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 $\Sigma$. 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 $\Sigma$ akzeptiert.

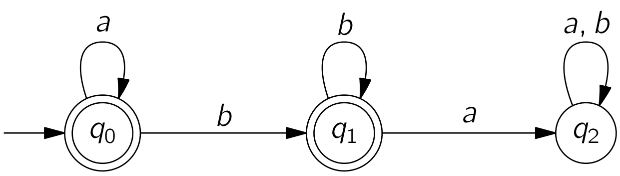
**a)** $L_1 = \{w \mid w = a^n b^m \text{ für } m, n \geq 0 \text{ und } m \neq 2n\} \cup \Sigma^* ba \Sigma^*$

**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 $\Sigma$ das Alphabet aus der vorherigen Aufgabe. Sei $L_3 := \{a^n b^n \mid n \geq 0\}$ eine Sprache über $\Sigma$. 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 $\Sigma$ akzeptiert.

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

### Lösung:

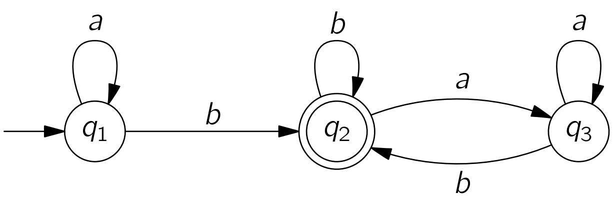
Es gilt $L_3 = (\Sigma^* \setminus L_4) \cap L'$. Da es einen DFA gibt, der $L'$ akzeptiert, es aber keinen DFA gibt, der $L_3$ akzeptiert, kann aufgrund der Abgeschlossenheit unter Schnitt und Komplement von Sprachen, die von einem DFA akzeptiert werden können, die Sprache $L_4$ nicht von einem DFA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

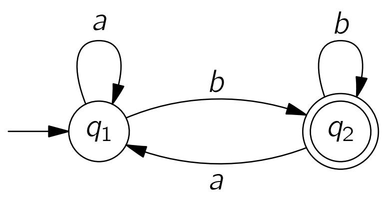
## Tutoraufgabe 3 (DFAs aus regulären Ausdrücken):

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

### Lösung:

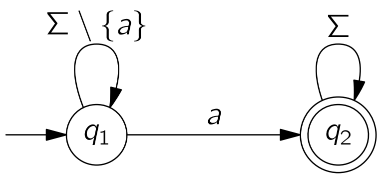
Zwei Lösungen, wobei die rechte minimal ist.




## Tutoraufgabe 4 (DFAs aus Wörtern):

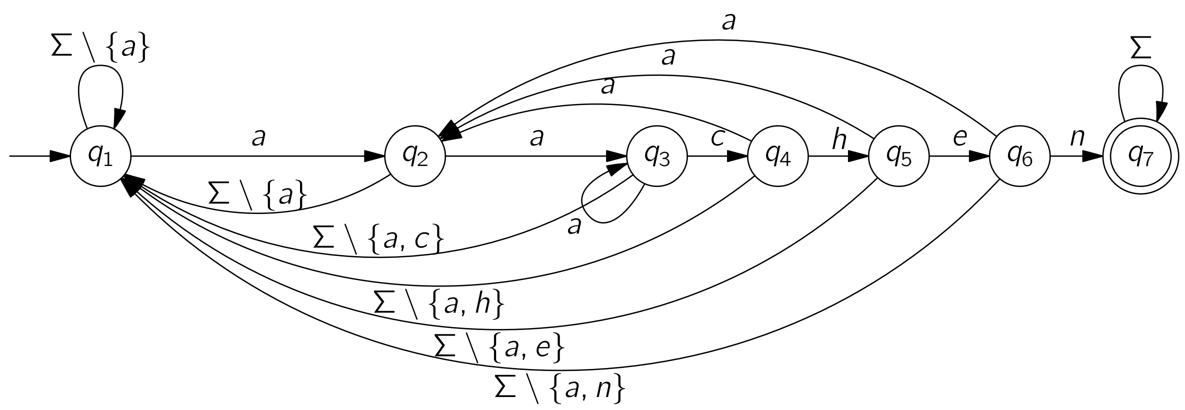
Sei $\Sigma := \{a, \ldots, z\}$ ein Alphabet. Wir führen als verkürzende Schreibweise ein, dass Kanten in einem Automaten mit Mengen (z.B. $\Sigma \setminus \{a\}$) beschriftet werden können und meinen damit, dass die beschriftete Kante einem Übergang entspricht, der für alle Symbole aus der Menge erlaubt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

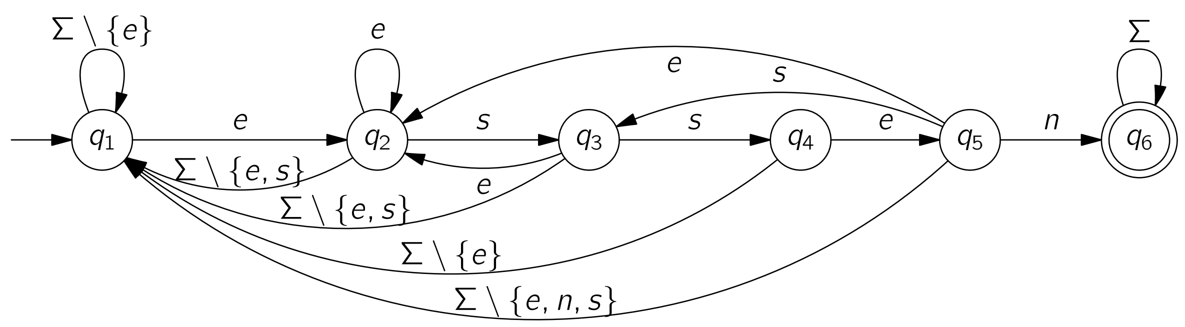
**Lösung:**




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

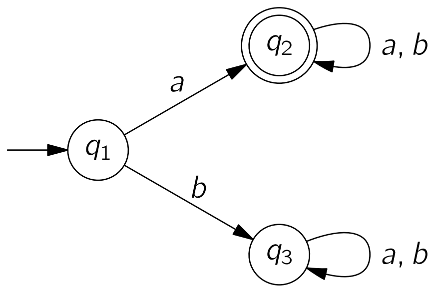
Sei $\Sigma$ ein Alphabet. Konstruieren Sie einen Automaten, der die Wörter erkennt, die mindestens einmal das Wort *essen* enthalten, also die Sprache $\Sigma^* essen \Sigma^*$.
Hinweis: Um die Darstellung des Automaten zu vereinfachen, nutzen Sie am besten die verkürzende Schreibweise, die in Tutoraufgabe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

**Lösung:**



## Tutoraufgabe 6 ($L^k_{ij}$-Algorithmus):

Bestimmen Sie mit Hilfe des $L^k_{ij}$-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, $L^3_{1,2}$ zu bestimmen.

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

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

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

$L^0_{1,2} = a$

$L^0_{1,1} = \varepsilon$

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

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

$L^0_{2,2} = \varepsilon + a + b \ L^0_{2,1} = \emptyset$

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

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

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

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

$L^0_{1,3} = b$

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

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

$L^0_{2,3} = \emptyset \ L^0_{2,1} = \emptyset$

$\cdots = \emptyset$

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

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

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

$L^0_{3,3} = \varepsilon + a + b \ L^0_{3,1} = \emptyset$

$\cdots = \varepsilon + a + b$

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

$L^0_{3,2} = \emptyset \ L^0_{3,1} = \emptyset$

$\cdots = \emptyset$

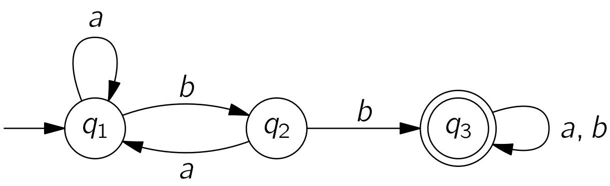
$\cdots = (\varepsilon + a + b) + \emptyset(\varepsilon + a + b)^* \emptyset = \varepsilon + a + b$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

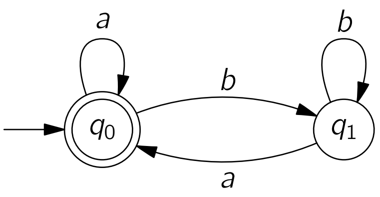
$$\cdots = (\varepsilon + a + b) + (\emptyset)(\varepsilon + a(\varepsilon + a)^* b)^* b = \varepsilon + a + b$$

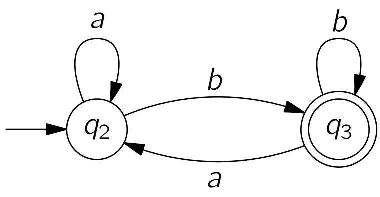
$$\cdots = (a^* b(\varepsilon + a^+ b)^* b) + (a^* b(\varepsilon + a^+ b)^* b)(\varepsilon + a + b)^*(\varepsilon + a + b) = a^* b(\varepsilon + a^+ b)^* b\Sigma^* = (a + b)^* bb(a + b)^*$$

## 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 $\Sigma^*$ mit $\Sigma = \{a, b\}$ von einem der folgenden Automaten akzeptiert wird.

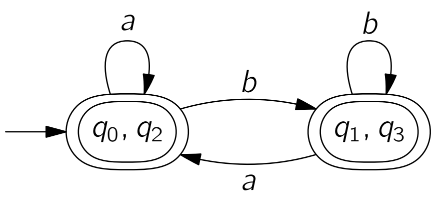



## Lösung:

**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.

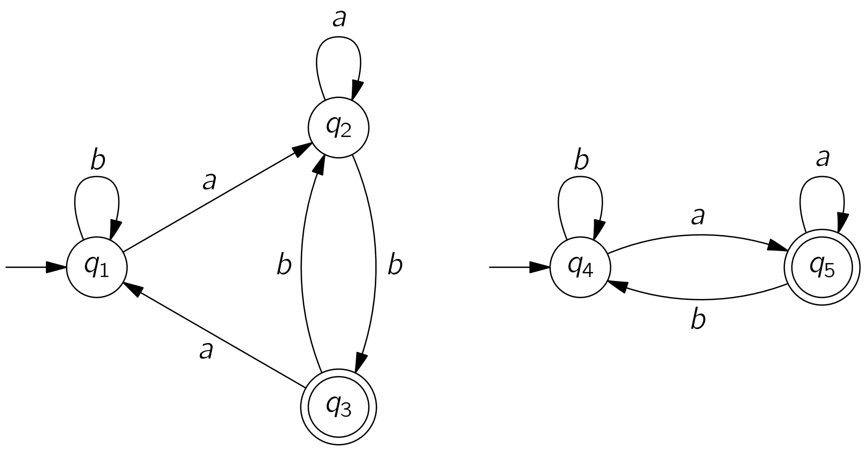
**b)** Der modifizierte Produktautomat




akzeptiert offensichtlich jedes Wort aus $\Sigma^*$, da er nur Endzustände besitzt.

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

Gegeben seien zwei endliche Automaten.

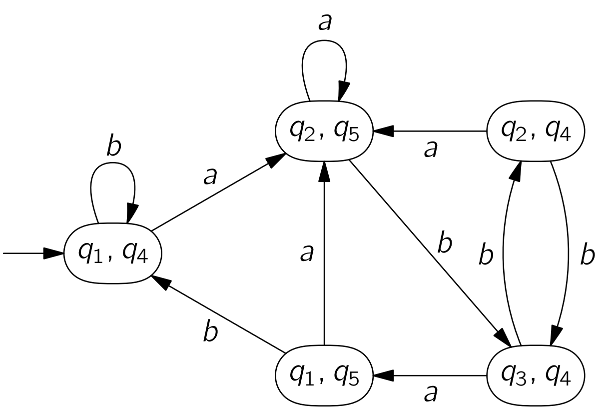



**a)** Konstruieren Sie den Produktautomaten der beiden Automaten.

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

**Lösung:**

**a)**



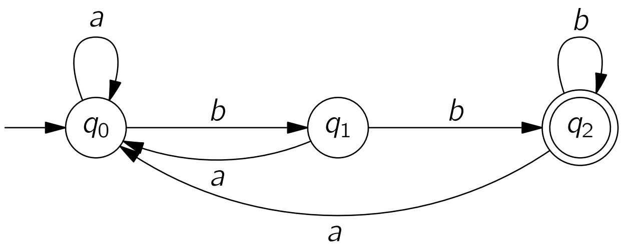

**b)** Falls es kein Wort gibt, das von beiden Automaten erkannt wird, hat ihr Produkt keinen erreichbaren Endzustand. Dies ist hier der Fall.

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 $b$s 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.

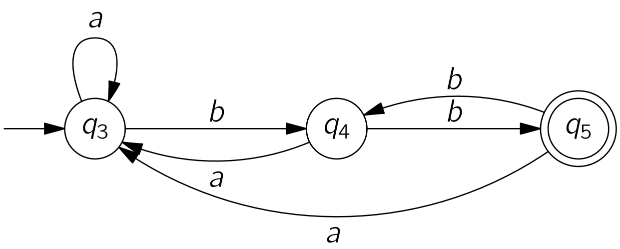
Hinweis: Zur Lösung der Aufgabe hilft es, sich klar zu machen, dass $L(M) \subseteq L(M')$ genau dann gilt, wenn $w \in L(M) \rightarrow w \in L(M')$ für alle $w \in \Sigma^*$ gilt.

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:

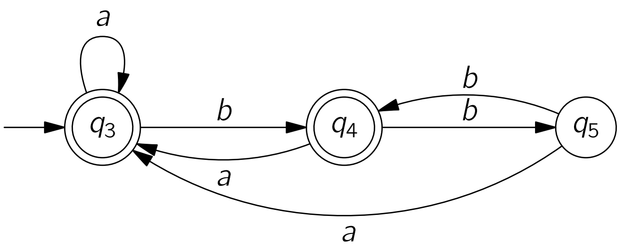
$$
\begin{aligned}
& L(M) \subseteq L(M') & \\
\Leftrightarrow\ & w \in L(M) \rightarrow w \in L(M') & \forall w \in \Sigma^* \\
\Leftrightarrow\ & w \notin L(M) \vee w \in L(M') & \forall w \in \Sigma^* \\
\Leftrightarrow\ & w \in L(\overline{M}) \vee w \in L(M') & \forall w \in \Sigma^* \\
\Leftrightarrow\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

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.

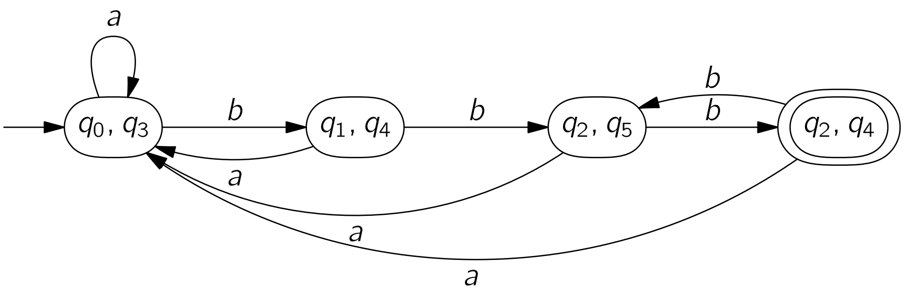
Damit ergibt sich der folgende Pseudocode:

```
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 $w \in L(M)$.