Vorlesungsmitschrift zur Vorlesung Theoretische Informatik vom 02. Juni 2000

Lutz Dittrich   Stefan Liske

15. Juni 2000

4.3 Reguläre Sprachen

Grammatiken

Grammatiken sind Gebilde, die zur Beschreibung von Wörtern bestimmter Sprachen dienen. Sie bestehen aus Regeln (Ersetzungsregeln, Produktionsregeln), die ihrerseits zwei Arten von Symbolen enthalten: Terminalsymbole, die in den Zeichenketten selbst vorkommen und Nonterminalsymbole (auch als Superzeichen oder Variablen bezeichnet), die nur zur Stukturbeschreibung benutzt werden.

Definition Grammatik:

Eine Grammatik G ist ein Quadrupel $G = (V_N, V_T, R, S)$ mit

- V_N = Alphabet der Nonterminalsymbole,
- V_T = Alphabet der Terminalsymbole,
- R = Menge von Ersetzungsregeln der Form $\alpha \to \beta$, wobei $\alpha \in (V^* V_N V^*)$ und $\beta \in V^*$ (Die linke Seite nennt man Prämisse und die rechte Seite nennt man Conclusio.),
- S = Satzsymbole (auch als Axiom oder Startsymbol bezeichnet) aus V_N.
- Es ist $V_N \cap V_T = \emptyset,\ V_N \cup V_T = V$.

Beispiel: $G_1 = (V_N, V_T, R, S)$ mit

- $V_N = \{A, B, C\}$
- $V_T = \{a, b, ;\}$
- $R = \{C \to A; ,\ A \to aB,\ A \to BBb,\ B \to b,\ B \to ab\}$
- $S = C$

Aus dem Satzsymbol C kann man durch Ableitung Zeichenketten erzeugen. Mittels der Grammatik G_1 lassen sich aus C u.a. folgende Zeichenketten ableiten:
$C \Rightarrow A; \Rightarrow aB; \Rightarrow ab;$
$C \Rightarrow A; \Rightarrow BBb; \Rightarrow abBb; \Rightarrow ababb;$
$C \Rightarrow A; \Rightarrow BBb; \Rightarrow bBb; \Rightarrow bbb;$

Für Grammatiken im allgemeinen gibt es nur zwei Einschränkungen: Sie dürfen nur endlich viele Regeln haben, und jede Regelprämisse muss mindestens ein nonterminales Symbol enthalten. Das Wort kann im Lauf der Ableitung beliebig wachsen und wieder schrumpfen. Wenn man die Form, die die Regeln einer Grammatik annehmen können, beschränkt, erhält man Grammatiktypen und damit auch Sprachtypen von verschiedenen Schwierigkeitsgraden.

Eine Grammatik $G = (V_N, V_T, R, S)$ heißt

- **rechtslinear** gdw. $\forall (P \rightarrow Q) \in R(P \in V_N$ und $Q \in (V_T^* \cup V_T^+ V_N))$.
  Es wird ein einzelnes Nonterminalsymbol ersetzt. Mit einer Regelanwendung wird jeweils höchstens ein terminales Symbol erzeugt, welches, wenn es auftritt, ganz rechts im Wort steht.

- **kontextfrei** gdw. $\forall (P \rightarrow Q) \in R(P \in V_N$ und $Q \in (V_N \cup V_T)^*)$.
  Es wird ein einzelnes Nonterminalsymbol ersetzt. Das Wort in der Conclusio kann nonterminale und terminale Symbole in beliebiger Mischung enthalten.

- **kontextsensitiv** gdw. $\forall (P \rightarrow Q) \in R(\exists u, v, \alpha \in (V_N \cup V_T)^* \quad \exists A \in V_N(P = uAv$ und $Q = u\alpha v$ mit $|\alpha| \geq 1)))$, oder die Regel hat die Form $S \rightarrow \varepsilon$. S kommt in keiner Regelconclusio vor.
  Es wird ein Nonterminalsymbol A in eine Zeichenkette α mit einer Länge von mindestens 1 überführt (d.h. das Wort wird durch die Regelanwendung nicht wieder kürzer). Diese Ersetzung von A durch α findet aber nur statt, wenn der in der Regel geforderte Kontext, links u und rechts v, im Wort vorhanden ist.

Die Chomsky[1]-Hierarchie

Noam Chomsky hat 1959 die Grammatiken in eine vierstufige Hierarchie eingeteilt, die seitdem als Gliederung benutzt wird.

Chomskys Grammatik-Typen: Eine Grammatik $G = (V_N, V_T, R, S)$ gehört

- **zum Typ 3** wenn sie rechtslinear ist,
- **zum Typ 2** wenn sie kontextfrei ist,
- **zum Typ 1** wenn sie kontextsensitiv ist,
- **zum Typ 0** in allen sonstigen Fällen.

Die Komplexität der mit diesen Grammatiken generierbaren Sprachen nimmt vom Typ 0 zum Typ 3 hin ab. Für jede Sprachklasse der Chomsky-Hierarchie gibt es einen Automatentyp, der es gestattet Sprachen dieser Klasse zu erkennen.

Sprachtyp	**Klassifizierungsmerkmal**	**zugehöriger Automatentyp**
0	aufzählbare Sprachen	Turingmaschine
1	kontextfreie Sprachen	Linear beschränkter Automat
2	kontextfreie Sprachen	Kellerautomaten
3	reguläre Sprachen	Deterministischer Endlicher Automat

Abschlusseigenschaften: Da formale Sprachen Mengen von Zeichenketten sind, lassen sich die folgenden mengen- und zeichenkettenspezifischen Operationen auf sie anwenden: Vereinigung, Schnitt, Differenz, Komplement, Verkettung und Sternoperation. Für die Chomsky-Hierarchie gelten die folgenden Abschlusseigenschaften:

[1]Noam Chomsky, geb. 7.12.1928, hat als Professor für Linguistik am Massechusetts Institue of Technology (bei Boston, USA) ab 1955 grundlegende linguistische Theorien entwickelt.

Operation	**Typ 0**	**Typ 1**	**Typ 2**	**Typ 3**
Vereinigung	+	+	+	+
Schnitt	+	+	-	+
Komplement	-	+	-	+
Verkettung	+	+	+	+
Sternoperation	+	+	+	+

(+ bedeutet abgeschlossen)

Hinweis: Die Beweise der Abschlusseigenschaften sind unter anderem in [Erk98] beschrieben.

Sprachklassen können auf verschiedene Weisen definiert werden: Durch Erzeugungsverfahren, die Sprachen erzeugen, oder durch Erkennungsverfahren, die Zeichenketten als Elemente einer Sprache erkennen. Erkennungen können durch Akzeptoren beschrieben werden. Erzeugungen werden zum Beispiel durch Grammatiken und reguläre Ausdrücke beschrieben. Sprachen können auf viele verschiedenen Arten definiert werden, u.a. durch Aufzählung der Wörter, durch Grammatiken, durch Automaten (Akzeptoren) und durch Eigenschaftsspezifikation.
Bisher haben wir Sprachen immer nur abstrakt oder durch zugehörige Akzeptoren beschrieben. Reguläre Ausdrücke stellen eine andere Art dar, Sprachen zu spezifizieren.

Definition N: (reguläre Ausdrücke)
Sei X ein Alphabet und $V = \{(,), \cup, ^*, \circ, \emptyset\}$ ein Zeichenvorrat. Die Menge $Reg(X)$ der regulären Ausdrücke über X ist folgendermaßen definiert.

(i) $\forall x \in X : x \in Reg(X)$

(ii) $\emptyset \in Reg(X)$

(iii) $\forall R, Q \in Reg(X) : (R \cup Q) \in Reg(X), (R \circ Q) \in Reg(X)$

(iv) $\forall R \in Reg(X) : R^* \in Reg(X)$

(v) $Reg(X)$ ist die kleinste Teilmenge von $(X \cup V)^*$, die (i)-(iv) erfüllt.

Beispiele für reguläre Ausdrücke : $X = \{x_1, x_2, x_3\}$

- x_1
- $(((x_2 \circ x_1) \circ x_3) \cup x_3)$
- $((x_1 \circ x_2)^* \cup x_2)^*$

Reguläre Ausdrücke sind sinnleere Zeichenfogen, denen über folgende Abbildung einen Bedeutung zugeordnet wird.

Definition O: (Bedeutungszuordnung zum regulären Ausdruck)
Die Bedeutung eines regulären Ausdrucks über X wird durch folgende Abbildung $\sigma : Reg(X) \to 2^{X^*}$ festgelegt, die induktiv definiert ist durch:

(i) Wenn $x \in X$, dann sei $\sigma(x) = \{x\}$

(ii) $\sigma(\emptyset) = \emptyset$

(iii) $\forall R, Q \in Reg(X)$ sei $\sigma((R \cup Q)) = \sigma(R) \cup \sigma(Q)$ und
$\sigma((R \circ Q)) = \sigma(R) \circ \sigma(Q)$

(iv) $\forall R \in Reg(X) : \sigma(R^*) = \sigma(R)^*$

$Reg(X)$ ist genaugenommen nur eine Menge von Zeichenreihen, den regulären Ausdrücken. Diesen Ausdrücken gibt σ eine Bedeutung, eine Interpretation. Diese Bedeutung ist eine Sprache. $\sigma(x)$ ist die Sprache, die der reguläre Ausdruck x beschreibt.
Ist $\sigma(r) = \sigma(r')$, so heißen r und r' äquivalent.

Prioritätsregeln: Einige Regeln mit denen der Umfang(Anzahl der Symbole) regulärer Ausdrücke verringert werden kann. Bei der Kürzung muss die durch die reguläre Ausdrücke beschriebene Sprache erhalten bleiben.

- „$\circ$“ vor „$\cup$“
  Dies bedeutet $\circ$ bindet stärker als $\cup$.

  Beispiel: $(a \circ b) \cup c = a \circ b \cup c$ (Es wird zuerst a mit b konkateniert und dann wird ab mit c vereinigt.)

- Weglassen von Klammern
  Man kann Klammern weglassen, sofern sich die beschriebene Sprache nicht ändert.

  Beispiel: $(a \circ b) = a \circ b$
  $(a \circ b) \circ c = a \circ b \circ c$

- Weglassen des Konkatenationspunktes $\circ$
  Man kann $\circ$ weglassen.

  Beispiel: $a \circ b = ab$

Durch Kombination dieser Regeln erhält man zum Beispiel: $(x_1 \cup (x_2 \circ x_3)) = x_1 \cup x_2 x_3$

Definition P: (reguläre Sprachen)
Sei X ein Alphabet. Eine Sprache $L \subseteq X^*$ heißt reguläre Sprache, wenn es einen regulären Ausdruck $R \in Reg(X)$ mit $\sigma(R) = L$ gibt. Die Menge aller regulären Sprachen über X bezeichnet man mit $\mathcal{R}(X)$.
Die Klasse aller regulären Sprachen bezeichet man mit $\mathcal{R}$.

Beispiel: $R = a(a^* \cup b^*)^* \cup (b \cup c)(a \cup b \cup c)^* \cup (ca)^*$
$\sigma(R)$ ist die durch den regulären Ausdruck R beschriebene Sprache.
Nach Definition O Punkt (iii) gilt:
$\sigma(R) = \sigma(a(a^* \cup b^*)^*) \cup \sigma((b \cup c)(a \cup b \cup c)^*) \cup \sigma((ca)^*)$.
$\sigma(a(a^* \cup b^*)^*)$ ist die Sprache aller Wörter über dem Alphabet X, die mit a beginnen und kein c enthalten. Also $\sigma(a(a^* \cup b^*)^*) = \{\omega \in X^* | \omega \in \{a, b\}^*$ und ω beginnt mit a $\}$
$\sigma((b \cup c)(a \cup b \cup c)^*)$ ist die Sprache aller Wörter über X, die mit b oder c beginnen. Also $\sigma((b \cup c)(a \cup b \cup c)^*) = \{\omega \in X^* | \omega \in X^*$ und ω beginnt nicht mit a oder $\omega = \varepsilon\}$.
$\sigma(\sigma((ca)^*))$ ist die Sprache aller Wörter über X, die nur aus Konkatenationen der Konkatenation ab bestehen, vereinigt mit dem leeren Wort. Also $\sigma((ca)^*) = \{\omega \in X^* | \omega \in \{ac\}^*\}$. Es ist $(\sigma(\sigma((ca)^*) \ \setminus \ \emptyset)) \subset \sigma((b \cup c)(a \cup b \cup c)^*)$. Daraus folgt: $\sigma(R) = \{\omega \in X^* | \omega \in \{a, b\}^*$ und ω beginnt mit a oder $\omega \in \{a, b, c\}^*$ und ω beginnt nicht mit a oder $\omega = \varepsilon\}$
Man kann folgenden zu R äquivalenten regulären Ausdruck R' aufschreiben:
$R' = a(a \cup b)^* \cup (b \cup c)(a \cup b \cup c)^* \cup \emptyset^*$

Satz Q: (Gleicheit der Menge der regulären Sprachen R und der Menge der deterministisch akzeptierten Sprachen DA.) $\mathcal{R} = DA$

Beweis:

- **„$\mathcal{R} \subseteq$ DA"** : Zu zeigen: Zu jeder Sprache, die durch einen regulären Ausdruck definiert ist, gibt es einen endlichen deterministischen Akzeptor.
  Nach Definition N ist $\mathcal{R}$ genau die kleinste Klasse von Sprachen,
  - die die leere Menge enthält,
  - die alle Mengen $\{x\}$, $x \in X$ enthält,
  - die gegen Vereinigung, Konkatenation und Sternbildung abgeschlossen ist.

  Lemma E: DA enthält die leere Menge und alle endlichen Mengen.
  Nach Satz D und K ist DA abgeschlossen gegen Vereinigung, Konkatenation und Sternbildung. Es gilt somit $DA \supseteq \mathcal{R}$

- **„$\mathcal{R} \supseteq DA$"** Zu zeigen: Zu jeder Sprache, die von einem endlichen Automaten akzeptiert wird, gibt es einen regulären Ausdruck, der diese Sprache beschreibt.
  Beweisidee: Wähle einen beliebigen deterministischen Akzeptor. Konstruiere dazu einen regulären Ausdruck R mit $\sigma(R) = L(A)$.
  Sei $A = (X, S, \delta, q_1, F)$ ein endlicher deterministischer Akzeptor mit n Zuständen $S = \{q_1, ..., q_n\}$.

  Beweismethode: Vollständige Induktion über die Kompliziertheit der Wege, die man verwendet um vom Zustand q_1 zu einem Zustand q_j zu gelangen, wobei $j \in \{1, ..., n\}$. Der einfachste Weg von q_1 nach q_f verwendet keine Zwischenzustände. Der nächstkompliziertere Weg darf als Zwischenzustand q_2 benutzen, der nächstkompliziertere q_2 und q_3. Definiere für alle $i, j, k \in \{1, ..., n\}$ $W_{i,j}^k := \{\omega \in X^* | \delta(q_i, \omega) = q_j$ und für alle $u, v \in X^*$ mit $u \neq \varepsilon \neq v$ und $\omega = uv$ gilt $\delta(q_i, u) \in \{q_1, ..., q_k\}\}$

  $W_{i,j}^k$ ist Menge aller Wörter mit denen man vom Zustand q_i in den Zustand q_j kommt und unterwegs nur Zustände mit Indizes kleiner oder gleich k verwendet hat.

Induktionsanfang:

$$k = 0: \quad W_{i,j}^0 = \begin{cases} \{x \in X | \delta(q_i, x) = q_j\} & \text{falls} \quad i \neq j \\ \{\{\varepsilon\} \cup \{x \in X | \delta(q_i, x) = q_j\} & \text{falls} \quad i = j \end{cases}$$

Für eine endliche Menge $\{x \in X | \delta(q_i, x) = q_j\} = \{x_{i_1}, ..., x_{i_t}\}$ steht der regulären Ausdruck $x_{i_1} \cup x_{i_2} \cup ... \cup x_{i_t}$. Ist $\{x \in X | \delta(q_i, x) = q_j\} = \emptyset$ gehört dazu der reguläre Ausdruck $\emptyset$. Und zu ε gehört der reguläre Ausdruck $\emptyset^*$.
$W_{i,j}^0$ kann also durch einen regulären Ausdruck dargestellt werden.

Induktionsannahme: Für ein $k \geq 0$ und alle $i, j \in \{1, ..., n\}$ seien die Mengen $W_{i,j}^m$ mit $m \leq k$ bereits konstruiert und durch reguläre Ausdrücke darstellbar. $W_{i,j}^{k+1}$ enthält:

- $W_{i,j}^k$, da man hier mit den Zwischenzuständen bis q_k auskommt,
- vereinigt mit Wegen von q_i nach q_j, die den Zwischenzustand q_{k+1} mindestens einmal benutzen.

Letztere lassen sich dann darstellen als Wege, die

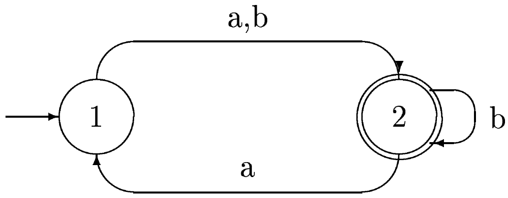
- von q_i nach q_{k+1} gehen und dabei nur Zwischenzustände bis höchstens q_k benutzen.
- von q_{k+1} noch beliebig of nach q_{k+1} zurückkehren, dabei aber wieder nur Zwischenzustände bis höchstens q_k benutzen
- und schließlich von q_{k+1} zum Zielzustand q_j führen (und dabei wieder nicht über q_k hinausgehen).

Dann gilt für alle $i, j \in \{1, ..., n\}$: $W_{i,j}^{k+1} = W_{i,j}^k \cup W_{i,k+1}^k \circ (W_{k+1,k+1}^k)^* \circ W_{k+1,j}^k$ ist ein regulärer Ausdruck.

Fazit: Alle $W_{i,j}^{k+1}$ werden durch eine regulären Ausdruck beschrieben.
Da $L(A) = \bigcup_{q_j \in F} W_{1,j}^n$ und es zu jedem dieser $W_{1,j}^n$ einen regulären Ausdruck r_j gibt, ergibt sich für $L(A)$ folgender regulärer Ausdruck $r_{j_1} \cup r_{j_2} \cup ... \cup r_{j_m}$ wobei alle $q_{j_m} \in F$. Es wurde also gezeigt, dass $L(A) \subseteq \mathcal{R}$.
Aus $R \supseteq L(A)$ und $R \subseteq L(A)$ folgt $R = L(A)$.

Der Beweis liefert eine konstruktive Methode um zu einem gegebenen deterministischen Akzeptor einen regulären Ausdruck zu finden, der genau die Sprache beschreibt, die von dem Akzeptor erkannt wird. Die so entstehenden Ausdrücke wachsen allerdings sehr schnell an. Es gibt zwar Vereinfachungsmöglichkeiten (Ersetzen von Teilausdrücken durch äquivalente einfachere, Bestimmung nur der zur Konstruktion benötigten $W^k_{i,j}$), jedoch ist im allgemeinen ein Explodieren der Größe des entstehenden regulären Ausdrucks unvermeidlich.

Beispiel: $X = \{a, b\}, S = \{1, 2\}, s_0 = 1, F = \{2\}$



$W^0_{1,1} = \{\varepsilon\}$ Man verbleibt in 1 wenn keinen Eingabe erfolgt.
$W^0_{1,2} = \{a, b\}$ Man gelangt von 1 nach 2 nur bei Eingabe eines a oder eines b.
$W^0_{2,2} = \{b, \varepsilon\}$ Man verbleibt in 2 bei Eingabe eines b und wenn keinen Eingabe erfolgt.
$W^0_{2,1} = \{a\}$ Man gelangt von 2 nach 1 bei Eingabe eines a.

$W^1_{1,1} = W^0_{1,1} \cup W^0_{1,1} \circ (W^0_{1,1})^* \circ W^0_{1,1} = \{\varepsilon\} \cup \{\varepsilon\} \circ \{\varepsilon\}^* \circ \{\varepsilon\} = \{\varepsilon\}$
$W^1_{1,2} = W^0_{1,2} \cup W^0_{1,1} \circ (W^0_{1,1})^* \circ W^0_{1,2} = \{a, b\} \cup \{\varepsilon\} \circ \{\varepsilon\}^* \circ \{a, b\} = \{a, b\}$
$W^1_{2,2} = W^0_{2,2} \cup W^0_{2,1} \circ (W^0_{1,1})^* \circ W^0_{1,2} = \{b, \varepsilon\} \cup \{a\} \circ \{\varepsilon\}^* \circ \{a, b\} = \{b, \varepsilon\} \cup \{aa, ab\}$
$W^1_{2,1} = W^0_{2,1} \cup W^0_{2,1} \circ (W^0_{1,1})^* \circ W^0_{1,1} = \{a\} \cup \{a\} \circ \{\varepsilon\}^* \circ \{\varepsilon\} = \{a\}$

$W^2_{1,1} = W^1_{1,1} \cup W^1_{1,2} \circ (W^1_{2,2})^* \circ W^1_{2,1} = \{\varepsilon\} \cup \{a, b\} \circ (\{b, \varepsilon\} \cup \{aa, ab\})^* \circ \{a\} = \{\varepsilon\} \cup \{a, b\} \circ \{b, aa, ab\}^* \circ \{a\}$
$W^2_{1,2} = W^1_{1,2} \cup W^1_{1,2} \circ (W^1_{2,2})^* \circ W^1_{2,2} = \{a, b\} \cup \{a, b\} \circ (\{b, \varepsilon\} \cup \{aa, ab\})^* \circ (\{b, \varepsilon\} \cup \{aa, ab\}) = \{a, b\} \circ \{b, aa, ab\}^*$
$W^2_{2,2} = W^1_{2,2} \cup W^1_{2,2} \circ (W^1_{2,2})^* \circ W^1_{2,2} = (\{b, \varepsilon\} \cup \{aa, ab\}) \cup (\{b, \varepsilon\} \cup \{aa, ab\}) \circ (\{b, \varepsilon\} \cup \{aa, ab\})^* \circ (\{b, \varepsilon\} \cup \{aa, ab\}) = \{b, aa, ab\}^*$
$W^2_{2,1} = W^1_{2,1} \cup W^1_{2,2} \circ (W^1_{2,2})^* \circ W^1_{2,1} = \{a\} \cup (\{b, \varepsilon\} \cup \{aa, ab\}) \circ (\{b, \varepsilon\} \cup$

$\{aa, ab\})^* \circ \{a\} = \{b, aa, ab\}^* \circ \{a\}$

$L(A) = \bigcup_{j \in F} W_{i,j}^2 = W_{1,2}^2 = \{a, b\} \circ \{b, aa, ab\}^*$
Man erhält also folgenden regulären Ausdruck $(a \cup b) \circ (b \cup (a \circ a) \cup (a \circ b))^*$

Äquivalenzproblem für reguläre Ausdrücke: Es gibt einen Algorithmus, der für jedes beliebige Alphabet X und je zwei reguläre Ausdrücke $R, R' \in Reg(X)$ entscheidet ob $\sigma(R) = \sigma(R')$.
Begründung: Da nach obigem Beweis $\mathcal{R} = DA$, gibt es zu jedem regulären Ausdruck r einen deterministischen Akzeptoren A_r der genau die von r beschriebene Sprache akzeptiert. Zu je zwei regulären Ausdrücken r_1 und r_2 gibt es deterministischen Akzeptoren A_{r_1} und A_{r_2} mit $\sigma(r_1) = L(A_{r_1})$ und $\sigma(r_2) = L(A_{r_2})$. Da DA gegen Vereinigung, Schnitt und Komplementbildung abgeschlossen ist, lässt sich zu A_{r_1} und A_{r_2} ein deterministischer Akzeptor A_g finden mit $L(A_g) = (L(A_{r_1}) \cap \overline{L(A_{r_2})}) \cup (\overline{(L(A_{r_1})} \cap L(A_{r_2}))$. Es gilt aber $(L(A_{r_1}) = L(A_{r_1}) \Leftrightarrow L(A_g) = \emptyset)$. Ob die von einem deterministischen Akzeptor erkannte Sprache die Sprache $\emptyset$ ist, lässt sich algorithmisch lösen, was in der zweiten Übungsaufgabe des Aufgabenblatts 4 zu zeigen war. Somit lässt sich auch die Frage der Äquivalenz zweier regulärer Ausdrücke algorithmisch lösen.

Zusammenfassung:

- Reguläre Sprachen sind die Sprachen, die sich durch reguläre Ausdrücke bilden lassen.

- Die Sprachklasse der regulären Sprachen stimmt mit dem Chomsky-Typ-3 überein und ist somit gleich DA.

- Es lässt sich zu jedem deterministischen Akzeptoren ein regulärer Ausdruck finden, der genau die Sprache beschreibt, die von dem Akzeptoren erkannt wird.

- Die zugehörigen Grammatiken heißen rechtslinear.

- Die regulären Sprachen sind abgeschlossen gegen Schnitt, Vereinigung, Konkatenation, Sternbildung und Komplementbildung.

Literaturverzeichnis:

[Erk98] Erk,K., Priese,L., “Theoretische Informatik”, Springer, 1998

[Wen91] Wendt,S., “Nichtphysikalische Grundlagen der Informationstechnik”, 1991