1 Endliche Automatenmodelle

Wir betrachten im Folgenden Automaten für endliche und unendliche Wörter.

1.1 Endliche Automaten

Im Folgenden wiederholen wir Aspekte des endlichen Automaten, der in der Vorlesung FGI-1 behandelten wurde.

Definition 1.1 *Ein* endlicher nichtdeterministischer Automat

$$A = (Q, \Sigma, \delta, Q_0, F)$$

besteht aus den folgenden Komponenten:

- Q *ist eine endliche Menge von Zuständen.*
- Σ *ist das Eingabealphabet.*
- $\delta \subseteq (Q \times \Sigma \times Q)$ *ist die Übergangsrelation.*
- $Q_0 \subseteq Q$ *ist die Menge der Startzustände.*
- $F \subseteq Q$ *ist die Menge der Endzustände.*

Beispiel 1.2 Betrachten wir den NFA in Abb. 1.1, der einen Getränkeautomat modelliert. Nach Einwurf einer Euro-Münze je nach Wahl von Tee oder Kaffee wird das entsprechende Getränk ausgeben. Es ist $Q = \{s_0, \cdots, s_5\}$, $Q_0 = F = \{s_0\}$ und

$$\Sigma = \{1€, \text{Tee}, \text{Kaffee}, \text{Tee\_kochen}, \text{Kaffee\_kochen}, \text{Tee\_einfüllen}, \text{Kaffee\_einfüllen}\}.$$

Statt $(p, a, q) \in \delta$ wird meist die intuitivere Schreibweise $p \xrightarrow{a} q$ benutzt.
Die Notation $p \xrightarrow{a} q$ wird wie folgt zu $p \xrightarrow{w} q$ auf Wörter $w \in \Sigma^*$ erweitert (Sei $a \in \Sigma$, $w \in \Sigma^*$ und $p, q, r \in Q$.):

- $p \xrightarrow{\epsilon} p$ für alle $p \in Q$.
- $p \xrightarrow{wa} r :\iff \exists q \in Q : (p \xrightarrow{w} q) \wedge (q \xrightarrow{a} r)$.

Die *akzeptierte Sprache* eines NFA A ist definiert als:

$$L(A) := \{w \in \Sigma^* \mid \exists p \in Q_0, q \in F : p \xrightarrow{w} q\}$$

Zwei NFA A_1 und A_2 heißen *äquivalent* falls $L(A_1) = L(A_2)$ gilt.
Die Familie aller von NFA akzeptierten Sprachen ist die Familie der *Regulären Mengen*.
Jeder NFA kann durch Äquivalenzumformungen vervollständigt werden, so daß zu jeder Eingabe ein Folgezustand definiert ist, d.h. $\delta^*(q, w)$ ist für alle $q \in Q$ und alle $w \in \Sigma^*$ definiert.

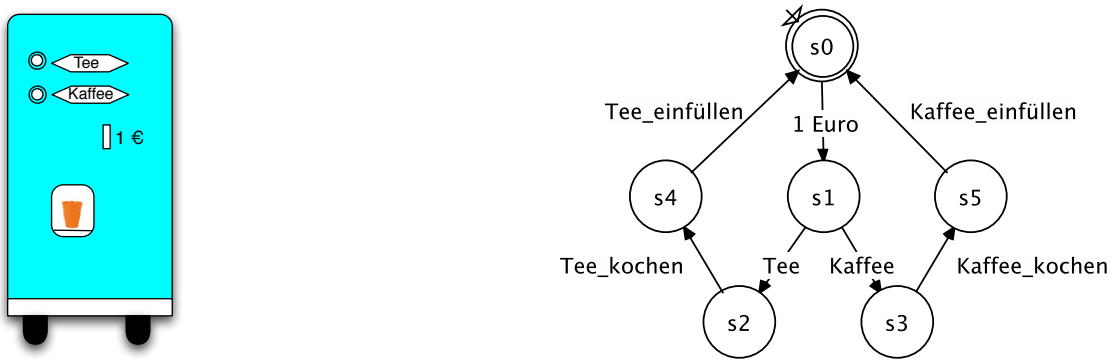


Abbildung 1.1: Getränkeautomat mit abstrakten Transitionsbezeichnern

1.1.1 Deterministische Endliche Automaten

Ein NFA A heißt deterministisch (DFA), wenn er genau einen Startzustand besitzt und die Relation δ funktional in den ersten beiden Argumenten ist, d.h. wenn $(p, a, q_1), (p, a, q_2) \in \delta$, dann gilt $q_1 = q_2$. Die Relation δ kann daher auch dann als Funktion $\delta : Q \times \Sigma \to Q$ aufgefasst werden, wobei δ i.a. keine totale Funktion ist.

Satz 1.3 *Deterministische Endliche Automaten sind äquivalent zu nichtdeterministischen.*

Beweis: Potenzautomatenkonstruktion. □

1.1.2 Abschlusseigenschaften

Satz 1.4 *Seien L_1 und L_2 reguläre Mengen, dann gilt:*

- *Die Vereinigung $L_1 \cup L_2$ ist eine reguläre Menge.*
- *Der Schnitt $L_1 \cap L_2$ ist eine reguläre Menge.*
- *Das Komplement $\Sigma^* \setminus L_1$ ist eine reguläre Menge.*
- *Der Kleene-Abschluss L_1^* ist eine reguläre Menge.*

1.1.3 Homomorphismen

Seien Σ und Γ zwei Alphabete. Ein Homomorphismus ist eine Abbildung $h : \Sigma^* \to \Gamma^*$ zwischen den Wortmengen, die die Monoidstruktur erhält, d.h. für alle $u, v \in \Sigma^*$ gilt:

$$h(u \cdot v) = h(u) \cdot h(v)$$

Also ist jeder Homomorphismus $h : \Sigma^* \to \Gamma^*$ eindeutig festgelegt, wenn wir $h(a)$ für alle $a \in \Sigma$ kennen:

$$h(w) = h(a_1 \cdots a_n) = h(a_1) \cdots h(a_n)$$

Satz 1.5 *Für jede reguläre Menge $R \subseteq \Sigma^*$ und jeden Homomorphismus $h : \Sigma^* \to \Gamma^*$ ist auch das homomorphe Bild $h(R)$ regulär.*

$$h(R) := \{h(w) \mid w \in R\}$$

Beweis: Zu der regulären Menge $R \subseteq \Sigma^*$ existiert ein akzeptierender DFA A. Ersetzen wir in A jede Kante $p \xrightarrow{a} q$ durch das Bild $p \xrightarrow{h(a)} q$, so erhalten wir einen verallgemeinerten FA, der $h(R)$ akzeptiert. □

1.1.4 Reguläre Ausdrücke

Definition 1.6 *Die regulären Ausdrücke über einem endlichen Alphabet Σ sind definiert durch:*

1. *$\emptyset$ ist ein regulärer Ausdruck, der die (leere) Menge $M_\emptyset := \emptyset$ beschreibt.*
2. *Für jedes $a \in \Sigma$ ist a ein regulärer Ausdruck, der die Menge $M_a := \{a\}$ beschreibt.*
3. *Sind A und B reguläre Ausdrücke, welche die Mengen M_A und M_B beschreiben, dann sind induktiv folgende reguläre Ausdrücke definiert:*
   - *$(A + B)$ beschreibt die Menge $M_A \cup M_B$,*
   - *$(A \cdot B)$ beschreibt die Menge $M_A \cdot M_B$,*
   - *A^* beschreibt die Menge M_A^*.*
4. *Nur die mit 1., 2. und 3. erzeugten Ausdrücke sind reguläre Ausdrücke.*

Oftmals nimmt man noch A^+ als Ausdruck hinzu. Dieser beschreibt die Menge M_A^+. Der Ausdruck A^+ kann auch als Makro für AA^* aufgefasst werden.
Die durch reguläre Ausdrücke A beschriebenen Mengen M_A sind genau die *reguläre Mengen* oder Sprachen über Σ.

Satz 1.7 *Die Menge der durch endliche Automaten $A = (Q, \Sigma, \delta, Q_0, F)$ akzeptierten Sprachen ist genau die Menge der durch reguläre Ausdrücke über Σ beschriebenen Sprachen.*

Beweis: Der Beweis erfolgt in einer Richtung dadurch, dass gemäß der induktiven Definition von regulären Sprachen nichtdeterministische endliche Automaten (*NFA*s) gekoppelt werden und dadurch der Abschluss unter Vereinigung, Produkt und *-Hülle gezeigt wird. In der anderen Richtung besteht der Beweis in der berühmten Konstruktion von Kleene. Beide Verfahren sind in den Unterlagen zur Vorlesung FGI-1 nachzulesen. □

1.1.5 Produktautomaten

Reguläre Mengen sind abgeschlossen unter Durchschnitt. Dies wird oft (so auch in FGI-1) darauf zurückgeführt, dass sie unter Vereinigung und Komplement abgeschlossen sind. Die Produktautomaten-Konstruktion erlaubt einen direkten Beweis, was im Rahmen des Model-Checking von besonderer Bedeutung ist.

Satz 1.8 *Gegeben seien die Automaten $A_i = (Q_i, \Sigma, \delta_i, Q_i^0, F_i)$ für $i = 1, 2$.*
Aus A_1 und A_2 kann effektiv der Automat $A_3 = (Q_3, \Sigma, \delta_3, Q_3^0, F_3)$ konstruiert werden, der den Durchschnitt der beiden Sprachen akzeptiert: $L(A_3) = L(A_1) \cap L(A_2)$

Beweis: Die Zustände von A_3 ergeben sich als kartesisches Produkt. Wir definieren $Q_3 = Q_1 \times Q_2$, $Q_3^0 := Q_1^0 \times Q_2^0$ und $F_3 := F_1 \times F_2$. Die Übergänge werden synchronisiert:

$$(s, r) \xrightarrow[3]{a} (s', r') \qquad \Longleftrightarrow \qquad (s \xrightarrow[1]{a} s' \wedge r \xrightarrow[2]{a} r')$$

Es sei nun $w \in L(A_1) \cap L(A_2)$ und $w = a_1 a_2 \cdots a_n$.
Dann gilt $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \cdots s_{n-1} \xrightarrow{a_n} s_n$ mit $s_0 \in Q_1^0$, $s_n \in F_1$ in A_1 und $r_0 \xrightarrow{a_1} r_1 \xrightarrow{a_2} r_2 \cdots r_{n-1} \xrightarrow{a_n} r_n$ mit $r_0 \in S_2^0$, $r_n \in F_2$ in A_2.
Dann ist nach Konstruktion auch

$$(s_0, r_0) \xrightarrow[3]{a_1} (s_1, r_1) \xrightarrow[3]{a_2} (s_2, r_2) \cdots (s_{n-1}, r_{n-1}) \xrightarrow[3]{a_n} (s_n, r_n)$$

mit $(s_0, r_0) \in Q_1^0 \times Q_2^0$ und $(s_n, r_n) \in F_1 \times F_2$ und somit ist $w \in L(A_3)$.
Der umgekehrte Schluss erfolgt entsprechend. □

1.1.6 Entscheidungsprozeduren

Folgende Fragen sind (u.a. auch im Kontext des Model-Checkings) von Interesse

- Das *Leerheitsproblem* für NFA ist die folgende Frage:
  „Gegeben ein NFA A. Gilt $L(A) = \emptyset$?“
- Das *Universalitätsproblem* für NFA ist die folgende Frage:
  „Gegeben ein NFA A. Gilt $L(A) = \Sigma^*$?“
- Das *Äquivalenzproblem* für NFA ist die folgende Frage:
  „Gegeben zwei NFA A_1 und A_2. Gilt $L(A_1) = L(A_2)$?“

Lemma 1.9 *Das Leerheitsproblem für NFA ist entscheidbar.*
Das Universalitätsproblem für NFA ist entscheidbar.
Das Äquivalenzproblem für NFA ist entscheidbar.

Beweis: Als Übung. □

1.2 Büchi-Automaten

Für die Beschreibung und Analyse von reaktiven Systemen werden meist unendliche Aktionsfolgen benutzt. Auch hier sind in Bezug auf Endzustände akzeptable Aktionsfolgen definiert und zwar dadurch, dass die zugehörige Zustandsfolge die Endzustandsmenge F unendlich oft treffen muss.
Ein ω-Wort $\sigma \in \Sigma^\omega$ wird als $\sigma = a_0a_1a_2\cdots$ dargestellt, wobei $a_i \in \Sigma$ für alle i.
Die Menge aller ω-Wörter über dem Alphabet Σ wird mit Σ^ω bezeichnet.
$\Sigma^\infty := \Sigma^* \cup \Sigma^\omega$ bezeichnet die Menge der endlichen oder unendlichen Folgen von Zeichen aus Σ.
Für ein ω-Wort $\sigma = a_0a_1a_2\cdots$ notieren wir die Menge der unendlich häufig vorkommenden Elemente wie folgt:

$$\text{infinite}(\sigma) := \{a \in \Sigma \mid a \text{ kommt in } \sigma \text{ unendlich oft vor}\}$$

Eine Menge $L \subseteq \Sigma^\omega$ heißt *ω-Sprache*.
Wir wollen ω-Sprachen jetzt mit Hilfe von Automaten definieren. Ein (nichtdeterministischer) *Büchi-Automat* $A = (Q, \Sigma, \delta, Q_0, F)$ besteht aus den gleichen Komponenten wie ein NFA. Die Akzeptanzbedingung muss aber anders formuliert werden, da ja das Wort nie zu Ende ist.
Ein Pfad ist eine Folge $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} s_3 \cdots$, die in einem Startzustand beginnt ($s_0 \in Q_0$).

Definition 1.10 *Das ω-Wort $\sigma = a_0a_1a_2\cdots \in \Sigma^\omega$ wird von dem Büchi-Automat $A = (Q, \Sigma, \delta, Q_0, F)$* akzeptiert*, wenn ein Pfad $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} s_3 \cdots$ mit $s_0 \in Q_0$ existiert, der mindesten einen Endzustand unendlich oft durchläuft, d.h. es gilt*

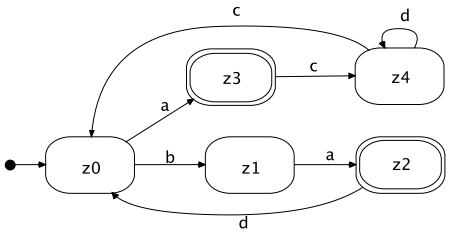
$$\text{infinite}(s_0s_1s_2\cdots) \cap F \neq \emptyset$$

$L^\omega(A) = \{w \in A^\omega \mid \sigma$ wird akzeptiert $\}$ ist die akzeptierte ω-Sprache *von A.*

Die Familie der von Büchi-Automaten akzeptierten ω-Sprachen wird als die Familie der *ω-regulären Sprachen* bezeichnet.

Beispiel 1.11 Der folgende Büchi-Automat akzeptiert die Sprache:

$$L^\omega(A) = \big(\{ac\}\{d\}^*\{c\} \cup \{bad\}\big)^\omega$$



1.2.1 Deterministische Büchi-Automaten

Ein Büchi-Automat A heißt *deterministisch*, wenn A als NFA betrachtet deterministisch ist.
Folgendes Lemma zeigt die Grenzen deterministischer Büchi-Automaten.

Lemma 1.12 *Es gibt keinen deterministischen Büchi-Automaten, der die Sprache $L = \{a, b\}^*\{b\}^\omega$ akzeptiert.*

Beweis: Angenommen es gäbe doch einen deterministischen Büchi-Automaten A, der die Sprache $L = \{a, b\}^*\{b\}^\omega$ akzeptiert.
Dann würde $\sigma_0 = b^\omega$ akzeptiert. Dann gäbe es einen Präfix u_0 von σ_0 (d.h. $u_0 \in \{b\}^*$), so dass $\delta^*(s_0, u_0) \in F$ gilt.
Betrachten wir nun $\sigma_1 = u_0 a b^\omega$. Das Wort σ_1 wird auch akzeptiert.
Da unendlich oft ein Endzustand eingenommen wird, gäbe es dann auch einen Präfix von σ_1 der Form $u_0 a u_1$, so dass $\delta^*(s_0, u_0 a u_1) \in F$ gilt.
Setzen wir das Argument fort, so erhalten wir endliche Wörter $u_i \in \{b\}^*$, so dass $\delta^*(s_0, u_0 a u_1 a \cdots a u_n) \in F$ gilt.
Da wir nur endlich viele Zustände haben, existieren zwei i, j mit $i < j$, so dass die erreichten Zustände gleich sind:

$$\delta^*(s_0, u_0 a u_1 a \cdots a u_i) = \delta^*(s_0, u_0 a u_1 a \cdots a u_j)$$

Also bildet das Wort $(a u_{i+1} a \cdots a u_j)$ einen Zyklus. Da $i < j$, ist das Wort nicht-leer. Dann würde auch das folgende ω-Wort akzeptiert:

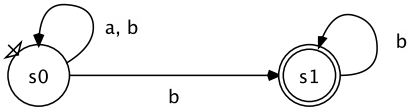
$$(u_0 a u_1 a \cdots a u_i) \cdot (a u_{i+1} a \cdots a u_j)^\omega$$

Dieses Wort hat allerdings unendlich viele Vorkommnisse von a, liegt also nicht mehr in $L = \{a, b\}^*\{b\}^\omega$. Widerspruch. □

Aus dem Lemma folgt sofort folgende Trennungseigenschaft.

Satz 1.13 *Nichtdeterministische Büchi-Automaten sind echt mächtiger als deterministische.*

Beweis: Die Sprache $L = \{a, b\}^*\{b\}^\omega$ kann zwar nicht von einem deterministischen Büchi-Automaten akzeptiert werden, sie wird aber von dem folgenden nichtdeterministischen Büchi-Automaten akzeptiert.



□

1.2.2 Abschlussoperationen auf ω-Sprachen

Wir definieren Operationen auf ω-Sprachen.

Definition 1.14 *Für ein (endliches) Wort $w = a_1a_2\cdots a_n \in \Sigma^*$ und ein ω-Wort $u = b_1b_2\cdots \in \Sigma^\omega$ sei $w \cdot u := a_1a_2\cdots a_nb_1b_2\cdots \in \Sigma^\omega$ die Konkatenation dieser Wörter.*
Entsprechend definiert man für Mengen $W \subseteq \Sigma^*$ und $U \subseteq \Sigma^\omega$ die Konkatenation $W \cdot U \subseteq \Sigma^\omega$ komponentenweise:

$$W \cdot U := \{w \cdot u \mid w \in W, u \in U\}$$

Ferner sei für $W \subseteq \Sigma^*$ der ω-Abschluss W^ω *folgendermaßen definiert:*

$$W^\omega := \{w_1 \cdot w_2 \cdot w_3 \cdots \mid w_i \in W \setminus \{\epsilon\}, i \geq 1\}$$

Vereinigung, Konkatenation und *ω-Abschluss* sind für ω-reguläre Mengen Abschlussoperatoren.

Lemma 1.15 *Wenn die Mengen $U, V \subseteq \Sigma^\omega$ zwei ω-reguläre Mengen sind und die Menge $W \subseteq \Sigma^*$ regulär ist, dann gilt:*

- *Die Vereinigung $U \cup V \subseteq \Sigma^\omega$ ist eine ω-reguläre Menge.*
- *Der ω-Abschluss $(W \setminus \{\epsilon\})^\omega$ ist eine ω-reguläre Menge.*
- *Die gemischte Konkatenation $W \cdot U \subseteq \Sigma^\omega$ ist eine ω-reguläre Menge.*

Beweis: Als Übung. □

Auch Komplementbildung ist für ω-reguläre Mengen eine Abschlussoperation.

Lemma 1.16 *Wenn die Menge $U \subseteq \Sigma^\omega$ eine ω-reguläre Menge ist, dann ist das Komplement $\Sigma^\omega \setminus U$ eine ω-reguläre Menge.*

Beweis: Zu aufwendig. □

Der Konstruktionsaufwand ist aber mehr als exponentiell in der Größe des Ausgangsautomaten!

1.2.3 Büchi-Automaten und ω-reguläre Ausdrücke

Wir definieren nun das Analogon der regulären Ausdrücke für unendliche Wörter.

Definition 1.17 *Seien $A_1, \cdots, A_n, B_1, \cdots, B_n$ reguläre Ausdrücke über Σ, wobei kein M_{B_i} das leere Wort ϵ enthält.*
Ein ω-regulärer Ausdruck *über dem Alphabet Σ ist ein Ausdruck der Form:*

$$G = A_1 \cdot B_1^\omega + \cdots + A_n \cdot B_n^\omega$$

Dieser Ausdruck beschreibt die ω-reguläre Sprache

$$M_G := M_{A_1} \cdot M_{B_1}^\omega \cup \cdots \cup M_{A_n} \cdot M_{B_n}^\omega.$$

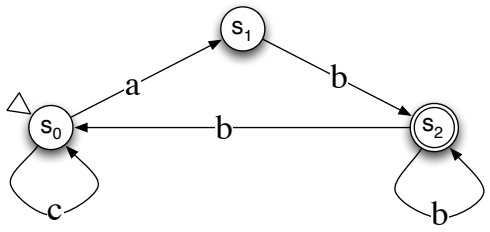


Abbildung 1.2: Büchi-Automat A mit ω-Sprache $L^\omega(A) = M_G$ mit $G = c^*ab(b^+ + bc^*ab)^\omega$

Der Büchi-Automat A in Abbildung 1.2 akzeptiert die ω-Sprache $L^\omega(A) = M_G$ mit

$$G = c^*ab(b^+ + bc^*ab)^\omega.$$

Der nächste Satz erweitert die Äquivalenz von regulären Ausdrücken und endlichen Automaten auf eine Beziehung von Büchi-Automaten und ω-regulären Ausdrücken.

Satz 1.18 *Die Familie der ω-regulären Sprachen ist genau die Familie der durch die ω-regulären Ausdrücke beschriebenen Sprachen.*

Beweis: Der Beweis für ω-Sprachen ist dem für reguläre Sprachen ähnlich.

Sei der ω-reguläre Ausdruck gegeben. Wir wissen bereits, wie man zu jedem regulären Ausdruck einen äquivalenten NFA konstruieren kann.

Nach Lemma 1.15 existieren Konstruktionen für Büchi-Automaten, die dann $U \cup V$, W^ω und $W \cdot U$ akzeptieren. So erzeugt man induktiv über den Aufbau des Ausdruck einen akzeptierenden Büchi-Automaten.

Sei umgekehrt ein Büchi-Automat A gegeben. Mit der Kleene-Konstruktion können wir die Menge $R_{p,q}$ aller Wörter bestimmen, die im Zustand p starten und in q enden. Diese Mengen sind jeweils regulär.

Ein ω-Wort wird genau dann akzeptiert, wenn es einen Endzustand q gibt, der auf einem Kreis liegt und der Kreis ist von einem Startzustand p aus erreichbar.

Das ist gleichbedeutend damit, dass q erreichbar ist und es einen Kreis von q nach q gibt. Alle diese Wörter haben dann folgende Darstellung:

$$L^\omega(A) = \sum_{p \in Q_0, q \in F} R_{pq} \cdot R_{qq}^\omega$$

Für Details siehe [BK08]. □

1.2.4 Produktautomat: Abschluss bzgl. Mengenschnitt

Auch die ω-reguläre Mengen sind bzgl. Durchschnitt abgeschlossen. Die Produktautomaten-Konstruktion für NFA ist aber hier nicht ausreichend. Es gilt die folgende Inklusion:

Lemma 1.19 *Sei A_3 der Produktautomat der NFA A_1 und A_2. Dann gilt*

$$L^\omega(A_3) \subseteq (L^\omega(A_1) \cap L^\omega(A_2))$$

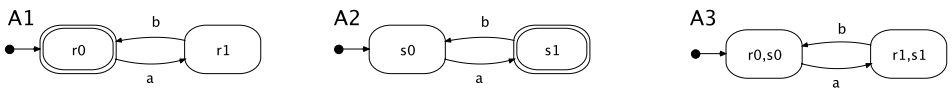
Beweis: Gegeben seien die Automaten $A_i = (Q_i, \Sigma, \delta_i, Q_i^0, F_i)$ für $i = 1, 2$. Sei A_3 der Produktautomat von A_1 und A_2.
Ist dann $\sigma = a_1 a_2 \cdots \in L^\omega(A_3)$, also

$$(s_0, r_0) \xrightarrow{a_1} (s_1, r_1) \xrightarrow{a_2} (s_2, r_2) \xrightarrow{a_3} \cdots$$

mit $(s_0, r_0) \in Q_1^0 \times Q_2^0$ und $(s_i, r_i) \in F_1 \times F_2$ für unendlich viele $i \geq 1$, dann gilt entsprechendes auch für die beiden Komponenten und es ist $w \in L^\omega(A_1) \cap L^\omega(A_2)$. Also gilt stets $L^\omega(A_3) \subseteq (L^\omega(A_1) \cap L^\omega(A_2))$. □

Der Umkehrschluss ist jedoch so nicht möglich, da die beiden Automaten ja immer zu unterschiedlichen Zeitpunkten in die Endzustände gelangen können, wie das folgende Beispiel zeigt.

Beispiel 1.20 Betrachte die beiden folgenden Automaten A_1 und A_2. Es gilt: $L^\omega(A_1) = L^\omega(A_2) = (ab)^\omega$. Also gilt: $L^\omega(A_1) \cap L^\omega(A_2) = (ab)^\omega$.
Das Produkt A_3 enthält nun gar keine erreichbaren Endzustände. Also gilt $L^\omega(A_3) = \emptyset$.



Es ist aber möglich, die Produktautomatenkonstruktion so zu erweitern, dass der Durchschnitt akzeptiert wird:

Satz 1.21 *Aus zwei Büchi-Automaten $A_i = (Q_i, \Sigma, \delta_i, Q_i^0, F_i), i = 1, 2$ kann effektiv ein Büchi-Automat $A_4 = (Q_4, \Sigma, \delta_4, Q_4^0, F_4)$ konstruiert werden, der den Durchschnitt der akzeptierten ω-Sprachen akzeptiert:*

$$L^\omega(A_4) = L^\omega(A_1) \cap L^\omega(A_2)$$

Beweis: Wir erweitern A_3, indem wir die Zustände (s, r) um eine dritte Komponente i erweitern, die erzwingt, dass beide Automaten jeweils unendlich oft einen Endzustand besuchen. Die dritte Komponente gibt den Automaten an, dessen Endzustand gerade erwartet wird:

- $(s, r, 1)$ bedeutet dabei, dass A_4 darauf wartet, dass wir einen Endzustand aus A_1 einnehmen. Wird dieser durchlaufen, so wechselt die dritte Komponenten auf $i = 2$.
- $(s, r, 2)$ bedeutet entsprechend, dass wir auf einen Endzustand aus A_2 warten.

Ist $w \in L^\omega(A_1)$ und $w \in L^\omega(A_2)$, dann muss immer wieder (in jeder Komponente) ein Endzustand eingenommen werden.

Wenn wir als Endzustände nur solche mit $i = 1$ zulassen, dann erzwingen wir, dass Zustände (s, r, i) mit $s \in F_1$ und $r \in F_2$ unendlich oft im Wechsel von $i = 1$ und $i = 2$ eingenommen werden. Also gilt $L^\omega(A_1) \cap L^\omega(A_2) \subseteq L^\omega(A_4)$.

Formal definieren wir $A_4 = (Q_4, \Sigma, \delta_4, Q_4^0, F_4)$ durch

- $Q_4 = \{(s, r, i) \mid (s, r) \in Q_3, i \in \{1, 2\}\}$
- $(s, r, i) \xrightarrow[4]{a} (s', r', i') \iff (s, r) \xrightarrow[3]{a} (s', r') \ \wedge \ i' := \begin{cases} 2, & \text{falls } i = 1 \wedge s \in F_1 \\ 1, & \text{falls } i = 2 \wedge r \in F_2 \\ i, & \text{sonst} \end{cases}$
- $Q_4^0 := \{(s, r, 1) \mid (s, r) \in Q_3^0 = Q_1^0 \times Q_2^0\}$
- $F_4 := \{(s, r, 1) \mid s \in F_1\}$

Weiterhin gilt auch (wie schon bei A_3), dass jedes $w \in L^\omega(A_4)$ sowohl in A_1 als auch in A_2 akzeptiert wird: $L^\omega(A_4) \subseteq L^\omega(A_1) \cap L^\omega(A_2)$.

Insgesamt gilt also $L^\omega(A_4) = L^\omega(A_1) \cap L^\omega(A_2)$. □

1.2.5 Verallgemeinerte Büchi-Automaten

Ein *verallgemeinerter Büchi-Automat* besitzt statt einer Endzustandsmenge gleich mehrere.

Definition 1.22 *Ein* verallgemeinerter Büchi-Automat *ist das Tupel* $A = (Q, \Sigma, \delta, Q_0, \mathcal{F})$*, wobei* $\mathcal{F} = \{F_1, \ldots, F_k\}$ *eine Menge von Endzustandsmengen ist.*

Das ω*-Wort* $\sigma = a_0 a_1 a_2 \cdots \in \Sigma^\omega$ *wird von einem verallgemeinerten Büchi-Automat* A akzeptiert*, wenn ein Pfad* $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} s_3 \cdots$ *mit* $s_0 \in Q_0$ *existiert, der in jeder Menge* $F \in \mathcal{F}$ *mindesten einen Endzustand unendlich oft durchläuft, d.h. es gilt*

$$\forall F \in \mathcal{F} : \text{infinite}(s_0 s_1 s_2 \cdots) \cap F \neq \emptyset$$

$L^\omega(A) = \{w \in A^\omega \mid \sigma$ *wird akzeptiert* $\}$ *ist die* akzeptierte ω-Sprache *von* A.

Satz 1.23 *Zu jedem verallgemeinerten Büchi-Automaten* A *existiert ein Büchi-Automat* A'*, der die gleiche Sprache akzeptiert:* $L^\omega(A) = L^\omega(A')$.

Beweis: Sei $\mathcal{F} = \{F_1, \ldots, F_n\}$.

(Skizze) Wir definieren den Büchi-Automaten A' folgendermaßen: Man erzeuge n Kopien von A und starte in der 1. Kopie. Wenn A in der i. Kopie einen Endzustand aus F_i erreicht, wird in den gleichen Zustand der $(i+1)$. Kopie gewechselt. Als Endzustandsmenge wähle man F_1.
Akzeptiert A' ein ω-Wort σ, dann müssen dabei alle n Kopien durchlaufen worden sein und in jeder Kopie muss ein Endzustand erreicht worden sein. Also hätte auch A das Wort σ akzeptiert.
Akzeptiert umgekehrt A das ω-Wort σ, dann wird mindestens ein Zustand s_i aus jedem F_i unendlich oft durchlaufen. In A' wird bei s_i in die $(i+1)$. Kopie gewechselt, bei s_n wieder zurück in in die 1. Kopie. Also hätte auch A das Wort σ akzeptiert. □

1.2.6 Entscheidungsprozeduren

Folgende Fragen sind (u.a. auch im Kontext des Model-Checkings) von Interesse:

- Das *Leerheitsproblem* für Büchi-Automaten ist die folgende Frage:
  „Gegeben ein Büchi-Automat A. Gilt $L^\omega(A) = \emptyset$?“
- Das *Universalitätsproblem* für Büchi-Automaten ist die folgende Frage:
  „Gegeben ein Büchi-Automat A. Gilt $L^\omega(A) = \Sigma^\omega$?“
- Das *Äquivalenzproblem* für Büchi-Automaten ist die folgende Frage:
  „Gegeben zwei Büchi-Automaten A_1 und A_2. Gilt $L^\omega(A_1) = L^\omega(A_2)$?“

Lemma 1.24 *Das Leerheitsproblem für Büchi-Automaten ist entscheidbar.*
Das Universalitätsproblem für Büchi-Automaten ist entscheidbar.
Das Äquivalenzproblem für Büchi-Automaten ist entscheidbar.

Beweis: Als Übung. □

2 Transitionssysteme und Kripke-Strukturen

Das ursprüngliche[1] Modell des endlichen Automaten wird für die Beschreibung und Analyse von Systemverhalten in der Form von *Transitionssystemen* fortgeführt. In den verschiedenen Anwendungen sind unterschiedliche Varianten von Transitionssystemen gebräuchlich, die aber wesentliche Komponenten gemeinsam haben, nämlich Zustände und Übergänge (Transitionen) zwischen Zuständen. In diesem Kapitel werden die grundlegenden Definitionen von Transitionssystemen vorgestellt, sowie einige ihrer wichtigsten Eigenschaften und Methoden, wie Bisimulation und Synchronisation.

2.1 Transitionssysteme

Transitionssysteme bestehen aus Zuständen, darunter Anfangs- und Endzustände, sowie einer Transitionsrelation. In manchen Darstellungen der Literatur ist nur ein einziger Anfangszustand vorgesehen oder es wird auf Endzustände verzichtet.
Beispielsweise generiert eine Turingmaschine ein Transitionssystem: Die Konfigurationen der TM sind die Zustände und die Übergänge die Aktionen. Endliche Automaten sind Beispiele endlicher Transitionssysteme. Die Semantik von prozessalgebraischen Ausdrücken (Kapitel 9) und von Petrinetzen (Kapitel 6) wird ebenfalls durch Transitionssysteme beschrieben. Beim Model-Checking (Abschnitte 2.5 und 3.1) kommen Transitionssysteme zum Einsatz, bei denen die Zustände mit gültigen Aussagen etikettiert werden.

Definition 2.1 *Ein* Transitionssystem $TS = (S, A, tr, S^0, S^F)$ *besteht aus*

- *einer Menge* S *von* Zuständen*,*
- *einer Menge* A *von* Aktionen *(auch:* Transitionen*),*
- *einer* Transitionsrelation $tr \subseteq S \times A \times S$*,*
- *einer Menge* $S^0 \subseteq S$ *von* Anfangszuständen *und*
- *einer Menge* $S^F \subseteq S$ *von* Endzuständen*,*

TS *heißt* endlich*, falls* S *und* A *endlich sind.*

Existiert nur ein Anfangszustand, dann schreibt man auch im Tupel $TS = (S, A, tr, S^0, S^F)$ nur s_0 statt $\{s_0\}$. Wird S^F in dem Tupel weggelassen, dann sind alle Zustände Endzustände, d.h. $S^F = S$.

[1] „ursprünglich“ sowohl im historischen Sinn als auch im Rahmen der Veranstaltungen FGI-1 und FGI-2

Ein Transitionssystem ist somit also ein kantenbeschrifteter Graph (evtl. unendlich groß), bei dem einige Knoten als Start- bzw. Endzustände gekennzeichnet sind.
Statt $(s, a, s') \in tr$ wird meist die intuitivere Schreibweise $s \xrightarrow{a} s'$ benutzt. Diese erweitert sich (wie zuvor für Automaten) auch auf A^*.
Ein *unendlicher Pfad* ist eine in einem Startzustand beginnende (d.h. $s_0 \in S^0$) Folge der Form:

$$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} s_3 \cdots$$

Das dynamische Verhalten („die Semantik“) eines Transitionssystems wird durch die erreichbaren Zustände und durch seine (endlichen oder unendlichen) Transitions- bzw. Aktionsfolgen beschrieben.

- $R(TS, S_1) := \{s \in S \mid \exists w \in A^*, s_1 \in S_1 : s_1 \xrightarrow{w} s\}$ ist die *Menge der von S_1 aus in TS erreichbaren Zustände* und
- $R(TS) := R(TS, S^0)$ ist die *Erreichbarkeitsmenge von TS*.

Entsprechend definiert man für Aktionsfolgen:

- $AS(TS, S_1) := \{w \in A^* \mid \exists s_1 \in S_1, s_1' \in S : s_1 \xrightarrow{w} s_1'\}$ ist die *Menge der von S_1 aus in TS möglichen Aktionsfolgen.*
- $AS(TS) := AS(TS, S^0)$ ist die *Aktionsfolgenmenge von TS*.
- $FS(TS) := \{w \in A^* \mid \exists s_1 \in S^0, s_1' \in S^F : s_1 \xrightarrow{w} s_1'\}$ ist die *terminale Aktionsfolgenmenge von TS*.
  Diese Menge heißt oft auch *akzeptable Aktionsfolgenmenge* oder *akzeptierte Sprache* und wird als $L(TS)$ bezeichnet.

Lemma 2.2 *Sei TS ein Transitionssystem, dann gilt $FS(TS) \subseteq AS(TS)$.*
Es gilt $FS(TS) = AS(TS)$, falls $S^F = S$.

Beweis: Als Übung. □

Analog definieren wir für unendliche Zustandsfolgen:

- $SS(TS, S_1) := \{s_0 s_1 s_2 \cdots \mid TS$ besitzt einen Pfad $s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots$ mit $s_0 \in S_1\}$ ist die Menge aller *Zustandsfolgen* von TS.
- $SS(TS) := SS(TS, S^0)$.

2.2 Bisimilarität bei reaktiven Systemen

Bei Transitionssystemen unterscheiden wir zwei Formen der Äquivalenz: Folgenäquivalenz und Akzeptanzäquivalenz. Akzeptanzäquivalenz entspricht der bei NFA gebräuchlichen Äquivalenz.

Definition 2.3 *Zwei Transitionssysteme TS_1 und TS_2 heißen* folgenäquivalent*, falls sie die gleiche Menge von Aktionsfolgen haben:*

$$TS_1 \sim_{AS} TS_2 :\Leftrightarrow AS(TS_1) = AS(TS_2)$$

Sie heißen terminal folgenäquivalent *oder* akzeptanzäquivalent*, falls gilt:*

$$TS_1 \sim_L TS_2 :\Leftrightarrow L(TS_1) = L(TS_2)$$

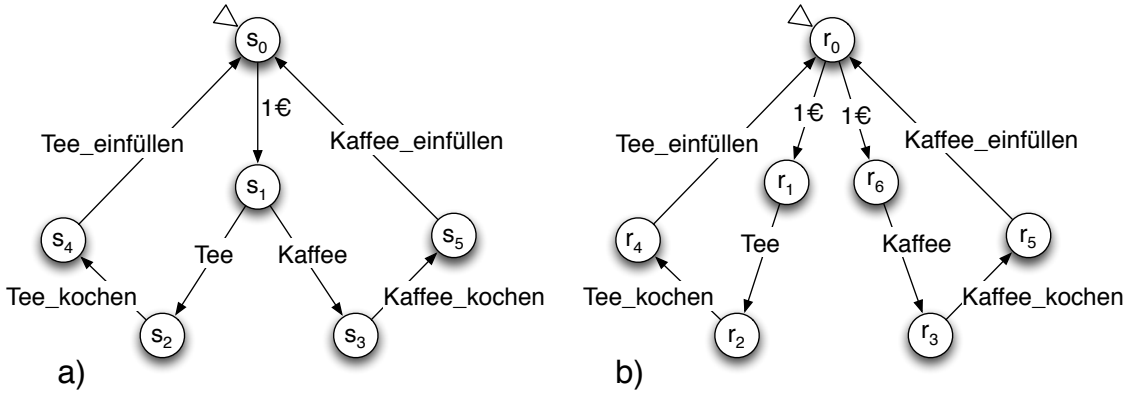


Abbildung 2.1: Zwei folgenäquivalente Transitionssysteme

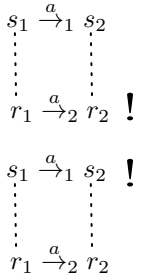
Abbildung 2.1 zeigt zwei folgenäquivalente Transitionssysteme. Das Beispiel zeigt auch, dass in manchen Fällen die Folgenäquivalenz kein adäquates Mittel ist, um gleiches Systemverhalten zu definieren: Im linken System sind nach dem Münzeinwurf beide Wahlmöglichkeiten geben, was in dem rechten System nicht der Fall ist.
Eine bessere Formalisierung für gleiches Systemverhalten ist durch den Begriff der *Bisimulation* möglich.
Wir betrachten für die folgende Definition zwei Transitionssysteme mit der gleichen Aktionsmenge A. Dies ist keine echte Einschränkung, da dies durch Obermengenbildung immer zu erreichen ist.

Definition 2.4 *Gegeben seien zwei Transitionssysteme $TS_i = (S_i, A, tr_i, S_i^0, S_i^F)$ für $i \in \{1, 2\}$.*
Eine (aktionsbasierte) Bisimulation *für (TS_1, TS_2) ist eine binäre Relation $\mathcal{B} \subseteq S_1 \times S_2$, für die folgendes gilt:*

a) $\forall s_0 \in S_1^0 : \exists r_0 \in S_2^0 : (s_0, r_0) \in \mathcal{B}$
$\forall r_0 \in S_2^0 : \exists s_0 \in S_1^0 : (s_0, r_0) \in \mathcal{B}$

b) Für alle $(s_1, r_1) \in \mathcal{B}$ *gilt:*
$s_1 \xrightarrow[1]{a} s_2 \Rightarrow \exists r_2 \in S_2 : r_1 \xrightarrow[2]{a} r_2 \wedge (s_2, r_2) \in \mathcal{B}$
$r_1 \xrightarrow[2]{a} r_2 \Rightarrow \exists s_2 \in S_1 : s_1 \xrightarrow[1]{a} s_2 \wedge (s_2, r_2) \in \mathcal{B}$

c) $\forall (s, r) \in \mathcal{B} : s \in S_1^F \Leftrightarrow r \in S_2^F$



Zustände mit $(s, r) \in \mathcal{B}$ *heißen* bisimilar *(in Zeichen* $s \underline{\leftrightarrow} r$*).*

TS_1 und TS_2 heißen bisimilar *(in Zeichen $TS_1 \underline{\leftrightarrow} TS_2$), falls eine solche Bisimulations-Relation $\mathcal{B}$ existiert.*

Die Intention der Bedingungen a) und c) in Definition 2.4 sind offensichtlich: durch sie gibt es zu jedem Anfangszustand einen bisimularen im anderen Transitionssystem. Ein Endzustand hat nur ebensolche bisimulare Partner. Die Bedingung b) wird durch die daneben stehende Graphik illustriert. Die obere dieser Graphiken entspricht dem ersten Teil der Bedingung b). Das Ausrufezeichen markiert das postulierte Element r_2, während die anderen Elemente s_1, s_2 und r_1 zur Voraussetzung der Bedingung gehören. Die durch die Relation $\mathcal{B}$ verbundenen Paare $s_1 \underline{\leftrightarrow} r_1$ und $s_2 \underline{\leftrightarrow} r_2$ sind durch die unterbrochenen Linien gekennzeichnet. Entsprechendes gilt für den zweiten Teil der Bedingung b) und die untere Graphik.

Beispiel 2.5 Wir zeigen nun, dass die (intuitiv) nicht verhaltensäquivalenten Transitionssysteme aus Abbildung 2.1 tatsächlich nicht (formal) bisimilar sind. Wären sie bisimilar, dann müssten wegen der Bedingung a) die Anfangszustände in der Relation stehen: $s_0 \underline{\leftrightarrow} r_0$. Aus der ersten Bedingung b) folgt dann, dass mit $a = 1€$ auch $s_1 \underline{\leftrightarrow} r_1$ oder $s_1 \underline{\leftrightarrow} r_6$ gelten muss. Im ersten Fall gibt es dann zwar (wieder nach Bedingung b) zu $s_1 \xrightarrow{\text{Tee}} s_2$ die Transition $r_1 \xrightarrow{\text{Tee}} r_2$, nicht aber zu $s_1 \xrightarrow{\text{Kaffee}} s_3$ eine Transition $r_1 \xrightarrow{\text{Kaffee}} r$ für irgend ein r. Da der zweite Fall in analoger Weise zum Widerspruch führt, können die Transitionssysteme nicht bisimilar sein.
Im Gegensatz dazu sind die Transitionssysteme aus Abbildung 2.2 bisimilar. Die Bisimulations-Relation ist $\mathcal{B} = \underline{\leftrightarrow} = \{(s_i, r_i) \mid i = 0, \cdots, 5\} \cup \{(s_1, r_6)\}$.

Lemma 2.6 *Seien TS_1 und TS_2 zwei beliebige Transitionssysteme mit $TS_1 \underline{\leftrightarrow} TS_2$, dann gilt f¸r alle $w \in A^*$:*

$$(s_0 \xrightarrow[1]{w} s_1 \ \wedge s_0 \underline{\leftrightarrow} r_0) \ \Rightarrow \ (\exists r_1 : r_0 \xrightarrow[2]{w} r_1 \wedge s_1 \underline{\leftrightarrow} r_1)$$

Beweis: Wir beweisen dies durch Induktion über die Wortlänge $|w|$.

- Induktionsanfang $w = \epsilon$: In diesem Fall gilt $s_1 = s_0$ und r_1 kann als r_0 gewählt werden.
- Induktionsschritt $w = va$ mit $v \in A^*, a \in A$:
  Sei also $s_0 \xrightarrow[1]{va} s_2$ und $s_0 \underline{\leftrightarrow} r_0$. Dann gibt es ein $s_1 \in S_1$ mit $s_0 \xrightarrow[1]{v} s_1 \xrightarrow[1]{a} s_2$.

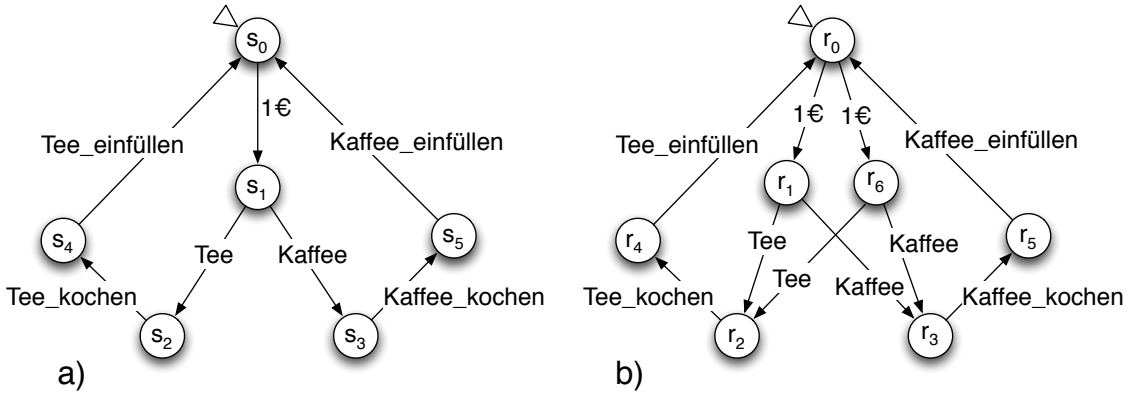


Abbildung 2.2: Zwei bisimilare Transitionssysteme

Nach Induktionsvoraussetzung gibt es einen Zustand r_1 mit $r_0 \xrightarrow[2]{v} r_1$ und $s_1 \underline{\leftrightarrow} r_1$.
Aufgrund der Bisimilarität gibt es r_2 mit $r_1 \xrightarrow[2]{a} r_2$ und $s_2 \underline{\leftrightarrow} r_2$. Also gilt $r_0 \xrightarrow[2]{va} r_2$ und $s_2 \underline{\leftrightarrow} r_2$.

$$\begin{array}{ccccc} s_0 & \xrightarrow{v}_1 & s_1 & \xrightarrow{a}_1 & s_2 \\ \vdots & & \vdots & & \vdots \\ r_0 & \xrightarrow{v}_2 & r_1 & \xrightarrow{a}_2 & r_2 \; ! \end{array}$$

Damit ist die Aussage bewiesen. □

Satz 2.7 *Seien TS_1 und TS_2 zwei beliebige bisimilare Transitionssysteme, dann hat jeder erreichbare Zustand des einen Systems einen bisimilaren Partner im anderen:*

$$\forall s \in R(TS_1) : \exists r \in R(TS_2) : s \underline{\leftrightarrow} r$$
$$\forall r \in R(TS_2) : \exists s \in R(TS_1) : s \underline{\leftrightarrow} r$$

Beweis: Wegen der Symmetrie der Aussage in TS_1 und TS_2 reicht es aus, nur eine Hälfte zu zeigen: Wir zeigen $\forall s \in R(TS_1) : \exists r \in R(TS_2) : s \underline{\leftrightarrow} r$.
Wenn s ein erreichbarer Zustand ist, dann existiert ein Wort w, mit dem der Zustand s von einem Startzustand $s_0 \in S_1^0$ erreicht werden kann: $s_0 \xrightarrow{w} s$.
Nach Def. 2.4 a) existiert zu s_0 ein bisimilarer Partner $r_0 \in S_2^0$ mit $(s_0, r_0) \in \mathcal{B}$.
Nach Lemma 2.6 gibt es $r \in S_2$ mit $r_0 \xrightarrow[2]{w} r$ und $s \underline{\leftrightarrow} r$.
Offensichtlich ist r auch erreichbar. □

Der folgende Satz zeigt, dass Bisimilarität feiner als Folgenäquivalenz ist, denn Bisimilarität trennt auch folgenäquivalente Systeme (vgl. dazu das Beispiel in Abbildung 2.1).