Automaten und Formale Sprachen
3. Vorlesung

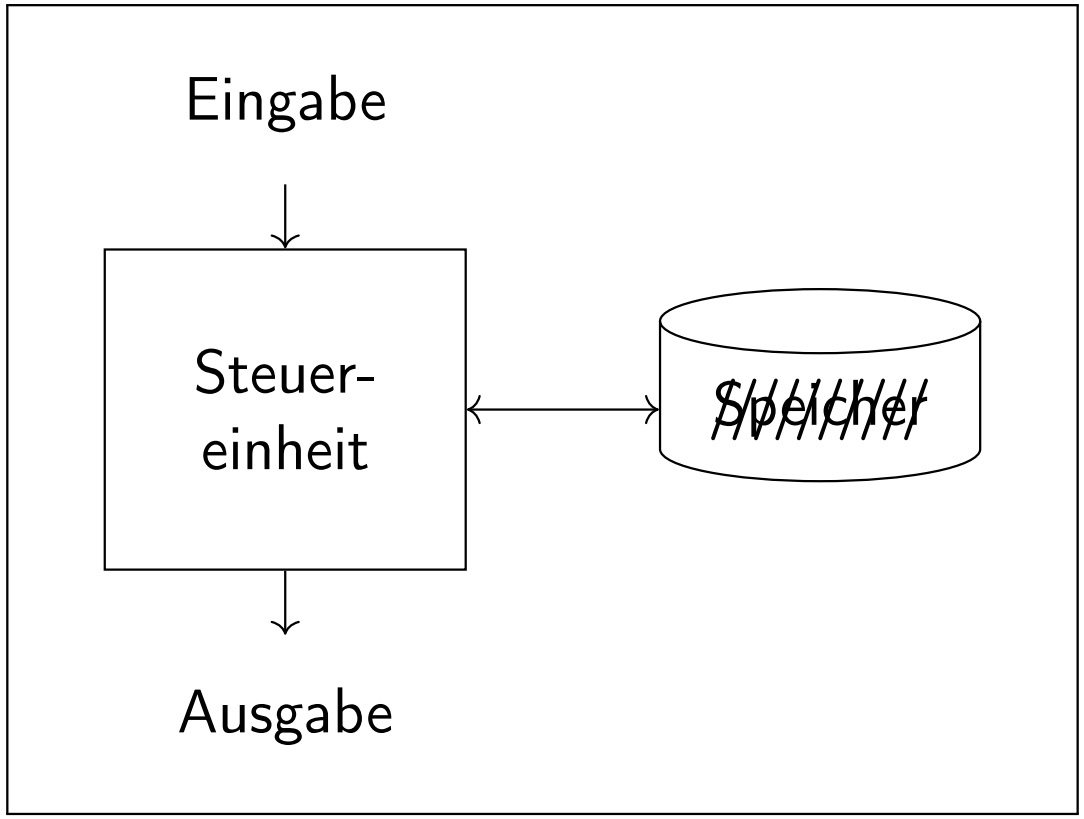
Prof. Dr. Dietrich Kuske

FG Automaten und Logik, TU Ilmenau

Wintersemester 2023/24

Maschinen: endliche Automaten

In diesem Abschnitt beschäftigen wir uns mit rechtslinearen Sprachen, aber zunächst unter einem anderen Blickwinkel.



Steuereinheit:

- endlich viele Zustände
- ändern sich in Abhängigkeit von Eingabe und Speicherkonfiguration
- produziert Ausgaben
- gibt Speicherbefehle

Eingabe: Folge von „Buchstaben“

Ausgabe: gut/schlecht

Deterministische endliche Automaten

Definition

Ein deterministischer endlicher Automat M ist ein 5-Tupel $M = (Z, \Sigma, z_0, \delta, E)$, wobei:

- Z eine endliche Menge von Zuständen ist,
- Σ das Eingabealphabet (mit $Z \cap \Sigma = \emptyset$) ist,
- $z_0 \in Z$ der Startzustand ist,
- $\delta\colon Z \times \Sigma \to Z$ die Überführungs- oder Übergangsfunktion ist und
- $E \subseteq Z$ die Menge der Endzustände ist.

Abkürzung: DFA (deterministic finite automaton)

Beispiel:

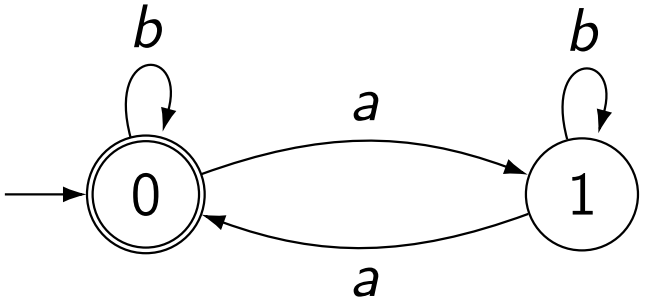
- $Z = \{0, 1\}$
- $\Sigma = \{a, b\}$
- $z_0 = 0$
- $\delta(0, a) = \delta(1, b) = 1$, $\delta(1, a) = \delta(0, b) = 0$

  tabellarisch:

	a	b
0	1	0
1	0	1

- $E = \{0\}$

Intuitiver beschreiben wir diesen DFA graphisch:



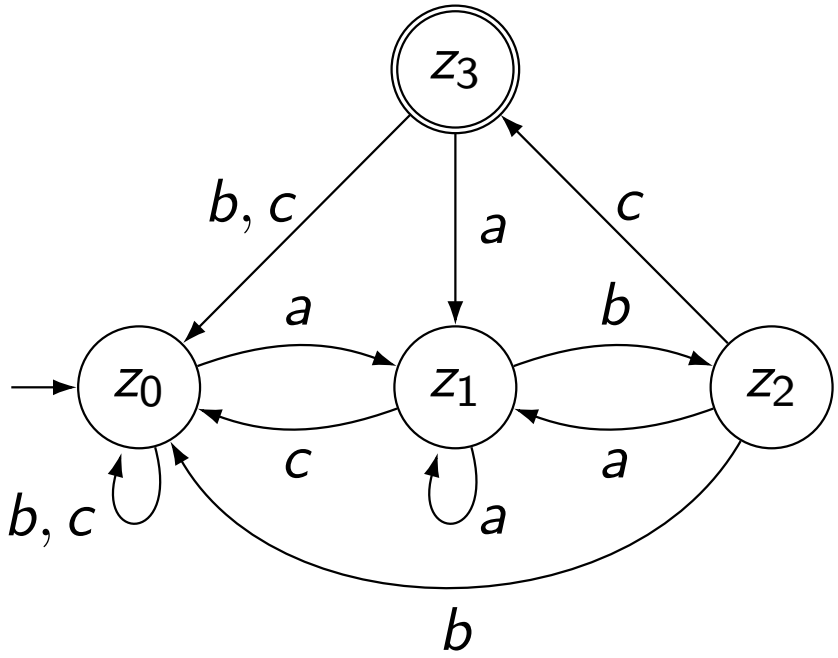
Beispiel:

- $Z = \{z_0, z_1, z_2, z_3\}$
- $\Sigma = \{a, b, c\}$
-

	a	b	c
z_0	z_1	z_0	z_0
z_1	z_1	z_2	z_0
z_2	z_1	z_0	z_3
z_3	z_1	z_0	z_0

- $E = \{z_3\}$

graphische Beschreibung:



Die bisherige Übergangsfunktion δ eines DFA liest nur ein Zeichen auf einmal ein. Wir verallgemeinern sie jetzt zu einer Übergangsfunktion $\widehat{\delta}$, die die Übergänge für ganze Wörter ermittelt.

Definition

Zu einem gegebenen DFA $M = (Z, \Sigma, z_0, \delta, E)$ definieren wir die Funktion $\widehat{\delta}\colon Z \times \Sigma^* \to Z$ induktiv wie folgt, wobei $z \in Z$, $w \in \Sigma^*$ und $a \in \Sigma$:

$$\begin{aligned} \widehat{\delta}(z, \varepsilon) &= z \\ \widehat{\delta}(z, aw) &= \widehat{\delta}\big(\delta(z, a), w\big) \end{aligned}$$

Intuition: der Zustand $\widehat{\delta}(z, w)$ ergibt sich, indem man vom Zustand z aus dem Pfad folgt, der mit w beschriftet ist.

Definition

Die von einem DFA $M = (Z, \Sigma, z_0, \delta, E)$ akzeptierte Sprache ist

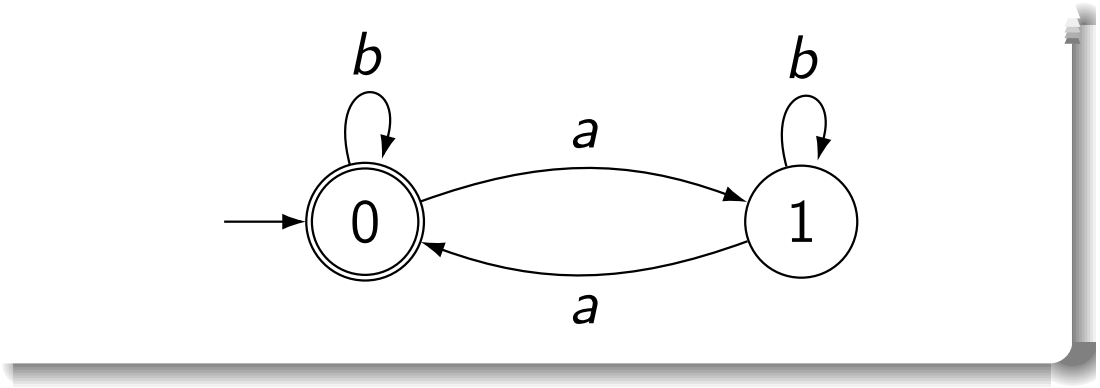
$$L(M) = \left\{ w \in \Sigma^* \mid \widehat{\delta}(z_0, w) \in E \right\}.$$

Mit anderen Worten: Ein Wort w wird genau dann akzeptiert, wenn derjenige Pfad, der im Anfangszustand beginnt und dessen Übergänge mit den Zeichen von w markiert sind, in einem Endzustand endet.

Definition

Eine Sprache $L \subseteq \Sigma^*$ ist regulär, wenn es einen DFA M mit $L(M) = L$ gibt.

Beispiel: Betrachte wieder den folgenden DFA



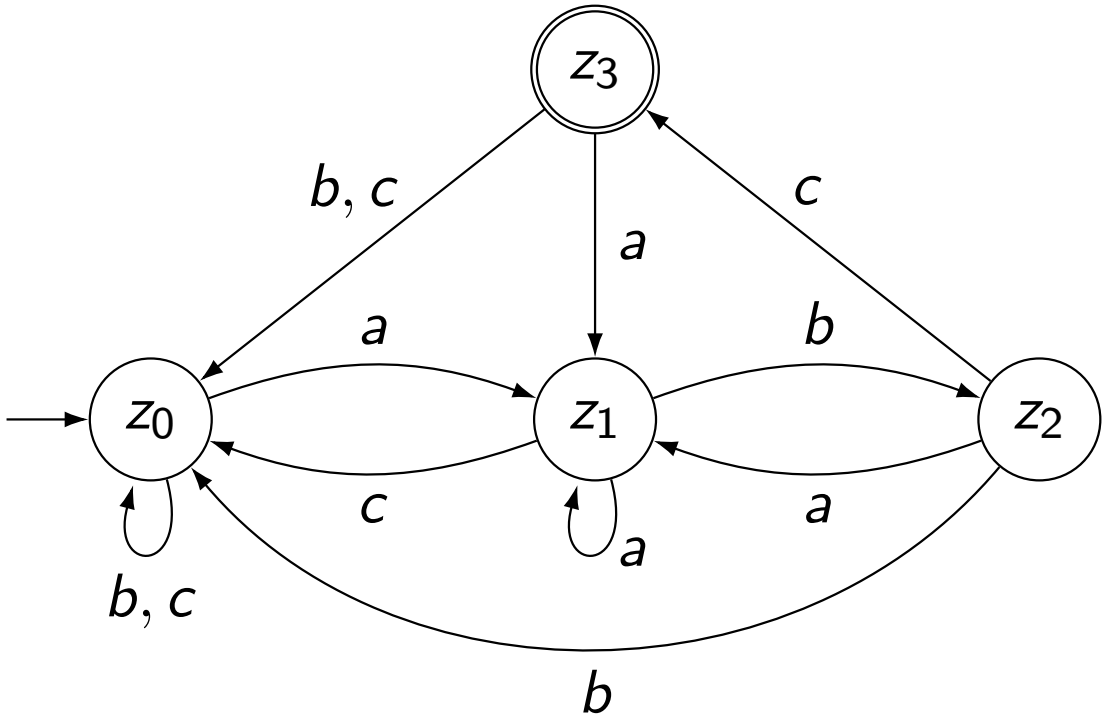
Bedeutung der Zustände:
0 – gerade Anzahl a's
1 – ungerade Anzahl a's

Für ein Wort $u \in \{a, b\}^*$ gilt

$$\widehat{\delta}(0, u) = \begin{cases} 0 & \text{falls } |u|_a \text{ gerade} \\ 1 & \text{falls } |u|_a \text{ ungerade} \end{cases}$$

Also erhalten wir $L(M) = \{u \in \Sigma^* : |u|_a \text{ ist gerade}\}$, diese Sprache ist also regulär.

Beispiel:



$$\widehat{\delta}(z_0, u) = \begin{cases} z_0 & \text{kein nichtleeres Präfix von } abc \text{ ist Suffix von } u \\ z_1 & u \text{ endet auf } a \\ z_2 & u \text{ endet auf } ab \\ z_3 & u \text{ endet auf } abc \end{cases}$$

$$\Longrightarrow \; L(M) = \{u \in \Sigma^* \mid u \text{ endet auf } abc\} = \Sigma^* \cdot \{abc\}$$

Ziel

Wir wollen zeigen, daß eine Sprache genau dann rechtslinear ist, wenn sie regulär ist. Hierfür drei Schritte:

- jede reguläre Sprache ist rechtslinear (Folie 3.12)
- jede rechtslineare Sprache wird von einem „NFA“ akzeptiert (Folie 4.2)
- jede „NFA-Sprache“ ist regulär (Folie 3.21)

Damit werden wir dann die Äquivalenz eines Erzeugungsmechanismus („rechtslineare Grammatik“) mit einem Akzeptanzmechanismus („DFA“) erhalten.

Proposition

Jede reguläre Sprache ist rechtslinear.

Beweis:

Sei $M = (Z, \Sigma, z_0, \delta, E)$ ein DFA.

Wir definieren eine Typ-3 Grammatik $G = (V, \Sigma, P, S)$ wie folgt:

$$\begin{aligned} V &= Z \\ S &= z_0 \\ P &= \left\{ z \to a\ \delta(z,a) \;\middle|\; z \in Z, a \in \Sigma \right\} \cup \\ &\quad \left\{ z \to \varepsilon \mid z \in E \right\} \end{aligned}$$

Behauptung: Für alle $z, z' \in Z$ und $w \in \Sigma^*$ gilt:

$$z \Rightarrow_G^* wz' \quad \Longleftrightarrow \quad \widehat{\delta}(z, w) = z' \,.$$

Dies zeigt man durch Induktion über $|w|$.

Für $w \in \Sigma^*$ gilt dann:

$$\begin{aligned} w \in L(G) &\iff \exists z \in V : z_0 \Rightarrow_G^* wz \Rightarrow_G w \\ &\overset{\text{Beh.}}{\iff} \exists z \in Z : \widehat{\delta}(z_0, w) = z \text{ und } (z \to \varepsilon) \in P \\ &\iff \widehat{\delta}(z_0, w) \in E \\ &\iff w \in L(M) \end{aligned}$$

Also haben wir $L(M) = L(G)$. □

Nichtdeterministische endliche Automaten

Ziel: Eine Sprache ist regulär gdw. sie rechtslinear ist.

erreicht: „$\Longrightarrow$“, wobei die Zustände zu Nichtterminalen wurden

Problem bei „$\Longleftarrow$“: DFAs sind deterministisch, Grammatiken nichtdeterministisch (es kann z.B. die Produktionen $S \to aA$ und $S \to aB$ geben)

Ausweg: erweitere DFAs um Nichtdeterminismus zu „NFAs“ und zeige:

1. jeder „NFA“ kann durch einen DFA simuliert werden (Folie 3.21)
2. jede rechtslineare Sprache kann durch einen „NFA“ akzeptiert werden (Folie 4.2)

Für eine Menge Z ist

$$\mathcal{P}(Z) = \{Y \mid Y \subseteq Z\}$$

die Potenzmenge von Z, d.h. die Menge aller Teilmengen von Z.
Diese Menge wird manchmal auch mit 2^Z bezeichnet.

Definition

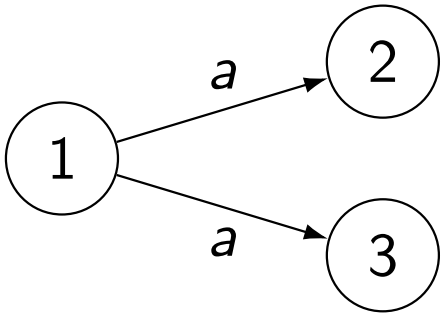
Ein nichtdeterministischer endlicher Automat M ist ein 5-Tupel $M = (Z, \Sigma, S, \delta, E)$, wobei:

- Z ist eine endliche Menge von Zuständen,
- Σ ist das Eingabealphabet (mit $Z \cap \Sigma = \emptyset$),
- $S \subseteq Z$ ist die Menge der Startzustände,
- $\delta\colon Z \times \Sigma \to \mathcal{P}(Z)$ ist die Überführungs- oder Übergangsfunktion und
- $E \subseteq Z$ ist die Menge der Endzustände.

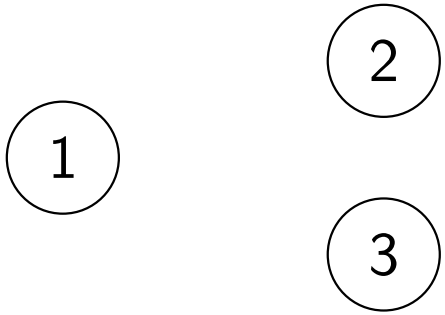
Abkürzung: NFA (nondeterministic finite automaton)

Beispiel:

$\delta(1, a) = \{2, 3\}$ bedeutet, daß der NFA aus dem Zustand 1 mit dem Buchstaben a sowohl in den Zustand 2 als auch in den Zustand 3 wechseln kann.



$\delta(1, a) = \emptyset$ bedeutet, daß der NFA aus dem Zustand 1 mit dem Buchstaben a in keinen Zustand wechseln kann, also blockiert.



Die Übergangsfunktion δ kann zu einer Mehr-Schritt-Übergangsfunktion $\widehat{\delta}\colon \mathcal{P}(Z) \times \Sigma^* \to \mathcal{P}(Z)$ erweitert werden.

Intuition: Die Zustandsmenge $\widehat{\delta}(Y, w)$ ist die Menge aller Zustände, die durch einen w-beschrifteten Pfad von einem der Zustände $z \in Y$ aus erreicht werden können.

Definition

Zu einem gegebenen NFA $M = (Z, \Sigma, S, \delta, E)$ definieren wir die Funktion

$$\widehat{\delta}\colon \mathcal{P}(Z) \times \Sigma^* \to \mathcal{P}(Z)$$

induktiv wie folgt, wobei $Y \subseteq Z$, $w \in \Sigma^*$ und $a \in \Sigma$:

$$\begin{aligned} \widehat{\delta}(Y, \varepsilon) &= Y \\ \widehat{\delta}(Y, aw) &= \widehat{\delta}\Big(\bigcup_{z \in Y} \delta(z, a),\ w \Big) \end{aligned}$$

Bemerkung

Es gilt

$$\begin{aligned}\widehat{\delta}(Y,a) &= \widehat{\delta}\Big(\bigcup_{z\in Y}\delta(z,a),\varepsilon\Big)\\ &= \bigcup_{z\in Y}\delta(z,a)\end{aligned}$$

und daher

$$\begin{aligned}\widehat{\delta}(Y,av) &= \widehat{\delta}\Big(\bigcup_{z\in Y}\delta(z,a),v\Big)\\ &= \widehat{\delta}\big(\widehat{\delta}(Y,a),v\big)\,.\end{aligned}$$

Diese Gleichung werden wir ab jetzt oft verwenden.

Definition

Die von einem NFA M akzeptierte Sprache ist

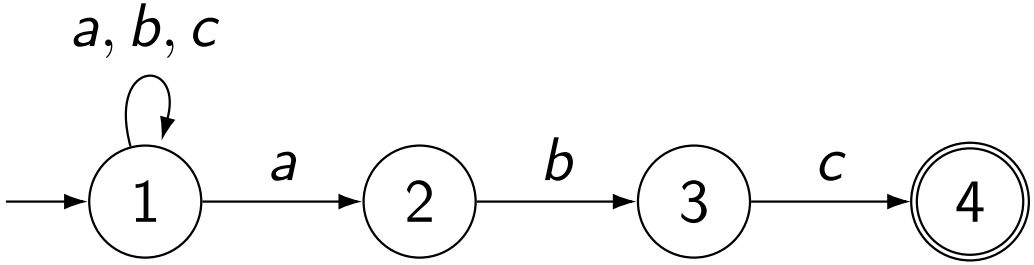
$$L(M) = \left\{ w \in \Sigma^* \;\middle|\; \widehat{\delta}(S, w) \cap E \neq \emptyset \right\}.$$

Mit anderen Worten: Ein Wort w wird genau dann akzeptiert, wenn es mindestens einen Pfad von einem Anfangszustand zu einem Endzustand gibt, dessen Übergänge mit den Zeichen von w markiert sind (daneben kann es auch Pfade geben, die nicht in einen Endzustand führen).

Beispiel: gesucht ist ein NFA, der die Sprache

$$L = \left\{ u \in \{a, b, c\}^* \;\middle|\; u \text{ endet auf } abc \right\}$$

akzeptiert.



Dieser NFA rät zu einem beliebigen Zeitpunkt nicht-deterministisch, daß jetzt das Suffix *abc* beginnt und überprüft dann, daß dies tatsächlich der Fall ist.

Proposition

Jede von einem NFA akzeptierbare Sprache ist regulär.

Beweis:

Idee: Der zu konstruierende DFA bestimmt, in welchen Zuständen sich der ursprüngliche NFA befinden könnte.

Das heißt, die Zustände des DFA sind Mengen von Zuständen des ursprünglichen NFA. Man nennt diese Konstruktion daher auch Potenzmengenkonstruktion.

Sei $M = (Z, \Sigma, S, \delta, E)$ ein NFA.

Definiere den DFA

$$\begin{aligned} M' &= \left(\mathcal{P}(Z), \Sigma, S, \gamma, F\right) \\ \text{mit } \gamma(Y, a) &= \widehat{\delta}(Y, a) \\ F &= \{Y \subseteq Z \mid Y \cap E \neq \emptyset\} \end{aligned}$$

Behauptung: Für alle $Y \subseteq Z$ und alle Wörter $w \in \Sigma^*$ gilt:

$$\widehat{\gamma}(Y, w) = \widehat{\delta}(Y, w)$$

Dies zeigt man durch Induktion über $|w|$.

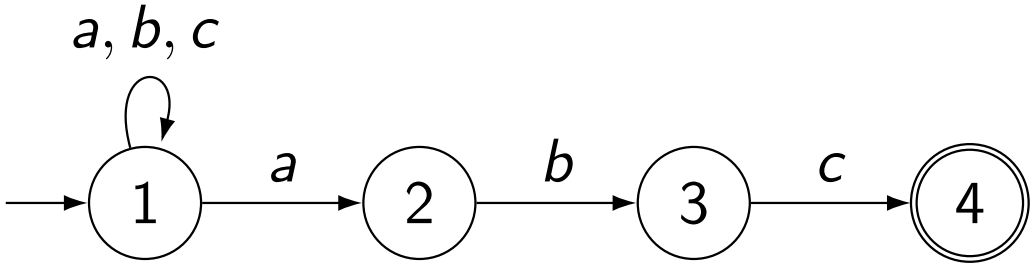
Also gilt für jedes Wort $w \in \Sigma^*$:

$$\begin{aligned} w \in L(M') &\iff \widehat{\gamma}(S, w) \in F \\ &\iff \widehat{\delta}(S, w) \cap E \neq \emptyset \\ &\iff w \in L(M) \end{aligned}$$

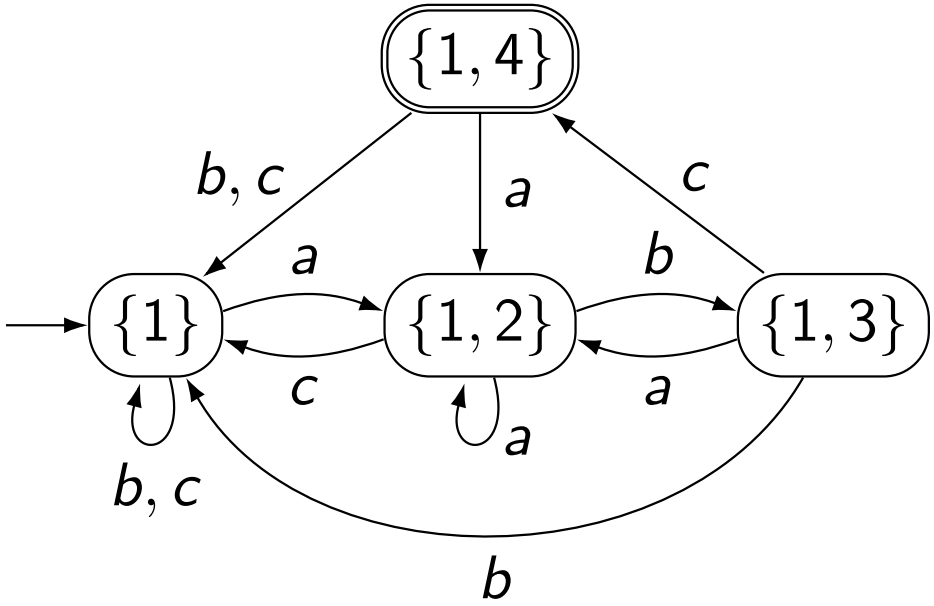
und damit $L(M') = L(M)$ □

Beispiel: (vgl. Folien 3.10 und 3.20)

NFA M:



Konstruierter DFA M' (vom Initialzustand aus erreichbarer Anteil):



Zusammenfassung 3. Vorlesung

in dieser Vorlesung neu

- Definitionen DFA, NFA und die von diesen akzeptierte Sprache
- jede reguläre Sprache (d.h. von einem DFA akzeptiert) ist rechtslinear
- jede „NFA-Sprache“ ist regulär

offen für kommende Vorlesung

Um zu zeigen, daß eine Sprache genau dann reguläre ist, wenn sie rechtslinear ist, benötigen wir noch:

- jede rechtslineare Sprache ist „NFA-Sprache“

weiterhin in nächster Vorlesung

- Abschlußeigenschaften der Klasse der rechtslinearen Sprachen, d.h. gilt z.B. K, L rechtslinear $\Longrightarrow$ $K \cdot L$ rechtslinear?