Grundlagen

Alphabet Σ

Ein *Alphabet* ist endliche Menge beliebiger Elemente (normalerweise Buchstaben) und wird meist mit Σ bezeichnet.

Die Elemente eines Alphabets nennen wir *Symbole* oder *Zeichen*.

Wort w über Σ

Ein *Wort* (*Zeichenkette*, *String*) w über einem Alphabet Σ ist eine endliche Folge von Zeichen aus Σ.

Beachte: Üblicherweise unterscheiden wir nicht zwischen Wörtern, die nur ein Zeichen enthalten, und dem Zeichen selbst.

Das *leere Wort* bezeichnen wir mit ε.

Die Menge aller Wörter über einem Alphabet Σ wird mit Σ^* bezeichnet. Beachte, dass $\varepsilon \in \Sigma^*$.

Wörter können auch als Funktionen

$$w : \{1, \ldots, |w|\} \rightarrow \Sigma$$

betrachtet werden, wobei $|w|$ die Länge des Wortes ist (siehe unten). $w(i)$ bezeichnet dann das $i$te Zeichen im Wort w.

Wortlänge |w|

Wir bezeichnen mit $|w|$ die Länge des Wortes w.

Zum Beispiel:

$$|drei| = 4$$

Für das leere Wort gilt:

$$|\varepsilon| = 0$$

Konkatenation von Wörtern

Wenn u und v Wörter über dem gleichen Alphabet sind, dann bezeichnet $u \circ v$ das Wort, das durch Hintereinanderschreiben von u und v entsteht. Statt $u \circ v$ schreiben wir meist einfach uv.

$w = u \circ v$ gdw

(i) $|w| = |u| + |v|$

(ii) $w(j) = u(j)$, für $1 \leq j \leq |u|$

(iii) $w(|u| + j) = v(j)$, für $1 \leq j \leq |v|$

Beispiele:

$$wort \circ spiel = wortspiel$$

$$wort \circ \varepsilon \circ spiel = wortspiel$$

Wiederholung

Für $u \in \Sigma^*$ bezeichnet u^n die n-malige Wiederholung von u, also

$$(ab)^0 = \varepsilon$$

$$(ab)^1 = ab$$

$$(ab)^2 = abab \text{ etc.}$$

Für jedes Wort w und jede natürliche Zahl n ist w^n definiert als

$$\begin{aligned} w^0 &= \varepsilon \\ w^{n+1} &= w^n \circ w \end{aligned}$$

Zum Beispiel:

$$\begin{aligned} (ab)^2 &= (ab)^1 \circ ab \\ &= (ab)^0 \circ ab \circ ab \\ &= \varepsilon \circ ab \circ ab \\ &= abab \end{aligned}$$

Spiegelung

Definition w^R:

(i) Wenn $|w| = 0$: $w^R = w = \varepsilon$

(ii) Wenn $|w| > 0$: dann ist $w = ua$, wobei u ein Wort und a ein Zeichen ist, und $w^R = au^R$.

Beispiel:

$$\begin{aligned} (espen)^R &= n(espe)^R \\ &= ne(esp)^R \\ &= nep(es)^R \\ &= neps(e)^R \\ &= nepse(\varepsilon)^R \\ &= nepse \end{aligned}$$

Teilwort, Präfix, Suffix

v ist *Teilwort* eines Wortes w, wenn es Wörter x und y gibt, so dass $w = xvy$.

v ist *Präfix* eines Wortes w, wenn es ein Wort y gibt, so dass $w = vy$.

v ist *Suffix* eines Wortes w, wenn es ein Wort x gibt, so dass $w = xv$.

Sprache

Eine Sprache L über einem Alphabet Σ ist eine – endliche oder unendliche – Menge von Wörtern über Σ.

Kleene'sche Hülle (L^+, L^*)

$L^+ = \{ w_1 \circ \ldots \circ w_n :$
$n \geq 1 \text{ und } w_i \in L, \text{ für } 1 \leq i \leq n\}$

$L^* = L^+ \cup \{\varepsilon\}$

Konkatenation von Sprachen

$L_1 \circ L_2 = L_1L_2 = \{x \circ y : x \in L_1 \text{ und } y \in L_2\}$

Mathematische Induktion

Durch mathematische Induktion lassen sich Eigenschaften natürlicher Zahlen beweisen.

Nehmen wir an, wir wollen die Aussage

„für alle n gilt: $E(n)$"

beweisen.

Ein Beweis durch Mathematische Induktion besteht aus:

1. *Induktionsbasis:* Ein Beweis für $E(0)$.
2. *Induktionsannahme:* Die Annahme, dass $E(k)$ für $k \leq n$ gilt.
3. *Induktionsschritt:* Ein Beweis für $E(n + 1)$, mit Hilfe der Induktionsannahme.

Beispiel

Ein konkretes Beispiel (aus L&P, Seite 30–31) für einen Induktionsbeweis:

Wir beweisen für Wörter w und x, dass

$(wx)^R = x^R w^R$

Zum Beispiel: $(mark)^R = (rk)^R(ma)^R = kram$

Das Wichtigste bei einem Induktionsbeweis ist, die richtige Induktionsannahme zu formulieren!

Hier ist in dem zu beweisenden Satz keine natürliche Zahl vorhanden. Aber der Satz lässt sich durch *Induktion über die Wortlänge* beweisen, und zwar über die Wortlänge von x:

Induktionsannahme

Für alle Wörter wx mit $|x| \leq n$ gilt: $(wx)^R = x^R w^R$

Induktionsbasis

Wenn $|x| = 0$, dann ist x das leere Wort ($x = \varepsilon$) und wir erhalten: $(wx)^R = w^R = \varepsilon^R w^R = x^R w^R$

Induktionsschritt

Wenn $|x| = n + 1$, dann ist $x = ua$ für ein Wort u der Länge n und ein Zeichen a.

$$
\begin{aligned}
(wx)^R &= (w(ua))^R && [x = ua] \\
&= ((wu)a)^R && [\text{Assoziativität, s. Übungsblatt}] \\
&= a(wu)^R && [\text{Def. Spiegelung}] \\
&= a(u^R w^R) && [\text{Induktionsannahme}] \\
&= (au^R)w^R && [\text{Assoziativität}] \\
&= (ua)^R w^R && [\text{Def. Spiegelung}] \\
&= x^R w^R
\end{aligned}
$$

Im Induktionsschritt haben wir gezeigt: Wenn die Induktionsannahme für $|x| = n$ gilt, muss sie auch für $|x| = n + 1$ gelten.

Von der Induktionsbasis wissen wir, dass die Induktionsannahme für $|x| = 0$ gilt, also gilt sie auch für $|x| = 1, 2, 3, \ldots$. D.h.: sie gilt für *Wörter x von beliebiger Länge*.

Gilt sie auch für Wörter *w* von beliebiger Länge? Nun, der Beweis wurde so konstruiert, dass er für *beliebige w* gilt. Eine *Doppelinduktion* ist also in diesem Falle nicht erforderlich! (Jedoch in manch anderen Fällen!)

Reguläre Ausdrücke

Mit regulären Ausdrücken lassen sich einige unendliche Sprachen endlich beschreiben.

Definition der Menge der regulären Ausdrücke

(i) $\emptyset$ ist ein regulärer Ausdruck

(ii) Jedes Zeichen $\sigma \in \Sigma$ ist ein regulärer Ausdruck

(iii) Wenn α und β reguläre Ausdrücke sind, dann sind auch $\boldsymbol{(}\alpha\beta\boldsymbol{)}$, $\boldsymbol{(}\alpha\boldsymbol{\cup}\beta\boldsymbol{)}$ und $\alpha^{\boldsymbol{*}}$ reguläre Ausdrücke

(iv) Die Menge der regulären Ausdrücke ist durch die Formationsregeln (i) bis (iii) vollständig beschrieben.

Die in (iii) vorkommenden Zeichen **(,), ∪,** * haben bis jetzt *keine* Bedeutung – und deshalb auch mit Absicht etwas anders als (,), ∪ und * geschrieben. Sie sind reine syntaktische Objekte.

Reguläre Sprachen (reguläre Mengen)

Die von einem regulären Ausdruck α beschriebene Sprache $L(\alpha)$ ist gegeben durch:

(i) $L(\emptyset) = \emptyset$

(ii) $L(\sigma) = \{\sigma\}$ für jedes $\sigma \in \Sigma$

(iii) Wenn α und β reguläre Ausdrücke sind, dann ist

a) $L(\boldsymbol{(}\alpha\beta\boldsymbol{)}) = L(\alpha) \circ L(\beta) = L(\alpha)L(\beta)$

b) $L(\boldsymbol{(}\alpha\boldsymbol{\cup}\beta\boldsymbol{)}) = L(\alpha) \cup L(\beta)$

c) $L(\alpha^{\boldsymbol{*}}) = L(\alpha)^*$

Die Sprachen, die mit einem regulären Ausdruck beschreibbar sind, heißen *reguläre Sprachen* oder *reguläre Mengen*.

Zur Verbesserung der Lesbarkeit werden unnötige Klammern weggelassen: Statt

$$\boldsymbol{((}a \boldsymbol{\cup} b\boldsymbol{)} \boldsymbol{\cup} c\boldsymbol{)}^{\boldsymbol{*}}\boldsymbol{(}de\boldsymbol{))}$$

schreiben wir einfach

$$\boldsymbol{(}a \boldsymbol{\cup} b \boldsymbol{\cup} c\boldsymbol{)}^{\boldsymbol{*}}de.$$

Beispiele

1) Sei $\Sigma = \{a, b\}$.

$$\begin{aligned} L(\boldsymbol{((}a\boldsymbol{\cup}b\boldsymbol{)}^{\boldsymbol{*}}a\boldsymbol{))} &= L(\boldsymbol{(}a \boldsymbol{\cup} b\boldsymbol{)}^{\boldsymbol{*}})L(a) \\ &= L(\boldsymbol{(}a \boldsymbol{\cup} b\boldsymbol{)}^{\boldsymbol{*}})\{a\} \\ &= L(\boldsymbol{(}a \boldsymbol{\cup} b\boldsymbol{)})^*\{a\} \\ &= (L(a) \cup L(b))^*\{a\} \\ &= (\{a\} \cup \{b\})^*\{a\} \end{aligned}$$

$= \{a,b\}^*\{a\}$

$= \{w \in \{a,b\}^* : w$ endet mit $a\}$

2) Sei $\Sigma = \{a, b\}$.

$L((a \cup b)^*aa(a \cup b)^*) =$
$\{w \in \Sigma^* :$ in w kommen zwei a's nacheinander vor$\}$

3) Sei $\Sigma = \{0,1,\ldots,9,,\}$.

$L(0\cup((1\cup\ldots\cup9)(0\cup\ldots\cup9)^*)),(0\cup\ldots\cup9)(0\cup\ldots\cup9))$
beschreibt Zahlen mit zwei Dezimalstellen.

Vereinfachte Darstellung:

Da die beiden Schreibweisen

$L((a \cup b)^*aa(a \cup b)^*)$

und

$(\{a\} \cup \{b\})^*\{a\}\{a\}(\{a\} \cup \{b\})^*$

ziemlich umständlich sind, werden wir folgende abgekürzte Darstellungsweise benutzen:

$(a \cup b)^*aa(a \cup b)^*$

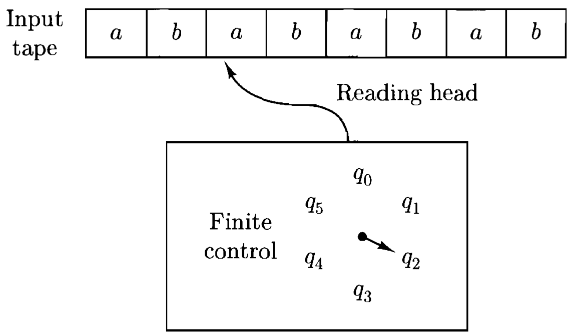
Deterministische Endliche Automaten (DEA)

Mit regulären Ausdrücken lässt sich jede reguläre Sprache auf endliche Weise beschreiben. Aber wie beantworten wir die Frage

„Ist $w \in L$?"

für eine reguläre Sprache L und ein beliebiges Wort w über Σ?

Ein *deterministischer endlicher Automat* (DEA) ist ein sehr einfaches Modell eines Computers: Er liest eine endliche Eingabe, befindet sich während des Lesens zu jeder Zeit in einem einer endlichen Zahl Zustände, und gibt schließlich „akzeptiert" oder „nicht akzeptiert" als *Ausgabe* aus.



Die Eingabe ist ein Wort einer Sprache und wird Zeichen für Zeichen, von links nach rechts, gelesen.

Der aktuelle Zustand und das Zeichen bestimmen zusammen den nächsten Zustand.

Komponenten eines DEA

Ein DEA ist ein Quadrupel $M = \langle K, \Sigma, \delta, s, F\rangle$ mit

- K – eine endliche Menge von *Zuständen*
- Σ – ein *Alphabet*
- s – der *Startzustand* ($s \in K$)
- F – eine Menge von *Endzuständen* ($F \subseteq K$)
- δ – eine Funktion von K und Σ in K.

Das Verhalten des Automaten ist zu jeder Zeit von zwei Faktoren abhängig:

- der aktuelle Zustand
- die noch zu lesenden Zeichen

Konfigurationen sind Paare $\langle q, w\rangle$ bestehend aus dem aktuellen Zustand $q \in K$ und der noch zu lesenden Eingabe $w \in \Sigma^*$ bilden. Konfigurationen bestimmen eindeutig den weiteren Verlauf der Berechnung.

Beispiel

Sei $M = \langle K, \Sigma, \delta, s, F\rangle$ mit

- $K = \{p, q\}$
- $\Sigma = \{a, b\}$
- $s = p$
- $F = \{p\}$
- $\delta(p, a) = p, \delta(q, a) = p, \delta(p, b) = q, \delta(q, b) = q$

Die Maschine M merkt sich in den Zuständen das zuletzt gelesene Zeichen:

- In p wurde a zuletzt gelesen (außer am Anfang der Berechnung)
- In q wurde b zuletzt gelesen.

Der einzige Endzustand ist der Startzustand, also der Zustand in dem entweder noch nichts gelesen wurde, oder zuletzt ein a gelesen wurde.

Eingabebeispiel

Die Startkonfiguration ist immer $\langle s, w\rangle$, wobei w die Eingabe ist. Bei Eingabe bba ist die Startkonfiguration also bei unserem Beispielautomaten M

$$\langle p, bba\rangle$$

Unter dem Lesekopf liegt ein b. Wegen $\delta(p, b) = q$ wechselt der Automat beim Einlesen des nächsten

Zeichens in den Zustand q, d.h. die nächste Konfiguration ist $\langle q, ba \rangle$. Wir schreiben:

$\langle p, bba \rangle \vdash_M \langle q, ba \rangle$

und sagen „$\langle p, bba \rangle$ ergibt $\langle q, ba \rangle$ in einen Schritt."

Es folgen zwei weitere Berechnungsschritte:

$\langle q, ba \rangle \vdash_M \langle q, a \rangle \vdash_M \langle p, \varepsilon \rangle$

Jetzt ist die gesamte Eingabe eingelesen und der Automat hält. Weil $p \in F$, wird die Eingabe „akzeptiert."

Die Relation $\vdash_M^*$

Die „ergibt-in-einem-Schritt"-Relation $\vdash_M$ wurde im Beispiel eingeführt. Formal sieht die Definition so aus:

$\langle q, w \rangle \vdash_M \langle q', w' \rangle$

gdw

$w = \sigma w'$ für ein $\sigma \in \Sigma$ und $\delta(q, \sigma) = q'$

$\vdash_M^*$ („ergibt") ist die reflexiv-transitive Hülle von $\vdash_M$. Auf Deutsch: $\langle q, w \rangle \vdash_M^* \langle q', w' \rangle$ heißt, dass sich die Konfiguration auf der rechten Seite in null oder mehreren Schritten aus der Konfiguration auf der linken Seite ergibt.

Akzeptanz, formal definiert

Die von einem DEA $M = \langle K, \Sigma, \delta, s, F \rangle$ akzeptierte Sprache $L(M)$ definieren wir folgendermaßen:

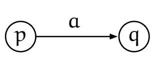
$L(M) = \{ w \in \Sigma^* :$ es gibt ein $q \in F$ so dass $\langle s, w \rangle \vdash_M^* \langle q, \varepsilon \rangle \}$

Auf Deutsch: $L(M)$ sind die Wörter, die akzeptiert werden, d.h., bei denen der Automat in einen Endzustand hält.

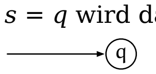
Zustandsdiagramme

Zur einfachen Darstellung endlicher Automaten werden *Zustandsdiagramme* benutzt. In den Zustandsdiagrammen werden Zustände durch kleine Kreise symbolisiert. δ, s und F werden folgendermaßen dargestellt:

- $\delta(p, a) = q$ wird dargestellt als:

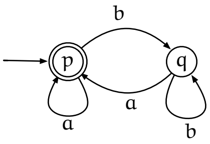


- $s = q$ wird dargestellt als:



- $q \in F$ wird dargestellt als:

Als Beispiel betrachten wir wieder M:



Oft werden die Zustandsnamen einfach weggelassen, weil sie bei einem mit einem Zustandsdiagramm dargestellten Automaten keine wichtige Rolle spielen.

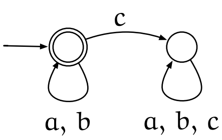
Toter Zustand

Ein *toter Zustand* ist ein nicht-Endzustand aus dem sich der Automat nicht mehr bewegen kann. Der Automat ist „gefangen."

Formal: p ist tot gdw $p \notin F$ und für alle $\sigma \in \Sigma$: $\delta(p, \sigma) = p$

Beispiel

$\Sigma = \{a, b, c\}$



$L(M) = \{w \in \Sigma^* : c$ kommt in w nicht vor$\}$

Nicht-deterministische endliche Automaten

Nicht-deterministische endliche Automaten (NEA) können in jedem Berechnungsschritt in einen von *null oder mehreren* Zuständen übergehen. Sie sind nicht deterministisch, weil man von der aktuellen Konfiguration nicht (immer) voraussagen kann, in welcher Konfiguration sich der Automat nach dem nächsten Berechnungsschritt befindet.

Ein Wort wird von einem nicht-deterministischen endlichen Automaten *akzeptiert*, wenn von allen möglichen Berechnungen mindestens eine zur Akzeptanz (im normalen Sinne) führt.

Wir können uns vorstellen, dass sich der Automat *selbst kopier*t, und dass jeder einzelne Automat die verschiedenen möglichen nächsten Zustände ausprobiert. Die Eingabe ist akzeptiert, wenn sich schließlich mindestens einer von diesen vielen Automaten in einem Endzustand mit leerer Eingabe befindet.

Bemerke, dass es bei NEAs auch „Sackgassen“ – Konfiguration, die keine neuen Konfigurationen ergeben – gibt. Bei DEAs dagegen ist die Funktion δ *total*.

Übergänge in denen mehrere Zeichen gelesen werden

Dies ist eine weitere Verallgemeinerung, die wir gleichzeitig mit dem Nichtdeterminismus einführen. Wir lassen zu, dass mehrere oder auch keine Zeichen in einem Schritt gelesen werden können.

Insgesamt haben wir:

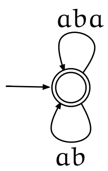
Ein nicht-deterministischer endlicher Automat (NEA) hat statt einer Übergangsfunktion eine *Übergangsrelation* Δ, die auf $K \times \Sigma^* \times K$ definiert ist.

Die sonstigen Komponenten sind wie bei einem DEA.

Bemerke: Als Sonderfall kann eine Relation auf $K \times \Sigma^* \times K$ auch eine totale Funktion $K \times \Sigma \to K$ sein. Also ist jeder DEA auch ein NEA.

Ein Beispiel für einen NEA

Ein Automat, der die Sprache $(ab \cup aba)^*$ akzeptiert:



Die „ergibt“-Relation für NEA

Konfigurationen sind wie bei einem DEA.

$(q, w) \vdash_M (q', w')$ gdw $w = uw'$ für ein $u \in \Sigma^*$ und $(q, u, q') \in \Delta$

Die „ergibt“-Relation (Notation: $\vdash_M^*$) ist wieder die reflexiv-transitive Hülle von $\vdash_M$

Bemerke: Die Definition von $\vdash_M$ und $\vdash_M^*$ für DEA ist ein Sonderfall der obigen Definition!

Die von einem NEA akzeptierte Sprache

Unsere Definition von $L(M)$ für einen DEA M ist auf einen NEA M direkt übertragbar:

$L(M) = \{\ w \in \Sigma^* :$ es gibt ein $q \in F$ so dass $\langle s, w\rangle \vdash^*_M \langle q, \varepsilon\rangle\ \}$

Lemma

Sei $M = \langle K, \Sigma, \Delta, s, F\rangle$ ein NEA, seien q, r Zustände, und seien x, y Wörter über Σ. Wenn es einen Zustand p gibt, so dass $\langle q, x\rangle \vdash^*_M \langle p, \varepsilon\rangle$ und $\langle p, y\rangle \vdash^*_M \langle r, \varepsilon\rangle$, dann $\langle q, xy\rangle \vdash^*_M \langle r, \varepsilon\rangle$

Beweis

$\langle q, x\rangle \vdash^*_M \langle p, \varepsilon\rangle$ bedeutet, dass es Zustände $q_0, \ldots, q_n$ und Wörter $x_0, \ldots, x_n$ für ein $n \geq 0$ gibt, so dass:

$$\begin{aligned}\langle q, x\rangle = \langle q_0, x_0\rangle &\vdash_M \langle q_1, x_1\rangle\\ &\vdash_M \ldots\\ &\vdash_M \langle q_i, x_i\rangle\\ &\vdash_M \langle q_{i+1}, x_{i+1}\rangle\\ &\vdash_M \ldots\\ &\vdash_M \langle q_n, x_n\rangle = \langle p, \varepsilon\rangle\end{aligned}$$

Laut Definition bedeutet $\langle q_i, x_i\rangle \vdash_M \langle q_{i+1}, x_{i+1}\rangle$, dass es ein Wort u_i gibt, so dass $x_i = u_i x_{i+1}$ und $\langle q_i, u_i, q_{i+1}\rangle \in \Delta$. Dann ist aber auch $x_i y = u_i x_{i+1} y$, und nach der Definition von $\vdash_M$ auch $\langle q_i, x_i y\rangle \vdash_M \langle q_{i+1}, x_{i+1} y\rangle$.

Deshalb gilt auch $\langle q, xy\rangle \vdash^*_M \langle p, y\rangle$.

Laut Annahme gilt $\langle p, y\rangle \vdash^*_M \langle r, \varepsilon\rangle$, deshalb muss, wie gezeigt werden sollte, auch $\langle q, xy\rangle \vdash^*_M \langle r, \varepsilon\rangle$ gelten (weil $\vdash^*_M$ transitiv ist).

Äquivalenz von DEA und NEA

Wie schon bemerkt, ist ein DEA auch immer ein NEA. Wir werden gleich feststellen, dass es zu jedem NEA auch einen äquivalenten DEA gibt.

Aber was heißt „äquivalent“?

Definition (äquivalent)

M und M' sind äquivalent gdw $L(M) = L(M')$

Theorem

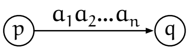
Zu jedem nicht-deterministischen endlichen Automaten gibt es einen äquivalenten deterministischen endlichen Automaten.

Beweis des Theorems

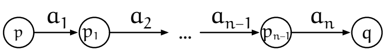
Wir fangen mit dem einfachsten Teil an:

Lemma 1: Sei $M = \langle K, \Sigma, \Delta, s, F\rangle$ ein NEA. Dann gibt es einen äquivalenten NEA M', der nur solche Übergänge $\langle p, u, q\rangle$ hat, in denen $|u| \leq 1$.

Beweis: Wir können von M einen erweiterten Automaten $M' = \langle K', \Sigma, \Delta', s, F\rangle$ folgendermaßen konstruieren: Jeder Übergang

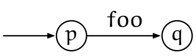


wird durch Übergänge

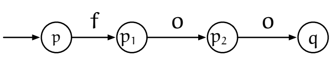


ersetzt, indem wir $n\text{-}1$ neue Zustände $p_1, \ldots, p_{n-1}$ einführen, und die Δ-Relation entsprechend erweitern.

Beispiel: Aus



bilden wir



Jetzt werden wir zu M' einen äquivalenten deterministischen Automaten $M'' = \langle K'', \Sigma, \delta'', s'', F''\rangle$ konstruieren.

Wir setzen:

$K'' = 2^{K'}$ (die Menge der Teilmengen von K')

Die Idee ist folgende: wenn sich M'' in einem Zustand $\{q_1, \ldots, q_n\}$ befindet, dann könnte sich M' gleich viel gelesener Eingabe in einem der Zustände $q_1, \ldots, q_n$ befinden (und umgekehrt).

Weiter setzen wir

$F'' = \{X \subseteq K' : X \cap F \neq \varnothing\}$

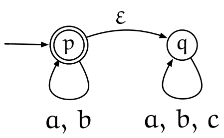
Bevor wir schließlich s'' und δ'' definieren, bereiten uns die ε-Übergänge die noch in M' vorhanden sind, noch einige Schwierigkeiten:

Definition (ε-Abschluss)

Für beliebige q setzen wir:

$E(q) = \{p \in K\colon \langle q, \varepsilon\rangle \vdash^*_{M'} \langle p, \varepsilon\rangle\}$

Beispiel: Im folgenden Automaten ist $E(p) = K$ und $E(q) = \{q\}$:



Jetzt setzen wir neue Startzustand ist die Menge der Zustände, die vom Ursprünglichen Startzustand ohne das Lesen der Eingabe erreichbar sind:

$s'' = E(s')$

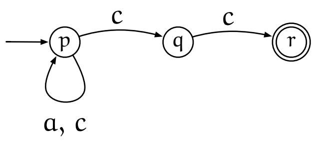
(der neue Startzustand ist die Menge der Zustände, die vom ursprünglichen Startzustand ohne das Lesen der Eingabe erreichbar sind)

und für jede Teilmenge Q von K' (= Element von K'') und Symbol σ:

$$\delta''(Q, \sigma) = \bigcup\{\, E(p) : \; p \in K' \text{ und für ein } q \in Q: \langle q, \sigma, p\rangle \in \Delta' \,\}$$

(der ursprüngliche NEA M' kann, wenn in q das Zeichen σ gelesen wird, in jeden solchen Zustand p übergehen, und kann auch ohne noch ein Zeichen zu lesen in einen der Zustände $E(p)$ übergehen)

Beispiel: Wir betrachten den folgenden einfachen Automaten M über dem Alphabet $\{a, c\}$:



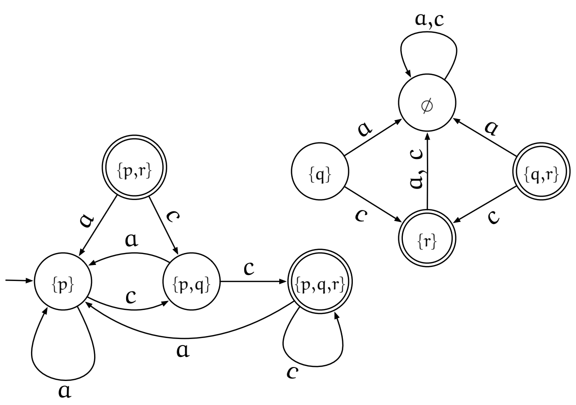
($L(M)$ ist die Menge der Wörter, die mit cc enden.)

Hier ist $M' = M$, und $E(p) = \{p\}$, $E(q) = \{q\}$ und $E(r) = \{r\}$.

$s'' = E(s) = \{p\}$

$K'' = \{\varnothing, \{p\}, \{q\}, \{r\}, \{p,q\}, \{p,r\}, \{q,r\}, \{p,q,r\}\}$.

M'' sieht dann folgendermaßen aus:



Die Zustände ∅, $\{q\}$, $\{r\}$, $\{q,r\}$ und $\{p,r\}$ erfüllen hier keine Funktion, da sie vom Startzustand aus nicht erreichbar sind.

Lemma 2: M'' ist deterministisch.

Beweis: Folgt direkt, da δ'' eine wohl definierte totale Funktion ist.

Jetzt sind wir bereit, das Theorem zu beweisen.

Wir werden folgendes durch Induktion über Wortlänge beweisen:

Lemma 3:

$\langle q, w\rangle \vdash^*_{M'} \langle p, \varepsilon\rangle$ gdw es gibt ein P, mit $p \in P$, so dass $\langle E(q), w\rangle \vdash^*_{M''} \langle P, \varepsilon\rangle$.

Bevor wir Lemma 3 beweisen, stellen wir schon fest, das dies mehr als ausreichend ist, um das Theorem zu beweisen:

$w \in L(M')$

gdw $\langle s, w\rangle \vdash^*_{M'} \langle f, \varepsilon\rangle$ für ein $f \in F$ [Defintion $L(M')$]

gdw $\langle E(s), w\rangle \vdash^*_{M''} \langle Q, \varepsilon\rangle$ für ein Q mit $f \in Q$ [Lemma 3]

gdw $\langle s'', w\rangle \vdash^*_{M''} \langle Q, \varepsilon\rangle$ für ein $Q \in F''$ [Definition M'']

gdw $w \in L(M'')$ [Definition $L(M'')$]

Beweis von Lemma 3

Beweis durch Induktion über die Wortlänge.

Induktionsannahme

Lemma 3 gilt für $|w| \leq n$.

Beachte: Statt Äquivalenz von M' und M'' direkt zu beweisen, haben wir eine etwas stärkere Induktionsannahme formuliert!

Induktionsbasis

$|w| = 0$, d.h. $w = \varepsilon$:

Wir setzen ε für w in Lemma 3 ein und erhalten:

$\langle q, \varepsilon\rangle \vdash^*_{M'} \langle p, \varepsilon\rangle$ gdw es gibt ein P mit $p \in P$, so dass $\langle E(q), \varepsilon\rangle \vdash^*_{M''} \langle P, \varepsilon\rangle$.

Die erste Aussage gilt gdw $p \in E(q)$.

M'' ist deterministisch, d.h. die zweite Aussage gilt gdw $P = E(q)$ und $p \in P$, d.h. gdw $p \in E(q)$.

Induktionsschritt

$|w| = n+1$, d.h. $w = va$, wobei a ein Symbol ist und v ein Wort mit $|v| = n$:

„⇒“

Wenn $\langle q, w\rangle \vdash^*_{M'} \langle p, \varepsilon\rangle$ gibt es Zustände r und t, wobei

$$\langle q, va\rangle \vdash^*_{M'} \langle r, a\rangle \vdash_{M'} \langle t, \varepsilon\rangle \vdash^*_{M'} \langle p, \varepsilon\rangle. \qquad (*)$$

Mit anderen Worten: Der Automat liest erst v, liest dann in einem Schritt a, und macht schließlich (vielleicht) einige ε-Schritte.

Es gilt dann:

$$\langle q, v\rangle \vdash^*_{M'} \langle r, \varepsilon\rangle$$

Weil $|v| \leq n$, können wir hier die Induktionsannahme anwenden, und erhalten:

Es gibt einen R, mit $r \in R$, so dass

$$\langle E(q), v\rangle \vdash^*_{M''} \langle R, \varepsilon\rangle.$$

(*) gibt uns auch, dass $\langle r, a, t\rangle \in \Delta'$ gilt, und dass $p \in E(t)$. Von der Konstruktion von δ'' wissen wir, dass $p \in \delta''(R, a)$, und deshalb dass $\langle R, a\rangle \vdash_{M''} \langle P, \varepsilon\rangle$ für einen P, mit $p \in P$, und dann auch $\langle E(q), va\rangle \vdash^*_{M''} \langle P, \varepsilon\rangle$, wie wir zeigen wollten.

„⇐“

Wenn $\langle E(q), va\rangle \vdash^*_{M''} \langle P, \varepsilon\rangle$, für ein P mit $p \in P$, gibt es einen Zustand R, mit $\delta''(R, a) = P$, so dass

$$\langle E(q), va\rangle \vdash^*_{M''} \langle R, a\rangle. \qquad (**)$$

Nach der Konstruktion von δ'' bedeutet $\delta''(R, a) = P$, dass es einen Zustand t von M' gibt, mit $p \in E(t)$, und wo für irgendeinen $r \in R$: $\langle r, a, t\rangle \in \Delta'$.

Wenn (**) gilt, gilt auch $\langle E(q), v\rangle \vdash^*_{M''} \langle R, \varepsilon\rangle$ und wenden wir hier die Induktionsannahme an ($|v| \leq n$), bekommen wir:

$$\langle q, v\rangle \vdash^*_{M'} \langle r, \varepsilon\rangle \qquad [r \in R]$$

und deshalb auch

$$\langle q, va\rangle \vdash^*_{M'} \langle t, \varepsilon\rangle \qquad [\langle r, a, t\rangle \in \Delta']$$

und schließlich

$$\langle q, va\rangle \vdash^*_{M'} \langle p, \varepsilon\rangle \qquad [p \in E(t)]$$

wie wir zeigen wollten.

Damit ist Lemma 3, und dadurch das Theorem, bewiesen. □

Endliche Automaten und reguläre Sprachen

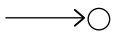
Theorem

Für jede reguläre Sprache L gibt es einen endlichen Automaten M so dass $L = L(M)$.

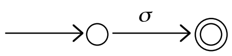
Beweis

Wenn L regulär ist, gibt es einen regulären Ausdruck α mit $L(\alpha) = L$. Wir konstruieren einen NEA M durch Induktion über α, d.h. wir werden für jedes α einen Teilautomaten $M(\alpha)$ konstruieren, wo $L(M(\alpha)) = L(\alpha)$. Jeder Automat in der Konstruktion benutzt das Alphabet Σ der Sprache L (die übrigen formalen Details der Konstruktion können von den Zeichnungen leicht abgeleitet werden):

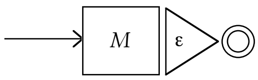
1) $\alpha = \varnothing$: Dann sieht $M(\alpha)$ so aus:



2) $\alpha = \sigma \in \Sigma$: Dann sieht $M(\alpha)$ so aus:

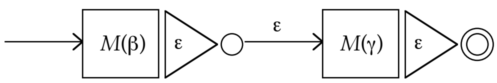


Wir werden jetzt Automaten schematisch zusammenbauen, indem

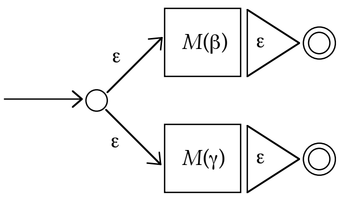


den Automaten darstellt, der wie M ist, außer dass die Endzustände durch ε-Übergänge mit einem einzelnen neuen Endzustand ersetzt sind.

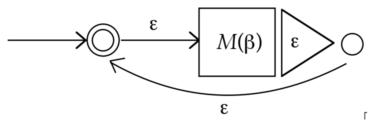
3) $\alpha = (\beta\gamma)$: Dann sieht $M(\alpha)$ so aus:



4) $\alpha = (\beta \cup \gamma)$: Dann sieht $M(\alpha)$ so aus:



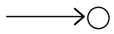
5) $\alpha = \beta^*$: Dann sieht $M(\alpha)$ so aus:



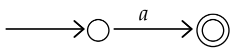
Ein Beispiel

Ein Automat der $L((a^* \cup \emptyset))$ akzeptiert:

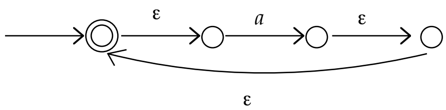
Der Automat für $\emptyset$:



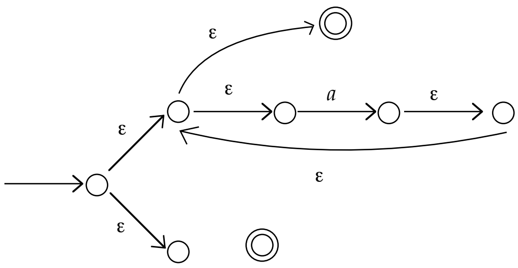
für a:



für a^*:



und schließlich für $(a^* \cup \emptyset)$:



Theorem

Jede Sprache, die von einem endlichen Automaten akzeptiert wird, ist regulär.

Beweis

Sei M ein endlicher Automat. Weil wir schon bewiesen haben, dass es zu jedem NEA einen äquivalenten DEA gibt, können wir annehmen, dass M deterministisch ist. Durch Konstruktion werden wir Zeigen, dass es eine reguläre Sprache R gibt, mit $R = L(M)$.

Erstens müssen die Zustände von M nummeriert werden:

$$K = \{q_1, \dots, q_n\}$$

wobei $q_1 = s$.

Wir werden R aus einer Menge von einfacheren Sprachen definieren: Wir definieren $R(i, j, k)$ als die

Wörter, die gelesen werden, wenn M sich vom Zustand q_i bis Zustand q_j bewegt, ohne sich zwischen q_i und q_j durch einen Zustand q_m, wo $m \geq k$, zu bewegen (i und j dürfen aber $\geq k$ sein!).

Dies hat einen Sinn, weil

$L(M) = \cup\{R(1, j, n+1) : q_j \in F\}$

(Gute Frage: Wozu sind jetzt i und k in $R(i, j, k)$ da? Um eine stärkere Induktionsannahme machen zu können!)

Formal definieren wir:

Für $1 \leq i \leq n$, $1 \leq j \leq n$, $1 \leq k \leq n+1$:

$R(i, j, k) = \{w \in \Sigma^* : \langle q_i, w\rangle \vdash_M{}^* \langle q_j, \varepsilon\rangle$ und für jedes $x \in \Sigma^*$ und jedes m: Wenn $\langle q_i, w\rangle \vdash^*_M \langle q_m, x\rangle$, dann ist entweder $m<k$ oder $x=\varepsilon$ und $m=j$ oder $x=w$ und $m=i$ $\}$

Wir werden jetzt beweisen, dass jedes $R(i, j, k)$ regulär ist. Es folgt dann, dass R – als Vereinigung endlich vieler $R(i, j, k)$ – auch regulär ist. Der Beweis wird durch Induktion über k durchgeführt:

Induktionsannahme

Für $m \leq k$ ist $R(i, j, m)$ regulär.

Induktionsbasis

Wir zeigen, dass $R(i, j, 1)$ regulär ist:

i) Wenn $i = j$ ist
   $R(i, j, 1) = \{\varepsilon\} \cup \{\sigma \in \Sigma : \delta(q_i, \sigma) = q_j\}$.

ii) Wenn $i \neq j$ ist
   $R(i, j, 1) = \{\sigma \in \Sigma : \delta(q_i, \sigma) = q_j\}$

In beiden Fällen ist $R(i, j, 1)$ endlich, und damit regulär.

Induktionsschritt

Wir stellen fest, dass

$R(i, j, k+1) =$
$\quad R(i, j, k) \cup R(i, k, k)R(k, k, k)^*R(k, j, k)$

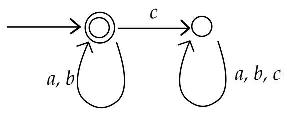
Zu Deutsch: Wenn der Automat sich von q_i nach q_j bewegt, und sich dazwischen nie durch einen Zustand q_m, wo $m > k$, bewegt, bewegt er sich entweder auch nicht durch q_k, oder er bewegt sich von q_i nach q_k ohne q_k zu passieren, bewegt sich dann beliebig viel Mal von q_k nach q_k, und bewegt sich schließlich nach q_j, ohne wieder durch q_k zu kommen.

Die Induktionsannahme sagt, dass jede der Sprachen auf der rechten Seite regulär ist. Und die regulären Sprachen sind unter den Operationen von

Union, Konkatenation und Kleene'scher Hüllenbildung abgeschlossen. □

Ein Beispiel

Wir betrachten wieder den deterministischen Automaten



der als Alphabet $\{a, b, c\}$ hat, und der die Sprache $\{a, b\}^*$ akzeptiert.

Nummeriere die Zustände von links nach rechts. Es gilt dann:

$L(M) = R(1, 1, 3)$

$R(1, 1, 3) = R(1, 1, 2) \cup R(1, 2, 2)R(2, 2, 2)^*R(2, 1, 2)$

$R(1, 1, 2) = R(1, 1, 1) \cup R(1, 1, 1)R(1, 1, 1)^*R(1, 1, 1)$

$R(2, 1, 2) = R(2, 1, 1) \cup R(2, 1, 1)R(1, 1, 1)^*R(1, 1, 1)$

$R(1, 1, 1) = a \cup b \cup \varepsilon$

$R(2, 1, 1) = \varnothing$

und dann:

$$\begin{aligned}
&R(1, 1, 2)\\
&= a \cup b \cup \varepsilon \cup (a \cup b \cup \varepsilon)(a \cup b \cup \varepsilon)^*(a \cup b \cup \varepsilon)\\
&= (a \cup b \cup \varepsilon)(a \cup b \cup \varepsilon)^*\\
&= (a \cup b)^*
\end{aligned}$$

$R(2, 1, 2) = \varnothing \cup \varnothing R(1, 1, 1)^*R(1, 1, 1) = \varnothing$

Mehr brauchen wir dann gar nicht zu berechnen:

$$\begin{aligned}
R(1, 1, 3) &= (a \cup b)^* \cup R(1, 2, 2)R(2, 2, 2)^*\varnothing\\
&= (a \cup b)^*
\end{aligned}$$

Abschlusseigenschaften der Klasse der regulären Sprachen

Theorem

Die regulären Sprachen sind abgeschlossen unter

a) Vereinigung

b) Konkatenation

c) Kleene'scher Hüllenbildung

d) Komplementbildung

e) Schnittbildung

Beweis

Für a) bis c) – wissen wir schon.

Für d) – Sei L eine reguläre Sprache. Dann gibt es einen DEA $M = \langle K, \Sigma, \delta, s, F\rangle$ der L akzeptiert. Die Komplementsprache $\Sigma^* - L$ wird dann von $M' = \langle K, \Sigma, \delta, s, K{-}F\rangle$ akzeptiert.

Für e) – Seien L_1 und L_2 reguläre Sprachen. Es gilt:

$L_1 \cap L_2 = \Sigma^* - ((\Sigma^* - L_1) \cup (\Sigma^* - L_2))$

(L_1 und L_2 können eventuell verschiedene Alphabete Σ_1 und Σ_2 haben, bilde dann $\Sigma = \Sigma_1 \cup \Sigma_2$) □

Entscheidbare Probleme bei endlichen Automaten (regulären Sprachen)

Theorem

Es gibt Entscheidungsalgorithmen für die folgenden Probleme:

(M, M_1, M_2 sind endliche Automaten, $w \in \Sigma^*$)

a) Ist $w \in L(M)$?

b) Ist $L(M) = \emptyset$?

c) Ist $L(M) = \Sigma^*$?

d) Ist $L(M_1) \subseteq L(M_2)$?

e) Ist $L(M_1) = L(M_2)$?

Beweis

Wir dürfen annehmen, dass die Automaten deterministisch sind.

a) Einfach dem Automaten die Eingabe w geben. Nach $|w|$ Berechnungsschritten erfolgt die Antwort.

b) Äquivalent zu der Frage: Ist mindestens ein Endzustand vom Startzustand erreichbar? Diese Frage lässt sich mit Hilfe des (endlichen!) Zustandsdiagrammes beantworten.

c) Äquivalent zu der Frage: Ist $\Sigma^* - L(M) = \varnothing$?

d) Äquivalent zu: Ist $(\Sigma^* - L(M_2)) \cap L(M_1) = \varnothing$?

e) Äquivalent zu: Ist $L(M_1) \subseteq L(M_2)$ und $L(M_2) \subseteq L(M_1)$?

Das Pumping-Lemma

Mit den Abschlusseigenschaften lässt sich positiv feststellen, dass eine Sprache regulär ist. Zum Beispiel ist die Sprache $\{w \in \{a, b\}^*$: die Länge von w ist nicht teilbar durch 3, die Anzahl der a's dagegen ist teilbar durch $3\}$ regulär.

Mit dem Pumping-Lemma lässt sich auch zeigen, dass Sprachen nicht regulär sind:

Theorem (Pumping-Lemma)

Sei L eine unendliche reguläre Sprache. Dann gibt es Wörter x, y und z, wobei $y \neq \varepsilon$ und für jedes $n \geq 0$ gilt, dass $xy^nz \in L$.

Vor dem Beweis erst ein Beispiel der Anwendung:

Theorem

$\{a^nb^n : n \geq 0\}$ ist nicht regulär.

Beweis: Wenn L regulär wäre, gäbe es nach dem Pumping-Lemma x, y und z, wobei $y \neq \varepsilon$ und für jedes $n \geq 0$: $xy^nz \in L$.

Wir können 3 Fälle unterscheiden:

1) $y = a^m$ für ein $m > 0$. In xyz sind dann die ganzen b's innerhalb z, und xy^2z hat dann m mehr a's als b's. Kontradiktion!
2) $y = b^m$ für ein $m > 0$ ist dann auch unmöglich.
3) y hat a's und b's. Dann kommen in y^2 b's vor a's vor – Kontradiktion!

Wir werden die vorgestellte Version des Pumping-Lemmas nicht direkt beweisen, sondern eine stärkere Version davon, die in Beweisen leichter einsetzbar ist:

Theorem

Sei $M = \langle K, \Sigma, \delta, s, F\rangle$ ein DEA, und sei w ein beliebiges Wort in $L(M)$ mit $|w| \geq |K|$. Dann gibt es Wörter x, y, z so dass $w = xyz$, $|xy| \leq |K|$, $y \neq \varepsilon$ und für jedes $n \geq 0$: $xy^nz \in L(M)$.

Beweis

Sei $l = |w|$. Dann ist $w = \sigma_1 \ldots \sigma_l$, wobei $\sigma_1, \ldots, \sigma_l \in \Sigma$. Weil M deterministisch ist, liest M das Wort w in l Schritten, d.h., es gibt Zustände $q_0, \ldots, q_l$, wobei $q_0 = s$ und $q_l \in F$, so dass:

$$\begin{aligned}\langle q_0, \sigma_1\ldots\sigma_l\rangle &\vdash_M \langle q_1, \sigma_2\ldots\sigma_l\rangle\\ &\vdash_M \ldots\\ &\vdash_M \langle q_i, \sigma_{i+1}\ldots\sigma_l\rangle\\ &\vdash_M \ldots\end{aligned}$$

$\vdash_M \langle q_j, \sigma_{j+1} \ldots \sigma_l \rangle$

$\vdash_M \ldots$

$\vdash_M \langle q_{l-1}, \sigma_l \rangle$

$\vdash_M \langle q_l, \varepsilon \rangle$

Nun ist $l = |w| \geq |K| = k$. Also müssen mindestens zwei der $k+1$ Zustände $q_0, \ldots, q_k$ gleich sein! Es gibt also i, j, wobei $0 \leq i < j \leq k$, so dass $q_i = q_j$. Das heißt, dass der Automat während er das Teilwort $\sigma_{i+1} \ldots \sigma_j$ liest, sich von q_i und zurück nach q_i bewegt. Das heißt wiederum, dass das Wort

$\sigma_1 \ldots \sigma_i (\sigma_{i+1} \ldots \sigma_j)^n \sigma_{j+1} \ldots \sigma_l$

für alle n akzeptiert wird.

Jetzt setzen wir $x = \sigma_1 \ldots \sigma_i$, $y = \sigma_{i+1} \ldots \sigma_j$ und $z = \sigma_{j+1} \ldots \sigma_l$. Dann ist auch $|xy| = j \leq k$. □

Beispiel

Ein Palindrom ist ein Wort (in diesem Zusammenhang nicht unbedingt ein echtes Wort), das mit seiner eigenen Spiegelung identisch ist. Beispiele: *abba*, *renner*, *saippuakauppias* (finnisch für Seifenverkäufer), *qcßcq*.

Formal:

$PAL = \{w \in \{a, \ldots, z, ä, ö, ü, ß\}^* \mid w^R = w\}$.

PAL ist keine reguläre Sprache, und dies lässt sich mit der stärkeren Version des Pumping-Lemmas zeigen:

Angenommen, *PAL* sei regulär. Sei M ein DEA, der *PAL* akzeptiert, und sei k die Zahl der Zustände in M. Betrachte das Wort $w = a^k b a^k \in PAL$. Das Theorem sagt uns, dass es Wörter x, y und z gibt, wobei $w = xyz$, $y \neq \varepsilon$ und $|xy| \leq k$, und wo $xy^n z \in PAL$ für jedes $n \geq 0$. Dann ist aber $y = a^m$, wobei $1 \leq m \leq k$, und wir stellen fest, dass $xy^2 z$ zu viele a's vor dem b hat – Kontradiktion!

Deterministischer endlicher Transducer (DET)

Ein deterministischer endlicher Transducer ist wie ein DEA, hat aber keine Endzustände[1], dafür hat er eine Ausgabe. Formal:

$M = \langle K, \Sigma, \delta, s\rangle$,

wobei δ eine Funktion von $K \times \Sigma$ in $K \times \Sigma^*$ ist. Der DET schreibt also in jedem Schritt *0, 1* oder mehrere Zeichen.

$\langle p, u, v\rangle$ bedeutet, dass der Transducer sich im Zustand p befindet, Eingabe u noch zu lesen hat, und Ausgabe v schon geschrieben hat.

Die *ergibt*-Relation

„ergibt in einem Schritt“:

$\langle q, u, v\rangle \vdash_M \langle q', u', v'\rangle$ gdw es gibt $\sigma \in \Sigma$ und $w \in \Sigma^*$ so dass

1. $u = \sigma u'$
2. $\delta(q, \sigma) = \langle q', w\rangle$
3. $v' = vw$

$\vdash_M^*$ (lese: ergibt) ist wieder die reflexiv-transitive Hülle von $\vdash_M$

Die Ausgabe eines DET

Da ein DET (nach unserer Definition) keine Endzustände hat, gibt es keinen Akzeptanz-Begriff.

Dafür haben wir folgende Definition:

M erzeugt Ausgabe u bei Eingabe w gdw es gibt ein $q \in K$ so dass $\langle s, w, \varepsilon\rangle \vdash_M^* \langle q, \varepsilon, u\rangle$

Sei f eine Zeichenketten-Funktion, $f\colon \Sigma^* \to \Sigma^*$:

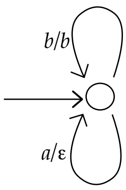
M berechnet f gdw für jedes $w \in \Sigma^*$ gibt es $q \in K$ so dass $\langle s, w, \varepsilon\rangle \vdash_M^* \langle q, \varepsilon, f(w)\rangle$

Zustandsdiagramme

Nur die Übergänge werden in den Zustandsdiagrammen eines DET anders gezeichnet als die eines DEA. „*σ/u*“ bedeutet, dass σ gelesen und u geschrieben wird.

1 Oft werden auch DETs mit Endzuständen definiert

Beispiele

1. 

– entfernt jedes a in der Eingabe.

2. Sei M ein DEA. Wir können M mit einem DET folgendermaßen simulieren: Wir erweitern das Alphabet um drei neue Symbole: *$*, Y („yes“) und N („no“). Einen M-Übergang in dem σ gelesen wird, ersetzen wir durch einen Übergang, in dem σ gelesen und ε geschrieben wird. Zusätzlich führen wir für jeden nicht-Endzustand einen Übergang *$/N* ein, und für jeden Endzustand einen Übergang *$/Y*.
M mit Eingabe w wird vom DET mit Eingabe *w$* folgendermaßen simuliert: Würde M w akzeptieren, erzeugt der DET die Ausgabe Y, sonst erzeugt er die Ausgabe N.