

20. REGULÄRE SPRACHEN

Zum Abschluss betrachten wir die kleinste der Chomsky-Klassen, die Klasse der rechtslinearen Sprachen, die auch als *reguläre* Sprachen bezeichnet werden.

Wir führen zunächst eine Normalform ein, aus der wir ein Pumpinglemma herleiten, das uns die Trennung der rechtslinearen von den linearen Sprachen erlaubt.

Danach werden wir die Mächtigkeit der Klasse der rechtslinearen Sprachen durch alternative Charakterisierungen (endliche Automaten, reguläre Ausdrücke) beschreiben und kurz auf Abschlusseigenschaften und Entscheidungs- und Komplexitätsfragen eingehen.

Detailliert behandelt werden die regulären Sprachen in der Vorlesung Formale Sprachen.

CHOMSKY-NORMALFORM UND PUMPINGLEMMA

DIE CHOMSKY-NORMALFORM FÜR RLIN

DEFINITION. Eine rechtslineare Grammatik $G = (N, T, P, S)$ ist in *Chomsky-Normalform*, falls G λ -treu ist und neben der eventuellen λ -Regel $S \rightarrow \lambda$ nur Regeln der Form

$$\begin{aligned} X &\rightarrow aY & (X, Y \in N, a \in T) \\ X &\rightarrow a & (X \in N, a \in T) \end{aligned}$$

besitzt.

NB. Die Herleitung eines Wortes $w = a_1 \dots a_n \in L(G)$ der Länge n hat ebenfalls Länge n :

$$S = X_1 \Rightarrow a_1 X_2 \Rightarrow a_1 a_2 X_3 \Rightarrow \dots \Rightarrow a_1 \dots a_{n-1} X_n \Rightarrow a_1 \dots a_n$$

ÜBERFÜHRUNG IN DIE CHOMSKY-NORMALFORM

- Zur Sicherung der λ -Treue und zur Elimination von Variablenumbenennungen geht man wie bei kontextfreien Grammatiken vor.
- Regeln der Form $X \rightarrow a_1 \dots a_n \beta$ mit $n \geq 2, a_1, \dots, a_n \in T, \beta \in N \cup \{\lambda\}$ ersetzt man durch

$$X \rightarrow a_1 Y_1$$

$$Y_1 \rightarrow a_2 Y_2$$

...

$$Y_{n-2} \rightarrow a_{n-1} Y_{n-1}$$

$$Y_{n-1} \rightarrow a_n \beta$$

wobei Y_1, \dots, Y_{n-1} neue Variablen sind.

EIN PUMPINGLEMMA FÜR RLIN

SATZ. Zu jeder rechtslinearen Sprache $L \in \Sigma^*$ gibt es eine Zahl $p \in \mathbb{N}$, sodass jedes Wort $z \in L$ mit $|z| \geq p$ eine Zerlegung $z = uvw$ in Teilwörter $u, v, w \in \Sigma^*$ besitzt, die folgende Eigenschaften hat:

$$(1) \quad v \neq \lambda$$

$$(2) \quad |vw| \leq p$$

$$(3) \quad \forall n \in \mathbb{N} (z_n = uv^nw \in L).$$

Weiter lässt sich ein geeignetes p effektiv aus jeder rechtslinearen Grammatik G , die L erzeugt, berechnen.

BEWEISIDEE

- Wähle $G = (N, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$.
- Setze $p = |N| + 1$.
- Für $z = a_1 \dots a_n \in L$ mit $n \geq p > |N|$ erhält man die gewünschte Zerlegung $z = uvw$ dann wie folgt:

Nehme eine Herleitung von w :

$$S = X_1 \Rightarrow a_1 X_2 \Rightarrow a_1 a_2 X_3 \Rightarrow \dots \Rightarrow a_1 \dots a_{n-1} X_n \Rightarrow a_1 \dots a_n$$

Betrachte die letzte Variablenwiederholung in dieser Herleitung, d.h. wähle k maximal, sodass es $m > k$ mit $X_k = X_m$ gibt.

Für $X = X_k = X_m$, $u = a_1 \dots a_{k-1}$, $v = a_k \dots a_{m-1}$ und $w = a_m \dots a_n$ gilt dann

$$v \neq \lambda \quad (\text{wegen } k < m)$$

$$|vw| \leq p \quad (\text{wegen Maximalität von } k)$$

sowie

$$S \Rightarrow^* uX = a_1 \dots a_{k-1} X_k$$

$$X_k = X \Rightarrow^* vX = a_k \dots a_{m-1} X_m$$

$$X_m = X \Rightarrow^* w = a_m \dots a_n$$

weshalb

$$\forall n \in \mathbb{N} (z_n = uv^n w \in L)$$

BEISPIEL

$L = \{0^n 1^n : n \geq 1\}$ ist nicht rechtlinear

Man zeigt dies indirekt. Widerspruchsannahme: L rechtlinear

Wähle p passend gemäss Pumpinglemma.

Betrachte $z = 0^p 1^p$ (NB: $|z| \geq p$).

Nach dem Pumpinglemma gibt es eine Zerlegung $z = uvw$ mit den Eigenschaften (i) - (iii).

Es gilt dann:

- Wegen (i) ist v nicht leer.
- Wegen (ii) gilt $|vw| \leq p$, d.h. v und w sind in dem 1-Block von z enthalten.
- Es ist also $v = 1^k$ für ein $k \geq 1$.

In $z_0 = uv^0w$ wird also der 1-Block verkürzt, während der 0-Block unverändert bleibt:

$$z_0 = 0^p 1^{p-k} \quad (k \geq 1)$$

ALSO: $z_0 \notin L$, im Widerspruch zu (iii).

FOLGERUNG: TRENNUNG VON LIN UND RLIN

KOROLLAR. $RLIN \subset LIN$.

BEWEIS. Da nach Definition $RLIN \subseteq LIN$ gilt, genügt es eine Sprache L mit

$$L \in LIN \setminus RLIN$$

anzugeben. Dies gilt aber für

$$L = \{0^n 1^n : n \geq 1\}.$$

- L wird von der linearen Grammatik mit den Regeln

$$S \rightarrow 0S1 \mid 01$$

erzeugt.

- $L \notin RLIN$ nach dem letzten Beispiel.

ENDLICHE AUTOMATEN:

EINE MASCHINENCHARAKTERISIERUNG VON RLIN

ENDLICHE AUTOMATEN

Ein *deterministischer endlicher Automat* M ist ein 5-Tupel

$$M = (\Sigma, Z, \delta, z_0, E),$$

wobei

- Σ ein Alphabet (*Eingabealphabet*)
- Z eine endliche Menge (*Menge der Zustände*)
- $\delta : Z \times \Sigma \rightarrow Z$ partiell (die *Übergangsfunktion*)
- $z_0 \in Z$ der *Startzustand* und
- $E \subseteq Z$ die Menge der *Endzustände*

sind. Ist $\delta \subseteq (Z \times \Sigma) \times Z$ eine Relation (*Übergangsrelation*), so heißt M ein *nichtdeterministischer endlicher Automat*.

Der deterministische Automat M arbeitet wie folgt: Bei Eingabe eines Wortes $w \in \Sigma^*$ liest M das Wort w Buchstabe für Buchstabe ein und aktualisiert dabei laufend seinen Zustand, wobei sich der jeweils nächste Zustand z' aus dem vorhergehenden Zustand z und dem gelesenen Buchstaben a gemäß δ ergibt ($\delta(z, a) = z'$). Die Rechnung beginnt mit dem Startzustand z_0 . Ist der Automat nach vollständigem Einlesen von w in einem Zustand $z \in E$, so *akzeptiert* er die Eingabe. Sonst wird die Eingabe verworfen. Insbesondere verwirft der Automat die Eingabe, wenn er diese nicht vollständig einlesen kann.

Im Falle eines nichtdeterministischen Automaten gilt für den Nachfolgezustand z' von z beim Lesen von a , dass $((z, a), z') \in \delta$ gilt. Hier ist z' i. Allg. nicht eindeutig bestimmt. Hier wird die Eingabe akzeptiert, falls es eine mögliche Rechnung gibt, die das vollständige Einlesen der Eingabe erlaubt und in einem Endzustand endet.

FORMALE BESCHREIBUNG DER ARBEITSWEISE VON M

- M -Konfiguration: $(z, w) \in Z \times \Sigma^* =: \text{KON}_M$
(= aktueller Zustand + Eingaberestwort)

- Startkonfiguration bei Eingabe w : (z_0, w)

- Ein-Schritt-Relation $\Rightarrow_M \subseteq \text{KON}_M \times \text{KON}_M$:

$(z, aw) \Rightarrow_M (z', w)$ genau dann, wenn $((z, a), z') \in \delta$,
wobei $z, z' \in Z$, $a \in \Sigma$ und $w \in \Sigma^*$.

- Mehrschrittrelation \Rightarrow_M^* : wie üblich

- Von M erkannte Sprache $L(M) \subseteq \Sigma^*$:

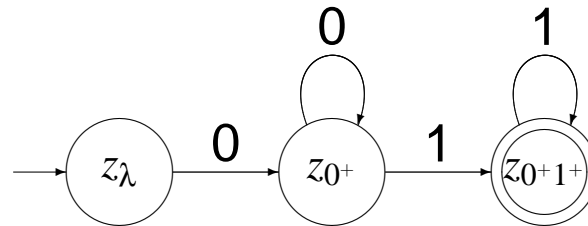
$$L(M) = \{w \in \Sigma^* : \exists z \in E((z_0, w) \Rightarrow_M^* (z, \lambda))\}.$$

BEISPIEL 1. Der deterministische endliche Automat $M = (\Sigma, Z, \delta, z_0, E)$ mit $\Sigma = \{0, 1\}$, $Z = \{z_\lambda, z_{0+}, z_{0+1+}, z_{\text{Fehler}}\}$, $z_0 = z_\lambda$, $E = \{z_{0+1+}\}$ und der Übergangsfunktion δ

Z	\times	Σ	\rightarrow	Z
z_λ		0		z_{0+}
z_{0+}		0		z_{0+}
z_{0+}		1		z_{0+1+}
z_{0+1+}		1		z_{0+1+}

erkennt die Sprache $L(M) = \{0^m 1^n : m, n \geq 1\}$.

Übergangsdiagramm von M :



Übergangsdiagramm (Übergangsgraph)

- Knoten = Zustände
- Kanten = Übergänge (aktuell gelesener Buchstabe)
- Startzustände durch leere Eingangskante markiert
- Endzustände durch Doppelkreislinien markiert

VERVOLLSTÄNDIGUNG

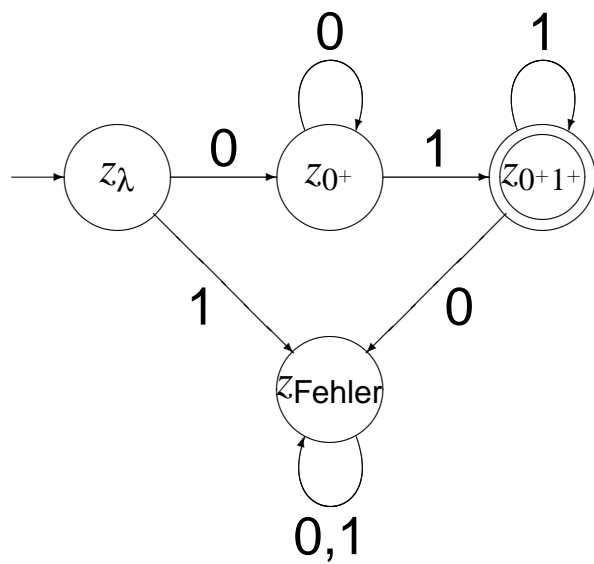
Ein det. endl. Automat $M = (\Sigma, Z, \delta, z_0, E)$ heisst *vollständig*, wenn die Übergangsfunktion δ total ist. (In diesem Fall wird jedes Eingabewort vollständig eingelesen; Akzeptieren und Verwerfen sind daher symmetrisch.)

Jeder Automat $M = (\Sigma, Z, \delta, z_0, E)$ lässt sich zu einem äquivalenten vollständigen Automaten $M' = (\Sigma, Z', \delta', z_0, E)$ erweitern:

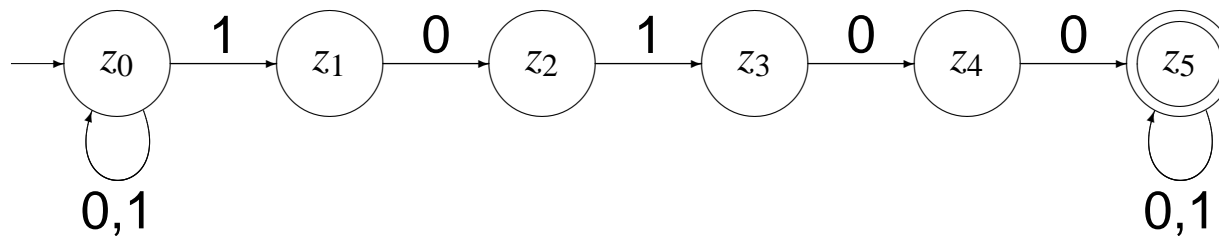
Man fügt einen neuen nichtakzeptierenden Zustand z_{Fehler} hinzu und setzt δ zu δ' fort, indem man $\delta'(z, a) = z_{\text{Fehler}}$ setzt für alle $(z, a) \in Z' \times \Sigma$ mit $\delta(z, a) \uparrow$. Der deterministische endliche Automat $M = (\Sigma, Z, \delta, z_0, E)$ aus Beispiel 1 wird so z.B. zu $M' = (\Sigma, Z \cup \{z_{\text{Fehler}}\}, \delta', z_0, E)$ erweitert, wobei δ' wie folgt definiert ist.

Z	\times	Σ	\rightarrow	Z
z_λ		0		z_{0^+}
z_λ		1		z_{Fehler}
z_{0^+}		0		z_{0^+}
z_{0^+}		1		$z_{0^+1^+}$
$z_{0^+1^+}$		0		z_{Fehler}
$z_{0^+1^+}$		1		$z_{0^+1^+}$
z_{Fehler}		0		z_{Fehler}
z_{Fehler}		1		z_{Fehler}

Übergangsdigramm von M' :



BEISPIEL 2. Ein nd. endl. Automat zur Erkennung der Binärwörter, die das Wort 10100 als Teilwort enthalten, wird durch folgendes Diagramm beschrieben:



Deterministischer Automat: Übung!

Wir werden gleich zeigen, dass jeder nd. endl. Automat äquivalent zu einem det. endl. Automaten ist.

MASCHINENCHARAKTERISIERUNG VON RLIN

SATZ. $RLIN = DEA = NEA$

Hierbei ist DEA (NEA) die Klasse der Sprachen, die von einem deterministischen (nichtdet.) endlichen Automaten erkannt werden.

Da offensichtlich $DEA \subseteq NEA$ gilt, genügt es zu zeigen:

- $RLIN \subseteq NEA$
- $NEA \subseteq DEA$
- $DEA \subseteq RLIN$

RLIN \subseteq NEA

Sei L rechtslinear.

Wähle $G = (N, T, P, S)$ rechtslinear in Chomsky - Normalform mit $L = L(G)$.

Definiere $M = (T, Z, \delta, z_0, E)$ mit $L(M) = L(G)$ wie folgt:

- $Z := N \cup \{+\}$
- $z_0 := S$
- $E := \{+\}$ (falls $\lambda \notin L(G)$) bzw. $E := \{+, S\}$ (falls $\lambda \in L(G)$)
- δ : $((X, a), Y) \in \delta \Leftrightarrow X \rightarrow aY \in P$
 $((X, a), +) \in \delta \Leftrightarrow X \rightarrow a \in P$

IDEE. Bei Eingabe $w = a_1 \dots a_n$ rät M eine Ableitung von w in G , wobei M sich im Zustand die Variablen X_k der aktuellen Satzform zur Ableitung des Restwortes $a_k \dots a_n$ merkt. Findet M solch eine Ableitung von w , beendet M die Rechnung im akzeptierenden Zustand $+$.

NEA \subseteq DEA

Sei $M = (\Sigma, Z, \delta, z_0, E)$ ein nd. endlicher Automat.

Einen deterministischen endlichen Automaten $M' = (\Sigma, Z', \delta', z'_0, E')$ mit $L(M) = L(M')$ erhält man mit folgender Idee (*Potenzmengenkonstruktion*): M' merkt sich nach Einlesen eines (Teil-)Wortes w alle M -Zustände in seinem Zustand, die M nach Einlesen von w möglicherweise erreicht haben kann. Enthält diese Menge am Ende einen M -Endzustand, so akzeptiert M' die Eingabe.

Formal ist M' wie folgt definiert:

- $Z' := \{A : A \subseteq Z\}$ (d.h. Z' ist die Potenzmenge von Z)
- $z'_0 := \{z_0\}$
- $E' := \{A \in Z' : A \cap E \neq \emptyset\}$
- $\delta' : Z' \times \Sigma \rightarrow Z'$ ist festgelegt durch

$$\delta'(A, a) = \{z' \in Z' : \exists z \in A ((z, a), z') \in \delta\}$$

DEA \subseteq RLIN

Sei $M = (\Sigma, Z, \delta, z_0, E)$ ein det. endlicher Automat, wobei o.B.d.A. die Übergangsfunktion δ total sei.

Eine rechtslineare Grammatik $G = (N, \Sigma, P, S)$ merkt sich in der Variablen den Zustand von M und simuliert jeden Übergang von M durch einen entsprechenden Ableitungsschritt:

- $S = z_0$
- $N = Z$
- P enthält folgende Regeln:
 - $z \rightarrow az'$ falls $\delta(z, a) = z'$
 - $z \rightarrow \lambda$ falls $z \in E$

EINE WEITERE CHARAKTERISIERUNG VON RLIN:

REGULÄRE AUSDRÜCKE UND SPRACHEN

Wir betrachten nun eine weitere Charakterisierung der regulären Sprachen mit Hilfe von Abschlusseigenschaften, die der Sprachklasse auch den Namen gegeben hat. Wir definieren hierzu induktiv *reguläre Ausdrücke* α und die von diesen dargestellten Sprachen $L(\alpha)$.

REGULÄRE AUSDRÜCKE

Die *regulären Ausdrücke* α über dem Alphabet Σ sind wie folgt definiert und stellen folgende Sprachen $L(\alpha)$ dar :

\emptyset	$L(\emptyset) = \emptyset$
$a \in \Sigma$	$L(a) = \{a\}$
$(\alpha\beta)$	$L((\alpha\beta)) = L(\alpha)L(\beta)$
$(\alpha \cup \beta)$	$L((\alpha \cup \beta)) = L(\alpha) \cup L(\beta)$
α^*	$L(\alpha^*) = L(\alpha)^*$

wobei α, β reguläre Ausdrücke sind.

Eine Sprache $L \subseteq \Sigma^*$ heißt *regulär*, wenn sie durch einen regulären Ausdruck über Σ dargestellt werden kann. REG (REG_Σ) ist die Klasse der regulären Sprachen (über Σ).

BEISPIELE

1. $L_1 = \{0^m 1^n : m, n \geq 0\} = \{0\}^* \{1\}^*$ wird durch $\alpha_1 = (0^* 1^*)$ dargestellt.

2. $L_2 = \{0^m 1^n : m, n \geq 1\} = \{0\} \{0\}^* \{1\} \{1\}^*$ wird durch $\alpha_2 = ((00^*)(11^*))$ dargestellt.

3. $L_3 = \{w \in \Sigma_2^* : |w| \text{ gerade}\} = \{00, 01, 10, 11\}^* = (\{00\} \cup \{01\} \cup \{10\} \cup \{11\})^*$ wird durch $\alpha_3 = (((00) \cup (01)) \cup (10)) \cup (11))^*$ dargestellt.

4. $L_4 = \{w \in \Sigma_2^* : x \text{ Teilwort von } w\} = \Sigma^* \{x\} \Sigma^* = (\{0\} \cup \{1\})^* \{x\} (\{0\} \cup \{1\})^*$ wird durch $\alpha_4 = ((0 \cup 1)^* x) (0 \cup 1)^*$ dargestellt.

Da Verkettung und Vereinigung assoziativ sind, lassen wir in der Regel unnötige Klammern in den Ausdrücken weg:

$\alpha_1 = 0^* 1^*$, $\alpha_2 = 00^* 11^*$, $\alpha_3 = ((00) \cup (01) \cup (10) \cup (11))^*$, $\alpha_4 = (0 \cup 1)^* x (0 \cup 1)^*$.

SATZ. Die Klasse der regulären Sprachen ist die kleinste Sprachklasse, die die endlichen Sprachen enthält und gegen Vereinigung, Verkettung und Iteration (*-Operator) abgeschlossen ist.

BEWEIS. Nach Definition ist REG die kleinste Klasse, die die Sprachen \emptyset und $\{a\}$ ($a \in \Sigma$) enthält und gegen Vereinigung, Verkettung und Iteration abgeschlossen ist. Es genügt also folgendes Lemma zu beweisen:

LEMMA. Endliche Sprachen sind regulär.

BEWEIS DES LEMMAS. Sei $L = \{w_1, \dots, w_n\} \subseteq \Sigma^*$ eine endliche Sprache. Wir zeigen die Regularität von L durch Induktion nach der Kardinalität n von L .

$n = 0$. Dann ist $L = \emptyset$, d.h. $L = L(\alpha)$ für $\alpha = \emptyset$.

$n = 1$. Dann ist $L = \{w\}$. Wir zeigen die Regularität von L durch eine Nebeninduktion nach der Länge m von $w = a_1 \dots a_m$.

$m = 0$. Dann ist $L = \{\lambda\} = L(\emptyset^*)$.

$m = 1$. Dann ist $L = \{a\} = L(a)$.

$m \geq 2$. Dann ist $L = \{a_1 \dots a_m\} = \{a_1\} \dots \{a_m\}$. Die Behauptung folgt also aus der NIV und dem Abschluss von REG unter Verkettung.

$n \geq 2$. Dann ist $L = \{w_1, \dots, w_n\} = \{w_1\} \cup \dots \cup \{w_n\}$. Die Behauptung folgt also aus der HIV und dem Abschluss von REG unter Vereinigung.

ÄQUIVALENZ VON REGULARITÄT UND RECHTSLINEARITÄT

SATZ. $RLIN = REG$.

BEWEIS: Wegen $RLIN = DEA$ genügt es die beiden folgenden Inklusionen zu zeigen:

$$REG \subseteq RLIN$$

$$DEA \subseteq REG$$

REG \subseteq RLIN:

Da REG die kleinste Klasse ist, die die endlichen Sprachen enthält und gegen Vereinigung, Verkettung und Iteration abgeschlossen ist, genügt es folgendes zu zeigen:

- Jede endliche Sprache $L = \{w_1, \dots, w_n\}$ ($n \geq 0$) ist rechtlinear.
 L lässt sich mittels der folgenden Regeln erzeugen: $S \rightarrow w_1 \mid w_2 \mid \dots \mid w_n$
- RLIN ist gegen Vereinigung abgeschlossen. Dies haben wir bereits gezeigt.
- RLIN ist gegen Verkettung abgeschlossen. D.h. sind L_1 und L_2 rechtslinear, so auch L_1L_2 .

Wähle rechtslin. Grammatiken $G_i = (N_i, T_i, P_i, S_i)$ in Chomsky-Normalform, die L_i erzeugen, wobei o.B.d.A. $N_1 \cap N_2 = \emptyset$. Ferner gelte $\lambda \notin L_1$ ($\lambda \in L_1$ s.u.).
Dann erzeugt $G = (N_1 \cup N_2, T_1 \cup T_2, P, S_1)$ die Sprache L_1L_2 , wobei

$$P = \{X \rightarrow aY : X \rightarrow aY \in P_1\} \cup \{X \rightarrow aS_2 : X \rightarrow a \in P_1\} \cup P_2$$

(Für $\lambda \in L_1$ nimmt man noch die Regel $S_1 \rightarrow S_2$ hinzu.)

- RLIN ist gegen Iteration abgeschlossen.

Dies zeigt man ähnlich wie den Abschluss gegen Verkettung.

DEA \subseteq REG:

Sei $M = (\Sigma, Z, \delta, z_0, F)$ ein det. endlicher Automat. Wir müssen zeigen, dass $L(M)$ regulär ist. Wir benutzen hierzu die Tatsache, dass REG alle endlichen Sprachen enthält und gegen Vereinigung, Verkettung und Iteration abgeschlossen ist.

O.B.d.A. können wir annehmen, dass $Z = \{0, \dots, p\}$, wobei 0 der Startzustand ist.

Wir definieren für $i, j, k \leq p$

$$L_{i,j}^k = \{a_1 \dots a_n \in \Sigma^* : n \geq 0 \ \& \ \exists q_2, \dots, q_n \leq k [$$

$$\delta(i, a_1) = q_2 \ \& \ \delta(q_m, a_m) = q_{m+1} \ (2 \leq m < n) \ \& \ \delta(q_n, a_n) = j]\}$$

D.h. $L_{i,j}^k$ enthält alle Wörter $w \in \Sigma^*$, die den Automaten M vom Zustand i in den Zustand j führen, wobei die zwischendurch durchlaufenen Zustände alle $\leq k$ sind.

Wegen

$$L(M) = \bigcup_{j \in E} L_{0,j}^p$$

genügt es $L_{i,j}^k \in \text{REG}$ (für alle $i, j, k \leq p$) zu zeigen (wegen Abschluss von REG gegen \cup).

Wir tun dies induktiv nach k (simultan für alle $i, j \leq p$).

$k = 0$.

$$L_{i,j}^0 = \begin{cases} \{a \in \Sigma : \delta(i, a) = j\} & \text{falls } i \neq j \\ \{a \in \Sigma : \delta(i, a) = j\} \cup \{\lambda\} & \text{falls } i = j \end{cases}$$

In jedem Fall ist $L_{i,j}^0$ endlich, also regulär.

$k \rightarrow k + 1$.

Es gilt $w \in L_{i,j}^{k+1}$ falls

- $w \in L_{i,j}^k$ oder
- $w = w_1 \dots w_m$ ($m \geq 2$), wobei

$$w_1 \in L_{i,k+1}^k \ \& \ w_l \in L_{k+1,k+1}^k \ (2 \leq l < m) \ \& \ w_m \in L_{k+1,j}^k$$

Also:

$$L_{i,j}^{k+1} = L_{i,j}^k \cup L_{i,k+1}^k (L_{k+1,k+1}^k)^* L_{k+1,j}^k$$

Hiermit folgt die Regularität von $L_{i,j}^{k+1}$ aus der Induktionsvoraussetzung wegen Abschluss von REG gegen Vereinigung, Verkettung und Iteration.

ÄQUIVALENZSATZ FÜR RLIN

RECHTSLINEAR vs. LINKSLINEAR

Aus der Äquivalenz $RLIN = REG$ ergibt sich leicht eine weitere Charakterisierung der rechtslinearen Sprachen:

SATZ. $RLIN = LLIN$

BEWEISIDEE. Man zeigt folgendes:

(1) Die Klasse der regulären Sprachen ist gegen Spiegelbilder abgeschlossen, d.h. für reguläres $L \subseteq \Sigma$ ist $L^R = \{w^R : w \in \Sigma\}$ ebenfalls regulär. Man zeigt dies durch eine einfache Induktion nach dem Aufbau der regulären Ausdrücke.

(2) Unter Verwendung der Chomsky-Normalform für die rechtslinearen Grammatiken und die entsprechende, symmetrische Chomsky-Normalform für die linkslinearen Grammatiken beobachtet man, dass

$$L \in RLIN \Leftrightarrow L^R \in LLIN$$

gilt.

ÄQUIVALENZSATZ FÜR DIE RECHTSLINEAREN SPRACHEN

Insgesamt erhalten wir also folgende äquivalenten Charakterisierungen der rechtslinearen Sprachen:

$$\text{RLIN} = \text{LLIN} = \text{DEA} = \text{NEA} = \text{REG}$$

Diese Äquivalenzen sind effektiv, d.h. aus der gegebenen Darstellung einer Sprache eines Typs kann man effektiv Darstellungen der Sprache von den anderen Typen erhalten.

Weitere - z.T. rein algebraische - Charakterisierungen von RLIN werden in der Vorlesung Formale Sprachen betrachtet.

ABSCHLUSSEIGENSCHAFTEN VON RLIN

Die Klasse der regulären Sprachen hat sehr starke Abschlusseigenschaften. Z.B. ist RLIN abgeschlossen gegen

- (1) Vereinigung
- (2) Durchschnitt
- (3) Komplement
- (4) Verkettung
- (5) Iteration
- (6) Homomorphe Bilder

(1), (4) und (5) folgen unmittelbar aus $RLIN = REG$ und der Definition der regulären Sprachen; (3) ergibt sich aus $RLIN = DEA$, da man durch Vertauschen der End- und Nichtendzustände eines vollständigen det. endl. Automaten M einen Automaten M' erhält, der das Komplement der von M erkannten Sprache erkennt; (2) folgt aus (1) und (3) mit den DeMorganschen Regeln. (6) zeigt man wie bei den kontextfreien Sprachen (unter Verwendung der Chomsky-Normalform von RLIN).

Dabei gelten die Abschlusseigenschaften effektiv. Z.B. lässt sich aus rechtlinearen Grammatiken G_1 und G_2 für Sprachen L_1 und L_2 effektiv eine Grammatik G für die Vereinigung $L_1 \cup L_2$ gewinnen.

ÜBERSICHT ABSCHLUSSEIGENSCHAFTEN

Wir können hiermit unsere Tabelle über die Abschlusseigenschaften der Sprachen der (erweiterten) Chomsky-Hierarchie vervollständigen:

	\cup	\cap	Komplement	Verkettung	Iteration	hom. Bild
CH	+	+	–	+	+	+
REK	+	+	+	+	+	–
KS	+	+	+	+	+	–
KF	+	–	–	+	+	+
LIN	+	–	–	–	–	+
RLIN	+	+	+	+	+	+

ENTSCHEIDBARKEITSFragen FÜR RLIN

Da der Übergang von einer rechtslinearen Grammatik zu einem äquivalenten nd. oder det. endlichen Automaten oder zu einem äquivalenten regulären Ausdruck effektiv ist, sind die Entscheidungsfragen für

rechtslineare Grammatiken

det. endliche Automaten

nd. endliche Automaten

reguläre Ausdrücke

äquivalent.

Im Gegensatz zu den mächtigeren Sprachklassen sind hier die üblichen Entscheidungsprobleme alle lösbar, d.h. entscheidbar.

DAS WORTPROBLEM FÜR RLIN IST ENTSCHEIDBAR

GEGEBEN: Rechtlineare Grammatik $G = (N, T, P, S)$ und $w \in T^*$

FRAGE: Gilt $w \in L(G)$?

LÖSUNG: Konvertiere G in einen äquivalenten DEA M und simuliere M bei Eingabe w . Gebe *JA* aus, falls M akzeptiert. Sonst: *NEIN*.

ZEITKOMPLEXITÄT: $|w|$ (=linear) für G fest.

DAS LEERHEITS-, ENDLICHKEITS- UND UNENDLICHKEITSPROBLEM FÜR RLIN IST ENTSCHEIDBAR

Diese Fragen lassen sich wie im kontextfreien Fall mit Hilfe des Pumpinglemmas auf das Wortproblem zurückführen.

Da im rechtslinearen Fall die Konstante p im Pumpinglemma linear beschränkt ist, sind die (naiven) Verfahren hier jedoch schneller (aber immer noch exponentiell). Durch Verfeinerungen kann man diese Verfahren in Polynomial-Zeit-Verfahren überführen.

DAS TOTALITÄTSPROBLEM FÜR RLIN IST ENTSCHEIDBAR

Da RLIN effektiv gegen Komplement abgeschlossen ist, kann man das Totalitätsproblem auf das Leerheitsproblem reduzieren.

DAS ÄQUIVALENZPROBLEM FÜR RLIN IST ENTSCHEIDBAR

GEGEBEN: Rechtlineare Grammatiken G_1 und G_2

FRAGE: Gilt $L(G_1) = L(G_2)$?

LÖSUNG: Wegen der Effektivität der Abschlusseigenschaften von RLIN können wir aus G_1 und G_2 eine rechtslineare Grammatik G konstruieren, die

$$\begin{aligned}L(G_1) \Delta L(G_2) &= [L(G_1) \setminus L(G_2)] \cup [L(G_2) \setminus L(G_1)] \\ &= [L(G_1) \cap \overline{L(G_2)}] \cup [L(G_2) \cap \overline{L(G_1)}]\end{aligned}$$

erzeugt. Es gilt dann aber $L(G_1) = L(G_2)$ genau dann, wenn $L(G)$ leer ist. Letzteres können wir aber entscheiden.

ÜBERSICHT UNENTSCHEIDBARE PROBLEME

Wir können hiermit unsere Tabelle über die Abschlusseigenschaften der Sprachen der (erweiterten) Chomsky-Hierarchie vervollständigen:

	Wortp.	Leerheit	Endlichkeit	Unendlichkeit	Totalität	Äquivalenz
CH	<i>U</i>	<i>U</i>	<i>U</i>	<i>U</i>	<i>U</i>	<i>U</i>
KS	<i>E</i>	<i>U</i>	<i>U</i>	<i>U</i>	<i>U</i>	<i>U</i>
KF	<i>E</i>	<i>E</i>	<i>E</i>	<i>E</i>	<i>U</i>	<i>U</i>
RLIN	<i>E</i>	<i>E</i>	<i>E</i>	<i>E</i>	<i>E</i>	<i>E</i>

U = unentscheidbar, E = entscheidbar