
Grundlagen der theoretischen Informatik

Kurt Sieber

Fachbereich Mathematik/Theoretische Informatik
Universität Siegen

Vorlesung vom 18.10.2004 (Stand: 18.10.2004)

Rechenregeln für Potenzen

... von Wörtern

- $w^0 = \varepsilon$ (neutrales Element)
- $w^1 = w$
- $w^m w^n = w^{m+n}$
- $(w^m)^n = w^{mn}$

... von Sprachen

- $L^0 = \{\varepsilon\}$ (neutrales Element)
- $L^1 = L$
- $L^m L^n = L^{m+n}$
- $(L^m)^n = L^{mn}$

Diese Regeln gelten in **jedem** Monoid (wenn man das Potenzieren nach dem üblichen Schema definiert). Denn sie folgen allein aus der Assoziativität und der Eigenschaft des neutralen Elements.

Warnung: Es gilt **nicht** $(w_1 w_2)^n = w_1^n w_2^n$ oder $(L_1 L_2)^n = L_1^n L_2^n$. Denn dazu bräuchte man die Kommutativität.

Zeichen, Wörter, Sprachen

Rechenregeln für * und +

- $L \subseteq L^*$
- $(L^*)^* = L^*$
- $L \subseteq L^+$
- $(L^+)^+ = L^+$

Mengenoperatoren mit diesen beiden Eigenschaften nennt man **Abchluss-** oder **Hüllenoperatoren**.

Beweis für *

1. $L \subseteq L^*$ ist klar wegen $L = L^1$ und $L^* = \bigcup_{n \geq 0} L^n$.

2. $L^* = (L^*)^*$:

‘ \subseteq ’ ist klar wegen 1.

‘ \supseteq ’: Sei $w \in (L^*)^*$, d.h. $w = w_1 \dots w_n$ mit $n \geq 0$ und $w_i \in L^*$ für $i = 1, \dots, n$. Dann existiert für jedes i ein $m_i \geq 0$ mit $w_i \in L^{m_i}$, also folgt $w = w_1 \dots w_n \in L^{m_1} \dots L^{m_n} = L^{m_1 + \dots + m_n} \subseteq L^*$. \square

Zeichen, Wörter, Sprachen

Die Operatoren $\cup, \circ, *, \dots$ lassen sich auch zur **Beschreibung** von Sprachen verwenden. Konvention: Die Postfixoperatoren $n, +$ und $*$ binden am stärksten, \circ bindet stärker als \cup .

Beispiele

- Die Menge aller Dezimaldarstellungen natürlicher Zahlen:

$$L_{nat} = \{0, \dots, 9\}^+ = \{0, \dots, 9\}\{0, \dots, 9\}^*$$

- Die Menge aller Dezimaldarstellungen ganzer Zahlen:

$$L_{int} = \{-, \varepsilon\}L_{nat}$$

- Die Menge aller dezimalen Festpunktzahlen:

$$L_{fix} = L_{int}\{.\} \cup \{.\}L_{nat} \cup L_{int}\{.\}L_{nat}$$

- Die Menge aller dezimalen Fließpunktzahlen:

$$L_{float} = L_{int}\{e, E\}L_{int} \cup L_{fix}(\{\varepsilon\} \cup \{e, E\}L_{int})$$

Zeichen, Wörter, Sprachen

Die Menge aller Wörter über dem Alphabet $\{0, 1\}$, ...

- ... die mit 00 beginnen:

$$L_1 = \{00\}\{0, 1\}^*$$

- ... die 00 als Teilwort enthalten:

$$L_2 = \{0, 1\}^*\{00\}\{0, 1\}^*$$

- ... die mindestens zwei Nullen enthalten:

$$L_3 = \{0, 1\}^*\{0\}\{0, 1\}^*\{0\}\{0, 1\}^* = \{1\}^*\{0\}\{1\}^*\{0\}\{0, 1\}^*$$

- ... die genau zwei Nullen enthalten:

$$L_4 = \{1\}^*\{0\}\{1\}^*\{0\}\{1\}^*$$

- ... die höchstens zwei Nullen enthalten:

$$L_5 = \{1\}^*\{0, \varepsilon\}\{1\}^*\{0, \varepsilon\}\{1\}^*$$

- ... deren erstes und letztes Zeichen übereinstimmen:

$$L_6 = \{0, 1\} \cup \{0\}\{0, 1\}^*\{0\} \cup \{1\}\{0, 1\}^*\{1\}$$

Reguläre Sprachen

Definition 1.1 Die Menge aller *regulären Sprachen* über Σ ist induktiv definiert durch:

- Jede endliche Menge $L \subseteq \Sigma^*$ ist regulär.
- Wenn $L_1, L_2 \subseteq \Sigma^*$ regulär sind, dann sind auch $L_1 \cup L_2$ und $L_1 \circ L_2$ regulär.
- Wenn L regulär ist, dann ist auch L^* regulär.

Mit anderen Worten:

Eine Sprache ist genau dann *regulär*, wenn sie sich durch wiederholte Anwendung der Operatoren \cup, \circ und $*$ aus endlichen Sprachen aufbauen lässt.

Damit haben wir eine *endliche Beschreibung* für jede reguläre Sprache.

Beispiele: $L_{nat}, L_{int}, L_{fix}, L_{float}, L_1, \dots, L_6$ sind reguläre Sprachen.

Reguläre Sprachen

Die ‘Mengenausdrücke’ aus Definition 1.1 sind gut lesbare Beschreibungen für reguläre Sprachen (zumindest für die Beispiele aus der Praxis wie Zahldarstellungen, Variablennamen,)

Aber:

- **Nachteil in der Praxis:**
Ein Mengenausdruck für eine Sprache liefert noch keinen Algorithmus, um die Zugehörigkeit zur Sprache zu testen.
- **Nachteil in der Theorie:**
Mit Definition 1.1 kann man nur schwer zeigen, dass eine Sprache **nicht** regulär ist.

Deshalb:

- Alternative Charakterisierungen der regulären Sprachen
 - Insbesondere **algorithmische** Charakterisierungen
-

Endliche Automaten

Ein **endlicher Automat** ist eine 'Maschine', die (nur) **endlich viele Zustände** annehmen kann. Einer der Zustände heißt **Startzustand**, einige Zustände heißen **Endzustände** (besser: **akzeptierende Zustände**).

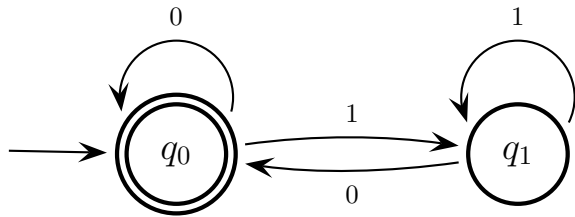
Arbeitsweise

- Der Automat erhält ein Wort w als Eingabe.
- Zu Beginn befindet er sich im Startzustand.
- Er liest w zeichenweise von links nach rechts, wobei jedes Zeichen einen Zustandsübergang bewirkt.
- Ist er nach Abarbeitung von w in einem Endzustand, so wird w akzeptiert, andernfalls wird w abgelehnt.

Ein endlicher Automat kann also benutzt werden, um die Zugehörigkeit zu einer Sprache zu testen.

Endliche Automaten

Graphische Darstellung eines endlichen Automaten



Konvention:

- sind die Zustände
- ist der Startzustand
- ⊙ sind die Endzustände

■ **Akzeptierte Wörter:** 0, 010, 01100

■ **Abgelehnte Wörter:** 1, 101, 11011

Endliche Automaten

Noch zu klären:

- Darf der Startzustand zugleich Endzustand sein?
- Dürfen alle Zustände Endzustände sein?
- Dürfen die Endzustände ganz fehlen?
- Darf von einem Zustand mehr als ein Pfeil mit ein und demselben Zeichen ausgehen?
- Muss von jedem Zustand ein Pfeil mit jedem Zeichen ausgehen?

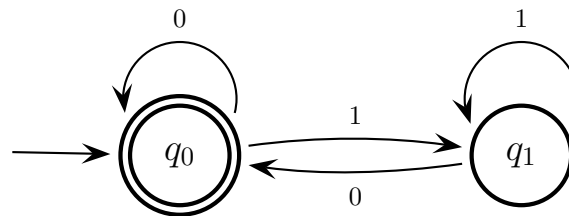
Endliche Automaten

Definition 1.2 Ein *deterministischer endlicher Automat* (kurz: *DEA*) ist ein 5-Tupel $A = (\Sigma, Q, s, F, \delta)$ mit:

- Σ ist ein Alphabet
- Q ist eine endliche Menge, deren Elemente wir *Zustände* nennen
- $s \in Q$ ist der sogenannte *Startzustand*
- $F \subseteq Q$ ist die Menge der sogenannten *Endzustände* oder *akzeptierenden Zustände*
- $\delta : Q \times \Sigma \rightarrow Q$ ist die sogenannte *Übergangsfunktion* (totale Funktion!)

Endliche Automaten

Beispiel (wie oben):



$A = (\Sigma, Q, s, F, \delta)$ mit

- $\Sigma = \{0, 1\}$
- $Q = \{q_0, q_1\}$
- $s = q_0$
- $F = \{q_0\}$
- $\delta : Q \times \Sigma \rightarrow Q$
 - $(q_0, 0) \mapsto q_0$
 - $(q_0, 1) \mapsto q_1$
 - $(q_1, 0) \mapsto q_0$
 - $(q_1, 1) \mapsto q_1$

Oder: δ als Tabelle

	0	1
q_0	q_0	q_1
q_1	q_0	q_1

Endliche Automaten

Definition 1.3 Sei $A = (\Sigma, Q, s, F, \delta)$ ein DEA.

- Eine *Konfiguration* von A ist ein Element $(q, w) \in Q \times \Sigma^*$.
- Auf der Menge der Konfigurationen von A definieren wir die *Übergangsrelation* \vdash_A und die *Schreibweisen* \vdash_A^n für $n \geq 0$, \vdash_A^+ und \vdash_A^* durch

$(q, w) \vdash_A (q', w') \Leftrightarrow$ es existiert ein $a \in \Sigma$ mit
 $w = aw'$ und $\delta(q, a) = q'$

$(q, w) \vdash_A^n (q', w') \Leftrightarrow$ es existieren $(q_0, w_0), \dots, (q_n, w_n) \in Q \times \Sigma^*$
mit $(q, w) = (q_0, w_0)$, $(q', w') = (q_n, w_n)$
und $(q_0, w_0) \vdash_A \dots \vdash_A (q_n, w_n)$.

$(q, w) \vdash_A^+ (q', w') \Leftrightarrow$ es existiert ein $n > 0$ mit $(q, w) \vdash_A^n (q', w')$

$(q, w) \vdash_A^* (q', w') \Leftrightarrow$ es existiert ein $n \geq 0$ mit $(q, w) \vdash_A^n (q', w')$

Endliche Automaten

Intuition

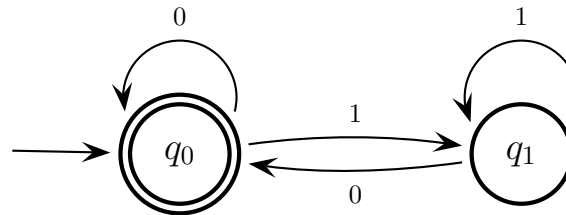
- Eine Konfiguration (q, w) beschreibt die Situation des Automaten zu einem Zeitpunkt: Er befindet sich im Zustand q und hat noch das Wort w zu lesen.
- \vdash_A beschreibt den **Übergangsschritt** von einem Zeitpunkt zum nächsten: Das erste Zeichen des Wortes wird gelesen und der Automat wechselt den Zustand gemäß der Übergangsfunktion δ .
- \vdash_A^n steht für eine Folge von n Übergangsschritten, \vdash_A^+ für eine nichtleere und \vdash_A^* für eine möglicherweise leere Folge

\vdash_A^+ ist der **transitive Abschluss** der Relation \vdash_A , d.h. die kleinste transitive Relation, die \vdash_A enthält.

\vdash_A^* ist der **reflexive transitive Abschluss** von \vdash_A , d.h. die kleinste reflexive transitive Relation, die \vdash_A enthält.

Endliche Automaten

Beispiel (wie oben):



Eine mögliche Folge von Übergangsschritten:

$(q_0, 1001) \vdash_A (q_1, 001)$ wegen $\delta(q_0, 1) = q_1$

$\vdash_A (q_0, 01)$ wegen $\delta(q_1, 0) = q_0$

$\vdash_A (q_0, 1)$ wegen $\delta(q_0, 0) = q_0$

$\vdash_A (q_1, \varepsilon)$ wegen $\delta(q_0, 1) = q_1$

Also: $(q_0, 1001) \vdash_A^* (q_1, \varepsilon)$

Oder: $(q_0, 1001) \vdash_A^* (q_0, 1)$

Endliche Automaten

Warum schon wieder $n, +, *$?

Wo ist das Monoid?

Definition 1.4 Sei M eine Menge (z.B. $M = Q \times \Sigma^*$). Die **Komposition** $R_1 \circ R_2$ zweier Relationen $R_1, R_2 \subseteq M \times M$ ist definiert durch

$$R_1 \circ R_2 = \{(x, y) \in M \times M \mid \\ \text{es existiert ein } z \in M \text{ mit } (x, z) \in R_1 \text{ und } (z, y) \in R_2\}$$

- Dann ist $(\wp(M \times M), \circ)$ ein Monoid.
- Neutrales Element ist die **Gleichheit** $E_M = \{(x, x) \mid x \in M\}$.
- Also können wir Potenzen R^n von Relationen $R \subseteq M \times M$ wie üblich definieren (insbesondere \vdash_A^n).
- Und weil auch Relationen Mengen sind, definieren wir analog zu Sprachen: $R^+ = \bigcup_{n>0} R^n$ und $R^* = \bigcup_{n \geq 0} R^n$ (insbesondere \vdash_A^+ und \vdash_A^*)

Endliche Automaten

Definition 1.5 Sei $A = (\Sigma, Q, s, F, \delta)$ ein DEA.

- Ein Wort $w \in \Sigma^*$ wird von A *akzeptiert* genau dann wenn es ein $q \in F$ gibt mit $(s, w) \vdash_A^* (q, \varepsilon)$.
- Die von A *akzeptierte* (oder: *erkannte*) *Sprache* $L(A)$ ist definiert durch

$$\begin{aligned} L(A) &= \{w \in \Sigma^* \mid w \text{ wird von } A \text{ akzeptiert}\} \\ &= \{w \in \Sigma^* \mid \text{es existiert ein } q \in F \text{ mit } (s, w) \vdash_A^* (q, \varepsilon)\} \end{aligned}$$

Beispiel

Für den oben definierten Automaten A gilt:

$$\begin{aligned} L(A) &= \{w \in \{0, 1\}^* \mid (q_0, w) \vdash^* (q_0, \varepsilon)\} \\ &= \{w \in \{0, 1\}^* \mid w \text{ endet auf } 0\} \cup \{\varepsilon\} \end{aligned}$$