Automatentheorie

WS 1994/95
RWTH Aachen

Mitschrift

Vorwort

Die vorliegende Mitschrift der Vorlesung “Automatentheorie”, gelesen von Prof. Dr. F. Baader im Wintersemester 1994/95 an der RWTH Aachen, ist genau das, was der Name sagt: Die geTEXte Form der Vorlesungsmitschrift einer unwissenden Studentin, angereichert mit einigen wenigen, nicht weiter gekennzeichneten privaten Ergänzungen (in der Regel triviale Beweise, auf die Prof. Baader in der Vorlesung verzichtet hat).
Es ist wahrscheinlich, daß sich an der einen oder anderen Stelle Fehler eingeschlichen haben. (Verbesserungsvorschläge und “Fehlermeldungen” aller Art sind mir natürlich willkommen.) Deshalb ist dieses Schriftstück mit Vorsicht zu genießen und von mir nur als kleine Unterstützung für Leute gedacht, die sich mit der Vorlesung auseinandersetzen (wollen).
Auf gar keinen Fall sollte es als “Skript” mißverstanden werden. Dies ist es nicht!

Aachen, März 1996
Claudia Krobb

e-mail: claudia@kawo2.rwth-aachen.de

Inhaltsverzeichnis

Motivation und Einordnung

Man ist interessiert an der Untersuchung von Sprachen und deren Eigenschaften.
Unter den formalen Sprachen versteht man die Menge $L \subseteq \Sigma^*$, wobei Σ meist endlich ist.
Eine Klasse K formaler Sprachen ordnet jedem (endlichen) Alphabet Σ eine Menge $K_\Sigma \subseteq 2^{\Sigma^*}$ zu, d.h. eine Menge von Sprachen über Σ.
Drei Fragestellungen sind für solche Sprachen interessant

1. Charakterisierungen
   Gesucht ist eine Eigenschaft E, die die Bedingung

   $$L \in K_\Sigma \text{ gdw } L \subseteq \Sigma^* \text{ und } L \text{ erfüllt } E$$

   erfüllt. Meist sind verschiedene äquivalente Charakterisierungen gesucht.

2. Abschlußeigenschaften
   Unter welchen Operationen auf Sprachen (Vereinigung, Komplement, homomorphe Bilder, ...) ist die Klasse abgeschlossen?

3. Entscheidbarkeit
   Welche Fragestellungen ($w \in L$, $L \neq \emptyset$, $L_1 \subseteq L_2$, ...) sind für die Sprachklasse entscheidbar?
   Dabei geht man davon aus, daß die Sprachen durch eine der Charakterisierungen endlich beschrieben sind.

Beispiel 0.1 (Chomsky-Hierarchie)

Klasse	Charakterisierung
Typ 0	allgemeine Chomsky-Grammatiken (Regeln der Form $u \to v$, u enthält mindestens ein Nonterminal) akzeptiert von Turing-Maschinen
Typ 1 (kontextsensitiv)	kontexsensitive Grammatiken (Regeln der Form $u \to v$, $1 \leq \lvert u \rvert \leq \lvert v \rvert$) akzeptiert von Turing-Maschinen mit linearer Beschränkung
Typ 2 (kontextfrei)	kontextfreie Grammatiken (Regeln der Form $X \to v$, X ist Nonterminal) akzeptiert durch Kellerautomaten (PDA)
Typ 3 (regulär)	rechtslineare Grammatiken (Regeln der Form $X \to uY$ oder $X \to u$, u ist Terminalwort, X,Y sind Nonterminals) akzeptiert durch endliche Automaten

□

Beispiel 0.2 (Äquivalente Charakterisierungen)
Liefern deterministische Automaten / Maschinen bereits die gesamte Sprachklasse?

Typ 0	ja
Typ 1	offen
Typ 2	nein
Typ 3	ja

□

Beispiel 0.3 (Abschlußeigenschaften am Beispiel Komplement)
Fragestellung: Gilt $L \in K_\Sigma \Rightarrow (\Sigma^* \setminus L) \in K_\Sigma$?

Typ 0	nein	(Komplement einer rekursiv aufzählbaren Menge ist im allgemeinen nicht rekursiv aufzählbar)
Typ 1	ja	(1987)
Typ 2	nein	$(\Rightarrow L(DPDA) \neq L(PDA))$
Typ 3	ja	

□

Beispiel 0.4 (Entscheidungsprobleme)

	Elementproblem $w \in L$	Äquivalenzproblem $L_1 = L_2$
Typ 0	unentscheidbar	unentscheidbar
Typ 1	entscheidbar	unentscheidbar
Typ 2	entscheidbar	unentscheidbar
Typ 3	entscheidbar	entscheidbar

□

Inhalt der Vorlesung
Die Vorlesung befaßt sich mit den Sprachen vom Typ 3, insbesondere mit ihren Unterklassen, verschiedenen Charakterisierungen und Verallgemeinerungen auf unendliche Wörter und Bäume. Hauptmotivation der Vorlesung sind Anwendungen der Ergebnisse in der Logik (z.B. Entscheidbarkeitsresultate).

- **Teil 1** befaßt sich mit
  - Regulären Sprachen endlicher Wörter
  - Charakterisierungen
    - ∗ algebraische Charakterisierung
      Jeder Sprache kann ein Monoid (der syntaktische Monoid) zugeordnet werden. Reguläre Sprachen sind genau die Sprachen mit endlichem syntaktischen Monoid.
    - ∗ logische Charakterisierung
      Logische Formeln können Sprachen definieren. Es gibt eine Logik (die monadische Logik 2-ter Stufe), deren Formeln genau die regulären Sprachen definieren.

Mit Hilfe dieser Charakterisierungen können Unterklassen der Klasse der regulären Sprachen definiert werden. Eine solche Unterklasse ist die Klasse der sternfreien Sprachen. Diese werden durch Formeln der Logik 1-ter Stufe definiert. Ihre syntaktischen Monoide sind genau die aperiodischen Monoide.

- **Teil 2** befaßt sich mit Sprachen unendlicher Wörter. Statt endlicher Wörter (endliche Folgen von Elementen des Alphabets) kann man unendliche Wörter (unendliche Folgen) als Eingabe für einen endlichen Automaten betrachten. Die Akzeptanzbedingung muß dabei geändert werden.
  Für endliche Wörter wird das Wort durch Erreichen eines Endzustandes nach dem Lesen akzeptiert.
  Für unendliche Wörter wird eine Bedingung an die unendlich oft erreichten Zustände gestellt.
  Wir werden Abschlußeigenschaften, Entscheidbarkeit und den Zusammenhang zur Logik betrachten.

- **Teil 3** befaßt sich mit Baumsprachen (Wäldern). Wörter können als beschriftete Bäume vom Verzweigungsgrad 1 aufgefaßt werden. Baumautomaten sind eine Verallgemeinerung von endlichen Automaten, die Bäume vom Verzweigungsgrad > 1 behandeln können.
  Viele Resultate für reguläre Sprachen lassen sich auf Baumsprachen übertragen. Auch hier ist der Zusammenhang zur Logik wichtig. Anstelle von endlichen Bäumen kann man auch unendliche Bäume betrachten.

Kapitel 1

Reguläre Sprachen, endliche Monoide und logische Formeln

1.1 Reguläre Sprachen und endliche Automaten

Definition 1.1
Für ein endliches Alphabet Σ ist die Klasse der regulären Sprachen Reg_Σ über Σ die kleinste Klasse mit

- $\emptyset$, $\{\varepsilon\}$, $\{a\}$ mit $a \in \Sigma$ sind in Reg_Σ
- Sind L_1 und L_2 aus Reg_Σ, so auch
  - $L_1 \cup L_2$,
  - $L_1 \cdot L_2$ $(= \{uv \mid u \in L_1, v \in L_2\})$,
  - L_1^* $(=\{u_1 \dots u_n \mid u_i \in L_1, n \geq 0\})$.

□

Wie üblich beschreiben wir reguläre Sprachen durch reguläre Ausdrücke, bei denen Mengenklammern (und häufig auch Konkatenationspunkte) wegfallen. Zum Beispiel beschreibt $(ab)^*a$ den Ausdruck $(\{a\} \cdot \{b\})^* \cdot \{a\}$. Aus der Definition ergibt sich Abschluß unter Vereinigung, Konkatenation und Stern. Um Abschluß unter Komplement und Durchschnitt zu zeigen ist die Charakterisierung durch endliche Automaten besser geeignet.

Definition 1.2
Ein nicht-deterministischer endlicher Automat $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ besteht aus

- einer endlichen Menge von Zuständen Q
- einem endlichen Alphabet Σ
- einer Menge $I \subseteq Q$ von Anfangszuständen
- einer Übergangsrelation $\Delta \subseteq Q \times \Sigma \times Q$
- einer Menge $F \subseteq Q$ von Endzuständen.

Wie üblich werden Automaten graphisch dargestellt.
Ein Pfad in einem Automaten ist eine Folge $q_0 a_1 q_1 a_2 q_2 \dots a_n q_n$, wobei für $1 \leq i \leq n$ gilt, daß $(q_{i-1}, a_i, q_i) \in \Delta$.
Abkürzend schreiben wir $q_0 \xrightarrow{a_1 \dots a_n}_{\mathcal{A}} q_n$. Der Pfad ist erfolgreich, falls $q_0 \in I$ und $q_n \in F$.
Die von $\mathcal{A}$ akzeptierte Sprache ist $L(\mathcal{A}) := \{w \mid q_0 \xrightarrow{w}_{\mathcal{A}} q_n \text{ ist erfolgreich}\}$.

□

Der Satz von Kleene sagt, daß eine Sprache genau dann regulär ist, wenn sie von einem endlichen Automaten akzeptiert wird.
Als Beispiel verwenden wir diese Charakterisierung, um Abschluß der Klasse der regulären Sprachen unter Durchschnitt zu zeigen.

Satz 1.3
Sind $L_1, L_2 \in Reg_\Sigma$, so ist auch $L_1 \cap L_2 \in Reg_\Sigma$.

Beweis
Es seien $\mathcal{A}_1 = (Q_1, \Sigma, I_1, \Delta_1, F_1)$ und $\mathcal{A}_2 = (Q_2, \Sigma, I_2, \Delta_2, F_2)$ Automaten mit $L_1 = L(\mathcal{A}_1)$ und $L_2 = L(\mathcal{A}_2)$.
Wir definieren

$$\mathcal{A} := (Q_1 \times Q_2, \Sigma, I_1 \times I_2, \Delta, F_1 \times F_2),$$

wobei

$$\Delta = \{((q_1, q_2), a, (q_1', q_2')) \mid (q_1, a, q_1') \in \Delta_1, (q_2, a, q_2') \in \Delta_2\}.$$

Man sieht leicht: $(q_1, q_2) \xrightarrow{w}_{\mathcal{A}} (q_1', q_2')$ gdw $q_1 \xrightarrow{w}_{\mathcal{A}} q_1'$ und $q_2 \xrightarrow{w}_{\mathcal{A}} q_2'$.
Nach Definition der Anfangs- und Endzustandsmengen in $\mathcal{A}$ ist daher

$$w \in L(\mathcal{A}) \text{ gdw } w \in L(\mathcal{A}_1) \cap L(\mathcal{A}_2).$$

□

Um Abschluß unter Komplement zu zeigen, sind nicht-deterministische Automaten nicht gut geeignet.
Beachte: Ist $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein NEA mit $L \subseteq L(\mathcal{A})$, so ist im allgemeinen $\overline{\mathcal{A}} = (Q, \Sigma, I, \Delta, Q \backslash F)$ *kein* Automat für $\overline{L} := \Sigma^* \backslash L$. Zum Beispiel ist für $\mathcal{A} := (\{1,2\}, \{a\}, \{1\}, \Delta, \{1\})$ mit $\Delta := \{(1,a,1), (1,a,2)\}$ $L(\mathcal{A}) = a^*$, $L(\overline{\mathcal{A}}) = a^+ \neq \emptyset = \overline{L(\mathcal{A})}$.
Damit eine solche Konstruktion funktioniert, benötigt man deterministische Automaten.

Definition 1.4
Der Automat $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ heißt deterministisch, falls gilt:

- $|I| = 1$, d.h. $I = \{q_0\}$
- Δ ist funktional, d.h. für $q \in Q$ und $a \in \Sigma$ existiert genau ein $q' \in Q$ mit $(q, a, q') \in \Delta$

Statt der Übergangsrelation Δ verwendet man daher die Übergangsfunktion

$$\delta : Q \times \Sigma \to Q : \ (q, a) \mapsto q', \text{ wobei } (q, a, q') \in \Delta.$$

Die Funktion δ kann auf Wörter erweitert werden: $\delta(q, w) := q'$, wobei q' der eindeutig bestimmte Zustand ist, für den es einen Pfad $q \xrightarrow{w}_{\mathcal{A}} q'$ gibt. Es ist dann $L(\mathcal{A}) = \{w \mid \delta(q_0, w) \in F\}$.

□

Durch die sogenannte Potenzmengenkonstruktion kann jedem nicht-deterministischen Automaten effektiv ein deterministischer Automat zugeordnet werden, der dieselbe Sprache akzeptiert.
Dazu sei $\mathcal{A} = (Q,\Sigma,I,\Delta,F)$ ein nicht-deterministischer Automat. Wir definieren $P(\mathcal{A}) = (2^Q,\Sigma,q_0,\delta',F')$ durch

- $q_0 := I$
- $\delta'(P,a) := \{q \mid \exists p \in P: \ (p,a,q) \in \Delta\}$
- $F' := \{P \subseteq Q \mid P \cap F \neq \emptyset\}$

Man sieht leicht, daß $L(\mathcal{A}) = L(P(\mathcal{A}))$ ist .

Satz 1.5
Ist $L \in Reg_\Sigma$, so auch $\overline{L} := \Sigma^* \backslash L \ \in Reg_\Sigma$.

Beweis
Sei $L = L(\mathcal{A})$ für einen deterministischen Automaten $\mathcal{A} = (Q,\Sigma,q_0,\delta,F)$. Dann gilt:
$w \in L$ gdw $\delta(q_0,w) \in F$
$w \notin L$ gdw $\delta(q_0,w) \notin F$
$w \in \overline{L}$ gdw $\delta(q_0,w) \in Q\backslash F$
Das zeigt: $\overline{\mathcal{A}} = (Q,\Sigma,q_0,\delta,Q\backslash F)$ ist ein Automat für $\overline{L}$.

□

Zu jeder regulären Sprache L exisitert ein (bis auf Zustandsumbenennung) eindeutiger deterministischer Automat $\mathcal{A}$ mit minimaler Zustandsanzahl und $L = L(\mathcal{A})$. Dieser minimale Automat kann wie folgt effektiv konstruiert werden:
Es sei $\mathcal{A} = (Q,\Sigma,q_0,\delta,F)$ ein deterministischer Automat für L.

1. Entferne unerreichbare Zustände, d.h. Zustände $q \in Q$, für die kein $w \in \Sigma^*$ exisitiert mit $\delta(q_0,w) = q$.
2. Identifiziere äquivalente Zustände. Für $q \in Q$ sei $L_q(\mathcal{A}) = \{w \mid \delta(q,w) \in F\}$. Wir definieren $q \sim_\mathcal{A} q'$ gdw $L_q(\mathcal{A}) = L_{q'}(\mathcal{A})$. Die Relation $\sim_\mathcal{A}$ ist eine Äquivalenzreaktion. Faßt man äquivalente Zustände zusammen, so erhält man den minimalen Automaten.

Alternativ kann man auch mit Hilfe der Nerode-Rechtskongruenz den minimalen Automaten beschreiben: Für eine (beliebige) Sprache $L \in \Sigma^*$ definieren wir

$$u\rho_L v \text{ gdw } \forall w \in \Sigma^* : uw \in L \Leftrightarrow vw \in L$$

(d.h. zwei Wörter sind äquivalent, wenn sie bei Multiplikation eines beliebigen Wortes von rechts beide in L oder beide nicht in L liegen). Diese Relation ist eine Äquivalenzrelation auf Σ^* und es gilt (für alle $u,v,w \in \Sigma^*$) $u\rho_L v \Rightarrow uw\rho_L vw$ (Rechtskongruenz).
Der Satz von Nerode sagt aus, daß L genau dann regulär ist, wenn ρ_L endlichen Index hat (d.h. nur endlich viele Äquivalenzklassen besitzt).

Diese Klassen können nun als Zustände eines Automaten aufgefaßt werden. Für $u \in \Sigma^*$ bezeichne $[u] := \{v \mid u\rho_L v\}$ die ρ_L-Klasse von u.
Wir definieren $\mathcal{A}_L := (Q,\Sigma,q_0,\delta,F)$, wobei

- $Q := \{[u] \mid u \in \Sigma^*\}$
- $q_0 := [\varepsilon]$
- $\delta([u], a) := [ua]$ (repräsentantenunabhängig, da ρ_L Rechtskongruenz)
- $F := \{[u] \mid u \in L\}$

Für eine reguläre Sprache L ist $\mathcal{A}_L$ der minimale Automat für L.

1.2 Reguläre Sprachen und endliche Monoide

Ein Monoid $(M, \cdot_M, 1_M)$ besteht aus einer nichtleeren Trägermenge M, einer assoziativen binären Operation $\cdot_M$ und einem Einselement 1_M.
Assoziativ: $(x \cdot_M y) \cdot_M z = x \cdot_M (y \cdot_M z)$ für alle $x, y, z \in M$
Einselement: $x \cdot_M 1_M = 1_M \cdot_M x = x$ für alle $x \in M$
Wir werden meist M anstatt $(M, \cdot_M, 1_M)$ schreiben und häufig den Index M bei $\cdot_M$ und 1_M weglassen.
Ein Monoid heißt endlich, wenn M (Trägermenge) endlich ist.
Sind M, N Monoide, so heißt eine Abbildung $\Phi : M \to N$ Homomorphismus, falls gilt

- $\Phi(x \cdot_M y) = \Phi(x) \cdot_N \Phi(y)$
- $\Phi(1_M) = 1_N$.

Beispiel 1.6
Für ein Alphabet Σ ist die Menge Σ^* aller Wörter über Σ zusammen mit Konkatenation als Operation $\cdot$ und mit dem leeren Wort ε als Einselement ein Monoid. Σ^* heißt freies Monoid über Σ, da es die folgende universelle Eigenschaft besitzt:
Für jedes Monoid M und jede Abbildung $f : \Sigma \to M$ gibt es genau einen Homomorphismus $\Phi : \Sigma^* \to M$ mit $\Phi_{|\Sigma} = f$.

□

Homomorphismen von Σ^* in ein Monoid M können verwendet werden, um Sprachen über Σ zu definieren.

Definition 1.7
Es sei M ein Monoid, Σ ein Alphabet und $\Phi : \Sigma^* \to M$ ein Homomorphismus. Jede Teilmenge $N \subseteq M$ definiert eine Teilmenge von Σ^*:

$$\Phi^{-1}(N) := \{w \in \Sigma^* \mid \Phi(w) \in N\}.$$

Die Sprache $L \subseteq \Sigma^*$ wird von M akzeptiert, wenn es $N \subseteq M$ und $\Phi : \Sigma^* \to M$ mit $L = \Phi^{-1}(N)$ gibt.

□

Satz 1.8
Für eine Sprache $L \subseteq \Sigma^*$ sind äquivalent:

1. L wird von einem *endlichen* Monoid akzeptiert
2. L ist regulär

Beweis
1 ⇒ 2
Es sei $\Phi : \Sigma^* \to M$ mit der Eigenschaft $L = \Phi^{-1}(N)$ für $N \subseteq M$ gegeben, wobei M endlich sei. Der deterministische endliche Automat $\mathcal{A}_M := (M, \Sigma, 1_M, \delta, N)$ mit der Übergangsfunktion $\delta(m, \sigma) := m \cdot_M \Phi(\sigma)$ akzeptiert L.
Dazu zeigt man:

$$\text{Für alle } w \in \Sigma^* \text{ und alle } m \in M \text{ gilt } \delta(m, w) = m \cdot \Phi(w). \tag{1.1}$$

Beweis von (1.1) durch vollständige Induktion über $|w|$*:*
Induktionsanker: $|w| = 0 \Rightarrow w = \varepsilon$
Sei nun $m \in M$, dann gilt $\delta(m, w) = \delta(m, \varepsilon) = m \cdot \Phi(\varepsilon) = m \cdot \Phi(w)$.
Induktionsschluß: $|w| = n + 1$
Es seien $w = v \cdot \sigma$, $|v| = n$, $\sigma \in \Sigma$, $v \in \Sigma^*$. Dann gilt

$$\begin{aligned}
\delta(m, w) &= \delta(m, v\sigma) \\
&= \delta(\delta(m, v), \sigma) \\
&\overset{IV}{=} \delta(m \cdot \Phi(v), \sigma) \\
&= m \cdot \Phi(v) \cdot \Phi(\sigma) \\
&\overset{Homomorphie}{=} m \cdot \Phi(v \cdot \sigma) \\
&= m \cdot \Phi(w)
\end{aligned}$$

Damit ergibt sich $\delta(1, w) = 1 \cdot \Phi(w) = \Phi(w)$. Also gilt

$$\begin{aligned}
w \in L = \Phi^{-1}(N) \quad &\text{gdw} \quad \Phi(w) \in N \\
&\text{gdw} \quad \delta(1, w) \in N \\
&\text{gdw} \quad w \in L(\mathcal{A}_M)
\end{aligned}$$

2 ⇒ 1
Es sei $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ ein endlicher deterministischer Automat mit $L = L(\mathcal{A})$. Offenbar definiert jedes $w \in \Sigma^*$ eine Funktion $\delta_w : Q \to Q : q \mapsto \delta(q, w)$. Es sei $M = Q^Q$ die Menge der Funktionen von Q nach Q. Da Q endlich ist, ist M auch endlich. Mit Komposition von Funktionen und der Identitätsfunktion ist M ein Monoid.
Komposition: $(\delta_1 \circ \delta_2)(q) = \delta_2(\delta_1(q))$.
Man sieht leicht, daß die Abbildung

$$\Phi : \Sigma^* \to M : w \mapsto \delta_w$$

ein Homomorphismus ist, denn

$$\begin{aligned}
1.\ \Phi(\varepsilon) &= \delta_\varepsilon \\
&= Q \to Q : (q \mapsto \delta(q, \varepsilon)) \\
&= Q \to Q : (q \mapsto q) \\
&= Id_Q = 1_M \\
2.\ \Phi(v \cdot w) &= \delta_{v \cdot w}
\end{aligned}$$

$$\begin{aligned}
&= Q \to Q: \ (q \mapsto \delta(q, vw)) \\
&= Q \to Q: \ (q \mapsto \delta(\delta(q, v), w)) \\
&= Q \to Q: \ (q \mapsto \delta_v \circ \delta_w(q)) \\
&= \delta_v \circ \delta_w \\
&= \Phi(v) \circ \Phi(w)
\end{aligned}$$

Wie muß $N \subseteq M$ definiert werden? Gesucht ist ein N, das die folgende Bedingung erfüllt:

- $w \in L(\mathcal{A})$ gdw $\Phi(w) \in N$.

Dabei ist $(w \in L(\mathcal{A}) \Leftrightarrow \delta_w(q_0) = \delta(q_0, w) \in F)$ und $(\Phi(w) \in N \Leftrightarrow \delta_w \in N)$.
Wir definieren daher: $N := \{\delta_w \mid \delta_w(q_0) \in F\}$. Damit gilt $L(\mathcal{A}) = \Phi^{-1}(N)$.

□

Das Bild von Φ bezeichnet man als Übergangsmonoid von $\mathcal{A}$.

Definition 1.9
Für einen endlichen deterministischen Automaten $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ ist das Übergangsmonoid von $\mathcal{A}$ das Untermonoid $M_{\mathcal{A}} := \{\delta_w \mid w \in \Sigma^*\}$ von $M = Q^Q$.

□

Definition 1.10
Für eine reguläre Sprache L heißt der Übergangsmonoid des minimalen Automaten syntaktischer Monoid von L. Wir bezeichnen diesen mit M_L. Da der minimale Automat eindeutig durch L bestimmt ist, hängt M_L nur von L ab.

□

Man kann M_L auch direkt, d.h. ohne Bezugnahme auf den minimalen Automaten, definieren.

Definition 1.11
Für eine beliebige Sprache $L \subseteq \Sigma^*$ ist die syntaktische Kongruenz $\sim_L$ auf Σ^* definiert durch $u \sim_L v$ gdw ($\forall x, y \in \Sigma^*: \ xuy \in L$ gdw $xvy \in L$).
Man zeigt leicht, daß $\sim_L$ eine Kongruenzrelation ist, d.h. eine Äquivalenzrelation, die zusätzlich erfüllt:

$$\forall u, v, x \in \Sigma^*: \ u \sim_L v \Rightarrow \begin{cases} ux \sim_L vx \\ xu \sim_L xv \end{cases}$$

Daher kann man den Quotientenmonoid $\Sigma^* / \sim_L$ bilden:
Trägermenge: $\{[w]_{\sim_L} \mid w \in \Sigma^*\}$ (Menge der Äquivalenzklassen von $\sim_L$), wobei $[w]_{\sim_L} := \{w' \mid w \sim_L w'\}$.

Operation : $[u]_{\sim_L} \cdot [v]_{\sim_L} := [uv]_{\sim_L}$
repräsentantenunabhängig, da $\sim_L$ Kongruenzrelation ist
Beweis: Seien $u \sim_L u'$ und $v \sim_L v'$.

$$\begin{array}{ll}
\Rightarrow & \forall x,y \in \Sigma^*: \; xuy \in L \; gdw \; xu'y \in L \text{ und} \\
 & \forall x,y \in \Sigma^*: \; xvy \in L \; gdw \; xv'y \in L \\
\Rightarrow & \forall x,y \in \Sigma^*: \; xuvy \in L \; gdw \; xu'vy \in L \; gdw \; xu'v'y \in L \\
\Rightarrow & [uv]_{\sim_L} = [u'v']_{\sim_L} \\
\Rightarrow & uv \sim_l u'v'
\end{array}$$

Einselement: $[\varepsilon]_{\sim_L}$

□

Satz 1.12
Sei L regulär. $\Sigma^*/\sim_L$ ist isomorph zum syntaktischen Monoid M_L.

Beweis
Es sei $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ der minimale deterministische Automat für L. Wir definieren

$$\begin{array}{llll}
\psi: & \Sigma^*/\sim_L & \to & M_L \\
 & [u]_{\sim_L} & \mapsto & \delta_u
\end{array}$$

Zu zeigen ist:

1. Repräsentantenunabhängigkeit(= Wohldefiniertheit, d.h. ψ ist Funktion): $u \sim_L v \Rightarrow \delta_u = \delta_v$
Beweis: Es gelte $u \sim_L v$. Angenommen $\delta_u \neq \delta_v$. Dann gibt es $q \in Q$ mit
$$\delta_u(q) = \delta(q,u) = q_1 \neq q_2 = \delta(q,v) = \delta_v(q).$$
Da $\mathcal{A}$ minimal ist, ist q erreichbar, d.h. es gibt ein $x \in \Sigma^*$ mit $\delta(q_0, x) = q$. Da $u \sim_L v$ gilt, folgt für alle $y \in \Sigma^*: \; xuy \in L$ gdw $xvy \in L$.
Das heißt aber für alle y:
$$\delta(q_1, y) = \delta(q_0, xuy) \in F \text{ gdw } \delta(q_2, y) = \delta(q_0, xvy) \in F$$
d.h. $L_{q_1}(\mathcal{A}) = L_{q_2}(\mathcal{A})$ (zur Definition von L_q siehe Definition des minimalen Automaten). Da $\mathcal{A}$ minimal ist (nach Voraussetzung) folgt $q_1 = q_2$ (weil $(L_{q_1} = L_{q_2} \Rightarrow q_1 \sim_{\mathcal{A}} q_2) \Rightarrow q_1 = q_2$, weil q_1 und q_2 bei der Minimierung des Automaten zusammengefaßt werden).
Widerspruch

2. ψ ist injektiv, d.h. $\delta_u = \delta_v \Rightarrow u \sim_L v$
Beweis: Es sei $\delta_u = \delta_v$ und $xuy \in L$. Zu zeigen ist, daß aus diesen Bedingungen $xvy \in L$ folgt.
Es gilt $xuy \in L \Rightarrow \delta(q_0, xuy) \in F$. Für $q_1 := \delta(q_0, x)$ ist

$$\begin{aligned}\delta(q_0, xuy) &= \delta(q_1, uy)\\ &= \delta_y(\delta_u(q_1))\\ &= \delta_y(\delta_v(q_1)) \quad (\text{da } \delta_u = \delta_v)\\ &= \delta(q_1, vy)\\ &= \delta(q_0, xvy)\end{aligned}$$

Es folgt $\delta(q_0, xvy) \in F$ und damit $xvy \in L$.
(Dies war $xuy \in L \Rightarrow xvy \in L$, $xvy \in L \Rightarrow xuy \in L$ geht völlig analog.)

3. ψ ist surjektiv
   Beweis: δ_u ist das Bild von $[u]_{\sim_L}$
   ausführlich: zu zeigen ist $\forall y \in M_L : \exists x : \psi(x) = y$
   Beweis: Sei $y \in M_L$ beliebig aber fest. $\Rightarrow \exists w \in \Sigma^* : \delta_w = y$. Setze $x = [w]$. $\Rightarrow \psi(x) = \psi([w]) \overset{Def}{=} \delta_w = y \Rightarrow$ für alle $y \in M_L$ existiert ein x mit $\psi(x) = y$. Dies ist aber gerade die Behauptung.

4. ψ ist Homomorphismus
   Beweis:
   $$\begin{aligned}\psi([u]_{\sim_l} \cdot [v]_{\sim_L}) &= \psi([u \cdot v]_{\sim_L})\\ &= \delta_{uv} = \delta_u \circ \delta_v\\ &= \psi([u]_{\sim_L}) \circ \psi([v]_{\sim_L})\end{aligned}$$

□

Korollar 1.13
L ist regulär gdw $\sim_L$ hat endlichen Index.

Beweis
"⇒" M_L ist der Übergangsmonoid des minimalen Automaten, also endlich. Die Trägermenge von $\Sigma^*/\sim_L$ sind die $\sim_L$-Klassen. Mit Satz 1.12 gilt, daß M_L isomorph zu $\Sigma^*/\sim_L$ ist, also hat $\sim_L$ endlichen Index.
"⇐" L wird von $\Sigma^*/\sim_L$ akzeptiert, denn für

$$\Phi : \Sigma^* \to \Sigma^*/\sim_L \; : \; u \mapsto [u]_{\sim_L}$$

ist $\Phi^{-1}(\Phi(L)) = L$.
Dazu muß man zeigen:

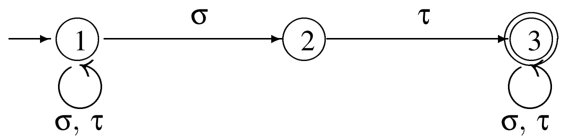
$$u \in L \text{ und } u \sim_L v \Rightarrow v \in L. \tag{1.2}$$

Hat $\sim_L$ endlichen Index, so ist $\Sigma^*/\sim_L$ endlich. Mit Satz 1.8 folgt: L regulär.

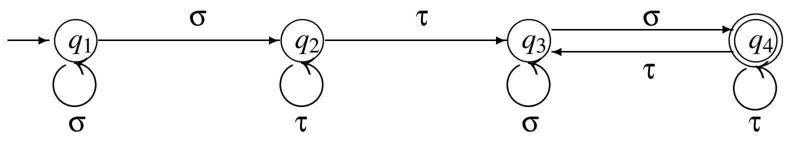
□

Beispiel 1.14 (Berechnung des syntaktischen Monoids zu einer Sprache)
$\Sigma = \{\sigma, \tau\}$, Ł $= \Sigma^*\sigma\tau\Sigma^*$

1. Bestimmung eines nicht-deterministischen endlichen Automaten für L:



2. Potenzmengenkonstruktion zur Erzeugung des äquivalenten deterministischen endlichen Automaten. Folgende Zusammenfassungen werden zur Vereinfachung vorgenommen:
   $q_1 := \{1\}$ $q_2 := \{1,2\}$ $q_3 := \{1,3\}$ $q_4 := \{1,2,3\}$



3. Minimierung: Zusammenfassung äquivalenter Zustände

   Äquivalenzrelation $\sim_{\mathcal{A}}$ auf Zuständen:

$$q \sim_{\mathcal{A}} q' \text{ gdw } L_q(\mathcal{A}) = L_{q'}(A)$$

   Wir definieren die Relation $\sim_n$ $(n \geq 0)$:

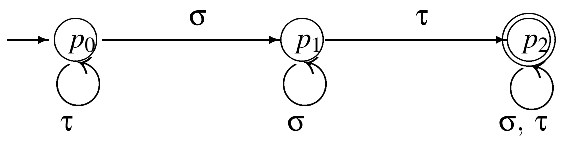
   - $q \sim_0 q'$ gdw $(q \in F$ und $q' \in F)$ oder $(q \notin F$ und $q' \notin F)$
   - $q \sim_{n+1} q'$ gdw $q \sim_n q'$ und $\forall \sigma :\ \delta(q,\sigma) \sim_n \delta(q',\sigma)$

   Da Q endlich ist, exisitiert ein k mit $\sim_k = \sim_{k+1}$. Man zeigt leicht, daß $\sim_k = \sim_{\mathcal{A}}$.

   Im Beispiel:

   - $\sim_0$ hat die Klassen $\overline{F} = \{q_1, q_2\}$ und $F = \{q_3, q_4\}$
   - $\sim_1$ hat die Klassen $\{q_1\}$, $\{q_2\}$, $\{q_3, q_4\}$
   - $\sim_2 = \sim_1$

   Der minimale Automat ist somit



4. Bildung des Übergangsmonoids
   Trägermenge: $\{\delta_u \mid u \in \Sigma^*\} \subseteq Q^Q$

$$\begin{aligned}
\delta_\varepsilon &= \frac{p_0\ p_1\ p_2}{p_0\ p_1\ p_2} \\
&\ (gelesen:\ von\ p_0\ mit\ \varepsilon\ nach\ p_0\ usw.) \\
\delta_\sigma &= \frac{p_0\ p_1\ p_2}{p_1\ p_1\ p_2} = \delta\sigma\sigma \\
\delta_\tau &= \frac{p_0\ p_1\ p_2}{p_0\ p_2\ p_2} = \delta\tau\tau \\
\delta_{\sigma\tau} &= \frac{p_0\ p_1\ p_2}{p_2\ p_2\ p_2} = \delta_{\sigma\tau} \text{ für alle } u \in \Sigma^*
\end{aligned}$$

$$\begin{aligned}
\delta_{\tau\sigma} &= \frac{p_0\ p_1\ p_2}{p_1\ p_2\ p_2} \\
\delta_{\tau\sigma\sigma} &= \frac{p_0\ p_1\ p_2}{p_1\ p_2\ p_2} = \delta_{\tau\sigma} \\
\delta_{\tau\sigma\tau} &= \frac{p_0\ p_1\ p_2}{p_2\ p_2\ p_2} = \delta_{\sigma\tau}
\end{aligned}$$

Alle Möglichkeiten wurden ausprobiert, es ergab sich nichts neues. Es ist damit

$$M_L = \{\delta_\varepsilon,\ \delta_\sigma,\ \delta_\tau,\ \delta_{\sigma\tau},\ \delta_{\tau\sigma}\}.$$

Die Multiplikationstabelle ergibt sich aus den festgestellten Identitäten.

□

Beispiel 1.15
endlicher Monoid, der nicht syntaktischer Monoid einer regulären Sprache ist

$M = \{1, a, b, c\}$

·	1	a	b	c
1	1	a	b	c
a	a	a	b	c
b	b	a	b	c
c	c	a	b	c

a,b,c sind Rechtsnullen, d.h. $\forall x \in M:\ \ x \cdot a = a,\ \ x \cdot b = b,\ \ x \cdot c = c$. Es läßt sich leicht nachrechnen, daß M Monoid ist.
Angenommen $\psi: M \to \Sigma^* / \sim_L$ ist ein Isomorphismus, wobei $L \subseteq \Sigma^*$ eine reguläre Sprache ist. Wir wissen: $u \in L \Rightarrow [u]_{\sim_L} \subseteq L$. (*Erinnerung:* $u \sim_L v$ gdw $\forall x, y:\ xuy \in L$ gdw $xvy \in L$.)
Für $m \in \{a, b, c\}$ gilt also: $\psi(m) \subseteq L$ oder $\psi(m) \subseteq \overline{L}$. Wir betrachten den Fall $\psi(a) \subseteq L$ und $\psi(b) \subseteq L$. (alle anderen Fälle, z.B. $\psi(b) \subseteq \overline{L}$, $\psi(c) \subseteq \overline{L}$, können entsprechend behandelt werden).
Behauptung: Es folgt $\psi(a) = \psi(b)$.
(Dies wäre ein Widerspruch zu ψ Isomorphismus.)
Beweis: Es sei $\psi(a) = [u]_{\sim_L}$ und $\psi(b) = [v]_{\sim_L}$. Wir zeigen $u \sim_L v$. Seien dazu $x, y \in \Sigma^*$ beliebig.

1. Fall $[y]_{\sim_L} \neq [\varepsilon]_{\sim_L}$

$$\begin{aligned}
\psi^{-1}([xuy]) &= \psi^{-1}([x] \cdot [u] \cdot [y]) \\
&= \psi^{-1}([x]) \cdot \psi^{-1}([u]) \cdot \underbrace{\psi^{-1}([y])}_{\neq 1 da [y] \neq [\varepsilon])} \\
&= \psi^{-1}([y]) \ \ (da\ [y]\ Rechtsnull\ ist) \\
&= \psi^{-1}([x]) \cdot \psi^{-1}([v]) \cdot \psi^{-1}([y]) \\
&= \psi^{-1}([xvy])
\end{aligned}$$

Daraus folgt$[xuy] = [xvy]$, da ψ Isomorphismus ist. Also gilt $xuy \sim_L xvy$. Damit gilt insbesondere $xuy \in L$ gdw $xvy \in L$ und somit ist $u \sim_L v$.

2. Fall $[y] = [\varepsilon]$

Dann ist

$$\begin{aligned}
\psi^{-1}([xuy]) &= \psi^{-1}([x]) \cdot \psi^{-1}([u]) \cdot \underbrace{\psi^{-1}([y])}_{=1} \\
&= \psi^{-1}([x]) \cdot \underbrace{\psi^{-1}([u])}_{=a,\ \psi(a)=[u]} \\
&= \psi^{-1}([x]) \cdot a \\
&= a
\end{aligned}$$

Entsprechend gilt

$$\psi^{-1}([xvy]) = b$$

Da $\psi(a) \subseteq L$ und $\psi(b) \subseteq L$ folgt $[xuy] \subseteq L$ und $[xvy] \subseteq L$ und damit $xuy \in L$ gdw $xvy \in L$. Also $u \sim_L v$.
Insgesamt folgt der Widerspruch zur Annahme, daß M syntaktisches Monoid ist.

□

Der Zusammenhang zwischen regulären Sprachen und endlichen Monoiden kann dazu verwendet werden, Teilklassen der Klasse der regulären Sprachen zu definieren.

Definition 1.16
Es sei V eine Klasse endlicher Monoide. Wir definieren die zugehörige *Klasse* $L(V)$ von Sprachen durch

$$L(V) := \{L \subseteq \Sigma^* \mid M_L \in V\}.$$

□

$L(V)$ ist eine Klasse von regulären Sprachen. Um “vernünftige” Klassen von Sprachen zu erhalten, muß man Klassen von Monoiden betrachten, die ebenfalls “vernünftige” Abschlußeigenschaften haben.

Definition 1.17
Eine nicht-leere Klasse V endlicher Monoide, die abgeschlossen ist unter Bildung von Untermonoiden, (binärer) direkter Produkte und homomorpher Bilder, heißt M-Varietät .

□

Untermonoid	$N \subseteq M$ ist Untermonoid von $(M,\cdot,1)$, falls gilt: • $1 \in N$ • $n,\ n' \in N \ \Rightarrow\ n \cdot n' \in N$
Abschluß unter Untermonoidbildung	• $M \in V$, N Untermonoid von $M \Rightarrow\ N \in V$
Direktes Produkt	$M_1 \times M_2$ • $(m_1,m_2) \times (m_1',m_2') = (m_1 \cdot m_1', m_2 \cdot m_2')$ • $1_{M_1 \times M_2} = (1_{M_1}, 1_{M_2})$ M_1, M_2 Monoide $\Rightarrow M_1 \times M_2$ Monoid
Homomorphes Bild	Ist $\Phi:\ M_1 \to M_2$ surjektiver Homomorphismus, so ist M_2 homomorphes Bild von M_1.

Beispiel 1.18
Es sei V_G die Klasse aller *endlichen* Gruppen. Man zeigt leicht, daß V_G M-Varietät ist.

- Abschluß unter homomorphen Bildern und direktem Produkt ist trivial
- Untermonoidbildung:
  Wir benötigen hier Endlichkeit. (Beispiel: $(\mathbb{Z},+,0)$ ist Gruppe, $(\mathbb{N},+,0)$ ist Untermonoid, das keine Gruppe ist.)
  Im endlichen Fall:
  Es sei G eine endliche Gruppe und N ein Untermonoid von G. Warum ist für $n \in N$ auch das Gruppeninverse $n^{-1} \in N$?
  Betrachte die Abbildung $\varphi_n:\ N \to N:\ m \mapsto m \cdot n$. Diese Abbildung ist injektiv, da G Gruppe ist: In G gilt ja $m \cdot n = m' \cdot n \ \Rightarrow\ m = m'$. Da N endlich ist, ist φ_n auch surjektiv. Wegen $1 \in N$ gibt es daher $n' \in N$ mit $\varphi_n(n') = 1$, das heißt $n' \cdot n = 1$. Damit ist $n^{-1} = n' \in N$.

□

Varietäten können auch mit Hilfe von Gleichungen definiert werden. Es sei X ein abzählbar unendliches Alphabet (von Variablen). Eine Gleichung ist von der Form

$$u = v \text{ mit } u,v \in X^* \text{ (statt } \varepsilon \text{ schreiben wir } 1).$$

Ein Monoid M <u>erfüllt</u> die Gleichung $u = v$ gdw für jeden Homomorphismus $\Phi: X^* \to M$ gilt: $\Phi(u) = \Phi(v)$.

Beispiel

$xy = yx$ $(x,y \in X)$ wird genau von den kommutativen Monoiden erfüllt.
Kommutativ: $\forall m,n \in M:\ m \cdot n = n \cdot m$.
M erfülle $xy = yx$. Seien $m,n \in M$ und sei $\Phi:\ X^* \to M$ Homomorphismus mit $m = \Phi(x)$, $n = \Phi(y)$. Dann gilt
$m \cdot n = \Phi(x) \cdot \Phi(y) = \Phi(xy) = \Phi(yx) = \Phi(y) \cdot \Phi(x) = n \cdot m.$ □

Definition 1.19
Es sei $(u_n = v_n)_{n\geq 1}$ eine Folge von Gleichungen. Die Klasse V von endlichen Monoiden wird schließlich definiert von dieser Folge, falls es für alle $M \in V$ je ein k gibt mit der Eigenschaft

$$M \text{ erfüllt } u_n = v_n \text{ für alle } n > k$$

und umgekehrt ist jedes M, das diese Eigenschaft hat, in V.

□

Satz 1.20 (Eilenberg/Schützenberg)

1. Jede M-Varietät kann durch Gleichungen schließlich definiert werden.
2. Umgekehrt ist jede durch Gleichungen schließlich definierte Klasse endlicher Monoide eine M-Varietät.

□

Beispiel 1.18 (Fortsetzung)
Mit dem obigen Satz muß eine Folge von Gleichungen existieren, die die M-Varietät V_G definiert.
Die M-Varietät V_G aller endlichen Gruppen wird schließlich definiert durch

$$(x^{\overline{n}} = 1)_{n\geq 1}$$

wobei $\overline{n} = kgV(1,\ldots,n)$.

1. Es sei $G \in V_G$. Offenbar gilt für $k = |G|$, daß $g^k = 1$ für alle $g \in G$.
   Für $n \geq k$ ist k ein Teiler von $\overline{n}$. Es folgt $g^{\overline{n}} = 1$ für alle $g \in G$ und damit erfüllt G die Gleichung $x^{\overline{n}} = 1$ für alle $n \geq k$.
2. Erfüllt G die Gleichung $x^{\overline{n}} = 1$ (für irgendein n), dann ist G eine Gruppe: für $g \in G$ ist $g^{\overline{n}-1}$ Inverses zu g.

□

Aus den Abschlußeigenschaften für M-Varietäten ergeben sich Abschlußeigenschaften für die dadurch definierten Klassen von Sprachen. Wir betrachten nur die Booleschen Operatoren (Schnitt, Vereinigung, Komplement). Dazu benötigen wir ein Lemma:

Lemma 1.21
Es sei V eine M-Varietät und $L \subseteq \Sigma^*$ eine Sprache, die von $M \in V$ akzeptiert wird. Dann ist $M_L \simeq \Sigma^*/\sim_L \ \in V$.

Beweis Da M die Sprache L akzeptiert, gibt es einen Homomorphismus $\Phi: \Sigma^* \to M$ und ein $N \subseteq M$ mit $L = \Phi^{-1}(N)$.

1. Œ sei Φ surjektiv
   Dies ist keine Einschränkung, da man sonst den Untermonoid $M' := \Phi(\Sigma^*)$ von M als Bildraum von Φ nehmen kann. Mit $N' := N \cap M'$ gilt
   - $L = \Phi^{-1}(N')$
   - $M' \in V$ (da M' Untermonoid von $M \in V$ und V Varietät)

2. Sei also $\Phi: \Sigma^* \to M$ surjektiv. Definiert man
$$u \sim_\Phi v \text{ gdw } \Phi(u) = \Phi(v)$$
so gilt: $\sim_\Phi \subseteq \sim_L$.
denn *(Beweis)* Gelte $u \sim_\Phi v$ und sei $xuy \in L$. Dann ist $\Phi(xuy) \in N$ und
$$\begin{aligned} \Phi(xvy) &= \Phi(x) \cdot \Phi(v) \cdot \Phi(y) \\ &= \Phi(x) \cdot \Phi(u) \cdot \Phi(y) \quad (\mathit{da}\ \Phi(u) = \Phi(v)\ \mathit{wegen}\ u \sim_\Phi v) \\ &= \Phi(xuy) \in N \end{aligned}$$
Also folgt $xvy \in L = \Phi^{-1}(N)$. Damit ist $\sim_\Phi \subseteq \sim_L$ gezeigt.

3. Wir definieren nun $\psi: M \to \Sigma^*/\sim_L$ durch $m \mapsto [u]_{\sim_L}$ falls $\Phi(u) = m$. Wegen 2 ($u \sim_\Phi v \Leftrightarrow \Phi(u) = \Phi(v) = m \;\Rightarrow\; u \sim_L v$) ist diese Definition repräsentantenunabhängig. Wegen 1 gibt es zu jedem m ein passendes u. Man sieht leicht, daß ψ surjektiver Homomorphismus ist. Also ist $\Sigma^*/\sim_L$ homomorphes Bild von $M \in V \;\Rightarrow \Sigma^*/\sim_L \in V$ (da eine Varietät bezüglich der Bildung homomorpher Bilder abgeschlossen ist).

□

Satz 1.22
Es sei V M-Varietät. Dann ist $L(V)$ – die Menge der Sprachen, deren syntaktischer Monoid in V liegt – abgeschlossen unter Vereinigung, Durchschnitt und Komplement.

Beweis
Wir können uns auf Schnitt und Komplement beschränken, da die Abgeschlossenheit bezüglich Vereinigung daraus folgt.

1. *Komplement:* $M_{\Sigma^* \setminus L} = M_L$ (dies gilt aufgrund der Definition von $\sim_L$)

2. *Durchschnitt:* Es seien $L, L' \in L(V)_\Sigma$, das heißt $M_L = \Sigma^*/\sim_L \in V$ und $M_{L'} = \Sigma^*/\sim_{L'} \in V$. Es seien weiter
$$\Phi: \Sigma^* \to \Sigma^*/\sim_L: \; u \mapsto [u]_{\sim_L}$$
und
$$\Phi': \Sigma^* \to \Sigma^*/\sim_{L'}: \; u \mapsto [u]_{\sim_{L'}}$$
die kanonischen Homomorphismen.
Wir wissen (vergleiche den Beweis zu Korollar 1.13), daß für $N = \Phi(L)$ und $N' = \Phi'(L')$ gilt $L = \Phi^{-1}(N)$ und $L' = \Phi^{-1}(N')$. Wir definieren
$$\begin{aligned} \psi: \Sigma^* &\to \Sigma^*/\sim_L \times \Sigma^*/\sim_{L'} \in V \quad (\mathit{da}\ V\ \mathit{abgeschlossen\ ist\ bzgl.}\ \times) \\ u &\mapsto (\Phi(u), \Phi'(u)) \end{aligned}$$
Dann ist
$$\begin{aligned} \psi^{-1}(N \times N') &= \{u \mid \Phi(u) \in N\ \mathit{und}\ \Phi'(u) \in N'\} \\ &= \Phi^{-1}(N) \cap \Phi'^{-1}(N') \\ &= L \cap L' \end{aligned}$$
Also akzeptiert $M = \Sigma^*/\sim_L \times \Sigma^*/\sim_{L'}$ die Sprache $L \cap L'$. Wegen $M \in V$ folgt mit Lemma 1.21, daß $M_{L \cap L'} \in V$.

□

Es ist manchmal günstiger, Halbgruppen statt Monoide zu betrachten. Eine Halbgruppe hat wie ein Monoid eine binäre assoziative Operation, aber im allgemeinen kein Einselement. Die bisher betrachteten Begriffe und Ergebnisse lassen sich auf Halbgruppen übertragen:

- syntaktische Halbgruppe: Die syntaktische Kongruenz $\sim_L$ ist auch eine Kongruenz auf der freien Halbgruppe $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$. Die syntaktische Halbgruppe S_L der Sprache L ist $\Sigma^+ / \sim_L$.
  Alternativ bezeichnet man mit $S_{\mathcal{A}}$ die Übergangshalbgruppe $\{\delta_u \mid u \in \Sigma^+\}$ des minimalen Automaten $\mathcal{A}$ für L.

- S-Varietät ("S" steht für "semigroup"): Abgeschlossen unter Unterhalbgruppenbildung, direktem Produkt und homomorphen Bildern.
  Beachte: Unterhalbgruppen von Monoiden müssen keine Untermonoide sein.
  S-Varietäten können ebenfalls schließlich durch Gleichungen definiert werden. Diese Gleichungen sind von der Form $u = v,\ u, v \in X^+$, das heißt Satz 1.20 gilt in einer Halbgruppenvariante 1.20S.
  S-Varietäten liefern eine feinere Aufteilung in Klassen. Zum Beispiel wird die Gleichung $xy = x$ nur von dem trivialen Monoid erfüllt ($1 \cdot m = 1 \Rightarrow m = 1$). Es gibt aber nicht-triviale Halbgruppen, die $xy = x$ erfüllen.

- zugehörige Sprachklasse: Für eine S-Varietät V ist
  $$L(V)_\Sigma := \{L \subseteq \Sigma^* \mid S_L \in V\}.$$
  Lemma 1.21 und Satz 1.22 gelten ebenfalls in Halbgruppenvarianten 1.21S und 1.22S.

1.3 Reguläre Sprachen und logische Formeln

Wir verwenden im folgenden eine Logik mit Gleichheit, das heißt "=" ist ein binäres Prädikat, das als Identität interpretiert wird. Als nicht-logische Symbole verwenden wir eine binäre Relation "<", sowie endlich viele einstellige Relationen $P_1, \ldots, P_k$.
Wir betrachten endliche Interpretationen, in denen "<" als totale Ordnung definiert ist. Solche Interpretationen lassen sich als Wörter über dem Alphabet $\Sigma = \{0,1\}^k$ auffassen.

- $dom(I) = \{d_1, \ldots, d_n\}$, wobei $d_1 < d_2 < \ldots < d_n$
- Sei $\sigma_i = (b_{i_1}, \ldots, b_{i_k})$, wobei
  $$b_{i_j} = \begin{cases} 1 & d_i \in P_j^I \\ 0 & d_i \notin P_j^I \end{cases}$$
  dann entspricht die Interpretation I dem Wort $\sigma_1 \ldots \sigma_n$. Umgekehrt liefert jedes Wort über Σ eine Interpretation.

Beispiel 1.23
$k = 1$, das heißt $\Sigma = \{0,1\}$. In diesem Fall entspricht das Wort 010 einer Interpretation mit

- $dom(I) = \{d_1, d_2, d_3\}$
- $d_1 < d_2 < d_3$
- $P_1^I = \{d_2\}$ (die zweite Stelle im Wort ist die einzige "1")

□

Anstelle von Interpretationen werden wir im folgenden stets Wörter verwenden. Es macht daher Sinn, zu sagen, daß ein Wort $w \in \Sigma^+$ Modell einer Formel φ ist ($w \models \varphi$).

Definition 1.24
Es sei φ eine geschlossene Formel (d.h. φ enthält keine freien Variablen) der Logik erster Stufe, die als nicht-logische Symbole nur $<$ und $P_1, \dots, P_k$ verwendet. Das Alphabet Σ sei $\{0,1\}^k$.
Dann definiert φ die *Sprache*

$$L(\varphi) = \{w \in \Sigma^+ \mid w \models \varphi\}.$$

□

Da Interpretationen stets nichtleeren Bereich haben, entspricht das leere Wort keiner Interpretation. Das ist keine wesentliche Einschränkung, zum Beispiel ist L regulär gdw $L \backslash \{\varepsilon\}$ regulär ist.

Beispiel 1.25
$k = 1$, das heißt $\Sigma = \{0,1\}$. Die Sprache 11^*0^* wird definiert durch die Formel

$$\exists x: \; (P_1(x) \;\; \wedge \;\; \forall y: \; (y \leq x \rightarrow P_1(y)) \;\; \wedge \;\; \forall z: \; (x < z \rightarrow \neg P_1(z)))$$

□

Satz 1.26
Boolesche Operatoren in Formeln entsprechen Booleschen Operatoren auf Sprachen, das heißt

- $L(\neg\varphi) = \Sigma^+ \backslash L(\varphi)$,
- $L(\varphi \vee \psi) = L(\varphi) \cup L(\psi)$,
- $L(\varphi \wedge \psi) = L(\varphi) \cap L(\psi)$.

□

Bei der Beschreibung von Sprachen durch logische Formeln sind die folgenden Abkürzungen hilfreich:

- Für jedes $\sigma \in \Sigma$ kann man mit einer Formel mit einer freien Variablen – $Q_\sigma(x)$ – ausdrücken, daß an der Stelle x das Symbol σ steht.

  Zum Beispiel für $k = 2$, das heißt $\Sigma = \{(0,0), (0,1), (1,0), (1,1)\}$ gilt

$$\begin{array}{lll} Q_{(1,1)}(x) & \textit{entspricht} & P_1(x) \wedge P_2(x) \\ Q_{(1,0)}(x) & \textit{entspricht} & P_1(x) \wedge \neg P_2(x) \end{array}$$

- Durch eine Formel $Min(x)$ kann man ausdrücken, daß x der Wortanfang ist:

$$\neg\exists y:\ y < x$$

- Entsprechend kann man $Max(x)$ als das Wortende verwenden:

$$\forall y:\ y \leq x$$

- Die auf x folgende Stelle im Wort wird ausgedrückt durch

$$Succ(x,y):\ (x < y \wedge \neg\exists z(x < z \wedge z < y))$$

- Häufig verwendet man statt einer Formel "$Succ(x,y)$" eine Funktion "s":
  Für x ist $s(x)$ die auf x folgende Position im Wort, falls diese existiert; sonst sei $s(x) = x$.

$$s(x) = y \mathrel{\hat{=}} Succ(x,y) \vee (Max(x) \wedge x = y)$$

  Entsprechend kann man anstelle von Formeln $Max(x)$ und $Min(x)$ Konstanten *max* und *min* verwenden.

- Wie den Nachfolger kann man auch den Vorgänger durch eine logische Formel $Pred(x,y)$ oder eine Funktion $p(x)$ beschreiben.

Beispiel 1.27
Die reguläre Sprache $a(ba)^*$ kann beschrieben werden durch
$Q_a(min) \wedge$
$\forall x\,\forall y:\ (Q_a(x) \wedge Succ(x,y) \rightarrow Q_b(y)) \wedge$
$\forall x\,\forall y:\ (Q_b(x) \wedge Succ(x,y) \rightarrow Q_a(y)) \wedge$
$Q_a(max)$

□

Welche Sprachen lassen sich durch Formeln der Logik erster Stufe beschreiben? Wir werden später sehen: Nur *reguläre Sprachen* sind so beschreibbar.
Kann man alle regulären Sprachen $L \subseteq \Sigma^+$ durch eine Formel der Logik erster Stufe beschreiben?

Beispiel 1.28
Die reguläre Sprache $L = a(aa)^*$ kann *nicht* durch eine Formel der Logik erster Stufe beschrieben werden. Wir werden später sehen, wie man das zeigt.

□

Erlaubt man Quantifizierungen über einstellige Prädikate, so kann man zum Beispiel $L = a(aa)^*$ durch eine Formel definieren. Als Variablen für einstellige Prädikate verwenden wir große Buchstaben (X, Y, $Z, \ldots$) und als Variablen für Objekte (Positionen im Wort) wie bisher kleine Buchstaben (x, y, $z, \ldots$).

Die Sprache $L = a(aa)^*$ kann nun beschrieben werden durch

$$\begin{array}{ll}
\exists X \exists Y & (X(min) \wedge \\
& \forall x\, \forall y: \; (X(x) \wedge Succ(x,y) \to Y(y)) \wedge \\
& \forall x\, \forall y: \; (Y(x) \wedge Succ(x,y) \to X(y)) \wedge \\
& X(max) \wedge \\
& \forall x: \; (X(x) \leftrightarrow \neg Y(x)) \wedge \\
& \forall x: \; Q_a(x))
\end{array}$$

Die Idee dabei ist, daß X für die ungeraden Positionen und Y für die geraden Positionen steht. $X(max)$ besagt, daß die letzte Position ungerade ist, das heißt daß das Wort ungerade Länge hat.

Wir werden später sehen, daß man durch Formeln, die derartige Quantifizierungen zweiter Stufe zulassen, genau die regulären Sprachen beschreiben kann.
In Kapitel 3 werden wir die Klasse von Sprachen untersuchen, die genau durch Formeln der Logik erster Stufe definiert werden können.
Wir betrachten zunächst in Kapitel 2 eine kleinere Klasse von Sprachen: die durch quantorenfreie Formeln beschreibbaren Sprachen.

Kapitel 2

Verallgemeinert-definite Sprachen und quantorenfreie Formeln

Wir führen zunächst die Sprachklasse direkt ein und zeigen dann, wie man sie mit Hilfe von Halbgruppen und Formeln charakterisieren kann.

2.1 Verallgemeinert-definite Sprachen

Informell sind diese Sprachen dadurch charakterisiert, daß Zugehörigkeit eines Wortes zu der Sprache nur von den k ersten und den k letzten Buchstaben abhängt.
Definition 2.1
Die Klasse B_0 der verallgemeinert-definiten Sprachen ist wie folgt definiert:
$L \subseteq \Sigma^*$ gehört zu $(B_0)_\Sigma$ gdw es gibt ein $k \geq 0$, so daß für alle $w \in L$ gilt: Ist $w = uv = v'u'$ mit $|u| = k = |u'|$, so ist

$$u\Sigma^* \cap \Sigma^* u' \subseteq L \text{ gdw } w \in L.$$

□

Ob ein Wort w' zu L gehört, hängt also nur von den ersten und letzten k Buchstaben ab.

Lemma 2.2
Für ein endliches Alphabet Σ ist $(B_0)_\Sigma$ der Boolesche Abschluß der Sprachen

$$\{u\Sigma^* \mid u \in \Sigma^*\} \cup \{\Sigma^* u' \mid u' \in \Sigma^*\}$$

Beweis

1. Es sei $L \in (B_0)_\Sigma$ und k die in Definition 2.1 geforderte Zahl.

1. Fall $k = 0$
In diesem Fall ist $L = \Sigma^*$ oder $L = \emptyset$.
- für $u = \varepsilon$ ist $u\Sigma^* = \Sigma^*$
- für beliebiges u ist $u\Sigma^* \cap \overline{u\Sigma^*} = \emptyset$

2. Fall $k > 0$

In diesem Fall gilt

$$L = \left(\bigcup_{\substack{|u|=k=|u'| \\ u\Sigma^* \cap \Sigma^* u' \subseteq L}} u\Sigma^* \cap \Sigma^* u' \right) \bigcup \{v \in L \mid |v| < k\}$$

denn *(Beweis)*

"$\supseteq$" trivial

"$\subseteq$" Es sei $w \in L$. Der Fall $|w| < k$ ist trivial. Sei also $|w| \geq k$. Damit gibt es u, u', v, v' mit $w = uv = v'u'$ und $|u| = k = |u'|$. Aus $w \in L$ und der Definition der verallgemeinert-definiten Sprachen folgt

$$\underbrace{u\Sigma^* \cap \Sigma^* u'}_{w\in} \subseteq L.$$

Es bleibt zu zeigen, daß $\{v \in L \mid |v| < k\}$ im Booleschen Abschluß liegt. Es gilt aber

$$\begin{aligned} \{v\} &= \Sigma^* \backslash v\Sigma\Sigma^* \\ &= v\Sigma^* \backslash \bigcup_{\sigma\in\Sigma} v\sigma\Sigma^* \end{aligned}$$

2. (a) Wir wollen zeigen, daß $w\Sigma^* \in (B_0)_\Sigma$
Wähle dazu $k = |w|$. Offenbar ist

$$w = uv = v'u' \in L = w\Sigma^* \text{ (mit } |u| = k = |u'|) \text{ gdw } u = w.$$

Es ist daher $u\Sigma^* \cap \Sigma^* u' \subseteq L = w\Sigma^*$ falls $w = uv \in L$.

(b) $\Sigma^* w$ kann entsprechend behandelt werden

(c) Abschluß von B_0 unter Vereinigung
Es seien $L_1 \in B_0, L_2 \in B_0$ mit Zahlen k_1, k_2. Für $L_1 \cup L_2$ kann man $k := max\{k_1, k_2\}$ verwenden. *Beachte:* Ist $u = u_1 u_2$, so ist $u_1\Sigma^* \supseteq u\Sigma^*$. Aus $u_1\Sigma^* \subseteq L$ folgt also $u\Sigma^* \subseteq L$.
Damit zeigt man leicht, daß k das Gewünschte leistet.

(d) Abschluß von B_0 unter Komplement
Es sei $L \in B_0$ und k die entsprechende Zahl. Dieses k kann auch für $\overline{L}$ verwendet werden. Es sei $w = uv = v'u' \in \overline{L}$, wobei $|u| = k = |u'|$ gelte.
Wäre $u\Sigma^* \cap \Sigma^* u' \not\subseteq \overline{L}$, so gäbe es $w' \in u\Sigma^* \cap \Sigma^* u'$ mit $w' \in L$. Daraus würde $u\Sigma^* \cap \Sigma^* u' \subseteq L$ folgen, was im Widerspruch zu $w \in \overline{L}$ stünde.
Damit wissen wir, daß $\overline{L}$ verallgemeinert-definit ist mit der Zahl k.

□

2.2 Die zugehörige *S*-Varietät

Um diese <u>*S*-Varietät</u> zu definieren, benötigen wir den Begriff des idempotenten Elements in einer Halbgruppe.

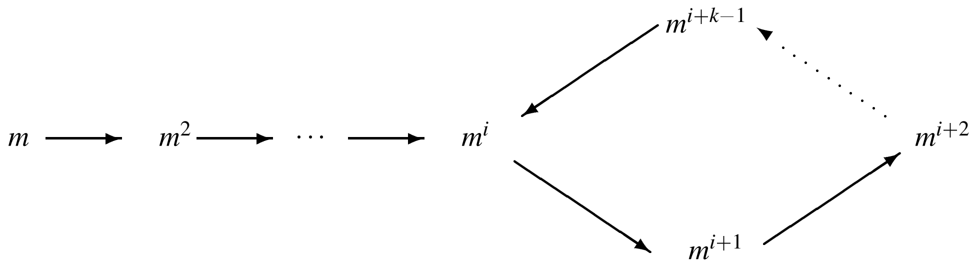
Definition 2.3
Es sei S eine Halbgruppe. Ein Element $e \in S$ heißt idempotent gdw $e \cdot e = e$.

$\square$

Offenbar ist ein Einselement immer Idempotent. Es kann aber noch weitere Idempotente geben.

Satz 2.4
Es sei S eine *endliche* Halbgruppe und $m \in S$. Dann enthält die Menge $\{m, m^2, m^3, \ldots\}$ ein Idempotentes.

Beweis
Da S endlich ist, existieren $i, k \geq 1$ mit $m^i = m^{i+k}$.



Es existiert ein l mit den Eigenschaften

- $i \leq l < i+k$, das heißt $l = i + p$ mit $p < k$
- $l \equiv 0 \; mod \; k$, das heißt $\exists \; l' : \; l = kl'$.

Behauptung: m^l ist idempotent, **denn**

$$\begin{aligned}
m^l \cdot m^l &= m^{i+p} \cdot m^{k \cdot l'} \\
&= m^i \cdot m^{k \cdot l'} \cdot m^p \\
&= m^i \cdot m^k \cdot m^{k \cdot (l'-1)} \cdot m^p \\
&= m^{i+k} \cdot m^{k \cdot (l'-1)} \cdot m^p \\
&= m^i \cdot m^{k \cdot (l'-1)} \cdot m^p \\
&\ldots \\
&= m^i \cdot m^p \\
&= m^{i+p} \\
&= m^l
\end{aligned}$$

$\square$

Beachte: Sind $m^i, \; m^{i+1}, \; \cdots, \; m^{i+k-1}$ alle verschieden (d.h. ist k minimal gewählt), so ist $\{m^i, \; m^{i+1}, \; \ldots, \; m^{i+k-1}\}$ eine zyklische Gruppe mit Einselement m^l.

Korollar 2.5
Es sei S eine endliche Halbgruppe, $|S| \leq n$. Dann gilt für $\overline{n} = kgV(1,\ldots,n)$ und alle $m \in S:\ m^{\overline{n}}$ ist idempotent.

Beweis
Offenbar findet man im Beweis des vorigen Satzes $i,\ k$ mit $i+k-1 \leq n$. Es ist daher $l \leq n$, wobei l wie im Beweis des Satzes gewählt wird. Somit gilt $l \mid \overline{n}$, das heißt es existiert ein l mit $\overline{n} = l \cdot l'$. Damit ist $m^{\overline{n}} = \underbrace{(m^l)}_{idempotent}{}^{l'} = m^l$. □

Definition 2.6
Die Klasse $\hat{\boldsymbol{D}}$ besteht aus allen endlichen Halbgruppen, für die gilt:

- Für alle Idempotenten $e \in S$ ($S \in \hat{\boldsymbol{D}}$) ist $eSe = e$, d.h. $\{e \cdot s \cdot e | s \in S\} = \{e\}$.

□

Satz 2.7
$\hat{\boldsymbol{D}}$ ist eine S-Varietät, die schließlich definiert ist durch

$$x^{\overline{n}} y x^{\overline{n}} = x^{\overline{n}} \quad (n \geq 1,\ \overline{n} = kgV(1,\ldots,n)) \tag{2.1}$$

Beweis
Es genügt zu zeigen, daß $\hat{\boldsymbol{D}}$ schließlich von (2.1) definiert wird.

1. Es sei $S \in \hat{\boldsymbol{D}}$. Mit Korollar 2.5 ist für $n \geq |S|$ und $m \in S$ stets $m^{\overline{n}}$ idempotent. Aus der Definition von $\hat{\boldsymbol{D}}$ folgt damit für alle $m' \in S:\ m^{\overline{n}} \cdot m' \cdot m^{\overline{n}} = m^{\overline{n}}$. Also erfüllt S (2.1) schließlich.

2. Gilt in S die Gleichung (2.1) für ein n, so gilt für alle Idempotenten $e \in S$ und alle $m \in S$:
$$eme = e^{\overline{n}} m e^{\overline{n}} \overset{(2.1)}{=} e^{\overline{n}} = e,$$
also ist $eSe = e$.

□

Lemma 2.8
Es sei $S \in \hat{\boldsymbol{D}}$. Dann gilt

1. Für $n > |S|$ und $m_1,\ldots,m_n \in S$ ist $m_1 \cdot m_2 \cdot \ldots \cdot m_n$ idempotent.

2. Sind $e, f \in S$ idempotent und ist $m \in S$, so gilt $emf = ef$.

Beweis

1. Betrachte $m_1,\ m_1m_2,\ m_1m_2m_3,\ \ldots,\ m_1m_2m_3\ldots m_n$. Wegen $n > |S|$ gibt es i und j mit $i < j$ und
$$m_1\ldots m_i = m_1\ldots m_i m_{i+1}\ldots m_j. \tag{2.2}$$
Mit Satz 2.4 existiert ein l mit $(m_{i+1}\ldots m_j)^l =: e$ idempotent.
$$\begin{aligned} m_1\ldots m_n &= m_1\ldots m_i m_{i+1}\ldots m_j m_{j+1}\ldots m_n \\ &\overset{(2.2)}{=} m_1\ldots m_i (m_{i+1}\ldots m_j)^l m_{j+1}\ldots m_n \\ &\overset{eSe=e}{=} m_1\ldots m_i \underbrace{(m_{i+1}\ldots m_j)^l}_{e} \underbrace{m_{j+1}\ldots m_n m_1\ldots m_i}_{s} \underbrace{(m_{i+1}\ldots m_j)^l}_{e} m_{j+1}\ldots m_n \\ &\overset{(2.2)}{=} m_1\ldots m_i m_{i+1}\ldots m_j m_{j+1}\ldots m_n m_1\ldots m_i m_{i+1}\ldots m_j m_{j+1}\ldots m_n \\ &= m_1\ldots m_n m_1\ldots m_n \end{aligned}$$

2. $$emf = \underbrace{efe}_{\in eSe} mf = e\underbrace{femf}_{\in fSf} = ef$$

□

Satz 2.9
$L(\hat{\boldsymbol{D}}) = B_0$, d.h. für $L \subseteq \Sigma^*:\ S_L \in \hat{\boldsymbol{D}}$ gdw $L \in (B_0)_\Sigma$.

Beweis

"$\supseteq$" Es sei $L \in (B_0)_\Sigma$, das heißt mit Lemma 2.2 ist L im Booleschen Abschluß der Sprachen
$$\{u\Sigma^* | u \in \Sigma^*\} \cup \{\Sigma^* u' | u' \in \Sigma^*\}$$
Nach Satz 1.22S ist $L(\hat{\boldsymbol{D}})$ unter Booleschen Operationen abgeschlossen. Es genügt also zu zeigen, daß $u\Sigma^*$ und Σ^*u' in $L(\hat{\boldsymbol{D}})$ liegen. Wir beschränken uns auf den Fall $L = u\Sigma^*$. Es sei $n = |u|$.
1. Fall $n = 0$, d.h. $L = \Sigma^*$
Dann ist $\sim_L = \Sigma^* \times \Sigma^*$ und somit besteht $S_L = \Sigma^+/\sim_L$ aus einer einzigen Klasse, die offenbar idempotent ist. Es folgt $S_L \in \hat{\boldsymbol{D}}$.
2. Fall $n > 0$
Behauptung: Ist $|w| \geq n$, so gilt für alle $v \in \Sigma^*:\ wv \sim_L w$
denn *(Beweis)* Für alle $x, y \in \Sigma^*$ gilt
$$\begin{aligned} xwvy \in L = u\Sigma^* &\Leftrightarrow xwvy \text{ beginnt mit } u \\ (|w| \geq |u|) &\Leftrightarrow xw \text{ beginnt mit } u \\ &\Leftrightarrow xwy \text{ beginnt mit } u \\ &\Leftrightarrow xwy \in L = u\Sigma^* \end{aligned}$$

Endes des Beweises der Behauptung

Es sei nun $e = [x]_{\sim_L} \in S_L = \Sigma^+/\sim_L$ idempotent. Wegen $|x| \geq 1$ ist $|x^n| \geq n$ und damit erfüllt $w = x^n$ die Voraussetzung der Behauptung.
Daher gilt für alle $m = [y]_{\sim_L} \in S_L$:

$$\begin{aligned} eme &= e^n me \\ &= [x^n]_{\sim_L}[y]_{\sim_l} e \\ &= [x^n y]_{\sim_L} e \\ (mit\ Beh.\ ist\ x^n y \sim_L x^n) &= [x^n]_{\sim_L} e \\ &= e^n e \\ (e\ ist\ idempotent) &= e \end{aligned}$$

Also gilt $S_L \in \hat{\boldsymbol{D}}$.

"$\subseteq$" Es sei $S_L \in \hat{\boldsymbol{D}}$, $n = |S_L| + 1$.
Mit Lemma 2.8(1) gilt für alle Wörter u der Länge n, daß $[u]_{\sim_L}$ idempotent ist:

$$u = \sigma_1 \ldots \sigma_n \qquad [u]_{\sim_L} = [\sigma_1]_{\sim_L} \ldots [\sigma_n]_{\sim_L}$$

Es sei nun $x \in L$ mit $|x| \geq 2n$. Dann ist $x = uvu'$ für $|u| = n = |u'|$. Außerdem gilt, daß $[u]_{\sim_L}$ und $[u']_{\sim_L}$ idempotent sind. Mit Lemma 2.8(2) gilt für alle Wörter w:

$$\begin{aligned} [u]_{\sim_L} \cdot [w]_{\sim_L} \cdot [u']_{\sim_L} &= [u]_{\sim_L} \cdot [u']_{\sim_L} \\ &= [u]_{\sim_L} \cdot [v]_{\sim_L} \cdot [u']_{\sim_L} \end{aligned}$$

Also gilt $uwu' \sim_L \underbrace{uvu'}_{\in L\ da\ x \in L}$ und damit auch $uwu' \in L$.
Das zeigt: $u\Sigma^* u' \subseteq L$.
Damit ist

$$L = \bigcup_{\substack{u\Sigma^* u' \subseteq L \\ |u|=n=|u'|}} u\Sigma^* u' \ \cup \ \underbrace{\{w | w \in L\ und\ |w| < 2n\}}_{\text{endliche Vereinigung von Einermengen } \{w\}} .$$

Wir wissen bereits, daß Einermengen in B_0 sind:

$$\{w\} = w\Sigma^* \setminus \cup_{\sigma \in \Sigma} w\sigma\Sigma^*.$$

Es bleibt zu zeigen: $u\Sigma^* u' \in B_0$. Dazu:

$$u\Sigma^* u' = \underbrace{u\Sigma^*}_{\in B_0} \cap \underbrace{\Sigma^* u'}_{\in B_0} \setminus \underbrace{\{w \mid |w| < 2n\}}_{\in B_0}.$$

□

Korollar 2.10
Es sei L eine reguläre Sprache (gegeben durch einen regulären Audruck, einen endlichen Automaten etc.). Dann kann man effektiv entscheiden, ob $L \in B_0$ gilt oder nicht.

Beweis
Zu L kann man effektiv die syntaktische Halbgruppe S_L konstruieren. Für diese (endliche) Halbgruppe kann man überprüfen, ob für alle Idempotenten e gilt, daß $eS_L e = e$ oder nicht.
□

2.3 Quantorenfreie Formeln

Als nicht-logische Symbole verwenden wir zusätzlich zu der binären Relation $<$ und den unären Relationen $P_1, P_2, \ldots, P_k$ die Konstanten *min* und *max* sowie die Funktionssymbole s und p. Die Interpretation dieser Symbole sei wie in Abschnitt 1.3 fixiert.
Es sei $\Sigma = \{0,1\}^k$. Für $\sigma \in \Sigma$ verwenden wir die Formeln $Q_\sigma(x)$ als Abkürzung.
Beachte: Im Abschnitt 1.3 haben wir gesehen, daß man die zusätzlichen Symbole *min*, *max*, s und p auch durch Formeln ausdrücken kann, die nur $=$, $<$ und $P_1, \ldots, P_k$ enthalten. Diese Formeln enthalten aber Quantoren.

Satz 2.11
Für $L \subseteq \Sigma^*$ sind äquivalent:

1. $L \in B_0$
2. $L \backslash \{\varepsilon\} = L(\varphi)$ für eine quantorenfreie geschlossene Formel φ, die nur die nicht-logischen Symbole $<$, $P_1 \ldots P_k$, *min*, *max*, s und p enthält.

Beachte: Da $\{\varepsilon\} \in B_0$ gilt, ist $L \in B_0$ gdw $L\backslash\{\varepsilon\} \in B_0$.

Beweis
"1 $\Rightarrow$ 2"
Da Disjunktion der Vereinigung und Negation dem Komplement entspricht, genügt es mit Lemma 2.2, Sprachen $u\Sigma^*$ und $\Sigma^* u'$ zu betrachten.
Zu $u\Sigma^*$ konstruieren wir die *zugehörige Formel* wie folgt:
1. Fall $u = \sigma_1 \ldots \sigma_l$ mit $l \geq 1$

$$Q_{\sigma_1}(min) \wedge Q_{\sigma_2}(s(min)) \wedge \ldots \wedge Q_{\sigma_l}(s^{l-1}(min)) \wedge s^{l-2}(min) < max$$

2. Fall $u = \varepsilon$, das heißt $L\backslash\{\varepsilon\} = \Sigma^+$
Man kann eine beliebige Formel nehmen, die immer wahr ist, zum Beispiel $min = min$.
"1 $\Leftarrow$ 2"
Es sei φ eine geschlossene, quantorenfreie Formel über den zur Verfügung stehenden Symbolen. Offenbar enthält φ keine Variablen. Terme sind aufgebaut aus *min*, *max*, p, s. Diese können normalisiert werden zu:

$$\left.\begin{array}{ll} s^n(min) & n \geq 0 \\ p^n(max) & n \geq 0 \end{array}\right\} \text{ es werden keine weiteren Formeln benötigt}$$

Dazu verwendet man $s(max) = max$, $p(min) = min$, $s(p(d)) = d$ und $p(s(d)) = d$ (außer an den Extremalpunkten *min* und *max*).
Die Formel φ ist daher Œ Boolesche Kombination von atomaren Formeln der Formen

$$P_i(t),\ \ t = t',\ \ t < t',\ \ t \leq t',\ \ \neg P_i(t)$$

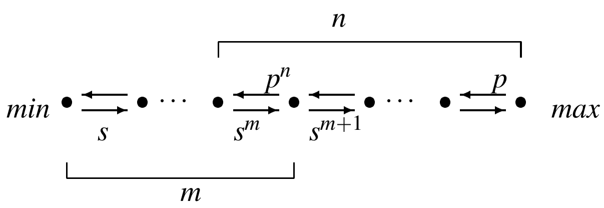
wobei t und t' normalisiert sind.
Beachte: $\neg(t = t') \mathrel{\hat{=}} (t < t') \vee (t' < t)$ und $\neg(t < t') \mathrel{\hat{=}} t' \leq t$.
Es genügt für jede dieser atomaren Formeln zu zeigen, daß die akzeptierte Sprache in B_0 ist.

- $P_i(s^n(min))$ wird erfüllt von

- Wörtern der Länge $\leq n$ (mit bestimmten Eigenschaften), d.h. von endlichen Mengen von Wörtern. Endliche Wortmengen sind aber immer in B_0, da sie Vereinigung von Einermengen sind
- Wörter der Länge $> n$, die als (n+1)-tes Symbol ein $\sigma \in \{0,1\}^k$ haben, für das die i-te Komponente 1 ist:

$$\bigcup_{\substack{|u|=n \\ \sigma \text{ hat } i\text{-te Komponente } 1}} u\sigma\Sigma^* \in B_0$$

- entsprechend können $P_i(p^n(max))$ und die negierten Formeln $\neg P_i(p^n(max))$, $\neg P_i(s^n(min))$ behandelt werden.
- Formeln $t = t'$, $t < t'$ und $t \leq t'$ sind entweder von allen Wörtern erfüllt oder sie schränken die Wortlänge ein:
  - $s^n(min) = s^m(min)$ für $n < m$ sagt, daß das Wort Länge $\leq n+1$ hat, d.h. die Formel kann nur von einer endlichen Wortmenge erfüllt werden. Diese sind aber in B_0.
  - $s^n(min) < s^m(min)$ für $n < m$ sagt, daß das Wort Länge $> n+1$ hat. Die Komplementmenge (Wörter der Länge $\leq n+1$) ist endlich, also in B_0. Da B_0 bezüglich den Booleschen Operationen abgeschlossen ist, liegt also auch die durch obige Formel beschriebene Menge in B_0.
  - $p^n(max) \leq s^m(min)$ ergibt eine Wortlänge $\leq n+m+1$ (vergleiche Skizze). Damit hat man wieder eine endliche Wortmenge, die in B_0 liegt.



□

Kapitel 3

Sternfreie Sprachen

Die sternfreien Sprachen sind genau die Sprachen, welche von den Formeln der Logik erster Stufe akzeptiert werden.

3.1 Die Sprachkasse

Die Klasse der regulären Sprachen erhält man, indem man mit *endlichen* Sprachen beginnt und dann den Abschluß unter

- Vereinigung ($L_1 \cup L_2$)
- Konkatenation ($L_1 \cdot L_2$)
- Stern (L^*)

bildet. Würde man hier den Stern-Operator verbieten, so könnte man keine unendlichen Mengen erzeugen. Reguläre Sprachen sind außerdem unter Durchschnitt und Komplement abgeschlossen.

Definition 3.1
Für ein endliches Alphabet Σ ist die Klasse der sternfreien Sprachen $(SF)_\Sigma$ über Σ die kleinste Klasse mit

- alle endlichen Sprachen über Σ sind in $(SF)_\Sigma$
- sind $L_1,\ L_2,\ L \in (SF)_\Sigma$, so auch $L_1 \cup L_2$, $L_1 \cap L_2$, $\Sigma^* \backslash L$, $L_1 \cdot L_2$ (d.h. Vereinigung, Schnitt, Komplement und Konkatenation).

$\square$

Beispiel 3.2

1. $\Sigma^* \in (SF)_\Sigma$, denn $\Sigma^* = \overline{\emptyset}$
2. Ist $\Delta \subseteq \Sigma$ und sind Δ und Σ endlich, so ist $\Delta^* \in (SF)_\Sigma$, denn

$$\begin{aligned} \Delta^* &= \Sigma^* \backslash \Sigma^* \cdot (\Sigma \backslash \Delta) \cdot \Sigma^* \\ &= \overline{\overline{\emptyset} \cdot (\Sigma \cap \overline{\Delta}) \cdot \overline{\emptyset}} \end{aligned}$$

3. Für $\Sigma = \{a,b\}$ ist $a(ba)^* \in (SF)_\Sigma$:

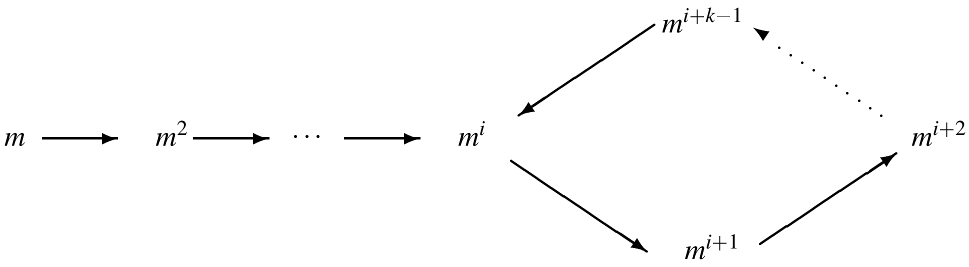
$$\begin{aligned} a(ba)^* &= a\cdot(\Sigma^*\backslash(a\cdot\Sigma^*\cup\Sigma^*\cdot b\cup\Sigma^*\cdot a\cdot a\cdot\Sigma^*\cup\Sigma^*\cdot b\cdot b\cdot\Sigma^*)) \\ &= a\cdot\overline{a\cdot\Sigma^*\cup\Sigma^*\cdot b\cup\Sigma^*\cdot a\cdot a\cdot\Sigma^*\cup\Sigma^*\cdot b\cdot b\cdot\Sigma^*} \end{aligned}$$

□

Kann man einer regulären Sprache (gegeben durch einen endlichen Automaten, regulären Ausdruck, etc.) ansehen, ob sie sternfrei ist? Konkreter: Wie zeigt man, daß zum Beispiel $a(aa)^*$ *nicht* sternfrei ist?
Man kann dazu die Charakterisierung der sternfreien Sprachen mit Hilfe von endlichen Monoiden verwenden.

3.2 Aperiodische Monoide

In Kapitel 2 hatten wir gesehen, daß für ein Element m einer endlichen Halbgruppe die Menge $\{m,\ m^2,\ m^3,\ \ldots\}$ ein Idempotentes enthält.



Bei aperiodischen Monoiden kann hier stets $k = 1$ gewählt werden.
Definition 3.3
Ein endlicher Monoid M heißt aperiodisch, falls es ein $n \geq 1$ gibt mit $m^{n+1} = m^n$ für alle $m \in M$.

□

Die Klasse der aperiodischen Monoide ist also eine M-Varietät, da sie schließlich durch $(x^{n+1} = x^n)_{n\geq 1}$ definiert ist.
Beachte dazu: Ist $m^{n+1} = m^n$, so folgt für alle $n' \geq n$, daß $m^{n'+1} = m^{n'}$. Also folgt, wenn ein n gefunden ist, so gilt die Gleichung auch für alle Werte $> n$. Damit ist die Klasse der aperiodischen Monoide schließlich definiert.
Ist $m^i = m^{i+k}$ und sind $m^i,\ m^{i+1},\ \ldots,\ m^{i+k-1}$ paarweise verschieden, so ist $\{m^i,\ \ldots,\ m^{i+k-1}\}$ eine zyklische Gruppe der Ordnung k. Das Einselement der Gruppe ist das Idempotente in $\{m^i,\ \ldots,\ m^{i+k-1}\}$, dieses ist im allgemeinen nicht das Einselement des Monoids.

Definition 3.4
Es sei $(M,\cdot,1)$ ein Monoid. Eine Teilmenge $G \subseteq M$ heißt Gruppe in M, falls G eine *Unterhalbgruppe* von M ist, die bezüglich der Monoidoperation von M eine Gruppe ist.

□

Das Einselement von G ist ein Idempotentes von M, kann aber verschieden von dem Einselement 1 von M sein.
Bei aperiodischen Monoiden sind die zyklischen Gruppen $\{m^i, \ldots, m^{i+k-1}\}$ stets trivial, das heißt sie haben Kardinalität 1. Dies gilt für alle Gruppen in aperiodischen Monoiden.

Satz 3.5
Ein endlicher Monoid M ist genau dann aperiodisch, wenn er nur triviale Gruppen enthält.

Beweis

"⇒" Es sei M aperiodisch und n so, daß $m^{n+1} = m^n$ für alle $m \in M$ gilt.
Angenommen $G \subseteq M$ ist nicht-triviale Gruppe in M, das heißt $|G| > 1$. Es sei $e \in G$ das Einselement von G und $g \in G$ mit $e \neq g$. Es gilt aber $g^{n+1} = g^n$ und damit $g = e$, da man in Gruppen durch Multiplikation mit dem Inversen g^{-1} kürzen kann (hier wird n-mal mit g^{-1} multipliziert).
Widerspruch
Aus diesem Widerspruch folgt, daß es nur triviale Gruppen in M geben kann.

"⇐" Es sei $m \in M$. Wir betrachten wieder $\{m, m^2, \ldots\}$. Es sei $k \geq 1$ minimal, so daß es ein $i \geq 1$ gibt mit $m^{i+k} = m^i$. Da $\{m^i, \ldots, m^{i+k-1}\}$ eine zyklische Gruppe ist, folgt $k = 1$ (da die Menge $\{m^i, \ldots, m^{i+k-1}\}$ k Elemente enthält und es nur triviale Gruppen gibt). Es gibt also für jedes $m \in M$ ein $i_m \geq 1$ mit $m^{i_m+1} = m^{i_m}$. Offenbar folgt für alle $j \geq i_m$ auch $m^{j+1} = m^j$. Daher liefert $max\{i_m | m \in M\}$ eine Zahl n mit

$$\forall m \in M: \quad m^{n+1} = m^n.$$

□

Satz 3.6 (Schützenberg)

$$\underbrace{L(Ap)}_{\substack{\text{Menge der Sprachen,}\\ \text{deren syntakt. Monoid}\\ \text{aperiodisch ist}}} = (SF)_\Sigma$$

das heißt für $L \subseteq \Sigma^*$ gilt

$$M_L \in Ap \ \text{ gdw } \ L \in (SF)_\Sigma$$

□

Der Beweis, insbesondere "⇒", ist sehr aufwendig. Vergleiche hierzu S. Eilenberg: Automata, Languages and Machines, Vol. B.

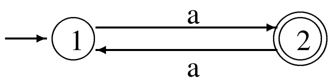
Korollar 3.7
Es sei L eine reguläre Sprache (gegeben durch endlichen Automaten, regulären Ausdruck etc.). Dann kann man effektiv entscheiden, ob $L \in (SF)_\Sigma$ gilt oder nicht.

Beweis
Konstruiere das syntaktische Monoid und überprüfe, ob es aperiodisch ist.

□

Beispiel 3.8
Es sei $\Sigma = \{a\}$. Dann ist $a(aa)^* \notin (SF)_\Sigma$.

Beweis
Der minimale deterministische Automat für $L = a(aa)^*$ ist



Übergangsmonoid:

$$\delta_\varepsilon = \frac{1\,2}{1\,2} \qquad \delta_a = \frac{1\,2}{2\,1} \qquad \delta_{aa} = \delta_a\delta_a = \delta_\varepsilon$$

Damit ist

$$M_L = \{ \underbrace{\delta_\varepsilon}_{Einselement}, \underbrace{\delta_a}_{\delta_a\delta_a = \delta_\varepsilon} \}$$

eine zyklische Gruppe der Kardinalität 2. Somit ist M_L offenbar *nicht* aperiodisch. □

3.3 Formeln der Logik erster Stufe

Ziel dieses Abschnitts ist es, zu zeigen, daß die sternfreien Sprachen genau die durch Formeln der Logik erster Stufe definierbaren Sprachen sind.
Satz 3.9
Für $L \subseteq \Sigma^+$ sind äquivalent

1. $L \in (SF)_\Sigma$
2. $L = L(\varphi)$
   für eine geschlossene Formel φ der Logik erster Stufe mit Gleichheit über den nichtlogischen Symbolen $<$ und Q_a mit $a \in \Sigma$.

□

Beachte: Anstelle der Prädikate $P_1, \ldots, P_n$ verwenden wir Œ direkt Q_a für $a \in \Sigma$.
Der Beweis dieses Satzes ist recht aufwendig, deshalb führen wir ihn in den folgenden beiden Unterabschnitten.

3.3.1 Beweis von "$1 \Rightarrow 2$" von Satz 3.9

Sternfreie Sprachen entstehen aus endlichen Sprachen durch Anwendung Boolescher Operatoren und Konkatenation.

- *Endliche Sprachen*
  Offenbar genügt es, Einermengen $\{w\}$ für $w \in \Sigma^+$ zu betrachten. Für $w = a_1 \ldots a_n \in \Sigma^+$ sei

$$\begin{array}{rll} \varphi_w := \exists x_1 \ldots \exists x_n & & (Q_{a_1}(x_1) \wedge \ldots \wedge Q_{a_n}(x_n)) \\ & \bigwedge_{j=1}^{n-1} & (x_j < x_{j+1} \wedge \neg\exists z(x_j < z \wedge z < x_{j+1})) \\ & \wedge & \neg\exists z(z < x_1 \vee x_n < z)) \end{array}$$

Dann ist $L(\varphi_w) = \{w\}$.

- *Boolesche Operatoren* entsprechen nach Satz 1.26 den logischen Junktoren $\wedge, \vee, \neg$.
- *Konkatenation* entspricht im Prinzip dem Existenzquantor. Zum besseren Verständnis betrachten wir zunächst ein Beispiel:

  $L_1 := a^+$ und $L_2 := b^+$ werden definiert durch $\varphi_1 = \forall x Q_a(x)$ und $\varphi_2 = \forall x Q_b(x)$. Die Sprache $L_1 \cdot L_2$ wird definiert durch

  $$\begin{aligned} \varphi = \exists z \quad & (\forall x(x \leq z \to Q_a(x)) \\ \wedge \quad & \forall x(z < x \to Q_b(x)) \\ \wedge \quad & \exists z'(z < z')) \end{aligned}$$

  Der Quantor in φ_1 wird also relativiert auf die Positionen $\leq z$ und der in φ_2 auf die Positionen $> z$.

  Allgemein wird die *Relativierung* $\varphi^{\leq z}$ wie folgt definiert:

  - ist φ atomar, so gilt $\varphi^{\leq z} = \varphi$
  - $(\psi_1 \wedge \psi_2)^{\leq z} = \psi_1^{\leq z} \wedge \psi_2^{\leq z}$
  - $(\psi_1 \vee \psi_2)^{\leq z} = \psi_1^{\leq z} \vee \psi_2^{\leq z}$
  - $(\neg\psi)^{\leq z} = \neg(\psi^{\leq z})$
  - $(\exists x \psi(x))^{\leq z} = \exists x(x \leq z \wedge \psi(x)^{\leq z})$
  - $(\forall x\ \psi(x))^{\leq z} = \forall x\ (x \leq z \to \psi(x)^{\leq z})$

  Entsprechend ist $\varphi^{>z}$ definiert. Dann gilt für $L_1 = L(\varphi_1)$ und $L_2 = L(\varphi_2)$, daß $L_1 \cdot L_2 = L(\exists z(\varphi_1^{\leq z} \wedge \varphi_2^{>z} \wedge \exists z'(z < z')))$

Damit ist der Beweis von “$1 \Rightarrow 2$” abgeschlossen.

3.3.2 Beweis von “$2 \Rightarrow 1$” von Satz 3.9

Um zu zeigen, daß jede geschlossene Formel der Logik erster Stufe eine Sternfreie Sprache definiert, betrachten wir die Quantorentiefe der Formel, das heißt die maximale Schachtelungstiefe von Quantoren in der Formel. Offenbar enthält jede geschlossene Formel mindestens einen Quantor, da es keine nicht-variablen Terme gibt (weil die hier betrachtete Logik keine Konstanten-Symbole enthält).
Da die Negation vorhanden ist, können wir Œ annehmen, daß alle Quantoren existentielle Quantoren sind. Da die Junktoren $\wedge$, $\vee$, $\neg$ den Booleschen Operatoren entsprechen, genügt es, sich auf Formeln der Form $\exists x\ \varphi(x)$ zu konzentrieren.

Induktionsverankerung Quantorentiefe 1
In diesem Fall enthält φ keinen Quantor und damit keine andere Variable als x. Das heißt φ ist Boolesche Kombination von Formeln

- $Q_a(x)$ oder $\neg Q_a(x)$
- $x < x$ oder $\neg(x < x)$
- $x = x$ oder $\neg(x = x)$

Œ sei φ in Disjunktiver Normalform, das heißt als Disjunktion von Konjunktionen obiger atomarer Formeln, gegeben.

Diese kann noch weiter normalisiert werden:

- ersetze $x < x$ und $\neg(x = x)$ durch “false”, das heißt lösche das gesamte Disjunkt, in dem die Formel vorkommt
- ersetze $\neg(x < x)$ und $x = x$ durch “true”, das heißt entferne sie aus dem Disjunkt

Bleibt nach dieser Normalisierung nur “true” oder “false” übrig, so gilt $L(\exists x\varphi(x)) = \Sigma^+$ oder $L(\exists x\varphi(x)) = \emptyset$. Diese beiden Sprachen sind aber sternfrei, das heißt in diesem Fall sind wir fertig.
Damit kommen Œ nur noch $Q_a(x)$ beziehungsweise $\neg Q_a(x)$ vor. Man kann weiter normalisieren: Enthält ein Disjunkt $Q_a(x)$

- lösche es, falls es auch $\neg Q_a(x)$ oder $Q_b(x)$ für $b \neq a$ enthält
- sonst lösche alle $\neg Q_b(x)$ für $a \neq b$ aus dem Disjunkt; diese enthalten dann keine zusätzliche Information

Damit haben wir nur noch Disjunkte der Formen

- $Q_a(x)$
- $\neg Q_{a_1}(x) \wedge \ldots \wedge \neg Q_{a_k}(x)$

Wir betrachten nun jeweils die *zugehörige Sternfreie Sprache*:

- $L(\exists x Q_a(x)) = \overbrace{\Sigma^*}^{\overline{\emptyset}} \cdot \overbrace{a}^{endlich} \cdot \overbrace{\Sigma^*}^{\overline{\emptyset}} \quad \Rightarrow \quad sternfrei$
- $L(\exists x:\ \neg Q_{a_1}(x) \wedge \ldots \wedge \neg Q_{a_k}(x)) = \overbrace{\Sigma^*}^{\overline{\emptyset}} \cdot \overbrace{\Sigma \backslash \{a_1, \ldots, a_k\}}^{endlich} \cdot \overbrace{\Sigma^*}^{\overline{\emptyset}} \quad \Rightarrow \quad sternfrei$

Damit ist die Quantorentiefe 1 abgehandelt, das heißt der Induktionsanker ist komplett.
Für den **Induktionsschritt** sei $\exists x\ \varphi(x)$ von Quantorentiefe $n+1$. Wir benötigen einige Notation und zwei Sätze, bevor wir den Induktionsschritt durchführen können.

Definition 3.10
Mit $L_{k,n}$ bezeichnen wir die Formeln der Logik erster Stufe über den nicht-logischen Symbolen $<$ und Q_a für $a \in \Sigma$, die k *freie Variable* und *Quantorentiefe* $\leq n$ haben. □

Zum Beispiel gehört $\varphi = \exists x(y < x \wedge x < z \wedge Q_a(x))$ zu $L_{2,1}$.
Um diese Formel zu interpretieren, genügt ein Wort (z. B. *baa*) nicht. Man muß auch angeben, wie die freien Variablen interpretiert werden (hier also y und z, z.B. y durch 1 (erste Position im Wort *baa*) und z durch 3 (dritte Position im Wort *baa*))). Wir sagen (*baa*, 1, 3) *erfüllt* φ.
Allgemein werden Formeln aus $L_{k,n}$ durch Tupel $(w, \vec{s})$ interpretiert, wobei $w \in \Sigma^+$ und $\vec{s} = s_1 \ldots s_k$ mit $s_i \in \{1, \ldots, |w|\}$.
$(w, \vec{s}) \models \varphi \in L_{k,n}$ (“$(w, \vec{s})$” efüllt φ) ist in der offensichtlichen Weise definiert. Für $k = 0$ läßt man im allgemeinen die leere Sequenz $\vec{s}$ weg.

Definition 3.11
Für $n \geq 0$ und $k \geq 0$ definieren wir

$$(w,\vec{s}) \equiv_{k,n} (v,\vec{t}) \text{ gdw } \textit{für alle } \varphi \in L_{k,n} \text{ gilt } (w,\vec{s}) \models \varphi \; \textit{gdw} \; (v,\vec{t}) \models \varphi$$

□

Offenbar ist $\equiv_{k,n}$ eine Äquivalenzrelation (dies erkennt man an dem *gdw* in der Definition). Für $k = 0$ schreiben wir $\equiv_n$ statt $\equiv_{0,n}$. $\{\varepsilon\}$ wird in diesem Fall zu einer eigenen $\equiv_{0,n}$-Klasse. Um den Beweis von "$2 \Rightarrow 1$" abzuschließen benötigen wir zwei Sätze, die wir später in einem eigenen Unterabschnitt beweisen werden.

Satz 3.12
Für alle $n \geq 0$ und $k \geq 0$ gibt es eine *endliche* Teilmenge $\Gamma_{k,n}$ von $L_{k,n}$, so daß gilt:
jedes Element von $L_{k,n}$ ist äquivalent zu einem Element von $\Gamma_{k,n}$.
Äquivalent heißt: die Formeln werden erfüllt durch genau dieselben Tupel $(w,\vec{s})$.

□

Korollar 3.13
Für $n \geq 0$, $k \geq 0$ hat $\equiv_{k,n}$ nur endlich viele Äquivalenzklassen.

Beweis
Die Klasse von $(w,\vec{s})$ ist eindeutig bestimmt durch eine Formelmenge

$$\Gamma := \{\varphi \in \Gamma_{k,n} \mid (w,\vec{s}) \models \varphi\} \subseteq \underbrace{\Gamma_{k,n}}_{endlich} .$$

Da es nur endlich viele solche Teilmengen gibt, kann es nur endlich viele verschiedene Klassen geben.

□

Der Begriff der durch eine Formel akzeptierten Sprache wird auf den Fall $k > 0$ ausgedehnt: für $\varphi \in L_{k,n}$ sei

$$L(\varphi) := \{(w,\vec{s}) \mid (w,\vec{s}) \models \varphi\}.$$

Korollar 3.14
Für alle $n \geq 0$, $k \geq 0$ und alle $\equiv_{k,n}$-Klassen W gibt es eine Formel $\varphi_W \in L_{k,n}$ mit $L(\varphi_W) = W$.

Beweis

$$\underbrace{[(w,\vec{s})]_{\equiv_{k,n}}}_{\substack{=W \\ =\{(u,\vec{t}) \mid (u,\vec{t}) \equiv_{k,n} (w,\vec{s})\}}} = L\left(\bigwedge_{\substack{\varphi \in \Gamma_{k,n} \\ (w,\vec{s}) \models \varphi}} \varphi \wedge \bigwedge_{\substack{\psi \in \Gamma_{k,n} \\ (w,\vec{s}) \not\models \psi}} \neg\psi \right)$$

□

Korollar 3.15
Für alle $n \geq 0$, $k \geq 0$ und alle $\varphi \in L_{k,n}$ ist φ äquivalent zu einer *endlichen* Disjunktion von Formeln φ_W für gewisse $\equiv_{k,n}$-Klassen W.

Beweis
φ ist äquivalent zu
$$\bigvee_{\substack{W=[(w,\vec{s})]_{\equiv_{k,n}}\\(w,\vec{s})\models\varphi}} \varphi_W.$$
□

Satz 3.16
Sei $n \geq 0$, $u,\ v,\ u',\ v' \in \Sigma^*$ und $a \in \Sigma$. Dann folgt aus $u \equiv_n u'$ und $v \equiv_n v'$, daß $(uav,\ |u|+1) \equiv_{1,n} (u'av',\ |u'|+1)$.
□

Der Beweis dieses Satzes folgt im nächsten Abschnitt.
Wir können damit nun den **Induktionsschritt** abschließen.
Sei also $\exists x\ \varphi(x)$ von Quantorentiefe $n+1$. Damit ist $\varphi(x) \in L_{1,n}$. O.B.d.A. sei $\varphi(x)$ eine Formel φ_W für eine $\equiv_{1,n}$-Klasse W. Dies ist keine Einschränkung, denn $\varphi(x)$ ist äquivalent zu
$$\bigvee_{\substack{W=[(w,\vec{s})]_{\equiv_{k,n}}\\(w,\vec{s})\models\varphi}} \varphi_W(x).$$
Dies gilt mit Korollar 3.15.
Damit ist $\exists x\ \varphi(x)$ äquivalent zu
$$\bigvee_{\substack{W=[(w,\vec{s})]_{\equiv_{k,n}}\\(w,\vec{s})\models\varphi}} \exists x\ \varphi_W(x).$$
Disjunktion kann durch Vereinigung behandelt werden, daher betrachten wir nun nur noch Formeln der Form $\exists x\ \varphi_W(x)$.

Lemma 3.17
$$L\left(\exists x \underbrace{\varphi_W(x)}_{W\ ist\ \equiv_{1,n}-Klasse}\right) = \bigcup_{\substack{U=[u_0]_{\equiv_{0,n}}\\V=[v_0]_{\equiv_{0,n}}\\a\in\Sigma\\(u_0av_0,\ |u_0|+1)\in W}} U\cdot a\cdot V$$

Beweis
"$\subseteq$"
$$\begin{aligned} w \in L(\exists x\ \varphi_W(x)) \quad &\text{gdw} \quad w = uav \wedge (uav,\ |u|+1) \models \varphi_W(x)\\ &\text{gdw} \quad w = uav \wedge (uav,\ |u|+1) \in W\end{aligned}$$
Wählt man $U = [u]_{\equiv_{0,n}}$ und $V = [v]_{\equiv_{0,n}}$, dann ist $w = uav \in [u]_{\equiv_{0,n}} \cdot a \cdot [v]_{\equiv_{0,n}}$, das heißt $w = uav$ ist in der Rechten Seite der Formel des Lemmas enthalten.

"$\supseteq$" Aus $(u_0av_0,\ |u_0|+1) \in W = L(\varphi_W)$ folgt $u_0av_0 \in L(\exists x\ \varphi_W(x))$. Es bleibt zu zeigen, daß auch für alle $u \equiv_{0,n} u_0$ und $v \equiv_{0,n} v_0$ gilt: $uav \in L(\exists x\ \varphi_W(x))$. Mit Satz 3.16 gilt aber:
$$(u_0av_0,\ |u_0|+1) \equiv_{1,n} (uav,\ |u|+1),$$
was $(uav,\ |u|+1) \in W$ liefert, das heißt $(uav,\ |u|+1) \models \varphi_W(x)$. Es folgt $uav \models \exists x\ \varphi_W(x)$.

□

Mit Korollar 3.14 sind die $\equiv_{0,n}$-Klassen U und V aus dem Lemma von der Form $U = L(\varphi_U)$ und $V = L(\varphi_V)$ für $\varphi_U,\ \varphi_V \in L_{0,n}$.
Induktion liefert $U,\ V \in (SF)_\Sigma$ und damit

$$\underbrace{U}_{\in (SF)_\Sigma} \cdot \underbrace{a}_{endlich} \cdot \underbrace{V}_{\in (SF)_\Sigma} \in (SF)_\Sigma.$$

Das schließt den Beweis von "2 $\Rightarrow$ 1" des Satzes 3.9 ab.

3.3.3 Beweis der Sätze 3.12 und 3.16

Satz 3.12
Für alle $n \geq 0$, $k \geq 0$ gibt es eine *endliche* Teilmenge $\Gamma_{k,n}$ von $L_{k,n}$, so daß gilt: Jedes Element von $L_{k,n}$ ist äquivalent zu einem Element von $\Gamma_{k,n}$.
Beweis
Es sei $\varphi \in L_{k,n}$. Ohne Einschränkung können wir davon ausgehen, daß φ Boolesche Kombination ist von

- Elementen aus $L_{k,n-1}$
- Formeln der Form $\exists x\ \psi(x, y_1, \ldots, y_k)$ mit $\psi(x, y_1, \ldots, y_k) \in L_{k+1,n-1}$.

Wir verwenden daher *Induktion über n*:

$n = 0$: Es sei $k \geq 0$ beliebig. Eine Formel $\varphi(y_1, \ldots, y_k) \in L_{k,0}$ ist eine Boolesche Kombination von Formeln

$$Q_a(x) \ \ (a \in \Sigma), \quad x < y, \quad x = y \tag{3.1}$$

wobei $x, y \in \{y_1, \ldots, y_k\}$. Es gibt offenbar nur endlich viele atomare Formeln der Form (3.1) und daher (bis auf Äquivalenz) nur endlich viele Boolesche Kombinationen.

$n-1 \to n$: φ ist Boolesche Kombination von Elementen aus $L_{k,n-1}$ und Formeln $\exists x\ \psi(x, y_1, \ldots, y_k)$ mit $\psi(x, y_1, \ldots, y_k) \in L_{k+1,n-1}$.
Damit ist φ äquivalent zu einer Booleschen Kombination von Elementen aus $\Gamma_{k,n-1}$ und Formeln $\exists x\ \gamma(x, y_1, \ldots, y_k)$ mit $\gamma(x, y_1, \ldots, y_k) \in \Gamma_{k+1,n-1}$.
Mit der Induktionsvoraussetzung wissen wir, daß diese endlichen Mengen $\Gamma_{k,n-1}$ und $\Gamma_{k+1,n-1}$ existieren.

□

Zum Beweis von Satz 3.16 benötigen wir eine spieltheoretische Charakterisierung der Relationen $\equiv_{k,n}$.

Definition 3.18 (Ehrenfeucht-Fraissé Spiele)
Es sei Σ ein endliches Alphabet. Wir betrachten zwei Spieler I und II, die auf Wörtern $u, v \in \Sigma^+$ spielen. Ein Zug besteht darin, eine Position in u oder v zu wählen.
Beginnend mit I wird abwechselnd gezogen. Zieht I in u, so muß II in v ziehen; zieht I in v, so muß II in u ziehen. Ein Spiel der Länge n besteht aus n Zügen von I und den n Antwortzügen von II.

Es seien $(i_1, j_1),\ \ldots,\ (i_n, j_n)$ die gezogenen Positionen eines solchen Spiels, wobei die i_ν die in u gezogene Position ist und j_ν die in v gezogene Position (unabhängig davon, wer gezogen hat).
Spieler II hat gewonnen, wenn für $u = a_1 \ldots a_p$ und $v = b_1 \ldots b_q$ gilt:

- $a_{i_\nu} = b_{j_\nu}$ für $\nu = 1,\ \ldots,\ n$
  an den gezogenen Positionen für den Zug ν steht dasselbe Symbol

- $i_\nu < i_{\nu'}$ gdw $j_\nu < j_{\nu'}$
  die relative Lage der gezogenen Positionen in u und v ist gleich

Sonst hat II verloren und damit I gewonnen.

□

Die Definition des *Gewinnens* bedeutet, daß I versucht so zu ziehen, daß man dadurch zwischen u und v unterscheiden kann, während II versucht, das zu verhindern.

Beispiel 3.19
Seien $n = 3$ und $\Sigma = \{a\}$, das heißt nur die Ordnung zwischen den gezogenen Positionen ist relevant. Folgender Spielverlauf wäre denkbar:

$$\begin{array}{ccccccccc}
 & & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
u & = & a & \overbrace{a}^{I,2} & a & \overbrace{a}^{II,1} & a & a & \\
v & = & a & \underbrace{a}_{II,2} & a & a & \underbrace{a}_{I,1} & a & a
\end{array}$$

Überlegungen zum ersten Zug von II:
Hätte II auf Position 6 gezogen, dann hätte I durch einen Zug auf Position 7 in v das Spiel gewonnen. Ein Zug auf Position 5 hätte dazu geführt, daß I durch Ziehen auf Position 6 und anschließend auf Position 7 gewonnen hätte.
Überlegungen zum zweiten Zug von II:
sowohl Position 2 als auch Position 3 in v sind geeignet, alle anderen Positionen würden einen Sieg von I zur Folge haben.
Man sieht, daß II jeden möglichen dritten Zug von I geeignet beantworten kann und damit für $n = 3$ das Spiel gewinnt.
Für $n = 4$ könnte I auf die Positionen 3 und 4 in v ziehen und damit gewinnen, da für den vierten Zug keine geeignete Antwort in u möglich ist.

□

Beispiel 3.20
$\Sigma = \{a, b\}$, $n = 3$

$$\begin{array}{cccccc}
 & & 1 & 2 & 3 & 4 \\
u & = & \overbrace{b}^{I,3} & \overbrace{b}^{I,2} & \overbrace{a}^{I,1} & \mathrm{b} \\
v & = & \underbrace{b}_{II,2} & \underbrace{a}_{II,1} & b & b
\end{array}$$

II kann den dritten Zug von I nicht mehr beantworten, damit hat I gewonnen.

□

Definition 3.21
Es seien $n \geq 1$ und $u, v \in \Sigma^+$. Wir sagen, daß II eine Gewinnstrategie für Spiele der Länge n auf u und v hat, wenn II alle möglichen Züge von I stets so beantworten kann, daß II gewinnt.

□

Um Induktionsargumente einsetzen zu können, betrachten wir Spiele, die schon ein Stück angefangen haben: Es seien dazu $u,\ v \in \Sigma^+$, $\vec{s} = s_1 \ldots s_k \in \{1,\ \ldots,\ |u|\}^k$ und $\vec{t} = t_1 \ldots t_k \in \{1,\ \ldots, |v|\}^k$. Dann beschreibt das Paar $(u, \vec{s})$ und $(v, \vec{t})$ ein Spiel, das schon k Züge angefangen ist. Ein Spiel der Länge n darauf ist eine Fortsetzung um n weitere Züge. Spieler II hat das Fortsetzungsspiel gewonnen, wenn II das Gesamtspiel gewonnen hat.

Definition 3.22
Es seien $(u, \vec{s})$ und $(v, \vec{t})$ mit $u,\ v \in \Sigma^+$ und $\vec{s} \in \{1, \ldots, |u|\}^k$ und $\vec{t} \in \{1,\ \ldots,\ |v|\}^k$ gegeben. Wir definieren:

$$(u, \vec{s}) \sim_{k,n} (v, \vec{t}) \;\; gdw \;\; \begin{array}{c}\text{II hat für das Forsetzungsspiel der Länge } n \text{ auf} \\ (u, \vec{s}) \text{ und } (v, \vec{t}) \text{ eine Gewinnstrategie}\end{array}$$

Für $k = 0$ erweitern wir wieder wie in Definition 3.11 $\sim_{0,n}$ auf Σ^*, indem wir $\{\varepsilon\}$ zu einer eigenen $\sim_{0,n}$-Äquivalenzklasse machen.

□

Lemma 3.23
$\sim_{k,n}$ ist eine Äquivalenzrelation.

Beweis

- reflexiv: II simuliert genau die Züge von I (da für die Reflexivität ja beide Spieler auf gleichen Worten spielen)
- symmetrisch: klar, da $(u, \vec{s})$ und $(v, \vec{t})$ im Spiel symmetrisch verwendet werden
- transitiv: Es seien
$$\underbrace{(u, \vec{s}) \sim_{k,n} (v, \vec{t})}_{Gewinnstrategie\ GS\ 1}$$
und
$$\underbrace{(v, \vec{t}) \sim_{k,n} (w, \vec{r})}_{Gewinnstrategie\ GS\ 2}$$
Die Strategie für II auf $(u, \vec{s})$ und $(w, \vec{r})$ lautet dann wie folgt:
  - zieht I in u, so reagiert II mit GS 1 in v und darauf mit GS 2 in w
  - ein Zug von I in w kann entsprechend behandelt werden

□

Für die Relation $\sim_{k,n}$ kann man eine Aussage beweisen, die der von Satz 3.16 für $\equiv_{k,n}$ entspricht.

Lemma 3.24
Seien $n \geq 0,\ u,\ u',\ v,\ v'\ \in \Sigma^*$ und $a \in \Sigma$. Dann folgt aus $u \sim_{0,n} u'$ und $v \sim_{0,n} v'$ auch $(uav, |u|+1) \sim_{1,n} (u'av', |u'|+1)$.

Beweis
Betrachte den Fall $u \neq \varepsilon \neq v$. Es seien GS 1 die Strategie für $u \sim_{0,n} u'$ und GS 2 die Strategie für $v \sim_{0,n} v'$.
Zieht I in u (bzw. u'), so antwortet II mit GS 1 in u' (bzw. u). Zieht I in v (bzw. v'), so antwortet II mit GS 2 in v' (bzw. v). Zieht I die Position $|u|+1$ in uav (bzw. die Position $|u'|+1$ in $u'av'$), so antwortet II mit $|u'|+1$ in $u'av'$ (bzw. $|u|+1$ in uav).
Offenbar liefert das eine Gewinnstrategie für II.
Die Fälle $u = \varepsilon = u'$ und $v = \varepsilon = v'$ können entsprechend behandelt werden.

□

Um Satz 3.16 zu erhalten genügt es zu zeigen, daß $\sim_{k,n}$ und $\equiv_{k,n}$ übereinstimmen.
Zunächst geben wir ein intuitives Argument für die Übereinstimmung der beiden Relationen:
Offenbar sind die Chancen für II um so größer, je ähnlicher die beiden Wörter u, v sind, auf denen gespielt wird.
$u \equiv_n v$ heißt, daß man u und v nicht durch eine Formel der Quantorentiefe n unterscheiden kann.
Der Zusammenhang zwischen der Quantorentiefe und der Anzahl der Züge ist auch einleuchtend: $\exists x\, \varphi(x)$ sagt aus, daß es eine Position mit bestimmten Eigenschaften gibt. Durch einen Zug wird eine Position ausgewählt.

Lemma 3.25
Für alle $n,k \geq 0$ gilt: $\quad \sim_{k,n} \quad = \quad \equiv_{k,n}$

Beweis Induktion über n

Induktionsanker: n=0
Es seien $(u,\vec{s})$ und $(v,\vec{t})$ gegeben.

1. Angenommen $(u,\vec{s}) \equiv_{k,0} (v,\vec{t})$. Œ seien $\{y_1,\ \ldots,\ y_k\}$ die (erlaubten) freien Variablen. Für $i = 1,\ \ldots,\ k$ entsprechen s_i und t_i der Variablen y_i. Für atomare Formeln der Formen $y_i < y_j$, $y_i = y_j$ und $Q_a(y_i)$ gilt wegen $(u,\vec{s}) \equiv_{k,0} (v,\vec{t})$:

$$(*) \left\{ \begin{array}{lll} (u,\vec{s}) \models y_i < y_j & \text{gdw} & (v,\vec{t}) \models y_i < y_j \\ (u,\vec{s}) \models Q_a(y_i) & \text{gdw} & (v,\vec{t}) \models Q_a(y_i) \\ (u,\vec{s}) \models y_i = y_j & \text{gdw} & (v,\vec{t}) \models y_i = y_j \end{array} \right.$$

   Das heißt aber (für $u = a_1 \ldots a_p,\ v = b_1 \ldots b_q$)

$$(**) \left\{ \begin{array}{lll} s_i < s_j & \text{gdw} & t_i < t_j \\ a_{s_i} = a & \text{gdw} & b_{t_i} = a \\ s_i = s_j & \text{gdw} & t_i = t_j \end{array} \right.$$

   Das bedeutet aber, daß II das Spiel mit $n = 0$ zusätzlichen Zügen gewonnen hat. Dies gilt mit den ersten beiden Zeilen von $(**)$.

2. Umgekehrt folgt aus $(u,\vec{s}) \sim_{k,0} (v,\vec{t})$, daß die ersten beiden Zeilen von $(**)$ erfüllt sind. Die dritte Zeile ergibt sich aus der ersten und der Tatsache, daß $<$ eine totale Ordnung ist.
   Damit erhalten wir $(*)$, was ja äquivalent zu $(**)$ ist. Dies zeigt $(u,\vec{s}) \equiv_{k,0} (v,\vec{t})$.

Beachte: Formeln der Quantorentiefe 0 sind Boolesche Kombinationen von atomaren Formeln. $(*)$ zeigt das Gewünschte nur für atomare Formeln.

Induktionsschritt: $n \to n+1$

1. Angenommen es gelte $(u,\vec{s}) \sim_{k,n+1} (v,\vec{t})$. Es sei $\varphi(y_1,\ldots,y_k) \in L_{k,n+1}$ mit $(u,\vec{s}) \models \varphi$ gegeben. Es genügt zu zeigen, daß daraus $(v,\vec{t}) \models \varphi$ folgt.
   Da Boolesche Operatoren unproblematisch sind, genügt es, Formeln φ der Form

   $$\varphi = \exists y_{k+1} : \ \psi(y_1,\ldots,y_k,y_{k+1})$$

   zu betrachten mit $\psi \in L_{k+1,n}$.
   Wegen $(u,\vec{s}) \sim_{k,n+1} (v,\vec{t})$ kann II jeden möglichen Zug von I so beantworten, daß II danach immernoch eine Gewinstrategie hat. Daher gibt es zu jedem $s_{k+1} \in \{1,\ldots,|u|\}$ ein $t_{k+1} \in \{1,\ldots,|v|\}$ mit

   $$(u,\vec{s}s_{k+1}) \sim_{k+1,n} (v,\vec{t}t_{k+1}).$$

   Induktion liefert, daß für alle s_{k+1} ein t_{k+1} existiert mit

   $$(u,\vec{s}s_{k+1}) \equiv_{k+1,n} (v,\vec{t}t_{k+1}) \tag{3.2}$$

   Wegen $(u,\vec{s}) \models \exists y_{k+1} : \ \psi(y_1,\ldots,y_k,y_{k+1})$ gibt es ein $s_{k+1} \in \{1,\ldots,|u|\}$ mit

   $$(u,\vec{s}s_{k+1}) \models \psi(y_1,\ldots,y_k,y_{k+1}) \tag{3.3}$$

   Sei nun t_{k+1} so, daß (3.2) gilt. Wegen $\psi \in L_{k+1,n}$ folgt aus $(**)$ auch

   $$(v,\vec{t}t_{k+1}) \models \psi(y_1,\ldots,y_k,y_{k+1})$$

   Daraus folgt offenbar

   $$(v,\vec{t}) \models \underbrace{\exists y_{k+1} : \ \psi(y_1,\ldots,y_k,y_{k+1})}_{\varphi}$$

   Damit ist gezeigt, daß u und v äquivalent sind für Formeln aus $L_{k,n+1}$.

2. Angenommen $(u,\vec{s}) \not\sim_{k,n+1} (v,\vec{t})$.
   Gesucht ist eine Formel $\varphi \in L_{k,n+1}$, die von einem der beiden Tupel erfüllt wird, aber nicht von dem anderen. Daraus folgt dann $(u,\vec{s}) \not\equiv_{k,n+1} (v,\vec{t})$.
   Wegen $(u,\vec{s}) \not\sim_{k,n+1} (v,\vec{t})$ gibt es einen ersten Zug von I, den II nicht so beantworten kann, daß II danach eine Gewinnstrategie hat. Œ liege dieser erste Zug von I in u. (Der Fall, daß der erste Zug in v liegt, kann symmetrisch behandelt werden.) Das heißt es gibt ein $s_{k+1} \in \{1,\ldots,|u|\}$, so daß für alle $t_{k+1} \in \{1,\ldots,|v|\}$ gilt:

   $$(u,\vec{s}s_{k+1}) \not\sim_{k+1,n} (v,\vec{t}t_{k+1})$$

Induktion liefert

$$(u,\vec{s}s_{k+1}) \not\equiv_{k+1,n} (v,\vec{t}\,t_{k+1})$$

Es sei dieses s_{k+1} fixiert. Für alle t_{k+1} gibt es daher eine Formel

$$\psi_{t_{k+1}}(y_1,\ldots,y_{k+1}) \in L_{k+1,n}$$

mit

$$(u,\vec{s}s_{k+1}) \models \psi_{t_{k+1}} \quad \text{und} \tag{3.4}$$

$$(v,\vec{t}\,t_{k+1}) \not\models \psi_{t_{k+1}} \tag{3.5}$$

Beachte: $(u,\vec{s}) \models \neg\psi$ gdw $(u,\vec{s}) \not\models \psi$.
Aus (3.4) folgt für $\varphi_{t_{k+1}} := \exists y_{k+1} : \ \psi_{t_{k+1}}$ $(u,\vec{s}) \models \varphi_{t_{k+1}}$.
Aus (3.5) kann man leider nicht folgern, daß $(v,\vec{t}) \not\models \varphi_{t_{k+1}}$. Wir wissen nur, daß t_{k+1} nicht für y_{k+1} in $\psi_{t_{k+1}}$ eingesetzt werden kann. Über andere t's ist aber nichts bekannt. Wir betrachten daher statt einem $\varphi_{t_{k+1}}$ die Formel

$$\varphi := \exists y_{k+1} : \bigwedge_{1\leq t_{k+1}\leq |u|} \psi_{t_{k+1}}$$

Aus (3.4) folgt $(u,\vec{s}) \models \varphi$. Aus (3.5) folgt, daß es zu jedem $t_{k+1} \in \{1,\ldots,|v|\}$ ein Konjunkt, nämlich $\psi_{t_{k+1}}$ in φ gibt, das nicht erfüllt ist. Daraus folgt $(v,\vec{t}) \not\models \varphi$.

□

Beachte: Lemma 3.25 gilt entsprechend auch für unendliche Wörter. Um dann die Konjunktion

$$\bigwedge_{t_{k+1}\ Position\ in\ v} \psi_{t_{k+1}}$$

endlich zu lassen, muß man Satz 3.12 verändern. ($L_{k+1,n}$ ist äquivalent zu endlicher Teilmenge $\Gamma_{k+1,n}$.)

3.4 Die Theorie der totalen Ordnung

Aus der Sicht der Logik ist Satz 3.9 interessant, weil er ein Entscheidbarkeitsresultat liefert.

Theorie der totalen Ordnung:

$$\begin{aligned} Total \ = \{ \ & \forall x \forall y \forall z : \ x<y \wedge y<z \rightarrow x<z \\ & \forall x : \ \neg(x<x) \\ & \forall x \forall y : \ (x<y \vee x=y \vee y<x)\} \end{aligned}$$

Satz 3.26
Es sei φ eine geschlossene Formel der Logik erster Stufe mit Gleichheit, die höchstens die nichtlogischen Symbole $<, P_1,\ldots,P_k$ (für einstellige Prädikate P_i) enthält. Dann ist es entscheidbar, ob $Total \cup \{\varphi\}$ ein endliches Modell hat oder nicht.

Beweis
Wir haben gesehen, daß $L(\varphi)$ eine sternfreie Sprache ist. Der Beweis von Satz 3.9 ist konstruktiv, das heißt man kann zu φ effektiv einen sternfreien Ausdruck (mit Booleschen Operatoren, Konkatenation aus endlichen Sprachen) bestimmen. Das liegt im wesentlichen daran, daß man $\Gamma_{k,n}$ effektiv bestimmen kann. Damit sind die Relationen $\equiv_{k,n}$ entscheidbar.
Daraus kann ein endlicher Automat für $L(\varphi)$ konstruiert werden.
Existenz eines endlichen Modells heißt: es gibt ein endliches Wort $u \in L(\varphi)$. Das Leerheitsproblem $(L(\mathcal{A}) = \emptyset)$ ist aber für endliche Automaten entscheidbar.

□

Häufig möchte man wissen, ob eine solche Formel ein Modell hat, bei dem $<$ als die gewöhnliche Ordnung auf den natürlichen Zahlen interpretiert wird.
Um solche Probleme handhaben zu können, betrachten wir im folgenden unendliche Wörter und Automaten auf unendlichen Wörtern.
Wir haben gesehen: nicht alle regulären Sprachen sind von der Form $L(\varphi)$ für eine Formel φ der Logik erster Stufe; zum Beispiel ist $L = a(aa)^*$ nicht sternfrei.
Alle regulären Sprachen sind definierbar, wenn man zusätzlich Quantifizierung über einstellige Prädikate zuläßt (vergleiche Beispiel 1.28).
Wir werden diese Aussage für den Fall unendlicher Wörter beweisen. Der Beweis für endliche Wörter ist entsprechend.

Kapitel 4

Unendliche Wörter und Büchi-Automaten

Sei Σ ein endliches Alphabet. Ein endliches Wort $w \in \Sigma^*$ der Länge k ist eine Folge $w = a_0 a_1 \ldots a_{k-1}$ von k Elementen aus Σ. Man kann daher w auffassen als eine Abbildung

$$w : \{0, 1, \ldots, k-1\} \to \Sigma$$

mit $w(i) = a_i$.
Ein endliches Wort ist also eine Abbildung von einem Anfangssegment der natürlichen Zahlen nach Σ.
Wir werden im folgenden für die natürlichen Zahlen ω schreiben. ω steht für den Ordnungstyp der natürlichen Zahlen.

Definition 4.1

1. Ein unendliches Wort über Σ ist eine Abbildung $\alpha : \omega \to \Sigma$.
2. Mit Σ^ω bezeichnen wir die Menge aller unendlichen Wörter über Σ.
3. Mengen unendlicher Wörter heißen ω-Sprachen.

□

Wir schreiben unendliche Wörter häufig als $\alpha = a_0 a_1 a_2 a_3 \ldots$, wobei $\alpha(i) = a_i$.

Beispiel 4.2
$\Sigma = \{a, b\}$

$$\alpha(i) = \begin{cases} a & i \text{ gerade} \\ b & i \text{ ungerade} \end{cases}$$

wird geschrieben als $\alpha = ababab\ldots$. □

Wir betrachten einige Operationen für unendliche Wörter

- Segment
  für $\alpha: \ \omega \to \Sigma$ bezeichne
  - $\alpha(m,n)$ das endliche Wort $\alpha(m)\alpha(m+1)\ldots\alpha(n)$ (für $m \leq n$).
  - $\alpha(m,\infty)$ das unendliche Wort $\alpha(m)\alpha(m+1)\alpha(m+2)\ldots$
- Konkatenation eines endlichen Wortes mit einem unendlichen Wort
  Sei $w = a_1 a_2 \ldots a_m \in \Sigma^*$ und $\alpha = \alpha(0)\alpha(1)\alpha(2)\ldots \ \in \Sigma^\omega$. Dann ist $w \cdot \alpha$ das unendliche Wort $a_1 a_2 \ldots a_m \alpha(0)\alpha(1)\alpha(2)\ldots$.
  Beachte: Ein unendliches Wort auf der linken Seite macht keinen Sinn.
  Wie üblich kann Konkatenation auf Mengen erweitert werden.
- Unendliche Iteration
  Es sei $L \subseteq \Sigma^*$ eine Menge endlicher Wörter. Wir definieren

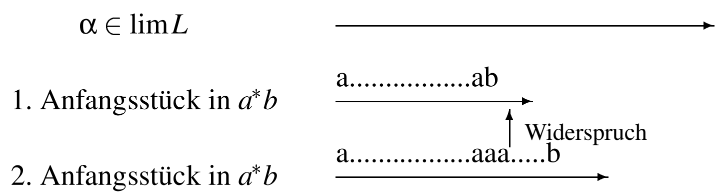
  $$L^\omega \ := \ \{\alpha \in \Sigma^\omega \mid \alpha = w_0 w_1 w_2 \ldots \text{ mit } w_i \in L \backslash \{\varepsilon\}\}$$

  Beispiel $L = \{ab\}$. Dann ist $L^\omega = \{abababab\ldots\}$. Wir schreiben in solchen Fällen häufig $(ab)^\omega$.
- Limes
  Es sei $L \subseteq \Sigma^*$. Wir definieren

  $$\lim L \ := \ \{\alpha \in \Sigma^\omega \mid \text{es gibt unendlich viele } n \text{ mit } \alpha(0,n) \in L\}$$

Beispiel 4.3
$\Sigma = \{a,b\}$

1. $L = a^* b$. Es gilt $\lim L = \emptyset$. Erklärung:



2. $L = ba^*$. Es gilt $\lim L = \{baa\ldots\} = ba^\omega$.
3. $L = (a^* b b^*)^*$. In diesem Fall gilt

$$\lim L \ = \{\alpha \in \Sigma^\omega \mid \ \text{nach jedem Vorkommen von } a \text{ kommt irgendwann noch ein } b \text{ in } \alpha\}$$

□

Als Automaten zum Akzeptieren von Mengen unendlicher Wörter (ω-Sprachen) betrachten wir normale endliche Automaten. Der Unterschied liegt in der Akzeptanzbedingung.

Definition 4.4
Ein Büchi-Automat ist ein (nicht-deterministischer) endlicher Automat $\mathcal{A} = (Q,\ \Sigma,\ I,\ \Delta,\ F)$, das heißt

- Q ist eine nicht-leere endliche Menge,
- Σ ist ein endliches Alphabet,
- $I \subseteq Q$ ist die Anfangszustandsmenge,
- $\Delta \subseteq (Q \times \Sigma \times Q)$ ist die Übergangsrelation,
- $F \subseteq Q$ ist die Endzustandsmenge.

Da wir uns für unendliche Wörter interessieren, betrachten wir unendliche Pfade in $\mathcal{A}$

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} q_3 \xrightarrow{a_4} \dots$$

wobei $(q_i, a_{i+1}, q_{i+1}) \in \Delta$ für $i \geq 0$.
Die Beschriftung dieses Pfads ist ein unendliches Wort $a_1 a_2 a_3 a_4 \dots$.
Wann ist nun ein Pfad erfolgreich?

- endlicher Pfad
  Der Pfad $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \dots \xrightarrow{a_n} q_n$
  ist erfolgreich, wenn
  1. $q_0 \in I$
  2. $q_n \in F$
- unendlicher Pfad
  Bei unendlichen Pfaden gibt es keinen letzten Zustand q_n. Deshalb fordern wir, daß die Menge F unendlich oft durchlaufen wird, das heißt der unendliche Pfad $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \dots$ heißt erfolgreich, falls
  1. $q_0 \in I$
  2. Es gibt unendlich viele i mit $q_i \in F$

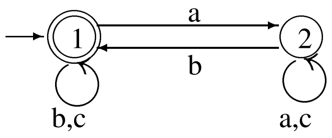
Die von $\mathcal{A}$ akzeptierte ω-Sprache ist

$$L_\omega(\mathcal{A}) = \{\alpha \in \Sigma^\omega \mid \alpha \text{ ist Beschriftung eines erfolgreichen Pfads}\}$$

Solche Sprachen heißen Büchi-erkennbar. □

Beispiel 4.5
Es sei $\Sigma = \{a,\ b,\ c\}$.

1. $\mathcal{A}_1$:



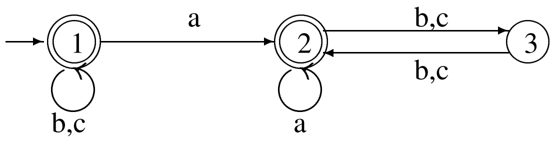
$$L_\omega(\mathcal{A}_1) = \{\alpha \in \Sigma^\omega \mid \text{nach jedem } a \text{ in } \alpha \text{ kommt irgendwann ein } b \text{ in } \alpha\}$$

denn Wörter, die kein a enthalten, sind auf jeden Fall in $L_\omega(\mathcal{A}_1)$. Jedes a führt in den Zustand 2 über. Der akzeptierende Zustand 1 wird erst wieder erreicht, wenn ein b kommt.

2.

$$L_2 = \{\alpha \in \Sigma^\omega \mid \quad \text{zwischen zwei } a's \text{ in } \alpha \text{ befindet sich eine gerade Anzahl von } b's \text{ und } c's\}.$$

Es reicht, zwei aufeinanderfolgende a's zu betrachten.
$\mathcal{A}_2$:



$L_\omega(\mathcal{A}_2) = L_2$.

□

Um die von Büchi-Automaten akzeptierten Sprachen genauer zu untersuchen, verwenden wir die folgende Abkürzung. Seien $\mathcal{A} = (Q,\ \Sigma,\ I,\ \Delta, F)$ ein Büchi-Automat und $p,\ q \in Q$. Wir definieren

$$L_{p,q} := \{w \in \Sigma^* \mid w \text{ ist Beschriftung eines endlichen Pfads in } \mathcal{A} \text{ von } p \text{ nach } q\}$$

Offenbar sind die Sprachen $L_{p,q}$ regulär, denn sie werden von einem endlichen Automaten (mit $I = \{p\}$ und $F = \{q\}$) akzeptiert.

Lemma 4.6

$$L_\omega(\mathcal{A}) = \bigcup_{\substack{i \in I \\ f \in F}} L_{i,f} \cdot L_{f,f}^\omega$$

Beweis

"$\subseteq$" Es sei $\alpha = a_1 a_2 a_3 \ldots \in L_\omega(\mathcal{A})$, das heißt es gibt einen unendlichen erfolgreichen Pfad $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \ldots$ mit $q_0 \in I$ und unendlich vielen i mit $q_i \in F$. Da F endlich ist gibt es ein $f \in F$, das unendlich oft erreicht wird. Es gibt also eine unendliche Folge $0 \leq i_1 < i_2 < i_3 < \ldots$ mit $q_{i_\nu} = f$. Somit gilt

$$\begin{array}{rcl} a_1 a_2 \ldots a_{i_1} & \in & L_{q_0,f} \\ a_{i_\nu+1} \ldots a_{i_{\nu+1}} & \in & L_{f,f} \end{array}$$

Damit ist $\alpha \in L_{q_0,f} \cdot L^{\omega}_{f,f}$.

"$\supseteq$" Es sei

$$\alpha = w_0 w_1 w_2 w_3 \ldots \in \bigcup_{\substack{i \in I \\ f \in F}} L_{i,f} \cdot L^{\omega}_{f,f},$$

das heißt für ein Paar (i,f) ist $w_0 \in L_{i,f}$ und $w_j \in L_{f,f}$ für $j \geq 1$.
Damit ist

$$i \xrightarrow{w_0} f \xrightarrow{w_1} f \xrightarrow{w_2} f \xrightarrow{w_3} \ldots$$

ein erfolgreicher Pfad, also ist $\alpha = w_0 w_1 w_2 w_3 \ldots \in L_\omega(\mathcal{A})$.

□

Das nächste Lemma beschreibt Abschlußeigenschaften für Büchi-erkennbare Sprachen.

Lemma 4.7

1. Ist $U \subseteq \Sigma^*$ regulär, so ist U^ω Büchi-erkennbar.
2. Ist $U \subseteq \Sigma^*$ regulär und L Büchi-erkennbar, so ist $U \cdot L$ Büchi-erkennbar.
3. Sind L_1 und $L_2 \subseteq \Sigma^\omega$ Büchi-erkennbar, so sind auch $L_1 \cup L_2$ und $L_1 \cap L_2$ Büchi-erkennbar.

Beweis

1. Es gilt $U^\omega = \{\alpha \in \Sigma^\omega \mid \alpha = u_1 u_2 u_3 \ldots,\ u_i \in U \backslash \{\varepsilon\}\}$. Mit U ist auch $U \backslash \{\varepsilon\}$ regulär. Man sieht leicht, daß es einen Automaten für $U \backslash \{\varepsilon\}$ gibt mit den Eigenschaften
   - $\mathcal{A} = (Q,\ \Sigma,\ q_0,\ \Delta,\ F)$, das heißt $\mathcal{A}$ hat genau einen Anfangszustand (der kein Endzustand ist, weil $\varepsilon \notin U \backslash \{\varepsilon\}$).
   - Für alle $a \in \Sigma$, $q \in Q$ gilt $(q,a,q_0) \notin \Delta$, das heißt q_0 ist von anderen Zuständen aus nicht erreichbar.

   Wir definieren nun $\mathcal{A}' := (Q,\ \Sigma,\ q_0,\ \Delta',\ \{q_0\})$, wobei

   $$\Delta' = \Delta \cup \{(q,a,q_0) \mid \exists f \in F :\ (q,a,f) \in \Delta\}.$$

   Man sieht leicht, daß $L_\omega(\mathcal{A}') = U^\omega$ gilt. Dabei ist die Idee, daß jedes Erreichen des einzigen Endzustandes q_0 den Beginn eines neuen Wortes u_i aus U bedeutet.
2. Seien $\mathcal{A} = (Q_1,\ \Sigma,\ I_1,\ \Delta_1,\ F_1)$ mit $L(\mathcal{A}) = U$ und $\mathcal{B} = (Q_2,\ \Sigma,\ I_2,\ \Delta_2,\ F_2)$ mit $L_\omega(\mathcal{B}) = L$. Ohne Einschränkung sei $Q_1 \cap Q_2 = \emptyset$. Wir definieren

   $$\mathcal{C} := (Q_1 \cup Q_2,\ \Sigma,\ I_1,\ \Delta',\ F_2),$$

   wobei $\Delta' := \Delta_1 \cup \Delta_2 \cup \{(q,a,q') \mid \exists f \in F_1 :\ ((q,a,f) \in \Delta_1\ \wedge\ q' \in I_2)\}$. Man sieht leicht, daß $L_\omega(\mathcal{C}) = U \cdot L^\omega$ gilt.

3. Der Beweis für den Abschluß der Büchi-erkennbaren Sprachen unter Vereinigung läuft analog zu dem für reguläre Sprachen.
   Für den Abschluß unter Schnitt seien $\mathcal{A}_1 = (Q_1,\ \Sigma,\ I_1,\ \Delta_1,\ F_1)$ mit $L_\omega(\mathcal{A}_1) = L_1$ und $\mathcal{A}_2 = (Q_2,\ \Sigma,\ I_2,\ \Delta_2,\ F_2)$ mit $L_\omega(\mathcal{A}_2) = L_2$. Gesucht ist ein Büchi-Automat $\mathcal{B}$ mit $L_\omega(\mathcal{B}) = L_1 \cap L_2$. Sei
   $$\mathcal{B} := (Q_1 \times Q_2 \times \{0,1,2\},\ \Sigma,\ I_1 \times I_2 \times \{0\},\ \Delta,\ F)$$
   $$\begin{aligned} \text{mit } \Delta = \{ \ & ((q_1,\ q_2,\ i),\ a,\ (q_1',\ q_2',\ j)) \mid \\ & \bullet\ (q_1,\ a,\ q_1') \in \Delta_1 \text{ und } (q_2,\ a,\ q_2') \in \Delta_2 \\ & \bullet\ i = 0 \text{ und } q_1' \in F_1 \Rightarrow j = 1 \\ & \bullet\ i = 1 \text{ und } q_2' \in F_2 \Rightarrow j = 2 \\ & \bullet\ i = 2 \Rightarrow j = 0 \\ & \bullet\ \text{sonst } i = j\ \} \end{aligned}$$
   Man beginnt also mit einer 0 in der dritten Komponente. Wenn man das erste Mal ein $f_1 \in F_1$ durchläuft, wird die dritte Komponente 1. Wenn man daraufhin das nächstemal ein $f_2 \in F_2$ durchläuft, so wird die dritte Komponente 2 und im nächsten Schritt wieder zu 0.
   Durchläuft man unendlich oft Elemente aus F_1 in der ersten Komponente und unendlich oft Elemente aus F_2 in der zweiten Komponente, so erreicht man unendlich oft einen Zustand mit dritter Komponente 2.
   Daher definieren wir $F := Q_1 \times Q_2 \times \{2\}$.

□

Satz 4.8 (Büchi, 1962)

1. Eine ω-Sprache $L \subseteq \Sigma^\omega$ ist Büchi-erkennbar genau dann, wenn es reguläre Sprachen $U_1,\ \ldots,\ U_n,\ V_1,\ \ldots,\ V_n \subseteq \Sigma^*$ gibt mit
   $$L = \bigcup_{i=1}^{n} U_i \cdot V_i^\omega.$$
2. Man kann ohne Einschränkung davon ausgehen, daß $V_i \cdot V_i = V_i$ gilt.

Beweis
"$\Rightarrow$"

1. mit Lemma 4.6 ist
   $$L = \bigcup_{\substack{q_0 \in I \\ f \in F}} L_{q_0,f} \cdot L_{f,f}^\omega$$
2. $L_{f,f} \cdot L_{f,f} = L_{f,f}$

□

"$\Leftarrow$"

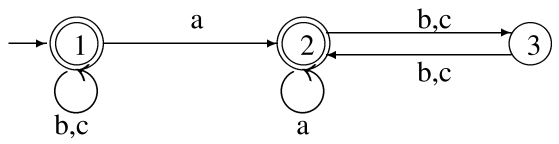
1. Diese Richtung folgt mit Lemma 4.7 (V_i^ω ist Büchi-erkennbar, $U_iV_i^\omega$ ist Büchi-erkennbar und Büchi-erkennbare Sprachen sind bezüglich Vereinigung abgeschlossen).

Wegen dieses engen Zusammenhangs zwischen Büchi-erkennbaren Sprachen und regulären Sprachen heißen Büchi-erkennbare Sprachen auch ω-regulär.
Sind U_i, V_i durch reguläre Ausdrücke gegeben, so heißt die Darstellung

$$L = \bigcup_{i=1}^{n} U_i V_i^{\omega}$$

ω-regulärer Ausdruck für L.

Beispiel 4.9
Sei $\mathcal{A}_2$ der endliche Automat aus Beispiel 4.5



$$\begin{aligned}
L_{1,1} &= (b\cup c)^* \\
L_{1,2} &= (b\cup c)^* a L_{2,2} \\
L_{2,2} &= (a\cup (b\cup c)(b\cup c))^* \\
L_{\omega}(\mathcal{A}_2) &= L_{1,1}\cdot L_{1,1}^{\omega} \;\cup\; L_{1,2}\cdot L_{2,2}^{\omega} \\
&= L_{1,1}^{\omega} \;\cup\; L_{1,2}\cdot L_{2,2}^{\omega} \\
&= ((b\cup c)^*)^{\omega} \;\cup\; (b\cup c)^*\cdot a\cdot L_{2,2}\cdot L_{2,2}^{\omega} \\
&= (b\cup c)^{\omega} \;\cup\; (b\cup c)^*\cdot a\cdot (a\cup (b\cup c)(b\cup c))^{\omega}
\end{aligned}$$

□

Eine weitere Konsequenz aus Lemma 4.6 ist, daß man für einen Büchi-Automaten entscheiden kann, ob er eine nicht-leere Sprache akzeptiert: Man muß ein $i \in I$ und ein $f \in F$ finden mit $L_{i,f} \neq \emptyset$ und $L_{f,f}\setminus\{\varepsilon\} \neq \emptyset$. Die von dem Automaten akzeptierte Sprache ist nicht-leer genau dann, wenn ein solches Paar $i,\ f$ existiert. Da $L_{i,f}$ und $L_{f,f}\setminus\{\varepsilon\}$ reguläre Sprachen sind, die von dem gegebenen Automaten akzeptiert werden (bei geeigneter Wahl der Anfangs- und Endzustände), ist entscheidbar, ob es $i,\ f$ mit dieser Eigenschaft gibt.

Satz 4.10

1. Das Leerheitsproblem für Büchi-Automaten ist entscheidbar.
2. Ist L eine nicht-leere Büchi-erkennbare ω-Sprache, so enthält L ein schließlich periodisches Wort, das heißt ein Wort der Form $uvvv\ldots,\ u \in \Sigma^*,\ v \in \Sigma^+$.

Beweis

1. Die erste Aussage wurde bereits durch die obige Argumentation gezeigt.
2. Ist $L \neq \emptyset$, so gibt es $i,\ f$ im Automaten mit $L_{i,f} \neq \emptyset$ und $L_{f,f}\setminus\{\varepsilon\} \neq \emptyset$. Seien $u \in L_{i,f}$ und $v \in L_{f,f}\setminus\{\varepsilon\}$ beliebig. Offenbar ist $uvvv\ldots \in L$.

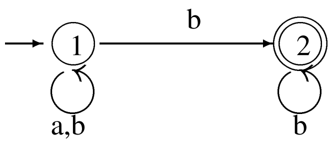
□

Für reguläre Sprachen kann man das Äquivalenzproblem ($L(\mathcal{A}_1) = L(\mathcal{A}_2)$?) auf das Leerheitsproblem reduzieren: $L_1 = L_2$ gdw $(L_1 \cap \overline{L_2}) \cup (L_2 \cap \overline{L_1}) = \emptyset$. Dies ist möglich, weil die regulären Sprachen unter Vereinigung, Schnitt und Komplement abgeschlossen sind. Bei ω-regulären Sprachen fehlt uns für ein analoges Vorgehen noch der Abschluß unter Komplementbildung.
Für reguläre Sprachen wurde der Abschluß unter Komplement in zwei Schritten nachvollzogen:

1. überführe den Automaten in einen deterministischen endlichen Automaten
2. betrachte als neue Endzustandmenge $Q \backslash F$, das heißt die Menge der Zustände, die zuvor keine Endzustände waren.

Bei ω-regulären Sprachen funktionieren sowohl 1. als auch 2. nicht.

Beispiel 4.11
für eine ω-reguläre Sprache, die nicht von einem deterministischen *Büchi-Automaten akzeptiert wird*
Sei $\Sigma = \{a,\ b\}$. Der folgende nicht-deterministische Automat akzeptiert die ω-reguläre Sprache $L = (a \cup b)^* b^\omega$.



Behauptung: Diese Sprache kann nicht von einem deterministischen Büchi-Automaten akzeptiert werden.
Beweis: Angenommen $\mathcal{A} = (Q,\ \Sigma,\ q_0,\ \delta,\ F)$ ist ein deterministischer Büchi-Automat mit $L_\omega(\mathcal{A}) = L$. $\delta: \ Q \times \Sigma \to Q$ ist also eine Funktion. Da $ab^\omega \in L$ gilt, gibt es ein $k_1 > 0$ und $f_1 \in F$ mit

$$q_0 \xrightarrow{ab^{k_1}} f_1$$

Da $ab^{k_1}ab^\omega \in L$ und da $\mathcal{A}$ deterministisch ist gibt es ein $k_2 > 0$ und $f_2 \in F$ mit

$$q_0 \xrightarrow{ab^{k_1}} f_1 \xrightarrow{ab^{k_2}} f_2$$

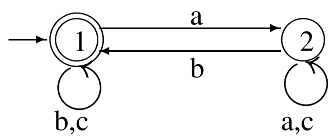
Iteriert man dieses Argument, so erhält man $k_1,\ k_2,\ k_3,\ \ldots > 0$ und $f_1,\ f_2, f_3,\ \ldots \in F$ mit

$$q_0 \xrightarrow{ab^{k_1}} f_1 \xrightarrow{ab^{k_2}} f_2 \xrightarrow{ab^{k_3}} f_3 \xrightarrow{ab^{k_4}} \ldots$$

Damit ist $ab^{k_1}ab^{k_2}ab^{k_3} \ldots \in L_\omega(\mathcal{A})$, da es einen unendlichen erfolgreichen Pfad in $\mathcal{A}$ gibt mit dieser Beschriftung. Aber $ab^{k_1}ab^{k_2}ab^{k_3} \ldots \notin L$, denn dieses Wort enthält unendlich viele a's. *Widerspruch*
□

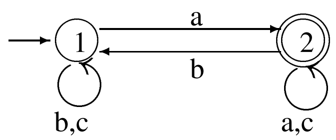
Beispiel 4.12
Auch bei deterministischen Büchi-Automaten liefert Vertauschen von Endzuständen und nicht-Endzuständen *nicht* den Automaten für die Komplementsprache.

Betrachtet man zum Beispiel den deterministischen Büchi-Automaten $\mathcal{A}_1$ aus Beispiel 4.5



$$L_\omega(\mathcal{A}_1) = \{\alpha \in \Sigma^\omega \mid \text{nach jedem } a \text{ in } \alpha \text{ kommt irgendwann ein } b \text{ in } \alpha\}$$

Dann gilt für den Komplement-Automaten $\overline{\mathcal{A}_1}$



$L_\omega(\mathcal{A}_1) \cap L_\omega(\overline{\mathcal{A}_1}) \neq \emptyset$, da zum Beispiel $(ab)^\omega$ in beiden Sprachen enthalten ist.

□

Wie Beispiel 4.11 zeigt, ist die Klasse der von Büchi-Automaten akzeptierten Sprachen kleiner als die Klasse der ω-regulären Sprachen.

Satz 4.13
Für $L \subseteq \Sigma^\omega$ sind äquivalent

1. L wird von einem *deterministischen* Büchi-Automaten akzeptiert
2. $L = \lim U$ für eine reguläre Sprache $U \subseteq \Sigma^*$

Beweis
Es sei $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ ein deterministischer endlicher Automat. Dann gilt

Behauptung: Für $\alpha \in \Sigma^\omega$ sind äquivalent

1. $\alpha \in L_\omega(\mathcal{A})$
2. $\alpha \in \lim L(\mathcal{A})$, das heißt unendlich viele Anfangsstücke von α sind in $L(\mathcal{A})$

denn (Beweis der Behauptung):

"$1 \Rightarrow 2$" Es sei $\alpha \in L_\omega(\mathcal{A})$, dann folgt daraus, daß es $f_1, f_2, f_3, \ldots \in F$ und $u_1, u_2, u_3, \ldots \in \Sigma^+$ mit $\alpha = u_1u_2u_3\ldots$ und

$$q_0 \xrightarrow{u_1} f_1 \xrightarrow{u_2} f_2 \xrightarrow{u_3} f_3 \xrightarrow{u_4} \ldots$$

gibt. Damit folgt $u_1, u_1u_2, u_1u_2u_3, \ldots \in L(\mathcal{A})$. Also liegen unendlich viele Anfangsstücke von α in $L(\mathcal{A})$.

"$2 \Rightarrow 1$" Es seien $u_1, u_1u_2, u_1u_2u_3, \ldots$ mit $u_i \in \Sigma^+$ für $i \in \mathbb{N}$ die unendlich vielen Anfangsstücke von α mit $u_1 \ldots u_i \in L(\mathcal{A})$. Da $\mathcal{A}$ deterministisch ist, heißt dies, daß es $f_1, f_2, f_3, \ldots \in F$ gibt mit

$$q_0 \xrightarrow{u_1} f_1 \xrightarrow{u_2} f_2 \xrightarrow{u_3} f_3 \xrightarrow{u_4} \ldots$$

Es folgt $\alpha = u_1u_2u_3\ldots \in L_\omega(\mathcal{A})$, da es einen unendlichen erfolgreichen Pfad mit der Beschriftung α in $\mathcal{A}$ gibt.

Beachte: Bei nicht-deterministischen Automaten könnte es die folgende Situation geben (vergleiche auch Beispiel 4.11):

$$q_0 \xrightarrow{u_1} f_1$$
$$q_0 \xrightarrow{u_1} q_1 \notin F \xrightarrow{u_2} f_2$$

Damit ist die Behauptung gezeigt und es ergibt sich der obige Satz.

□

Korollar 4.14
Die Klasse der von deterministischen Büchi-Automaten akzeptierten Sprachen ist nicht abgeschlossen bezüglich Komplementbildung.

Beweis
In Beispiel 4.11 haben wir gesehen, daß $L = (a \cup b)^* b^\omega$ nicht von einem deterministischen Büchi-Automaten akzeptiert wird.
Man sieht leicht, daß

$$\overline{L} = \{a,b\}^\omega \backslash L$$

die ω-Sprache ist, welche alle ω-Wörter α mit der Eigenschaft " α enthält unendlich viele a's " enthält. Daher ist

$$\overline{L} = \lim(b^* a)^*.$$

Diese Sprache wird aber von einem deterministischen Büchi-Automaten akzeptiert, da $\overline{L} = \lim U$ ist für eine reguläre Sprache U.

□

Die Klasse der ω-regulären Sprachen ist unter Komplement abgeschlossen. Dies ist schwieriger zu zeigen als für reguläre Sprachen.

Hauptsatz 4.15
Ist $L \subseteq \Sigma^\omega$ ω-regulär, so auch $\overline{L} = \Sigma^\omega \backslash L$.

□

Um diesen Satz zu beweisen stellen wir L und $\overline{L}$ als endliche Vereinigung von ω-Sprachen U, V dar. Diese Sprachen werden die Klassen einer Kongruenzrelation $\sim_{\mathcal{A}}$ von endlichem Index sein.
Es sei dazu $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein Büchi-Automat mit $L = L_\omega(\mathcal{A})$. Wir schreiben $p \overset{F}{\underset{w}{\to}} q$ um auszudrücken, daß es in $\mathcal{A}$ einen Pfad von p nach q gibt mit der Beschriftung w, der mindestens einen Zustand aus F enthält.
Die Mengen

$$L_{p,q}^F := \{w \mid p \overset{F}{\underset{w}{\to}} q\}$$

sind offenbar regulär, da sie sich als Vereinigung regulärer Sprachen darstellen lassen:

$$L_{p,q}^F = \bigcup_{f \in F} L_{p,f} \cdot L_{f,q}$$

Definition 4.16
$\sim_{\mathcal{A}}$ ist auf Σ^* wie folgt definiert:

$$u \sim_{\mathcal{A}} v \quad \text{gdw} \quad \forall\, p,q \in Q: \quad 1)\ p \underset{u}{\rightarrow} q \ \ gdw \ \ p \underset{v}{\rightarrow} q \qquad 2)\ p \overset{F}{\underset{u}{\rightarrow}} q \ \ gdw \ \ p \overset{F}{\underset{v}{\rightarrow}} q$$

□

Lemma 4.17
$\sim_{\mathcal{A}}$ ist eine Kongruenzrelation, die endlichen Index hat (das heißt sie hat endlich viele $\sim_{\mathcal{A}}$-Kongruenzklassen).

Beweis

1. $\sim_{\mathcal{A}}$ ist Kongruenzrelation
   Offenbar ist $\sim_{\mathcal{A}}$ Äquivalenzrelation (dies sieht man an dem *gdw* auf der rechten Seite der Definition).
   Kongruenz: $u \sim_{\mathcal{A}} v \Rightarrow \ \forall x,y \in \Sigma^*: \ xuy \sim_{\mathcal{A}} xvy$
   Es gelte $u \sim_{\mathcal{A}} v$. Angenommen es gilt $p \overset{F}{\underset{xuy}{\rightarrow}} q$, es gibt also p' und q' mit $p \underset{x}{\rightarrow} p' \underset{u}{\rightarrow} q' \underset{y}{\rightarrow} q$
   1. Fall $p \overset{F}{\underset{x}{\rightarrow}} p'$, das heißt $f \in F$ kommt in $p \underset{x}{\rightarrow} p'$ vor.
   Wegen $u \sim_{\mathcal{A}} v$ folgt aus $p' \underset{u}{\rightarrow} q'$ auch $p' \underset{v}{\rightarrow} q'$. Damit gibt es einen Pfad $p \overset{F}{\underset{x}{\rightarrow}} p' \underset{v}{\rightarrow} q' \underset{y}{\rightarrow} q$, das heißt $p \overset{F}{\underset{xvy}{\rightarrow}} q$.
   2. Fall $q' \overset{F}{\underset{y}{\rightarrow}} q$ (analog zum 1. Fall)
   3. Fall $p' \overset{F}{\underset{u}{\rightarrow}} q'$
   Wegen $u \sim_{\mathcal{A}} v$ folgt $p' \overset{F}{\underset{v}{\rightarrow}} q'$. Damit gibt es $p \underset{x}{\rightarrow} p' \overset{F}{\underset{v}{\rightarrow}} q' \underset{y}{\rightarrow} q$, das heißt $p \overset{F}{\underset{xvy}{\rightarrow}} q$.
   Die anderen Fälle ($p \overset{F}{\underset{xvy}{\rightarrow}} q \Rightarrow p \overset{F}{\underset{xuy}{\rightarrow}} q$, $p \underset{xuy}{\rightarrow} q \Leftrightarrow p \underset{xvy}{\rightarrow} q$) können entsprechend behandelt werden.

2. $\sim_{\mathcal{A}}$ hat endlichen Index
   Die $\sim_{\mathcal{A}}$-Klasse eines Wortes w ist charakterisiert durch das folgende Paar von Mengen:
   $$(\{(p,q) \mid p \underset{w}{\rightarrow} q\},\ \{(p,q) \mid p \overset{F}{\underset{w}{\rightarrow}} q\})$$
   Es gibt daher maximal $2^{|Q\times Q|} \cdot 2^{|Q\times Q|}$ verschiedene Klassen.

□

Wie sieht nun die $\sim_{\mathcal{A}}$-Klasse $[w]$ eines Wortes w aus?

Lemma 4.18

1.
$$[w] = \bigcap_{\substack{p,q\in Q\\ w\in L_{p,q}}} L_{p,q} \cap \bigcap_{\substack{p,q\in Q\\ w\notin L_{p,q}}} \overline{L_{p,q}} \cap \bigcap_{\substack{p,q\in Q\\ w\in L^F_{p,q}}} L^F_{p,q} \cap \bigcap_{\substack{p,q\in Q\\ w\notin L^F_{p,q}}} \overline{L^F_{p,q}}$$

2. Insbesondere ist daher $[w] \subseteq \Sigma^*$ eine reguläre Sprache.

Beweis

1. "$\subseteq$" Sei $u \in [w]$, das heißt $u \sim_{\mathcal{A}} w$. Ist $w \in L_{p,q}$ ($\overline{L_{p,q}}$, $L^F_{p,q}$, $\overline{L^F_{p,q}}$), so folgt wegen $u \sim_{\mathcal{A}} w$ (das heißt $p \underset{u}{\rightarrow} q \Leftrightarrow p \underset{w}{\rightarrow} q$), daß $u \in L_{p,q}$ ($\overline{L_{p,q}}$, $L^F_{p,q}$, $\overline{L^F_{p,q}}$)

   "$\supseteq$" Sei u aus dem Durchschnitt auf der rechten Seite. Es ist zu zeigen, daß $u \sim_{\mathcal{A}} w$ gilt.

   – Aus $p \underset{w}{\rightarrow} q$ folgt $w \in L_{p,q}$ ($\Rightarrow L_{p,q}$ liegt im Schnitt) und damit $u \in L_{p,q}$, das heißt $p \underset{u}{\rightarrow} q$.
   – Aus $p \underset{w}{\not\rightarrow} q$ folgt $w \in \overline{L_{p,q}}$ und damit ist auch $u \in \overline{L_{p,q}}$, das heißt $p \underset{u}{\not\rightarrow} q$.
   – Für $\overset{F}{\underset{w}{\rightarrow}}$ entsprechend.

2. folgt unmittelbar aus 1., da $L_{p,q}$ und $L^F_{p,q}$ regulär sind und reguläre Sprachen unter Schnitt abgeschlossen sind.

□

Hauptsatz 4.15 ergibt sich sehr leicht aus dem folgenden Satz:

Satz 4.19
Es sei $\mathcal{A}$ ein Büchi-Automat.

1. Für jedes Paar U, V von $\sim_{\mathcal{A}}$-Klassen gilt:
   (a) $U \cdot V^\omega \cap L_\omega(\mathcal{A}) \neq \emptyset \;\Rightarrow\; U \cdot V^\omega \subseteq L_\omega(\mathcal{A})$
   (b) $U \cdot V^\omega \cap \overline{L_\omega(\mathcal{A})} \neq \emptyset \;\Rightarrow\; U \cdot V^\omega \subseteq \overline{L_\omega(\mathcal{A})}$
2. Für jedes $\alpha \in \Sigma^\omega$ existieren $\sim_{\mathcal{A}}$-Klassen U, V mit $\alpha \in U \cdot V^\omega$.

□

Wir überlegen zunächst, warum hieraus folgt, daß $\overline{L_\omega(\mathcal{A})}$ ω-regulär ist.
Es folgt aus 1. und 2. des Satzes, daß

- $$L_\omega(\mathcal{A}) = \bigcup_{\substack{U,V \ \sim_{\mathcal{A}}-Klassen \\ U\cdot V^\omega \subseteq L_\omega(\mathcal{A})}} U \cdot V^\omega$$

  denn
  "⊇" eine Vereinigung über Teilmengen von $L_\omega(\mathcal{A})$ wird immer eine Teilmenge von $L_\omega(\mathcal{A})$ sein.
  "⊆" Mit 2. des Satzes existiert zu jedem $\alpha \in L_\omega(\mathcal{A})$ ein Paar $U,\ V$ von $\sim_{\mathcal{A}}$-Klassen mit $\alpha \in U \cdot V^\omega$. Mit 1. gilt dann $U \cdot V^\omega \subseteq L_\omega(\mathcal{A})$ und damit liegt $U \cdot V^\omega$ dann komplett in der Vereinigung.

- $$\overline{L_\omega(\mathcal{A})} = \overbrace{\bigcup_{\substack{U,V \ \sim_{\mathcal{A}}-Klassen \\ U\cdot V^\omega \subseteq \overline{L_\omega(\mathcal{A})}}}}^{\substack{\text{endlich, da } \sim_{\mathcal{A}} \\ \text{endl. Index hat}}} U \cdot V^\omega$$

Da alle $\sim_{\mathcal{A}}$-Klassen regulär sind, ist mit Satz 4.8 $\overline{L_\omega(\mathcal{A})}$ ω-regulär.

Wir verwenden zum Beweis von Satz 4.19 ein kombinatorisches Resultat, das auch in anderen Bereichen nützlich ist. Dazu zunächst eine Definition.

Definition 4.20
Für eine Menge M bezeichne $[M]^2$ die Menge aller zweielementigen Teilmengen von M. Es sei nun

$$[M]^2 = A_1 \,\dot\cup\, A_2 \,\dot\cup\, \ldots \,\dot\cup\, A_n$$

eine Zerlegung von $[M]^2$ in endlich viele disjunkte Klassen.
Eine Teilmenge X von M heißt <u>homogen</u> für die Partition, falls es ein i gibt mit $[X]^2 \subseteq A_i$.
□

Beispiel

$$[\boldsymbol{N}]^2 = A \,\dot\cup\, B,$$

wobei

$$A = \{\{i,\ j\} \mid i \neq j \text{ und } i \equiv j(2)\}$$

$$B = \{\{i,\ j\} \mid i \neq j \text{ und } i \not\equiv j(2)\}$$

$i \equiv j(2)$ heißt i ist kongruent zu j modulo 2.
Die Mengen $G = \{i \in \boldsymbol{N} \mid i \text{ gerade}\}$ und $U = \{i \in \boldsymbol{N} \mid i \text{ ungerade}\}$ sind beide homogen, da $[G]^2 \subseteq A$ und $[U]^2 \subseteq A$.

Satz 4.21 (Ramsey)
Es sei $[\boldsymbol{N}]^2 = A_1 \dot\cup \ldots \dot\cup A_n$ eine disjunkte Zerlegung von $[\boldsymbol{N}]^2$. Dann gibt es eine <u>unendliche</u> Teilmenge $X \subseteq \boldsymbol{N}$, die homogen für diese Zerlegung ist.

□

Der Beweis findet sich zum Beispiel in J. G. Rosenstein: "Linear Orderings", Academic Press, 1982, Seiten 111, 112.

Beweis von Satz 4.19

1. Es sei $\alpha \in UV^\omega \cap L_\omega(\mathcal{A})$, Das heißt

   (a) $\alpha = uv_1v_2v_3\ldots$, mit $u \in U$ und $v_i \in V \backslash \{\varepsilon\}$

   (b) Es gibt einen erfolgreichen unendlichen Pfad

   $$q_0 \xrightarrow{u} q_1 \xrightarrow{v_1} q_2 \xrightarrow{v_2} q_3 \xrightarrow{v_3} \ldots$$

   mit $q_0 \in I$. Da dieser Pfad erfolgreich ist werden unendlich oft Zustände aus F erreicht, das heißt es gibt unendlich viele $i \geq 1$, so daß sogar

   $$q_i \xrightarrow[v_i]{F} q_{i+1}$$

   Für den Beweis von Teil (a) sei nun $\beta \in UV^\omega$. Wir müssen $\beta \in L_\omega(\mathcal{A})$ zeigen.
   Es gilt $\beta = u'v'_1v'_2v'_3\ldots$ mit $u' \in U$, $v'_i \in V$. U, V sind $\sim_\mathcal{A}$-Klassen. Daher gilt $u \sim_\mathcal{A} u'$ und $v_i \sim_\mathcal{A} v'_i$. Es folgt daraus, daß es einen Pfad

   $$q_0 \xrightarrow{u'} q_1 \xrightarrow{v'_1} q_2 \xrightarrow{v'_2} q_3 \xrightarrow{v'_3} \ldots$$

   gibt, auf dem für unendlich viele i gilt

   $$q_i \xrightarrow[v_i]{F} q_{i+1}$$

   Damit ist der Pfad erfolgreich, das heißt $\beta = u'v'_1v'_2v'_3\ldots \in L_\omega(\mathcal{A})$.
   Teil (b) folgt unmittelbar: Es sei $\alpha \in \overline{L_\omega(\mathcal{A})} \cap UV^\omega$. Angenommen $\beta \in L_\omega(\mathcal{A}) \cap UV^\omega$.
   Mit Teil (a) folgt nun $UV^\omega \subseteq L_\omega(\mathcal{A})$, also $\alpha \in L_\omega(\mathcal{A})$. Dies ist ein Widerspruch.

2. Es sei $\alpha \in \Sigma^\omega$. Zusammen mit $\sim_\mathcal{A}$ liefert α eine <u>Zerlegung von $[\boldsymbol{N}]^2$</u>.
   Es seien $U_1, U_2, \ldots, U_n$ die (endlich vielen) $\sim_\mathcal{A}$-Klassen.

   $$A_\nu := \{\{i,j\} \mid i < j \text{ und } \alpha(i+1,j) \in U_\nu\}$$

   wobei $\nu = 1, \ldots, n$.
   Die Klassen einer Kongruenzrelation liefern eine disjunkte Zerlegung

   $$[\boldsymbol{N}]^2 = A_1 \dot\cup \ldots \dot\cup A_n$$

   Mit Satz 4.21 (Ramsey) gibt es eine unendliche Menge $X \subseteq \boldsymbol{N}$ für diese Zerlegung, die homogen ist. Es gibt also ein ν, $1 \leq \nu \leq n$, so daß $[X]^2 \subseteq A_\nu$, das heißt für $i,\ j \in X$

mit $i < j$ gilt $\alpha(i+1, j) \in U_\nu$.
Da X unendlich ist gibt es eine unendliche Folge $i_1,\ i_2,\ i_3,\ \ldots$ von Elementen von X mit $i_j + 1 < i_{j+1}$. Damit gilt

$$\alpha(i_j + 1, i_{j+1}) \in U_\nu \setminus \{\varepsilon\}$$

Es sei U die $\sim_{\mathcal{A}}$-Klasse von $\alpha(0, i_1)$. Dann ist

$$\alpha = \alpha(0, i_1)\alpha(i_1 + 1, i_2)\alpha(i_2 + 1, i_3)\ldots \in UU_\nu^\omega$$

□

Damit ist der Beweis abgeschlossen, daß die Klasse der Büchi-erkennbaren Sprachen unter Komplement abgeschlossen ist.

Korollar 4.22
Zu einem Büchi-Automaten $\mathcal{A}$ kann man effektiv einen Büchi-Automaten $\mathcal{B}$ konstruieren mit

$$L_\omega(\mathcal{B}) = \overline{L_\omega(\mathcal{A})}.$$

Beweis

1. Die $\sim_{\mathcal{A}}$-Klassen (bzw. endliche Automaten dafür) können effektiv konstruiert werden. Lemma 4.18 zeigt, wie man sie aus den Sprachen $L_{p,q}$ und $L^F_{p,q}$ erhält.
2. Für ein Paar $U,\ V$ von $\sim_{\mathcal{A}}$-Klassen ist mit Satz 4.10 entscheidbar, ob
$$U \cdot V^\omega \cap L_\omega(\mathcal{A}) = \emptyset.$$
3. Da
$$\overline{L_\omega(\mathcal{A})} = \bigcup_{\substack{U,V\ \sim_{\mathcal{A}}-Klassen \\ UV^\omega \cap L_\omega(\mathcal{A}) = \emptyset}} U \cdot V^\omega$$
kann man den Büchi-Automaten für die Vereinigung aus den Büchi-Automaten für die $U \cdot V^\omega$ erhalten.

□

Korollar 4.23
Das Äquivalenzproblem für Büchi-Automaten ist entscheidbar.

Beweis

$$L_\omega(\mathcal{A}_1) = L_\omega(\mathcal{A}_2) \ \text{ gdw } \ (L_\omega(\mathcal{A}_1) \setminus L_\omega(\mathcal{A}_2) \cup (L_\omega(\mathcal{A}_2) \setminus L_\omega(\mathcal{A}_1)) = \emptyset$$

□

Wir wissen, daß deterministische Büchi-Automaten nicht alle ω-regulären Sprachen erkennen können. Wir betrachten nun ein modifiziertes Automatenmodell, in dem bereits die deterministischen Automaten alle ω-regulären Sprachen erkennen.

Definition 4.24
Ein Muller-Automat ist ein deterministischer endlicher Automat

$$\mathcal{A} = (Q,\ \Sigma,\ q_0,\ \delta,\ \mathcal{F}),$$

wobei

- $Q,\ \Sigma,\ q_0,\ \delta$ sind wie bei deterministischen Büchi-Automaten definiert
- $\mathcal{F} \subseteq 2^Q$, das heißt $\mathcal{F}$ ist eine Menge von Mengen von Endzuständen.

Ein unendlicher Pfad

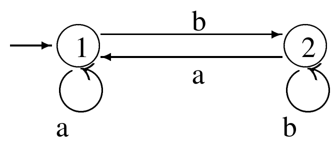
$$p_0 \xrightarrow{a_1} p_1 \xrightarrow{a_2} p_2 \xrightarrow{a_3} \ldots$$

in einem Muller-Automaten heißt erfolgreich, falls gilt

- er beginnt mit dem Anfangszustand, das heißt $p_0 = q_0$.
- $\{p \in Q \mid \text{es gibt unendlich viele } i \text{ mit } p = p_i\} \in \mathcal{F}$, das heißt die Menge der Zustände, die auf dem Pfad unendlich oft vorkommen, liegt in $\mathcal{F}$.

$L_\omega(\mathcal{A}) := \{\alpha \in \Sigma^\omega \mid \alpha \text{ beschriftet erfolgreichen Pfad in } \mathcal{A}\}$ □

Beispiel 4.25
Es sei $L = (a \cup b)^* b^\omega$. In Beispiel 4.11 haben wir gesehen, daß es keinen deterministischen Büchi-Automaten für L gibt.
Der folgende deterministische Muller-Automat akzeptiert L:



mit $\mathcal{F} = \{\{2\}\}$.
Damit 2 unendlich oft durchlaufen wird, 1 aber nicht unendlich oft, darf ab einer Stelle im Wort nur noch b vorkommen.
Beachte: Ein Büchi-Automat mit $F = \{2\}$ akzeptiert zum Beispiel auch $(ab)^\omega$.

□

Satz 4.26
Für eine ω-Sprache L sind äquivalent:

1. L ist ω-regulär, das heißt L wird von einem nicht-deterministischen Büchi-Automaten akzeptiert
2. L wird von einem (deterministischen) Muller-Automaten akzeptiert

Beweis

"2 ⇒ 1" Es sei $\mathcal{A} = (Q,\ \Sigma,\ q_0,\ \delta,\ \mathcal{F})$ ein deterministischer Muller-Automat, dann gilt

$$L = L_\omega(\mathcal{A}) = \bigcup_{F \subseteq \mathcal{F}} \left(\bigcap_{q \in F} \lim L_{q_0,q} \cap \bigcap_{q \in Q \setminus F} \overline{\lim L_{q_0,q}} \right)$$

$\lim L_{q_0,q}$ enthält die ω-Wörter α, die einen unendlichen Pfad beschriften, der q unendlich oft enthält.

Beachte: $\mathcal{A}$ ist deterministisch.

"1 ⇒ 2" Der Beweis dieser Richtung ist mindestens so aufwendig wie der Beweis von Satz 4.15. Wir lassen diesen Beweis daher weg.

□

Satz 4.27

Ist $L \subseteq \Sigma^\omega$ von einem deterministischen Muller-Automaten akzeptiert, so auch $\overline{L} := \Sigma^\omega \setminus L$.

Beweis

Es sei $L := L_\omega(\mathcal{A})$ für einen deterministischen Muller-Automaten $\mathcal{A} = (Q,\ \Sigma,\ q_0,\ \delta,\ \mathcal{F})$. Man sieht leicht, daß $\mathcal{B} = (Q,\ \Sigma,\ q_0,\ \delta,\ 2^Q \setminus \mathcal{F})$ die Sprache $\Sigma^\omega \setminus L$ akzeptiert.

□

Kapitel 5

Unendliche Wörter und logische Formeln

Ziel dieses Abschnitts ist es, eine Klasse von Formeln anzugeben, welche genau die ω-regulären Sprachen beschreibt.
Wie im Fall regulärer Sprachen endlicher Wörter genügen die Formeln der Logik erster Stufe nicht, um diese Sprachklasse zu beschreiben. Man benötigt zusätzlich Quantifizierung über einstellige Prädikate.

Definition 5.1
Formeln der Monadischen Logik zweiter Stufe eines Nachfolgers (S1S: secondorder logic of 1 successor) sind aufgebaut mit Hilfe von

- n einstelligen Prädikatensymbolen $P_1, \ldots, P_n$,
- einem einstelligen Funktionssymbol s,
- einem Konstantensymbol $\underline{0}$,
- einem zweistelligen Prädikatensymbol $<$,
- den Booleschen Operatoren $\wedge$, $\vee$, $\neg$,
- den Quantoren erster Stufe $\exists x$, $\forall x$, wobei x für Elemente des Interpretationsbereichs steht,
- den Quantoren zweiter Stufe $\exists X$, $\forall X$, wobei X für Teilmengen des Interpretationsbereichs steht.

Als Interpretationsbereich betrachten wir die natürlichen Zahlen ω, wobei

- $\underline{0}$ als 0,
- s als Nachfolgerfunktion $m \mapsto m+1$,
- $<$ als die übliche Ordnung auf ω

interpretiert werden.
Wie im endlichen Fall sei $\Sigma = \{0,\ 1\}^n$. Eine gegebene Interpretation $P_1^I,\ \dots, P_n^I$ der Symbole $P_1,\ \dots,\ P_n$ entspricht einem ω-Wort $\alpha(0)\alpha(1)\alpha(2)\ldots \in \Sigma^\omega$, wobei $\alpha(m) = (b_{m1},\ b_{m2},\ \dots,\ b_{mn})$ mit $b_{mi} = \begin{cases} 1 & falls\ m \in P_i^I \\ 0 & falls\ m \notin P_i^I \end{cases}$. Für eine geschlossene S1S-Formel φ schreiben wir wieder $\alpha \models \varphi$, wenn die Interpretation, welche dem α entspricht, die Formel φ wahr macht. Die von φ akzeptierte Sprache ist $L_\omega(\varphi) = \{\alpha \in \Sigma^\omega \mid \alpha \models \varphi\}$.

□

Beispiel 5.2
Es sei $n = 1$, das heißt $\Sigma = \{0,\ 1\}$.

1.
$$\begin{aligned} \varphi &= P_1(\underline{0}) \wedge \\ & \quad \forall x\ P_1(x) \to \neg P_1(s(x)) \wedge \\ & \quad \forall x\ \neg P_1(x) \to P_1(s(x)) \\ L_\omega(\varphi) &= \{101010\ldots\} = (10)^\omega \end{aligned}$$

2.
$L_1 = \{\alpha \in \Sigma^\omega \mid$ nach jeder 1 in α kommt noch eine $0\}$
Für $\varphi = \forall x\ (P_1(x) \to \exists y\ (x < y\ \wedge\ \neg P_1(y)))$ gilt $L_\omega(\varphi) = L_1$.

3.
$$L_2 = \{\alpha \in \Sigma^\omega \mid \begin{array}{l} \text{zwischen zwei Vorkommen von 1 in } \alpha \text{ befindet sich} \\ \text{eine gerade Anzahl von Vorkommen von } 0\} \end{array}$$

$$\begin{aligned} \varphi = \quad \forall x\ \forall y \quad & (x < y \wedge \\ & P_1(x) \wedge \\ & P_1(y) \wedge \\ & \forall z\ (x < z \wedge z < y \to \neg P_1(z)) \\ & \to \\ & \exists X\ \exists Y \\ & (\forall z\ \neg(X(z) \wedge Y(z)) \wedge \\ & X(s(x)) \wedge \\ & \forall z\ (X(z) \to Y(s(z))) \wedge \\ & \forall z\ (Y(z) \to X(s(z))) \wedge \\ & \forall z\ (s(z) = y \to Y(z))) \end{aligned}$$

□

Wie im endlichen Fall verwenden wir die Abkürzung $Q_a(x)$ für die Formel, die ausdrückt, daß an Position x das Symbol $a \in \Sigma$ steht.

Lemma 5.3
Sowohl $\underline{0}$ als auch $<$ kann in S1S mit Hilfe der übrigen Symbole ausgedrückt werden.

Beweis

1. $x = \underline{0}$ ist äquivalent zu $\neg\exists y\ (y < x)$

2. $x < y$ entspricht $\exists X\ (\neg X(x) \wedge X(y) \wedge \forall z\ (X(z) \to X(s(z))))$

□

Satz 5.4
Für eine ω-Sprache L sind äquivalent:

1. L ist ω-regulär
2. $L = L_\omega(\varphi)$ für eine geschlossene S1S-Formel φ

□

Beweis
"$1 \Rightarrow 2$"
Es sei $\mathcal{A} = (Q,\ \Sigma,\ I,\ \Delta,\ F)$ ein Büchi-Automat mit $L = L_\omega(\mathcal{A})$. Wir werden die Existenz eines erfolgreichen Pfads in $\mathcal{A}$ mit Hilfe einer S1S-Formel bechreiben.
Es sei $Q = \{q_0,\ q_1,\ \dots,\ q_m\}$ und ohne Einschränkung sei $I = \{q_0,\ q_1,\ \dots,\ q_k\}$ für ein $k \le m$. Wir verwenden Variablen $Y_0,\ Y_1,\ \dots,\ Y_m$ zweiter Stufe, wobei $Y_i(x)$ als "der x-te Zustand im Pfad ist q_i" gelesen werden soll.

$\exists Y_0 \dots \exists Y_m$	
$(\forall x \bigwedge_{0 \le i < j \le m} \neg(Y_i(x) \wedge Y_j(x))$	die Mengen sind disjunkt
$\wedge (Y_0(\underline{0}) \vee \dots \vee Y_k(\underline{0}))$	der Pfad beginnt mit einem Anfangszustand
$\wedge \forall x \bigvee_{(q_i,a,q_j) \in \Delta} Y_i(x) \wedge Q_a(x) \wedge Y_j(s(x))$	an Pos. $s(x)$ steht ein bzgl. Δ erlaubter Nachfolgezust.
$\wedge \bigvee_{q_i \in F} \forall x\ \exists y\ (x < y \wedge Y_i(y)))$	einer der Endzustände wird unendlich oft durchlaufen

Nach Konstruktion erfüllt $\alpha \in \Sigma^\omega$ diese Formel genau dann, wenn α Beschriftung eines erfolgreichen Pfads ist.
"$2 \Rightarrow 1$"
Wir bringen für diese Richtung zunächst S1S-Formeln in eine Normalform:

1. Es dürfen nur noch Variablen X_i zweiter Ordnung auftreten und keine Variablen x erster Ordnung.
2. Atomare Formeln haben die Gestalt
   - $X_i \subseteq X_j$ (mit Semantik $\forall x\ X_i(x) \to X_j(x)$)
   - $Succ(X_i) = X_j$ (mit Semantik $X_i = \{m\}, X_j = \{n\}$ und $n = m+1$)

   Dabei sind $X_i,\ X_j$ Variablen zweiter Stufe oder Prädikate $P_1,\ \dots,\ P_n$.

Formeln, welche aus diesen atomaren Formeln mit Booleschen Operatoren und Quantoren zweiter Stufe aufgebaut sind, heißen S1S$_0$-Formeln.
Behauptung:
Jede S1S-Formel kann in eine äquivalente S1S$_0$-Formel überführt werden.
Beweis (der Behauptung):

i) Wir haben in Lemma 5.3 gesehen, daß $\underline{0}$ und $<$ eliminiert werden können.
ii) Geschachtelte Anwendungen der Nachfolgerfunktion können wie folgt eliminiert werden:

$$x = \underbrace{s(s(s(\dots s}_{m-mal}(y)\dots)))$$

ist äquivalent zu

$$\exists y_1 \dots \exists y_{m-1}\ (y_1 = s(x) \wedge y_2 = s(y_1) \wedge \dots \wedge y = s(y_{m-1}))$$

iii) Wir haben nun ohne Einschränkung nur noch atomare Formeln der Gestalt

$$x = y,\ s(x) = y,\ P_i(x),\ X(x).$$

Wir verwenden folgende Abkürzungen:

- "$X = Y$" für "$X \subseteq Y \ \wedge\ Y \subseteq X$"
- "$X \neq Y$" für "$\neg(X = Y)$"
- "$Einer(X)$" für "$\exists Y \quad (Y \subseteq X) \wedge (Y \neq X) \wedge$ $\forall Z\ (Z \subseteq X \to Z = X \vee Z = Y)$"

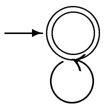
iv) Variablen erster Stufe werden wie im folgenden Beispiel eliminiert:

$$\begin{array}{llll} \forall x & \exists y & (s(x) = y \quad \wedge & Z(y)) \\ \forall X\ (Einer(X) \to & \exists Y\ (Einer(Y) \wedge & Succ(X) = Y \quad \wedge & Y \subseteq Z)) \end{array}$$

Wir können nun davon ausgehen, daß die S1S-Formel φ in S1S$_0$-Form gegeben ist.
Wir zeigen durch Induktion über den Aufbau von S1S$_0$-Formeln φ, daß $L_\omega(\varphi)$ ω-regulär ist. Wir betrachten hier auch offene Formeln mit freien Variablen zweiter Stufe, wobei diese Variablen bei der Definition der akzeptierten Sprache wie einstellige Prädikatensymbole behandelt werden.
Beispiel: Die Formel $\exists Y\ (X \subseteq Y\ \wedge\ P_1 \subseteq Y)$ liefert eine ω-Sprache über $\Sigma = \{0,1\}^2$, da P_1 und X "frei" vorkommen.
Induktionsanker (Konstruktion von Büchi-Automaten für atomare Formeln)
Atomare Formeln sind von der Gestalt $X \subseteq Y$ oder $Succ(X) = Y$. Diese definieren Sprachen über $\Sigma = \{0,1\}^2$, da zwei "freie Variablen" vorkommen. Ohne Einschränkung sei in Zeichen des Alphabets die erste Komponente X und die zweite Komponente Y.

1.
$$L_\omega(X \subseteq Y) = \{\alpha = \alpha_0\alpha_1\alpha_2 \ldots \mid \text{ für } (b_{i,1},\ b_{i,2}) = \alpha_i \in \Sigma \text{ gilt } \underbrace{b_{i,1} = 1}_{i \in X} \Rightarrow \underbrace{b_{i,2} = 1}_{i \in Y}\}$$

Die Sprache $L_\omega(X \subseteq Y)$ wird also akzeptiert von dem Büchi-Automaten

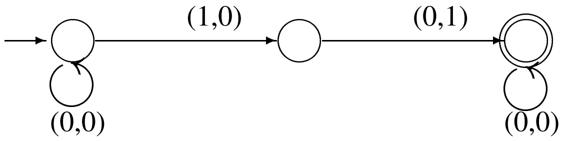


(0,0), (0,1), (1,1)

2.

$$L_\omega(Succ(X) = Y) = \{\alpha = \alpha_0\alpha_1\alpha_2 \ldots \mid \begin{array}{l} \text{für } \alpha_i = (b_{i,1},\ b_{i,2}) \text{ gilt:} \\ \text{es gibt ein } k \text{ mit} \\ \bullet\ \underbrace{b_{k,1} = 1}_{k \in X} \wedge \underbrace{b_{k+1,2} = 1}_{k+1 \in Y} \\ \bullet\ b_{j,1} = 0 \text{ für } j \neq k \\ \bullet\ b_{j,2} = 0 \text{ für } j \neq k+1\} \end{array}$$

Die Sprache $L_\omega(Succ(X) = Y)$ wird akzeptiert von dem Büchi-Automaten



Induktionsschritt (Die Annahme sei gezeigt für φ)
Wir beschränken uns auf Disjunktion, Negation und Existenzquantoren, da sich Konjunktion und Allquantoren damit darstellen lassen.

1. $L_\omega(\neg\varphi) = \Sigma^\omega \backslash L_\omega(\varphi)$
   Die Induktionsvoraussetzung liefert, daß $L_\omega(\varphi)$ ω-regulär ist. Mit Satz 4.15[1] ist daher auch $\overline{L_\omega(\varphi)} = \Sigma^\omega \backslash L_\omega(\varphi)$ ω-regulär.

2. $\varphi = \varphi_1 \vee \varphi_2$
   Die Disjunktion entspricht im Prinzip der Vereinigung. Allerdings können bei $\varphi = \varphi_1 \vee \varphi_2$ die in φ_1 und φ_2 vorkommenden freien Variablen und Prädikatensymbole verschieden sein.
   Beispiel $\varphi(X_1,X_2,X_3) = \varphi_1(X_1,X_2) \vee \varphi_2(X_2,X_3)$
   Wir erweitern φ_1 und φ_2 um die fehlenden Variablen, indem wir trivial-wahre Formeln anfügen, die diese enthalten, zum Beispiel
   $\hat{\varphi}_1(X_1,X_2,X_3) = \varphi_1(X_1,X_2) \wedge X_3 = X_3$
   $\hat{\varphi}_2(X_1,X_2,X_3) = \varphi_2(X_2,X_3) \wedge X_1 = X_1$
   Ist $\mathcal{A}_1$ ein Automat für $\varphi_1(X_1,X_2)$, so erhält man den Automaten $\hat{\mathcal{A}}_1$ für $\hat{\varphi}_1(X_1,X_2,X_3)$ wie folgt: Anstelle von $\Sigma = \{0,1\}^2$ verwendet man $\hat{\Sigma} = \{0,1\}^3$. In $\hat{\mathcal{A}}_1$ gibt es den Übergang
   $$q \stackrel{(b_1,b_2,b_3)}{\longrightarrow} q'$$
   genau dann, wenn
   $$q \stackrel{(b_1,b_2)}{\longrightarrow} q'$$
   ein Übergang in $\mathcal{A}_1$ ist.
   Entsprechend erhält man aus dem Automaten $\mathcal{A}_2$ für φ_2 einen Automaten $\hat{\mathcal{A}}_2$ für $\hat{\varphi}_2$.
   $$\begin{aligned} L_\omega(\varphi(X_1,X_2,X_3)) &= L_\omega(\hat{\varphi}_1(X_1,X_2,X_3)) \cup L_\omega(\hat{\varphi}_2(X_1,X_2,X_3)) \\ &= \underbrace{L_\omega(\hat{\mathcal{A}}_1) \cup L_\omega(\hat{\mathcal{A}}_2)}_{\substack{\omega-\text{regulär, da } \hat{\mathcal{A}}_1 \text{ und } \hat{\mathcal{A}}_2 \text{ Büchi}-\text{Automaten sind und} \\ \omega-\text{reguläre Sprachen bezüglich Vereinigung abgeschlossen sind}}} \end{aligned}$$

3. $\varphi(X_1,\ldots,X_n) = \exists Y \psi(Y,X_1,\ldots,X_n)$
   Ist $\mathcal{A}$ ein Automat für $\psi(Y,X_1,\ldots,X_n)$, so erhält man einen Automaten für $\varphi(X_1,\ldots,X_n)$ wie folgt: Ersetze jeden Übergang
   $$q \stackrel{(b_0,b_1,\ldots,b_n)}{\longrightarrow} q'$$
   in $\mathcal{A}$ durch einem Übergang
   $$q \stackrel{(b_1,\ldots,b_n)}{\longrightarrow} q'$$

□

[1] Satz 4.15 wurde bewiesen, um ihn an dieser Stelle anwenden zu können

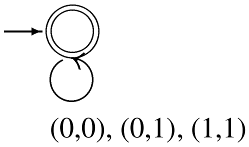
Damit ist der Beweis von Satz 5.4 abgeschlossen.

Beispiel 5.5
Dies ist ein Beispiel für die Anwendung von Teil 2 des Beweises von Satz 5.4.

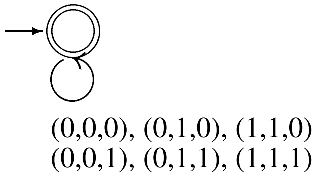
$$\varphi = \exists Y(X \subseteq Y \lor Succ(Y) = Z)$$

ist eine $S1S_0$-Formel mit zwei freien Variablen, X und Y.

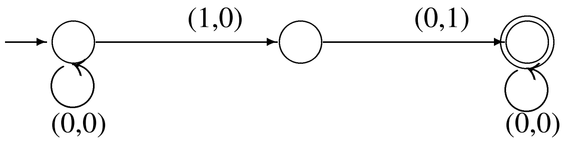
$\mathcal{A}_1$ ist ein Automat für $X \subseteq Y$, mit 1. Komponente X und 2. Komponente Y:



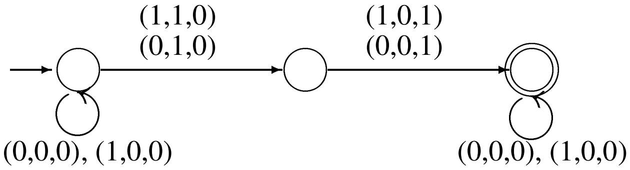
$\hat{\mathcal{A}}_1$ ist ein Automat für $X \subseteq Y$ mit einer zusätzlichen Stelle für Z:



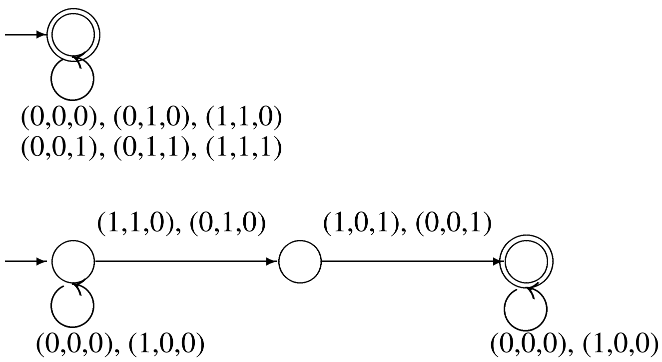
$\mathcal{A}_2$ ist ein Automat für $Succ(Y) = Z$, mit 1. Komponente Y und 2. Komponente Z:



$\hat{\mathcal{A}}_2$ ist ein Automat für $Succ(Y) = Z$ mit einer zusätzlichen Stelle für X als 1. Komponente:

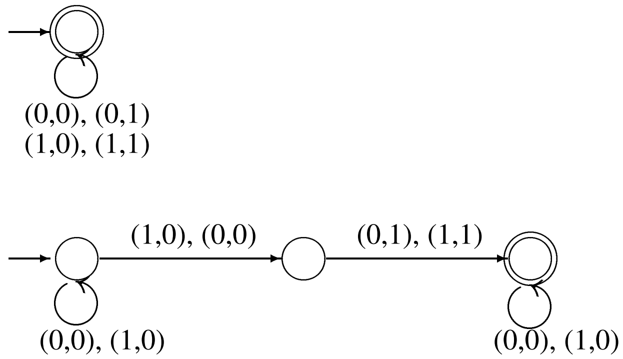


Der Automat $\hat{\mathcal{A}} := \hat{\mathcal{A}}_1 \dot{\cup} \hat{\mathcal{A}}_2$ ist ein Automat für $X \subseteq Y \lor Succ(Y) = Z$. Ein Bild dieses Automaten erhält man, indem man die Graphiken für $\hat{\mathcal{A}}_1$ und $\hat{\mathcal{A}}_2$ nebeneinander zeichnet.
Beachte: Dieser Automat hat zwei Anfangszustände.



Den Automaten $\mathcal{A}$ für die durch φ beschriebene Sprache erhält man aus $\hat{\mathcal{A}}$ durch Streichen

der zweiten Komponente innerhalb der Beschriftungen der Transitionen – dies ist genau die Komponente für Y.
$\mathcal{A}$:



□

Der Beweis zeigt insbesondere, wie man zu einer S1S-Formel φ effektiv einen Büchi-Automaten $\mathcal{A}$ konstruieren kann mit $L_\omega(\varphi) = L_\omega(\mathcal{A})$. Es folgt, daß man von einer (geschlossenen[2]) S1S-Formel φ entscheiden kann, ob sie in der Interpretation $(\omega, <, +1, 0)$ gültig ist oder nicht, denn es gilt ja

$$\varphi \text{ ist gültig } \quad gdw \quad \neg\varphi \text{ hat kein Modell } \quad gdw \quad L_\omega(\neg\varphi) = \emptyset$$

Damit ist das Gültigkeitsproblem auf das Leerheitsproblem reduziert, denn es gilt $L_\omega(\neg\varphi) = L_\omega(\mathcal{A})$ für einen Büchi-Automaten $\mathcal{A}$, der effektiv konstruiert werden kann. Das Leerheitsproblem für Büchi-Automaten ist mit Satz 4.10 entscheidbar.

Korollar 5.6
Gültigkeit in S1S ist entscheidbar.[3] □

Der Beweis von Satz 5.4 läßt sich leicht so modifizieren, daß er auch für reguläre Sprachen endlicher Wörter funktioniert.

Korollar 5.7
Für eine Sprache $L \subseteq \Sigma^*$ sind äquivalent:

1. L ist regulär
2. $L \setminus \{\varepsilon\} = L(\varphi)$ für eine geschlossene S1S-Formel φ. *Beachte:* Die Interpretation $(L(\varphi))$für S1S-Formeln muß für endliche Wörter angepaßt werden.

□

Wie in Abschnitt 1.3 eingeführt, faßt man hier endliche Wörter als endliche, total geordnete Interpretationen auf.
Im Beweis von "$1 \Rightarrow 2$" muß als Anpassung an die nun endlichen Wörter im wesentlichen die Akzeptanzbedingung geändert werden:

$$\bigvee_{q_i \in F} \forall x \, \exists y \, (x < y \wedge Y_i(y))$$

[2] bei offenen Formeln werden freie Variablen als Prädikatensymbole aufgefaßt
[3] Entscheidbarkeitsaussagen waren der Grund, warum Büchi die Büchi-Automaten entwickelte

wird ersetzt durch

$$\bigvee_{q_i \in F} \forall x \; (\underbrace{Max(x)}_{\substack{Abk\ddot{u}rzung \\ f\ddot{u}r s(x)=x}} \to Y_i(x))$$

Im Beweis von "2 $\Rightarrow$ 1" wendet man die Abschlußeigenschaften regulärer Sprachen (statt der für ω-reguläre Sprachen) an.
Beim Induktionsanfang muß man die spezielle Semantik von s beachten und mit der Akzeptanzbedingung für endliche Wörter arbeiten.

Korollar 5.8
Für eine S1S-Formel φ ist es entscheidbar, ob sie in allen endlichen S1S-Interpretationen gilt.

□

Büchi hat auch die sogenannte schwache Logik WS1S[4] betrachtet, bei der die Quantoren $\exists X$ und $\forall X$ zweiter Stufe nur über endliche Teilmengen von ω quantifizieren.
Man kann WS1S in S1S ausdrücken, denn der Quantor "$\exists X \; \varphi(X)$" in WS1S wird in S1S zu "$\exists X \; (\exists y \, \forall X \; (X(x) \to x < y) \wedge \varphi(X))$". Damit ist gezeigt, daß Gültigkeit in WS1S entscheidbar ist. Umgekehrt kann man zeigen, daß WS1S genauso ausdrucksstark ist wie S1S: zu jeder ω-regulären Sprache L gibt es eine WS1S-Formel φ mit $L_\omega(\varphi) = L$.
Wir verwenden die Entscheidbarkeit von WS1S, um die Entscheidbarkeit der Presburger Arithmetik zu zeigen.

Definition 5.9 (Presburger Arithmetik)
Man betrachtet hier Formeln der Logik erster Stufe, die als nicht-logische Symbole eine Konstante "$\underline{0}$" und ein zweistelliges Funktionssymbol "+" enthalten dürfen. Als Interpretation verwendet man die natürlichen Zahlen, wobei "$\underline{0}$" als 0 und "+" als Addition interpretiert wird.

□

Satz 5.10
Die Presburger Arithmetik ist entscheidbar, das heißt für eine Formel der Logik erster Stufe, die als nicht-logische Symbole nur "$\underline{0}$" und "+" enthält, ist es entscheidbar, ob sie in den natürlichen Zahlen gilt.

Beweis
Man stellt die natürlichen Zahlen in Binärcodierung dar und interpretiert die 0-1-Folgen als (endliche) Teilmengen von ω. Ist $n = 2^{i_1} + 2^{i_2} + \ldots + 2^{i_k}$ mit $i_1 < i_2 < \ldots < i_k$, so wird n durch die Menge $\{i_1, i_2, \ldots, i_k\} \subseteq \omega$ codiert.
Beispiel: 13 in Binärdarstellung ist 1101, also wird 13 durch $\{0,2,3\}$ codiert. 20 in Binärdarstellung ist 10100, also wird 20 codiert durch $\{2,4\}$.

[4] W steht für engl. weak

Wir zeigen nun, wie man die Addition natürlicher Zahlen in WS1S ausdrücken kann. Es gibt eine WS1S Formel $\varphi_+(X,Y,Z)$ mit der folgenden Eigenschaft: Ist $K \subseteq \omega$ Codierung von k, N Codierung von n und M Codierung von m, so gilt

$$\varphi_+(K,N,M) \text{ gdw } k+n=m.$$

Beispiel: 13+20=33, bzw. ausführlich

$$\begin{array}{rrcr}
 & 10100 & \widehat{=} & \{2,4\} =: X \\
 & 01101 & \widehat{=} & \{0,2,3\} =: Y \\
\text{Übertrag} & \overline{111000} & \widehat{=} & \{3,4,5\} =: R \\
 & \overline{100001} & \widehat{=} & \{0,5\} =: Z
\end{array}$$

Wir benötigen eine Hilfsvariable R zweiter Stufe für den Übertrag:

- x gehört zu Z ($\widehat{=}$ 1 an Position x in der Summe) gdw eine oder alle drei Mengen X, Y, R das x enthalten
- $x+1$ gehört zu R ($\widehat{=}$ 1 an Position $x+1$ im Übertrag) gdw x gehört zu mindestens zwei der Mengen X, Y, Z

Nun können wir φ_+ angeben:

$$\begin{array}{lll}
\varphi_+(X,Y,Z) := \exists R & (\neg R(\underline{0}) \wedge & \\
 & \forall x\ (R(s(x)) \leftrightarrow & ((X(x) \wedge Y(x)) \vee \\
 & & (X(x) \wedge R(x)) \vee \\
 & & (Y(x) \wedge R(x)))) \\
 & \forall x\ (Z(x) \leftrightarrow & ((X(x) \wedge Y(x) \wedge R(x)) \vee \\
 & & (X(x) \wedge \neg Y(x) \wedge \neg R(x)) \vee \\
 & & (\neg X(x) \wedge Y(x) \wedge \neg R(x)) \vee \\
 & & (\neg X(x) \wedge \neg Y(x) \wedge R(x)))))
\end{array}$$

Man kann damit beliebige Presburger Formeln in WS1S-Formeln übersetzen.
Beispiel: $\forall x\ (x+\underline{0}=x)$ wird übersetzt in $\exists Z\ (\forall x\ (\neg Z(x)) \ \wedge\ \forall X\ \varphi_+(X,Z,X))$.

□

Wir haben WS1S anstelle von S1S verwendet, da natürliche Zahlen endlichen Teilmengen von ω entsprechen.
Bei endlichen Wörtern konnte die Klasse von Sprachen, welche durch Formeln der Logik erster Stufe akzeptiert werden, mit Satz 3.9 folgendermaßen charakterisiert werden:
Für $L \subseteq \Sigma^*$ sind äquivalent:

1. L ist sternfrei
2. $L \setminus \{\varepsilon\} = L(\varphi)$ für eine geschlossene Formel der Logik erster Stufe über den nichtlogischen Symbolen $<$, Q_a ($a \in \Sigma$).

Bei ω-Sprachen gibt es einen entsprechenden Zusammenhang.

Definition 5.11
Eine ω-Sprache $L \subseteq \Sigma^\omega$ heißt sternfrei gdw $L = \bigcup_{i=1}^{n} U_i V_i^\omega$ für sternfreie Sprachen U_i, $V_i \subseteq \Sigma^*$.

□

Satz 5.12
Für eine ω-Sprache $L \subseteq \Sigma^\omega$ sind äquivalent:

1. L ist sternfrei
2. $L = L_\omega(\varphi)$ für eine geschlossene S1S-Formel φ, welche keine Quantifizierung über einstellige Prädikate enthält.

□

Wir erwähnen dieses und die folgenden Resultate ohne Beweis. Die Beweise oder Hinweise auf Literatur, in der die Beweise geführt werden, finden sich in
W. Thomas: Automata on Infinite Objects, J. van Leeuwen (Hrsg.) Handbook of Theoretical Computer Science, Vol. B, Elsevier 1990.
Man kann sternfreie ω-Sprachen mit Hilfe endlicher Monoide charakterisieren.

Definition 5.13
Es sei $L \subseteq \Sigma^\omega$ eine ω-Sprache. Die Kongruenzrelation $\approx_L$ auf Σ^* ist definiert durch

$$\begin{aligned} u \approx_L v \quad \text{gdw} \quad & \forall x,y,z \in \Sigma^* : \\ & (xuyz^\omega \in L \; gdw \; xvyz^\omega \in L) \;\wedge \\ & (x(yuz)^\omega \in L \; gdw \; x(yvz)^\omega \in L) \end{aligned}$$

□

Für ω-reguläre Sprachen L hat $\approx_L$ stets endlichen Index. Im Gegensatz zum Fall endlicher Wörter gilt die Umkehrung aber nicht, das heißt es gibt nicht-reguläre ω-Sprachen L, für die $\approx_L$ auch endlichen Index hat. Für eine Charakterisierung der ω-regulären Sprachen benötigt man daher eine zusätzliche Bedingung.

Definition 5.14
Es sei $L \subseteq \Sigma^\omega$ und $\sim$ eine (beliebige) Kongruenzrelation auf Σ^*. Die Relation $\sim$ saturiert L genau dann, wenn es endlich viele $\sim$-Klassen $U_1, U_2, \ldots, U_m$ und $V_1, V_2, \ldots, V_m$ gibt mit

$$L = \bigcup_{i=1}^{m} U_i V_i^\omega .$$

□

Satz 5.15
Für eine ω-Sprache $L \subseteq \Sigma^\omega$ sind äquivalent

1. L ist ω-regulär
2. $\approx_L$ hat endlichen Index und saturiert L

□

Die sternfreien ω-Sprachen entsprechen nun wieder den aperiodischen Monoiden.

Satz 5.16
Für eine ω-reguläre Sprache L sind äquivalent

1. L ist sternfrei
2. $\Sigma^*/\approx_L$ ist aperiodischer endlicher Monoid

□

Satz 5.12 und Satz 5.16 ermöglichen es wieder, für eine gegebene ω-reguläre Sprache zu überprüfen, ob sie durch eine Formel der Logik erster Stufe akzeptiert wird.

Beispiel 5.17
$\Sigma = \{a,b,c\}$, $L = \{\alpha \in \Sigma^\omega \mid$ zwischen zwei a's befindet sich eine gerade Anzahl b's und c's$\}$
Wir zeigen: $\Sigma^*/\approx_L$ ist nicht aperiodisch, das heißt L kann nicht von einer Formel der Logik erster Stufe akzeptiert werden.
Wir wissen: die M-Varietät der aperiodischen Monoide ist schließlich definiert durch $(x^{n+1} = x^n)_{n\geq 1}$.
$x := [b]_{\approx_L} \in \Sigma^*/\approx_L$ erfüllt $x^{n+1} = x^n$ für kein n, **denn** ohne Einschränkung sei n gerade (der Fall n ungerade läuft analog) und $n+1$ somit ungerade. Aus $x^{n+1} = x^n$ würde folgen $b^n \approx_L b^{n+1}$.
Es ist aber $ab^n aa^\omega \in L$ (da n gerade ist) und $ab^{n+1}aa^\omega \notin L$ (da $n+1$ ungerade ist). Es folgt $b^n \not\approx_L b^{n+1}$. *Widerspruch*

□

Wir betrachten nun eine weitere Logik, für die man Entscheidbarkeit mit Hilfe von Büchi-Automaten zeigen kann. Diese Logik soll das Ein-/Ausgabeverhalten eines deterministischen Programms p beschreiben.

- p führt Konfigurationen in Folgekonfigurationen über. Deterministisch heißt, daß es zu einer Konfiguration k höchstens eine Folgekonfiguration gibt.
- Nicht-Terminierung von p entspricht der Tatsache, daß es keine Folgekonfiguration gibt.
- p kann mehrmals hintereinander ausgeführt werden und es kann iteriert werden.

Wir gehen davon aus, daß endlich viele atomare Eigenschaften $A_1, \ldots, A_n$ von Konfigurationen gegeben sind.
Man will eine Logik, in der man Aussagen wie die folgenden beschreiben kann:

a1) Gilt A_1 in k, so gilt A_2 in der Folgekonfiguration.

a2) Gilt A_1 in k, so existiert keine Folgekonfiguration, das heißt p terminiert nicht.

a3) Beginnend mit der Konfiguration k, in der A_1 gilt, kann man durch endlich oftes Anwenden von p eine Konfiguration erreichen, in der A_2 gilt.

Definition 5.18
Formeln der Logik 1DPDL (Propositionale Dynamische Logik eines deterministischen Programms) sind wie folgt aufgebaut:

1. Programmformeln

   - p und ε sind Programmformeln.
   - Sind π_1 und π_2 Programmformeln, so auch $\pi_1;\pi_2$ (Hintereinanderausführung), $\pi_1 \cup \pi_2$ (nicht-deterministische Alternative) und π_1^* (Iteration).

2. Konfigurationsformeln

   - $A_1, \ldots, A_n$ sind Konfigurationsformeln.
   - Sind φ_1, φ_2 Konfigurationsformeln und ist π Programmformel, so sind $\varphi_1 \wedge \varphi_2$, $\varphi_1 \vee \varphi_2$, $\neg\varphi_1$, $< \pi > \varphi_1$ und $[\pi]\varphi_1$ Konfigurationsformeln.

Programmformeln beschreiben Programme, die man aus den atomaren Programmen p aufbauen kann. Zum Beispiel ist $p \cup (p;p)$ ein nicht-deterministisches Programm, bei dem man p einmal oder zweimal ausführt.

$< \pi > \varphi_1$:	es gibt eine π-Folgekonfiguration, in der φ_1 gilt.
$[\pi]\varphi_1$:	in allen π-Folgekonfigurationen gilt φ_1 (beachte: es kann mehrere Folgekonfigurationen geben).

□

Beispiel 5.19
Die Aussagen a1, a2 und a3 können wie folgt beschrieben werden:

a1) $A_1 \;\rightarrow\; < p > A_2$

a2) $A_1 \;\rightarrow\; [p](A_1 \wedge \neg A_1)$

a3) $A_1 \;\rightarrow\; < p^* > A_2$

□

Die Semantik von 1DPDL wird durch die folgende Definition festgelegt.

Definition 5.20
Eine Interpretation I besteht aus einer nicht-leeren Menge Δ^I von Konfigurationen und einer Interpretationsfunktion $\cdot^I$, die

1. dem atomaren Programm p eine funktionale Relation $p^I \subseteq \Delta^I \times \Delta^I$ zuordnet. Funktional heißt: $\forall k, k_1, k_2 :\ (k,k_1) \in p^I \wedge (k,k_2) \in p^I \;\rightarrow\; k_1 = k_2$

2. jeder atomaren Eigenschaft A_i eine Menge $A_i^I \subseteq \Delta^I$ zuordnet, das heißt die Menge der Konfigurationen, in denen A_i gilt

Die Interpretationsfunktion kann auf Programm- und Konfigurationsformeln erweitert werden. Dies geschieht durch Induktion über den Formelaufbau. Die Interpretation der atomaren Formeln $p, A_1, \ldots, A_n$ ist bereits definiert.

Programmformeln:

- $\varepsilon^I := \{(k,k) \mid k \in \Delta^I\}$
- $(\pi_1;\pi_2)^I := \pi_1^I \circ \pi_2^I = \{(k_1,k_2) \mid \exists k:\ (k_1,k) \in \pi_1^I \wedge (k,k_2) \in \pi_2^I\}$
- $(\pi_1 \cup \pi_2)^I := \pi_1^I \cup \pi_2^I$
- $(\pi_1^*)^I := \bigcup_{i\geq 0}(\pi_1^I)^i$, wobei $(\pi_1^I)^0 = \varepsilon^I$ und $(\pi_1^I)^{i+1} = (\pi_1^I)^i \circ \pi_1^I$

Jeder Programmformel π wird also eine Relation $\pi^I \subseteq \Delta^I \times \Delta^I$ zugeordnet. Konfigurationsformeln:

- $(\varphi_1 \wedge \varphi_2)^I := \varphi_1^I \cap \varphi_2^I$
- $(\varphi_1 \vee \varphi_2)^I := \varphi_1^I \cup \varphi_2^I$
- $(\neg\varphi_1)^I := \Delta^I \backslash \varphi_1^I$
- $(< \pi > \varphi_1)^I := \{k \in \Delta^I \mid \exists k' \in \Delta^I:\ (k,k') \in \pi^I \wedge k' \in \varphi_1^I\}$
- $([\pi]\varphi_1)^I := \{k \in \Delta^I \mid \forall k' \in \Delta^I:\ (k,k') \in \pi^I \rightarrow k' \in \varphi_1^I\}$

$\square$

Beispiel 5.21

$$\begin{array}{rcl} I & : & k_1 \in A_1^I \xrightarrow{p^I} k_2 \in (\neg A_1)^I \xrightarrow{p^I} k_3 \in A_1^I \xrightarrow{p^I} k_4 \in (\neg A_1)^I \xrightarrow{p^I} \ldots \\ \varphi & : & A_1 \wedge [p^*]((A_1 \rightarrow < p > \neg A_1) \wedge (\neg A_1 \rightarrow < p > A_1)) \\ \varphi^I & : & \{k_1,\ k_3,\ \ldots\} \end{array}$$

$\square$

Definition 5.22

1. Die Konfigurationsformel φ heißt erfüllbar, falls es eine Interpretation I gibt und ein $k \in \Delta^I$ mit $k \in \varphi^I$. Das I heißt dann Modell von φ.
2. Die Konfigurationsformel φ heißt gültig, falls für alle Interpretationen I gilt, daß $\varphi^I = \underbrace{\Delta^I}_{\substack{\text{Menge aller} \\ \text{Konfigurationsformeln}}}$.
3. φ und ψ sind äquivalent genau dann, wenn $\varphi \leftrightarrow \psi$ gültig ist, das heißt für alle Interpretationen I gilt $\varphi^I = \psi^I$.

$\square$

Offenbar ist φ gültig genau dann, wenn $\neg\varphi$ unerfüllbar ist.
Um Erfüllbarkeit zu entscheiden, ordnen wir jeder Konfigurationsformel φ einen Büchi-Automaten $\mathcal{A}_\varphi$ zu, für den gilt: $L_\omega(\mathcal{A}_\varphi) \neq \emptyset$ gdw φ ist erfüllbar.

Bei S1S wurde das Alphabet bestimmt durch die vorhandenen einstelligen Prädikatsymbole ($\hat{=}$ atomaren Eigenschaften). Für 1DPDL ist es günstiger, alle "Unterformeln" der gegebenen Formel zu betrachten.
Zunächst stellen wir Programmformeln als reguläre Sprachen endlicher Wörter über dem Alphabet $\{p\}$ dar:

$$\begin{array}{rcl} L(p) & = & \{p\} \\ L(\varepsilon) & = & \{\varepsilon\} \\ L(\pi_1;\pi_2) & = & L(\pi_1)\cdot L(\pi_2) \\ L(\pi_1\cup\pi_2) & = & L(\pi_1)\cup L(\pi_2) \\ L(\pi_1^*) & = & L(\pi_1)^* \end{array}$$

Es gilt für jede Interpretation I:

$$\begin{array}{lll} (k,k')\in\pi^I & \text{gdw} & \exists m\geq 0\ \exists k_0,\ldots,k_m\in\Delta^I: \\ & & p^m\in L(\pi)\wedge \\ & & k_0=k\wedge \\ & & k_m=k'\wedge \\ & & (k_i,k_{i+1})\in p^I \text{ für } 0\leq i<m \end{array}$$

Es sei $B=(Q,\{p\},\ .\ ,\Delta,F)$ ein endlicher Automat, bei dem die Anfangszustandsmenge noch nicht festgelegt ist. Für jedes $q\in Q$ bezeichne B_q den Automaten, der uas B entsteht, wenn man q zum Anfangszustand macht.

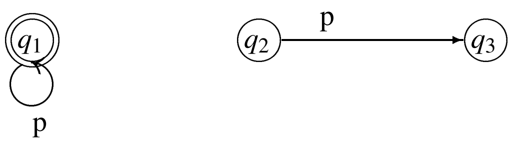
Lemma 5.23
Es sei φ_0 eine Konfigurationsformel, die die Programmformeln $\pi_1,\ldots,\pi_r$ enthält. Dann gibt es einen endlichen Automaten $B=(Q,\{p\},\ .\ ,\Delta,F)$, dessen Göße linear zu der Größe von φ_0 ist, so daß gilt: Für alle $i,\ 1\leq i\leq r$, gibt es $q_i\in Q$ mit $L(\pi_i)=L(B_{q_i})$.

□

Die Programmformel π_i wird damit eindeutig durch den Zustand q_i beschrieben. Wir werden daher im folgenden davon ausgehen, daß in φ_0 anstelle der π_i die Zustände q_i von B stehen.

Beispiel 5.24
$\varphi_0=[p^*]<p>A_1$, das heißt $\pi_1=p^*$ und $\pi_2=p$.

B:



$L(p^*) = L(B_{q_1}), L(p) = L(B_{q_2})$
φ_0 kann daher als $\hat{\varphi}_0 = [q_1] < q_2 > A_1$ geschrieben werden.

□

Es sei φ_0 eine Konfigurationsformel, B der zugehörige Automat und $\hat{\varphi}_0$ sei die Formel, die aus φ_0 entsteht, indem man jede Programmformel durch den entsprechenden Zustand ersetzt.
Wir gehen ohne Einschränkung davon aus, daß φ_0 und $\hat{\varphi}_0$ in Negationsnormalform (NNF) gegeben sind, das heißt Negation kommt nur unmittelbar vor atomaren Eigenschaften vor. Man schiebt dazu mit Hilfe von äquivalenzerhaltenden Regeln die Negationen nach innen:

$$\begin{array}{rcl}
\neg\neg\varphi & \to & \varphi \\
\neg(\varphi \vee \psi) & \to & \neg\varphi \wedge \neg\psi \\
\neg(\varphi \wedge \psi) & \to & \neg\varphi \vee \neg\psi \\
\neg < \pi > \varphi & \to & [\pi]\neg\varphi \\
\neg[\pi]\varphi & \to & < \pi > \neg\varphi
\end{array}$$

Definition 5.25
Der Fischer-Ladner-Abschluß von φ_0 ist die kleinste Menge $FL(\varphi_0)$ von Konfigurationsformeln, für die gilt:

- $\hat{\varphi}_0 \in FL(\varphi_0)$
- $\varphi_1 \wedge \varphi_2 \in FL(\varphi_0) \;\to\; \varphi_1, \varphi_2 \in FL(\varphi_0)$
- $\varphi_1 \vee \varphi_2 \in FL(\varphi_0) \;\to\; \varphi_1, \varphi_2 \in FL(\varphi_0)$
- $\neg A_i \in FL(\varphi_0) \;\leftrightarrow\; A_i \in FL(\varphi_0)$
- $[q]\varphi \in FL(\varphi_0) \;\to\; \varphi \in FL(\varphi_0)$ und $[q']\varphi \in FL(\varphi_0)$ für alle $q' \in Q$ mit $(q, p, q') \in \Delta$
- $< q > \varphi \in FL(\varphi_0) \;\to\; \varphi \in FL(\varphi_0)$ und $< q' > \varphi \in FL(\varphi_0)$ für alle $q' \in Q$ mit $(q, p, q') \in \Delta$

□

Beachte: $|FL(\varphi_0)|$ ist linear zu der Größe von φ_0.

Beispiel 5.24 (Fortsetzung)
$\hat{\varphi}_0 = [q_1] < q_2 > A_1$ ist in NNF.
$FL(\varphi_0) = \{\hat{\varphi}_0,\ < q_2 > A_1,\ \neg A_1,\ A_1,\ < q_3 > A_1\}$

□

Es sei nun φ_0 Konfigurationsformel in NNF und I ein Modell von φ_0 mit $k_0 \in \varphi_0^I$. Beginnend mit k_0 wenden wir das Programm p iteriert an. Es können zwei Fälle eintreten:

1. p terminiert stets, das heißt es gibt eine unendliche Folge $k_0, k_1, k_2, \ldots \in \Delta^I$ mit $(k_i, k_{i+1}) \in p^I$ für alle $i \geq 0$
2. p terminiert irgendwann nicht, das heißt es gibt eine endliche Folge $k_0, \ldots, k_m \in \Delta^I$ mit $(k_i, k_{i+1}) \in p^I$ für $0 \leq i < m$ und $(k_m, k) \notin p^I$ für alle $k \in \Delta^I$

Für $k \in \Delta^I$ sei $S(k) := \{\varphi \in FL(\varphi_0) \mid k \in \varphi^I\}$.
Beachte: $S(k) \neq \emptyset$, da für atomare Eigenschaften A_i in φ_0 gilt: $A_i \in S(k)$ oder $\neg A_i \in S(k)$.
Im ersten Fall erhalten wir ein unendliches Wort $\alpha(k_0) = S(k_0)S(k_1)S(k_2)\ldots$ über dem Alphabet $\Sigma := 2^{FL(\varphi_0)}$.
Im zweiten Fall verlängern wir das endliche Wort $S(k_0)\ldots S(k_m)$ zu einem unendlichen Wort $\alpha(k_0) \in \Sigma^\omega$ durch Anfügen von $\emptyset\emptyset\emptyset\ldots$.
Man erhält so ein Wort aus Σ^ω, das Eigenschaften hat, die sehr stark von der Struktur von φ_0 abhängen.

Definition 5.26
Das Wort $\alpha = \alpha_0\alpha_1\alpha_2\ldots \in \Sigma^\omega$ ist ein Hintikka-Wort für φ_0, falls gilt:

1. $\hat{\varphi}_0 \in \alpha_0$

und für alle $m \geq 0$

2. $\alpha_m = \emptyset$ oder $A_i \in \alpha_m$ gdw $\neg A_i \notin \alpha_m$ für alle atomaren Eigenschaften $A_i \in FL(\varphi_0)$
3. $\varphi \wedge \psi \in \alpha_m$ gdw $\varphi \in \alpha_m$ und $\psi \in \alpha_m$
4. $\varphi \vee \psi \in \alpha_m$ gdw $\varphi \in \alpha_m$ oder $\psi \in \alpha_m$
5. $[q]\varphi \in \alpha_m$, dann gelten
   5.1 falls $\varepsilon \in L(B_q)$, so $\varphi \in \alpha_m$
   5.2 $\alpha_{m+1} = \emptyset$ oder für alle q' mit $(q, p, q') \in \Delta$ gilt $[q']\varphi \in \alpha_{m+1}$
6. $< q > \varphi \in \alpha_m$, dann gibt es ein $r \geq 0$ mit $p^r \in L(B_q)$ und $\varphi \in \alpha_{m+r}$
7. Ist $\alpha_m = \emptyset$, so auch $\alpha_{m+1} = \emptyset$

□

Satz 5.27
φ_0 ist erfüllbar genau dann, wenn es ein Hitikka-Wort für φ_0 gibt.

Beweisskizze

"⇒" Es sei I ein Modell für φ_0 und k_0 so, daß $k_0 \in \varphi_0^I$. Dann ist $\alpha(k_0)$ ein Hintikka-Wort für φ_0.
Beweis: siehe Übungsaufgaben.

"⇐" Es sei $\alpha = \alpha_0\alpha_1\alpha_2\ldots$ ein Hitikka-Wort fir φ_0. Wir definieren ein Modell I wie folgt:

$$\begin{array}{rcl} \Delta^I & = & \{k_0, k_1, k_2, \ldots\} \\ p^I & = & \{(k_i, k_{i+1}) \mid \alpha_{i+1} \neq \emptyset\} \\ A_j^I & = & \{k_i \mid A_j \in \alpha_i\} \end{array}$$

Man zeigt durch Induktion über den Formelaufbau:

$$\text{Für alle } i \geq 0 \text{ und alle } \varphi \in FL(\varphi_0) \text{ gilt: } \varphi \in \alpha_i \rightarrow k_i \in \varphi^I$$

Wegen $\hat{\varphi}_0 \in \alpha_0$ folgt daraus $k_0 \in \hat{\varphi}_0^I$, das heißt $k_0 \in \varphi_0^I$.

□

Gesucht ist nun ein Büchi-Automat, der genau die Hintikka-Wörter für φ_0 akzeptiert.
Der <u>lokale Automat</u> $\mathcal{A}_L(\varphi_0)$ akzeptiert eine etwas größere Menge von Wörtern. Hier ist die globale Bedingung 6. in Definition 5.26 ersetzt durch die lokale Bedingung

6'. $< q > \varphi \in \alpha_m$, dann gilt:

6'.1 $\varepsilon \in L(B_q)$ und $\varphi \in \alpha_m$
oder
6'.2 es gibt ein $q' \in Q$ mit $(q, p, q') \in \Delta$ und $< q' > \varphi \in \alpha_{m+1}$

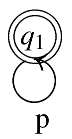
Definition 5.28
$\mathcal{A}_L(\varphi_0) := (Q_L,\ \Sigma,\ I_L,\ \Delta_L,\ F_L)$ mit

- $Q_L = \{\alpha_i \in 2^{FL(\varphi_0)} \mid \alpha_i \text{ erfüllt 2., 3. und 4. von Definition 5.26}\}$
- $I_L = \{\alpha_i \in Q_L \mid \hat{\varphi}_0 \in \alpha_i\}$
- $\Delta_L = \{(\alpha_m, \sigma, \alpha_{m+1}) \mid \alpha_m = \sigma \text{ und } \alpha_i,\ \alpha_{i+1} \text{ erfüllen 5., 6'. und 7.}\}$
- $F_L = Q_L$

□

Offenbar akzeptiert $\mathcal{A}_L(\varphi_0)$ genau die Wörter $\alpha_0\alpha_1\alpha_2\ldots \in \Sigma^\omega$, welche 1. bis 5. und 7. von Definition 5.26 sowie 6'. erfüllen.

Beispiel 5.29
Aus $\alpha \in L_\omega(\mathcal{A}_L(\varphi_0))$ folgt noch nicht, daß α ein Hintikka-Wort für φ_0 ist:
$\varphi_0 = < p^* > (A_1 \wedge \neg A_1)$
$B:$



$\hat{\varphi}_0 =< q_1 > (A_1 \wedge \neg A_1)$
Es gilt für $\alpha_0 = \{\hat{\varphi}_0, A_1\}$, daß $\alpha_0^\omega \in L_\omega(\mathcal{A}_L(\varphi_0))$.
Das Wort α_0^ω erfüllt zwar 6'., nicht aber 6. Das Problem ist, daß bei 6'. stets die Alternative 6'.2 gewählt wird. Damit 6. erfült ist, muß nach endlicher Zeit einmal 6'.1 gewählt werden.

□

Definition 5.30
$\mathcal{A}_H(\varphi_0) := (Q_H,\ \Sigma,\ I_H,\ \Delta_H,\ F_H)$ mit

- $Q_H = Q_L \times 2^{FL_{<>}(\varphi_0)}$, wobei $FL_{<>}(\varphi_0) := \{< q > \varphi \mid\ < q > \varphi \in FL(\varphi_0)\}$
- $\Sigma = 2^{FL(\varphi_0)}$
- $I_H = I_L \times \{\emptyset\}$
- $F_H = Q_L \times \{\emptyset\}$
- $((\alpha_m, s), \sigma, (\alpha_{m+1}, t)) \in \Delta_H$ gdw

entweder
  * $s = \emptyset$
  * $(\alpha_m, \sigma, \alpha_{m+1}) \in \Delta_L$
  * t enthält für jedes $< q > \varphi \in \sigma$ mit ($\varepsilon \notin L(B_q)$ oder $\varphi \notin \sigma$ ein ¡q'¿$\varphi \in \alpha_{m+1}$ mit $(q, p, q') \in \Delta$

oder
  * $s \neq \emptyset$
  * $(\alpha_m, \sigma, \alpha_{m+1}) \in \Delta_L$
  * t enthält für jedes $< q > \varphi \in s$ mit ($\varepsilon \notin L(B_q)$ oder $\varphi \notin \sigma$ ein ¡q'¿$\varphi \in \alpha_{m+1}$ mit $(q, p, q') \in \Delta$

Beachte: Nach Definition von Δ_L enthält α_{m+1} stets ein geeignetes $< q' > \varphi$, da 6' durch Δ_L erfüllt ist.

□

Satz 5.31
$\mathcal{A}_H(\varphi_0)$ akzeptiert genau die Hintikka-Wörter für φ_0.

□

Satz 5.32
Erfüllbarkeit in 1DPDL ist in exponentieller Zeit entscheidbar.

Beweis
Der Automat $\mathcal{A}_H(\varphi_0)$ ist exponentiell in der Größe von φ_0 (Teilmengen von $FL(\varphi_0)$). Das Leerheitsproblem für Büchi-Automaten ist in linearer Zeit entscheidbar.

□

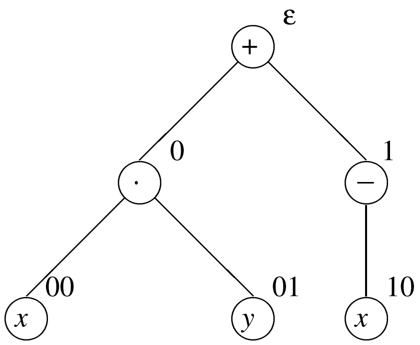
Beachte: Anders als bei S1S wurde hier der sehr aufwendige (das heißt schlimmer als exponentielle) Abschluß unter Komplement *nicht* verwendet.
Häufig möchte man nicht nur *ein* atomares Programm zur Verfügung haben, sondern endlich viele atomare Programme $p_1,\ p_2,\ \ldots,\ p_n$. Dann reicht die Betrachtung von Wörtern nicht mehr aus. Modelle liefern dann <u>Hintikka-Bäume</u>.

Kapitel 6

Automaten auf endlichen Bäumen

Wir betrachten Bäume mit beschrifteten Knoten, bei denen die Anzahl der Nachfolgeknoten durch die Stelligkeit der Knotenbeschriftung bestimmt ist.

Beispiel 6.1
$\Sigma = \{+, \cdot, -, x, y\}$ ist die Menge der Knotenbeschriftungen, wobei $+$ und $\cdot$ zweistellig, $-$ einstellig und x und y nullstellig seien. Dann ist



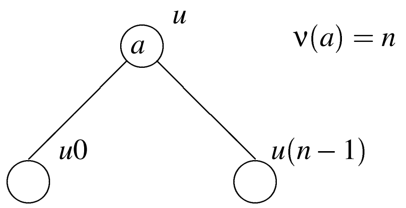
ein <u>endlicher Σ- beschrifteter Baum</u>. Man kann Knoten dieses Baumes eindeutig durch Wörter aus $\{0, 1\}^*$ ansprechen.
t entspricht einer partiellen Funktion $t: \{0, 1\}^* \to \Sigma$ mit dem Definitionsbereich $dom(t) = \{\varepsilon, 0, 1, 00, 01, 10\}$. Zum Beispiel ist $t(0) = \cdot$.

□

Definition 6.2
Es sei Σ ein Alphabet und $\nu: \Sigma \to \omega$ eine Funktion, die jedem $a \in \Sigma$ eine Stelligkeit $\nu(a)$ zuordnet (<u>Alphabet mit Stelligkeitsfunktion</u>). Für $n \in \omega$ sei $\Sigma_n := \{a \in \Sigma \mid \nu(a) = n\}$.
Ein <u>Σ-Baum</u> ist eine partielle Funktion $t: \omega^* \to \Sigma$, deren Definitionsbereich $dom(t)$ erfüllt:

1. $\varepsilon \in dom(t)$, das heißt jeder Baum ist nicht-leer

2. Für alle $u \in \omega^*$, $i \in \omega$ gilt:
   $ui \in dom(t)$ gdw $u \in dom(t)$ *und* $i < \nu(t(u))$



Ein <u>Blatt</u> u von t ist ein Knoten $u \in dom(t)$ mit $\nu(t(u)) = 0$, das heißt u hat keine Nachfolgerknoten.
Der Baum t ist <u>endlich</u>, falls $dom(t)$ endlich ist. Mit T_Σ bezeichnet man die Menge aller endlichen Σ-Bäume.
Es sei $\prec$ die <u>Präfixrelation</u> auf ω^*, das heißt $u \prec v$ gdw $\exists u' \in \omega^+ : \; uu' = v$.

□

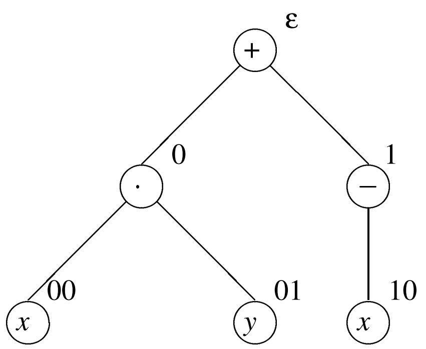
Wegen 2. von Definition 6.2 gilt für jeden Baum t, daß $dom(t)$ unter Präfixbildung abgeschlossen ist, das heißt ist $v \in dom(t)$ und $u \prec v$, so ist $u \in dom(t)$.

Definition 6.3

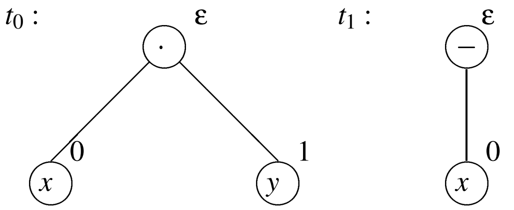
1. Ein <u>Pfad</u> durch t ist eine maximale, bezüglich $\prec$ total geordnete Teilmenge von $dom(t)$.
2. Der Unterbaum von t an der Stelle $u \in dom(t)$ ist der Baum t_u mit
   - $dom(t_u) = \{v \mid uv \in dom(t)\}$
   - $t_u(v) = t(uv)$

□

Beispiel 6.1 (Fortsetzung)

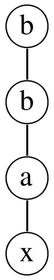


In diesem Baum existieren die Pfade $\{\varepsilon,\ 0,\ 00\}$, $\{\varepsilon,\ 0,\ 01\}$ und $\{\varepsilon,\ 1,\ 10\}$.
$\{\varepsilon,\ 0,\ 00,\ 01\}$ ist kein Pfad, denn er wäre nicht total geordnet; $\{\varepsilon,\ 00\}$ wäre nicht maximal.

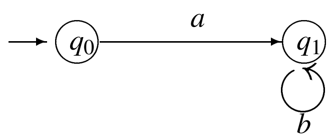


□

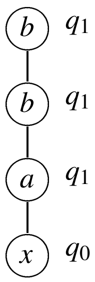
Ein Wort $w \in \Sigma^*$ für ein endliches Alphabet Σ kann als endlicher Baum über $\hat{\Sigma} = \Sigma \cup \{x\}$ angesehen werden, wobei $\nu(a) = 1$ für $a \in \Sigma$ und $\nu(x) = 0$ definiert wird.
Mit $\Sigma = \{a, b\}$ entspricht *abb* dann dem Baum



Wir betrachten nun den Automaten



Ein Pfad im Automaten mit Beschriftung *abb* kann in dem Baum durch eine Zusatzbeschriftung der Knoten mit Zuständen des Automaten beschrieben werden:



□

Definition 6.4
Ein BW-Baumautomat (Blatt zur Wurzel) $\mathcal{A} := (Q, \Sigma, I, \Delta, F)$ besteht aus

- einer endlichen Zustandsmenge Q
- einem endlichen Alphabet Σ mit Stelligkeitsfunktion (das heißt $\Sigma = \Sigma_0 \cup \Sigma_1 \cup \Sigma_2 \cup \ldots$)

- einer Anfangszuweisung $I: \Sigma_0 \to 2^Q$
- einer Übergangszuweisung Δ, die für $n > 0$ jedem $a \in \Sigma_n$ eine Funktion $\Delta_a: Q^n \to 2^Q$ zuweist
- einer Endzustandsmenge F

Ein Lauf dieses Baumautomaten auf einem Baum $t \in T_\Sigma$ ist eine Abbildung $l: dom(t) \to Q$ mit

- $l(u) \in \Delta_a(l(u0), \ldots, l(u(n-1)))$ für $t(u) = a \in \Sigma_n$.

Der Lauf l ist erfolgreich, falls

- $l(u) \in I(t(u))$ für alle Blätter u
- $l(\varepsilon) \in F$

Die von $\mathcal{A}$ akzeptierte Baumsprache ist

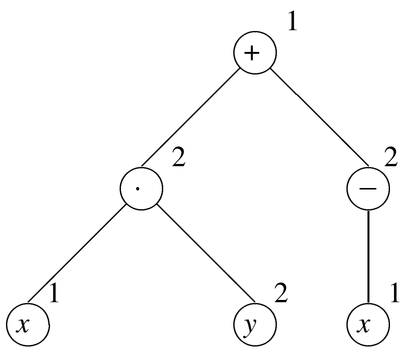
$$L(\mathcal{A}) := \{t \in T_\Sigma \mid es\ gibt\ erfolgreichen\ Lauf\ von\ \mathcal{A}\ auf\ t\}$$

$\mathcal{A}$ heißt deterministisch, wenn für alle $a \in \Sigma_0$ gilt: $|I(a)| = 1$ und für alle $n > 0$ und $a \in \Sigma_n, q_1, \ldots, q_n \in Q$ gilt $|\Delta_a(q_1, \ldots, q_n)| = 1$. Wir schreiben dann I und Δ_a als Funktionen $I: \Sigma_0 \to Q$ und $\Delta_a: Q^n \to Q$.

□

Beispiel 6.5
$\Sigma = \Sigma_0 \cup \Sigma_1 \cup \Sigma_2$ mit $\Sigma_0 = \{x, y\}$, $\Sigma_1 = \{-\}$ und $\Sigma_2 = \{+, \cdot\}$. $\mathcal{A} := (Q, \Sigma, I, \Delta, F)$ mit

- $Q = \{0, 1, 2\}$
- $I(x) = 1, I(y) = 2$
- $\Delta_-(q) = 3 - q\ mod\ 3$
  $\Delta_+(q, q') = q + q'\ mod\ 3$
  $\Delta_\cdot(q, q') = q \cdot q'\ mod\ 3$
- $F = \{1\}$



ist ein erfolgreicher Lauf.
$L(\mathcal{A})$ besteht aus den arithmetischen Ausdrücken, deren Auswertung mit $x = 1$ und $y = 2\ mod\ 3$ als Resultat 1 liefert.

□

Wie bei Automaten für Wörter kann man einen nicht-deterministischen BW-Automaten durch Potenzmengenkonstruktion in einen äquivalenten deterministischen BW-Automaten überführen.

Satz 6.6
Für eine Baumsprache $L \subseteq T_\Sigma$ sind äquivalent:

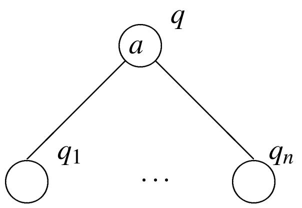
1. L wird von einem deterministischen BW-Automaten akzeptiert
2. L wird von einem BW-Automaten akzeptiert

□

Anstatt von den Blättern zur Wurzel kann man auch von der Wurzel zu den Blättern gehen.

Definition 6.7
Ein WB-Automat (Wurzel zum Blatt) $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ besteht aus

- einer endlichen Zustandsmenge Q
- einem Alphabet Σ mit Stelligkeitsfunktion
- einer Menge $I \subseteq Q$ von Anfangszuständen
- einer Übergangszuweisung, die jedem $a \in \Sigma$ für $n > 0$ eine Funktion $\Delta_a : Q \to 2^{(Q^n)}$ zuweist



- einer Endzuweisung $F: \ \Sigma_0 \to 2^Q$

Ein Lauf von $\mathcal{A}$ auf $t \in T_\Sigma$ ist eine Abbildung $l: \ dom(t) \to Q$ mit $(l(u0), \ldots, l(u(n-1))) \in \Delta_a(l(u))$ für $t(u) = a \in \Sigma_n$.
Der Lauf l ist erfolgreich, falls $l(\varepsilon) \in I$ und $l(u) \in F(t(u))$ für alle Blätter u.

$$L(\mathcal{A}) = \{t \in T_\Sigma \mid es\ gibt\ einen\ erfolgreichen\ Lauf\ von\ \mathcal{A}\ auf\ t\}.$$

$\mathcal{A}$ heißt deterministisch, wenn $|I| = 1$ und $|\Delta_a(q)| = 1$ für alle $n > 0$, $a \in \Sigma_n$, $q \in Q$.

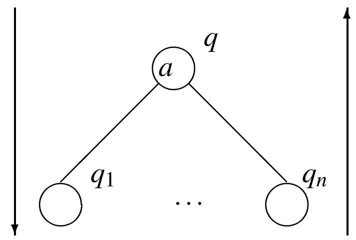
□

Satz 6.8
Für eine Baumsprache $L \subseteq T_\Sigma$ sind äquivalent:

1. L wird von einem BW-Automaten akzeptiert
2. L wird von einem WB-Automaten akzeptiert

Beweis

"$1 \Rightarrow 2$" Es sei $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein BW-Automat. Wir betrachten den WB-Automaten $\mathcal{B} = (Q, \Sigma, F, \Delta', I)$ mit $\Delta'_a : \quad q \mapsto \{(q_1, \ldots, q_n) \mid q \in \Delta_a(q_1, \ldots, q_n)\}$



Man sieht leicht, daß jeder Lauf von $\mathcal{A}$ auf t auch ein Lauf von $\mathcal{B}$ auf t ist und umgekehrt.

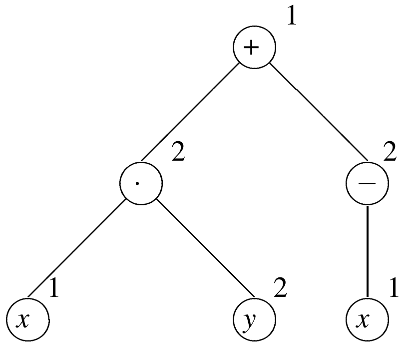
"$2 \Rightarrow 1$" entsprechend

□

Beispiel 6.9

Es sei $\mathcal{A}$ der BW-Automat aus Beispiel 6.5. Der zugehörige WB-Automat $\mathcal{B} = (Q, \Sigma, I', \Delta', F')$ ist gegeben durch:

- $Q = \{0, 1, 2\}$
- $\Sigma = \{+, \cdot, -, x, y\}$
- $I' = \{1\}$
- $\Delta'_-(q) = \{q' \mid \Delta_-(q') = q\} = \{q' \mid 3 - q' \; mod \; 3 = q\}$
  $\Delta'_+(q) = \{(q', q'') \mid q = q' + q'' \; mod \; 3\}$
  $\Delta'_\cdot(q) = \{(q', q'') \mid q = q' \cdot q'' \; mod \; 3\}$
- $F'(x) = \{1\}$, $F'(y) = \{2\}$

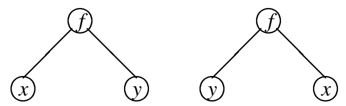


ist ein erfolgreicher Lauf, der dem aus Beispiel 6.5 entspricht.

Beachte: Obwohl $\mathcal{A}$ deterministisch war, ist $\mathcal{B}$ nicht-deterministisch: $\Delta'_+(1) = \{(q', q'') \mid 1 = q' + q'' \; mod \; 3\} = \{(0,1), (1,0), (2,2)\}$ □

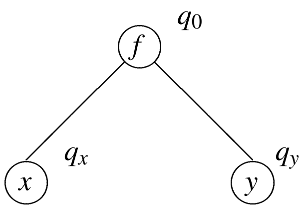
Beispiel 6.10

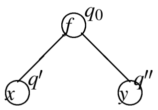
Nicht jede von einem nicht-deterministischen WB-Automaten akzeptierte Sprache kann auch von einem deterministischen WB-Automaten akzeptiert werden.

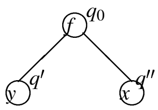
Es sei $\Sigma = \{ \underbrace{x,\ y}_{0-stellig}\ ,\ \underbrace{f}_{2-stellig} \}$ und $L = \Big\{$  $\Big\}$

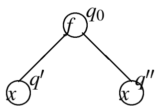
1. L wird akzeptiert von dem *nicht*-deterministischen WB-Automaten $\mathcal{A} = (\{q_0,\, q_1,\, q_x,\, q_y\},\, \Sigma,\, \{q_0,\, q_1\},\, \Delta,\, F)$ mit

   - $\Delta_f(q_0) = \{(q_x,\, q_y)\}$
     $\Delta_f(q_1) = \{(q_y,\, q_x)\}$
     $\Delta_f(q_x) = \Delta_f(q_y) = \emptyset$
   - $F(x) = \{q_x\}\ F(y) = \{q_y\}$

2. Angenommen $\mathcal{B} = (Q',\, \Sigma,\, \{i\},\, \Delta',\, F')$ ist ein *deterministischer* WB-Automat, der L akzeptiert. Dann gilt für $(q', q'') = \Delta'_f(i)$ wegen  $\in L$, daß $q' \in F(x)$.

   Wegen  $\in L$ folgt $q'' \in F(x)$.

   Daher akzeptiert $\mathcal{B}$ auch  $\notin L$.

□

Wir haben gesehen: deterministische und nicht-deterministische BW-Automaten sowie nicht-deterministische WB-Automaten akzeptieren dieselbe Sprachklasse. Deterministische WB-Automaten akzeptieren eine kleinere Klasse von Baumsprachen.

Definition 6.11
Eine Baumsprache $L \subseteq T_\Sigma$ heißt <u>erkennbar</u>, wenn sie von einem BW-Automaten akzeptiert wird.

□

Beispiel 6.12
Nicht jede Baumsprache ist erkennbar.
Es sei Σ ein Alphabet mit Stelligkeitsfunktion, für das gilt $|\Sigma_0| > 0$ und $|\Sigma_2| > 0$. Offenbar folgt daraus, daß T_Σ unendlich ist.
$L := \{f(t,t) \mid t \in T_\Sigma\} \;=\; \{t' \in T_\Sigma \mid t'(\Sigma) = f,\ t'_0 = t'_1\}$
Dabei ist $f(t,t)$ der Baum, der f als Wurzel hat und nur zwei identische Unterbäume t.
Behauptung: L ist nicht erkennbar.
Beweis: Angenommen $\mathcal{A} = (Q,\Sigma,I,\Delta,F)$ ist ein ohne Einschränkung deterministischer BW-Automat für L. Für jeden Baum $t \in T_\Sigma$ sei q_t der (eindeutig bestimmte) Zustand aus Q mit der Eigenschaft, daß bei dem Lauf l auf t mit $I(t(u)) = l(u)$ für alle Blätter u q die Wurzel beschriftet.
Da Q endlich und T_Σ unendlich ist gibt es $t,\ t' \in T_\Sigma$ mit $t \neq t'$, aber $q_t = q_{t'}$. Wir betrachten einen Lauf von $\mathcal{A}$ auf $\Delta_f(q_t,q_t) \in L$ und $\Delta_f(q_t,q_{t'}) \notin L$. Wegen $q_t = q_{t'}$ ist $\Delta_f(q_t,q_t) = \Delta_f(q_t,q_{t'}) \in F$, das heißt $\Delta_f(q_t,q_{t'})$ wird von $\mathcal{A}$ akzeptiert.
Widerspruch

□

Bei Wörtern können erkennbare Sprachen durch reguläre Ausdrücke beschrieben werden.
Bei erkennbaren Baumsprachen ist eine ähnliche Charakterisierung möglich.
Dazu muß man geeignete Operationen auf Baumsprachen einführen.
Zur Erinnerung (bei Wörtern galt folgendes):

- $\emptyset \in Reg_\Sigma,\ a \in Reg_\Sigma$ für alle $a \in \Sigma$
- $L_1,\ L_2 \in Reg_\Sigma \;\Rightarrow\; L_1 \cup L_2,\ L_1 \cdot L_2,\ L_1^* \in Reg_\Sigma$

Satz 6.13

1. Die leere Baumsprache ist erkennbar
2. Für $a \in \Sigma_0$ ist der Baum mit Wurzel a erkennbar

Beweis

1. Verwende zum Beispiel einen BW-Automaten mit $F = \emptyset$
2. $Q = \{0,1\}, I(a) = 1, I(b) = 0$ für $b \in \Sigma_0 \backslash \{a\}$, $\Delta_f(q_1,\ldots,q_n) = \emptyset$ für alle $f \in \Sigma_0$ mit $n > 0, I = \{1\}$

□

Satz 6.14
Die Klasse der erkennbaren Baumsprachen ist abgeschlossen unter Vereinigung, Komplement und Durchschnitt.

Beweis
Der Beweis läuft wie bei Wörtern. Man braucht nur Abschluß unter Vereinigung und Komplement zu zeigen, da Durchschnitt sich daraus ergibt.

- Vereinigung: vereinige zwei BW-Automaten mit disjunkten Zustandsmengen zu einem nicht-deterministischen BW-Automaten
- Komplement: verwende einen *deterministischen* BW-Automaten und mache Endzustände zu nicht-Endzuständen und umgekehrt

□

Anstelle von Wortkonkatenation wird Baumkonkatenation verwendet. Der Baum mit Wurzel f und den Unterbäumen $t_1, \ldots, t_n$ wird als $f(t_1,\ldots,t_n)$ geschrieben.

Definition 6.15
Es sei Σ ein Alphabet mit Stelligkeitsfunktion und $\overline{x} = (x_1,\ldots,x_k)$ ein k-Tupel verschiedener Elemente von Σ_0.

1. Für $t \in T_\Sigma$ und $L_1,\ldots,L_k \subseteq T_\Sigma$ wird $t \cdot^{\overline{x}} (L_1,\ldots,L_k)$ induktiv definiert:
   - $t \in \Sigma_0$:
     - $t = x_i : t \cdot^{\overline{x}} (L_1,\ldots,L_k) = L_i$
     - $t \neq x_i$ für alle i: $t \cdot^{\overline{x}} (L_1,\ldots,L_k) = \{t\}$
   - $t = f(t_1,\ldots,t_n)$ für $f \in \Sigma_n$, $n > 0$:
     $t \cdot^{\overline{x}} (L_1,\ldots,L_k) = \{f(t'_1,\ldots,t'_n) \mid t'_i \in t_i \cdot^{\overline{x}} (L_1,\ldots,L_k)\}$
2. für $L, L_1,\ldots,L_k \subseteq T_\Sigma$ ist
   $L \cdot^{\overline{x}} (L_1,\ldots,L_k) := \{t \cdot^{\overline{x}} (L_1,\ldots,L_k) \mid t \in L\}$

Ein Baum in $L \cdot^{\overline{x}} (L_1,\ldots,L_k)$ entsteht aus einem Baum $t \in L$ dadurch, daß man jedes Blatt mit Beschriftung x_i durch ein Element von L_i ersetzt. □

Beispiel
$\{f(x,x)\} \cdot^{x} \{a,b\} = \{f(a,a), f(b,b), f(a,b), f(b,a)\}$
Beachte: Verschiedene Vorkommen von x_i können durch verschiedene Elemente von L_i ersetzt werden.
Insgesamt ist $\{f(x,x)\} \cdot^{x} T_\Sigma = \{f(t,t') \mid t,t' \in T_\Sigma\} \neq \{f(t,t) \mid t \in T_\Sigma\}$ □

Satz 6.16
Sind L, $L_1, \ldots, L_k$ erkennbare Baumsprachen, so ist auch $L \cdot^{\overline{x}} (L_1,\ldots,L_k)$ erkennbar.

Beweis
Es sei $\mathcal{A} = (Q,\Sigma,I,\Delta,F)$ ein WB-Automat für L und $\mathcal{A}_i = (Q^{(i)},\Sigma,I^{(i)},\Delta^{(i)},F^{(i)})$ sei WB-Automat für L_i mit $i = 1,\ldots,k$.
Ohne Einschränkung sind Q, $Q^{(1)}, \ldots, Q^{(k)}$ paarweise disjunkt.
Wir betrachten $\mathcal{B} = (Q \cup Q^{(1)} \cup \ldots \cup Q^{(k)},\Sigma,I,\Delta',F')$ mit

- für $a \in \Sigma_n$ mit $n > 0$:

- für $q \in Q^{(j)}$ mit $j = 1, \ldots, k$:

$$\Delta'_a(q) := \Delta_a^{(j)}(q)$$

- für $q \in Q$:

$$\Delta'_a(q) := \Delta_a(q) \cup \bigcup_{\substack{j \ mit \\ q \in F(x_j)}} \{\Delta_a^{(j)}(i) \mid i \in I^{(j)}\}$$

- für $a \in \Sigma_0$:
  - $a \notin \{x_1, \ldots, x_k\}$:
  $$\begin{array}{lll} F'(a) := & F(a) \cup & F^{(1)}(a) \cup \ldots \cup F^{(k)}(a) \cup \\ & \{q \in Q \mid & q \in F(x_j) \ \mathit{für\ ein}\ j\ \mathit{mit}\ 1 \leq j \leq k \\ & & \mathit{und}\ F^{(j)}(a) \cap I^{(j)} \neq \emptyset\} \end{array}$$
  - $a \in \{x_1, \ldots, x_k\}$:
  $$\begin{array}{lll} F'(a) := & F^{(1)}(a) \cup & \ldots \cup F^{(k)}(a) \cup \\ & \{q \in Q \mid & q \in F(x_j) \ \mathit{für\ ein}\ j\ \mathit{mit}\ 1 \leq j \leq k \\ & & \mathit{und}\ F^{(j)}(a) \cap I^{(j)} \neq \emptyset\} \end{array}$$

Man sieht leicht, daß $\mathcal{B}$ ein WB-Automat für $L \cdot^{\overline{x}} (L_1, \ldots, L_k)$ ist.

□

Wir werden Baumkonkatenation später in zwei Spezialfällen verwenden:

1. Anwenden eines $f \in \Sigma_n$ $(n > 0)$ auf Baumsprachen $L_1, \ldots, L_n$:

$$f(L_1, \ldots, L_n) := \{f(x_1, \ldots, x_n)\} \cdot^{(x_1, \ldots, x_n)} (L_1, \ldots, L_n)$$

2. $\overline{x} = x$, das heißt $\overline{x}$ ist ein Tupel der Dimension 1: $L \cdot^x L'$

Der Sternoperator kann auf Baumsprachen wie folgt verallgemeinert werden:

Definition 6.17
Es sei $L \subseteq T_\Sigma$ und $x \in \Sigma_0$. Wir definieren

- $L^{0,x} := \{x\}$
- $L^{n+1,x} := L^{n,x} \cup L \cdot^x L^{n,x}$
- $L^{*,x} := \bigcup_{n \geq 0} L^{n,x}$

□

Satz 6.18
Ist L eine erkennbare Baumsprache, so auch $L^{*,x}$.

Beweis
Es sei $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein WB-Automat für L. Definiere $\mathcal{B} = (Q \cup \{\hat{i}\}, \Sigma, I', \Delta', F')$ mit $\hat{i} \notin Q$

- $I' = I \cup \{\hat{i}\}$
- für $a \in \Sigma_n,\ n > 0$
  $\Delta'_a(q) = \Delta_a(q) \cup \begin{cases} \emptyset & q \notin F(x) \\ \bigcup_{i \in I} \Delta_a(i) & q \in F(x) \end{cases}$
- für $a \in \Sigma_0$:
  - $a \neq x$: $F'(a) = F(a) \cup \{q \in Q \mid q \in F(x) \cap F(a) \cap I \neq \emptyset\}$
  - $a = x$: $F'(x) = F(x) \cup \underbrace{\{\hat{i}\}}_{\substack{\text{sorgt dafür, daß} \\ x \text{ akzeptiert wird}}}$

□

Definition 6.19
Es sei Σ ein Alphabet mit Stelligkeitsfunktion und Z eine Menge nullstelliger Symbole mit $Z \cap \Sigma = \emptyset$ und $\hat{\Sigma} = \Sigma \cup Z$.
$\underline{Reg(T_\Sigma, Z)}$ ist die kleinste Klasse von Baumsprachen über $\hat{\Sigma}$ mit

1. $\emptyset \in Reg(T_\Sigma, Z)$
2. $\{x\} \in Reg(T_\Sigma, Z)$ für alle $x \in \Sigma_0 \cup Z$
3. $L_1, L_2 \in Reg(T_\Sigma, Z) \ \Rightarrow\ L_1 \cup L_2 \in Reg(T_\Sigma, Z)$
4. $L_1, L_2 \in Reg(T_\Sigma, Z),\ z \in Z \ \Rightarrow\ L_1 \cdot^z L_2 \in Reg(T_\Sigma, Z)$
5. $L \in Reg(T_\Sigma, Z),\ z \in Z \ \Rightarrow\ L^{*,z} \in Reg(T_\Sigma, Z)$
6. $n > 0,\ f \in \Sigma_n,\ L_1, \ldots, L_n \in Reg(T_\Sigma, Z) \ \Rightarrow\ f(L_1, \ldots, L_n) \in Reg(T_\Sigma, Z)$

Eine Sprache $L \subseteq T_\Sigma$ heißt regulär, falls es ein Hilfsalphabet Z von nullstelligen Symbolen gibt mit $L \in Reg(T_\Sigma, Z)$.

□

Satz 6.20
Für $L \subseteq T_\Sigma$ sind äquivalent:

1. L regulär
2. L ist erkennbar

Beweis

"$1 \Rightarrow 2$" ergibt sich unmittelbar aus den bereits gezeigten Abschlußeigenschaften erkennbarer Baumsprachen.

"$2 \Rightarrow 1$" Es sei $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein WB-Automat mit $L = L(\mathcal{A})$. Ohne Einschränkung sei $Q = \{1, \ldots, k\}$ und $Q \cap \Sigma = \emptyset$. Wir definieren $Z := Q$, wobei jedes $q \in Q$ Stelligkeit 0 erhält.
Es sei $\mathcal{A}' = (Q, \Sigma \cup Z, I, \Delta, F')$ mit $F'(q) = \{q\}$ für alle $q \in Z$, $F'(q) = F(q)$ sonst. Für $K \subseteq Q$, $0 \leq h \leq k$ und $i \in Q$ sei $L(K, h, i)$ die Menge aller Bäume $t \in T_{\Sigma \cup K}$, für die es einen Lauf l von $\mathcal{A}'$ auf t gibt mit

- $l(\varepsilon) = i$
- $l(u) \leq h$ für alle $u \neq \varepsilon$, die nicht Blätter sind
- $l(u) \in F'(t(u))$ für alle Blätter u

$L(K, h, i)$ enthält Bäume, die zusätzlich Blätter haben können, welche mit $q \in K$ beschriftet sind.
Ein Lauf des Automaten $\mathcal{A}'$ muß i an der Wurzel liefern, an den Blättern u mit Zustand aus $F'(t(u))$ enden (bei $t(u) = q \in K$ also mit q) und er darf nur Zwischenzustände $\leq h$ verwenden.
Offenbar gilt $L(\mathcal{A}) = \bigcup_{i \in I} L(\emptyset, k, i)$.
Es genügt daher zu zeigen, daß alle $L(K, h, i)$ in $Reg(T_\Sigma, Z)$ sind.
Induktion über h:
$h = 0$
Es darf keine Zwischenzustände geben, das heißt in $L(K, 0, i)$ gibt es keine Bäume, welche Knoten u haben, die $\neq \varepsilon$ und keine Blätter sind. Damit ist $L(K, h, i)$ endlich. Man zeigt leicht, daß alle endlichen Teilmengen von $T_{\Sigma \cup Z}$ in $Reg(T_\Sigma, Z)$ sind.
$h > 0$
Es gilt

$$\begin{aligned}(*)\ L(K, h+1, i) \quad = \quad & L(K, h, i) \cup \\ & L(K \cup \{h+1\}, h, i) \cdot^{h+1} \\ & L(K \cup \{h+1\}, h, h+1)^{*, h+1} \cdot^{h+1} \\ & L(K, h, h+1)\end{aligned}$$

Der Beweis von $(*)$ erfolgt in der Übung.
Nach Induktion und wegen der Definition von $Reg(T_\Sigma, Z)$ liefert $(*)$, daß $L(K, h+1, i) \in Reg(T_\Sigma, Z)$ ist.

□

Für den Abschluß unter Alphabetsumbenennung seien $\Sigma^{(1)}$ und $\Sigma^{(2)}$ Alphabete mit Stelligkeitsfunktion. $\varphi : \Sigma^{(1)} \to \Sigma^{(2)}$ für die gilt: $\varphi(\Sigma_n^{(1)}) \subseteq \Sigma_n^{(2)}$. Sei $t : dom(t) \to \Sigma^{(1)}$ aus $T_{\Sigma^{(1)}}$. Dann ist $\varphi(t) : dom(t) \to \Sigma^{(2)}$ definiert durch $\varphi(t)(u) := \varphi(t(u))$.

Satz 6.21
Ist $L \subseteq T_{\Sigma^{(1)}}$ erkennbar, so auch $\varphi(L) := \{\varphi(t) \mid t \in L\}$.

Beweis
Es sei $\mathcal{A} = (Q, \Sigma^{(1)}, I, \Delta, F)$ ein BW-Automat für L. Wir definieren $\mathcal{A}' = (Q, \Sigma^{(2)}, I', \Delta', F)$, wobei

- für $a \in \Sigma_0^{(2)}$:

$$I'(a) = \bigcup_{\substack{a' \in \Sigma_0^{(1)} \\ \varphi(a') = a}} I(a')$$

- für $a \in \Sigma_n^{(2)}$, $n > 0$:

$$\Delta'_a(q_1, \ldots, q_n) = \bigcup_{\substack{a' \in \Sigma_n^{(1)} \\ \varphi(a') = a}} \Delta_{a'}(q_1, \ldots, q_n)$$

Man zeigt leicht, daß $L(\mathcal{A}') = \varphi(L)$.

□

Satz 6.22
Für reguläre Baumsprachen sind das Äquivalenzproblem und das Leerheitsproblem entscheidbar.

Beweis
Da reguläre Baumsprachen unter den Booleschen Operationen abgeschlossen sind genügt es, das Leerheitsproblem zu betrachten.
Es sei $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein BW-Automat mit $|Q| = k$ Zuständen.
Behauptung: $L(\mathcal{A}) \neq \emptyset$ genau dann, wenn es einen Baum t der Tiefe $\leq k$ gibt mit $t \in L(\mathcal{A})$.
Da es für ein endliches Alphabet Σ nur endlich viele Bäume aus T_Σ der Tiefe $\leq k$ gibt, kann man alle diese Bäume testen.
Beweis der Behauptung:

“$\Leftarrow$” trivial

“$\Rightarrow$” Es sei t ein Baum minimaler Größe in $L(\mathcal{A})$. Angenommen t hat eine Tiefe $> k$. Betrachte einen erfolgreichen Lauf von $\mathcal{A}$ auf t. Es gibt auf diesem Pfad der Länge $> k$ zwei verschiedene Stellen u, u' mit $l(u) = l(u')$. Ersetzt man in t den Unterbaum t_u durch $t_{u'}$, so hat man auf dem so erhaltenen Baum immernoch einen erfolgreichen Lauf. Der neue Baum ist aber kleiner als t. Dies ist ein Widerspruch zu der Voraussetzung, daß t minimale Größe hat.

□

Kapitel 7

Automaten auf unendlichen Bäumen

Der Einfachheit halber betrachten wir nur binäre unendliche Bäume, das heißt das Alphabet Σ enthält nur zweistellige Symbole ($\Sigma = \Sigma_2$). Alle Resultate lassen sich aber auf den allgemeinen Fall übertragen.
Mit $\underline{T_\Sigma^\omega}$ bezeichnen wir die Menge der unendlichen Bäume über dem Alphabet Σ. Eine Menge $L \subseteq T_\Sigma^\omega$ heißt <u>ω-Baumsprache</u> und $t \in T_\Sigma^\omega$ heißt <u>ω-Baum</u> .
Man kann ω-Baumsprachen aus Baumsprachen durch unendliche Iteration erhalten.
Definition 7.1
Sei $Z = \{z_1, \dots, z_k\}$ eine Menge nullstelliger Symbole und Σ ein Alphabet binärer Symbole. $U, U_1, \dots, U_k \subseteq T_{\Sigma \cup Z}$ seien Baumsprachen über $\Sigma \cup Z$. Die ω-Baumsprache

$$U \cdot^{(z_1,\dots,z_k)} (U_1, \dots, U_k)^{\omega,(z_1,\dots,z_k)}$$

besteht aus allen ω-Bäumen $t \in T_\Sigma^\omega$, für die es eine Folge $t_0, t_1, t_2, \dots$ von Bäumen aus $T_{\Sigma \cup Z}$ gibt mit

1. $t_0 \in U$
2. für alle $i \geq 0$ ist $t_{i+1} \in \{t_i\} \cdot^{(z_1,\dots,z_k)} (U_1, \dots, U_k)$
3. t ist Limes dieser Folge, das heißt für alle $u \in \{0,1\}^*$ gibt es ein $m \geq 0$ mit
   - $u \in dom(t_m)$ und u ist kein Blatt von t_m (d.h. $t_m(u) \in \Sigma$)
   - $t(u) = t_m(u)$

□

Beispiel 7.2
$Z = \{z_1, z_2\}, U = \{f(z_1, z_2)\}, U_1 = \{h(z_1, z_1)\}, U_2 = \{g(z_2, z_2)\}$
Da $t_0 \in U$ gelten muß, ist $t_0 = f(z_1, z_2)$.
$t_1 = f(h(z_1, z_1), g(z_2, z_2))$ (ersetze z_1 durch U_1 und z_2 durch U_2)

$t_2 = f(h(h(z_1, z_1), h(z_1, z_1)), g(g(z_2, z_2), g(z_2, z_2)))$

Es gibt *hier* nur eine mögliche Folge, deren Limes
$t = f(h(h(h(\dots), h(\dots)), h(h(\dots), h(\dots))), g(g(g(\dots), g(\dots)), g(g(\dots), g(\dots))))$
ist. □

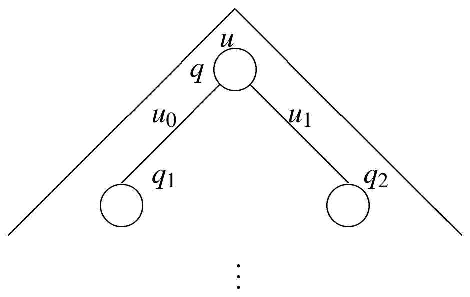
Da unendliche Bäume keine Blätter enthalten, betrachten wir Automaten, deren Lauf an der Wurzel eines Baums beginnt.

Definition 7.3

1. Ein Büchi-Baumautomat über dem Alphabet Σ (mit $\Sigma = \Sigma_2$) ist von der Form $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$, wobei Q, I, Δ wie bei WB-Automaten definiert sind und $F \subseteq Q$ eine Menge von Endzuständen ist.

2. Ein Rabin-Baumautomat über dem Alphabet Σ (mit $\Sigma = \Sigma_2$) ist von der Form $\mathcal{A} = (Q, \Sigma, I, \Delta, \Omega)$, wobei Q, I, Δ wie bei Büchi-Baumautomaten definiert sind und $\Omega = \{(F_1, G_1), \dots, (F_n, G_n)\}$ mit $F_i, G_i \subseteq Q$ für $i = 1, \dots, n$.

Ein Lauf l eines (Büchi- oder Rabin-) Baumautomaten auf einem Baum $t \in T_\Sigma^\omega$ ist wie im endlichen Fall definiert, das heißt $l : dom(t) = \{0,1\}^* \to Q$ mit $(l(u0), l(u1)) \in \Delta_f(l(u))$, falls $t(u) = f$.
Ein Lauf ist also ein unendlicher binärer Baum über dem Alphabet Q, wobei jedes $q \in Q$ zweistellig ist.



Ein Lauf l eines Büchi-Baumautomaten heißt erfolgreich, falls $l(\varepsilon) \in I$ und jeder Pfad in l enthält unendlich oft einen Zustand aus F.
Ein Lauf l eines Rabin-Baumautomaten heißt erfolgreich, falls $l(\varepsilon) \in I$ und für jeden Pfad in l gibt es ein i ($1 \leq i \leq n$) mit:

- der Pfad enthält unendlich oft einen Zustand aus F_i
- der Pfad enthält keinen Zustand aus G_i unendlich oft

□

Es wird hier also die Akzeptanzbedingung für Büchi- bzw. Rabin-Baumautomaten auf unendlichen Wörtern auf die Pfade im Baum angewendet. *Beachte:* Jeder Pfad in $t \in T_\Sigma^\omega$ liefert ein Wort aus Σ^ω.
$L_\omega(\mathcal{A}) = \{t \in T_\Sigma^\omega \mid \text{es gibt einen erfolgreichen Lauf von } \mathcal{A} \text{ auf } t\}$ ist die von dem Automaten $\mathcal{A}$ erkannte Sprache. $L \subseteq T_\Sigma^\omega$ heißt Büchi-erkennbar (Rabin-erkennbar) falls L von einem Büchi-Baumautomaten (Rabin-Baumautomaten) erkannt wird.

Satz 7.4
Jede Büchi-erkennbare Sprache ist auch Rabin-erkennbar.

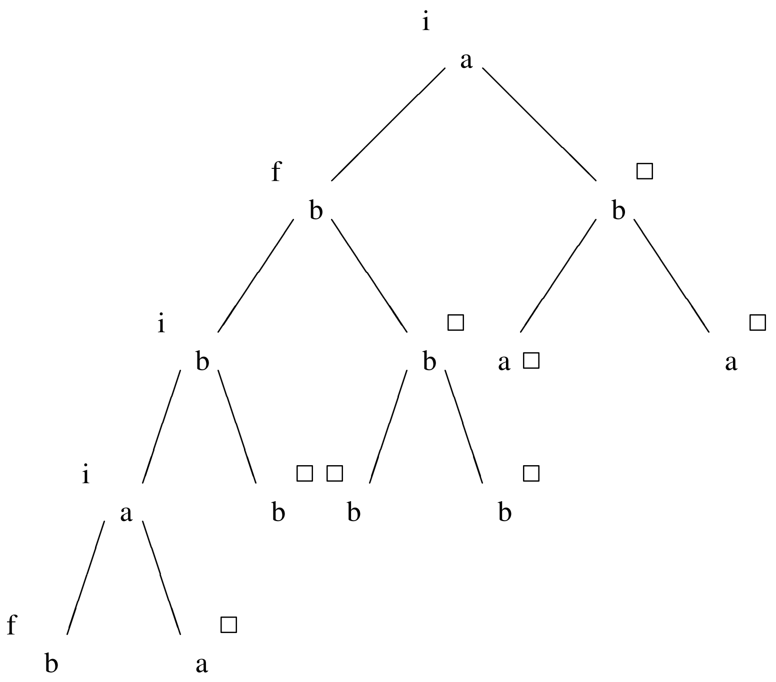
Beweis
Es sei $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein Büchi-Baumautomat. Wir definieren den Rabin-Automaten $\mathcal{A}' = (Q, \Sigma, I, \Delta, \{F, \emptyset\})$. Offenbar git $L_\omega(\mathcal{A}) = L_\omega(\mathcal{A}')$. $\square$

Die folgenden Beispiele sollen den Unterschied zwischen Büchi- und Rabin-Automaten zeigen.

Beispiel 7.5
$\Sigma = \{a, b\}$ und $L_1 = \{t \in T_\Sigma^\omega \mid$ es gibt einen Pfad in t, der unendlich viele a's enthält $\}$.
Der folgende *nicht-deterministische* Büchi-Baumautomat für L_1 "rät" einen geeigneten Pfad und nimmt immer, wenn er a liest, einen Endzustand f an. Bei den nicht geratenen Pfaden wird ein anderer Endzustand reproduziert (der unendlich oft vorkommt).
$\mathcal{A}_1 = (\{i, f, \square\}, \Sigma, \{i\}, \Delta_1, \{f, \square\})$ mit

$$
\begin{array}{llll}
\Delta_{1a}: & i & \mapsto & \{(f,\square),(\square,f)\} \\
 & f & \mapsto & \{(f,\square),(\square,f)\} \\
 & \square & \mapsto & \{(\square,\square)\} \\
\Delta_{1b}: & i & \mapsto & \{(i,\square),(\square,i)\} \\
 & f & \mapsto & \{(i,\square),(\square,i)\} \\
 & \square & \mapsto & \{(\square,\square)\}
\end{array}
$$

Der geratene Pfad ist derjenige, welcher mit Zuständen aus $\{i, f\}$ beschriftet ist. Er enthält unendlich viele a's genau dann, wenn der Lauf dort unendlich viele f's enthält. Alle anderen Pfade enthalten unendlich oft $\square$.



$\square$

Beispiel 7.6
$\Sigma = \{a, b\}$ und $L_2 = \{t \in T_\Sigma^\omega \mid$ jeder Pfad in t enthält nur endlich viele a's $\}$, das heißt $L_2 = T_\Sigma^\omega \setminus L_1$.

Der Rabin-Automat für L_2 ist $\mathcal{A}_2 = (\{i,f\},\Sigma,\{i\},\Delta_2,\Omega_2)$ mit

$$\begin{array}{llll} \Delta_{2a} & i & \mapsto & \{(f,f)\} \\ & f & \mapsto & \{(f,f)\} \\ \Delta_{2b} & i & \mapsto & \{(i,i)\} \\ & f & \mapsto & \{(i,i)\} \end{array}$$

und $\Omega_2 = \{(\{i,f\},\{f\})\}$.

Die Idee ist, daß i bedeutet, ein a wurde gelesen, f bedeutet ein b wurde gelesen. Ω_2 enthält nur ein Element, nämlich $(\{i,f\},\{f\})$, das heißt damit ein Baum akzeptiert wird müssen unendlich oft Zustände aus $\{i,f\}$ erreicht werden (was keine Einschränkung ist, da i und f die einzigen Zustände des Automaten sind) und der Zustand f (zweite Komponente des Tupels) darf nicht unendlich oft erreicht werden.

Gibt es einen Pfad mit unendlich vielen a's, so hat der zugehörige Lauf einen Pfad mit unendlich vielen f's, ist also nicht erfolgreich. □

Satz 7.7

Die Sprache $L = \{t \in T_\Sigma^\omega \mid \text{jeder Pfad in } t \text{ enthält nur endlich viele } a\text{'s}\}$ über dem Alphabet $\Sigma = \{a,b\}$ ist Rabin erkennbar, aber nicht Büchi-erkennbar.

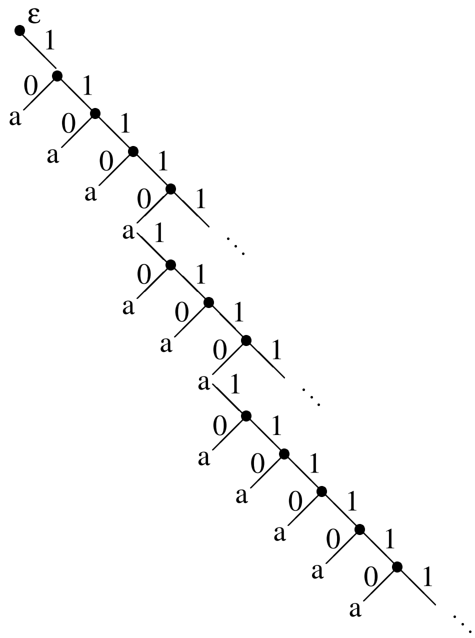
Beweis

Rabin-erkennbar wurde bereits in Beispiel 7.6 gezeigt.

Angenommen $\mathcal{A} = (Q,\Sigma,I,\Delta,F)$ ist ein Büchi-Baumautomat für L. Es sei n so, daß $n > |Q|$. Der Baum $t^{(n)}$ sei wie folgt definiert:

$$\begin{array}{lll} U & = & \{\varepsilon\} \cup \\ & & \{1^{m_1}0 \mid m_1 > 0\} \cup \\ & & \{1^{m_1}01^{m_2} \mid m_1,m_2 > 0\} \cup \\ & & \vdots \\ & & \{1^{m_1}01^{m_2}0\dots1^{m_n}0 \mid m_1,\dots,m_n > 0\} \end{array}$$

$$\begin{array}{llll} t^{(n)}: & \{0,1\}^n & \to & \Sigma \\ & u & \mapsto & \begin{cases} a & u \in U \\ b & u \notin U \end{cases} \end{array}$$

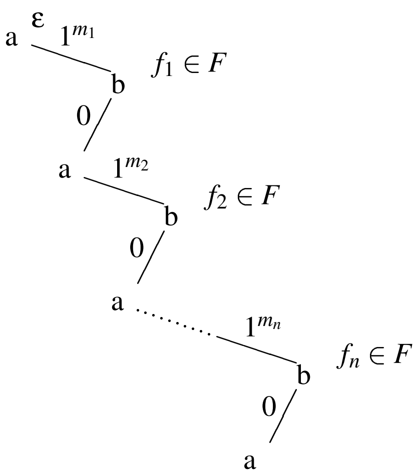


Dieser Baum enthält unendlich viele a's, aber jeder Pfad enthält nur endlich viele a's: um ein a zu erreichen, muß man in dem Baum (irgendwann einmal) nach links gehen. Spätestens nach n-maligem nach links gehen hat man das letzte a auf diesem Pfad erreicht. Also gilt $t^{(n)} \in L$.

Wir konstruieren nun einen Pfad in $t^{(n)}$ wie folgt:

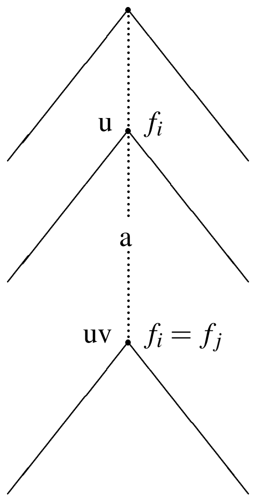
- es sei $m_1 > 0$ minimal mit $l(1^{m_1}) = f_1 \in F$. Ein solches f_1 existiert, da auf dem unendlichen Pfad $\{\varepsilon, 1, 11, 111, \ldots\}$ sogar unendlich viele Endzustände vorkommen.
- es seien $m_1, \ldots, m_i > 0$ für ein $i < n$ bereits definiert. Sei $m_{i+1} > 0$ minimal mit $l(1^{m_1}01^{m_2}0 \ldots 1^{m_i}01^{m_{i+1}}) = f_{i+1} \in F$.

Dies definiert $m_1, \ldots, m_n > 0$ mit $t^{(n)}$:



Da $|Q| < n$ angenommen war, gibt es $i < j$ mit $f_i = f_j$.

Wir haben die folgende Situation:



Setzt man daher in $t^{(n)}$ an der Stelle uv den Baum $t_u^{(n)}$ ein, so erhält man einen Baum, für den es ebenfalls einen erfolgreichen Pfad gibt.
Iteriert man dieses Vorgehen unendlich oft, so erhält man einen Baum, der von $\mathcal{A}$ akzeptiert wird, aber einen Pfad enthält, auf dem a unendlich oft vorkommt (nämlich jeweils zwischen den Positionen $u, uv, uvv, uvvv, \ldots$). □

Korollar 7.8
Die Klasse der Büchi-erkennbaren ω-Baumsprachen ist *nicht* unter Komplement abgeschlossen.
Beweis
$\Sigma = \{a,b\}$. $L_1 = \{t \in T_\Sigma^\omega \mid$ es gibt einen Pfad in t, der unendlich oft a's enthält $\}$, $L_2 = \{t \in T_\Sigma^\omega$ $\mid$ jeder Pfad in t enthält nur endlich viele a's $\}$. L_1 ist Büchi-erkennbar (vergleiche Beispiel 7.5), $L_2 = T_\Sigma^\omega \setminus L_1$ ist nicht Büchi-erkennbar (vergleiche Satz 7.7). □

Satz 7.9 (Rabin)
Die Klasse der Rabin-erkennbaren ω-Baumsprachen ist unter Komplement abgeschlossen. □

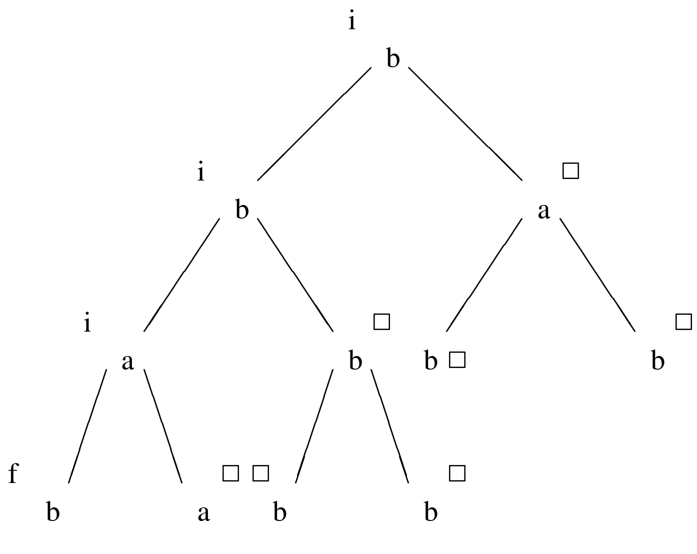
Der Beweis dieses Satzes ist sehr aufwendig. Es gibt verschiedene Beweisansätze (siehe zum Beispiel Thomas: Handbook of Theoretical Computer Science für Literaturhinweise.)
Spieltheoretische Ansätze haben sich auch hier als sehr hilfreich erwiesen. Wieso ist dieses Problem schwieriger als bei Wörtern? Dies liegt an der Quantifizierung über Pfade in der Akzeptanzbedingung: *Für alle Pfade ist die Akzeptanzbedingung erfüllt.*
Negiert man das, so erhält man "Es gibt einen Pfad, für den die Akzeptanzbedingung nicht erfüllt ist."
Man muß nicht nur (wie bei Wörtern) von "nicht erfüllt" wieder zu "erfüllt" kommen, sondern auch von "existiert Pfad" wieder zu "alle Pfade".

Satz 7.10
Das Leerheitsproblem ist für Büchi-erkennbare ω-Baumsprachen entscheidbar.

Den Beweis zu diesem Satz werden wir später führen.
Es sei $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein Büchi-Baumautomat, wobei $F = \{f_1, \ldots, f_n\}$. Es sei $l : \{0,1\}^* \to Q$ ein erfolgreicher Lauf von $\mathcal{A}$ auf dem Baum $t \in T_\Sigma^\omega$.
Der Baum t soll zerlegt werden in endliche Teilbäume, welche von Automaten auf endlichen Bäumen akzeptiert werden. Es sei $u \in \{0,1\}^*$ beliebig. Wir interessieren uns dafür, wo in l nach u das erste Mal wieder Endzustände auftreten.

$$D_u := \{w \in \{0,1\}^* \mid l(uv) \notin F \ \textit{für alle}\ \varepsilon < v \leq w\}$$

Offenbar stellt D_u einen endlich verzweigten Baum dar, in dem es keinen unendlichen Pfad gibt. Dies gilt, da l erfolgreich ist, und daher, um unendlich oft einen Endzustand zu erreichen, auch nach der Stelle u noch Endzustände unendlich oft erreicht werden müssen. Also ist D_u endlich.
Es sei jetzt $D_u^+ := D_u \cup \{w\sigma \mid \sigma \in \{0,1\}, w \in D_u, w\sigma \notin D_u\}$
Nach der Definition von D_u gilt jetzt für alle $w\sigma \in D_u^+ \backslash D_u : l(w\sigma) \in F$.
Beispiel
Betrachte den Lauf des Büchi-Automaten aus Beispiel 7.5



Zur Erinnerung: f und $\Box$ sind Endzustände. Es gilt:

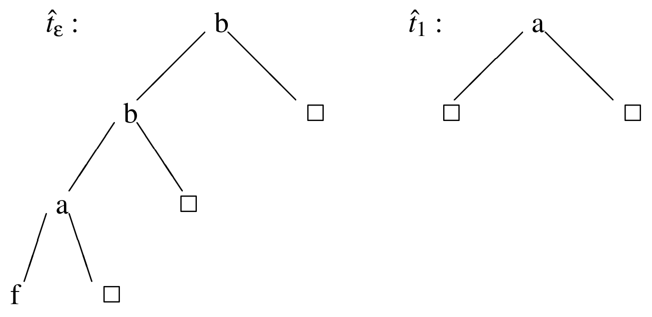
$$\begin{aligned}
D_\varepsilon &= \{\varepsilon, 0, 00\} \\
D_\varepsilon^+ \backslash D_\varepsilon &= \{1, 01, 000, 001\} \\
D_1 &= \{\varepsilon\} \\
D_1^+ \backslash D_1 &= \{00, 1\}
\end{aligned}$$

Für $u \in \{0,1\}^*$ definieren wir den endlichen Baum $\hat{t}_u : D_u^+ \to \Sigma \cup F$ (in dem die Endzustände zusätzlich als Blätter eingeführt werden):

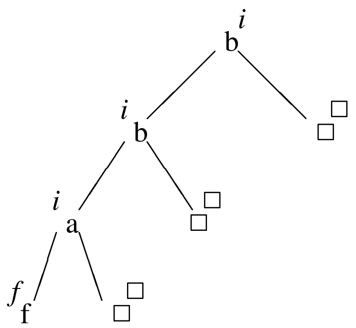
- für $w \in D_u : \hat{t}_u(w) = t(uw)$
- für $w \in D_u^+ \backslash D_u : \hat{t}_u(w) = l(uw)$

$\hat{t}_u \in T_{\Sigma \cup F}$, $f \in F$ ist 0-stellig.

Beispiel



Für jedes $q \in Q$ des Büchi-Automaten $\mathcal{A}$ betrachten wir nun den WB-Automaten $\mathcal{A}_q = (Q, \Sigma \cup F, \{q\}, \Delta, \hat{F})$, wobei Δ wie in $\mathcal{A}$ definiert ist und für $f \in F : \hat{F}(f) = \{f\}$.
Es sei nun $L_q = L(\mathcal{A}_q)$ (dies ist eine endliche Sprache eines endlichen Baumes).
Es gilt: Ist $l(u) = q$, so ist $\hat{t}_u \in L_q$. Das folgende Bild zeigt beispielhaft einen erfolgreichen Lauf von $\mathcal{A}_i$ auf $\hat{t}_\varepsilon$:



Dies zeigt, daß

$$t \in \hat{L} := \left(\bigcup_{i \in I} L_i \right) \cdot^{(f_1,\ldots,f_m)} (L_{f_1}, \ldots, L_{f_m})^{\omega,(f_1,\ldots,f_m)}$$

Umgekehrt ist jedes Element von $\hat{L}$ auch in $L_\omega(\mathcal{A})$, das heißt

$$\hat{L} = L_\omega(\mathcal{A}).$$

Satz 7.11
Für eine Sprache $L \subseteq T_\Sigma^\omega$ sind äquivalent

1. L ist Büchi-erkennbar
2. Es gibt erkennbare Sprachen $L_0, L_1, \ldots L_m \subseteq T_{\Sigma \cup F}$ für ein Alphabet $F = \{f_1, \ldots, f_m\}$ von 0-stelligen Symbolen, so daß gilt

$$L = L_0 \cdot^{(f_1,\ldots,f_m)} (L_1, \ldots, L_m)^{\omega,(f_1,\ldots,f_m)}$$

Beweis

$1 \Rightarrow 2$ wurde bereits gezeigt

$2 \Rightarrow 1$ Konstruiere aus WB-Automaten für $L_0, \ldots, L_m$ einen Büchi-Automaten für L.

□

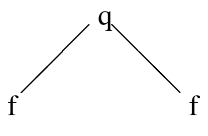
Beweis von Satz 7.10
Es sei $\mathcal{A} = (Q, \Sigma, I, \Delta, F)$ ein Büchi-Automat für L. Die Sprachen L_q seien wie oben definiert: $L_q = L(\mathcal{A}_q)$.
Wir eliminieren jetzt sukzessive Zustände, die nicht in einem erfolgreichen Lauf auftreten können.

1. Eliminationsschritt
   Der Automat $\mathcal{A}_1 = (Q_1, \Sigma, I_1, \Delta_1, F_1)$ entsteht aus $\mathcal{A}$ durch Entfernen aller Zustände $q \in Q$ mit $L_q = \emptyset$. Da L_q erkennbare Baumsprache ist, kann man $L_q = \emptyset$? nach Satz 6.22 entscheiden:

   - $Q_1 = Q \setminus \{q \in Q \mid L_q = \emptyset\}$
   - $\Delta_{1a} = Q_1 \to 2^{Q_1 \times Q_1} : q \mapsto \Delta_a(q) \cap Q_1 \times Q_1$
   - $F_1 = F \cap Q_1$

   Behauptung: $L_\omega(\mathcal{A}_1) = L_\omega(\mathcal{A})$
   denn Ist $L_q = \emptyset$, so kann q *nicht* in einem erfolgreichen Lauf l vorkommen. Wir hatten nämlich gesehen, daß dann für $l(u) = q$ gilt: $(\hat{t})_u \in L_q$.

   Dadurch kann sich die Sprache L_q ändern:



In $\mathcal{A} : L_q \neq \emptyset$, aber $L_f = \emptyset$. In $\mathcal{A}_1$ kann $L_q = \emptyset$ sein!

2. Iteriert man dieses Vorgehen, so erhält man nach spätestens $|Q|$ Schritten einen Automaten $\mathcal{A}_n$ mit $L_\omega(\mathcal{A}_n) = L_\omega(\mathcal{A})$ und $L_q \neq \emptyset$ in $\mathcal{A}_n$ für alle $q \in Q_n$ (beachte: $Q_n = \emptyset$ ist möglich).

3. $L_\omega(\mathcal{A}_n) \neq \emptyset$ gdw $I_n \neq \emptyset$
   Wir wissen:

$$L_\omega(\mathcal{A}_n) = \left(\bigcup_{i \in I_n} L_i \right) \cdot^{(f_1,\ldots,f_m)} (L_{f_1}, \ldots, L_{f_m})^{\omega,(f_1,\ldots,f_m)}$$

   wobei $F_n = \{f_1, \ldots, f_m\}$.

   Der Ausdruck auf der rechten Seite ist offensichtlich leer, wenn $I_n = \emptyset$. Ist umgekehrt $I_n \neq \emptyset$, so folgt $F_n \neq \emptyset$ (da sonst $L_q = \emptyset$ für alle $q \in Q_n$).

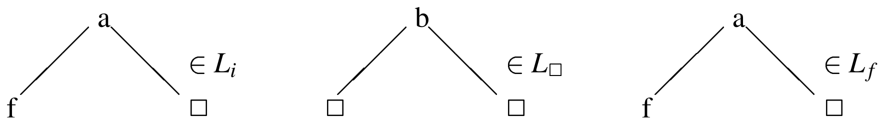
   Damit enthält der Ausdruck auf der rechten Seite einen unendlichen Baum:

   Sei $t_0 \in L_i$ (für ein $i \in I_n$), $t_1 \in L_{f_1}$, …, $t_m \in L_{f_m}$. Dann ist

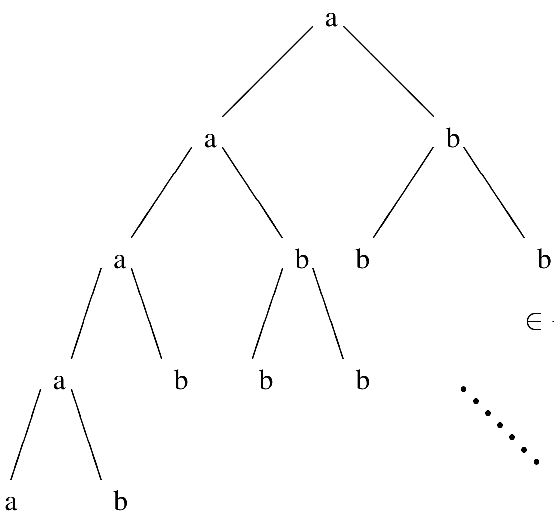
$$\{t_0\} \cdot^{(f_1,\ldots,f_m)} (\{t_1\}, \ldots, \{t_m\})^{\omega,(f_1,\ldots,f_m)} \in L_\omega(\mathcal{A}_n)$$

□

Beispiel
Betrachte den Automaten aus Beispiel 7.5.



Der unendliche Baum, den wir erhalten, sieht nun folgendermaßen aus:



$$\in \{a(f,\square)\} \cdot^{(f,\square)} (\{a(f,\square)\}, \{b(\square,\square)\})^{\omega,(f,\square)}$$

Satz 7.12
Das Leerheitsproblem für Rabin-erkennbare ω-Baumsprachen ist entscheidbar.

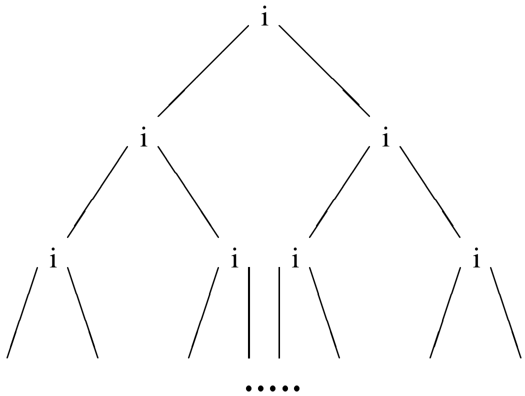
□

Beweisidee
Es sei $\mathcal{A} = (Q, \Sigma, I, \Delta, \Omega)$ ein Rabin-Baumautomat. Ein Zustand $q \in Q$ heißt passiv, falls für alle $a \in \Sigma$ und $(q_1, q_2) \in \Delta_a(q)$ gilt: $q_1 = q = q_2$ (anschaulich: ist man einmal in diesem Zustand, bleibt man auch darin). $q \in Q$ heißt aktiv, falls gilt:

1. q ist nicht passiv und
2. q ist nicht Anfangszustand, der nicht mehr erreichbar ist (das heißt $(q_1, q_2) \in \Delta_a(q)$ $\Rightarrow q \notin \{q_1, q_2\}$).

Entscheidbarkeit des Leerheitsproblems wird jetzt durch Induktion über die Anzahl der aktiven Zustände gezeigt.
Induktionsanfang: keine aktiven Zustände
Ein erfolgreicher Lauf, der mit Zustand $i \in I$ beginnt, sieht jetzt wie folgt aus:

1. Fall: i ist passiv (i reproduziert sich)

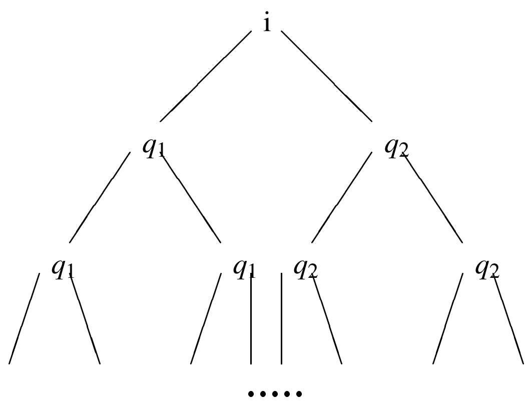


   Damit dieser Lauf erfolgreich ist, muß gelten:

   - $i \in I$
   - es gibt ein $a \in \Sigma$ mit $(i, i) \in \Delta_a(i)$
   - es gibt $(F, G) \in \Omega$ mit $i \in F$ und $i \notin G$

   Man kann feststellen, ob es ein solches i gibt.

2. Fall: i ist Anfangszustand, der nicht mehr erreicht wird



Da es nach Voraussetzung keine aktiven Zustände gibt, sind q_1, q_2 passiv, das heißt sie reproduzieren sich. Man kann jetzt entsprechend zum 1. Fall leicht feststellen, ob es einen erfolgreichen Lauf geben kann.

Induktionsschritt:
Der Induktionsschritt ist recht kompliziert, vergleiche Thomas: Handbook of Theoretical Computer Science.

Kapitel 8

Baumautomaten und logische Formeln

S1S wird durch die Einführung von zwei Nachfolgerfunktionen verallgemeinert. Als Interpretation betrachtet man statt $\mathbb{N}$ den unendlichen binären Baum.
Definition 8.1

1. Formeln von S2S sind wie S1S-Formeln aufgebaut. Der Unterschied besteht darin, daß statt der Nachfolgerfunktion s nun zwei Funktionen s_0 und s_1 und statt $\underline{0}$ die Konstante ε verwendet wird.

2. Als Interpretationsbereich verwenden wir $\{0,1\}^*$ (Träger des unendlichen binären Baums), wobei

   - $\underline{\varepsilon}$ als ε interpretiert wird
   - $s_0 : \; u \mapsto u0$
     $s_1 : \; u \mapsto u1$
   - $<$ als Präfixordnung auf $\{0,1\}^*$ interpretiert wird, das heißt $u < v$ gdw $\exists w \in \{0,1\}^+$ mit $uw = v$
   - $P_1, \ldots, P_n$ als Teilmengen von $\{0,1\}^*$ interpretiert werden

3. WS2S entsteht aus S2S dadurch, daß Quantoren zweiter Stufe nur über endlichen Teilmengen von $\{0,1\}^*$ quantifizieren dürfen.

□

S2S-Formeln können dazu verwendet werden, ω-Baumsprachen zu definieren. Wie bei S1S ist das Alphabet $\Sigma = \{0,1\}^n$, wobei n die Anzahl der einstelligen Prädikatsymbole ist. Jedes Element von Σ erhält die Stelligkeit 2. Eine S2S-Interpretation I kann als ω-Baum $t_I \in T_\Sigma^\omega$ aufgefaßt werden:

$$t_I(u) = (b_1, \ldots, b_n) \text{ mit } b_i = \begin{cases} 1 & n \in P_i^I \\ 0 & n \notin P_i^I \end{cases}$$

Definition 8.2
Sei φ eine geschlossene S2S-Formel.

$$L_\omega(\varphi) := \{t_I \in T_\Sigma^\omega \mid I \models \varphi\}$$

□

Beispiele für S2S-Formeln:

- $Kette(X)$ gdw $\forall x \forall y\ (X(x) \wedge X(y)\ \rightarrow\ x < y\ \vee\ x = y\ \vee\ y < x)$
  beschreibt eine Teilmenge X von $\{0,1\}^*$, in der je zwei Elemente präfixvergleichbar sind
- $Pfad(X)$ gdw $Kette(X)\ \wedge\ \forall Y\ (X \subseteq Y\ \wedge\ Kette(Y)\ \rightarrow\ X = Y)$
  Pfade sind maximale Ketten
- $UnendlicheKette(X)$ gdw $Kette(X)\ \wedge\ \forall x\ (X(x)\ \rightarrow\ \exists y\ (x < y\ \wedge\ X(y)))$
- $Endlich(X)$ gdw $\neg\exists Y\ (Y \subseteq X\ \wedge\ UnendlicheKette(Y))$

Mit Hilfe des Lemmas von König kann man zeigen: *Ist X unendliche Teilmenge von $\{0,1\}^*$, so enthält das X eine unendliche Kette*
Die folgende Formel zeigt, daß man WS2S in S2S ausdrücken kann. Die Umkehrung gilt hier nicht.

$$Z = Präfixabschluß(Y)\ \Leftrightarrow\ \forall z\ (Z(x)\ \leftrightarrow\ \exists y\ (Y(y)\ \wedge\ z \leq y))$$

Beispiel 8.3
Es sei $n = 1$, das heißt $\Sigma = \{0,1\}$.

1. $L_1 = \{t \in T_\Sigma^\omega \mid es\ gibt\ einen\ Pfad\ in\ t,\ der\ unendlich\ viele\ 1en\ enthält\}$
   $L_1 = L_\omega(\varphi_1)$, wobei

$$\begin{aligned}\varphi_1 = \quad \exists Y \quad & (Pfad(Y)\ \wedge \\ & \forall x\ (Y(x)\ \rightarrow\ \exists y\ (Y(y)\ \wedge\ x < y\ \wedge\ P_1(y))))\end{aligned}$$

2. $L_2 = T_\Sigma^\omega \backslash L_1$
   $L_2 = L_\omega(\neg\varphi_1)$

□

Satz 8.4
Für eine ω-Baumsprache $L \subseteq T_\Sigma^\omega$ sind äquivalent:

1. L ist Rabin-erkennbar
2. $L = L_\omega(\varphi)$ für eine geschlossene S2S-Formel φ

□

Beweis (sehr ähnlich zum Beweis von Satz 5.4)

"$1 \Rightarrow 2$" Sei $\mathcal{A} = (Q, \Sigma, I, \Delta, \Omega)$ ein Rabin-Automat mit $Q = \{q_1, \ldots, q_m\}$. Für jedes q_i führen wir eine Variable zweiter Ordnung Y_i ein. Die Idee dabei ist, daß $Y_i(x)$ gilt genau dann, wenn q_i an Position x im Lauf steht. Die Existenz eines erfolgreichen Laufs wird dann durch folgende Formel beschrieben:

$$
\begin{array}{ll}
\multicolumn{2}{l}{\exists Y_1 \ldots \exists Y_m :} \\
\multicolumn{2}{l}{(\neg(Y_i(x) \ \wedge \ Y_j(x)) \ \wedge} \\
\multicolumn{2}{l}{\bigvee_{q_i \in I} \ Y_i(\varepsilon) \ \wedge} \\
\forall x \ (\bigvee_{\substack{a \in \Sigma \\ (q_i, q_j) \in \Delta_a(q_k)}} & Y_k(x) \ \wedge \\
& Q_a(x) \ \wedge \\
& Y_i(s_0(x)) \ \wedge \\
& Y_j(s_1(x))) \ \wedge \\
\forall Z \ (Pfad(Z) \ \rightarrow \ \bigvee_{(F,G) \in \Omega} & (\bigvee_{q_i \in F} \forall x \ (Z(x) \ \rightarrow \\
& \quad \exists y \ (x < y \ \wedge \ Z(y) \ \wedge \ Y_i(y))) \ \wedge \\
& \bigwedge_{q_i \in G} \exists x \forall y \ (x < y \ \wedge \ Z(y) \ \rightarrow \\
& \quad \neg Y_i(y)))))
\end{array}
$$

"$2 \Rightarrow 1$" Wie für S1S reduziert man S2S-Formeln auf $S2S_0$-Formeln. Man verwendet Induktion über den Aufbau von $S2S_0$-Formeln. Wichtig ist hier wieder der Abschluß der Rabin-erkennbaren Sprachen unter Booleschen Operationen und Alphabetumbenennung.

□

Korollar 8.5
Gültigkeit in S2S ist entscheidbar.

□

Es gibt einen interessanten Zusammenhang zwischen Büchi-erkennbaren ω-Baumsprachen und WS2S.

Satz 8.6

1. $L \subseteq T_\Sigma^\omega$ ist Büchi-erkennbar genau dann, wenn $L = L_\omega(\varphi)$ für eine S2S-Formel φ der Gestalt $\exists Y_1 \ldots \exists Y_m \ \psi(Y_1, \ldots, Y_m)$, wobei ψ eine WS2S-Formel ist.
2. $L \subseteq T_\Sigma^\omega$ kann durch eine WS2S-Formel definiert werden genau dann, wenn L und $\overline{L}$ Büchi-erkennbar sind.

□

Insbesondere ist also WS2S sogar noch ausdrucksschwächer als Büchi-Automaten, zum Beispiel kann L_1 aus Beispiel 8.3 nicht durch WS2S-Formeln definiert werden, da $\overline{L_1}$ nicht Büchi-erkennbar ist.

Anstelle binärer Bäume kann man auch Bäume vom Verzweigungsgrad k betrachten. Es gelten die entsprechenden Resultate wie für $k = 2$.

Man kann Büchi-Automaten für Bäume von Verzweigungsgrad k verwenden, um Entscheidbarkeit der Logik DPDL zu zeigen (für k atomare Programme $p_0, \ldots, p_{k-1}$).

Definition 8.7
Die Formeln der Logik DPDL sind wie die von 1DPDL definiert, mit dem Unterschied, daß man nun k atomare Programme $p_0, \dots, p_{k-1}$ zum Aufbau von Programmformeln verwenden kann.
Eine Interpretation weist jedem p_i eine funktionale Relation $p_i^I \subseteq \Delta^I \times \Delta^I$ zu.

□

Wie im Fall $k = 1$ kann man Programmformeln als reguläre Sprachen (über $\Sigma = \{p_0, \dots, p_{k-1}\}$) beschreiben und damit wieder Programmformeln durch Zustände eines geeigneten Automaten $\mathcal{B}$ ersetzen. Der Fisher-Ladner-Abschluß einer DPDL-Formel φ_0 in Negations-Normalform (NNF) wird entsprechend zum Fall $k = 1$ definiert, zum Beispiel gilt $< q > \varphi \in FL(\varphi_0) \ \rightarrow \ \varphi \in FL(\varphi_0) \ \wedge \ < q' > \varphi \in FL(\varphi_0)$ für alle q' mit $(q, p_i, q') \in \Delta$ für ein i, $0 \leq i < k$.
Die Rolle des Hintikka-Worts im Fall $k = 1$ wird von dem Hintikka-Baum übernommen. Es sei $\Sigma = 2^{FL(\varphi_0)}$.

Definition 8.8
Ein Baum $t: \ \{0, 1, \dots, k-1\}^* \rightarrow \Sigma$ ist ein Hintikka-Baum für φ_0, falls gilt[1]

1. $\hat{\varphi}_0 \in t(\varepsilon)$

und für alle $u \in \{0, 1, \dots, k-1\}^*$ gilt

2. $t(u) = \emptyset$ oder ($A_i \in t(u)$ gdw $\neg A_i \notin t(u)$ für alle atomaren Eigenschaften A_i)
3. $\varphi \wedge \psi \in t(u)$ gdw $\varphi \in t(u)$ und $\psi \in t(u)$
4. $\varphi \vee \psi \in t(u)$ gdw $\varphi \in t(u)$ oder $\psi \in t(u)$
5. $[q]\varphi \in t(u)$, dann gilt
   5.1 $\varepsilon \in L(\mathcal{B}_q)$, so $\varphi \in t(u)$
   5.2 für alle i, $0 \leq i < k$ gilt
   $t(ui) = \emptyset$ oder für alle q' mit $(q, p_i, q') \in \Delta : [q']\varphi \in t(ui)$
6. $< q > \varphi \in t(u)$, dann gibt es ein Wort $w = p_{i_1} \dots p_{i_r}$ und Zustände $q_1, \dots, q_r$ von $\mathcal{B}$ mit
   - q_r ist Endzustand, das heißt $w \in L(\mathcal{B}_q)$
   - $q \xrightarrow{p_{i_1}} q_1 \xrightarrow{p_{i_2}} q_2 \dots \xrightarrow{p_{i_r}} q_r$
   - $< q_i > \varphi \in t(ui_1 \dots i_j)$ $1 \leq j < r$
   - $\varphi \in t(ui_1 \dots i_r)$
7. $t(u) = \emptyset$, so auch $t(ui) = \emptyset$ für alle i $0 \leq i < k$

□

[1] Die Definition eines Hintikka-Wortes sollte an diese Definition angepasst werden

Satz 8.9
φ_0 ist erfüllbar genau dann, wenn es einen Hintikka-Baum für φ_0 gibt.

□

Man betrachtet zunächst wieder einen lokalen Büchi-Baumautomaten $\mathcal{A}_L(\varphi_0)$, der statt der globalen Bedingung 6 aus Definition 8.8 die lokale Bedingung 6' überprüft.

6'. $<q>\varphi \in t(u)$, dann gilt

6'.1 $\varepsilon \in L(\mathcal{B}_q)$ und $\varphi \in t(u)$
oder
6'.2 es gibt ein i, $0 \leq i < k$ und q' mit $(q, p_i, q') \in \Delta$ und $<q'>\varphi \in t(ui)$

Definition 8.10
$\mathcal{A}_L(\varphi_0) := (Q_L, \Sigma, I_L, \Delta_L, F_L)$ mit

- $Q_L = \{\alpha \in 2^{FL(\varphi_0)} \mid \alpha \text{ erfüllt 2,3,4 von Definition 8.8}\}$
- $\Sigma = 2^{FL(\varphi_0)}$
- $I_L = \{\alpha \in Q_L \mid \hat{\varphi}_0 \in \alpha\}$
- $F_L = Q_L$
- Für alle $\sigma \in \Sigma$ und $t(u) \in Q_L$ gilt
$t(u) \neq \sigma \Rightarrow \Delta_{L,\sigma}(t(u)) = \emptyset$
$t(u) = \sigma \Rightarrow \Delta_{L,\sigma}(t(u)) = \{(t(u0), \ldots, t(u(k-1)))\mid$ 5, 6' und 7 sind erfüllt $\}$

□

Um zu erzwingen, daß $<>$-Formeln schließlich erfüllt werden, erweitert man $\mathcal{A}_L$ zum Hintikka-Automaten (der ein Büchi-Automat ist).

Definition 8.11
$\mathcal{A}_H(\varphi_0) = (Q_H, \Sigma, I_H, \Delta_H, F_H)$ mit

- $Q_H = Q_L \times 2^{FL_{<>}(\varphi_0)}$, wobei $FL_{<>}(\varphi_0 = \{<q>\varphi \mid \; <q>\varphi \in FL(\varphi_0)\}$
- $\Sigma = 2^{FL(\varphi_0)}$
- $I_H = I_L \times \{\emptyset\}$
- $F_H = F_L \times \{\emptyset\} = Q_L \times \{\emptyset\}$
- $\Delta_{H,\sigma}((\alpha, s)) =$
1. Fall $s = \emptyset$
$((\beta_0, t_0), \ldots, (\beta_{k-1}, t_{k-1})) \in \Delta_{H,\sigma}((\alpha, \emptyset))$ gdw
  - $(\beta_0, \ldots, \beta_{k-1}) \in \Delta_{L,\sigma}(\alpha)$ und
  - für jedes $<q>\varphi \in \sigma$ mit $(\varepsilon \notin L(\mathcal{B}_q)$ oder $\varphi \notin \alpha)$ gibt es ein i, $0 \leq i < k$ und ein q' so daß $<q'>\varphi \in t_i \cap \beta_i$ und $(q, p_i, q') \in \Delta$

  2. Fall $s \neq \emptyset$
  entsprechend, wobei (wie bei $k = 1$) "$<q>\varphi \in \sigma$" ersetzt wird durch "$<q>\varphi \in s$"

□

Satz 8.12
$\mathcal{A}_H(\varphi_0)$ akzeptiert genau die Hintikka-Bäume für φ_0.

$\square$

Satz 8.13
Erfüllbarkeit in DPDL ist entscheidbar.

$\square$

Beachte: Entscheidbarkeit in exponentieller Zeit kann gezeigt werden, da das Leerheitsproblem für Büchi-erkennbare Sprachen in polynomieller Zeit entschieden werden kann. Diese Entscheidbarkeit in polynomieller Zeit folgt aber nicht aus dem von uns betrachteten Verfahren.

Index