## 3. Reguläre Sprachen

Bisher wurden Automaten behandelt und Äquivalenzen zwischen den verschiedenen Automaten gezeigt. DEAs erkennen formale Sprachen. Gibt es formale Sprachen, die nicht erkannt werden? Wenn ja, welche? Wir werden sehen, dass genau die sogenannten regulären Sprachen erkannt werden.

### *3.1 Reguläre Ausdrücke*

Reguläre Sprachen lassen sich über reguläre Ausdrücke definieren. Wir bezeichnen die Sprache, die durch einen Ausdruck E repräsentiert wird, mit L(E).

Def.: Die Menge der regulären Ausdrücke (über dem Alphabet Σ) ist die kleinste Menge, die die folgenden Bedingungen erfüllt:

| | Ausdruck | Sprache |
|---|---|---|
| 1. | ε und ∅ sind reguläre Ausdrücke. | $L(\varepsilon) = \{\varepsilon\}$ und $L(\emptyset) = \emptyset$. |
| 2. | $a \in \Sigma$ ist regulärer Ausdruck. | $L(a) = \{a\}$ |
| Wenn E und F reguläre Ausdrücke sind, dann | | |
| 3. | (E+F) und (EF) sind reg. Ausdrücke. | $L((E+F)) = L(E) \cup L(F)$<br>$L((EF)) = L(E)L(F)$ |
| 4. | E* ist regulärer Ausdruck. | $L(E^*) = (L(E))^*$ |

Bindungsstärken: * > Konkatenation > +

Beispiele über Alphabet {0,1}:
(01)*, ((01)* + (00)*), ((11)1)

äußere Klammern können entfallen, ebenso solche, die wegen Assoziativität von + und Konkatenation nicht notwendig sind:

$(E_1+E_2)+E_3 = E_1+(E_2+E_3)$ deshalb einfach $E_1+E_2+E_3$
$(E_1E_2)E_3 = E_1(E_2E_3)$ deshalb einfach $E_1E_2E_3$

aber: $(01)^* \neq 01^*$ * bindet stärker
$(01)+0 \neq 0(1+0)$ {01,0} versus {01,00} Wichtig: $\emptyset S = S\emptyset = \emptyset$

Beispiel: gesucht regulärer Ausdruck für die Sprache der Wörter über Σ={a,b,c}, die mindestens ein a und ein b enthalten:

$$((a+b+c)^*a(a+b+c)^*b(a+b+c)^*) + ((a+b+c)^*b(a+b+c)^*a(a+b+c)^*)$$

(a+b+c) abgekürzt durch Σ also $\Sigma^*a\Sigma^*b\Sigma^* + \Sigma^*b\Sigma^*a\Sigma^*$

Def.: Sprachen, die durch reguläre Ausdrücke beschrieben werden können, heißen regulär.

### *3.2 Vereinfachungsregeln für reguläre Ausdrücke*

Bei der Beschreibung von regulären Sprachen können komplexe Ausdrücke entstehen, die sich vereinfachen lassen.

Def.: Zwei reguläre Ausdrücke $E_1$ und $E_2$ heißen äquivalent (Notation: $E_1 = E_2$) wenn gilt: $L(E_1) = L(E_2)$.

Satz: Seien R, S, T reguläre Ausdrücke. Es gelten folgende Äquivalenzen:

| | |
|---|---|
| $R + S = S + R$ | Kommutativität |
| $(R + S) + T = R + (S + T)$ | Assoziativität + |
| $(R\ S)\ T = R\ (S\ T)$ | Assoziativität Konkatenation |
| $\varnothing + R = R + \varnothing = R$ | Einheit + |
| $\varepsilon R = R\varepsilon = R$ | Einheit Konkatenation |
| $\varnothing R = R\varnothing = \varnothing$ | Nulloperator Konkatenation |
| $R\ (S + T) = RS + RT$ | Distributivität |
| $(S + T)\ R = SR + TR$ | |
| $R + R = R$ | Idempotenz |
| $(R^*)^* = R^*$ | |
| $(\varepsilon + R)^* = R^*$ | |
| $\varnothing^* = \varepsilon$ | |
| $\varepsilon^* = \varepsilon$ | |
| $(\varepsilon + R)R^* = R^*\ (\varepsilon + R) = R^*$ | |
| $RR^* = R^*R$ | |
| $R^* + R = R^*$ | |
| $\varepsilon + RR^* = R^*$ | |

Beispiel 1: Beweis für $R\ (S + T) = RS + RT$

| | |
|---|---|
| $w \in L(R(S+T))$ | $w = uv$ mit $u \in L(R)$ und $[v \in L(S)$ oder $v \in L(T)]$ |
| | $w = uv$ mit $[u \in L(R)$ und $v \in L(S)]$ oder $[u \in L(R)$ und $v \in L(T)]$ |
| | $w \in L(RS+RT)$ |

Bemerkung:
Obige Gleichungen gelten für beliebige reguläre Ausdrücke R, S, T, d.h. wir können beliebige konkrete reguläre Ausdrücke über dem Alphabet für die (Meta-) Variablen R, S, T einsetzen. Man kann zeigen, dass es für den Nachweis der Gültigkeit einer Gleichung genügt, die Variablen selbst als Zeichen des Alphabets aufzufassen (oder durch Zeichen eines Alphabets zu ersetzen) und die Gleichheit/Ungleichheit der dadurch beschriebenen Sprachen zu zeigen.

Beispiel 2: um nachzuprüfen, ob für alle R, S gilt: $(R+S)^* = R^* + S^*$ genügt es zu prüfen, ob $(r+s)^* = r^* + s^*$, wobei r und s Symbole aus einem Alphabet sind. In diesem Fall:

$L((r+s)^*) = \{\varepsilon, r, s, rr, rs, sr, ss, rrr, rrs, rsr, rss, srr, srs, ssr, sss, ...\}$
$L(r^*+s^*) = \{\varepsilon, r, rr, rrr, ...\} \cup \{\varepsilon, s, ss, sss, ...\}$

rs in der ersten, aber nicht in der zweiten Sprache, deshalb gilt obige Gleichung nicht.

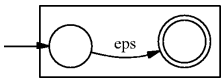
Beispiel 1 entsprechend: $L(r(s+t)) = \{rs, rt\} = L(rs+rt)$

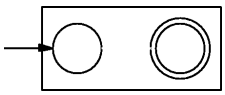
*3.3 Reguläre Sprachen und endliche Automaten*

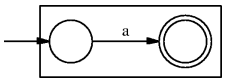
Satz: Sei L eine reguläre Sprache. Es gibt einen endlichen Automaten, der L akzeptiert.

Beweis durch Induktion über den Aufbau des regulären Ausdrucks E, der L repräsentiert.

Induktionsanfang:

E = ε Automat: 


E = ∅ Automat: 

E = a Automat: 


Induktionsschritt: Seien $E_1$ und $E_2$ reguläre Ausdrücke, für die endliche Automaten $M_1$ und $M_2$ konstruiert werden können (Induktionsvoraussetzung). Wir haben bereits im Kapitel 2 gezeigt, wie aus $M_1$ und $M_2$ endliche Automaten für Vereinigung, Konkatenation und Kleene-Abschluss von Sprachen erzeugt werden können. Damit existieren also auch Automaten für die regulären Ausdrücke (E+F), (EF) und E*.

q.e.d.

Satz: Sei L die von einem DEA A akzeptierte Sprache. L ist reguläre Sprache.

Beweis: Wir konstruieren einen regulären Ausdruck R, der L(A) repräsentiert.
Ohne Beschränkung der Allgemeinheit seien im folgenden die Zustände von A $Q=\{q_1,...,q_n\}$, $q_1$ der Startzustand, $F=\{q_{f1},...,q_{fs}\}$ die Menge der Finalzustände. Weiter definieren wir für $i,j \in \{1,...,n\}$ und $k \in \{0,...,n\}$reguläre Ausdrücke $R_{i,j}^{k}$, so dass gilt::

$L(R_{i,j}^{k})=\{x \in \Sigma^* \mid \delta^*(q_i,x) = q_j$, alle zwischen $q_i$ u. $q_j$ durchlaufenen Zustände haben Index $\leq k\}$

Wir zeigen durch Induktion über k, dass sich all diese Sprachen durch reguläre Ausdrücke beschreiben lassen.
Induktionsanfang (k=0)
Für $i \neq j$ gilt: $R_{i,j}^{0} = a_1+\ldots+a_m$ , wobei $\{a_1,\ldots,a_m\}=\{a \in \Sigma \mid \delta(q_i,a) = q_j\}$
$R_{i,j}^{0}=\varnothing$ , falls m=0, d.h. $q_j$ kein direkter Nachfolger von $q_i$
Für i = j gilt: $R_{i,i}^{0} = \varepsilon+a_1+\ldots+a_m$ , wobei $\{a_1,\ldots,a_m\}=\{a \in \Sigma \mid \delta(q_i,a) = q_i\}$

Induktionsschritt: (Induktionsvoraussetzung: es existieren alle Ausdrücke $R_{i,j}^{k}$)

$$R_{i,j}^{k+1} = R_{i,j}^{k} + R_{i,k+1}^{k} (R_{k+1,k+1}^{k})^* R_{k+1,j}^{k}$$

Der reguläre Ausdruck R für L(A) ist $R=R_{1,f1}^{n} + \ldots + R_{1,fs}^{n}$. Es gilt L(A)=L(R).

qed.

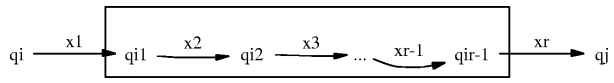
Ausführliche Erläuterung des Beweises:

Sei A ein Automat, gegeben durch $Q=\{q_1,...,q_n\}$, Startzustand $q_1$, Finalzustände $F=\{q_{f1},...,q_{fs}\}$. Weiter seinen $i,j \in \{1,...,n\}$ und $k \in \{0,...,n\}$.
Wir konstruieren Sprachen, die uns von einem Zustand $q_i$ in einen Zustand $q_j$ bringen. Wenn es gelingt, diese Sprachen durch reguläre Ausdrücke zu charakterisieren, folgt, dass die Sprache des Automaten regulär ist. Die Sprache des Automaten ist dann nämlich die Vereinigung der Sprachen, die uns vom Startzustand in die verschiedenen Finalzustände bringen.

$R_{i,j}$ wird den regulären Ausdruck bezeichnen, dessen Instanzen den Automaten vom Zustand $q_i$ in den Zustand $q_j$ überführen, d.h. wenn $w \in L(R_{i,j})$, dann $\delta^*(q_i,w)=q_j$.

Da aber eventuell verschiedene Pfade von $q_i$ nach $q_j$ existieren und wir nicht alle auf einmal betrachten können, schränken die möglichen Wege durch den Automaten zunächst ein und erweitern sie dann, bis keine Einschränkungen mehr vorliegen. Diese Einschränkung sei durch den Parameter k ($\in \{0,...,n\}$) folgendermaßen definiert.

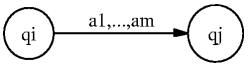
$L(R_{i,j}^k)=\{x \in \Sigma^* \mid \delta^*(q_i,w)=q_j$ und alle zwischen $q_i$ und $q_j$ durchlaufenen Zustände haben einen Index $\leq k\}$
D.h. $x \in L(R_{i,j}^k)$ falls $x=x_1...x_r$ und die Zustandsübergänge sehen folgendermaßen aus, wobei $i1,...,ir-1 \leq k$:



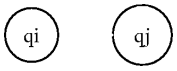

Falls $i>k$, darf also $q_i$ nicht mehr in dem umrandeten Kasten vorkommen, analog für j. Für $k=0$ müsste also ein direkter Übergang (ohne einen Zwischenzustand) von $q_i$ nach $q_j$ existieren, da jeder Zustand einen Index>0 hat.

Der Beweis erfolgt dann per Induktion über k, beginnend mit 0. Dabei werden reguläre Ausdrücke definiert, die einen direkten Übergang zwischen zwei Zuständen beschreiben.

Sei also k=0:
Für $i \neq j$ gilt:



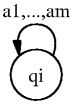

$R_{i,j}^0 = a_1+\ldots+a_m$ , wobei $\{a_1,\ldots,a_m\}=\{a \in \Sigma \mid \delta(q_i,a) = q_j\}$




$R_{i,j}^0=\varnothing$ , falls m=0, d.h. $q_j$ ist kein direkter Nachfolger von $q_i$

Für $i = j$ gilt:

Beachte – prinzipiell gilt $\delta(q,\varepsilon)=q$

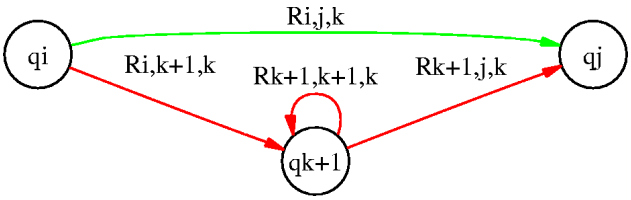



$R_{i,i}^0 = \varepsilon+a_1+\ldots+a_m$ , wobei $\{a_1,\ldots,a_m\}=\{a \in \Sigma \mid \delta(q_i,a) = q_i\}$
Dabei kann m wie oben 0 sein (dann ist $R_{i,i}^0 = \varepsilon$).

Es sollte einleuchtend sein, dass die gegebenen Ausdrücke, die richtige Sprachen beschreiben. Wenn z.B. kein direkter Nachfolger $q_j$ für einen Zustand $q_i$ existiert, gibt es auch kein Wort, das einen Übergang ohne Zwischenzustände erlaubt, also ist diese Sprache leer. $L(\varnothing)$ ist aber die leere Sprache also ist $\varnothing$ genau der passende reguläre Ausdruck.

Beim Lesen eines leeren Wortes bleibt der Automat im selben Zustand, also muss das leere Wort zur Sprache gehören, falls i=j.

Als Induktionsvoraussetzung seien alle regulären Ausdrücke $R_{i,j}^{k}$ gegeben.
In jedem Induktionsschritt erhöht sich k um 1, d.h. wir lassen einen weiteren Zustand ($q_{k+1}$) für die Übergänge zu. Es gibt nun zwei Möglichkeiten für den Übergang von $q_i$ nach $q_j$.

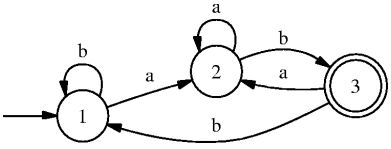



- $q_{k+1}$ wird nicht durchlaufen, d.h. alle Zustände haben einen Index ≤ k – dieser Pfad (grün) ist also durch $R_{i,j}^{k}$ beschreibbar.
- $q_{k+1}$ wird durchlaufen. In diesem Fall müssen wir zunächst von $q_i$ nach $q_{k+1}$ kommen (dabei wird kein Zustand mit Index >k durchlaufen, denn sonst wären wir ja schon bei $q_{k+1}$ und $q_{k+2}$ etc. dürfen sowieso noch nicht benutzt werden), und am Ende von $q_{k+1}$ nach $q_j$. Dazwischen ist eine beliebig lange Schleife von $q_{k+1}$ nach $q_{k+1}$ möglich. Dieser Pfad (rot) ist durch den regulären Ausdruck $R_{i,k+1}^{k}(R_{k+1,k+1}^{k})^{*} R_{k+1,j}^{k}$ beschreibbar.

Folglich ist $R_{i,j}^{k+1} = R_{i,j}^{k} + R_{i,k+1}^{k}(R_{k+1,k+1}^{k})^{*} R_{k+1,j}^{k}$ – es sollte klar sein, dass auch $R_{i,j}^{k+1}$ wieder ein regulärer Ausdruck ist. Beachten Sie, dass in der Tabellennotation $R_{i,j}^{k+1}$ ausschließlich aus Elementen der Vorgängerspalte berechnet wird.

Erklärung, warum $(R_{k+1,k+1}^{k})^{*}$ und nicht $(R_{k+1,k+1}^{k+1})^{*}$, obwohl die Schleife doch offensichtlich $q_{k+1}$ durchläuft: Das steckt in dem Stern. Ein Durchlauf von $q_{k+1}$ nach $q_{k+1}$, wobei zwischendurch $q_{k+1}$ eventuell mehrfach durchlaufen wird kann interpretiert werden als viele nacheinander ausgeführte Durchläufe von $q_{k+1}$ nach $q_{k+1}$, wobei in keinem mehr $q_{k+1}$ durchlaufen wird.


Für k=n existieren keine Beschränkungen mehr, was die Pfade vom Startzustand zu den Finalzuständen betrifft, also ist die Sprache des Automaten durch $L(A)=L(R_{1,f1}^{n}+...+R_{1,fs}^{n})$ beschreibbar.
Beispiel: Sei der folgende Automat gegeben.




Da der Automat nur einen Finalzustand hat, interessieren eigentlich nur die für die Berechnung von $R_{1,3}^{3}$ nötigen Ausdrücke (das sind nicht alle angegebenen).

| i | j | $R_{i,j}^{0}$ | $R_{i,j}^{1}$ $(\varepsilon+b)^{*}=b^{*}$ | $R_{i,j}^{2}$ $(\varepsilon+a)^{*}=a^{*}$ | $R_{i,j}^{3}$ $(\varepsilon+b^{*}a^{*}ab)^{*}=(b^{*}a^{*}ab)^{*}$ |
|---|---|---|---|---|---|
| 1 | 1 | $\varepsilon+b$ | $(\varepsilon+b)+(\varepsilon+b)\,b^{*}(\varepsilon+b)=\ b^{*}$ | $b^{*}+(b^{*}a)a^{*}\varnothing\ =\ b^{*}$ | |
| 1 | 2 | a | $a+(\varepsilon+b)b^{*}a=a+b^{*}a=\ b^{*}a$ | $b^{*}a+(b^{*}a)a^{*}(\varepsilon+a)=\ b^{*}a^{*}a$ | |
| 1 | 3 | $\varnothing$ | $\varnothing+(\varepsilon+b)b^{*}\varnothing=\ \varnothing$ | $\varnothing+(b^{*}a)a^{*}(b)=\ b^{*}a^{*}ab$ | $b^{*}a^{*}ab+(b^{*}a^{*}ab)(b^{*}a^{*}ab)^{*}(\varepsilon+b^{*}a^{*}ab)=$ $b^{*}a^{*}ab+(b^{*}a^{*}ab)(b^{*}a^{*}ab)^{*}=$ $(b^{*}a^{*}ab)(b^{*}a^{*}ab)^{*}$ |
| 2 | 1 | $\varnothing$ | $\varnothing+\varnothing\ b^{*}(\varepsilon+b)=\ \varnothing$ | $\varnothing+(\varepsilon+a)a^{*}\varnothing=\ \varnothing$ | $\varnothing+(a^{*}b)(b^{*}a^{*}ab)^{*}bb^{*}=a^{*}b(b^{*}a^{*}ab)^{*}bb^{*}$ |
| 2 | 2 | $\varepsilon+a$ | $(\varepsilon+a)+\varnothing b^{*}a=\ \varepsilon+a$ | $(\varepsilon+a)+(\varepsilon+a)a^{*}(\varepsilon+a)=\ a^{*}$ | |
| 2 | 3 | b | $b+\varnothing\ b^{*}\varnothing=\ b$ | $b+(\varepsilon+a)a^{*}b=b+a^{*}b=\ a^{*}b$ | |
| 3 | 1 | b | $b+bb^{*}(\varepsilon+b)=b+bb^{*}+bb^{*}b=\ bb^{*}$ | $bb^{*}+(b^{*}a)a^{*}\varnothing=\ bb^{*}$ | |
| 3 | 2 | a | $a+bb^{*}a=(\varepsilon+bb^{*})a=\ b^{*}a$ | $b^{*}a+(b^{*}a)a^{*}(\varepsilon+a)=\ b^{*}a^{*}a$ | |
| 3 | 3 | $\varepsilon$ | $\varepsilon+bb^{*}\varnothing=\ \varepsilon$ | $\varepsilon+(b^{*}a)a^{*}b=\ \varepsilon+b^{*}a^{*}ab$ | |

Hinweis: da in allen $R_{i,j}$ mit demselben Index k+1 $(R_{k+1,k+1}{}^{k})^*$ als Teilausdruck vorkommt, ist dieser in den Spaltenköpfen notiert und vereinfacht. In den einzelnen Zellen wird er dann schon vereinfacht angegeben.

Ein weiteres Beispiel: Gegeben sei der DEA A = ({1,2}, {a,b}, δ, 1, {2}) mit

δ(1,a) = δ(2,a) = 1
δ(1,b) = δ(2,b) = 2

Da nur 2 akzeptierend ist R = $R_{12}{}^{2}$.Wir erhalten rekursiv folgende Ausdrücke:

$$R_{12}{}^{2} = R_{12}{}^{1} + R_{12}{}^{1} (R_{22}{}^{1})^* R_{22}{}^{1}$$

$$R_{12}{}^{1} = R_{12}{}^{0} + R_{11}{}^{0} (R_{11}{}^{0})^* R_{12}{}^{0}$$

$$R_{22}{}^{1} = R_{22}{}^{0} + R_{21}{}^{0} (R_{11}{}^{0})^* R_{12}{}^{0}$$

$$R_{12}{}^{0} = b$$

$$R_{11}{}^{0} = \varepsilon + a$$

$$R_{22}{}^{0} = \varepsilon + b$$

$$R_{21}{}^{0} = a$$

also:

$$R_{12}{}^{1} = b + (\varepsilon + a) (\varepsilon + a)^* b = b + (\varepsilon + a) (a)^* b = b + a^*b = a^*b$$

$$R_{22}{}^{1} = (\varepsilon + b) + a (\varepsilon + a)^* b = (\varepsilon + b) + a (a)^* b = \varepsilon + a^*b$$

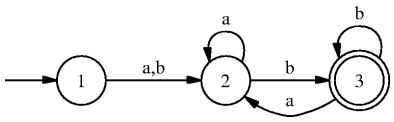
$$R_{12}{}^{2} = a^*b + a^*b (\varepsilon + a^*b)^* (\varepsilon + a^*b) = a^*b (\varepsilon + a^*b)^* = a^*b (a^*b)^* = (a+b)^*b$$

Satz: Die Menge der regulären Sprachen ist abgeschlossen unter (1) Vereinigung, (2) Schnitt, (3) Komplement, (4) Produkt (Konkatenation), (5) Stern.

folgt unmittelbar aus der Äquivalenz reguläre Sprachen / von endlichen Automaten akzeptierte Sprachen und den Konstruktionen am Ende von Kapitel 2.

*3.4 Das Pumping Lemma für reguläre Sprachen*

Vorbemerkung:
gegeben DEA mit n Zuständen.
Eingabe eines Wortes w der Länge ≥ n: mindestens ein Zustand mehr als einmal erreicht.




Beispiel: Eingabe abab erreicht 2 zweimal:
a führt hin, ba Schleife, b zu 3.
Schleife kann offensichtlich beliebig oft durchlaufen werden, d.h nicht nur abab wird akzeptiert, sondern auch a ba ... ba b, wobei das ba beliebig of wiederholt wird.

Satz (Pumping Lemma, uvw-Theorem):
Sei L eine reguläre Sprache. Dann gibt es eine Zahl n, so dass sich alle Wörter $x \in L$ mit $|x| \geq n$ zerlegen lassen in $x = uvw$, so dass gilt:

1. $|v| \geq 1$,
2. $|uv| \leq n$,
3. für alle $i \in \{0,1,2,...\}$ : $uv^iw \in L$.

Lemma hat Form einer Implikation: wenn ... dann (nicht Äquivalenz!). Formaler:

L regulär
$\Rightarrow \exists n \; \forall x.\; x \in L$ und $|x| \geq n \Rightarrow \exists u,v,w.\; [x = uvw$ und $|uv| \leq n$ und $|v| \geq 1$ und $\forall i.\; uv^iw \in L]$

Kontraposition:

$\forall n \; \exists x.$ x in L und $|x| \geq n$ und $\forall u,v,w.\; [x = uvw$ und $|uv| \leq n$ und $|v| \geq 1 \Rightarrow \exists i.\; uv^iw \notin L]$
$\Rightarrow$
L nicht regulär

insbesondere nützlich, um zu zeigen, dass bestimmte Sprachen NICHT regulär sind!

Widerspruchsbeweis nach folgendem Schema:
Zu zeigen: Sprache L, deren Definition vorliegt, ist nicht regulär.
Wir nehmen an, L wäre regulär, suchen ein Wort w aus L (abhängig von n), so dass sich aus dem Pumping Lemma ergibt, dass auch ein w' in L sein müsste, von dem wir aber wissen, dass es nicht zu L gehört. Damit ist die Annahme, L sei regulär, widerlegt.

Beispiel 1: $L = \{a^ib^i \mid i \geq 1\}$

Wir zeigen, dass L nicht regulär ist.
Angenommen, L wäre regulär. Sei n die Zahl aus dem Pumping Lemma (Pumping Zahl).
Betrachte $a^nb^n \in L$. Dieses Wort lässt sich nach dem Pumping Lemma in uvw zerlegen.
Aufgrund von Bedingung 2 kann v nur aus a's bestehen. Sei $k > 0$ die Anzahl der a's in v. Für beliebiges $j \geq 0$ muss $a^{n-k}a^{jk}b^n$ zu L gehören. Das führt z.B. für j=0 zum Widerspruch zur Definition von L, denn $a^{n-k}b^n$ ist nicht in L. L ist also nicht regulär.

Beispiel 2: $L = \{w \in \{a,b\}^* \mid w \text{ enthält mehr a's als b's}\}$

Angenommen, L wäre regulär. Sei n die Zahl aus dem Pumping Lemma (Pumping Zahl). Betrachte $b^n a^{n+1} \in L$. Dieses Wort lässt sich nach dem Pumping Lemma in uvw zerlegen. Aufgrund von Bedingung 2 kann v nur aus b's bestehen. Sei $k > 0$ die Anzahl der b's in v. Für beliebiges $j \geq 0$ muss $b^{n-k}b^{jk}a^{n+1}$ zu L gehören, was der Definition von L widerspricht, denn z.B. für $j = 2$ erhalten wir ein Wort, das nicht mehr a's als b's enthält. L ist also nicht regulär.

### *3.5 Entscheidungsprobleme*

Zu untersuchende Entscheidungsprobleme im Kontext regulärer (und anderer) Sprachen:

Wortproblem: gegeben w, gehört w zu L?
entscheidbar für reguläre Sprachen:
konstruiere DEA, teste, ob w zu akzeptierendem Zustand führt

Leerheitsproblem: ist $L = \varnothing$?
entscheidbar für reguläre Sprachen:
konstruiere DEA, teste, ob ein akzeptierender Zustand erreichbar

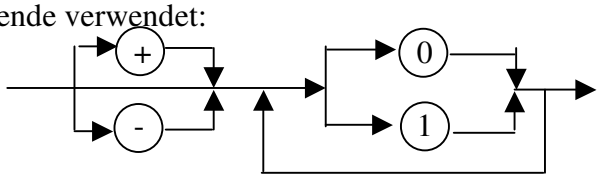
Endlichkeitsproblem: ist L endlich?
entscheidbar für reguläre Sprachen:
konstruiere DEA der nur erreichbare Zustände enthält, teste, ob ein Zyklus im Automaten existiert, von dem aus Endzustand erreichbar

Schnittproblem: ist $L_1 \cap L_2$ leer?
entscheidbar für reguläre Sprachen:
konstruiere Kreuzproduktautomaten, teste auf Leerheit

Äquivalenzproblem: ist $L_1 = L_2$ ?
entscheidbar für reguläre Sprachen:
konstruiere Minimalautomaten, teste ob diese (bis auf Benennung der Zustände) übereinstimmen

### *3.6 Reguläre Sprachen und Syntaxdiagramme*

Bei der Beschreibung von Programmiersprachen werden oft Syntaxdiagramme wie das folgende verwendet:



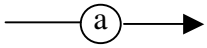

Die durch ein Syntaxdiagramm beschriebene Sprache ist die Menge aller Wörter, die durch Aneinanderreihung der Symbole entsteht, die man beim kompletten Durchlauf des Graphen vom Eingang (hier links) in Pfeilrichtung bis zum Ausgang (hier rechts) „besucht“. Im

Beispiel ist die Sprache die Menge aller Folgen von 0 und 1 mit Mindestlänge 1, wobei optional ein + oder ein – vorangestellt werden kann.

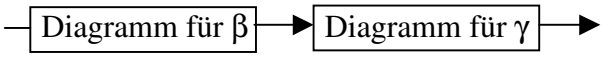
Auch reguläre Ausdrücke $\alpha \neq \varnothing$ sind durch Syntaxdiagramme beschreibbar, wie wir induktiv über den Aufbau von $\alpha$ zeigen wollen:

$\alpha = \varepsilon$

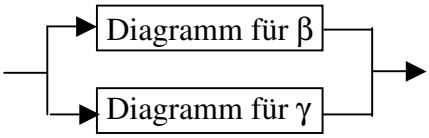
$\alpha = a,\ a \in \Sigma$



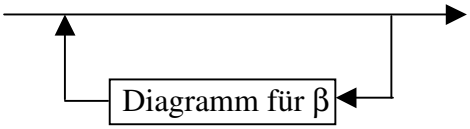

$\alpha = \beta\gamma$




$\alpha = \beta+\gamma$




$\alpha = \beta^*$




Es lassen sich damit alle regulären Ausdrücke bis auf $\varnothing$ mittels Syntaxdiagrammen darstellen.

Umgekehrt kann man auch zeigen, dass sich wiederum jedes Syntaxdiagramm in einen endlichen Automaten überführen lässt, der dieselbe Sprache akzeptiert. Damit stellen Syntaxdiagramme eine weitere Möglichkeit dar, reguläre Sprachen – bis auf die leere – darzustellen.