Inhaltsverzeichnis

Vorwort

Diese Aufgabensammlung in zwei Bänden über die Grundlagen der Informatik ist als Übungsbuch und Nachschlagewerk für Studierende der Informatik und verwandter Studiengänge konzipiert. Sie kann zur Prüfungsvorbereitung, vorlesungsbegleitend oder zur Wissensauffrischung genutzt werden. Der vorliegende erste Band behandelt die Grundlagen der Theoretischen Informatik, also insbesondere die Theorie der formalen Sprachen, Automaten und Grammatiken sowie die Berechenbarkeits- und Komplexitätstheorie.

Entstehung

Die Aufgabensammlung ist in den Jahren 2009 bis 2013 im Rahmen der Vorlesung „Grundlagen der Informatik II" entstanden, die als Grundlagenvorlesung für die Studiengänge Wirtschaftsingenieurwesen, Technische Volkswirtschaftslehre und Wirtschaftsmathematik am Karlsruher Institut für Technologie (KIT) üblicherweise für das dritte Semester angeboten wird. Dabei wurde dem Wunsch der Studierenden nach einer größeren Aufgabenfülle zur Prüfungsvorbereitung und nach Aufgaben, die den Stoff in verständlicher und intuitiver Weise veranschaulichen, nachgegangen. Es ist jedoch schwierig, gerade Anfängern eine intuitive Veranschaulichung der sehr formalen Inhalte zu ermöglichen, die erst über mehrere Ecken ihren praktischen Nutzen entfalten. Oft entsteht eine Kluft zwischen den Dozenten, die eine korrekte, in das Gesamtbild passende Sicht vermitteln wollen, und den Studierenden, denen dieses Gesamtbild zu Beginn noch nicht verständlich ist und für die eine schrittweise Heranführung, möglicherweise über Umwege, hilfreicher wäre. Um diesem Problem zu begegnen, wurden die Ideen für die Mehrzahl der Aufgaben und deren Lösungen nicht von Dozenten geliefert, sondern von Tutoren, also von studentischen Hilfskräften, die zur Unterstützung des Übungsbetriebs eingesetzt werden. Diese sind hauptsächlich Studierende, welche selbst die Vorlesung im Vorsemester gehört haben. So entstanden Aufgaben, die sich aus Studierendensicht mit den essentiellen Verständnisproblemen der jeweiligen Inhalte beschäftigen und auch in Studierendensprache die Lösung dieser Probleme darstellen. Die Aufgaben wurden teilweise in den vergangenen zwei Vorlesungszyklen den Studierenden zur Prüfungsvorbereitung angeboten und stießen dabei auf sehr gute Resonanz. Natürlich entziehen sich die Autoren dennoch nicht der Verantwortung, ausgewogene, hochwertige und fehlerfreie Aufgaben in diesem Buch zur Verfügung zu stellen. Die Aufgaben, die jetzt den Hauptteil dieser Aufgabensammlung bilden, wurden aus einem noch größeren Pool studentischer Aufgaben ausgewählt, sie wurden einzeln mehrfach überarbeitet, korrigiert, ergänzt und in ein einheitliches Format gebracht.

Struktur

Jeder Band der Aufgabensammlung ist in zwei Hauptteile gegliedert, von denen der erste eine Zusammenfassung der theoretischen Grundlagen und die Aufgabenstellungen enthält und der zweite Lösungen zu den Aufgaben. Der erste Hauptteil ist inhaltlich nach Kapiteln gegliedert, die jeweils mit einer Einführung in das Thema und einer Erläuterung aller für das Lösen der

Aufgaben nötigen Voraussetzungen beginnen. In diesem Rahmen wurde auch versucht, auf typische Probleme und Missverständnisse einzugehen, die Studierende erfahrungsgemäß häufig mit den jeweiligen Inhalten haben. Im Anschluss an die Einführung folgt in jedem Kapitel der Aufgabenteil, der Aufgabenstellungen unterschiedlicher Schwierigkeitsgrade enthält. Die Schwierigkeit einer Aufgabe wird zu deren Beginn durch die Symbole

- $\star$ leicht,
- $\star^{\star}$ mittel oder
- $\star^{\star^{\star}}$ schwer

gekennzeichnet. Einige wenige Aufgaben sind durch das Symbol $\star^{\star^{\star^{\star}}}$ als sehr schwer bezeichnet; die Lösung dieser Aufgaben kann unter Umständen sehr viel Zeit beanspruchen und sollte nicht unterschätzt werden. Zu jeder Aufgabe gibt es im zweiten Hauptteil einen Lösungsvorschlag mit ausführlichen Erklärungen und Rechenwegen. Allgemeine Grundlagen werden ganz am Ende jedes Bandes im Anhang behandelt.

Vorlesungsaufzeichnungen

Die Vorlesung „Grundlagen der Informatik II“, auf deren Inhalten die Aufgabensammlung beruht, wird seit Jahren aufgezeichnet, und die Videos sind für den öffentlichen Zugriff freigeschaltet. Im Rahmen der Arbeit an diesem Buch wurden die Videos neu zusammengeschnitten, sodass sie thematisch zu den Kapiteln passen. Am Ende der Einführung jedes Kapitels ist ein Link abgedruckt, unter dem man zu dem entsprechenden Thema das Vorlesungsvideo abrufen kann. Das Video kann sowohl im Browser angesehen als auch im AVI-Format heruntergeladen werden.

Diskussionsforum

Im Rahmen der Forschung im Bereich der Verbesserung der Lehre in universitären Großveranstaltungen wurde in den Vorlesungszyklen der Wintersemester 2011/12 und 2012/13 der Vorlesung „Grundlagen der Informatik II“ ein Konzept entwickelt, das die Diskussion von Lösungen und Problemen einzelner Aufgaben zwischen Studierenden und Dozenten ermöglicht [Pfe+12]. Im Zentrum steht ein Forum, das für jede Aufgabe einen Thread enthält, in dem die Aufgabe diskutiert werden kann. Dieses Forum wird für die Buchversion der Aufgabensammlung fortgeführt und kann unter der folgenden Adresse abgerufen werden:

`http://www.dasinfobuch.de/links/Forum.html`

Um eine bestimmte Aufgabe im Forum wiederzufinden, ist jeder Aufgabe eine fünfstellige ID vorangestellt, beispielsweise „END-AA“. Die Threads des Forums sind mit genau diesen IDs betitelt, die Diskussion zur Aufgabe END-AA findet also im Thread END-AA des Forums statt.

Im Forum können die in der Vergangenheit bereits in über 2000 Beiträgen diskutierten Themen eingesehen oder neue eigene Beiträge verfasst werden. Selbstverständlich werden auch zukünftig die Dozenten im Forum weiter mitdiskutieren und die Probleme und Lösungen kommentieren.

Zusammen mit Vorlesungsaufzeichnungen und Diskussionsforum stellt diese Aufgabensammlung ein interaktives Gesamtkonzept dar, mit dem die Autoren hoffen, der Leserin und dem Leser auf eine ansprechende und verständliche Art die Grundlagen der Informatik nahebringen zu können. In diesem Sinne wünschen sie ihnen eine angenehme Lektüre und viel Erfolg beim Lernen und beim Lösen der Aufgaben.

Danksagung

Die Autoren bedanken sich für die kreative Mitwirkung an den Aufgaben bei Jonas Benterbusch, Philippe Blättchen, Marcel Braith, Leonard Brauck, Johannes Burchardt, Jose Antonio Cepeda, Vivian Cheng, Jacob Eberhardt, Theresa Euring, Yvonne Freytag , Philip Grefe, Anna-Lena Hau, Christiane Haubitz, Matthias Hauser, Stefanie Häuser, Michael Heck, Jonas Hemlein, Patrick Henkes, Yannick Hilgers, Lisa Hoffmann, Julian Holz, Lukas Jansen, Maximilian Jentsch, Fabian Jost, Claus Kadelka, Tobias Käfer, Nico Kaltenbacher, Fabian Kerstholt, Christian Kiefer, Isabelle Krämer, Martin Kreuser, Daniel Kübler, Melanie Lampert, Tobias Lurz, Florian Mägerlein, Christina Marx, Bastian Prell, Jördis Rappl, Adam Rodacki, Benedict Schendzielorz, Sebastian Schienle, Irena Schips, Sven Schömer, Marcel Schrumpf, Anja Schuler, Alexander Sigmund, Birgit Spitz, Eva Steffes, Tobias Stengel, Michael Vormittag, Simon Wiedemann, Lea Wild und Kai Windscheid.

Für Arbeiten an den Vorlesungsaufzeichnungen und der Verlinkung des Forums danken sie Solveig Gronau und Caroline Strunz.

Für das Durchsehen der Texte hinsichtlich Grammatik und Stil danken sie Christian Gitte, Christian Hirsch, Sebastian Kochanneck, Ingo Mauser, Sabrina Merkel, Marc Mültin, Daniel Pathmaperuma, Fabian Rigoll, Felix Vogel und Micaela Wünsche.

Karlsruhe, September 2013

Lukas König, Friederike Pfeiffer-Bohnen, Hartmut Schmeck

1 Endliche Automaten mit Ausgabe

Einführung

Endliche Automaten stellen ein einfaches Berechnungsmodell dar, bei dem im Laufe einer Berechnung nur endlich viele verschiedene Zustände zur Verfügung stehen, die in einer vordefinierten Weise abhängig von der Eingabe durchlaufen werden. Im Gegensatz zu mächtigeren Automatenmodellen wie Kellerautomaten und Turingmaschinen haben endliche Automaten keinen Zugriff auf ein unendliches beschreibbares Band, das für die Speicherung des Zustands zusätzlich genutzt werden kann. Endliche Automaten lesen eine aus Zeichen über einem gegebenen Eingabealphabet bestehende Eingabe sequentiell ein, wobei sie nach jedem Zeichen in einen Folgezustand wechseln. Endliche Automaten mit Ausgabe erzeugen dabei entweder bei jedem Besuch eines Zustands (**Moore-Automaten**) oder bei jedem Wechsel von einem Zustand zu einem anderen (**Mealy-Automaten**) ein Zeichen aus einem Ausgabealphabet als Ausgabe. Moore- und Mealy-Automaten sind in ihrer Berechnungsmächtigkeit äquivalent und jeweils durch einen Algorithmus ineinander überführbar.

Definition

Formal sind endliche Automaten mit Ausgabe 6-Tupel $(E, S, A, \delta, \gamma, s_0)$, wobei E das **Eingabealphabet** ist, S die **Zustandsmenge**, A das **Ausgabealphabet**, δ die **Zustandsüberführungsfunktion** oder **Zustandsübergangsfunktion**, γ die **Ausgabefunktion** und $s_0 \in S$ der **Anfangszustand**. Die Zustandsüberführungsfunktion

$$\delta : S \times E \to S$$

definiert, in welchen Folgezustand ein endlicher Automat nach dem Lesen eines Zeichens wechselt. Die Ausgabefunktion

$$\gamma : S \to A \text{ (Moore)}$$

bzw.

$$\gamma : S \times E \to A \text{ (Mealy)}$$

definiert, welche Ausgabe der Automat in einem Zustand (Moore) bzw. bei einem Zustandswechsel (Mealy) auf das Ausgabeband schreibt. Die von einem Moore- oder Mealy-Automaten **berechnete Funktion** $f : E^\star \to A^\star$ ist für ein Eingabewort $w \in E^\star$ definiert als das Ausgabewort $f(w) =_{def} w' \in A^\star$, das der Automat beim Einlesen von w als Ausgabe erzeugt.

Darstellung

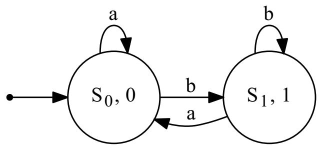
Die Zustandsüberführungs- und Ausgabefunktionen endlicher Automaten werden üblicherweise als **gerichtete Graphen** dargestellt, wobei jeder Zustand $s \in S$ als Knoten und jede Überführung vom Zustand s zu einem anderen Zustand $s' \in S$ als Kante vom Knoten s zum Knoten

s' gezeichnet wird. Ein Eingabezeichen, das zum Wechsel von Zustand s zu Zustand s' führt, wird als Kantenbeschriftung zur Kante von s nach s' hinzugefügt. Ausgabezeichen $a \in A$ werden entsprechend bei Moore-Automaten zusätzlich zur Beschriftung des Zustands hinzugefügt und bei Mealy-Automaten zur Beschriftung der Kanten. Der Anfangszustand s_0 wird mit einem eingehenden Pfeil gekennzeichnet. Zu einer vollständigen Darstellung eines endlichen Automaten mit Ausgabe gehören neben einer solchen Abbildung auch immer eine Definition der Alphabete und der Zustandsmenge sowie des Anfangszustands.

Beispielsweise stellt die folgende Abbildung einen Moore-Automaten A_{moore} dar, der im Zustand s_0 startet und dort bleibt, solange a eingelesen wird; wenn in der Eingabe ein b gelesen wird, wird in den Zustand s_1 gewechselt und bei weiteren b's in s_1 geblieben, bis durch a wieder ein Zustandswechsel erfolgt. Dabei gibt der Automat im Zustand s_0 bei jedem gelesenen Eingabezeichen das Ausgabezeichen 0 aus und im Zustand s_1 das Ausgabezeichen 1. Der Automat berechnet also eine Funktion, durch die jedes a in der Eingabe durch 0 und jedes b durch 1 ersetzt wird.

$$A_{moore} = (\{a, b\}, \{s_0, s_1\}, \{0, 1\}, \delta, \gamma_1, s_0)$$

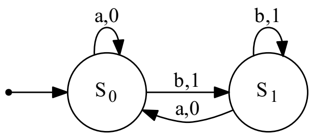
δ, γ_1:



Der folgende Mealy-Automat A_{mealy} zeigt dasselbe Verhalten und berechnet dieselbe Funktion, nur dass die Ausgabe hier an den Zustandsübergängen erfolgt.

$$A_{mealy} = (\{a, b\}, \{s_0, s_1\}, \{0, 1\}, \delta, \gamma_2, s_0)$$

δ, γ_2:



Endliche Automaten mit und ohne Ausgabe (vgl. Kapitel 2) finden, neben ihrer Bedeutung für die Theoretische Informatik, auch praktische Anwendung, beispielsweise im Hardware-Design [Ric73], zur Steuerung von Maschinen und Robotern [Jun+12; KMS09] oder im Softwareentwicklungsprozess.

Themen dieses Kapitels

Dieses Kapitel beschäftigt sich mit deterministischen endlichen Automaten mit Ausgabe, also mit Moore- bzw. Mealy-Automaten, auch Moore- bzw. Mealy-Maschinen genannt. Die fünf Aufgaben dieses Kapitels behandeln die Erzeugung von Moore- bzw. Mealy-Automaten für bestimmte Berechnungsaufgaben, das Ablesen der von einem gegebenen Automaten berechneten Funktion und die Umwandlung von Mealy-Automaten in Moore-Automaten. Die benutzten

Verfahren können im Buch von Hopcroft, Motwani und Ullman [HMU11] nachgelesen werden. Weiterführende Informationen gibt es auch in den Büchern von Ingo Wegener [Weg05] und Uwe Schöning [Sch08]. Die Umwandlung eines Moore- in einen Mealy-Automaten wird nicht behandelt, da sich der Automat dabei fast nicht verändert und nur die Ausgabe, die beim Moore-Automaten an einem Zustand erfolgt, beim Mealy-Automaten zu all den eingehenden Übergängen des entsprechenden Zustands zugewiesen wird. Darüber hinaus werden als Anwendungsbeispiele die Interpretation von Zahlen in BCD-Darstellung und die binäre Subtraktion (vgl. Band 2, Kapitel 5) durch endliche Automaten mit Ausgabe behandelt.

Dieses Thema in der Vorlesungsaufzeichnung zur Vorlesung „Grundlagen der Informatik II“:

`http://www.dasinfobuch.de/links/Endliche-Automaten-mit-Ausgabe.html`

Aufgaben

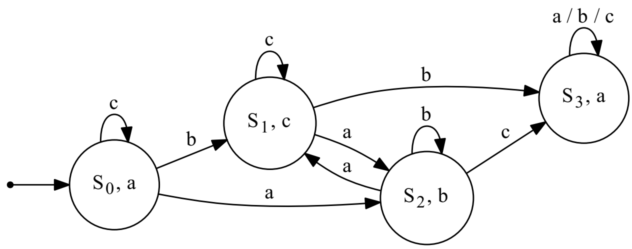
Aufgabe 1	**END-AU**

Umwandlung von Moore-Automaten in Mealy-Automaten ⋆

Gegeben sei folgender Moore-Automat M.

$$M = (\{a, b, c\}, \{s_0, s_1, s_2, s_3\}, \{a, b, c\}, \delta, \gamma, s_0)$$

δ, γ:



Wandeln Sie M in einen äquivalenten Mealy-Automaten M' um und geben Sie diesen vollständig an.

→ *Lösung: S. 91*

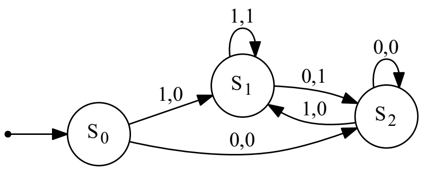
Aufgabe 2	**END-AT**

Mealy- und Moore-Automaten ⋆⋆

Gegeben sei der folgende Mealy-Automat $M = (E, S, A, \delta, \gamma, s_0)$.

$$M = (\{0, 1\}, \{s_0, s_1, s_2\}, \{0, 1\}, \delta, \gamma, s_0)$$

δ, γ:



a) Welche Funktion $f : E^\star \to A^\star$ berechnet M?

b) Was ist der Unterschied zwischen Mealy- und Moore-Automaten?

c) Geben Sie zu dem Mealy-Automaten M den zugehörigen Moore-Automaten M' an. Definieren Sie den M' vollständig.

d) Warum kommen Mealy-Automaten gegenüber Moore-Automaten bei gleicher Funktionsweise mit weniger Zuständen aus?

→ *Lösung: S. 91*

Aufgabe 3 END-AA

Mealy-Automaten ★*

Entwerfen Sie einen Mealy-Automaten M zur bitseriellen Subtraktion zweier gleichlanger Zahlen. Geben Sie M vollständig an.

→ *Lösung: S. 92*

Aufgabe 4 END-AR

Mealy-Automaten ★*

Zur Darstellung von Zahlen kann die BCD-Kodierung verwendet werden (vgl. Band 2, Kapitel 5). Diese Blockkodierung ist eine Tetradenkodierung, wobei jede Ziffer einer Dezimalzahl durch vier Bits mit der Stellenwertigkeit $2^3 = 8, 2^2 = 4, 2^1 = 2$ und $2^0 = 1$ dargestellt wird. Der BCD-Code 0110 1001 0000 0100 entspricht beispielsweise der Dezimalzahl 6904.

Definieren Sie einen Mealy-Automaten M, der bei Eingabe einer BCD kodierten Zahl die entsprechende Dezimalzahl ausgibt. Geben Sie M vollständig an.

Hinweis: Gehen Sie davon aus, dass es sich bei allen eingegebenen Binärzahlen um BCD-Kodierungen handelt, eine Überprüfung dessen also nicht notwendig ist. Der andernfalls notwendige Auffangzustand wird als implizit vorhanden angenommen.

→ *Lösung: S. 93*

Aufgabe 5 **END-AS**

Mealy-Automaten

$\star^{\star}$

Auf den meisten Golfplätzen findet man im Übungsbereich eine Driving Range, auf der lange Schläge geübt werden können. Die einzelnen Abschlagplätze sind nebeneinander aufgereiht und die Trainierenden schlagen aus Sicherheitsgründen alle in die gleiche Richtung. Die Golfbälle zum Trainieren kann man sich an speziellen Ballautomaten ausgeben lassen.

Konstruieren Sie den im folgenden Abschnitt beschriebenen Ballautomaten als Mealy-Automat $M = (E, S, A, \delta, \gamma, s_0)$. Geben Sie M vollständig an.

> Der zu modellierende Automat nimmt ausschließlich 1 Euro und 2 Euro Münzen an, alle anderen Münzen sind ungültig und werden zurückgegeben. Maximal wird vom Automaten eine Gesamteingabe von 3 Euro akzeptiert. Sofern mit einer weiteren eingeworfenen Münze dieser Betrag überschritten wird, wird die betreffende Münzen direkt zurückgegeben. Über eine Rückgabetaste kann der Kunde sich zudem das eingezahlte Geld auch direkt zurückgeben lassen, sollte er sich doch keine Golfbälle ausgeben lassen wollen. Kunden können zwischen drei verschiedenen Ballpaketen wählen und für jede dieser Optionen gibt es am Automaten eine Taste:
>
> 1) Die Ausgabe von 10 Golfbällen kostet 1 Euro,
> 2) 25 Golfbälle gibt es für 2 Euro und
> 3) 50 Golfbälle kosten 3 Euro.
>
> Bei einer Eingabe von beispielsweise 3 Euro kann der Kunde sich direkt 50 Golfbälle ausgeben lassen, es ist jedoch auch möglich, sich erst 25 Golfbälle ausgeben zu lassen und sich den Restbetrag von 1 Euro auszahlen zu lassen oder mit diesem Restbetrag 10 weitere Golfbälle zu ordern.

→ *Lösung: S. 94*

2 Endliche Automaten ohne Ausgabe

Einführung

Im Gegensatz zu endlichen Automaten mit Ausgabe, die dazu dienen, Funktionen zu berechnen, haben endliche Automaten ohne Ausgabe (im Folgenden nur „endliche Automaten“) eine Klassifizierungsfunktionalität. Formal berechnen sie Funktionen aus der Menge aller Wörter über einem Alphabet E in die Menge $\{true, false\}$, sortieren also jedes einzelne dieser Wörter in eine von zwei „Schubladen“. Dabei definieren sie eine Sprache, die durch die Menge der Wörter gegeben ist, die in der Schublade *true* landen. Diese wird als (akzeptierte) **Sprache** $L(A)$ eines endlichen Automaten A bezeichnet. Die Klassifizierung erfolgt, indem ein Wort $w = e_1 \ldots e_n \in E^\star$ Zeichen für Zeichen durch den Automaten eingelesen wird; befindet er sich nach Einlesen des letzten Zeichens e_n in einem „Endzustand“ (was das genau heißt, wird unten definiert), gilt das Wort als **akzeptiert** (*true*), sonst nicht ($false$).

Definition

Endliche Automaten werden als 5-Tupel (E, S, δ, s_0, F) definiert, wobei E das Eingabealphabet ist, S die Zustandsmenge, δ die Zustandsüberführungsfunktion, $s_0 \in S$ der Anfangszustand und $F \subseteq S$ die Menge der Endzustände. Die Zustandsüberführungsfunktion δ definiert, in welche Folgezustände ein endlicher Automat nach dem Lesen eines Zeichens wechselt. Wir unterscheiden dabei zwischen deterministischen und nichtdeterministischen endlichen Automaten. **Deterministische endliche Automaten** haben für jeden Zustand $s \in S$ und jedes eingelesene Eingabezeichen $e \in E$ **genau einen definierten Folgezustand** $s' \in S$. Daher wird die Zustandsüberführungsfunktion hier als

$$\delta : S \times E \to S \,:\, (s, e) \mapsto s'$$

definiert. Der Automat akzeptiert das Wort $w \in E^\star$ genau dann, wenn er sich nach Einlesen von w in einem Endzustand $f \in F$ befindet. **Nichtdeterministische endliche Automaten** können dagegen keinen, einen oder mehrere Folgezustände haben, also eine **Menge von Folgezuständen** $S' \subseteq S$, die sich für denselben Zustand $s \in S$ und dasselbe Eingabezeichen $e \in E$ ergibt. Daher definieren wir hier die Zustandsüberführungsfunktion als

$$\delta : S \times E \to \wp(S) : (s, e) \mapsto S'$$

Der Automat akzeptiert das Wort w genau dann, wenn er sich nach Einlesen von w in einer Zustandsmenge $F' \subseteq S$ befindet, die einen Endzustand enthält, also falls $F' \cap F \neq \emptyset$ gilt.

Zwei endliche Automaten A und A' heißen genau dann **äquivalent**, wenn ihre Sprachen gleich sind, also $L(A) = L(A')$. Die Klassen der von deterministischen bzw. nichtdeterministischen endlichen Automaten akzeptierten Sprachen heißen L_{EA} bzw. $L_{ndet-EA}$; es gilt $L_{EA} = L_{ndet-EA}$.

Nichtdeterminismus und die Potenzmengenkonstruktion

Nichtdeterministische Automaten stellen ein mathematisches Konstrukt dar, das man in der physikalischen Welt nicht bauen kann, sondern das nur von theoretischem Interesse ist. Man kann sich Nichtdeterminismus etwas vereinfacht vorstellen, als würde der Automat unter mehreren möglichen Rechenwegen immer den richtigen raten oder als verfolgte er an jeder nichtdeterministischen Abzweigung alle möglichen Wege auf einmal. Trotz dieses mächtigen Ratemechanismus, der nichtdeterministischen endlichen Automaten zur Verfügung steht, unterscheiden sich deterministische und nichtdeterministische endliche Automaten nicht in ihrer **Sprachmächtigkeit**. Beide liegen im Bereich des Chomsky-Typs 3, können also genau die Sprachen der Sprachklasse L_3 akzeptieren, die durch rechts- oder linkslineare Grammatiken erzeugt oder durch reguläre Ausdrücke definiert werden können (vgl. Kapitel 4).

Während deterministische endliche Automaten definitionsgemäß ein Spezialfall der nichtdeterministischen endlichen Automaten sind, können nichtdeterministische endliche Automaten durch eine **Potenzmengenkonstruktion** in äquivalente deterministische endliche Automaten umgewandelt werden [HMU11]. Dabei wird für jede mögliche **Teilmenge von Zuständen** $S' \subseteq S$, die im nichtdeterministischen Ausgangsautomaten über irgendein Wort $w \in E^\star$ **erreichbar** ist, ein Zustand $s_{[S']}$ im neuen deterministischen Automaten angelegt. Eine Überführung von einem Zustand $s_{[S']}$ zu einem Zustand $s_{[S'']}$ über ein Eingabezeichen $e \in E$ existiert im neuen Automaten genau dann, wenn von einem der Zustände $s' \in S'$ eine Überführung über e zu einem der Zustände $s'' \in S''$ im Ausgangsautomaten existiert. Endzustände sind alle Zustände $s_{[F']}$, für die $F' \cap F \neq \emptyset$ gilt. Dieses Verfahren ist für alle nichtdeterministischen endlichen Automaten anwendbar, allerdings kann die Zustandszahl dabei exponentiell ansteigen. Ein Beispiel für die Potenzmengenkostruktion eines deterministischen aus einem nichtdeterministischen endlichen Automaten wird unten gezeigt.

Darstellung

Wie bei endlichen Automaten mit Ausgabe wird auch bei endlichen Automaten ohne Ausgabe die Zustandsüberführungsfunktion als **gerichteter Graph** dargestellt, dessen Knoten die Zustände und dessen Kanten die Zustandsübergänge symbolisieren. Anstelle einer Darstellung der Ausgabefunktion müssen hier jedoch die Endzustände von den Nicht-Endzuständen unterschieden werden. Dafür werden Endzustände durch einen Doppelkreis und Nicht-Endzustände durch einen Einfachkreis gekennzeichnet. Zu einer vollständigen Darstellung eines endlichen Automaten gehören neben einer solchen Abbildung auch immer eine Definition des Eingabealphabets und der Zustandsmenge sowie des Anfangszustands und der Menge der Endzustände. Beispielsweise stellt die folgende Abbildung einen deterministischen endlichen Automaten A dar, der alle Wörter über dem Alphabet $\{a, b\}$ akzeptiert, die auf b enden.

$$A = (\{a, b\}, \{s_0, s_1\}, \delta, s_0, \{s_1\})$$

δ: