

Prüfen Sie, ob folgende Sprachen über dem Alphabet $\Sigma = \{a, b, c\}$ kontextfrei sind. Falls die Sprache nicht kontextfrei ist, beweisen Sie dies mit Hilfe des Pumping-Lemmas für kontextfreie Sprachen. Falls die Sprache kontextfrei ist, geben Sie eine CFG an, welche die Sprache erzeugt. Wir erinnern an die Funktionen $\sharp_z$ für ein Zeichen $z \in \Sigma$, welche die Anzahl der Zeichen z in einem Wort zählen.

a) $L_3 = \{a^n b^{2n} c^{3n} \mid n \geq 0\}$

b) $L_4 = \{w \in \Sigma^* \mid \sharp_a(w) = \sharp_b(w) = \sharp_c(w)\}$

c) $L_5 = \{a^n b^{n^2} \mid n \geq 0\}$

Lösung:

a) Sei $n \in \mathbb{N}$. Wir wählen das Wort $z = a^n b^{2n} c^{3n} \in L_3$. Sei $uvwxy = z$ eine Zerlegung mit $|vwx| \leq n$ und $|vx| > 0$.

Wir wählen nun $i = 0$ und betrachten damit das Wort $z' = uwy$. Wegen $|vwx| \leq n$ gibt es nur zwei Fälle.

Fall $vwx = a^k b^\ell$ für $k, \ell \in \mathbb{N}_0$: Da $|vx| > 0$ gilt, muss mindestens eins von beiden nicht-leer sein. Durch das Löschen von v und x aus z erhalten wir also ein Wort, das weiterhin mit c^{3n} endet, die Zahl der a oder b ist aber strikt kleiner geworden und daher gilt $z' \notin L_3$.

Fall $vwx = b^k c^\ell$ für $k, \ell \in \mathbb{N}_0$: Analog zum vorherigen Fall enthält z' mindestens ein b oder c zu wenig und wieder gilt $z' \notin L_3$.

Die Fälle $vwx = a^k$, $vwx = b^k$ und $vwx = c^k$ sind in den beiden betrachteten Fällen enthalten, da dort k bzw. ℓ jeweils als 0 gewählt werden können.

Die Bedingungen des Pumping Lemmas gelten also nicht für die Sprache L_3 und das Wort z. Daher ist L_3 keine kontextfreie Sprache.

b) Sei $n \in \mathbb{N}$. Wir wählen das Wort $z = a^n b^n c^n \in L_4$. Sei $uvwxy = z$ eine Zerlegung mit $|vwx| \leq n$ und $|vx| > 0$.

Wir wählen nun $i = 0$ und betrachten das Wort $z' = uwy$. Wegen $|vwx| \leq n$ gibt es nur zwei Fälle.

Fall $vwx = a^k b^\ell$ für $k, \ell \in \mathbb{N}_0$: Da $|vx| > 0$ gilt, muss mindestens eins von beiden nicht-leer sein. Durch das Löschen von v und x aus z erhalten wir also ein Wort z', das weiterhin mit c^n endet, die Zahl der a oder b ist aber strikt kleiner geworden und daher gilt $z' \notin L_4$.

Fall $vwx = b^k c^\ell$ für $k, \ell \in \mathbb{N}_0$: Analog zum vorherigen Fall enthält z' mindestens ein b oder c zu wenig und wieder gilt $z' \notin L_4$.

Die Fälle $vwx = a^k$, $vwx = b^k$ und $vwx = c^k$ sind in den beiden betrachteten Fällen enthalten, da dort k bzw. ℓ jeweils als 0 gewählt werden können.

Die Bedingungen des Pumping Lemmas gelten also nicht für die Sprache L_4 und das Wort z. Daher ist L_4 keine kontextfreie Sprache.

c) Sei $n \in \mathbb{N}$. Wir wählen das Wort $z = a^n b^{n^2} \in L_5$. Sei $uvwxy = z$ eine Zerlegung mit $|vwx| \leq n$ und $|vx| > 0$.

Wir wählen nun $i = 2$ und betrachten das Wort $z' = uv^2wx^2y$. Falls $vwx \in L(a^*)$ oder $vwx \in L(b^*)$, gilt direkt $z' \notin L_5$, da nur die Anzahl von a's oder b's verändert wird und die Bedingung $\sharp_a(z') = \sharp_b(z')^2$ verletzt wird.

Sei also $vwx = a^k b^\ell$. Falls $v = a^k b^t$ oder $x = a^t b^\ell$ ist, gilt $z' = uvvwxxy = ua^k b^t a^k b^t wxxy$ bzw. $z' = uvvwa^t b^\ell a^t b^\ell y$, d.h. z' enthält das Teilwort ba und damit ist $z' \notin L_5$.

Im letzten verbleibenden Fall gilt $v = a^p$ und $x = b^q$ für $p, q \in \mathbb{N}_0$. Dann ist $w = a^{k-p}b^{\ell-q}$ und $z' = uvvwxxy = a^{n-k}a^p a^p a^{k-p}b^{\ell-q}b^q b^q b^{n^2-q} = a^{n+p}b^{n^2+q}$. Es gilt $(n+p)^2 = n^2 + 2np + p^2$. Damit $z' \in L_5$ gilt, müsste also $n^2 + 2np + p^2 = n^2 + q \Leftrightarrow 2np + p^2 = q$ gelten. Wegen $n \geq |vwx| \geq q$ ist dies ein Widerspruch und es gilt $z' \notin L_5$. Damit ist L_5 nicht kontextfrei.

Hausaufgabe 11 (Pumping-Lemma): (2 Punkte)

Zeigen Sie, dass das Pumping-Lemma für kontextfreie Sprachen für jede endliche Sprache erfüllt ist. Beweisen Sie dazu die folgende Aussage:

Sei L eine endliche Sprache. Dann existiert ein $n \in \mathbb{N}$, sodass für jedes $z \in L$ mit $|z| \geq n$ gilt, dass Wörter u, v, w, x, y existieren mit $z = uvwxy$, $|vwx| \leq n$ und $|vx| > 0$, sodass für jedes $i \in \mathbb{N}$ gilt, dass $uv^i wx^i y \in L$.

Sie dürfen in diesem Beweis nicht verwenden, dass *jede* kontextfreie Sprache das Pumping-Lemma für kontextfreie Sprachen erfüllt.

Lösung:

Sei L eine endliche Sprache. Sei $n \in \mathbb{N}$ größer als die Länge aller Wörter in L. Dann gibt es kein Wort $z \in L$ mit $|z| \geq n$ und die Aussage gilt trivialerweise.

Alternativer Beweis mit Hilfe des Pumping-Lemmas für reguläre Sprachen:
Sei L eine endliche Sprache. Da L regulär ist, gilt nach dem Pumping-Lemma für reguläre Sprachen, dass ein $n \in \mathbb{N}$ existiert, sodass für jedes $z \in L$ mit $|z| \geq n$ gilt, dass Wörter r, s, t existieren mit $z = rst$, $|rs| \leq n$ und $|s| > 0$, sodass für jedes $i \in \mathbb{N}$ gilt, dass $rs^i t \in L$. Wähle $u = v = \epsilon$, $w = r$, $x = s$ und $y = t$. Dann gilt $|vwx| = |rs| \leq n$, $|vx| = |s| > 0$ und $uv^i wx^i y = rs^i t \in L$ für alle $i \in \mathbb{N}$. Damit ist die Aussage bewiesen.

Prof. Dr. J. Giesl

M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Mi., 7.07.2010 im Tutorium abgegeben werden. Alternativ ist es bis 17 Uhr möglich, diese in den Kasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet.

Tutoraufgabe 1 ((Deterministische) Kellerautomaten):

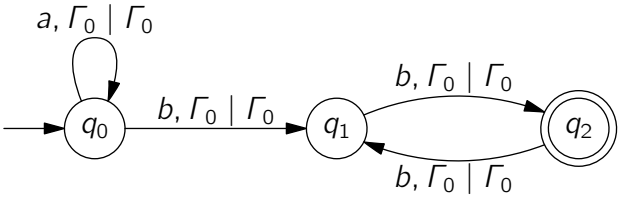
Geben Sie für die folgenden kontextfreien Sprachen L_i einen PDA M_i an, so dass $L_i = L(M_i)$. Wenn möglich, soll M_i ein deterministischer PDA sein.

a) $L_1 = \{a^n b^{2m} \mid n \in \mathbb{N}_0, m \in \mathbb{N}_{>0}\}$

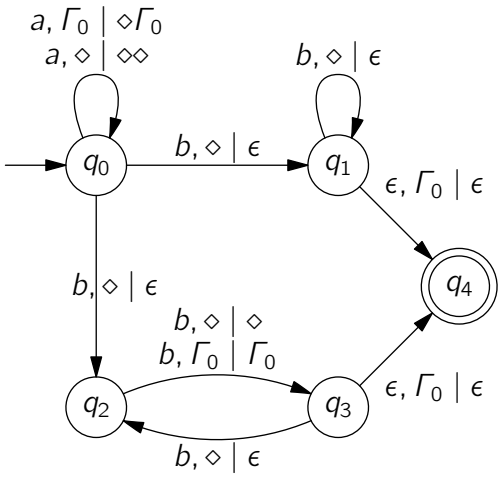
b) $L_2 = \{a^n b^n \mid n \in \mathbb{N}_{>0}\} \cup \{a^n b^{2n} \mid n \in \mathbb{N}_{>0}\}$

Lösung:

a) Der folgende deterministische PDA erkennt L_1:



b) Der folgende nicht-deterministische PDA erkennt L_2:



Hausaufgabe 2 ((Deterministische) Kellerautomaten): (2 + 3 = 5 Punkte)

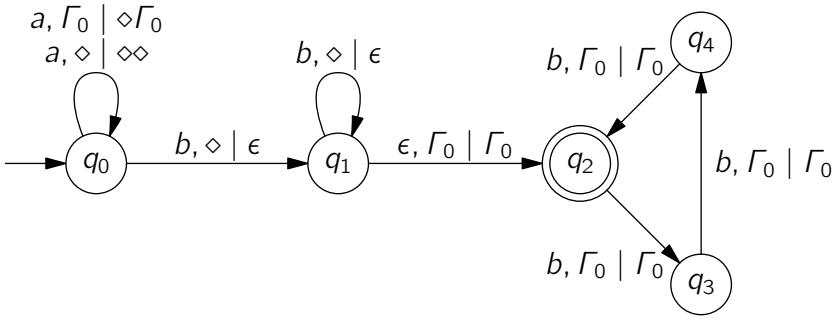
Geben Sie für die folgenden kontextfreien Sprachen L_i einen PDA M_i an, so dass $L_i = L(M_i)$. Wenn möglich, soll M_i ein deterministischer PDA sein.

a) $L_1 = \{a^n b^{n+3k} \mid n \in \mathbb{N}_{>0}, k \in \mathbb{N}_0\}$

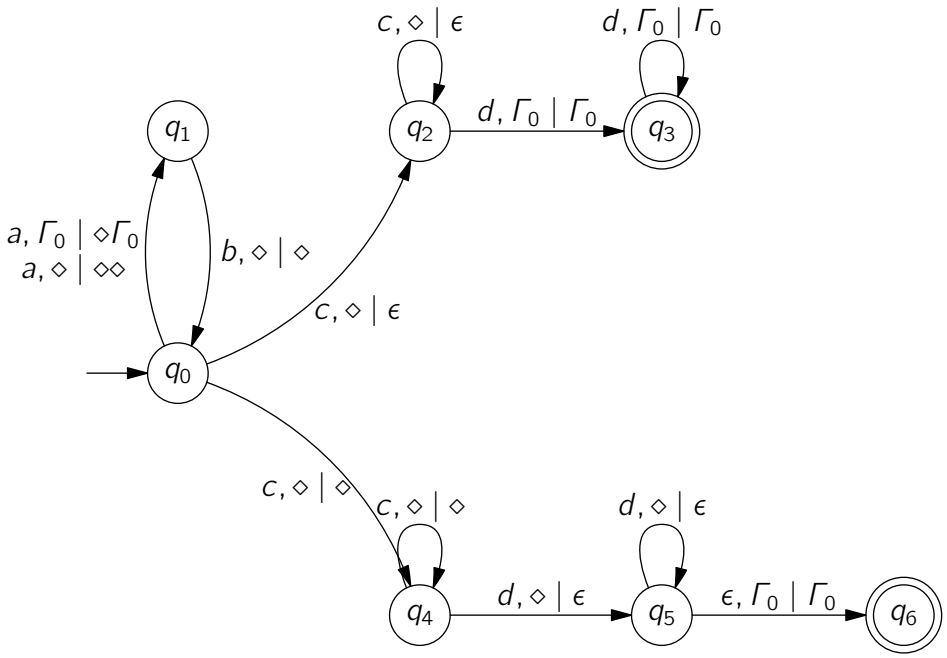
b) $L_2 = \{(ab)^n c^m d^\ell \mid n, m, \ell \in \mathbb{N}_{>0} \wedge (n = m \vee n = \ell)\}$

Lösung:

a) Der folgende deterministische PDA erkennt L_1:

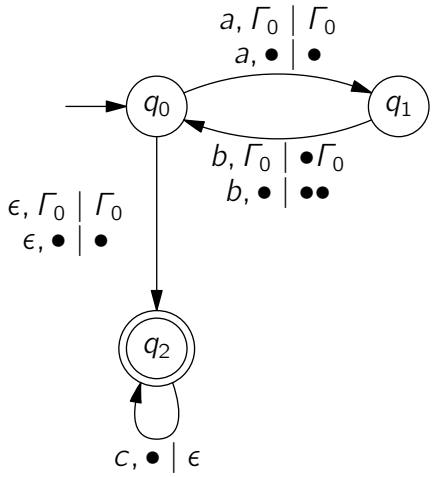


b) Der folgende nicht-deterministische PDA erkennt L_2:



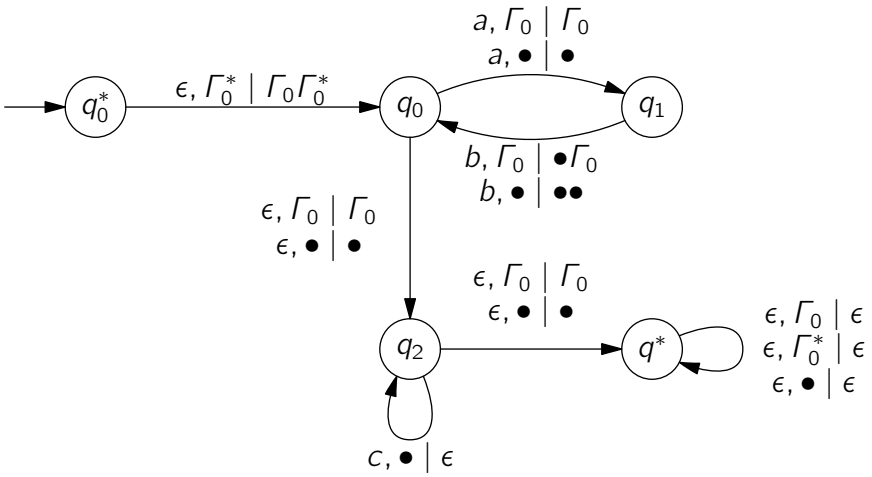
Tutoraufgabe 3 (Akzeptanzbedingungen von Kellerautomaten):

Gegeben sei der PDA $A = (Q, \Sigma, \Gamma, \delta, q_0, \Gamma_0)$.



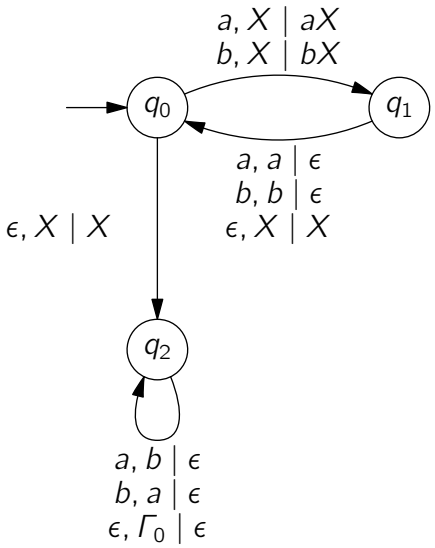
Verwenden Sie das Verfahren aus der Vorlesung, um einen PDA B zu konstruieren, so dass $N(B) = L(A)$ gilt.

Lösung:



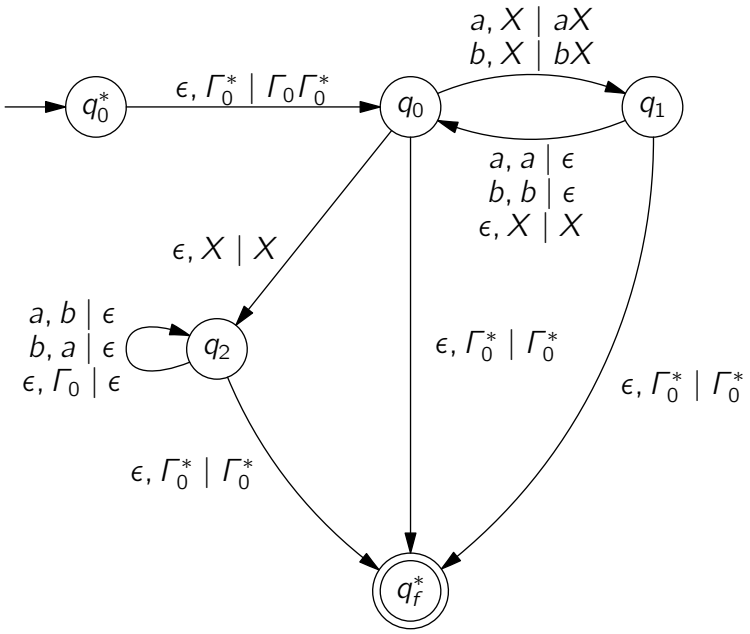
Hausaufgabe 4 (Akzeptanzbedingungen von Kellerautomaten): (3 Punkte)

Gegeben sei der PDA $A = (Q, \Sigma, \Gamma, \delta, q_0, \Gamma_0)$.



Hierbei steht X abkürzend für a oder b.
Verwenden Sie das Verfahren aus der Vorlesung, um einen PDA B zu konstruieren, so dass $L(B) = N(A)$ gilt.

Lösung:



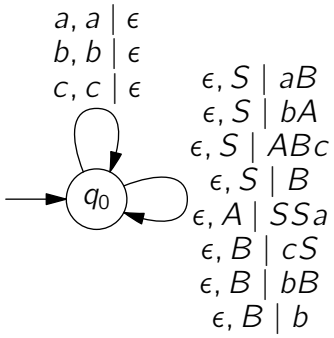
Tutoraufgabe 5 (CFGs und Kellerautomaten):

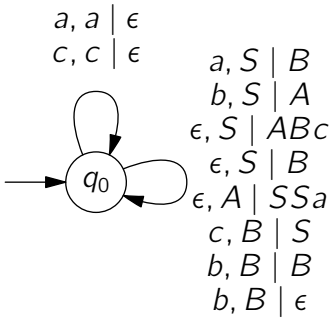
Überführen Sie die folgende Grammatik G mit dem in der Vorlesung vorgestellten Verfahren in einen Kellerautomaten M mit $L(G) = N(M)$.

$$\begin{aligned} S &\rightarrow aB \mid bA \mid ABc \mid B \\ A &\rightarrow SSa \\ B &\rightarrow cS \mid bB \mid b \end{aligned}$$

Lösung:

Links ist die direkte Lösung zu sehen, rechts eine Lösung, in der Terminalsymbole direkt konsumiert werden, statt sie auf den Keller zu legen. Das Kellerbodensymbol ist jeweils S:





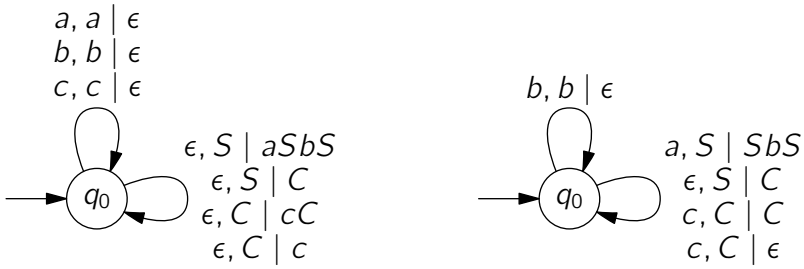
Hausaufgabe 6 (CFGs und Kellerautomaten): (2 Punkte)

Überführen Sie die folgende Grammatik G mit dem in der Vorlesung vorgestellten Verfahren in einen Kellerautomaten M mit $L(G) = N(M)$.

$$\begin{aligned} S &\rightarrow aSbS \mid C \\ C &\rightarrow cC \mid c \end{aligned}$$

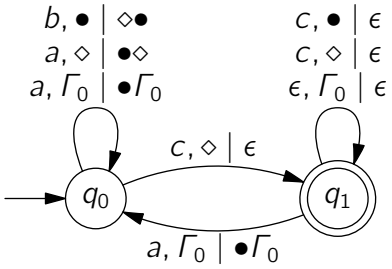
Lösung:

Links ist die direkte Lösung zu sehen, rechts eine Lösung, in der Terminalsymbole direkt konsumiert werden, statt sie auf den Keller zu legen. Das Kellerbodensymbol ist jeweils S:



Tutoraufgabe 7 (Kellerautomaten und CFGs):

Betrachten Sie den folgenden Kellerautomaten M über dem Eingabealphabet $\{a, b, c\}$ und dem Kelleralphabet $\{\circ, \diamond, \Gamma_0\}$:



a) Geben Sie die Sprachen $L(M)$ und $N(M)$ (ohne Beweis) an.

b) Geben Sie eine kontextfreie Grammatik G an, so dass $L(G) = N(M)$ gilt. Verwenden Sie hierzu das Verfahren aus der Vorlesung. Geben Sie zu jeder Transition $\delta(q, a, X) = \{(p, Y)\}$ die daraus entstehenden Grammatikregeln an.

c) Geben Sie für das Wort $abcc$ einen Lauf durch den Automaten M an, d.h. eine Folge von Konfigurationen, die mit $(q_0, abcc, \Gamma_0)$ beginnt und mit (p, ϵ, ϵ) für ein $p \in \{q_0, q_1\}$ endet.

Geben Sie außerdem einen Ableitungsbaum für $abcc$ auf Basis der Grammatik G aus Teil (b) an.

Lösung:

a)

$$N(M) = (\{(ab)^n c^{2n} \mid n \in \mathbb{N}_{>0}\})^+$$
$$L(M) = (\{(ab)^n c^{2n} \mid n \in \mathbb{N}_{>0}\})^* \cdot \{(ab)^n c^m \mid n, m \in \mathbb{N}_{>0} \wedge m \leq 2n\}$$

b) Wir benötigen zuerst die Menge N der Nonterminale $\{[q, Z, r] \mid q, r \in Q, Z \in \Gamma\}$. Diese ist in unserem Falle

$$\{[q_0, \diamond, q_0], [q_0, \diamond, q_1], [q_0, \bullet, q_0], [q_0, \bullet, q_1], [q_0, \Gamma_0, q_0], [q_0, \Gamma_0, q_1],$$
$$[q_1, \diamond, q_0], [q_1, \diamond, q_1], [q_1, \bullet, q_0], [q_1, \bullet, q_1], [q_1, \Gamma_0, q_0], [q_1, \Gamma_0, q_1]\}$$

Wir müssen nun die Regelmenge P erzeugen. Dafür betrachten wir nun jede der Transitionen in M. Im Folgenden sind jeweils die Zusammenhänge zwischen den Teilen der erzeugenden Transition und der erzeugten Regel farblich dargestellt:

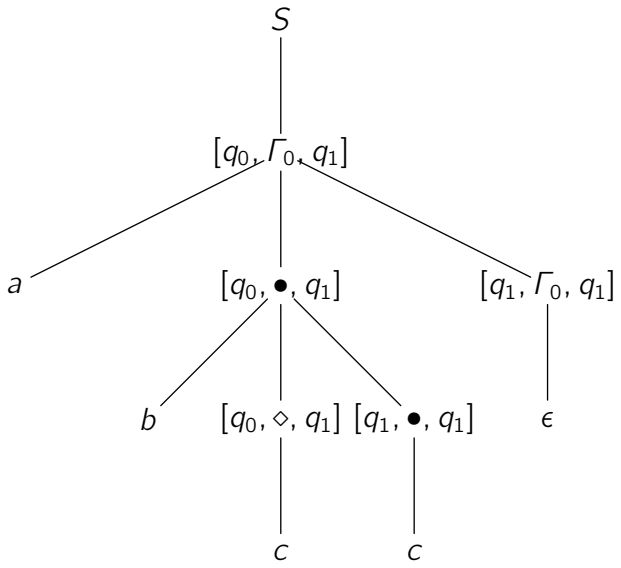
- $\delta(q_0, a, \Gamma_0) = \{(q_0, \bullet\Gamma_0)\}$:
  - $[q_0, \Gamma_0, q_0] \to a[q_0, \bullet, q_0][q_0, \Gamma_0, q_0]$
  - $[q_0, \Gamma_0, q_0] \to a[q_0, \bullet, q_1][q_1, \Gamma_0, q_0]$
  - $[q_0, \Gamma_0, q_1] \to a[q_0, \bullet, q_0][q_0, \Gamma_0, q_1]$
  - $[q_0, \Gamma_0, q_1] \to a[q_0, \bullet, q_1][q_1, \Gamma_0, q_1]$
- $\delta(q_0, a, \diamond) = \{(q_0, \bullet\diamond)\}$:
  - $[q_0, \diamond, q_0] \to a[q_0, \bullet, q_0][q_0, \diamond, q_0]$
  - $[q_0, \diamond, q_0] \to a[q_0, \bullet, q_1][q_1, \diamond, q_0]$
  - $[q_0, \diamond, q_1] \to a[q_0, \bullet, q_0][q_0, \diamond, q_1]$
  - $[q_0, \diamond, q_1] \to a[q_0, \bullet, q_1][q_1, \diamond, q_1]$
- $\delta(q_0, b, \bullet) = \{(q_0, \diamond\bullet)\}$:
  - $[q_0, \bullet, q_0] \to b[q_0, \diamond, q_0][q_0, \bullet, q_0]$
  - $[q_0, \bullet, q_0] \to b[q_0, \diamond, q_1][q_1, \bullet, q_0]$
  - $[q_0, \bullet, q_1] \to b[q_0, \diamond, q_0][q_0, \bullet, q_1]$
  - $[q_0, \bullet, q_1] \to b[q_0, \diamond, q_1][q_1, \bullet, q_1]$
- $\delta(q_0, c, \diamond) = \{(q_1, \epsilon)\}$:
  - $[q_0, \diamond, q_1] \to c$
- $\delta(q_1, c, \bullet) = \{(q_1, \epsilon)\}$:
  - $[q_1, \bullet, q_1] \to c$
- $\delta(q_1, c, \diamond) = \{(q_1, \epsilon)\}$:
  - $[q_1, \diamond, q_1] \to c$
- $\delta(q_1, \epsilon, \Gamma_0) = \{(q_1, \epsilon)\}$:
  - $[q_1, \Gamma_0, q_1] \to \epsilon$
- $\delta(q_1, a, \Gamma_0) = \{(q_0, \bullet\Gamma_0)\}$:
  - $[q_1, \Gamma_0, q_0] \to a[q_0, \bullet, q_0][q_0, \Gamma_0, q_0]$
  - $[q_1, \Gamma_0, q_0] \to a[q_0, \bullet, q_1][q_1, \Gamma_0, q_0]$
  - $[q_1, \Gamma_0, q_1] \to a[q_0, \bullet, q_0][q_0, \Gamma_0, q_1]$
  - $[q_1, \Gamma_0, q_1] \to a[q_0, \bullet, q_1][q_1, \Gamma_0, q_1]$

Die gewünschte Grammatik ist nun $(N \cup \{S\}, \Sigma, P \cup \{S \to [q_0, \Gamma_0, q_0], S \to [q_0, \Gamma_0, q_1]\}, S)$.

c) Zuerst der gewünschte Konfigurationslauf:

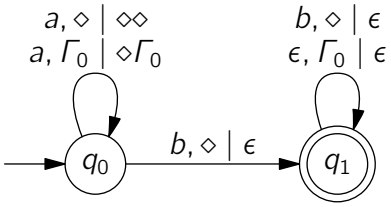
$$\begin{aligned}(q_0, abcc, \Gamma_0) &\vdash (q_0, bcc, \bullet\Gamma_0)\\ &\vdash (q_0, cc, \diamond\bullet\Gamma_0)\\ &\vdash (q_1, c, \bullet\Gamma_0)\\ &\vdash (q_1, \epsilon, \Gamma_0)\\ &\vdash (q_1, \epsilon, \epsilon)\end{aligned}$$

Der Ableitungsbaum:



Hausaufgabe 8 (Kellerautomaten und CFGs): (2 + 6 + 2 = 10 Punkte)

Betrachten Sie den folgenden Kellerautomaten M über dem Eingabealphabet $\{a, b\}$ und dem Kelleralphabet $\{\diamond, \Gamma_0\}$:



a) Geben Sie die Sprachen $L(M)$ und $N(M)$ (ohne Beweis) an.

b) Geben Sie eine kontextfreie Grammatik G an, so dass $L(G) = N(M)$ gilt. Verwenden Sie hierzu das Verfahren aus der Vorlesung. Geben Sie zu jeder Transition $\delta(q, a, X) = \{(p, Y)\}$ die daraus entstehenden Grammatikregeln an.

c) Geben Sie für das Wort $aabb$ einen Lauf durch den Automaten M an, d.h. eine Folge von Konfigurationen, die mit $(q_0, aabb, \Gamma_0)$ beginnt und mit (p, ϵ, ϵ) für ein $p \in \{q_0, q_1\}$ endet.

Geben Sie außerdem einen Ableitungsbaum für $aabb$ auf Basis der Grammatik G aus Teil (b) an.

Lösung:

a) Es gilt $L(M) = \{a^n b^m \mid n, m \in \mathbb{N}_{>0} \wedge 1 \leq m \leq n\}$ und $N(M) = \{a^n b^n \mid n \in \mathbb{N}_{>0}\}$.

b) Wir benötigen zuerst die Menge N der Nonterminale $\{[q, Z, r] \mid q, r \in Q, Z \in \Gamma\}$. Diese ist in unserem Falle

$$\{[q_0, \diamond, q_0], [q_0, \diamond, q_1], [q_0, \Gamma_0, q_0], [q_0, \Gamma_0, q_1],$$
$$[q_1, \diamond, q_0], [q_1, \diamond, q_1], [q_1, \Gamma_0, q_0], [q_1, \Gamma_0, q_1]\}$$

Wir müssen nun die Regelmenge P erzeugen. Dafür betrachten wir nun jede der Transitionen in M. Im Folgenden sind jeweils die Zusammenhänge zwischen den Teilen der erzeugenden Transition und der erzeugten Regel farblich dargestellt:

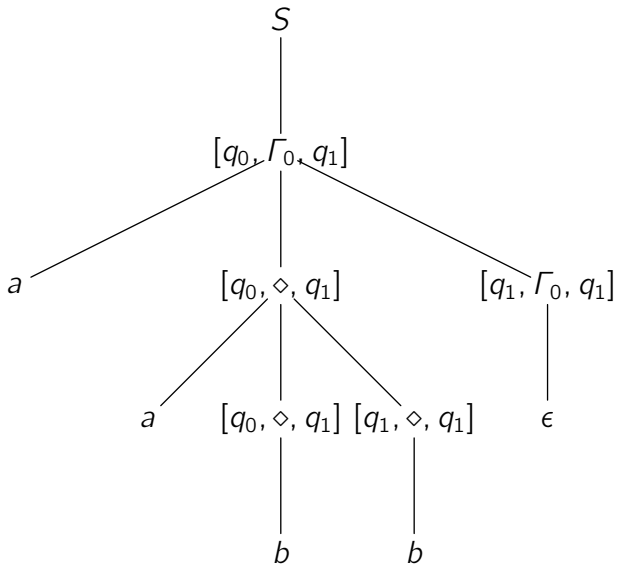
- $\delta(q_0, a, \Gamma_0) = \{(q_0, \diamond\Gamma_0)\}$:
  - $[q_0, \Gamma_0, q_0] \to a[q_0, \diamond, q_0][q_0, \Gamma_0, q_0]$
  - $[q_0, \Gamma_0, q_0] \to a[q_0, \diamond, q_1][q_1, \Gamma_0, q_0]$
  - $[q_0, \Gamma_0, q_1] \to a[q_0, \diamond, q_0][q_0, \Gamma_0, q_1]$
  - $[q_0, \Gamma_0, q_1] \to a[q_0, \diamond, q_1][q_1, \Gamma_0, q_1]$
- $\delta(q_0, a, \diamond) = \{(q_0, \diamond\diamond)\}$:
  - $[q_0, \diamond, q_0] \to a[q_0, \diamond, q_0][q_0, \diamond, q_0]$
  - $[q_0, \diamond, q_0] \to a[q_0, \diamond, q_1][q_1, \diamond, q_0]$
  - $[q_0, \diamond, q_1] \to a[q_0, \diamond, q_0][q_0, \diamond, q_1]$
  - $[q_0, \diamond, q_1] \to a[q_0, \diamond, q_1][q_1, \diamond, q_1]$
- $\delta(q_0, b, \diamond) = \{(q_1, \epsilon)\}$:
  - $[q_0, \diamond, q_1] \to b$
- $\delta(q_1, b, \diamond) = \{(q_1, \epsilon)\}$:
  - $[q_1, \diamond, q_1] \to b$
- $\delta(q_1, \epsilon, \Gamma_0) = \{(q_1, \epsilon)\}$:
  - $[q_1, \Gamma_0, q_1] \to \epsilon$

Die gewünschte Grammatik ist nun $(N \cup \{S\}, \Sigma, P \cup \{S \to [q_0, \Gamma_0, q_0], S \to [q_0, \Gamma_0, q_1]\}, S)$.

c) Zuerst der gewünschte Konfigurationslauf:

$$\begin{aligned}(q_0, aabb, \Gamma_0) &\vdash (q_0, abb, \diamond\Gamma_0)\\ &\vdash (q_0, bb, \diamond\diamond\Gamma_0)\\ &\vdash (q_1, b, \diamond\Gamma_0)\\ &\vdash (q_1, \epsilon, \Gamma_0)\\ &\vdash (q_1, \epsilon, \epsilon)\end{aligned}$$

Der Ableitungsbaum:



Tutoraufgabe 9 (Abschlusseigenschaften kontextfreier Sprachen):

Sei $\Sigma = \{a, b, c\}$ ein Alphabet. Zu welchen der drei Klassen

- deterministisch kontextfrei
- kontextfrei, aber nicht deterministisch kontextfrei
- nicht kontextfrei

gehören folgende Sprachen über Σ? Beweisen Sie Ihre Antwort ausschließlich mit Hilfe der Abschlusseigenschaften (deterministisch) kontextfreier Sprachen, d.h. Sie dürfen nicht das Pumping-Lemma, CFGs oder Automaten verwenden. Sie dürfen jedoch verwenden, dass folgende Sprachen zu folgenden Klassen gehören:

- $L_1 = \{a^m b^m c^n \mid m, n \geq 0\}$ deterministisch kontextfrei
- $L_2 = \{a^l b^m c^n \mid l, m, n \geq 0\}$ regulär
- $L_3 = \{a^m b^n a^m b^n \mid m, n \geq 0\}$ nicht kontextfrei
- $L_4 = \{a^k b^l a^m b^n \mid k, l, m, n \geq 0\}$ regulär

Außerdem dürfen Sie verwenden, dass deterministisch kontextfreie Sprachen unter Schnitt mit regulären Sprachen abgeschlossen sind.

a) $L_x = \{a^l b^m c^n \mid l, m, n \geq 0 \wedge l \neq m\}$

Beispielsweise gilt:

- $abbc \in L_x$
- $aabc \in L_x$
- $ac \in L_x$
- $abcc \notin L_x$
- $ab \notin L_x$

b) $L_y = \{a^l c^m b^n a^l b^n c^m \mid l, m, n \geq 0\}$

Beispielsweise gilt:

- $acbabc \in L_y$
- $aa \in L_y$
- $cbbc \in L_y$
- $a \notin L_y$
- $abcabc \notin L_y$

Lösung:

a) Es gilt $L_x = \overline{L_1} \cap L_2$. Aufgrund der Abgeschlossenheit deterministisch kontextfreier Sprachen unter Komplement und Schnitt mit regulären Sprachen ist L_x damit deterministisch kontextfrei.

b) Es gilt $L_3 = L_y \cap L_4$. Aufgrund der Abgeschlossenheit kontextfreier Sprachen unter Schnitt mit regulären Sprachen ist L_y damit nicht kontextfrei.

Hausaufgabe 10 (Abschlusseigenschaften kontextfreier Sprachen): (2 + 3 + 4 = 9 Punkte)

Sei $\Sigma = \{a, b, c\}$ ein Alphabet. Zu welchen der drei Klassen

- deterministisch kontextfrei
- kontextfrei, aber nicht deterministisch kontextfrei
- nicht kontextfrei

gehören folgende Sprachen über Σ? Beweisen Sie Ihre Antwort ausschließlich mit Hilfe der Abschlusseigenschaften (deterministisch) kontextfreier Sprachen, d.h. Sie dürfen nicht das Pumping-Lemma, CFGs oder Automaten verwenden. Sie dürfen jedoch verwenden, dass folgende Sprachen zu folgenden Klassen gehören:

- $L_1 = \{a^m b^m c^n \mid m, n \geq 0\}$ deterministisch kontextfrei
- $L_2 = \{a^m b^n c^n \mid m, n \geq 0\}$ deterministisch kontextfrei
- $L_3 = \{a^l b^m c^n \mid l, m, n \geq 0\}$ regulär
- $L_4 = \{a^n b^n c^n \mid n \geq 0\}$ nicht kontextfrei
- $L_5 = \{w \in \Sigma^* \mid \sharp_a(w)$ ist gerade$\}$ regulär

Außerdem dürfen Sie verwenden, dass deterministisch kontextfreie Sprachen unter Schnitt mit regulären Sprachen abgeschlossen sind.

a) $L_a = \{w \in \Sigma^* \mid w$ enthält mindestens eines der Teilwörter $ba, cb, ac, ca \vee \sharp_a(w) \neq \sharp_b(w)\}$

Beispielsweise gilt:

- $aababb \in L_a$
- $abb \in L_a$

- $caabb \in L_a$
- $cc \notin L_a$
- $aabbc \notin L_a$

b) $L_b = \{w \in \Sigma^* \mid \sharp_a(w) \text{ ist ungerade } \Rightarrow \sharp_a(w) = \sharp_b(w) = \sharp_c(w)\}$
Beispielsweise gilt:

- $acab \in L_b$
- $bb \in L_b$
- $abccababc \in L_b$
- $acc \notin L_b$
- $abbcc \notin L_b$

c) $L_c = \{w \in \Sigma^* \mid w \text{ enthält mindestens eines der Teilwörter } ba, cb, ac, ca \vee \sharp_a(w) \neq \sharp_b(w) \vee \sharp_b(w) \neq \sharp_c(w)\}$
Beispielsweise gilt:

- $caabb \in L_c$
- $cc \in L_c$
- $aabbc \in L_c$
- $aabbcc \notin L_c$

Lösung:

a) Es gilt $L_a = \overline{L_1}$. Aufgrund der Abgeschlossenheit deterministisch kontextfreier Sprachen unter Komplement ist L_a damit deterministisch kontextfrei.

b) Es gilt $L_4 = (L_b \setminus L_5) \cap L_3$. Aufgrund der Abgeschlossenheit kontextfreier Sprachen unter Schnitt und Differenz mit regulären Sprachen ist L_b damit nicht kontextfrei.

c) Es gilt $L_4 = \overline{L_c}$. Aufgrund der Abgeschlossenheit deterministisch kontextfreier Sprachen unter Komplement ist L_c damit nicht deterministisch kontextfrei. Weiterhin gilt $L_c = \overline{L_1} \cup \overline{L_2}$. Aufgrund der Abgeschlossenheit deterministisch kontextfreier Sprachen unter Komplement sind $\overline{L_1}$ und $\overline{L_2}$ deterministisch kontextfrei. Damit ist dann auch aufgrund der Abgeschlossenheit kontextfreier Sprachen unter Vereinigung L_c kontextfrei. Insgesamt ist L_c also kontextfrei, aber nicht deterministisch kontextfrei.

Prof. Dr. J. Giesl

M. Brockschmidt, F. Emmes, C. Fuhs, C. Otto, T. Ströder

Hinweise:

- Die **Hausaufgaben** sollen in Gruppen von je 2 Studierenden aus dem gleichen Tutorium bearbeitet werden.
- Die Lösungen der Hausaufgaben müssen bis Fr., 16.07.2010, 11:45 in der Globalübung abgegeben werden. Alternativ ist es möglich, die Lösungen in den Tutorien am Mittwoch abzugeben oder bis Fr., 16.07.2010 11 Uhr im Abgabekasten im Flur des LuFG I2 einzuwerfen (Ahornstr. 55, E1, 2. Etage).
- Namen und Matrikelnummern der Studierenden sowie **die Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. **Heften bzw. tackern Sie die Blätter**!
- Die **Tutoraufgaben** werden in den jeweiligen Tutorien gemeinsam besprochen und bearbeitet.
- Am **14.07.2010** finden die **letzten Tutorien** zur Vorlesung Formale Systeme, Sprachen, Prozesse statt. Dort werden Sie die Möglichkeit haben, unter Anleitung des Tutors verschiedene klausurrelevante Aufgaben aus dem Semester zu wiederholen.
- Am **8.7.2010** findet die **letzte Vorlesung** Formale Systeme, Sprachen, Prozesse im Sommersemester 2010 statt.

Hausaufgabe 1 (Induktion): (5 Punkte)

Gegeben sei ein NFA $M = (Q, \Sigma, \delta, q_0, F)$. Wir konstruieren damit die linkslineare Grammatik $G = (Q, \Sigma, P, q_o)$ mit den Produktionen $P = \{q \to ap \mid p \in \delta(q, a), q \in Q, a \in \Sigma\} \cup \{q \to \epsilon \mid q \in F\}$.
Beweisen Sie, dass für alle $p, q \in Q$, $w \in \Sigma^*$ gilt: $p \in \hat{\delta}(q, w) \Leftrightarrow q \Rightarrow_G^* wp$.[1]
Hinweis: Beweisen Sie beide Richtungen getrennt. Verwenden Sie für $\Rightarrow$ eine Induktion über die Wortlänge und für $\Leftarrow$ eine Induktion über die Ableitungslänge.

Lösung:

Wir zeigen zuerst $p \in \hat{\delta}(q, w) \Rightarrow q \Rightarrow_G^* wp$ (†) per Induktion über die Länge des Wortes w.
Induktionsanfang: Sei $|w| = 0$, d.h. $w = \epsilon$. Sei $p \in \hat{\delta}(q, \epsilon)$. Dann gilt $p = q$ und die Ableitung $q \Rightarrow_G^0 q = p$ existiert.
Wir nehmen nun als Induktionshypothese an, dass für beliebige $n \in \mathbb{N}_0$ die Aussage (†) für Wörter der Länge n bereits gilt.
Induktionsschritt: Sei $|w| = n + 1$ mit $w = w' \cdot a$, $w' \in \Sigma^n, a \in \Sigma$. Dann gibt es ein q' mit $p \in \delta(q', a)$ und $q' \in \hat{\delta}(q, w')$. Nach Induktionshypothese gilt $q \Rightarrow_G^* w'q'$. Nach Konstruktion gilt außerdem $q' \to ap \in P$. Damit können wir die Ableitung $q \Rightarrow_G^* w'q' \Rightarrow_{q' \to ap} w'ap = wp$ bilden.

Wir zeigen nun die Rückrichtung $q \Rightarrow_G^* wp \Rightarrow p \in \hat{\delta}(q, w)$ (‡) per Induktion über die Länge n der Ableitung.
Induktionsanfang: Sei $n = 0$, d.h. $q \Rightarrow_g^0 q = \epsilon q$. Es gilt offensichtlich $q \in \hat{\delta}(q, \epsilon)$.
Wir nehmen nun als Induktionshypothese an, dass für beliebige $n \in \mathbb{N}_0$ die Aussage (‡) für Ableitungen der Länge n bereits gilt.
Induktionsschritt: Nach Konstruktion von G gibt es nur linkslineare Regeln. Es gibt also ein $q' \in Q$ mit einer Regel $q' \to ap$ mit $q \Rightarrow_G^n w'q' \Rightarrow_{q' \to ap} w'ap = wp$. Nach Induktionshypothese gilt $q' \in \hat{\delta}(q, w')$ und nach Konstruktion der Regel $q' \to ap$ gilt auch $p \in \delta(q', a)$. Damit gilt dann direkt $p \in \hat{\delta}(q, w)$.

---

[1] Dies ist Lemma 4.0.4 aus der Vorlesung.

Tutoraufgabe 2 (Kontextsensitive Sprachen):

Um zu zeigen, dass L_i für $i \in \{1, 2\}$ eine kontextsensitive Sprache ist, geben Sie jeweils eine monotone Grammatik G_i an, so dass $L(G_i) = L_i$ gilt. Geben Sie außerdem eine Ableitung für das Wort w_i mit Ihrer Grammatik an.

a) $L_1 = \{a^n b^n c^n \mid n \in \mathbb{N}_0\}$ und $w_1 = aabbcc$.

b) $L_2 = \{a^n b^m c^n d^m \mid n, m \in \mathbb{N}_{>0}\}$ und $w_2 = aabbbccddd$.

Lösung:

a)

$$S \to \epsilon \quad (1)$$
$$S \to T \quad (2)$$
$$T \to aTBC \quad (3)$$
$$T \to aBC \quad (4)$$
$$CB \to BC \quad (5)$$
$$aB \to ab \quad (6)$$
$$bB \to bb \quad (7)$$
$$bC \to bc \quad (8)$$
$$cC \to cc \quad (9)$$

Hier werden mit Regeln (3) und (4) für jedes Terminalsymbol a jeweils gleich viele B und C erzeugt. Diese sind noch nicht notwendigerweise in der richtigen Ordnung, können aber mit Regel (5) Schritt für Schritt so geordnet werden, dass kein B mehr rechts von einem C steht. Nun kann man von den a-Symbolen auf der linken Seite ausgehend jeweils das erste Nonterminal in ein Terminal umwandeln.

Beispielableitung für $aabbcc$ (abgeleitete Teile sind jeweils unterstrichen):

$$\underline{S}$$
$$\Rightarrow^{(2)} \underline{T}$$
$$\Rightarrow^{(3)} a\underline{T}BC$$
$$\Rightarrow^{(4)} aaB\underline{CB}C$$
$$\Rightarrow^{(5)} a\underline{aB}BCC$$
$$\Rightarrow^{(6)} aa\underline{bB}CC$$
$$\Rightarrow^{(7)} aab\underline{bC}C$$
$$\Rightarrow^{(8)} aabb\underline{cC}$$
$$\Rightarrow^{(9)} aabbcc$$

b)

$$S \to aNM \quad (1)$$
$$S \to aM \quad (2)$$
$$N \to aNY \quad (3)$$
$$N \to aY \quad (4)$$
$$M \to bMd \quad (5)$$
$$M \to bXd \quad (6)$$
$$Yb \to bY \quad (7)$$
$$YX \to Xc \quad (8)$$
$$X \to c \quad (9)$$

Hier werden die Teillängen n, m durch die Anzahl der N- bzw. M-Produktionen festgelegt. Während die a-, b- und c-Terminale bereits beim Verwenden der N und M-Produktionen aufgebaut werden, wird das Nonterminal Y verwendet, um die Anzahl der noch zu erzeugenden c-Terminale zu zählen. Das Nonterminal X wird genau einmal erzeugt und markiert die Stelle, an der die c-Terminale eingefügt werden müssen.

Mit Hilfe von Produktion (7) werden dann die vor den bereits erzeugten b stehenden Y im Wort nach rechts verschoben, bis sie X erreichen. Dort werden sie dann durch ein c ersetzt. Zuletzt kann X durch ein einziges c ersetzt werden.

Beispielableitung für $aabbbccddd$ (abgeleitete Teile sind jeweils unterstrichen):

$$\begin{aligned}
&\underline{S}\\
\Rightarrow^{(1)} &a\underline{NM}\\
\Rightarrow^{(4)} &aaY\underline{M}\\
\Rightarrow^{(5)} &aaYb\underline{M}d\\
\Rightarrow^{(5)} &aaYbb\underline{M}dd\\
\Rightarrow^{(6)} &aa\underline{Yb}bbXddd\\
\Rightarrow^{(7)} &aab\underline{Yb}bXddd\\
\Rightarrow^{(7)} &aabb\underline{Yb}Xddd\\
\Rightarrow^{(7)} &aabbb\underline{YX}ddd\\
\Rightarrow^{(8)} &aabbb\underline{X}cddd\\
\Rightarrow^{(9)} &aabbbccddd
\end{aligned}$$

Hausaufgabe 3 (Kontextsensitive Sprachen): (3 + 3 + 3 = 9 Punkte)

Um zu zeigen, dass L_i für $i \in \{3, 4, 5\}$ eine kontextsensitive Sprache ist, geben Sie jeweils eine monotone Grammatik G_i an, so dass $L(G_i) = L_i$ gilt. Geben Sie außerdem eine Ableitung für das Wort w_i mit Ihrer Grammatik an.

a) $L_3 = \{w \in \{a, b, c\}^* \mid \sharp_a(w) = \sharp_b(w) = \sharp_c(w)\}$ und $w_3 = caabcb$.

b) $L_4 = \{(ab)^n(cb)^n(ac)^n \mid n \in \mathbb{N}_0\}$ und $w_4 = ababcbcbacac$.

c) $L_5 = \{a^n b^m c^k \mid n, m, k \in \mathbb{N}_{>0} \wedge n \leq m \leq k\}$ und $w_5 = aabbccc$.

Lösung:

a)

$$S \to \epsilon \quad (1)$$
$$S \to T \quad (2)$$
$$T \to TT \quad (3)$$
$$T \to ABC \quad (4)$$
$$AB \to BA \quad (5)$$
$$AC \to CA \quad (6)$$
$$BA \to AB \quad (7)$$
$$BC \to CB \quad (8)$$
$$CA \to AC \quad (9)$$
$$CB \to BC \quad (10)$$
$$A \to a \quad (11)$$
$$B \to b \quad (12)$$
$$C \to c \quad (13)$$

Hier werden mit den Regeln (3) und (4) eine beliebige Anzahl von ABC-Gruppen erzeugt. Mit Hilfe der Regeln (5)-(10) können diese dann frei vertauscht werden. Mit den Regeln (11)-(13) werden die Nonterminale dann in die entsprechenden Terminale übersetzt.

Beispielableitung für $caabcb$ (abgeleitete Teile sind jeweils unterstrichen):

$$\begin{aligned}
&\underline{S}\\
\Rightarrow^{(2)} &\underline{T}\\
\Rightarrow^{(3)} &\underline{T}T\\
\Rightarrow^{(4)} &ABC\underline{T}\\
\Rightarrow^{(4)} &A\underline{BC}ABC\\
\Rightarrow^{(8)} &\underline{AC}BABC\\
\Rightarrow^{(6)} &CA\underline{BA}BC\\
\Rightarrow^{(7)} &CAAB\underline{BC}\\
\Rightarrow^{(8)} &\underline{CAABCB}\\
\Rightarrow^{*} &\ caabcb
\end{aligned}$$

b)

$$S \to \epsilon \quad (1)$$
$$S \to T \quad (2)$$
$$T \to abTHK \quad (3)$$
$$T \to abHK \quad (4)$$
$$KH \to HK \quad (5)$$
$$abH \to abcb \quad (6)$$
$$cbH \to cbcb \quad (7)$$
$$cbK \to cbac \quad (8)$$
$$acK \to acac \quad (9)$$

Die Grammatik ist bis auf die Terminalsymbole identisch zur Grammatik für $L_1 = \{a^n b^n c^n \mid n \in \mathbb{N}_0\}$.
Beispielableitung für *ababcbcbacac* (abgeleitete Teile sind jeweils unterstrichen):

$$\begin{aligned}
&\underline{S}\\
\Rightarrow^{(2)} &\underline{T}\\
\Rightarrow^{(3)} &ab\underline{T}HK\\
\Rightarrow^{(4)} &ababH\underline{KH}K\\
\Rightarrow^{(5)} &ab\underline{abH}HKK\\
\Rightarrow^{(6)} &abab\underline{cbH}KK\\
\Rightarrow^{(7)} &ababcbc\underline{bK}K\\
\Rightarrow^{(8)} &ababcbcb\underline{acK}\\
\Rightarrow^{(9)} &ababcbcbacac
\end{aligned}$$

c)

$$S \to aTbX \quad (1)$$
$$S \to abX \quad (2)$$
$$T \to aTbC \quad (3)$$
$$T \to TbC \quad (4)$$
$$T \to TC \quad (5)$$
$$T \to bC \quad (6)$$
$$T \to C \quad (7)$$
$$Cb \to bC \quad (8)$$
$$CX \to Xc \quad (9)$$
$$X \to c \quad (10)$$

Hier werden *a*- und *b*-Symbole erneut beim initialen Aufbau mit Hilfe der Produktionen für S und G generiert. Mit X wird die Stelle im Wort markiert, in der im Folgenden die *c*-Symbole erzeugt werden müssen, deren Zahl von der Anzahl der erzeugten C-Nonterminale abhängt. Diese werden mitten im Wort erzeugt und können dann mit Regel 8 nach rechts bewegt werden.

Beispielableitung für $aabbccc$ (abgeleitete Teile sind jeweils unterstrichen):

$$
\begin{aligned}
&\underline{S}\\
\Rightarrow^{(1)} &a\underline{T}bX\\
\Rightarrow^{(3)} &aa\underline{T}bCbX\\
\Rightarrow^{(4)} &aaCb\underline{Cb}X\\
\Rightarrow^{(8)} &aa\underline{Cb}bCX\\
\Rightarrow^{(8)} &aab\underline{Cb}CX\\
\Rightarrow^{(8)} &aabbC\underline{CX}\\
\Rightarrow^{(9)} &aabb\underline{CX}c\\
\Rightarrow^{(9)} &aabb\underline{X}cc\\
\Rightarrow^{(10)} &aabbccc
\end{aligned}
$$

Tutoraufgabe 4 (Synchronisiertes Produkt):

```
while (true) {
    x := 1;                          x := 0;
    /* wait for x != 1 */            /* wait for x != 0 */
    while (x = 1);                   while (x = 0);
    print ('a');                     print ('b');
}
```

Programm P_1 | Programm P_2

a) Modellieren Sie den Kontrollfluss von P_1 durch einen NFA M_1 mit 3 Zuständen, sowie den den Kontrollfluss von P_2 durch einen NFA M_2 mit 4 Zuständen. Verwenden Sie hierbei für beide Automaten die Signatur $\Sigma = \{\texttt{x = 0?}, \texttt{x = 1?}, \texttt{x := 0}, \texttt{x := 1}, \texttt{print('a')}, \texttt{print('b')}\}$.

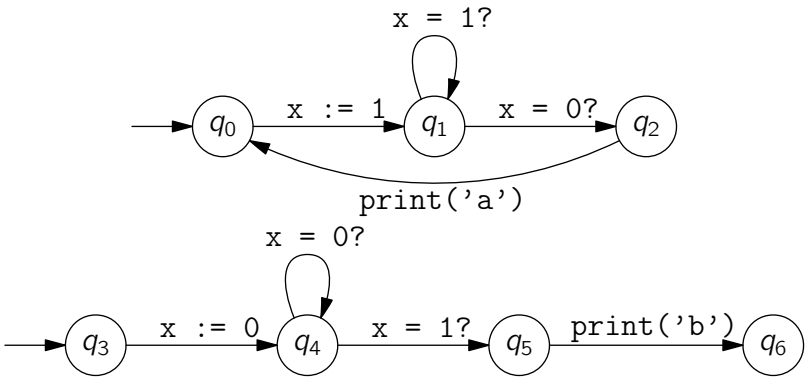
b) Berechnen Sie das unsynchronisierte Produkt $M_1 \sqcup\!\sqcup M_2$. Geben Sie den resultierenden NFA an.

c) Modellieren Sie die boolesche Variable `x` des Programms durch den NFA B_x. Berechnen Sie den NFA $M := (M_1 \sqcup\!\sqcup M_2) \circ B_x$ und geben Sie die Automaten B_x und M an.

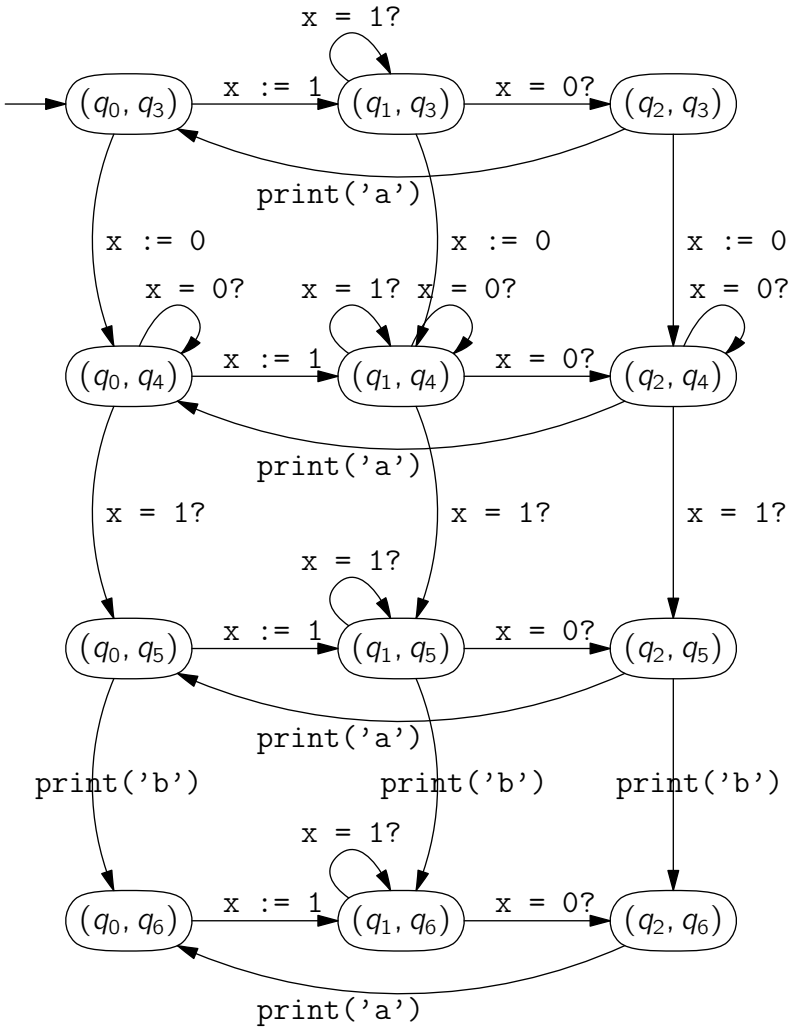
d) Ist es möglich, dass Programm P_1 und P_2 (wenn sie gleichzeitig ausgeführt werden) die Zeichenfolge `aba` ausgeben? Begründen Sie ihre Antwort mit dem NFA M.

Lösung:

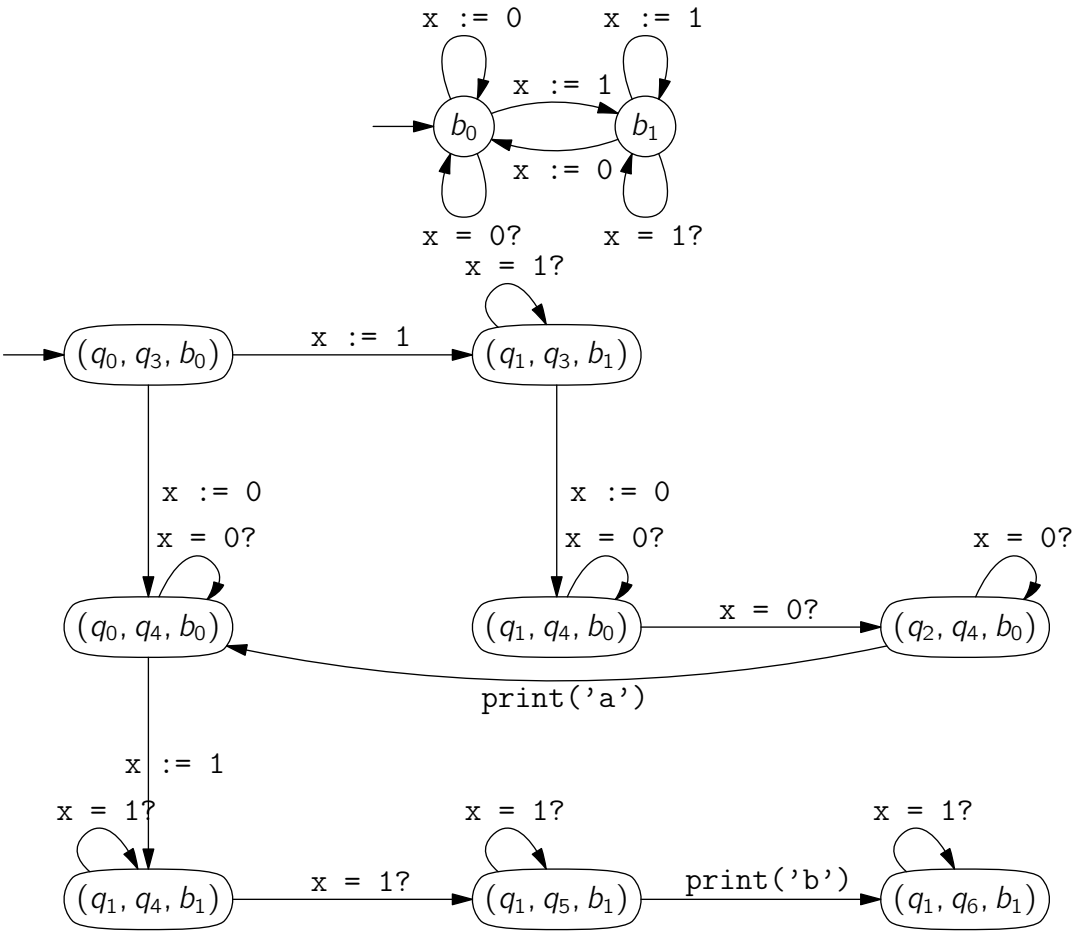
a)


x = 1?
x := 1
x = 0?
q_0
q_1
q_2
print('a')
x = 0?
x := 0
x = 1?
print('b')
q_3
q_4
q_5
q_6

b)

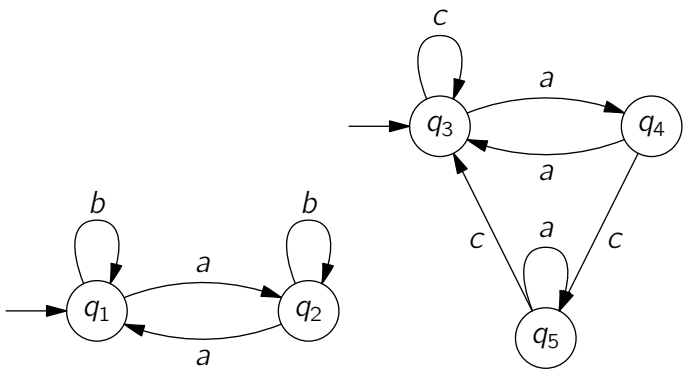

x = 1?
x := 1
x = 0?
(q_0, q_3)
(q_1, q_3)
(q_2, q_3)
print('a')
x := 0
x := 0
x := 0
x = 0?
x = 1?
x = 0?
x = 0?
(q_0, q_4)
(q_1, q_4)
(q_2, q_4)
print('a')
x = 1?
x = 1?
x = 1?
x = 1?
(q_0, q_5)
(q_1, q_5)
(q_2, q_5)
print('a')
print('b')
print('b')
print('b')
x = 1?
(q_0, q_6)
(q_1, q_6)
(q_2, q_6)
print('a')

c)



d) Nein, der NFA M beschreibt keinen Pfad, auf dem nach einem `print('b')` noch ein `print('a')` folgen kann.

Hausaufgabe 5 (Synchronisiertes Produkt): (4 Punkte)



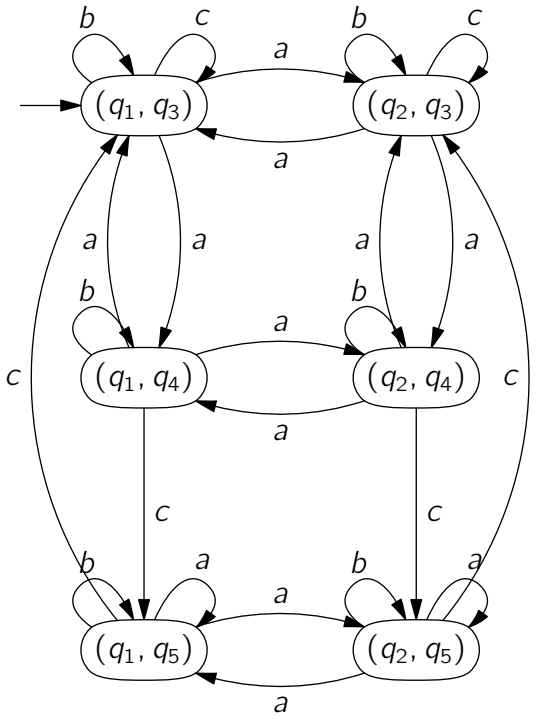
Gegeben seien die DFAs M_1 und M_2. Berechnen Sie:

a) $M_1 \sqcup\!\sqcup M_2$

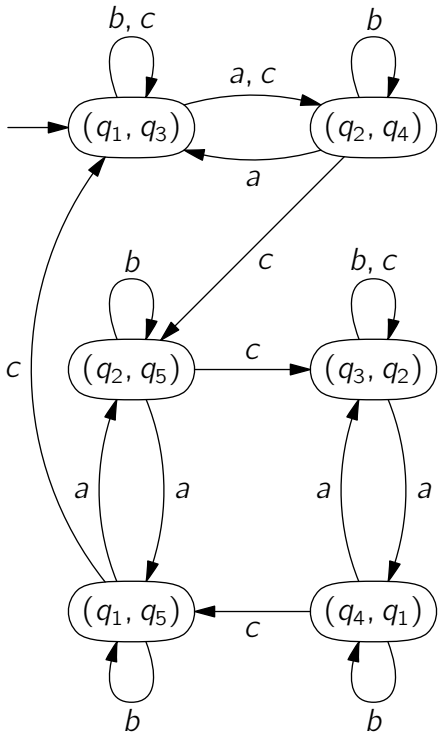
b) $M_1 \circ M_2$

Lösung:

a)



b)



Tutoraufgabe 6 (Petrinetze):

In dieser Aufgabe sollen Sie Petrinetze nutzen, um Prozesse aus dem Alltag zu modellieren. Verwenden Sie die unterstrichenen Begriffe als Stellen und Transitionen und wählen Sie geeignete Kanten zwischen diesen, um die im Aufgabentext dokumentierten Zusammenhänge darzustellen.

a) Wir betrachten ein Callcenter. Dort gibt es Agenten, die Anrufe entgegennehmen und den berichteten Problemfall im internen Ticketsystem registrieren. Im Folgenden versuchen die Agenten nun, das Problem direkt zu lösen, z.B. mit Hilfe eines Fragenkataloges und Standardantworten. Gelingt dies, ist der Problemfall abgeschlossen und der Agent steht für einen weiteren Anruf zur Verfügung. In allen anderen Fällen wird der Problemfall eskaliert und im Ticketsystem als ungelöstes Problem markiert, damit ein Techniker den Fall analysieren kann.

Erstellen Sie nun ein Petrinetz, das diesen Prozess dokumentiert.

b) Wir betrachten nun die Technik-Abteilung eines Software-Unternehmens. In einem internen System werden berichtete Probleme verwaltet. Hat ein Techniker gerade keine andere Aufgabe, kann eines dieser Probleme analysiert werden. Dabei wird ein minimaler Testfall erstellt, der das Problem reproduziert. Danach kann sich der Techniker wieder anderen Aufgaben zuwenden.

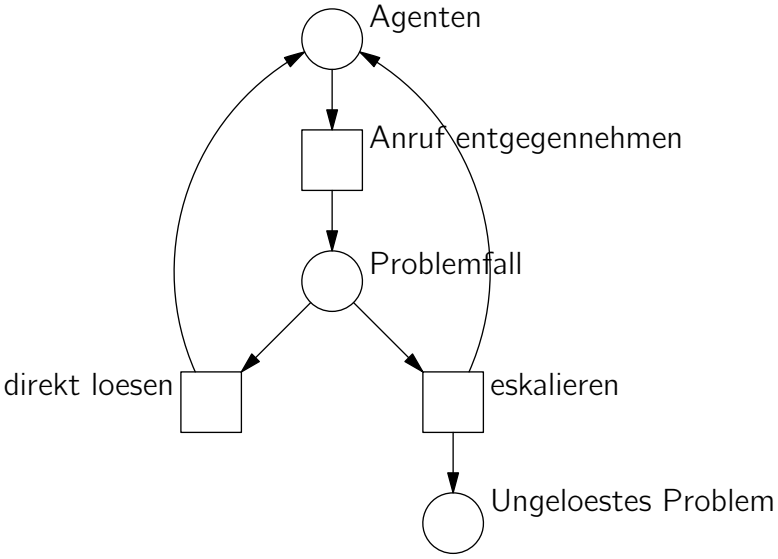
Mit Hilfe des Testfalls kann ein Techniker diesen lösen. Um zu verhindern, dass das Problem wieder eingeführt wird, wird der Testfall als Regressionstest gespeichert.

Erstellen Sie nun ein Petrinetz, das diesen Prozess dokumentiert.

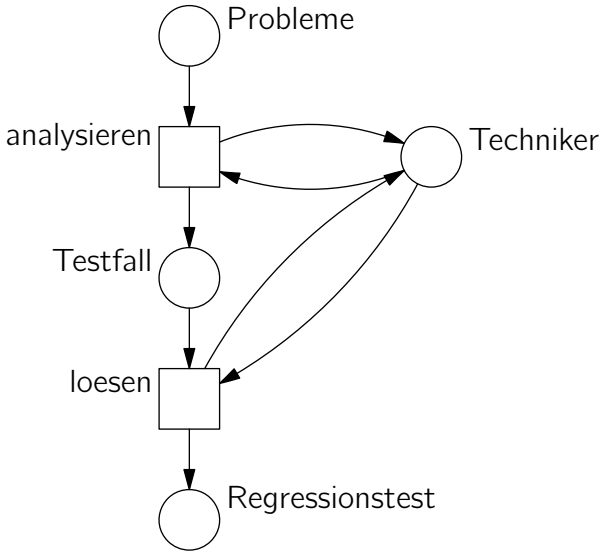
c) Erklären Sie, wie Sie die beiden Petrinetze sinnvoll miteinander verbinden können. Beschreiben Sie, welche Komponenten sich in den beiden Netzen entsprechen und welchen Prozess das Ergebnis modelliert.

Lösung:

a)



b)



c) Die Stellen „Ungeloest“ im Callcenter-Petrinetz und „Probleme“ im Petrinetz für die Technik-Abteilung entsprechen sich. Die Komposition der beiden Netze an dieser Stelle entspricht dann der Übergabe eines Problems an einen Techniker.

Hausaufgabe 7 (Petrinetze): (3 + 2 = 5 Punkte)

In dieser Aufgabe sollen Sie Petrinetze nutzen, um Prozesse aus dem Alltag zu modellieren. Verwenden Sie die unterstrichenen Begriffe als Stellen und Transitionen und wählen Sie geeignete Kanten zwischen diesen, um die im Aufgabentext dokumentierten Zusammenhänge darzustellen.

a) Wir betrachten den Übungsbetrieb einer Vorlesung "Formale Sprachen, Automaten und Petrinetze". Hier werden Übungsblätter bereitgestellt, die dann von Studierenden bearbeitet werden. Die so entstehenden Übungsabgaben können dann entweder in einem Tutorium abgegeben oder in den Abgabekasten geworfen werden. Abgaben im Tutorium landen direkt auf dem Korrekturstapel eines Tutors. Abgaben im Übungskasten werden von einem Assistenten sortiert (wenn er dafür Zeit hat) und kommen dann auf den Korrekturstapel eines Tutors.

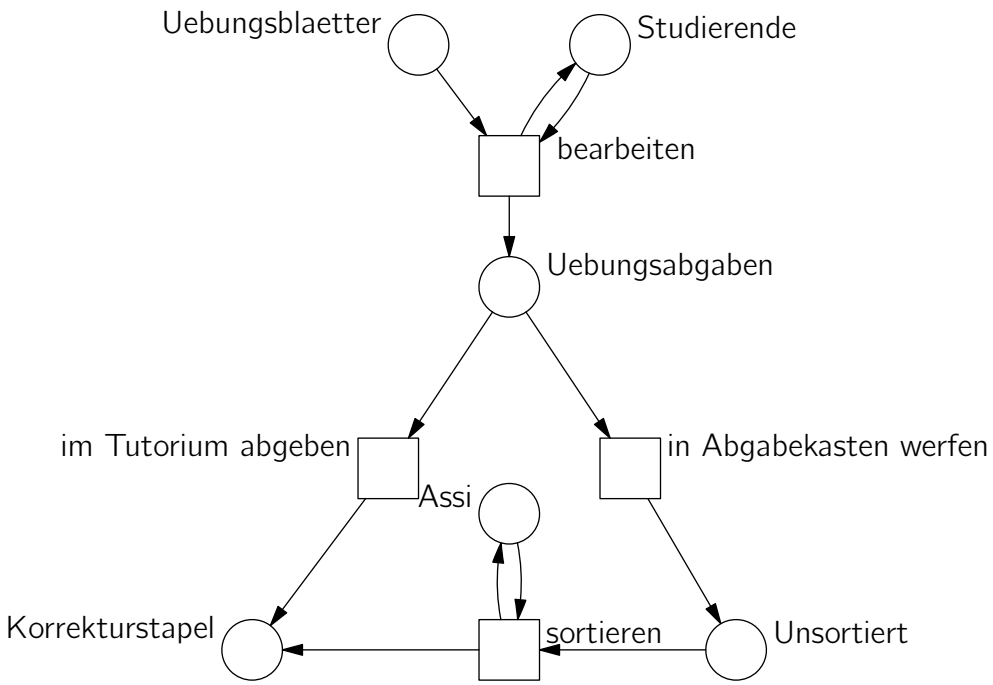
Erstellen Sie nun ein Petrinetz, das diesen Prozess dokumentiert.

b) Wir betrachten nun die Arbeit von Tutoren. Übungsabgaben vom Korrekturstapel werden korrigiert, wenn der Tutor gerade keine anderen Verpflichtungen hat. Bei der Korrektur wird ein Rückgabestapel aufgebaut. Gleichzeitig entsteht eine Punkteliste. Die korrigierten Übungsabgaben vom Rückgabestapel werden zurückgeben, wenn ein Tutor gerade nicht anderweitig beschäftigt ist, meist im Tutorium. Außerdem werden die Punkte von der Punkteliste eingetragen, wenn der Tutor Zeit hat.

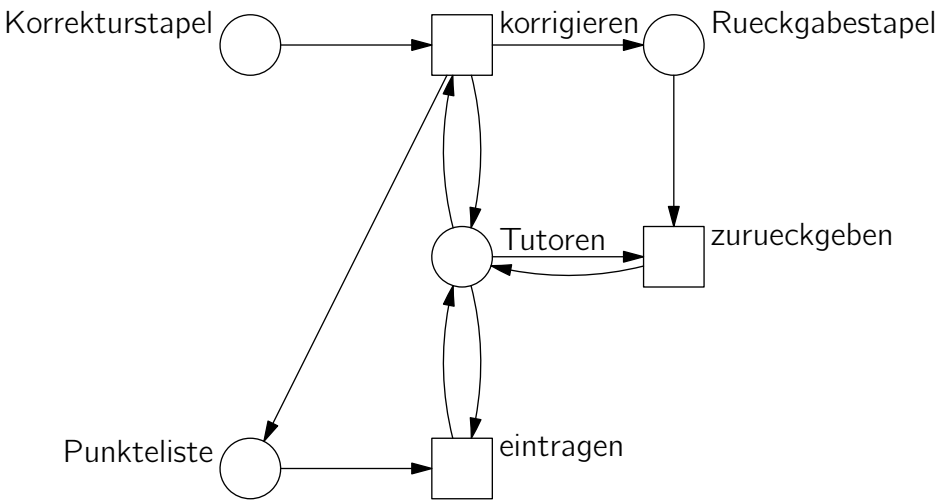
Erstellen Sie nun ein Petrinetz, das diesen Prozess dokumentiert.

Lösung:

a)

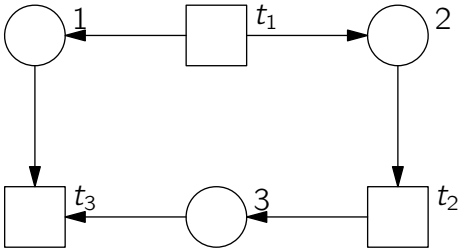


b)



Tutoraufgabe 8 (Erreichbarkeit):

Gegeben sei folgendes Petrinetz P:



a) Berechnen Sie die Inzidenzmatrix $D = D^+ + D^-$.

b) Zeigen oder widerlegen Sie, dass folgende Ableitungsequenzen existieren. Geben Sie im Falle der Existenz die Folge der ausgelösten Transitionen an und im Falle der Nichtexistenz einen Beweis mit Hilfe von Satz 5.2.5.

Hinweis: Verwenden Sie zum Konstruieren einer Folge für die Ableitung $m \longrightarrow^* m'$ eine Lösung x der Gleichung $m' = m + x \cdot D$.

- $(0,0,0) \longrightarrow^* (3,0,2)$
- $(0,0,0) \longrightarrow^* (3,2,1)$

Lösung:

a)

$$D^+ = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

$$D^- = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ -1 & 0 & -1 \end{pmatrix}$$

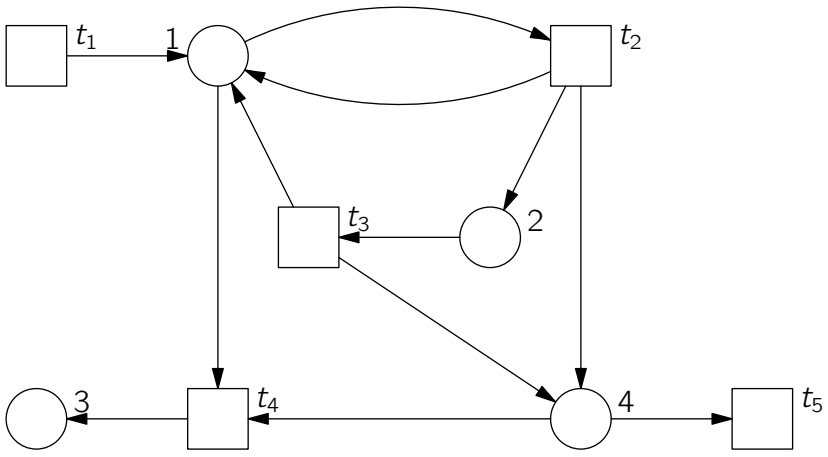
$$D = \begin{pmatrix} 1 & 1 & 0 \\ 0 & -1 & 1 \\ -1 & 0 & -1 \end{pmatrix}$$

b)
- Nach Satz 5.2.5 der Vorlesung, existiert eine solche Transitionenfolge nicht, da $(3,0,2) = \vec{x} \cdot D$ keine Lösung hat.
- Wir suchen eine Lösung für x in $D^T\vec{x} = (3,2,1)^T$. Dies ist der Fall für $\vec{x} = (3,1,0)^T$ (d.h. drei mal die Transition t_1 und ein mal die Transition t_2). Wir suchen nun eine Reihenfolge dieser Transitionen, in der sie anwendbar sind:

$$\begin{aligned} (0,0,0) &\xrightarrow{t_1} (1,1,0) \\ &\xrightarrow{t_2} (1,0,1) \\ &\xrightarrow{t_1} (2,1,1) \\ &\xrightarrow{t_1} (3,2,1) \end{aligned}$$

Hausaufgabe 9 (Erreichbarkeit): (2 + 2 = 4 Punkte)

Gegeben sei folgendes Petrinetz P:



a) Berechnen Sie die Inzidenzmatrix $D = D^+ + D^-$.

b) Zeigen oder widerlegen Sie, dass folgende Ableitungsequenzen existieren. Geben Sie im Falle der Existenz die Folge der ausgelösten Transitionen an und im Falle der Nichtexistenz einen Beweis mit Hilfe von Satz 5.2.5.

Hinweis: Verwenden Sie zum Konstruieren einer Folge für die Ableitung $m \longrightarrow^* m'$ eine Lösung x der Gleichung $m' = m + x \cdot D$.

- $(0,0,0,0) \longrightarrow^* (1,1,1,1)$
- $(0,0,0,0) \longrightarrow^* (1,2,3,4)$

Lösung:

a)

$$D^+ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$D^- = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$D = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & -1 & 0 & 1 \\ -1 & 0 & 1 & -1 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

b)

- Wir suchen eine Lösung für x in $D^T\vec{x} = (1,1,1,1)^T$. Dies ist der Fall für $\vec{x} = (1,2,1,1)^T$. Wir suchen nun eine Reihenfolge dieser Transitionen, in der sie anwendbar sind:

$$\begin{aligned} (0,0,0,0) &\xrightarrow{t_1} (1,0,0,0) \\ &\xrightarrow{t_2} (1,1,0,1) \\ &\xrightarrow{t_4} (0,1,1,0) \\ &\xrightarrow{t_3} (1,0,1,1) \\ &\xrightarrow{t_2} (1,1,1,2) \\ &\xrightarrow{t_5} (1,1,1,1) \end{aligned}$$

- Wir suchen eine Lösung für x in $D^T\vec{x} = (3,2,1)^T$. Dies ist der Fall für $\vec{x} = (1,5,3,3,1)^T$. Wir suchen

nun eine Reihenfolge dieser Transitionen, in der sie anwendbar sind:

$$
\begin{aligned}
(0,0,0,0) &\xrightarrow{t_1} (1,0,0,0)\\
&\xrightarrow{t_2} (1,1,0,1)\\
&\xrightarrow{t_2} (1,2,0,2)\\
&\xrightarrow{t_2} (1,3,0,3)\\
&\xrightarrow{t_2} (1,4,0,4)\\
&\xrightarrow{t_2} (1,5,0,5)\\
&\xrightarrow{t_3} (2,4,0,6)\\
&\xrightarrow{t_3} (3,3,0,7)\\
&\xrightarrow{t_3} (4,2,0,8)\\
&\xrightarrow{t_4} (3,2,1,7)\\
&\xrightarrow{t_4} (2,2,2,6)\\
&\xrightarrow{t_4} (1,2,3,5)\\
&\xrightarrow{t_5} (1,2,3,4)
\end{aligned}
$$