
21. Termersetzungssysteme und Chomsky-Grammatiken

In der Linguistik beschreibt man die Syntax von Sprachen mit Hilfe von Grammatiken. Definiert man eine Sprache als die Menge ihrer Sätze, so kann man eine Grammatik als Regelwerk zur Beschreibung der syntaktisch korrekten Sätze der Sprache, d.h. der Syntax der Sprache, auffassen: Durch (eventuell mehrfache) Anwendung von Regeln der Grammatik lässt sich die syntaktische Korrektheit eines gegebenen Satzes nachweisen (Verifikation, Akzeptor), oder – alternativ – lassen sich durch Ausführung aller möglichen Anwendungsfolgen der Regeln alle korrekten Sätze der Sprache erzeugen (Aufzählung, Erzeugendensystem, generative Grammatik). Mathematisch ist eine Grammatik eine induktive Definition einer Sprache, insbesondere eine Darstellung der Sprache im Sinne von Abschnitt 1. In der Theorie der formalen Sprachen betrachtet man verschiedene Typen von Grammatiken für beliebige formale Sprachen (im Sinne von Definition 1.11, d.h. für beliebige Wortmengen) und vergleicht die hierdurch gegebenen Sprachdarstellungen bezüglich ihrer Mächtigkeit und Qualität. Hierbei stehen generative Grammatiken, d.h. Grammatiken als Erzeugendensysteme im Vordergrund. Korrespondierende Verifikationsmethoden oder Akzeptoren werden durch Angabe äquivalenter Maschinenkonzepte gegeben.

Den formalen Beschreibungen (generativer) Grammatiken liegen mathematisch *Termersetzungssysteme* (oder *Semi-Thue-Systeme*) zugrunde. Ein Termersetzungssystem basiert auf einem Alphabet Σ und besteht aus einer endlichen Menge von *Regeln* (oder *Produktionen*) $r = (u, v)$, bestehend aus Wortpaaren, wobei u die *Prämisse* und v die *Konklusion* der Regel ist. Die Regel r ist auf ein Wort w anwendbar, wenn w die Prämisse u von r als Teilwort enthält. Die Anwendung der Regel auf $w = xuy$ bewirkt, dass das Vorkommen von u in w an der gewählten Stelle durch die Konklusion v der Regel ersetzt wird, also w in xvy übergeht. (Es werden also nicht alle Vorkommen von u simultan durch v ersetzt (was bei Überlappen der Vorkommen auch mehrdeutig sein könnte), sondern nur ein ausgewähltes Vorkommen.) Man kann ein Wort w auf diese Weise solange verändern, bis schließlich keine Regel mehr anwendbar ist, d.h. man eine *Normalform* von w erreicht hat. (Da möglicherweise verschiedene Regeln in einem Schritt anwendbar sind, ist die Normalform i. Allg. nicht eindeutig bestimmt. Auch gibt es i. Allg. Wörter, die keine Normalform besitzen.) Eine mögliche Darstellung einer Sprache ist die Beschreibung der Wörter der Sprache als Normalformen eines Termersetzungssystems. Bevor wir dies an einem Beispiel demonstrieren, definieren wir Termersetzungssysteme und ihre Arbeitsweise noch einmal etwas formaler.

21.1 DEFINITION. Ein *Termersetzungssystem* oder *Semi-Thue-System* ist ein Paar $E = (\Sigma, P)$ bestehend aus einem Alphabet Σ und einer endlichen Relation $P \subseteq \Sigma^* \times \Sigma^*$. Die Elemente $r = (u, v)$ von P heißen die *Regeln* (oder *Produktionen*) von E , wobei u die *Prämisse* und v die *Konklusion* von r ist.

Für eine Regel (u, v) schreiben wir meist $u \rightarrow v$. Die Semantik eines Termersetzungs-systems lässt sich durch ein Umformungssystem (s. Definition 3.1) beschreiben.

21.2 DEFINITION. Das zu einem Termersetzungs-system $E = (\Sigma, P)$ gehörende Umformungssystem U_E ist gegeben durch $U_E = (\Sigma^*, \xrightarrow[E]{})$, wobei

$$\forall w, w' \in \Sigma^* (w \xrightarrow[E]{=} w' \Leftrightarrow \exists (u, v) \in P \exists x, y \in \Sigma^* (w = xuy \ \& \ w' = xvy)).$$

Mit der in Abschnitt 3 eingeführten Terminologie zur Arbeitsweise von Umformungssystemen können wir dann weiter definieren.

21.3 DEFINITION. Sei $E = (\Sigma, P)$ ein Termersetzungs-system und $U_E = (\Sigma^*, \xrightarrow[E]{})$ das zugehörige Umformungssystem. Eine *Herleitung* von w' aus w (der *Länge* n) ist eine mit w beginnende und mit w' endende U_E -Rechnung (der Länge n). w' ist aus w *herleitbar*, wenn es eine Herleitung von w' aus w gibt, d.h. falls $w \xrightarrow[E]{*} w'$ gilt. Ein Wort w' ist eine *Normalform*, falls w' eine Stoppkonfiguration von U_E ist. Gilt zusätzlich, dass w' aus w herleitbar ist, so heißt w' *Normalform von w* .

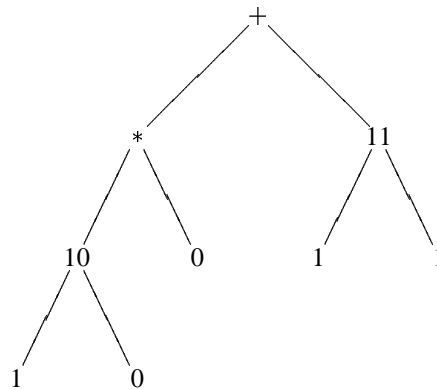
21.4 BEISPIEL. Als Beispiel geben wir ein Termersetzungs-system zur Beschreibung der korrekt geklammerten, variablenfreien arithmetischen Ausdrücke über den Binärzahlen an. Induktiv lassen sich diese Terme definieren durch:

- (T1) Jede Binärzahl ist ein Term.
- (T2) Sind t_1, t_2 Terme, so ist auch $(t_1 + t_2)$ ein Term.
- (T3) Sind t_1, t_2 Terme, so ist auch $(t_1 * t_2)$ ein Term.

Diese Darstellung ist nicht vollständig, da sie auf die (unendliche Menge der) Binärzahlen zurückgreift. Diese müssen also ebenfalls endlich beschrieben werden, was wiederum induktiv mit Hilfe des (simultan definierten) Konzepts der nichtleeren Binärwörter wie folgt möglich ist:

- (Z1) 0 und 1 sind Binärzahlen und Binärwörter.
- (Z2) Ist w ein Binärwort, so sind $0w$ und $1w$ ebenfalls Binärwörter und $1w$ eine Binärzahl.

Zum Nachweis, dass z.B. $t = ((10 * 0) + 11)$ ein Term ist, zeigt man zunächst, dass nach (Z1) $t_2 = 0$ eine Zahl und 0 und 1 Wörter sind, weshalb $t_1 = 10$ und $t_3 = 11$ nach (Z2) ebenfalls Zahlen sind. Nach (T1) sind also t_1, t_2, t_3 Terme. Nach (T3) ist dann auch $t_4 = (t_1 * t_2) = (10 * 0)$ ein Term und damit schließlich nach (T2) $t = (t_4 + t_3)$ ein Term. D.h. man verifiziert die Korrektheit von t , indem man den Strukturbaum



von unten nach oben (bottom up) durchläuft und dabei den Term t aus seinen Teiltermen synthetisiert.

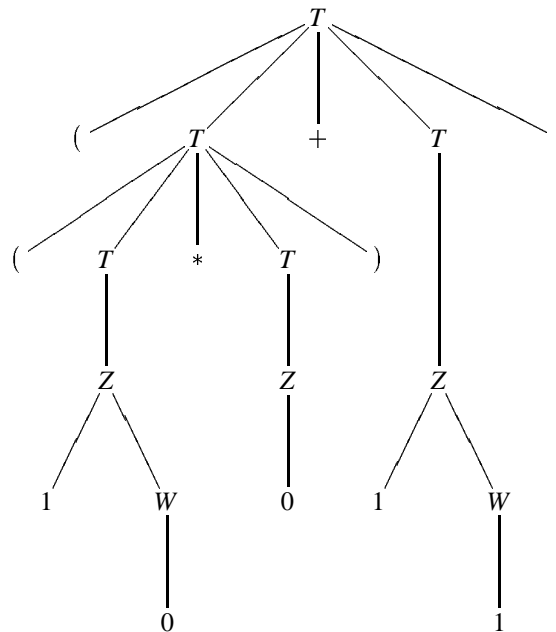
Ein Termersetzungssystem $E = (\Sigma, P)$ zur Beschreibung der Terme benutzt das Alphabet $\Sigma = \{0, 1, +, *, (,), T, Z, W\}$, wobei T, Z, W „Variablen“ für Terme, Zahlen und Wörter sind. Die Regeln entsprechen gerade den Klauseln der induktiven Definition:

(R1)	$T \rightarrow Z$	(T1)
(R2)	$T \rightarrow (T + T)$	(T2)
(R3)	$T \rightarrow (T * T)$	(T3)
(R4)	$Z \rightarrow 0$	(Z1)
(R5)	$Z \rightarrow 1$	(Z1)
(R6)	$W \rightarrow 0$	(Z1)
(R7)	$W \rightarrow 1$	(Z1)
(R8)	$W \rightarrow 0W$	(Z2)
(R9)	$W \rightarrow 1W$	(Z2)
(R10)	$Z \rightarrow 1W$	(Z2)

Man zeigt dann, dass die Normalformen von W , Z und T gerade die nichtleeren Binärwörter, die Binärzahlen und die Terme sind. Eine Herleitung des Terms T von oben erhält man, indem man den Strukturbaum von oben nach unten (top down) durchläuft und die entsprechenden Regeln anwendet:

$$\begin{aligned}
 \underline{T} &\Rightarrow (\underline{T} + T) && \text{(R2)} \\
 &\Rightarrow ((\underline{T} * T) + T) && \text{(R3)} \\
 &\Rightarrow ((Z * \underline{T}) + T) && \text{(R1)} \\
 &\Rightarrow ((Z * Z) + \underline{T}) && \text{(R1)} \\
 &\Rightarrow ((\underline{Z} * Z) + Z) && \text{(R1)} \\
 &\Rightarrow ((1W * \underline{Z}) + Z) && \text{(R10)} \\
 &\Rightarrow ((1W * 0) + \underline{Z}) && \text{(R4)} \\
 &\Rightarrow ((1\underline{W} * 0) + 1W) && \text{(R10)} \\
 &\Rightarrow ((10 * 0) + 1\underline{W}) && \text{(R6)} \\
 &\Rightarrow ((10 * 0) + 11) && \text{(R7)}
 \end{aligned}$$

(Die Stelle, an der die angegebene Regel angewendet wird, ist jeweils unterstrichen.) Man kann diese Herleitung auch durch einen Baum darstellen, da die Prämissen nur aus einem Buchstaben bestehen. Dabei enthalten die Söhne die Konklusion einer Regel, der Vater deren Prämisse



Die Blätter von links nach rechts gelesen ergeben dann das hergeleitete Wort. In seiner Grundstruktur entspricht dieser Herleitungsbaum dem Strukturbaum von t .

In diesem Beispiel wurden neben den Grundzeichen, die in den Termen vorkommen, Variablen hinzugenommen, die für das gewünschte syntaktische Objekt (T) bzw. benötigte Hilfsobjekte (W, Z) stehen. Weiter ist man nur an Herleitungen von Wörtern interessiert, die diese Variablen nicht mehr enthalten, wobei die Herleitungen mit der Variablen T beginnen. Da man ähnliche Beobachtungen allgemein macht, hat Chomsky den Begriff des Termersetzungssystems entsprechend modifiziert, und so den für die Sprachtheorie grundlegenden, folgenden Grammatikbegriff geprägt.

21.5 DEFINITION. Eine (*Chomsky-*)*Grammatik* $G = (N, T, P, S)$ besteht aus zwei disjunkten Alphabeten N und T , einer endlichen Relation $P \subseteq ((N \cup T)^* - T^*) \times (N \cup T)^*$, und einem ausgezeichneten Buchstaben S aus N . Die Elemente aus N bzw. T heißen (*syntaktische*) *Variablen* (oder *Nichtterminalzeichen*) bzw. *Terminalzeichen*, die Elemente aus P *Regeln* oder *Produktionen*, die Variable S *Axiom*.

Von der Prämisse u einer Regel (u, v) verlangt man hier also, dass sie zumindest eine Variable enthält, während die Konklusionen v ein beliebiges (möglicherweise leeres) Wort aus Terminal- und/oder Nichtterminalzeichen ist. Ein nur aus Terminalzeichen bestehendes Wort heißt auch *Terminalwort* (oder *Satz*), ein beliebiges Wort w , das (möglicherweise auch) Variablen enthält *Satzform*. Kommt hierbei in w tatsächlich eine Variable vor, sprechen wir von einer *echten Satzform*.

Einer Grammatik $G = (N, T, P, S)$ kann man das Termersetzungssystem $E_G = E = (N \cup T, P)$ zuordnen und über das zugehörige Umformungssystem U_E dann die Semantik von G erklären. Die für E_G eingeführten Begriffe übertragen sich damit auf G :

21.6 DEFINITION. Das zu der Grammatik $G = (N, T, P, S)$ gehörende Termersetzungssystem $E = E_G$ ist durch $E = (N \cup T, P)$ gegeben. Eine (G)-*Herleitung* von w' aus w

(der Länge n) in G ist eine solche Herleitung in E . Das Wort w' ist aus dem Wort w in G herleitbar (G -herleitbar), in Zeichen $w \xrightarrow{*}_G w'$, falls es eine G -Herleitung von w' aus w gibt (d.h. $w \xrightarrow{*}_E w'$ gilt). Die von G erzeugte Sprache $L(G)$ (über dem Alphabet T) ist die Menge der aus dem Axiom herleitbaren Terminalwörter, d.h. $L(G) = \{w \in T^* : S \xrightarrow{*}_G w\}$.

Ist die Grammatik G aus dem Kontext bekannt, so schreiben wir \Rightarrow statt $\xrightarrow{*}_G$ (etc.). Weiter sagen wir, dass L eine *Chomsky-Sprache* ist, falls L von einer Chomsky-Grammatik erzeugt wird und bezeichnen mit CH (CH_k) die Menge aller Chomsky-Sprachen (über dem k -ären Alphabet).

21.7 BEISPIEL. Das im Beispiel 21.4 angegebene Termerssetzungssystem zur Erzeugung der arithmetischen Terme lässt sich in folgende Grammatik $G = (N, T', P, S)$ überführen: $N = \{T, W, Z\}$, $T' = \{0, 1, +, *, (,)\}$, $S = T$, und P enthält die Regeln (R1)-(R10).

21.8 BEISPIEL. Die Sprache $L = \{0^m 1^n : m, n \geq 1\}$ wird von der Grammatik $G = (N, \{0, 1\}, P, S)$ erzeugt, wobei $N = \{S, T\}$, und P aus den vier Regeln

- (G1) $S \rightarrow 0S$
- (G2) $S \rightarrow 0T$
- (G3) $T \rightarrow 1T$
- (G4) $T \rightarrow 1$

besteht. Um zu zeigen, dass $L = L(G)$ gilt, muss man die Inklusionen $L \subseteq L(G)$ und $L(G) \subseteq L$ zeigen. Zum Nachweis von $L \subseteq L(G)$ muss man $0^m 1^n \in L(G)$ für gegebenes m, n zeigen, d.h. eine Herleitung von $0^m 1^n$ aus S angeben:

$$\begin{array}{lll} S & \xrightarrow{m-1} & 0^{m-1}S & (m-1 \times (\text{G1})) \\ & \Rightarrow & 0^m T & (1 \times (\text{G2})) \\ & \xrightarrow{n-1} & 0^m 1^{n-1} T & (n-1 \times (\text{G3})) \\ & \Rightarrow & 0^m 1^n & (1 \times (\text{G4})) \end{array}$$

Zum Nachweis von $L(G) \subseteq L$ charakterisieren wir (durch Induktion nach k) die in k Schritten aus S herleitbaren Wörter (Sätze und Satzformen):

$$\begin{aligned} S \xrightarrow{k} w & \Rightarrow w = 0^k S \text{ oder} \\ & \exists m \geq 1 \exists n \geq 0 (k = m + n \ \& \ w = 0^m 1^n T) \text{ oder} \\ & \exists m \geq 1 \exists n \geq 1 (k = m + n \ \& \ w = 0^m 1^n) \end{aligned} \quad (21.1)$$

Für terminales w zeigt dies, dass $S \xrightarrow{k} w$ impliziert, dass $w \in L$ gilt. Für $k = 0$ ist die Behauptung klar, da $S \xrightarrow{0} w$ nur für $w = S = 0^0 S$ gilt. Im Induktionsschritt von k nach $k + 1$ betrachtet man den letzten Schritt in der Herleitung $S \xrightarrow{k+1} w$, d.h. $S \xrightarrow{k} w' \Rightarrow w$. Erfolgt dieser vermöge der Regeln (G1) oder (G2), so muss w' die Variable S enthalten, also nach Induktionsvoraussetzung $w' = 0^k S$ gelten. Es ist dann aber $w = 0^{k+1} S$ bzw. $w = 0^{k+1} T$ von der verlangten Gestalt. Wurde die Regel (G3) oder (G4) angewendet, muss entsprechend T in w' vorkommen und daher $w' = 0^m 1^n T$ für geeignete $m \geq 1, n \geq 0$ mit $m + n = k$ gelten. Hier gilt dann $w = 0^m 1^{n+1} T$ oder $w = 0^m 1^{n+1}$, weshalb wegen $m + (n + 1) = k + 1$ das Wort w wieder eine der gewünschten Gestalten hat.

Bei einer Induktion wie in dem vorhergehenden Beispiel spricht man von einer *Herleitungsinduktion*. Die Verifikation, dass eine zur Erzeugung einer Sprache L entworfene Grammatik G diese Sprache tatsächlich erzeugt, d.h. dass $L = L(G)$ gilt, ist oft recht mühselig. Die Richtung $L \subseteq L(G)$ ist dabei meist leichter zu zeigen, da man die Regeln ja gerade so entwirft, um in kanonischer Weise alle Wörter aus L zu erzeugen. Der Nachweis der Umkehrung $L(G) \subseteq L$ ist meist schwieriger. Meist weist man hier durch Herleitungsinduktion gewisse Invarianten in den hergeleiteten Satzformen nach (s.(21.1) im Beispiel oben). Das Problem für diese Richtung ist, dass die Regeln in beliebiger Reihenfolge an beliebiger Stelle ausgeführt werden können, während man bei dem Entwurf zur Ableitung der Wörter in L meist die Regeln in einer gewissen Ordnung benutzt. Man muss daher zeigen, dass eine Verletzung dieser Ordnung nicht die Ableitung unerwünschter zusätzlicher Terminalwörter ermöglicht. (Mögliche „Sackgassen“, d.h. Ableitungen echter Satzformen auf diese Weise, die sich nicht in Terminalwörter weiter ableiten lassen, sind dagegen bedeutungslos.) In den folgenden Beispielen verzichten wir meistens auf den Korrektheitsbeweis.

21.9 BEISPIEL. Die Sprache $L = \{0^n 1^n : n \geq 1\}$ variiert die Sprache in Beispiel 21.8, indem nun zusätzlich gefordert wird, dass der 0-Block und der 1-Block gleiche Länge haben. Diese Sprache wird von der Grammatik $G = (\{S\}, \{0, 1\}, P, S)$ mit der Produktionenmenge

$$\begin{aligned} S &\rightarrow 0S1 \\ S &\rightarrow 01 \end{aligned}$$

erzeugt.

21.10 BEISPIEL. Die Sprache $L = \{0^n 1^n 0^m : n, m \geq 1\}$ wird von der Grammatik G mit Axiom S und Regeln

$$\begin{aligned} S &\rightarrow TU \\ T &\rightarrow 0T1 \\ T &\rightarrow 01 \\ U &\rightarrow 0U \\ U &\rightarrow 0 \end{aligned}$$

erzeugt.

21.11 BEISPIEL. Die Sprache $L = \{0^n 1^n 0^n : n \geq 1\}$ unterscheidet sich vom vorhergehenden Beispiel dadurch, dass alle Blöcke nun gleichlang sind. Hierdurch wird eine Grammatik G , die L erzeugt, recht kompliziert. Die unten angegebene Grammatik basiert auf folgender Idee. Man führt Variablen A, B, C für die Buchstaben der drei Blöcke ein und wählt Regeln zur Durchführung der folgenden Ableitungsschritte:

- I. Erzeuge $w_1 = (ABC)^n$ ($n \geq 1$ beliebig).
- II. Durch Umordnen führe w_1 in $w_2 = A^n B^n C^n$ über.
- III. Durch Substitution der Werte 0,1,0 für die Variablen A, B, C forme w_2 in das gewünschte Terminalwort $w = 0^n 1^n 0^n$ um.

Das Problem ist hierbei, dass sichergestellt sein muss, dass diese Schritte in der angegebenen Reihenfolge ausgeführt werden. Insbesondere sollte Phase III erst nach Abschluss von Phase II ausgeführt werden (wogegen eine zeitliche Mischung von Phase I und Phase II unproblematisch ist). In der folgenden Grammatik G für L erreichen wir dies dadurch, dass wir die Variablen am Anfang und Ende des Wortes besonders markieren (wodurch wir sicherstellen können, dass gewisse Umformungen an einer dieser Stellen beginnen müssen), und wir bereits geordnete Variablen umbenennen (in α, β, γ). Die intendierte Ordnung wird hierbei von rechts nach links hergestellt, so dass eine aus dem Axiom hergeleitete, mit $\underline{\alpha}$ beginnende Satzform geordnet, d.h. für diese Phase II abgeschlossen ist, sodass man nun (von links nach rechts) die Substitutionsphase III beginnen kann. (Bei dem Entwurf der Grammatik war dabei der Leitgedanke, die Verifikation möglichst einfach zu machen, wodurch man etwas mehr Regeln als notwendig benötigt.)

Die Variablenmenge von G ist $N = \{S, T, \underline{A}, A, B, C, \underline{C}, \underline{\alpha}, \alpha, \beta, \gamma, \underline{\gamma}\}$, wobei S das Axiom ist. Zur Herleitung des Wortes 010 führen wir explizit die Regel $S \rightarrow 010$ ein. Herleitungen der Wörter $0^n 1^n 0^n$ mit $n \geq 2$ werden durch folgende Regeln ermöglicht, die wir gemäß den (intendierten) Phasen der Herleitungen gruppieren:

- | | | | |
|---------|--|---------------|--|
| (I.1) | S | \rightarrow | $\underline{A}BC\underline{T}$ |
| (I.2) | T | \rightarrow | $ABC\underline{T}$ |
| (I.3) | T | \rightarrow | $ABC\underline{C}$ |
| — | | | |
| (IIa.1) | BA | \rightarrow | AB |
| (IIa.2) | CA | \rightarrow | AC |
| (IIa.3) | CB | \rightarrow | BC |
| — | | | |
| (IIb.1) | $\underline{C}\underline{C}$ | \rightarrow | $\underline{\gamma}\underline{\gamma}$ |
| (IIb.2) | $C\underline{\gamma}$ | \rightarrow | $\underline{\gamma}\underline{\gamma}$ |
| (IIb.3) | $B\underline{\gamma}$ | \rightarrow | $\beta\underline{\gamma}$ |
| (IIb.4) | $B\underline{\beta}$ | \rightarrow | $\beta\underline{\beta}$ |
| (IIb.5) | $A\underline{\beta}$ | \rightarrow | $\alpha\underline{\beta}$ |
| (IIb.6) | $A\underline{\alpha}$ | \rightarrow | $\alpha\underline{\alpha}$ |
| (IIb.7) | $\underline{A}\underline{\alpha}$ | \rightarrow | $\underline{\alpha}\underline{\alpha}$ |
| — | | | |
| (III.1) | $\underline{\alpha}\underline{\alpha}$ | \rightarrow | 00 |
| (III.2) | $0\underline{\alpha}$ | \rightarrow | 00 |
| (III.3) | $0\underline{\beta}$ | \rightarrow | 01 |
| (III.4) | $1\underline{\beta}$ | \rightarrow | 11 |
| (III.5) | $1\underline{\gamma}$ | \rightarrow | 10 |
| (III.6) | $0\underline{\gamma}$ | \rightarrow | 00 |
| (III.7) | $0\underline{\gamma}$ | \rightarrow | 00 |

Um ein Wort $w = 0^n 1^n 0^n$ mit $n \geq 2$ herzuleiten, geht man dann wie folgt vor:

$$\begin{aligned}
 S &\Rightarrow \underline{A}BC\underline{T} && \text{(I.1)} \\
 &\stackrel{*}{\Rightarrow} \underline{A}BC(ABC)^{n-2}T && \text{(I.2)} \\
 &\Rightarrow \underline{A}BC(ABC)^{n-2}ABC\underline{C} && \text{(I.3)} \\
 &\stackrel{*}{\Rightarrow} \underline{A}A^{n-1}B^n C^{n-1}\underline{C} && \text{(IIa.1)-(IIa.3)} \\
 &\stackrel{*}{\Rightarrow} \underline{\alpha}\alpha^{n-1}\beta^n \gamma^{n-1}\underline{\gamma} && \text{(IIb.1)-(IIb.7)} \\
 &\stackrel{*}{\Rightarrow} 0^n 1^n 0^n && \text{(III.1)-(III.7)}
 \end{aligned}$$

Um umgekehrt zu zeigen, dass $L(G) \subseteq L$ gilt, zeigen wir, dass jede Herleitung H eines Terminalwortes $w \neq 010$ in G im Wesentlichen (d.h. bis auf unkritische Änderungen der Reihenfolge von Regelanwendungen) die oben angegebene Gestalt hat. Sei also eine Herleitung

$$H: S = w_0 \Rightarrow w_1 \Rightarrow \dots \Rightarrow w_m \Rightarrow w_{m+1} = w \in \Sigma_2^* - \{010\}$$

gegeben. Für einen Buchstaben $a \in N \cup \Sigma_2$ bezeichnen wir mit $i(a)$ das kleinste $i \leq m+1$, sodass a in w_i vorkommt (falls existent). Wir behaupten, dass

$$\begin{aligned} 0 = i(S) < i(\underline{A}) = i(T) = 1 < i(\underline{C}) < i(\underline{\gamma}) = \dots \\ \dots = i(\underline{\gamma}) < i(\underline{\beta}) < i(\underline{\alpha}) < i(\underline{0}) < i(\underline{1}) \leq m+1 \end{aligned} \quad (21.2)$$

Hierbei gilt $0 = i(S)$ und $i(\underline{A}) = i(T) = 1$ offensichtlich, da jede Herleitung eines Wortes $w \neq 010$ mit der Regel $S \rightarrow \underline{A}BCT$ beginnen muss, also $w_1 = \underline{A}BCT$ gilt. Insgesamt zeigen wir diese Ungleichungen in (21.2) induktiv von rechts nach links. Dass $i(1) \leq m+1$ ist, folgt dabei aus der Beobachtung, dass für $1 \leq j \leq m+1$ das Wort w_j mindestens einen der Buchstaben B , β oder 1 enthält. (Dies zeigt man induktiv dadurch, dass man beobachtet, dass $w_1 = \underline{A}BCT$ gilt und jede Regel, die B oder β oder 1 in der Prämisse enthält auch einen dieser Buchstaben in der Konklusion enthält.) Hat man für ein benachbartes Paar $i(x), i(y)$ in (21.2) nach Induktionsvoraussetzung $i(y) \leq \dots < i(1) \leq m+1$ schon gezeigt, so folgt $i(x) \leq m+1$ und $i(x) < i(y)$ dadurch, dass man beobachtet, dass alle Regeln mit y in der Konklusion x oder ein Zeichen z mit $i(y) \leq i(z)$ in der Prämisse enthalten, x also vor y erzeugt worden sein muss.

Nach (21.2) hat also die Herleitung H die Gestalt

$$S \Rightarrow w_1 = \underline{A}BCT \xrightarrow{*} w_{i(\underline{C})} \xrightarrow{*} w_{i(\underline{\alpha})} \xrightarrow{*} w$$

Wir zeigen, dass diese Herleitung (hinreichend) mit der oben gegebenen Herleitung von $0^n 1^n 0^n$ übereinstimmt. Hierzu beweisen wir folgende Behauptungen.

Behauptung 1. Es gibt eine Zahl $n \geq 2$ und ein Wort $v \in \{A, B, C\}^*$, sodass $w_{i(\underline{C})} = \underline{A}v\underline{C}$ und

$$\#_B(v) = \#_A(v) + 1 = \#_C(v) + 1 = n$$

BEWEIS. In der Herleitung $\underline{A}BCT = w_1 \Rightarrow w_2 \Rightarrow \dots \Rightarrow w_k \Rightarrow w_{i(\underline{C})}$ können nur Regeln verwendet werden, deren Prämissen nur Variablen X mit $i(X) < i(\underline{C})$ enthalten. Wegen (21.2) trifft dies nur auf die Regeln der Gruppen (I) und (IIa) zu, wobei die Regel (I.1) entfällt, da w_1 S nicht enthält, und S auf keiner rechten Regelseite vorkommt. Weiter wird (nach Definition von $i(\underline{C})$) die Regel (I.3), die als einzige \underline{C} erzeugt, in dieser Herleitung nur im Schritt $w_k \Rightarrow w_{i(\underline{C})}$ (und damit dort notwendigerweise) angewendet. Die Herleitung $w_1 \xrightarrow{*} w_k$ benutzt also höchstens die Regeln (I.2) und (IIa.1)-(IIa.3). Diese überführen aber ein Wort $\underline{A}uT$ mit $u \in \{A, B, C\}^*$, wobei $\#_B(u) = \#_A(u) + 1 = \#_C(u) + 1 \geq 1$, in ein Wort mit derselben Eigenschaft. Mit w_1 hat also auch w_k diese Eigenschaft (Induktion). Da $w_k \Rightarrow w_{i(\underline{C})}$ via (I.3), ergibt sich hieraus die behauptete Gestalt von $w_{i(\underline{C})}$.

Behauptung 2. Es gilt $w_{i(\underline{\alpha})} = \underline{\alpha}\alpha^{n-1}\beta^n\gamma^{n-1}\underline{\gamma}$, wobei $n \geq 2$ die Zahl aus Behauptung 1 ist.

BEWEIS. Entsprechend wie im Beweis der ersten Behauptung zeigt man, dass in der Herleitung

$$w_{i(\underline{C})} = \underline{A}v\underline{C} \Rightarrow w_{i(\underline{C})+1} \xrightarrow{*} w_{i(\underline{C})+l} \Rightarrow w_{i(\underline{\alpha})}$$

nur Regeln aus den Gruppen (IIa) und (IIb) verwendet werden, wobei die Regel (IIb.7) nur im letzten Schritt angewendet wird. Man zeigt dann durch Herleitungsinduktion, dass jedes Wort $z = w_{i(\underline{C})+m}$ ($0 \leq m \leq l$) die Gestalt $\underline{A}u\underline{C}$ oder $\underline{A}u'\underline{\gamma}$ hat, wobei $u \in \{A, B, C\}^*$ und $u' \in \{A, B, C, \alpha, \beta, \gamma\}^*$,

$$\#_A(z) + \#_\alpha(z) + 1 = \#_B(z) + \#_\beta(z) = \#_C(z) + \#_\gamma(z) + 1 = n,$$

und rechts von einem α (bzw. β bzw. γ) in u' nur die Buchstaben α, β, γ (bzw. β, γ bzw. γ) vorkommen. Da im letzten Schritt der Herleitung die Regel (IIb.7) angewendet wird, muss daher $w_{i(\underline{C})+l} = \underline{A}\alpha^{n-1}\beta^n\gamma^{n-1}\underline{\gamma}$ gelten, weshalb $w_{i(\underline{\alpha})}$ die gewünschte Gestalt hat.

Behauptung 3. $w = 0^n 1^n 0^n$, wobei $n \geq 2$ wie oben.

BEWEIS. Für die Herleitung

$$w_{i(\underline{\alpha})} = \underline{\alpha}\alpha^{n-1}\beta^n\gamma^{n-1}\underline{\gamma} \xrightarrow{*} w$$

zeigt man induktiv, dass die hergeleiteten Wörter nur Buchstaben aus dem Alphabet $\{\underline{\alpha}, \alpha, \beta, \gamma, \underline{\gamma}, 0, 1\}$ enthalten, also nur Regeln der Gruppe (III) anwendbar sind. Aufgrund der Gestalt dieser Regeln folgt (wiederum durch Herleitungsinduktion), dass jedes Wort z in der Herleitung von w aus $w_{i(\underline{\alpha})}$ die Gestalt $z = u_A u_B u_C$ hat, wobei $|u_A| = |u_B| = |u_C| = n$, $u_A \in \{\underline{\alpha}, \alpha, 0\}^*$, $u_B \in \{\beta, 1\}^*$ und $u_C \in \{\gamma, \underline{\gamma}, 0\}^*$. Da w terminal ist, gilt also $w = 0^n 1^n 0^n$ wie behauptet.

Wir wollen nun die Mächtigkeit des Grammatik-Konzeptes untersuchen, indem wir den Umfang der Klasse CH der Chomsky-Sprachen bestimmen. Hierzu beobachten wir zunächst, dass wegen ihrer Lokalität die Anwendungen von Regeln einer Grammatik G effektiv ausführbar sind. Hieraus folgt, dass die von G erzeugte Sprache effektiv aufgezählt werden kann, also nach Churchscher These rekursiv aufzählbar ist.

21.12 LEMMA. *Sei G eine Chomsky-Grammatik. Dann ist $L(G)$ rekursiv aufzählbar. Darüberhinaus lässt sich aus G effektiv eine Turingmaschine M angeben, die $L(G)$ akzeptiert¹.*

BEWEIS. Sei $G = (N, T, P, S)$. Eine Herleitung eines Wortes $w \in T^*$ in G hat die Form

$$S \Rightarrow v_1 \Rightarrow v_2 \Rightarrow \dots \Rightarrow v_n \Rightarrow w \quad (n \geq 0, v_i \in (N \cup T)^*),$$

kann also als Wort über dem Alphabet $\Sigma = N \cup T \cup \{\Rightarrow\}$ aufgefasst werden. Da wir für Wörter v und v' durch Mustervergleich überprüfen können, ob v durch Anwendung einer Regel in v' überführbar ist, also $v \Rightarrow v'$ gilt, kann man effektiv feststellen, ob ein

¹Wie schon in der Komplexitätstheorie akzeptiert M eine rekursiv aufzählbare Sprache L , wenn L die Menge der Eingaben ist, für die die Rechnung von M in einem Endzustand endet (wobei die Menge E der Endzustände zusätzlich zur früheren Definition der Turingmaschine M zu spezifizieren ist). Dies ist offensichtlich äquivalent zu der früheren Definition, nach der M eine Eingabe x akzeptiert, falls M für diese eine Ausgabe liefert. Durch Zusammenführen der Endzustände können wir bei Bedarf von der Existenz eines einzigen akzeptierenden Zustandes + ausgehen.

Wort $z \in \Sigma^*$ eine Herleitung beschreibt, und im positiven Fall lässt sich das hergeleitete Wort effektiv bestimmen. D.h. die Menge

$$H = \{(w, z) : w \in T^*, z \in \Sigma^*, z \text{ Herleitung von } w\}$$

ist entscheidbar. Entweder durch Rückgriff auf die Church-Turing-These oder durch eine einfache Formalisierung obiger Überlegungen folgt hieraus, dass H rekursiv ist. Da

$$w \in L(G) \Leftrightarrow \exists z((w, z) \in H)$$

gilt, ist also $L(G)$ nach dem Projektionslemma rekursiv aufzählbar. Der zweite Teil des Lemmas folgt aus der Effektivität des Argumentes. D.h. eine Turingmaschine, die (die charakteristische Funktion von) H berechnet, lässt sich effektiv aus G gewinnen, und aus dieser wiederum ein Akzeptor für die Projektion $L(G)$ von H . \square

Umgekehrt haben wir schon vorher beobachtet, dass die Konfigurationen einer Turingmaschine M durch Wörter beschrieben werden können. Wegen der Lokalität der Turingmaschinen-Operationen können die Übergänge durch Regeln beschrieben werden, weshalb M als Termersetzungssystem aufgefasst werden kann. Diese Beobachtung erlaubt uns, die von M akzeptierte Sprache durch eine Chomsky-Grammatik zu erzeugen, und damit die Umkehrung von Lemma 21.12 zu zeigen.

21.13 LEMMA. *Sei M eine deterministische (1-Band-)Turingmaschine, und sei $L(M)$ die von M akzeptierte Sprache. Dann lässt sich effektiv eine Chomsky-Grammatik G angeben, die $L(M)$ erzeugt.*

BEWEIS. Seien Σ, Γ das Eingabe- bzw. Bandalphabet, $Z = \{z_0, z_1, \dots, z_p\}$ die Zustandsmenge und P das Programm von M . Dabei können wir o.B.d.A. davon ausgehen, dass $\Gamma \cap Z = \emptyset$, z_0 der Startzustand und z_1 der einzige akzeptierende Endzustand von M ist. Weiter bestehe P aus bedingten Anweisungen.

Um M nun durch eine Grammatik G zu simulieren, beobachten wir, dass die Rechenschritte, d.h. Konfigurationsübergänge, von M durch Regeln beschrieben werden können, dass das Herleitungs- bzw. Rechnungsziel bei Maschinen und Grammatiken aber unterschiedlich ist. Bei der Maschine wird, grob gesprochen, ein Wort w aus der akzeptierten Sprache durch Regelanwendungen in den akzeptierenden Zustand z_1 überführt. (Genauer: die zu w gehörende Startkonfiguration wird in eine Endkonfiguration überführt. Wir können jedoch annehmen, dass M bei der Endkonfiguration das Band noch vollständig löscht, es also genau eine – im Wesentlichen nur aus dem Zustand z_1 bestehende – Endkonfiguration gibt.) Bei der Grammatik ist das Vorgehen im Wesentlichen umgekehrt: Ein ausgezeichneter Buchstabe wird durch Regelanwendungen in ein Wort der erzeugten Sprache überführt.

Eine Möglichkeit, M durch eine Grammatik G zu beschreiben, ist daher, die Endkonfiguration von M als Axiom zu wählen und die Übergänge von einer Konfiguration zurück zu ihrer Vorgängerkonfiguration durch die Regeln von G zu ermöglichen. (D.h. die Regeln von G sind invers zu den „Regeln“ von M .)

Will man die Rechnung nicht invertieren, so kann man alternativ mit einer Spurentechnik arbeiten: Zunächst erzeugt G ein beliebiges Wort $w \in \Sigma^*$. In Spur 1 wird dieses bewahrt, in den Spuren 2 und 3 dann die zugehörige Rechnung von M simuliert (Spur 2 = Bandinschrift, Spur 3 = Positionen des Arbeitsfeldes und Zustand). Akzeptiert M , so

werden die Spuren 2 und 3 gelöscht und Spur 1 als Terminalwort erzeugt. Im Folgenden formalisieren wir diesen zweiten Ansatz.

Die Produktionenmenge P' der Grammatik $G = (N, \Sigma, P', S)$, die $L(M)$ erzeugt, besteht aus den folgenden drei Gruppen:

Die erste Gruppe besteht aus Regeln zur Erzeugung eines Terminalwortes $w \in \Sigma^*$ in der ersten Spur und der zugehörigen Startkonfiguration in der zweiten und dritten Spur. Diese Satzform wird zusätzlich durch die Randzeichen [und] geklammert:

$$\begin{array}{c}
 * \\
 S \rightarrow [\begin{array}{c} B \\ z_0 \end{array} T \\
 * \\
 T \rightarrow \begin{array}{c} a \\ a T \end{array} |] \\
 *
 \end{array}$$

(Die aus drei Komponenten bestehenden Variablen aus $(\Sigma \cup \{*\}) \times \Gamma \times (Z \times \{*\})$ schreiben wir als Spaltenvektoren.) Für jedes Wort $w = a_1 \dots a_n \in \Sigma^*$ ($n \geq 0$) lässt sich hiermit

$$S \xrightarrow{*} [\begin{array}{c} * \\ B \\ z_0 \end{array} \begin{array}{ccc} a_1 & \dots & a_n \\ a_1 & \dots & a_n \\ * & \dots & * \end{array}]$$

herleiten.

Die zweite Gruppe von Regeln erlaubt (bei Bedarf), den beschriebenen Bandteil am Rande durch Anfügen von Blanks zu erweitern:

$$\begin{array}{c}
 * \\
 [\rightarrow [\begin{array}{c} B \\ * \end{array} \\
 * \\
] \rightarrow \begin{array}{c} * \\ B \\ * \end{array}] \\
 *
 \end{array}$$

Die Schritte von M lassen sich dann durch die Regeln der 3. Gruppe simulieren, die für jede M -Instruktion $I = (z, a, a', B, z')$ die folgenden, von der Bewegung $B \in \{L, R, S\}$ abhängenden Regeln enthält (für alle $\alpha, \alpha' \in \Sigma \cup \{*\}, a'' \in \Gamma$):

$$\begin{array}{ccc}
 \alpha & & \alpha \\
 a & \rightarrow & a' \\
 z & & z'
 \end{array} \quad (\text{falls } B = S)$$

$$\begin{array}{ccc}
 \alpha & \alpha' & \alpha & \alpha' \\
 a & a'' & \rightarrow & a' & a'' \\
 z & * & & * & z'
 \end{array} \quad (\text{falls } B = R)$$

$$\begin{array}{ccc}
 \alpha' & \alpha & \alpha' & \alpha \\
 a'' & a & \rightarrow & a'' & a' \\
 * & z & & z' & *
 \end{array} \quad (\text{falls } B = L)$$

Gilt $w = a_1 \dots a_n \in L(M)$, so kann man mit den Regeln der zweiten und dritten Gruppe

$$\begin{array}{c}
 * \\
 [\begin{array}{c} B \\ a_1 \\ z_0 \end{array} \begin{array}{ccc} a_1 & \dots & a_n \\ a_1 & \dots & a_n \\ * & \dots & * \end{array}] \xrightarrow{*} [\begin{array}{c} \alpha_1 \\ \beta_1 \\ * \end{array} \begin{array}{ccc} \dots & & \dots \\ \dots & & \dots \\ * & \dots & * \end{array} \begin{array}{c} \alpha_{i-1} \\ \beta_{i-1} \\ * \end{array} \begin{array}{c} \alpha_i \\ \beta_i \\ z_1 \end{array} \begin{array}{c} \alpha_{i+1} \\ \beta_{i+1} \\ * \end{array} \begin{array}{ccc} \dots & & \dots \\ \dots & & \dots \\ * & \dots & * \end{array} \alpha_m] \\
 *
 \end{array}$$

zeigen, wobei $\alpha_1 \dots \alpha_m = * \dots * w * \dots *$ und $\beta_1 \dots \beta_m \in \Gamma^*$ geeignet zu wählen ist.

Die Regeln der vierten Gruppe erlauben eine den Endzustand z_1 enthaltende Satzform v auf das Terminalwort in der ersten Spur von v zu reduzieren ($\alpha, \alpha' \in \Sigma \cup \{*\}$, $\beta \in \Gamma$):

$$\begin{array}{l} \alpha \\ \beta \quad \rightarrow \quad \alpha \\ z_1 \end{array}$$

$$\begin{array}{l} \alpha \\ \beta \alpha' \quad \rightarrow \quad \alpha \alpha' \\ * \end{array}$$

$$\begin{array}{l} \alpha \\ \alpha' \beta \quad \rightarrow \quad \alpha' \alpha \\ * \end{array}$$

$$\begin{array}{l} * \quad \rightarrow \quad \lambda \\ [\quad \rightarrow \quad \lambda \\] \quad \rightarrow \quad \lambda \end{array}$$

Hiermit kann man also für $\alpha_1 \dots \alpha_m$ und $\beta_1 \dots \beta_m$ wie oben zunächst mit den ersten drei, dann mit den letzten drei Regeln der vierten Gruppe

$$\begin{array}{cccccccc} \alpha_1 & \dots & \alpha_{i-1} & \alpha_i & \alpha_{i+1} & \dots & \alpha_m \\ [\beta_1 & \dots & \beta_{i-1} & \beta_i & \beta_{i+1} & \dots & \beta_m] \xrightarrow{*} [\alpha_1 \dots \alpha_m] \xrightarrow{*} w \\ * & \dots & * & z_1 & * & \dots & * \end{array}$$

herleiten.

Aufgrund der Bemerkungen zu den Regelgruppen lässt sich $L(M) \subseteq L(G)$ leicht zeigen. Auf den Beweis der Umkehrung verzichten wir. \square

21.14 SATZ. *Zu jeder Turingmaschine M kann man effektiv eine Chomsky-Grammatik G angeben, die die von M akzeptierte Sprache erzeugt, und umgekehrt. Insbesondere gilt $CH = RA$.*

BEWEIS. Dies folgt direkt aus den Lemmata 21.12 und 21.13. \square

Mit diesem Satz überträgt sich die Unentscheidbarkeit semantischer Eigenschaften von Turingmaschinen auf Grammatiken. Insbesondere gilt:

21.15 KOROLLAR. (a) *Es gibt eine Chomsky-Sprache, die nicht rekursiv ist.*

(b) *Das Wortproblem für Chomsky-Grammatiken*

$$W_{CH} = \{([G], x) : x \in L(G)\}$$

ist nicht rekursiv.

(c) *Folgende Probleme für Chomsky-Grammatiken sind nicht rekursiv:*

$$\begin{array}{ll} \text{LEER}_{CH} & = \{[G] : L(G) = \emptyset\} \quad (\text{Leerheitsproblem}) \\ \text{INF}_{CH} & = \{[G] : L(G) \text{ unendlich}\} \quad (\text{Unendlichkeitsproblem}) \\ \text{ÄQU}_{CH} & = \{([G], [G']) : L(G) = L(G')\} \quad (\text{Äquivalenzproblem}) \end{array}$$

Hierbei gehen wir von einer geeigneten Gödelisierung von Grammatiken aus und bezeichnen mit $\lceil G \rceil$ die Gödelnummer der Grammatik G .

BEWEIS. Da das Halteproblem K rekursiv aufzählbar aber nicht rekursiv ist, folgt (a) direkt aus Satz 21.14. Wählt man eine Grammatik G mit $L(G) = K$, so gilt $K \leq_m W_{\text{CH}}$ via $f(x) = (\lceil G \rceil, x)$, weshalb mit K auch W_{CH} nicht rekursiv ist. Teil (c) des Korollars zeigt man ebenfalls mit Hilfe der Reduktionsmethode, wobei man die nach dem Satz von Rice nicht rekursiven analogen Probleme für Turingmaschinen reduziert. Da man nach Satz 21.14 jeder Turingmaschine M effektiv eine Grammatik G mit $L(M) = L(G)$ zuordnen kann, ist die entsprechende Abbildung g der zugehörigen Gödelnummern berechenbar, also (nach Church-Turing-These oder Formalisierung des Argumentes) rekursiv. Hieraus erhält man dann direkt die gewünschten Reduktionen. Z.B. gilt für das Leerheitsproblem $\text{LEER} = \{e : W_e = \emptyset\}$ für (normierte) Turingmaschinen, dass $\text{LEER} \leq_m \text{LEER}_{\text{CH}}$ via g . \square

Von einer Grammatik kann man i. Allg. also wenig über die Struktur der erzeugten Sprache effektiv ablesen. In der Praxis schränkt man daher die zulässigen Regeln in einer Grammatik ein. Hierdurch erhält man Darstellungen, die mehr über die dargestellte Sprache verraten (dafür aber auch weniger Sprachen darstellen). Die wichtigsten Spezialfälle von Grammatiken führen wir im nächsten Abschnitt ein.