Chomsky-Grammatiken

Noam Chomsky (geb. 1928) hat 1956 ein Papier über verschiedene Arten von *formalen Grammatiken* veröffentlicht, die heute als *Chomsky-Grammatiken* bezeichnet werden.

Def.: Eine Chomsky-Grammatik G=(Z, E, S, P) besteht aus
1. einem Alphabet Z von **Zwischensymbolen** (non terminal symbols)
2. einem Alphabet E von **Endsymbolen** (terminal symbols)
3. einem **Startsymbol** S (einem Zwischsymbol, d.h. $S \in T$) und
4. einer Menge P von **Produktions-Regeln**, $P=\{R_1, R_2, \ldots, R_n\}$

Def.: Eine **Satzform** ist eine Folge von Zwischen- und/oder Endsymbolen (sie kann also nur Zwischensymbolen, nur Endsymbole oder ein Mischung aus beiden enthalten). Die *leere Satzform* (die 0 Symbole enthält) wird häufig mit dem griechischen Buchstaben ε bezeichnet ("epsilon", soll an "empty" erinnern).

Für eine Chomski-Grammatik muss gelten:

1. Die Mengen Z und E müssen beide *nicht-leer* und *endlich* sein und dürfen *keine gemeinsamen Elemente* besitzen, d.h. es muss $Z \cap E = \{\}$ gelten.
2. Jede Regel R besteht aus einer *linken Seite* LS, einem *Trennzeichen*, z.B. "→", und einer *rechten Seite* RS, etwa so:
R: LS → RS
3. LS und RS sind *Satzformen* und LS enthält mindestens ein *Zwischensymbol*.
4. Es gibt mindestens eine Regel, deren linke Seite nur aus dem Startsymbol S besteht.

Ende der Definition von Chomsky-Grammatiken.

Beispiel: Die Chomsky-Grammatik G0 :

```
R01: S   -> A C a B
R02: C a -> a a C
R03: C B -> D B
R04: C B -> E
R05: a D -> D a
R06: A D -> A C
R07: a E -> E a
R08: A E -> ε
```

beruht auf folgenden Alphabeten:

Alphabet Z der Zwischensymbole: `{A B C D E S}`
Alphabet E der Endsymbole: `{a}`

Diese Grammatik beschreibt die Sprache S0 = {aa aaaa aaaaaaaa aaaaaaaaaaaaaaaa ...},
d.h. die Menge aller a-Folgen, deren Länge eine positive Zweierpotenz (2^1, 2^2, 2^3, ...) ist
(ursprünglich aus dem Buch "Introduction to Automata Theory, Languages, and Computation" von Hopcroft/Motwani/Ullmann).

In jeder Grammatik muss man *Endsymbole* und *Zwischensymbole* irgendwie deutlich unterscheiden, z.B. mit Anführungszeichen, mit Unterstreichen oder Überstreichen, mit Farben, ...

Eines der nicht-terminalen Symbole muss als *Startsymbol* ausgezeichnet werden
(das Startsymbol hat eine ähnliche Aufgabe wie die `main`-Methode in einem Java-Programm).

Vereinbarung: Um kleine Beispiel-Grammatiken möglichst kurz und einfach beschreiben zu können sei vereinbart:

1. **Große Buchstaben** (A, B, ..., Z) sind **Zwischensymbole**.
2. **Kleine Buchstaben** (a, b, ... , z) und **Sonderzeichen** (z.B. + - * / . , etc.) sind **Endsymbole**.
3. Die erste Regel beginnt mit dem **Startsymbol** (`S -> ...`).

Mit dieser Vereinbarung genügt zur Beschreibung einer Grammatik die Angabe ihrer *Regeln*. Nur wenn von dieser Vereinbarung abgewichen wird, werden in den folgenden Beispielen die Endsymbole, die Zwischensymbole und das Startsymbol einer Grammatik explizit angegeben.

Spezielle Grammatikformen

Je strengeren Einschränkungen man die Regeln einer Chomsky-Grammatik unterwirft,
desto *weniger Sprachen* kann man damit noch beschreiben, aber
desto leichter ist es, *allgemeine Eigenschaften* solcher Grammatiken und Sprachen zu beweisen.
Deshalb hat man sich intensiv mit *eingeschränkten Formen von Chomsky-Grammatiken* befasst, vor allem mit den folgenden 4 Formen (Typ 3 bis Typ 0).

Typ 3 Grammatiken (lineare Grammatiken, reguläre Grammatiken)

Def.: Eine Regel R : LS $\rightarrow$ RS heißt **abschließend**, wenn LS (nur) **ein** Zwischensymbol ist und RS aus einem oder mehreren Endsymbolen besteht (aber keine Zwischensymbole enthält).

Beispiele:

R1: A $\rightarrow$ b c d
R2: A $\rightarrow$ a

Gegenbeispiele:

R4: A $\rightarrow$ B c d -- RS enthält ein Zwischensymbol
R5: A B $\rightarrow$ c d -- LS besteht aus mehr als einem Symbol
R6: A a $\rightarrow$ b c -- LS besteht aus mehr als einem Symbol

Def.: Eine Regel R : LS $\rightarrow$ RS heißt **links-linear**, wenn LS (nur) **ein** Zwischensymbol ist und RS (außer Endsymbolen) genau *ein* Zwischensymbol enthält und dieses am *Anfang* von RS (ganz **links**) steht.

Beispiele:

R1: A $\rightarrow$ B c d
R2: A $\rightarrow$ B
R3: A $\rightarrow$ A b c c b

Gegenbeispiele:

R4: A B $\rightarrow$ C -- LS besteht aus mehr als einem Symbol
R5: A b $\rightarrow$ C -- LS besteht aus mehr als einem Symbol
R6 A $\rightarrow$ B c d E -- RS enthält mehr als ein Zwischensymbol
R7: A $\rightarrow$ b c D e -- das Zwischensymbol D steht nicht am *Anfang* von RS

Def.: Eine Regel R : LS $\rightarrow$ RS heißt **rechts-linear**, wenn LS (nur) **ein** Zwischensymbol ist und RS (außer Endsymbolen) genau *ein* Zwischensymbol enthält und dieses am *Ende* von RS (ganz **rechts**) steht.

Beispiele:

R1: A $\rightarrow$ b c D
R2: A $\rightarrow$ B
R3: A $\rightarrow$ a b a b C

Gegenbeispiele:

R4: A B $\rightarrow$ C -- LS besteht aus mehr als einem Symbol
R5: A b $\rightarrow$ C -- LS besteht aus mehr als einem Symbol
R6 A $\rightarrow$ B c d E -- RS enthält mehr als ein Zwischensymbol
R7: A $\rightarrow$ b c D e -- das Zwischensymbol D steht nicht am *Ende* von RS

Def.: Eine Grammatik ist vom **Typ 3** (man sagt auch: sie ist **linear**, oder: sie ist **regulär**) wenn
entweder gilt: Jede Regel ist *abschließend* oder *links-linear*.
oder wenn gilt: Jede Regel ist *abschließend* oder *rechts-linear*.

Zusätzlich ist in einer Typ 3 Grammatik die Regel S $\rightarrow$ ε erlaubt, wenn S in keiner Regel auf der rechten Seite vorkommt. Dadurch kann auch das leere Wort zu einer Typ-3-Sprache gehören.

Anmerkung: Eine Grammatik, die außer abschließenden Regeln rechts-lineare *und* links-lineare Regeln enthält, ist nicht vom Typ 3, sondern vom Typ 2.

Beispiel für eine Typ 3 Grammatik, G6:

R1:	S	$\rightarrow$	0	R3:	S	$\rightarrow$	S 0
R2:	S	$\rightarrow$	1	R4:	S	$\rightarrow$	S 1

Aufgabe 2-1: Begründen Sie kurz, warum folgende Grammatik G7 nicht vom Typ 3 ist:

R1:	A	$\rightarrow$	a B
R2:	B	$\rightarrow$	A b
R3:	A	$\rightarrow$	a b

Aufgabe 2-2: Geben Sie eine Typ 3 Grammatik G8 an für die Menge aller in ihrer Lieblings-Programmiersprache erlaubten Bezeichner (identifier).

Aufgabe 2-3: Geben Sie eine Typ 3 Grammatik G9 an für die Menge aller Binärzahlen (Ganzzahlen wie 1011 und Brüche wie 10.010, wenn ein Punkt vorhanden ist, muss davor und dahinter mindestens eine Binärziffer stehen, führende und nachfolgende Nullen wie bei den Zahlen 00.000 und 00101.10100 etc. sollen erlaubt sein).

Aufgabe 2-4: Geben Sie eine Typ 3 Grammatik G9A an für die Menge aller Binärzahlen (Ganzzahlen wie 1011 und Brüche wie 10.01, wenn ein Punkt vorhanden ist, muss davor und dahinter mindestens eine Binärziffer stehen, führende und nachfolgende Nullen wie bei den Zahlen 00.000 und 00101.10100 etc. sollen **nicht** erlaubt sein).

Typ 2 Grammatiken (kontextfreie Grammatiken)

Für jede Regel R : LS $\rightarrow$ RS muss gelten: LS ist (nur) **ein** Zwischensymbol.

Beispiele:

R1:	A	$\rightarrow$	B C D
R2:	A	$\rightarrow$	b c d
R3:	A	$\rightarrow$	A b C d e F G
R4:	A	$\rightarrow$	ε

Gegenbeispiele:

R5:	A B	$\rightarrow$	c d	-- LS besteht aus mehr als einem Symbol
R6:	A b	$\rightarrow$	c d	-- LS besteht aus mehr als einem Symbol

Beispiel: Die Grammatik G04 :

```
R01: Zahl     : Vorzeich ZiffFo    // Zahl ist das Startsymbol von G04
R02: Zahl     : ZiffFo
R03: Vorzeich : "+"
R04: Vorzeich : "-"
R05: ZiffFo   : Ziff
R06: ZiffFo   : Ziff ZiffFo
R07: Ziff     : "0"
R08: Ziff     : "1"
```

beruht auf folgenden Alphabeten:

Alphabet Z der Zwischensymbole: {Zahl Vorzeich ZiffFo Ziff}
Alphabet E der Endsymbole: {+ - 0 1}

Bessere Bezeichnung für kontextfreie Grammatiken:

Die Bezeichnung *kontextfreie Grammatik* klingt so, als wäre *kontextfrei* eine *positive* Eigenschaft (etwa so wie "frei von einem lästigen oder schädlichen Kontext"). Tatsächlich soll die Bezeichnung eigentlich darauf hindeuten, dass man mit solchen Grammatiken sog. Kontextbedingungen *nicht beschreiben kann* (obwohl man das eigentlich gern täte). Eine bessere Bezeichnung wäre also *kontextunfähige Grammatik*. Da die Bezeichnung *kontextfrei* aber allgemein verbreitet ist, werden wir sie auch verwenden. Man sollte aber versuchen, sich durch den falschen positiven Klang nicht verwirren zu lassen. Die Kontextfreiheit einer Grammatik ist eine Schwäche, keine Stärke.

Typ 1 Grammatiken (kontextsensitive Grammatiken)

Für jede Regel R : LS $\rightarrow$ RS muss gelten: LS ist nicht länger ("besteht nicht aus mehr Symbolen") als RS. Zusätzlich ist die Regel S $\rightarrow$ ε erlaubt, wenn S in keiner Regel auf der rechten Seite vorkommt.

Beispiele:

R1: A b C d $\rightarrow$ e F g H
R2: A B C $\rightarrow$ D E F G
R3: A B $\rightarrow$ c d e
R4: S $\rightarrow$ ε

Gegenbeispiele:

R5: A b C d $\rightarrow$ e F g -- RS ist kürzer als LS
R6: A B C $\rightarrow$ D E -- RS ist kürzer als LS
R7: A $\rightarrow$ ε -- nur das Startsymbol darf "nach ε gehen"

Typ 0 Grammatiken

Keine Einschränkung der Regeln.

Universelle Parser

Def.: Ein *universeller Typ-n-Parser* ist ein Programm, das als Eingabe eine Typ-n-Grammatik G und eine Zeichenkette S erwartet und feststellt, ob S aus G ableitbar ist oder nicht.

Merke:

1. Es ist ("aus mathematischen Gründen") *unmöglich*, einen **universellen Typ-0-Parser** zu schreiben.

2. Es ist theoretisch möglich, einen **universellen Typ-1-Parser** zu schreiben, aber ein solcher Parser würde für viele Eingaben viel zu viel Zeit (Jahrhunderte und mehr) benötigen.

3. Es gibt **universelle Typ-2-Parser**, die in den meisten praktischen Fällen schnell genug sind (z.B. den Earley-Algorithmus, auf dem der Parser-Generator Accent beruht, welcher zum Gentle-System gehört).

4. Es gibt **universelle Typ-3-Parser**, die für alle Eingaben sehr schnell sind.

5. Mit *Typ-3-Grammatiken* kann man genau dieselben Sprachen beschreiben, wie mit *regulären Ausdrücken*, und genau diese Sprachen kann man auch mit *endlichen Automaten* erkennen.

Hinweis: Der Begriff *Syntaxbaum* ist nur im Zusammenhang mit Typ-2- und Typ-3-Grammatiken sinnvoll, aber nicht im Zusammenhang mit Typ-0- oder Typ-1-Grammatiken.