9 Vorlesung vom 04. Mai 2000 Abschlußeigenschaften regulärer Mengen

9.1 Motivation

Viele Operationen auf regulären Mengen erhalten die Eigenschaft der Regularität. Dies bedeutet, daß die Operationen angewandt auf den regulären Mengen wieder zu einer regulären Menge führt.

Mit Hilfe dieser als *Abgeschlossenheit* bezeichneten Eigenschaften kann man also aus gegebenen regulären Mengen wieder neue Mengen schaffen, die von endlichen Automaten akzeptiert werden.

Man kann die Abschlußeigenschaften auch dazu verwenden, um zu zeigen, dass eine bestimmte gegebene Sprache regulär (bzw. nicht regulär) ist, indem man über die Abschlußeigenschaften auf bereits bekannte Sprachen zurückgeht.

9.2 Sammlung von Abschlußeigenschaften

Es gelten die folgenden Abschlußeigenschaften regulärer Mengen:

(9.1)	Vereinigung:	$L_1, L_2 \in REG$	$\Rightarrow L_1 \cup L_2 \in REG$
(9.2)	Konkatenation:	$L_1, L_2 \in REG$	$\Rightarrow L_1 \circ L_2 \in REG$ $L_1 \circ L_2 = \{xy \mid x \in L_1, y \in L_2\}$
(9.3)	Kleenesche Hülle:	$L \in REG$	$\Rightarrow L^* \in REG, L^* = \bigcup_{i=0}^{\infty} L^i$
(9.4)	Komplement:	$L \subseteq \Sigma^*, L \in REG$	$\Rightarrow \Sigma^* - L \in REG$
(9.5)	Schnitt:	$L_1, L_2 \in REG$	$\Rightarrow L_1 \cap L_2 \in REG$
(9.6)	Homomorphishmus:	$L_1 \subseteq \Sigma_1^*, L_1 \in REG, h: \Sigma_1^* \to \Sigma_2^*$	$\Rightarrow h(L_1) \in REG$
(9.7)	inv. Homomorphism.:	$L_1 \subseteq \Sigma_2^*, L_2 \in REG, h: \Sigma_1^* \to \Sigma_2^*$	$\Rightarrow h^{-1}(L_2) \in REG$ $h^{-1}(L_2) = \{x \mid x \in \Sigma_1^*, h(x) \in L_2\}$

9.2.1 Beweise Vereinigung, Konkatenation, Kleenesche Hülle

Zu (9.1), (9.2) und (9.3) findet man die entsprechenden Beweise im Beweis über die Äquivalenz von regulären Ausdrücken und einem NFA mit ε-Bewegungen.

9.2.2 Komplementbildung (9.4)

Zu Zeigen: $L \subseteq \Sigma^*, L \in REG \quad \Rightarrow \Sigma^* - L \in REG$

Beweis:

Sei $M = (Q, \Sigma, \delta, q_0, F)$ ein DFA mit L=T(M). Dann definieren wir einen DFA M‘ wie folgt:

$$M' := (Q, \Sigma, \delta, q_0, Q - F) \tag{9.8}$$

Es zeigt sich nun folgendes:

Fehler! Es ist nicht möglich, durch die Bearbeitung von Feldfunktionen Objekte zu erstellen. (9.9)

Also: $T(M') = \Sigma^* - T(M) = \Sigma^* - L$

9.2.3 Schnitt (9.5)

Zu zeigen: $L_1, L_2 \in REG \qquad \Rightarrow L_1 \cap L_2 \in REG$

Beweis:

Dies zeigt man sehr schnell über die Regeln von deMorgan. Es gilt nämlich $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ und mit der Abgeschlossenheit unter Komplement- und Vereinigungsbildung folgt automatisch die Abgeschlossenheit unter Schnittbildung.

9.2.4 Substitution

Definition:

Eine **Substitution f** ist eine Abbildung $f : \Sigma \rightarrow L, L \subseteq \Delta^*$ mit Δ ein Alphabet . F assoziiert also mit jedem Symbol aus Σ eine Sprache. Wir erweitern f auf Zeichenketten:

1. $f(\varepsilon) = \varepsilon$
2. $f(xa) = f(x) f(a)$

Beispiel:

Ist $f(0) = a$ und $f(1) = b^*$. Damit ist f(010) die reguläre Menge ab^*a .

Satz:

Die Klasse der regulären Mengen ist unter Substitution abgeschlossen.

Beweis:

Sei $R \subseteq \Sigma^*$ eine reguläre Menge und $\hat{\delta}(q, a)$. Sei weiter $f : \Sigma \rightarrow 2^{\Delta^*}$ die Substitution, die durch $f(a)=R_a$ definiert ist.

Wir wählen reguläre Ausdrücke zur Bezeichnung von R und allen R_a.

Man ersetze nun jedes Auftreten des Symbols **a** im regulären Ausdruck für R durch den regulären Ausdruck für R_a.

Um zu beweisen, dass der resultierende reguläre Ausdruck $f(R)$ beschreibt, wird verwendet, dass die Substitution einer Vereinigung, eines Produkts oder einer Hülle gleich der Vereinigung, dem Produkt bzw. der Hülle der Substitution ist (also homomorph).

Mit einer Induktion über die Anzahl der Operatoren im regulären Ausdruck vervollständigt man leicht den Beweis.

9.2.5 Homomorphismus (9.6)

Zu zeigen: $L_1 \subseteq \Sigma_1^*, L_1 \in REG, h : \Sigma_1^* \rightarrow \Sigma_2^* \qquad \Rightarrow h(L_1) \in REG$

Beweis:

Die Abgeschlossenheit unter Homomorphismen folgt sofort aus der Abgeschlossenheit unter Substitutionen, da ein Homomorphismus ein Spezialfall der Substitution ist, bei der h(a) für alle a gegeben wird.

9.2.6 Inverser Homomorphismus (9.7)

Zu zeigen: $L_1 \subseteq \Sigma_2^*, L_2 \in REG, h : \Sigma_1^* \rightarrow \Sigma_2^* \qquad \Rightarrow h^{-1}(L_2) \in REG$

Beweis:

Es seien $M_2 = (Q, \Sigma, \delta_2, q_0, F)$ ein DFA mit $\mathrm{T}(M_2) = L_2$ und $h : \Sigma_1^* \rightarrow \Sigma_2^*$ ein Homomorphismus.

Wir konstruieren einen DFA M_1, der $h^{-1}(L)$ akzeptiert, indem er ein Symbol a aus Σ_1 liest und M_1 auf h(a) simuliert. Wir geben die Definition an:

$$\begin{aligned} &M_1 := (Q, \Sigma_1, \delta_1, q_0, F) \\ &\delta_1(q,a) := \delta_2(q, h(a)) \forall q \in Q, a \in \Sigma_1 \end{aligned} \tag{9.10}$$

Zu beachten ist, dass h(a) eine lange Zeichenkette oder aber auch das leere Wort sein kann. Dies ist aber kein Problem, da δ mittels Erweiterung auch auch Zeichenketten operiert.

Durch Induktion über die Wortlänge des Wortes x kann man leicht zeigen, dass gilt

$$\delta_1(q_0, x) = \delta_2(q_0, h(x)) \tag{9.11}$$

Daher wird x von M_1 genau dann akzeptiert, wenn h(x) von M_1 akzeptiert wird. Wir haben also einen DFA gebaut mit $T(M_1) = h^{-1}(T(M_2))$. Damit gilt auch $h^{-1}(L_2) \in REG$.

10 Vorlesung vom 9. Mai 2000

10.1 Beispiele zur Nutzung der Abschlusseigenschaften

10.1.1 Einleitung

Aus den vorhergehenden Vorlesungseinheiten sind die sogenannten „Abschlusseigenschaften“ bekannt:

$$\begin{aligned} &L_1, L_2 \in \mathrm{REG} \Rightarrow L_1 \circ L_2 \in \mathrm{REG} \quad && \circ \in \{\cap, \cup, \cdot\} \\ &L_1 \in \mathrm{REG} \Rightarrow \circ(L_1) \in \mathrm{REG} && \circ \in \{-, h, h^{-1}, *\} \end{aligned} \tag{10.1}$$

Weiterhin nehmen wir zur Kenntnis[4]:

$$L = \{a^n b^n \mid n \geq 0\} \notin \mathrm{REG} \tag{10.2}$$

In den folgenden Beispielen wird durch die Kombination des Wissens von (10.2) mit den Aussagen der Abschlusseigenschaften der Beweis für einen neuen Zusammenhang erbracht.

Behauptung:

Die folgende Aussage ist falsch: „Wenn $a^n b^n$ nicht regulär ist, kann auch eine „größere“ Sprache wie $a^* b^*$ nicht regulär sein.“ Tatsächlich gilt, dass $a^n b^n$ nicht regulär ist, und $a^* b^*$ regulär ist.

10.1.2 Beispiel 1

Zu zeigen sei:

$$L = \{a^n b^r c^n d^s \mid n \geq 0, r \geq 0, s \geq 0\} \notin \mathrm{REG} \tag{10.3}$$

Wir führen einen Widerspruchsbeweis durch, und nehmen an:

$$L \in \mathrm{REG} \tag{10.4}$$

Wir definieren eine homomorphe Funktion *h()* wie folgt:

$$h(a) := a \quad h(b) := \varepsilon \quad h(c) := b \quad h(d) := \varepsilon \tag{10.5}$$

(10.3) lässt sich mit Hilfe von (10.5) umschreiben zu:

$$h(L) = \{a^n b^n \mid n \geq 0\} \tag{10.6}$$

Gemäß (10.1) gilt:

$$L \in \mathrm{REG} \Rightarrow h(L) \in \mathrm{REG} \tag{10.7}$$

Aus (10.6) und (10.7) folgt:

$$h(L) = \{a^n b^n \mid n \geq 0\} \in \mathrm{REG} \qquad \text{Widerspruch!} \tag{10.8}$$

10.1.3 Beispiel 2

Zu zeigen sei:

$$L = \{w \mid w \in \{a,b\}^* \wedge \#_a(w) = \#_b(w)\} \notin \mathrm{REG} \tag{10.9}$$

Wir führen einen Widerspruchsbeweis durch, und nehmen an:

$$L \in \mathrm{REG} \tag{10.10}$$

[4] (10.2) ist zwar beweisbar, an dieser Stelle verzichten wir aber darauf.

Wir definieren eine zweite Sprache L_1 wie folgt:

$$L_1 := \underbrace{L}_{\substack{\text{angeblich}\\\text{reguläre}\\\text{Sprache}}} \cap \underbrace{a^*b^*}_{\substack{\text{reguläre}\\\text{Sprache}}} \tag{10.11}$$

Aus (10.1) folgt, dass die Schnittmenge zweier regulärer Mengen erneut regulär ist, daher folgt:

$$L_1 \in \text{REG} \tag{10.12}$$

(10.11) lässt sich umschreiben zu:

$$L_1 := L \cap a^*b^* = \left\{ a^n b^n \,\middle|\, n \geq 0 \right\} \tag{10.13}$$

Aus (10.13) und (10.10) folgt:

$$L_1 := L \cap a^*b^* = \left\{ a^n b^n \,\middle|\, n \geq 0 \right\} \in \text{REG} \qquad \text{Widerspruch!} \tag{10.14}$$

10.1.4 Beispiel 3

Zu zeigen sei:

$$L = \left\{ a^n b^{k-n} \,\middle|\, n \geq 0 \right\} \notin \text{REG} \tag{10.15}$$

Wir führen einen Widerspruchsbeweis durch, und nehmen an:

$$L \in \text{REG} \tag{10.16}$$

Wir definieren eine homomorphe Funktion *h()* wie folgt:

$$h(a) := a \quad h(b) := b^n \tag{10.17}$$

(10.15) lässt sich mit Hilfe von (10.17) umschreiben zu:

$$h^{-1}(L) = \left\{ a^n b^n \,\middle|\, n \geq 0 \right\} \tag{10.18}$$

Gemäß (10.1) gilt:

$$L \in \text{REG} \Rightarrow h^{-1}(L) \in \text{REG} \tag{10.19}$$

Aus (10.18) und (10.19) folgt:

$$h^{-1}(L) = \left\{ a^n b^n \,\middle|\, n \geq 0 \right\} \in \text{REG} \qquad \text{Widerspruch!} \tag{10.20}$$

10.1.5 Beispiel 4

Zu zeigen sei:

$$L = \left\{ a^n b^m \,\middle|\, 0 \leq m \leq n \right\} \notin \text{REG} \tag{10.21}$$

Wir führen einen Widerspruchsbeweis durch, und nehmen an:

$$L \in \text{REG} \tag{10.22}$$

Wir definieren eine zweite Sprache L_1 wie folgt:

$$L_1 := \underbrace{\overline{L}}_{\substack{\text{angeblich}\\\text{reguläre}\\\text{Sprache}}} \cap \underbrace{a^*b^*}_{\substack{\text{reguläre}\\\text{Sprache}}} = \left\{ a^n b^m \,\middle|\, 0 \leq n < m \right\} \tag{10.23}$$

Aus (10.1) folgt, dass das Komplement einer regulären Menge, und auch die Schnittmenge zweier regulärer Mengen erneut regulär ist, daher folgt:

$$L_1 \in \text{REG} \tag{10.24}$$

Wir definieren eine homomorphe Funktion *h()* in eine weitere, neue Sprache L_2 wie folgt:

$$\begin{aligned} &h(a) := a \quad h(b) := b \quad h(c) := c \\ &L_2 := h^{-1}(L_1) = \left\{ a^n x \middle| |x| = m = \#_b(x) + \#_c(x) \right\} \quad 0 \le n < m \end{aligned} \tag{10.25}$$

Aus (10.1) folgt, dass die umgekehrte Anwendung einer homomorphen Abbildung auf eine reguläre Menge erneut regulär ist:

$$L_2 \in \text{REG} \tag{10.26}$$

Wir definieren nun eine dritte Sprache:

$$L_3 := \underbrace{L_2}_{\substack{\text{reguläre}\\\text{Sprache}}} \cap \underbrace{a^* b^* c}_{\substack{\text{reguläre}\\\text{Sprache}}} = \left\{ a^n b^m c \middle| 0 \le n < m+1 \right\} \tag{10.27}$$

Aus (10.1) folgt, dass die Schnittmenge zweier regulärer Menge erneut regulär ist:

$$L_3 \in \text{REG} \tag{10.28}$$

Wir definieren eine zweite, homomorphe Funktion *h()* in eine weitere, neue Sprache L_4 wie folgt:

$$\begin{aligned} &h(a) := a \quad h(b) := b \quad h(c) := \varepsilon \\ &L_4 := h(L_3) = \left\{ a^n b^m \middle| 0 \le n \le m \right\} \end{aligned} \tag{10.29}$$

Wir definieren nun eine fünfte Sprache:

$$L_5 := \underbrace{L}_{\substack{\text{angeblich}\\\text{reguläre}\\\text{Sprache}}} \cap \underbrace{L_4}_{\substack{\text{reguläre}\\\text{Sprache}}} = \left\{ a^n b^n \middle| n \ge 0 \right\} \tag{10.30}$$

Aus (10.1) folgt, dass die Schnittmenge zweier regulärer Menge erneut regulär ist:

$$L_5 \in \text{REG} \qquad \text{Widerspruch!} \tag{10.31}$$

10.1.6 Beispiel 5

Wir verwenden dieses Mal ein erweitertes Alphabet *Σ*:

$$\Sigma := \{a, b_1, \ldots, b_k\} \qquad k \ge 1 \tag{10.32}$$

Zu zeigen sei:

$$L = \left\{ a_n b_1^{i_1} b_2^{i_2} b_3^{i_3} \ldots b_n^{i_n} \middle| 0 \le i_1 + i_2 + i_3 + \ldots + i_n \le n \right\} \notin \text{REG} \tag{10.33}$$

Wir führen einen Widerspruchsbeweis durch, und nehmen an:

$$L \in \text{REG} \tag{10.34}$$

Wir definieren eine homomorphe Funktion *h()* wie folgt:

$$\begin{aligned} &h(a) := a \quad h(b_j) := b \qquad 1 \le j \le n \\ &\Downarrow \\ &h(L) = \left\{ a^n b^m \middle| 0 \le m \le n \right\} \end{aligned} \tag{10.35}$$

Gemäß (10.1) gilt:

$$L \in \text{REG} \Rightarrow h(L) \in \text{REG} \tag{10.36}$$

Aus (10.35) und (10.36) folgt:

$$h(L) = \left\{ a^n b^m \middle| 0 \le m \le n \right\} \in \text{REG} \qquad \text{Widerspruch!} \tag{10.37}$$

Der Widerspruch entsteht durch die Aussage von (10.21).

10.2 Automaten mit ε-Bewegungen

10.2.1 Einleitung

Bisher sind der deterministische endliche Automat (*DFA*) und der nichtdeterministische endliche Automat (*NFA*) betrachtet worden, wobei der *DFA* lediglich ein Spezialfall des *NFA* war. Diese Automaten haben sich dadurch ausgezeichnet, dass jede Transition (jeder Zustandswechsel) durch ein Eingabesymbol $x \in \Sigma$ ausgelöst wurde.

Bei den im Folgenden betrachteten „Automaten mit ε-Bewegungen" gibt es Transitionen, die durch das Eingabesymbol „ε", d.h. das „leere Wort" ausgelöst werden.

Das Transitionsdiagramm eines derartigen Automaten kann z.B. wie folgt aussehen:

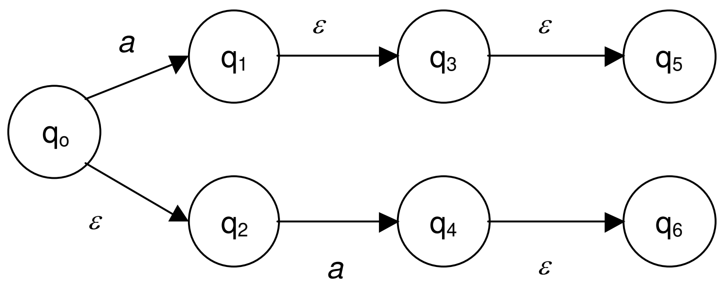

Abbildung 16 – ein Transitionsdiagramm

Bei dem obigen Automaten führt aus dem Startzustand q_0 für das Eingabezeichen a lediglich ein direkter Pfad zu dem Zustand q_1. Allerdings führen zahlreiche weitere Pfade für das Eingabesymbol ε zu den restlichen Zuständen.

Das Eingabesymbol ε unterscheidet sich von allen anderen Eingabesymbolen, und zwar insofern, als dass es der Automat „jederzeit" einlesen kann – unabhängig von den wirklichen Eingabesymbolen.

Der obige Automat kann also für das Eingabezeichen a in jeden, eingezeichneten Zustand wechseln. Um dieses Verhalten beschreibungstechnisch in den Griff zu kriegen, definiert man den Begriff der ε-Hülle (siehe folgender Abschnitt).

10.2.2 ε-Hülle

Man bezeichnet die Menge aller Konten *p*, zu denen es einen mit ε bezeichneten Weg vom Knoten *q* gibt, als ε-Hülle(q)[5].

Für den Automaten aus Abbildung 16 gilt, dass alle Knoten vom Startzustand aus erreichbar sind, also:

$$\varepsilon\text{-Hülle}(q_0) = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\} \tag{10.38}$$

Definitionsgemäß ist jeder Knoten/Zustand in seiner eigenen ε-Hülle enthalten. Man kann sich das so vorstellen, als dass von jedem Knoten ein mit ε markierter Übergang auf sich selber existiert, den man beim Zeichnen weglässt:

$$k = 1 \tag{10.39}$$

[5] Im englischen: „ε-Closure"

Auf das obige Beispiel angewendet ergibt sich die ε-Hülle wie folgt:

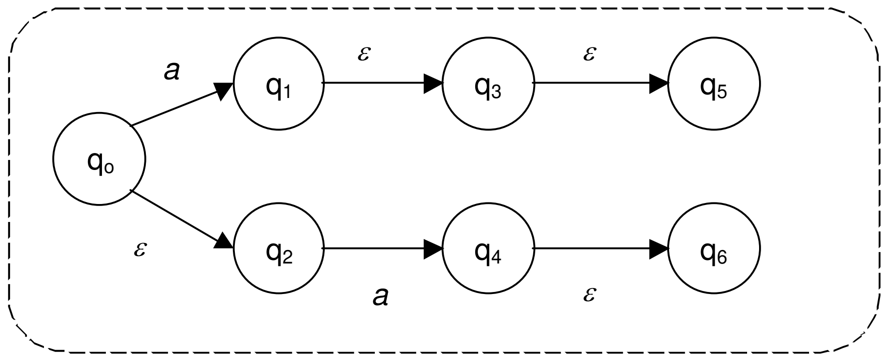

Abbildung 17 – die Epsilon-Hülle im Transitionsdiagramm

10.2.3 Formale Unterschiede zum gewöhnlichen NFA

Die Definition des Automaten mit ε-Bewegungen entspricht der des „normalen" NFA:

$$\varepsilon\text{-NFA} := \left(Q, \Sigma, \delta, q_0, F\right) \tag{10.40}$$

Der einzige Unterschied zum „normalen" NFA ist die Übergangsfunktion δ, die nun wie folgt definiert ist:

$$\delta : Q \times \left(\Sigma \cup \{\varepsilon\}\right) \to 2^Q \tag{10.41}$$

Die Erweiterung der Übergangsfunktion δ (die nur für einzelne Symbole taugt) auf ganze Eingabewörter geschieht mit Hilfe der Übergangsfunktion $\hat{\delta}$. $\hat{\delta}(q,w)$ soll als Ergebnis die Menge aller Zustände liefern, zu denen es von q aus einen mit w markierten Pfad gibt, welcher ε-Kanten beinhalten darf:

$$\hat{\delta} : Q \times \Sigma^* \to 2^Q \tag{10.42}$$

$$\hat{\delta}(q, \varepsilon) := \varepsilon\text{-Hülle}(q) \tag{10.43}$$

$$\forall w \in \Sigma^*, a \in \Sigma : \quad \hat{\delta}(q, wa) := \varepsilon\text{-Hülle}\left(\delta\left(\hat{\delta}(q,w), a\right)\right) \tag{10.44}$$

$$\delta(R, a) = \bigcup_{q \in R} \delta(q, a) \tag{10.45}$$

$$\hat{\delta}(R, w) = \bigcup_{q \in R} \hat{\delta}(q, w) \tag{10.46}$$

Anmerkung: Beim Automaten mit ε-Bewegungen können die Ergebnisse von $\hat{\delta}(q,a)$ und $\delta(q,a)$ verschieden sein; zwischen $\hat{\delta}(\ )$ und $\delta(\ )$ muss also unterschieden werden.

10.2.4 Äquivalenz vom NFA mit und ohne ε-Bewegungen

Auch die ε-Bewegungen erlauben es dem Automaten nicht, mehr als reguläre Mengen zu akzeptieren. Der Automaten mit ε-Bewegungen reiht sich damit in die Klasse von NFA und DFA ein.

Wir werden uns darauf beschränken, zu zeigen, dass jeder NFA mit ε-Bewegungen durch einen „normalen NFA" ersetzt werden kann. Der Beweis in der umgekehrten Richtung ist zwar theoretisch ebenfalls nötig, erübrigt sich aber bei näherer Betrachtung[6].

Zu einem gegebenen ε-NFA $M = (Q, \Sigma, \delta, q_0, F)$ definieren wir einen „normalen" NFA $M' = (Q, \Sigma, \delta', q_0, F')$. Wie man an der Definition sehen kann, muss an dem Automaten nicht viel verändert werden. Die Menge der akzeptierenden Zustände *F* kann weitestgehend übernommen werden, abgesehen von der Ausnahme, dass die ε-Hülle von q_0 einen akzeptierenden Zustand enthält:

$$F' := \begin{cases} F \cup \{q_0\} & \text{falls die } \varepsilon\text{-Hülle}(q_0) \text{ einen Zustand aus F enthält} \\ F & \text{sonst} \end{cases} \tag{10.47}$$

Diese Veränderung ist nötig, weil *M'* keine ε-Transitionen kennt. Tritt nämlich der Fall ein, dass die ε-Hülle von q_0 einen akzeptierenden Zustand enthält, dann würde *M* ja beenden, ohne ein einziges Zeichen gelesen zu haben. Für *M'* lässt sich dies nur nachbilden, indem der Startzustand (q_0) selber akzeptierend ist. Und genau das erreichen wir mit der obigen Fallunterscheidung.

Nun wird noch eine Übergangsfunktion δ' benötigt. Dazu behalten wir die bereits definierte Funktion $\hat{\delta}$ bei:

$$\delta' := \hat{\delta} \tag{10.48}$$

Wie kann das sein? Fangen wir ganz vorne an. Wir möchten zwei äquivalente Automaten erhalten. Wir haben bereits definiert, dass beide Automaten auf der selben Menge von Zuständen operieren. Was ist nun also naheliegender, als auch die Übergangsfunktion komplett zu übernehmen? Genau dies tun wir laut (10.48). Der Sicherheit halber werden wir aber im folgenden beweisen, dass es auch richtig ist, was wir da tun... Wir betrachten zunächst einmal den einfachsten Fall, nämlich, dass die Eingabe ein einzelnes Symbol ist: Dann folgt aus (10.44), (10.43) und (10.45):

$$\begin{aligned}
&\forall w \in \Sigma^*, a \in \Sigma: \quad \hat{\delta}(q, wa) = \varepsilon\text{-Hülle}\left(\delta\left(\hat{\delta}(q, w), a\right)\right) \\
&\Downarrow \qquad w := \varepsilon a \\
&\hat{\delta}(q, \varepsilon a) = \varepsilon\text{-Hülle}\left(\delta\left(\hat{\delta}(q, \varepsilon), a\right)\right) \\
&\Downarrow \qquad \hat{\delta}(q, \varepsilon) := \varepsilon\text{-Hülle}(q) \\
&\hat{\delta}(q, \varepsilon a) = \varepsilon\text{-Hülle}\left(\delta\left(\varepsilon\text{-Hülle}(q), a\right)\right) \\
&\Downarrow \qquad \delta(R, a) = \bigcup_{q \in R} \delta(q, a) \\
&\hat{\delta}(q, \varepsilon a) = \varepsilon\text{-Hülle}\left(\bigcup_{p \in \varepsilon\text{-Hülle}(q)} \delta(p, a)\right)
\end{aligned} \tag{10.49}$$

In klarer Sprache bedeutet das, dass $\hat{\delta}$ angewendet auf ein einzelnes Symbol *a* (und einen Zustand *q*) nichts anderes ist, als die Vereinigung der Ergebnisse von δ für jeden einzelnen Zustand aus der ε-Hülle von *q*. Die ε-Hülle von *q* wiederum ist nichts anderes als eine Menge von Zuständen, zwischen denen der Automat indifferent ist. Genau dieses Verhalten ist auch die Forderung, die wir an die Übergangsfunktion δ' des NFA stellen. Wir haben jetzt für den Sonderfall $|x| = 1$ bewiesen, dass die Forderung, die wir an die „neue" Funktion δ' stellen, durch die „alte" Funktion $\hat{\delta}$ erfüllt werden.

Die Korrektheit für den Sonderfall $|x| = 0$, d.h. $x = \varepsilon$ haben wir bereits bei der Definition der akzeptierenden Zustände unter (10.47) verbal erklärt.

[6] Ein NFA wird ja schon dadurch zum ε-NFA, dass man von jedem Zustand einen mit „ε" markierten Pfeil auf sich selber einzeichnet. Dadurch verändert sich das Verhalten des Automaten nicht.

Für $|x|>1$ erbringen wir den Beweis induktiv. Da $|x|>1$, lässt sich die Eingabe zerlegen in $x = wa \quad w \in \Sigma^*, a \in \Sigma$. Durch die Induktionsvoraussetzung gilt

$$\delta'(q_0, w) = \hat{\delta}(q_0, w) \tag{10.50}$$

Weiterhin gilt:

$$\delta'(q_0, wa) = \delta'\left(\delta'(q_0, w), a\right) \tag{10.51}$$

Aus (10.50) und (10.51) folgt:

$$\delta'(q_0, wa) = \delta'\left(\hat{\delta}(q_0, w), a\right) \tag{10.52}$$

Aus (10.46) folgt unter Zuhilfenahme der Induktionsannahme[7]:

$$\begin{aligned} &\delta'(R,a) = \bigcup_{q \in R} \delta'(q,a) \\ &\Downarrow \quad \delta'(q,a) = \hat{\delta}(q,a) \\ &\delta'(R,a) = \bigcup_{q \in R} \hat{\delta}(q,a) \end{aligned} \tag{10.53}$$

Mit dem Wissen von (10.53) lässt sich (10.52) wie folgt umschreiben:

$$\begin{aligned} &\delta'(q_0, wa) = \delta'\left(\hat{\delta}(q_0, w), a\right) \\ &\Downarrow \quad \delta'(R,a) = \bigcup_{q \in R} \hat{\delta}(q,a) \\ &\delta'(q_0, wa) = \bigcup_{p \in \hat{\delta}(q_0, w)} \hat{\delta}(p,a) \end{aligned} \tag{10.54}$$

Nach der Definition von $\hat{\delta}$ (siehe (10.44) gilt:

$$\bigcup_{p \in R} \hat{\delta}(p,a) = \hat{\delta}(R,a) \tag{10.55}$$

Damit folgt aus (10.54), (10.55) und (10.51):

$$\begin{aligned} &\delta'(q_0, wa) = \bigcup_{p \in \hat{\delta}(q_0, w)} \hat{\delta}(p,a) = \hat{\delta}\left(\hat{\delta}(q_0, w), a\right) = \hat{\delta}(q_0, wa) \\ &\Downarrow \\ &\delta'(q_0, wa) = \hat{\delta}(q_0, wa) \end{aligned} \tag{10.56}$$

[7] Hier wird die Induktionsannahme für Länge(x)=1 verwendet. Da in dem aktuellen Induktionsschritt Wörter der Länge > 1 betrachtet werden, ist dieses Vorgehen erlaubt.

11 Vorlesung vom 11. Mai 2000

11.1 Äquivalenz von endlichen Automaten und regulären Ausdrücken

11.1.1 Von regulären Ausdrücken zu endlichen Automaten

In den vorangegangen Wochen haben wir verschiedene Typen von endlichen Automaten kennengelernt. Dabei haben wir gezeigt, dass deterministische und nichtdeterministische endliche Automaten (DFA und NFA) sowie nichtdeterministische Automaten mit ε-Übergängen die gleiche Klasse von Sprachen, die regulären Sprachen, beschreiben und beliebig ineinander konvertiert werden können.

Jetzt wollen wir zeigen, dass die Sprachen, die durch reguläre Ausdrücke definiert werden, ebenfalls äquivalent zu diesen endlichen Automaten sind. Diese Sprachen werden auch reguläre Mengen genannt (nach dem *Satz von Kleene*: „Die Menge der durch reguläre Ausdrücke beschreibbaren Sprachen ist genau die Menge der regulären Sprachen.“)

Folgendes Schaubild verdeutlicht den Zusammenhang und die Abhängigkeit zwischen regulären Ausdrücken und endlichen Automaten, die wir im Anschluß zeigen wollen.

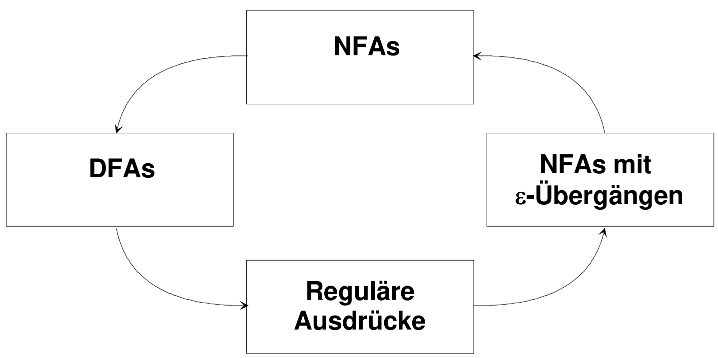

Abbildung 18 - Äquivalenzkreislauf reguläre Ausdrücke - endliche Automaten

Satz:

Sei r ein regulärer Ausdruck, dann existiert ein NFA mit ε-Übergängen, der die zum regulären Ausdruck gehörende Sprache L(r) akzeptiert.

Beweis

Für einfache reguläre Ausdrücke läßt sich die Behauptung einfach zeigen. Wenn r die leere Menge $r = \emptyset$, das leere Wort $r = \varepsilon$ oder ein einzelnes Zeichen $r = x \in \Sigma$ ist, dann liegt ein einfacher regulärer Ausdruck vor.

Zu zeigen ist, dass ein solcher regulärer Ausdruck in einen endlichen Automaten mit ε-Übergängen überführt werden kann, wir konstruieren also einen solchen Automaten gemäß der Vorgabe des regulären Ausdrucks. Diese drei ein- oder keinelementigen Ausdrücke ohne Operator stellen gewissermaßen den Induktionsanfang der Induktion über die Anzahl der Operatoren des regulären Ausdrucks vor.

Wenn wir zeigen können, dass ein beliebiger regulärer Ausdruck mit beliebig vielen Operatoren n in einen NFA mit ε-Übergängen und einem einzigen Endzustand, aus dem keine Übergänge herausführen, überführt werden kann, dann existiert für jeden regulären Ausdruck ein entsprechender Automat und die Äquivalenz ist bewiesen.

Für die drei elementaren Zustände können folgende Automaten gebaut werden, die einen einzigen Endzustand haben, der erreicht wird, wenn ein Wort der Sprache L(r) erkannt wird und von dem keine Übergänge ausgehen.

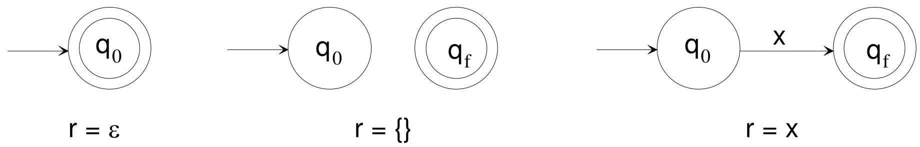

Abbildung 19 - NFAs für reguläre Ausdrücke ohne Operatoren

Für einen regulären Ausdruck, der das leere Wort erkennt, geht der Automat direkt in den Endzustand über, da ε-Übergänge erlaubt sind. Der reguläre Ausdruck leere Menge tritt niemals in den Endzustand ein und der reguläre Ausdruck für ein bestimmtes Zeichen geht genau dann in den Endzustand über, wenn dieses Zeichen erkannt wird.

Für den Induktionsschritt von i - 1 auf i Operatoren lassen sich NFAs mit ε-Übergängen für reguläre Ausdrücke mit weniger als i Operatoren bereits konstruieren ($i \geq 1$). Dabei muß zwischen drei Verknüpfungsarten unterschieden werden

1. Vereinigung:

   $r = r_1 + r_2$

2. Konkatenation:

   $r = r_1 \cdot r_2$

3. Kleensche Hülle:

   $r = r_1{}^*$

11.1.2 Vereinigung zweier regulärer Ausdrücke als NFAs

Gegeben sind zwei reguläre Ausdrücke r_1 und r_2 mit jeweils weniger als i Operatoren, die regulären Sprachen $L(r_1)$ und $L(r_2)$ beschreiben. Nach Induktionsannahme gibt es zwei entsprechende NFAs $M_1 = \left(Q_1, \Sigma_1, \delta_1, q_1, \{f_1\}\right)$ und $M_1 = \left(Q_2, \Sigma_2, \delta_2, q_2, \{f_2\}\right)$, die diese Sprachen erkennen. Die beiden Automaten haben voneinander disjunkte Zustände, die unterschiedlich zu benennen sind, um sie in einer gemeinsamen Zustandmenge vereinigen zu können. Damit bilden wir einen neuen Automaten als Vereinigung von M_1 und M_2.

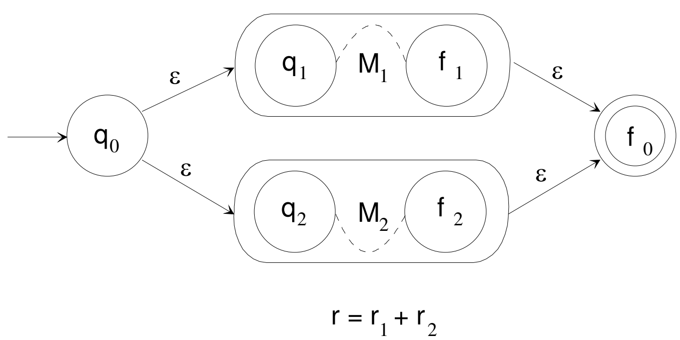

Abbildung 20 - Vereinigung von regulären Ausdrücken

Zu den Zuständen der beiden Automaten kommt ein gemeinsamer Startzustand q_0 und ein gemeinsamer Endzustand f_0, aus dem keine weiteren Übergänge herausführen. Die erzeugte reguläre Sprache L(r) soll entweder Wörter akzeptieren, die in $L(r_1)$ oder in $L(r_2)$ enthalten. Daher geht der konstruierte Automat beim Start von q_0 in einem ε-Übergang entweder in M_1 oder in M_2. Gemäß Induktionsannahme erkennt er dort ein entsprechendes Wort des jeweiligen Automaten und verläßt den Teilautomaten nicht, bevor er in den Endzustand f_1 bzw. f_2 eingetreten ist. In einem weiteren ε-Übergang geht der Automat vom Endzustand eines Teilautomaten in den gemeinsamen neuen Endzustand f_0.

Die Vereinigung zweier regulärer Ausdrücke, die sich in NFAs überführen lassen, liefert also wieder einen regulären Ausdruck, für den ein NFA existiert. Formal läßt sich der Automat, der L(r) erkennt als M ausdrücken:

$$M = \left(Q_1 \cup Q_2 \cup \{q_0, f_0\}, \Sigma_1 \cup \Sigma_2, \delta, q_0, \{f_0\}\right) \tag{11.1}$$

Die Übergangsfunktion δ ist folgendermaßen gestaltet:

$$\begin{aligned} &\delta(q_0, \varepsilon) = \{q_2, q_2\} \\ &\delta(\{f_1, f_2\}, \varepsilon) = \{f_0\} \end{aligned} \tag{11.2}$$

Die Übergänge in den beiden Teilautomaten M_1 und M_2 entsprechen den Übergangsfunktionen δ_1 und δ_2. Dabei werden alle Zustände außer den beiden Endzuständen f_1 und f_2 betrachtet. Da f_1 und f_2 Endzustände von M_1 und M_2 sind, aus denen per Induktionsannahme keine Übergänge herausführen, können sie nicht in der folgenden Definition enthalten sein, sondern gehen gemäß obigem Abschnitt mit einem ε-Übergang zum gemeinsamen Endzustand f_0 über.

$$\begin{aligned} &\delta(q, a) = \delta_1(q, a) \quad \text{für } q \in Q_1 \setminus \{f_1\} \text{ und } a \in \Sigma_1 \cup \varepsilon \\ &\delta(q, a) = \delta_2(q, a) \quad \text{für } q \in Q_2 \setminus \{f_2\} \text{ und } a \in \Sigma_2 \cup \varepsilon \end{aligned} \tag{11.3}$$

Somit gilt der Induktionsschritt für Vereinigung:

$$L(M) = L(M_1) \cup L(M_2) \Leftrightarrow L(r) = L(r_1) \cup L(r_2) \tag{11.4}$$

11.1.3 Schnitt zweier regulärer Ausdrücke als NFAs

Die Konkatenation zweier regulärer Ausdrücke r_1 und r_2 zu einem neuen regulären Ausdruck r gestaltet sich ähnlich. Seien M_1 und M_2 NFAs, die $L(r_1)$ und $L(r_2)$ akzeptieren, dann sei M ein Automat, der die Sprachen beider Automaten konkateniert.

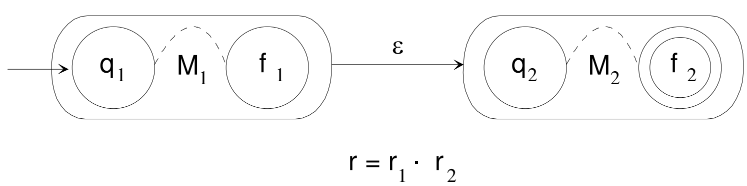

Abbildung 21 - Konkatenation von regulären Ausdrücken

Der Startzustand von M_1 ist gleichzeitig Startzustand von M. Der Automat durchläuft in M_1 die einzelnen Zustände, bis er ein Wort gemäß r_1 erkennt und in f_1 übergeht. In einem ε-Übergang geht der Automat von f_1 in q_2, den Startzustand von M_2, über. Hier tritt er über mögliche Zwischenzustände in f_2, den Endzustand von M_2, der auch gleichzeit Endzustand von M ist.

M akzeptiert also eine reguläre Sprache L(M), die gemäß Abschlußeigenschaften regulärer Sprachen aus der Konkatenation der per Induktionsannahme regulären Sprachen $L(M_1)$ und $L(M_2)$ entsteht. Für r existiert so ein NFA mit ε-Übergängen. Formal definiert sich M als:

$$M = \left(Q_1 \cup Q_2, \Sigma_1 \cup \Sigma_2, \delta, q_1, \{f_2\}\right) \tag{11.5}$$

Die Übergangsfunktion δ für f_1 ist folgendermaßen gestaltet. Da f_1 Endzustand von M_1, führen keine Übergänge aus f_1 heraus, eine Verbindung zwischen M_1 und M_2 wird daher benötigt.

$$\delta(f_1, \varepsilon) = \{q_2\} \tag{11.6}$$

In den beiden Teilautomaten wird δ entsprechend mit δ_1 und δ_2 definiert. Der Übergang von f_1 wurde bereits definiert, f_2 verhält sich wie in M_2 und ist gleichzeitig Endzustand von M.

$$\begin{aligned} &\delta(q, a) = \delta_1(q, a) \quad \text{für } q \in Q_1 \setminus \{f_1\} \text{ und } a \in \Sigma_1 \cup \varepsilon \\ &\delta(q, a) = \delta_2(q, a) \quad \text{für } q \in Q_2 \text{ und } a \in \Sigma_2 \cup \varepsilon \end{aligned} \tag{11.7}$$

Somit ist der Induktionsschritt für Konkatenation gezeigt:

$$L(M) = L(M_1) \cdot L(M_2) \quad \Leftrightarrow \quad L(r) = L(r_1) \cdot L(r_2) \tag{11.8}$$

11.1.4 Kleensche Hülle eines regulären Ausdruckes als NFA

Bei der Hüllenbildung (auch Stern genannt) über den regulären Ausdruck r_1 kann r_1 einmal, keinmal oder beliebig oft auftreten. Dementsprechend führen wir einen Automaten M ein, der den zu r_1 passenden Automaten $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, \{f_1\})$ erweitert.

Tritt r_1 keinmal auf, so geht M vom neuen Startzustand q_0 in einem ε-Übergang direkt in den neuen Endzustand f_0 über. Andernfalls ist r_1 mindestens einmal vorhanden und M geht in den Bereich von M_1. Hier kann der reguläre Ausdruck beliebig oft wiederholt werden, da von f_1 mit einem ε-Übergang in q_1 verzweigt werden kann. Folgen keine weiteren Wiederholungen von r_1, geht M in einem weiteren ε-Übergang in den Endzustand f_0.

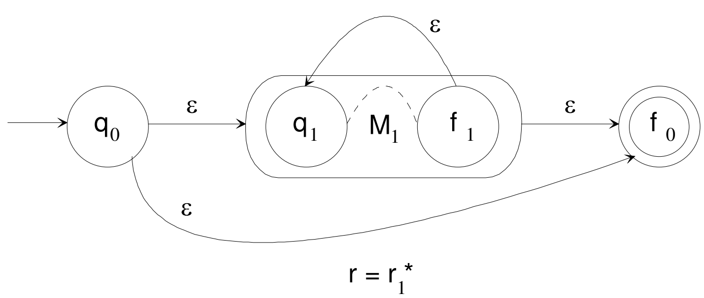

Abbildung 22 - Kleensche Hülle über regulärem Ausdruck

M akzeptiert somit reguläre Sprachen, die L(M) = $L(M_1)^*$ genügen und kann folgendermaßen definiert werden

$$M = (Q_1 \cup \{q_0, f_0\}, \Sigma_1, \delta, q_0, \{f_0\}) \tag{11.9}$$

Die zugehörige Übergangsfunktion δ entspricht im Wesentlichen δ_1, muß allerdings für die Sterneigenschaften erweitert werden

$$\begin{aligned} &\delta(q_0, \varepsilon) = \{q_1, f_0\} \\ &\delta(q, a) = \delta_1(q, a) \quad \text{für } q \in Q_1 \setminus \{f_1\} \text{ und } a \in \Sigma_1 \cup \varepsilon \\ &\delta(f_1, a) = \{q_1, f_0\} \end{aligned} \tag{11.10}$$

Somit ist der Induktionsschritt für die Kleensche Hülle (oder Stern) gezeigt. Es existiert ein NFA M, der beliebig viele Konkatenationen der von r_1 akzeptierten Sprache sowie das leere Wort akzeptiert.

$$L(M) = L(M_1)^* \quad \Leftrightarrow \quad L(r) = L(r_1)^* \tag{11.11}$$

Mit den drei Fällen Vereinigung, Konkatenation und Hüllenbildung ist die Induktion über die Anzahl der Operatoren abgeschlossen. Verschachtelte, vollständig geklammerte Ausdrücke können mit Hilfe der Abschlußeigenschaften in diese Grundformen zerlegt werden.

11.1.5 Beispiel: Konstruktion eines NFA mit ε-Übergängen für einen regulärem Ausdruck

Als Beispiel betrachten wir einen regulären Ausdruck r = 01*+1. Mit vollständiger Klammerung und Aufspaltung in die Operationen Vereinigung, Konkatenation und Stern sowie Substitution mit symbolischen Ausdrücken r_i (ihrerseits wieder reguläre Ausdrücke) erhält man

$$
\begin{aligned}
01^* + 1 &= \left(0\left(1^*\right)\right) + 1 \\
&= \left(r_2 \cdot \left(r_1^*\right)\right) + r_3 \\
&= \left(r_2 \cdot r_4\right) + r_3 \\
&= r_5 + r_3 \\
&= r
\end{aligned}
\tag{11.12}
$$

So zerlegt lassen sich für jeden Ausdruck r_i wie im vorangegangenen Abschnitt beschrieben Automaten konstruieren und diese dann zusammensetzen. Dabei beginnt man mit dem am weitesten in der Klammerhierarchie innenliegenden Ausdruck r_1 und erweitert den Automaten immer um weitere Zustände für die umgebenden Ausdrücke.

Für r_1 ist die Operation Stern anzuwenden, daher konstruiert man einen Automaten mit 4 Zuständen, einem Start- und einem Endzustand q_1 und q_4, sowie zwei weiteren Zuständen q_2 und q_3. Vom Startzustand q_1 kommt man in einem ε-Übergang zu q_2 oder direkt zum Endzustand q_4 (leeres Wort). Mit einer 1 kommt man zu q_3 und von dort per ε-Übergang zu q_2 zurück (Stern) oder zum Endzustand q_4.

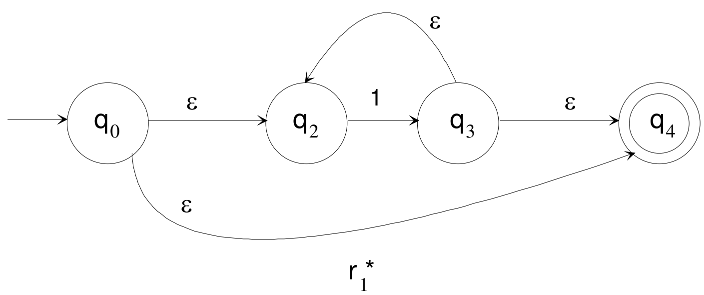

Abbildung 23 - Automat für 1*

Die Konkatenation (Und) von r_2 und r_4 benötigt lediglich zwei weitere Zustände q_5 und q_6, die vor den Automaten für r_4 geschaltet werden. Von q_5 geht der Automat nur in q_6 über, wenn eine 0 als Zeichen anliegt. Von q_6 gelangt der Automat in einem ε-Übergang in q_0, den Anfangszustand des oben beschriebenen Automaten. Dadurch ergibt sich folgender Automat

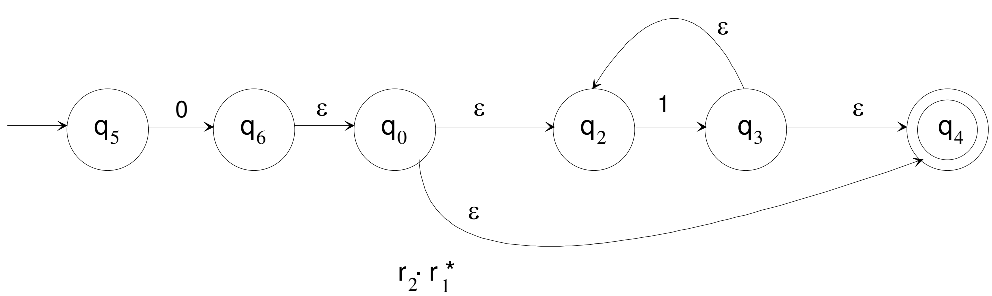

Abbildung 24 - Automat für 01*

Der noch verbleibende Ausdruck r_3 muß mit $r_5 = r_2 \cdot r_1{}^*$ in Vereinigung (Oder) aufgehen. Dazu benötigt man einen neuen Startzustand q_7 und einen neuen Endzustand q_8. Für die Konstruktion eines Automaten, der r_3 erkennt, benötigt man weitere zwei Zustände q_9 und q_{10}. Vom neuen Startzustand q_7 gehen ε-Übergänge zu den Startzuständen der beiden alternativen Automaten q_5 und q_9. Genauso gehen von den Endzuständen der Teilautomaten q_4 und q_{10} ε-Übergänge zum neuen Endzustand q_9. Weiter tritt der Automat von q_9 bei Zeichen 1 in den Zustand q_{10}.

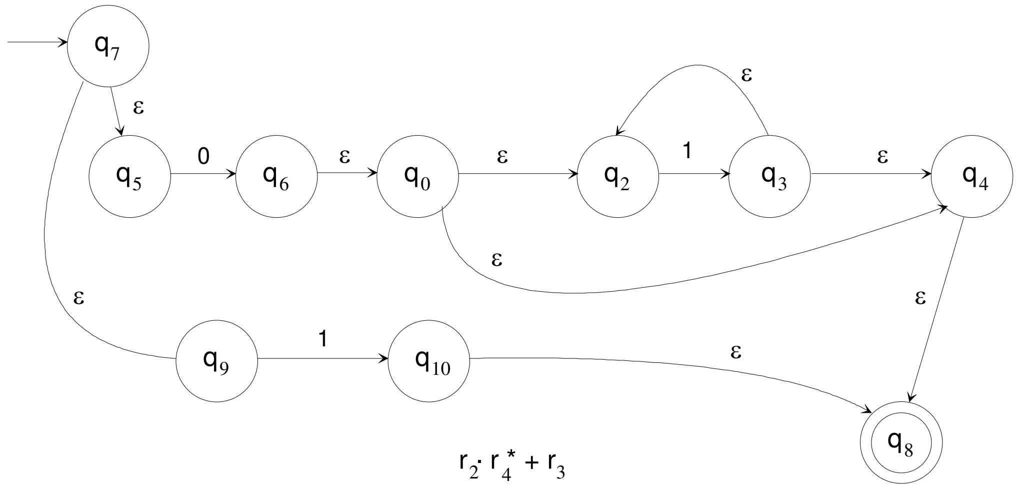

Abbildung 25 - Gesamter Automat

Für den regulären Ausdruck 01*+1 exisiert wie gezeigt ein endlicher, deterministischer Automat mit ε-Übergängen, der folgende Definition hat. Für die zehn Zustände gibt es einen Startzustand q_7 und einen Endzustand q_8. Die Übergangsfunktion entspricht dem oben aufgeführten Graphen für den gesamten Automaten.

$$M = \left(\left\{q_1, q_2, \ldots, q_{10}\right\}, \Sigma, \delta, q_7, \left\{q_8\right\}\right) \tag{11.13}$$

Auf diese Weise lassen sich für alle regulären Ausdrücke DFAs und NFAs konstruieren, die die gleiche Sprache wieder reguläre Ausdruck erkennen.

11.1.6 Von endlichen Automaten zu regulären Ausdrücken

Gemäß der Darstellung des Schaubildes zu Beginn des Protokolls, haben wir gezeigt, das sich reguläre Ausdrücke in nichtdeterministische Automaten mit ε-Übergängen umwandeln lassen. Aus vorangegangenen Protokollen ist bekannt, dass DFAs, NFAs und NFAs mit ε-Übergängen äquivalent sind. Jetzt ist der Weg zurück zu den regulären Ausdrücken zu zeigen, der letzte Pfeil im Schaubild von DFAs zu regulären Ausdrücken.

Satz:

Wenn eine Sprache L von einem DFA akzeptiert wird, dann läßt sich diese Sprache L durch einen regulären Ausdruck r beschreiben.

Beweis

Es ist zu zeigen, dass der zu bestimmende reguläre Ausdruck r die gleiche Sprache beschreibt wie der gegebene Automat M, also $L(r) = L(M)$ gilt. Sei M ein endlich deterministischer Automat, dessen Zustände von 1 bis n durchnummeriert sind und q_1 der Startzustand ist.

$$\begin{aligned} M &= \left(Q, \Sigma, \delta, q_1, F\right) \\ Q &= \left\{q_1, q_2, \ldots, q_n\right\} \end{aligned} \tag{11.14}$$

$R_{i,j}^k$ seien Sprachen mit $i, j \in \left\{1, \ldots, n\right\}$ und $k \in \left\{1, \ldots, n\right\}$, die wir durch reguläre Ausdrücke beschreiben wollen. $R_{i,j}^k$ enthält alle Wörter x, die den Automaten von Zustand q_i über verschiedene Zwischenzustände nach q_j überführen. Dabei seien q_l die besuchten Zustände[8] und es gelte $l \leq k$ für alle Wörter x.

$$R_{i,j}^k = \left\{x \in \Sigma^* \middle| \delta(q_i, x) = q_j \text{ mit Zwischenzuständen } q_l \setminus \left\{q_i, q_j\right\} \middle| l < k\right\} \tag{11.15}$$

[8] Besuchen bezeichnet hier in den Zustand einzutreten und wieder herauszutreten

Diese Sprachen R können rekursiv definiert und per Induktion über k bewiesen werden. Für k = 0 als *Induktionsannahme* geht der Automat direkt von Zustand q_i nach q_j ohne dabei Zwischenzustände zu besuchen, da kein Zustand q_p mit $p \le 0$ existiert. Für den Fall $i \neq j$ enthält die Sprache $R_{i,j}^0$ alle Zeichen, für die der Automat beginnend in q_i direkt in q_j übergeht.

$$R_{i,j}^0 = \left\{ a \in \Sigma \middle| \delta(q_i, a) = q_j \right\} \tag{11.16}$$

Für den Fall, dass $i = j$ gegeben ist, erkennt der Automat alle Zeichen ohne über Zwischenzustände zu gehen, bei denen er von q_i in q_i geht, also im gleichen Zustand verbleibt. Im „Zustand Verbleiben" heißt aber auch, dass der Automat keinen Zustandswechsel macht und so dass leere Wort zur Sprache gehört

$$R_{i,i}^0 = \left\{ \varepsilon \right\} \cup \left\{ a \in \Sigma \middle| \delta(q_i, a) = q_i \right\} \tag{11.17}$$

Für diese Fälle ist $R_{i,j}^0$ eine endliche Menge von Zeichen und ein regulärer Ausdruck läßt sich aufstellen. Die *Induktionsannahme* ist damit bewiesen. Ein entsprechender regulärer Ausdruck besteht aus der Veroderung der auftretenden Zeichen sowie dem leeren Wort, falls $i = j$

$$\begin{aligned} i \neq j: &\quad r_{i,j}^0 = a_1 + a_2 + \ldots + a_t \\ i = j: &\quad r_{i,i}^0 = a_1 + a_2 + \ldots + a_t + \varepsilon \end{aligned} \tag{11.18}$$

Für den *Induktionsschritt* von k auf k+1 verwenden wir die Rekursionsgleichung und setzen $R_{i,j}^k$ als gegeben voraus.

$$R_{i,j}^{k+1} = R_{i,j}^k \cup R_{i,k+1}^k \left(R_{k+1,k+1}^k \right)^* R_{k+1,j}^k \tag{11.19}$$

Wörter x, bei denen der Automat von q_i nach q_j übergeht, müssen den Zustand q_{k+1} nicht passieren. Dann wird dieser Zustand nicht benötigt und die Sprache $R_{i,j}^{k+1}$ läßt sich durch das bereits bewiesene $R_{i,j}^k$ ausdrücken. Wird der Zustand benötigt, geht der Automat von q_i in q_{k+1} über und kann innerhalb von $\left(R_{k+1,k+1}^k \right)^*$ q_{k+1} beliebig oft aufgerufen werden (Stern, Kleensche Hülle). Anschließend tritt er in q_j ein und erkennt somit Wörter, bei denen der Automat durch q_{k+1} als Zwischenzustand geht.

Ein entsprechender regulärer Ausdruck $r_{i,j}^{k+1}$ für die Sprache $R_{i,j}^{k+1}$ läßt sich aufstellen, da die Rekursionsgleichung nur die Operationen für reguläre Ausdrücke Vereinigung, Konkatenation und Hüllenbildung enthält. Nach Induktionsannahme beschreiben die regulären Ausdrücke für k bereits reguläre Sprachen. Die Anwendung der Operationen auf regulären Ausdrücke ergibt wieder reguläre Ausdrücke genauso wie die Anwendung der Operationen auf reguläre Sprachen wieder eine reguläre Sprache ergibt (siehe Abschlußeigenschaften). Damit läßt sich der folgende reguläre Ausdruck aufstellen

$$r_{i,j}^{k+1} = \left(r_{i,j}^k + r_{i,k+1}^k \left(r_{k+1,k+1}^k \right)^* r_{k+1,j}^k \right) \tag{11.20}$$

Der *Induktionsschritt* ist damit bewiesen und es lassen sich beliebige reguläre Ausdrücke aufstellen, die Teilmengen von L(M) beschreiben. Für dem Automaten M betrachten wir jetzt alle Wörter, die den Automaten vom Startzustand q_1 in einen Endzustand q_i überführen und dabei alle n Zustände des Automaten (siehe Definition am Anfang des Beweises) besuchen. Die Menge dieser Wörter ist die Sprache, die der Automat erkennt.

$$L(M) = \bigcup_{q_i \in F} R_{1,i}^n \tag{11.21}$$

Damit läßt sich analog zum vorangeganen Abschnitt ein regulärer Ausdruck konstruieren und so der Beweis abschließen. Für die möglichen Endzustände

$$F = \left\{ q_{i_1}, q_{i_2}, \ldots, q_{i_p} \right\} \tag{11.22}$$

verwenden wir die Indizes $i_1, i_2, \ldots, i_p$. Der reguläre Ausdruck ist so die Veroderung regulärer Ausdrücke, die die Wörter beschreiben, bei denen der Automat in jeweils einem gemeinsamen Endzustand hält.

$$r = r_{1,i_1}^n + r_{1,i_1}^n + \ldots + r_{1,i_p}^n \tag{11.23}$$

Somit wurde wurde gezeigt, dass man zu einer von einem DFA M akzeptierten Sprache L(M) einen regulären Ausdruck r angeben kann, der die gleiche reguläre Sprache L(r) = L(M) beschreibt.

Reguläre Ausdrücke, endliche deterministische, endliche nichtdeterministische sowie endliche nichtdeterministische Automaten mit ε-Übergängen sind somit wie im heutigen und den vorangegangen Protokollen gezeigt, äquivalent und beschreiben die Menge der regulären Sprachen.

12 Vorlesung vom 16. Mai 2000

12.1 Wiederholung der letzten Vorlesung

12.1.1 Abschlusseigenschaften

Für die folgenden Sprachen soll überprüft werden, ob sie zur Klasse REG der regulären Sprachen gehören.

Beispiel 1

Wir wissen, dass die Sprache $L = \left\{ a^n b^n \,\middle|\, n \geq 0 \right\}$ nicht regulär ist, folgt daraus, dass die Sprache $L' = \left\{ a^n c b^n \,\middle|\, n \geq 0 \right\}$ auch nicht regulär ist? Der argumentative Beweis nach Myhill-Nerode sieht wie folgt aus:

$$L_1 = \left\{ a^n c b^n \,\middle|\, n \geq 0 \right\} \notin REG \Leftarrow \mathrm{L}_2 = \left\{ \mathrm{a^n b^n} \,\middle|\, n \geq 0 \right\} \notin REG \tag{12.1}$$

Es ist nun zu zeigen, dass der $Index(L_1)$ unendlich ist. Dies ist aber leicht zu sehen, wenn man die Wörter der Sprache, als Zusammensetzung aus $a^i c$ und b^i für alle i sieht. Es ist Fakt, dass es unendlich viele Wörter $a^i c$ gibt. Somit ist der Index unendlich und damit $L_1 \notin REG$.

Beispiel 2

Nun ist die Frage, ob auch die Rückrichtung gilt. In anderen Worten gilt:

$$\mathrm{L}_1 = \left\{ a^n c b^n \,\middle|\, n \geq 0 \right\} \notin REG \Rightarrow L_2 = \left\{ \mathrm{a^n b^n} \,\middle|\, n \geq 0 \right\} \notin REG \tag{12.2}$$

Nehmen wir an $\mathrm{L}_1 = \left\{ a^n c b^n \,\middle|\, n \geq 0 \right\} \notin REG$ sei regulär. Sei weiterhin h ein Homomorphismus, welcher die folgenden Abbildungen durchführt:

$$\begin{aligned} h(a) &= a \\ h(b) &= b \\ h(c) &= \varepsilon \end{aligned} \tag{12.3}$$

Bildet man nun die Sprache:

$$L' = h^{-1}(L_2) \tag{12.4}$$

und vereinigt diese mit beliebigen Folgen der Form $a^* c b^*$ so ergibt sich die Sprache:

$$L'' = L' \cap a^* c b^* \tag{12.5}$$

Diese wäre aufgrund der Abgeschlossenheit der Regulären Sprachen immer noch regulär. Jedoch kann man diese Sprache auch umformulieren zu:

$$L'' = \left\{ a^n c b^n \,\middle|\, n \geq 0 \right\} \tag{12.6}$$

Jedoch wissen wir, von unserer Annahme, $L_2 = \left\{ \mathrm{a^n b^n} \,\middle|\, n \geq 0 \right\}$ sei regulär, dass diese falsch ist. Somit kann auch die Sprache $L_1 = L'' = \left\{ a^n c b^n \,\middle|\, n \geq 0 \right\}$ nicht regulär sein.

Beispiel 3:

Man kann weiterhin zeigen, dass gilt:

$$L_1 = \left\{ a^n c b^n \,\middle|\, n \geq 0 \right\} \notin REG \Rightarrow L_2 = \left\{ a^n b^r c^n d^2 \,\middle|\, n, r, s \geq 0 \right\} \notin REG \tag{12.7}$$

Angenommen die Sprache L_2 wäre regulär, dann wäre auch die Sprache

$$\begin{aligned} &L' = L_2 \cap a^* b c^* d^* \\ &\Updownarrow \\ &L' = \left\{ a^n b c^n d^s \middle| n, s \geq 0 \right\} \end{aligned} \tag{12.8}$$

regulär. Nun kann man einen Homomorphismus h verwenden mit den Abbildungen:

$$\begin{aligned} &h(a) = a \\ &h(b) = c \\ &h(c) = b \\ &h(d) = \varepsilon \end{aligned} \tag{12.9}$$

Sei nun L'' die Sprache, welche erzeugt wird, indem man den Homomorphismus auf die Sprache L' anwendet:

$$L'' = h(L') = \left\{ a^n c b^n \middle| n \geq 0 \right\} \tag{12.10}$$

Diese müsste auch regulär sein, da wir jeweils nur strukturerhaltende Operationen verwendet haben. Wir wissen jedoch, dass die Sprache bestehend aus Wörtern der Form $\left\{ a^n c b^n \middle| n \geq 0 \right\}$ nicht regulär somit ergibt sich ein Widerspruch.

Beispiel 4:

Gilt:

$$L_1 = \left\{ a^n c b^n \middle| n \geq 0 \right\} \notin REG \Rightarrow L_2 = \left\{ w \middle| w \in \{a, b\}^*, \#_a(w) = 2\#_b(w) \right\} \notin REG \text{ ?} \tag{12.11}$$

Angenommen, dass L_2 regulär ist, dann ist auch die Sprache:

$$\begin{aligned} &L' = L_2 \cap a^* b^* \\ &\Updownarrow \\ &L' = \left\{ a^{2n} b^n \middle| n \geq 0 \right\} \end{aligned} \tag{12.12}$$

regulär. Wieder definieren wir einen Homomorphismus mit den Abbildungen:

$$\begin{aligned} &h(a) = aa \\ &h(b) = b \\ &h(c) = \varepsilon \end{aligned} \tag{12.13}$$

Sei nun L'' die Sprache, welche erzeugt wird, indem man den inversen Homomorphismus auf die Sprache L' anwendet:

$$\begin{aligned} &L'' = h^{-1}(L') \\ &\text{Man definiert nun:} \\ &L''' = L'' \cap a^* c b^* \\ &\Updownarrow \\ &L'' = \left\{ a^n c b^n \middle| n \geq 0 \right\} \end{aligned} \tag{12.14}$$

Diese müsste auch regulär sein, da wir jeweils wieder nur strukturerhaltende Operationen verwendet haben. Wir wissen jedoch, dass die Sprache bestehend aus Wörtern der Form $\left\{ a^n c b^n \middle| n \geq 0 \right\}$ nicht regulär somit ergibt sich ein Widerspruch.

Beispiel 5:

Zum Schluss gilt noch zu überprüfen, ob:

$$L_1 = \{a^n b^n \mid n \geq 0\} \notin REG \Rightarrow L_2 = \{a^n b^m \mid 0 \leq m \leq n\} \notin REG \tag{12.15}$$

Wieder nehmen wir zunächst an, dass die Sprache L_2 regulär sei. Dann ist auch die Sprache:

$$L' = a^* b^* \cap \overline{L}_2 = \{a^n b^m \mid 0 \leq n < m\} \tag{12.16}$$

regulär. Durch Konkatenation bilden wir:

$$\begin{aligned} L'' &= \{a\} \cdot L' = \{a^n b^m \mid 0 < n \leq m\} \\ L''' &= L_2 \cap L'' = \{a^n b^n \mid n \geq 1\} \\ L'''' &= L''' \cap \{\varepsilon\} \end{aligned} \tag{12.17}$$

Dann sind $L'''' = L_2$. Doch wiederum gilt, da die Sprache L_1 nicht regulär ist, dass auch L_2 nicht regulär ist.

12.2 Beispiel zur Umwandlung eines Automaten dem entsprechenden Regulären Ausdruck

In diesem Abschnitt beschäftigen wir uns hauptsächlich mit dem folgenden Automaten:

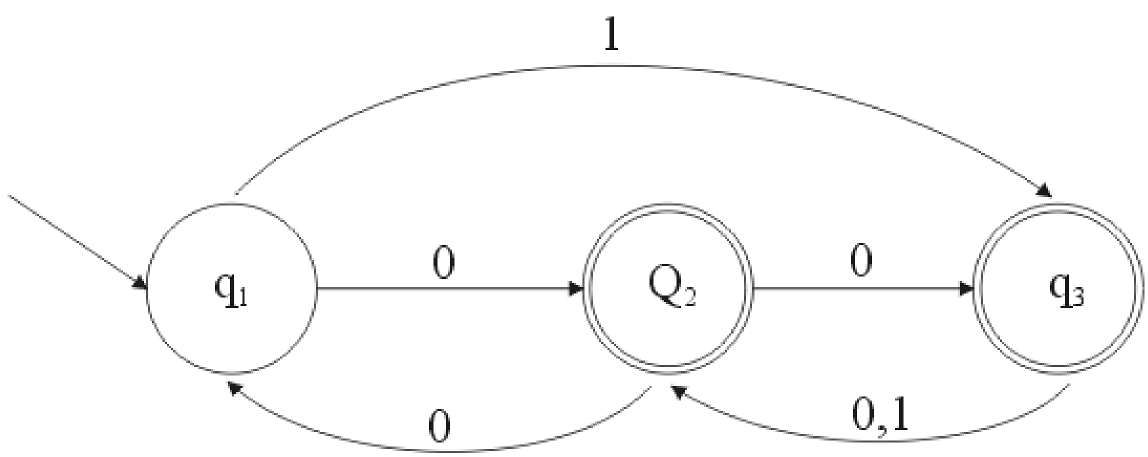

Abbildung 26: der endliche Automat des Beispiels

	k=0	k=1	k=2
r_{11}^k	ε	ε	(00)
r_{12}^k	0	0	$0(00)$
r_{13}^k	1	1	0^*1
r_{21}^k	0	0	$0(00)^*$
r_{22}^k	ε	$\varepsilon+00$	$(00)^*$
r_{23}^k	1	$1+01$	0^*1

	k=0	k=1	k=2
r_{31}^k	$\varnothing$	$\varnothing$	$(0+1)(00)^*0$
r_{32}^k	$0+1$	$0+1$	$(0+1)(00)^*$
r_{33}^k	ε	ε	$\varepsilon+(0+1)0^*1$

Sei M der abgebildete endliche Automat. Die Wert $r_{ij}^{\ k}\ \forall\ i$ und j, und $\mathrm{k}=1{,}2$ oder 3 sind in Tabelle 1 abgebildet. Dabei wurden die regulären Ausdrücke vereinfacht, und zwar in der Form: $(r+s)t=rt+st$ oder $(\varepsilon+r)^*=r^*$. Zum Beispiel ergibt sich für $r_{22}^{\ 1}$:

$$r_{22}^{\ 1}=r_{21}^{\ 0}\left(r_{11}^{\ 0}\right)^*r_{12}^{\ 0}+r_{22}^{\ 0}=0(\varepsilon)^*0+\varepsilon \tag{12.18}$$

Ebenso ist:

$$r_{13}^{\ 2}=r_{12}^{\ 1}\left(r_{22}^{\ 1}\right)^*r_{23}^{\ 1}+r_{13}^{\ 1}=0(\varepsilon+00)^*(1+01) \tag{12.19}$$

Bedenkt man, dass $(\varepsilon+00)^*=(00)^*$ und weiterhin gilt $1+01=(\varepsilon+0)1$, dann ist:

$$r_{13}^{\ 2}=0(00)^*(\varepsilon+0)1+1 \tag{12.20}$$

Da weiterhin der Ausdruck $(00)^*(\varepsilon+0)$ äquivalent zu 0^* ist, ergibt sich für $0(00)^*(\varepsilon+0)1+1$ zunächst 00^*1+1 und schließlich 0^*1. Die Regulären Ausdrucke für M ergeben sich aus den akzeptierenden Zuständen. Also ist $r_{12}^{\ 3}+r_{13}^{\ 3}$ zu bestimmen. Mit:

$$\begin{aligned} r_{12}^{\ 3} &= r_{12}^{\ 2}\left(r_{33}^{\ 2}\right)^*r_{32}^{\ 2}+r_{12}^{\ 2} \\ &= 0^*1\left(\varepsilon+(0+1)0^*1\right)^*(0+1)(00)^*+0(00)^* \\ &= 0^*1\left((0+1)0^*1\right)^*(0+1)(00)^*+0(00)^* \end{aligned} \tag{12.21}$$

und

$$\begin{aligned} r_{13}^{\ 3} &= r_{13}^{\ 2}\left(r_{33}^{\ 2}\right)^*r_{33}^{\ 2}+r_{13}^{\ 2} \\ &= 0^*1\left(\varepsilon+(0+1)0^*1\right)^*\left(\varepsilon+(0+1)0^*1\right)+0^*1 \\ &= 0^*1\left((0+1)0^*1\right)^* \end{aligned} \tag{12.22}$$

ergibt sich dann:

$$r_{12}^{\ 3}+r_{13}^{\ 3}=0^*\left((0+1)0^*1\right)^*\left(\varepsilon+(0+1)(00)^*\right)+0(00)^* \tag{12.23}$$

Dies stellt den Regulären Ausdruck für den Beispielautomaten dar.

12.3 Endliche 2-Wege Automaten

Dieser Abschnitt dient nur zur Information, da diese Automaten kein Thema dieser Vorlesung sein sollen. Deterministische endliche Zweiwegautomaten sind definiert als Quintupel $M=(Q,\Sigma,\delta,q_0,F)$ unterschiedlich zum „normalen“ DFA ist hier zum einem dass die Übergangsfunktion neben einem Zustand noch eine Laufrichtung zurückgibt, in welcher der nächste Zustand besucht werden soll. δ ist also auf $Q \times \Sigma^{*} \rightarrow Q \times \{L, R\}$ definiert. Es kann jedoch gezeigt werden, das ein und zwei-Wege endliche Automaten gleichwertig sind.