



# Grammatiken

**Grammatiken** sind regelbasierte Kalküle zur

- Konstruktion von Systemen und Sprachen

# Grammatiken

**Grammatiken** sind regelbasierte Kalküle zur

- Konstruktion von Systemen und Sprachen
- Überprüfung von Systemen und Sprachen

# Grammatiken

**Grammatiken** sind regelbasierte Kalküle zur

- Konstruktion von Systemen und Sprachen
- Überprüfung von Systemen und Sprachen

Grammatiken eignen sich besonders zur Modellierung

- beliebig tief geschachtelter, rekursiver Strukturen.

# Grammatiken

**Grammatiken** sind regelbasierte Kalküle zur

- Konstruktion von Systemen und Sprachen
- Überprüfung von Systemen und Sprachen

Grammatiken eignen sich besonders zur Modellierung

- beliebig tief geschachtelter, rekursiver Strukturen.

Es existieren verschiedene Typen von Grammatiken

- allgemeine Grammatiken (Typ 0)

# Grammatiken

**Grammatiken** sind regelbasierte Kalküle zur

- Konstruktion von Systemen und Sprachen
- Überprüfung von Systemen und Sprachen

Grammatiken eignen sich besonders zur Modellierung

- beliebig tief geschachtelter, rekursiver Strukturen.

Es existieren verschiedene Typen von Grammatiken

- allgemeine Grammatiken (Typ 0)
- kontextsensitive Grammatiken (Typ 1)

# Grammatiken

**Grammatiken** sind regelbasierte Kalküle zur

- Konstruktion von Systemen und Sprachen
- Überprüfung von Systemen und Sprachen

Grammatiken eignen sich besonders zur Modellierung

- beliebig tief geschachtelter, rekursiver Strukturen.

Es existieren verschiedene Typen von Grammatiken

- allgemeine Grammatiken (Typ 0)
- kontextsensitive Grammatiken (Typ 1)
- kontextfreie Grammatiken (Typ 2)

# Grammatiken

**Grammatiken** sind regelbasierte Kalküle zur

- Konstruktion von Systemen und Sprachen
- Überprüfung von Systemen und Sprachen

Grammatiken eignen sich besonders zur Modellierung

- beliebig tief geschachtelter, rekursiver Strukturen.

Es existieren verschiedene Typen von Grammatiken

- allgemeine Grammatiken (Typ 0)
- kontextsensitive Grammatiken (Typ 1)
- kontextfreie Grammatiken (Typ 2)
- reguläre Grammatiken (Typ 3)



# Grammatiken

**Grammatiken** sind regelbasierte Kalküle zur

- Konstruktion von Systemen und Sprachen
- Überprüfung von Systemen und Sprachen

Grammatiken eignen sich besonders zur Modellierung

- beliebig tief geschachtelter, rekursiver Strukturen.

Es existieren verschiedene Typen von Grammatiken

- allgemeine Grammatiken (Typ 0)
- kontextsensitive Grammatiken (Typ 1)
- kontextfreie Grammatiken (Typ 2)
- reguläre Grammatiken (Typ 3)

Betrachten

- **kontextfreie Grammatiken**

# Geschachtelte Strukturen - Beispiele

- Tabellen mit Tabellen als Einträgen

## Geschachtelte Strukturen - Beispiele

- Tabellen mit Tabellen als Einträgen
- Aufbau von Webseiten: Aufzählungen von Aufzählungen

## Geschachtelte Strukturen - Beispiele

- Tabellen mit Tabellen als Einträgen
- Aufbau von Webseiten: Aufzählungen von Aufzählungen
- Arithmetische und Boolesche Ausdrücke

## Geschachtelte Strukturen - Beispiele

- Tabellen mit Tabellen als Einträgen
- Aufbau von Webseiten: Aufzählungen von Aufzählungen
- Arithmetische und Boolesche Ausdrücke
- Geschachtelte Schleifen in Programmiersprachen

## Geschachtelte Strukturen - Beispiele

- Tabellen mit Tabellen als Einträgen
- Aufbau von Webseiten: Aufzählungen von Aufzählungen
- Arithmetische und Boolesche Ausdrücke
- Geschachtelte Schleifen in Programmiersprachen
- Klammersausdrücke

## Geschachtelte Strukturen - Beispiele

- Tabellen mit Tabellen als Einträgen
- Aufbau von Webseiten: Aufzählungen von Aufzählungen
- Arithmetische und Boolesche Ausdrücke
- Geschachtelte Schleifen in Programmiersprachen
- Klammersausdrücke

### Klammersausdrücke

- das leere Wort  $\epsilon$  ist ein Klammersausdruck
- eine Liste von Klammersausdrücke ist wieder ein Klammersausdruck, z.B.  $()()()()$
- ist  $K$  ein Klammersausdruck, dann auch  $(K)$ , z.B.  $K = ()()$  und  $((()))$

## Definition 1

Eine Grammatik  $G$  ist ein 4-Tupel  $(T, N, P, S)$ . Dabei ist

- $T$  eine endliche Menge von Terminalen
- $N$  eine endliche Menge von Nichtterminalen mit  $N \cap T = \emptyset$
- $P$  eine endliche Menge von Produktionen
- $S \in N$  das Startsymbol.

Die Elemente aus  $V := T \cup N$  heißen Symbole und es gilt

$$P \subseteq (V^+ \setminus T^*) \times V^*.$$



## Definition 1

Eine Grammatik  $G$  ist ein 4-Tupel  $(T, N, P, S)$ . Dabei ist

- $T$  eine endliche Menge von Terminalen
- $N$  eine endliche Menge von Nichtterminalen mit  $N \cap T = \emptyset$
- $P$  eine endliche Menge von Produktionen
- $S \in N$  das Startsymbol.

Die Elemente aus  $V := T \cup N$  heißen Symbole und es gilt

$$P \subseteq (V^+ \setminus T^*) \times V^*.$$

- Für eine Produktion  $(A, x) \in P$  schreiben wir auch  $A ::= x$ .
- Die erste Komponente einer Produktion ist eine nicht-leere Folge von Symbolen, wobei mindestens ein Nichtterminal in der Folge auftauchen muss.

## Definition 1

Eine Grammatik  $G$  ist ein 4-Tupel  $(T, N, P, S)$ . Dabei ist

- $T$  eine endliche Menge von Terminalen
- $N$  eine endliche Menge von Nichtterminalen mit  $N \cap T = \emptyset$
- $P$  eine endliche Menge von Produktionen
- $S \in N$  das Startsymbol.

Die Elemente aus  $V := T \cup N$  heißen Symbole und es gilt

$$P \subseteq (V^+ \setminus T^*) \times V^*.$$

- Sagen: In der Produktion  $A ::= x$  steht  $A$  auf der linken Seite und  $x$  auf der rechten Seite.
- Geben Produktionen häufig unterschiedliche Namen, wie  $p_1 : A ::= x$ .

# Grammatiken

$\tilde{G} = (T, N, P, S)$  mit

$$T = \{a, b, c\}$$

$$N = \{S, B, C\}$$

$$P = \left\{ \begin{array}{l} p1 : S ::= aSBC \\ p2 : S ::= aBC \\ p3 : CB ::= BC \\ p4 : aB ::= ab \\ p5 : bB ::= bb \\ p6 : bC ::= bc \\ p7 : cC ::= cc \end{array} \right.$$

# Kontextfreie Grammatiken

## Definition 2

Eine Grammatik  $G = (T, N, P, S)$  heißt kontextfrei, wenn

$$P \subseteq N \times V^*,$$

also, wenn die linke Seite jeder Produktion aus einem einzelnen Nichtterminal besteht.

# Kontextfreie Grammatiken

## Definition 2

Eine Grammatik  $G = (T, N, P, S)$  heißt kontextfrei, wenn

$$P \subseteq N \times V^*,$$

also, wenn die linke Seite jeder Produktion aus einem einzelnen Nichtterminal besteht.

Kontextfreie Grammatiken werden angewandt zur Definition von

- Programmen einer Programmiersprache und deren Struktur, z.B. Java, C, Pascal
- Sprachen als Schnittstellen zwischen Software-Werkzeugen und Datenaustauschformaten, z.B. HTML, XML
- Bäumen zur Repräsentation von strukturierten Daten, z.B. XML
- Strukturen von Protokollen beim Austausch von Nachrichten zwischen Prozessen oder Geräten

# Kontextfreie Grammatiken

## Definition 2

Eine Grammatik  $G = (T, N, P, S)$  heißt kontextfrei, wenn

$$P \subseteq N \times V^*,$$

also, wenn die linke Seite jeder Produktion aus einem einzelnen Nichtterminal besteht.

Kontextfreie Grammatiken sind

- komplex genug, um interessante Strukturen mit ihnen zu beschreiben
- einfach genug, um Aussagen über durch sie definierten Strukturen herleiten zu können, z.B. ist ein Programm syntaktisch korrekt

# Grammatiken

$G_1 = (T, N, P, S)$  mit

$T = \{\text{MenüName}, \text{OperationsName}\}$

$N = \{\text{Menü}, \text{EintragsFolge}, \text{Eintrag}\}$

$S = \text{Menü}$

$P = \{\text{Menü} ::= \text{MenüName EintragsFolge},$   
     $\text{EintragsFolge} ::= \text{Eintrag},$   
     $\text{Eintrag} ::= \text{OperationsName},$   
     $\text{Eintrag} ::= \text{Menü}$   
     $\}$

# Grammatiken

$G_1 = (T, N, P, S)$  mit

$T = \{\text{MenüName}, \text{OperationsName}\}$

$N = \{\text{Menü}, \text{EintragsFolge}, \text{Eintrag}\}$

$S = \text{Menü}$

$P = \{\text{Menü} ::= \text{MenüName EintragsFolge},$   
 $\text{EintragsFolge} ::= \text{Eintrag},$   
 $\text{Eintrag} ::= \text{OperationsName},$   
 $\text{Eintrag} ::= \text{Menü}$   
 $\}$

Die Grammatik  $G_1$  ist kontextfrei.



# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = \text{Klammerung}$$

$$P = \{ \text{Klammerung} ::= '(Liste)', \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \epsilon \\ \}$$

- $\epsilon$  bezeichnet das **leere Wort**
- Um Sonderzeichen, die Terminale einer Grammatik sind, von Zeichen einer Grammatikdefinition zu unterscheiden, setzen wir sie häufig in Apostrophe, z.B. '(' und ')' in der Grammatik  $G_2$ .

# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = \text{Klammerung}$$

$$P = \{ \text{Klammerung} ::= '(Liste)', \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \epsilon \\ \}$$

ist kontextfrei.

- $\epsilon$  bezeichnet das **leere Wort**
- Um Sonderzeichen, die Terminale einer Grammatik sind, von Zeichen einer Grammatikdefinition zu unterscheiden, setzen wir sie häufig in Apostrophe, z.B. '(' und ')' in der Grammatik  $G_2$ .

# Grammatiken

$\tilde{G} = (T, N, P, S)$  mit

$$T = \{a, b, c\}$$

$$N = \{S, B, C\}$$

$$P = \left\{ \begin{array}{l} p1 : S ::= aSBC \\ p2 : S ::= aBC \\ p3 : CB ::= BC \\ p4 : aB ::= ab \\ p5 : bB ::= bb \\ p6 : bC ::= bc \\ p7 : cC ::= cc \end{array} \right.$$

# Grammatiken

$\tilde{G} = (T, N, P, S)$  mit

$$T = \{a, b, c\}$$

$$N = \{S, B, C\}$$

$$P = \left\{ \begin{array}{l} p1 : S ::= aSBC \\ p2 : S ::= aBC \\ p3 : CB ::= BC \\ p4 : aB ::= ab \\ p5 : bB ::= bb \\ p6 : bC ::= bc \\ p7 : cC ::= cc \end{array} \right.$$

Die Grammatik  $\tilde{G}$  ist nicht kontextfrei.

## Definition 3

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

- Sei  $w = uAv \in V^+$ . Existiert in  $P$  eine Produktion  $A ::= x$ , so ist  $w' = uxv$  aus  $uAv$  (direkt) ableitbar. Wir schreiben  $w \rightarrow w'$ .

## Definition 3

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

- Sei  $w = uAv \in V^+$ . Existiert in  $P$  eine Produktion  $A ::= x$ , so ist  $w' = uxv$  aus  $uAv$  (direkt) ableitbar. Wir schreiben  $w \rightarrow w'$ .
- Eine solche einmalige Anwendung einer Produktion heißt Ableitungsschritt.

## Definition 3

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

- Sei  $w = uAv \in V^+$ . Existiert in  $P$  eine Produktion  $A ::= x$ , so ist  $w' = uxv$  aus  $uAv$  (direkt) ableitbar. Wir schreiben  $w \rightarrow w'$ .
- Eine solche einmalige Anwendung einer Produktion heißt Ableitungsschritt.
- Seien  $w \in V^+, w' \in V^*$ . Dann ist  $w'$  aus  $w$  (indirekt) ableitbar, wenn wir  $w'$  aus  $w$  durch endlich viele Ableitungsschritte erhalten können, d.h. wenn  $w_0 = w, w_1, \dots, w_n = w'$  existieren mit  $w_{i-1} \rightarrow w_i, i = 1, \dots, n$ . Ist  $w'$  aus  $w$  ableitbar, so schreiben wir  $w \rightarrow^* w'$ .

## Definition 3

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

- Sei  $w = uAv \in V^+$ . Existiert in  $P$  eine Produktion  $A ::= x$ , so ist  $w' = uxv$  aus  $uAv$  (direkt) ableitbar. Wir schreiben  $w \rightarrow w'$ .
- Eine solche einmalige Anwendung einer Produktion heißt Ableitungsschritt.
- Seien  $w \in V^+, w' \in V^*$ . Dann ist  $w'$  aus  $w$  (indirekt) ableitbar, wenn wir  $w'$  aus  $w$  durch endlich viele Ableitungsschritte erhalten können, d.h. wenn  $w_0 = w, w_1, \dots, w_n = w'$  existieren mit  $w_{i-1} \rightarrow w_i, i = 1, \dots, n$ . Ist  $w'$  aus  $w$  ableitbar, so schreiben wir  $w \rightarrow^* w'$ .
- Eine Folge von Ableitungsschritten nennen wir eine Ableitung.



# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = \text{Klammerung}$$

$$P = \{ \text{Klammerung} ::= '(Liste)', \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \epsilon \\ \}$$

# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = Klammerung$$

$$P = \{Klammerung ::= '(Liste)', \\ Liste ::= Klammerung Liste, \\ Liste ::= \epsilon \\ \}$$

Ableitungsschritte in  $G_2$

$$(Liste) \rightarrow (Klammerung Liste)$$

$$((Liste)Liste) \rightarrow (( )Liste)$$

$$Klammerung \rightarrow (Liste)$$

# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = Klammerung$$

$$P = \{Klammerung ::= '(Liste)', \\ Liste ::= Klammerung Liste, \\ Liste ::= \epsilon \\ \}$$

Eine Ableitung in  $G_2$

# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = \text{Klammerung}$$

$$P = \{ \text{Klammerung} ::= '(Liste)', \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \epsilon \\ \}$$

Eine Ableitung in  $G_2$

$$\text{Klammerung} \rightarrow (Liste)$$

$$\rightarrow (\text{Klammerung Liste})$$

$$\rightarrow (\text{Klammerung Klammerung Liste})$$

$$\rightarrow (\text{Klammerung (Liste) Liste})$$

$$\rightarrow ((Liste)(Liste) Liste)$$

# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = Klammerung$$

$$P = \{Klammerung ::= '(Liste)', \\ Liste ::= Klammerung Liste, \\ Liste ::= \epsilon \\ \}$$

Eine Ableitung in  $G_2$

$$\begin{aligned} ((Liste)(Liste) Liste) &\rightarrow (( ) (Liste) Liste) \\ &\rightarrow (( ) ( ) Liste) \\ &\rightarrow (( ) ( )) \end{aligned}$$

# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = \text{Klammerung}$$

$$P = \{ \text{Klammerung} ::= '(Liste)', \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \epsilon \\ \}$$

Eine Ableitung in  $G_2$

$$\begin{aligned} ((Liste)(Liste) Liste) &\rightarrow (( ) (Liste) Liste) \\ &\rightarrow (( ) ( ) Liste) \\ &\rightarrow (( ) ( )) \end{aligned}$$

**Kurz:**  $\text{Klammerung} \rightarrow^* (( ) ( ))$

## Definition 4

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik. Dann heißt

$$L(G) = \{w \in T^* \mid S \rightarrow^* w\}$$

die von  $G$  erzeugte Sprache.  $L(G)$  besteht also aus allen Folgen von Terminalen, die aus dem Startsymbol  $S$  der Grammatik  $G$  abgeleitet werden können.

## Definition 4

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik. Dann heißt

$$L(G) = \{w \in T^* \mid S \rightarrow^* w\}$$

die von  $G$  erzeugte Sprache.  $L(G)$  besteht also aus allen Folgen von Terminalen, die aus dem Startsymbol  $S$  der Grammatik  $G$  abgeleitet werden können.

$G_3 = (T, N, P, S)$  mit

$$T = \{a\}$$

$$N = \{A\}$$

$$S = A$$

$$P = \{A ::= aA, A ::= a\}$$



## Definition 4

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik. Dann heißt

$$L(G) = \{w \in T^* \mid S \rightarrow^* w\}$$

die von  $G$  erzeugte Sprache.  $L(G)$  besteht also aus allen Folgen von Terminalen, die aus dem Startsymbol  $S$  der Grammatik  $G$  abgeleitet werden können.

$G_3 = (T, N, P, S)$  mit

$$T = \{a\}$$

$$N = \{A\}$$

$$S = A$$

$$P = \{A ::= aA, A ::= a\}$$

$$L(G_3) = \{a^n \mid n \geq 1\}$$

## Definition 4

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik. Dann heißt

$$L(G) = \{w \in T^* \mid S \rightarrow^* w\}$$

die von  $G$  erzeugte Sprache.  $L(G)$  besteht also aus allen Folgen von Terminalen, die aus dem Startsymbol  $S$  der Grammatik  $G$  abgeleitet werden können.

$G_4 = (T, N, P, S)$  mit

$$T = \{a, b\}$$

$$N = \{S\}$$

$$S = S$$

$$P = \{S ::= aSb, S ::= \epsilon\}$$

## Definition 4

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik. Dann heißt

$$L(G) = \{w \in T^* \mid S \rightarrow^* w\}$$

die von  $G$  erzeugte Sprache.  $L(G)$  besteht also aus allen Folgen von Terminalen, die aus dem Startsymbol  $S$  der Grammatik  $G$  abgeleitet werden können.

$G_4 = (T, N, P, S)$  mit

$$T = \{a, b\}$$

$$N = \{S\}$$

$$S = S$$

$$P = \{S ::= aSb, S ::= \epsilon\}$$

$$L(G_4) = \{a^n b^n \mid n \geq 0\}$$

# Ableitungsbäume

Ableitungen kontextfreier Grammatiken können graphisch durch Ableitungsbäume dargestellt werden.

# Ableitungsbäume

Ableitungen kontextfreier Grammatiken können graphisch durch Ableitungsbäume dargestellt werden.

## Ableitungsbäume

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik. Für eine Ableitung eines Wortes  $w \in T^*$ ,  $w = w_1 \dots w_n$ , ist der zugehörige Ableitungsbaum ein Baum mit Wurzel mit Knotenmarkierungen in  $V$ . Dabei gilt weiter

- Die Wurzel ist mit dem Startsymbol  $S$  markiert.
- Der Baum besitzt  $n$  Blätter und nur die Blätter sind mit Terminalen oder dem leeren Wort  $\epsilon$  markiert. Die Markierungen der Blätter ergeben, von links nach rechts gelesen, das Wort  $w$ .
- Innere Knoten des Baums sind mit Nichtterminalen markiert. Für jeden inneren Knoten repräsentieren der Knoten und seine unmittelbaren Nachfolger die Anwendung einer Produktion.

# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$T = \{(, )\}$

$N = \{Klammerung, Liste\}$

$S = Klammerung$

$P = \{Klammerung ::= '(Liste)'$ ,

$Liste ::= Klammerung Liste,$

$Liste ::= \epsilon\}$

Eine Ableitung in  $G_2$

$Klammerung \rightarrow (Liste)$

$\rightarrow (Klammerung Liste)$

$\rightarrow (Klammerung Klammerung Liste)$

$\rightarrow (Klammerung (Liste) Liste)$

$\rightarrow ((Liste)(Liste) Liste)$

$\rightarrow (( ) (Liste) Liste)$

$\rightarrow (( ) ( ) Liste)$

$\rightarrow (( ) ( ))$

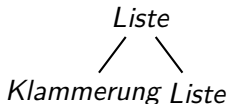
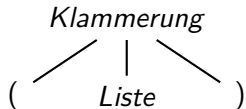
## Ableitungsbaum für Klammersausdruck

$$P = \{ \textit{Klammerung} ::= '(\textit{Liste})', \\ \textit{Liste} ::= \textit{Klammerung} \textit{Liste}, \\ \textit{Liste} ::= \epsilon \}$$

# Ableitungsbaum für Klammersausdruck

$$P = \{ \text{Klammerung} ::= '(\text{Liste})', \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \epsilon \}$$

## Ableitungen als Teilbäume:

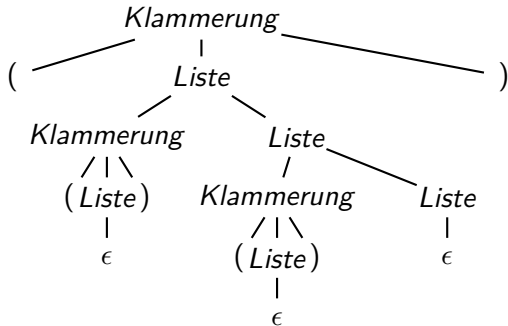




# Ableitungsbaum für Klammersausdruck

$P = \{ \text{Klammerung} ::= '( \text{Liste} )',$   
 $\text{Liste} ::= \text{Klammerung Liste},$   
 $\text{Liste} ::= \epsilon \}$

Ableitungsbaum für  $\text{Klammerung} \rightarrow^* (( ) ( )):$



## Arithmetische Ausdrücke

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= '('Ausdruck')'

BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'

## Arithmetische Ausdrücke

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= '('Ausdruck')'

Startsymbol : Ausdruck

BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'

# Arithmetische Ausdrücke

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= '('Ausdruck')'

Startsymbol : Ausdruck

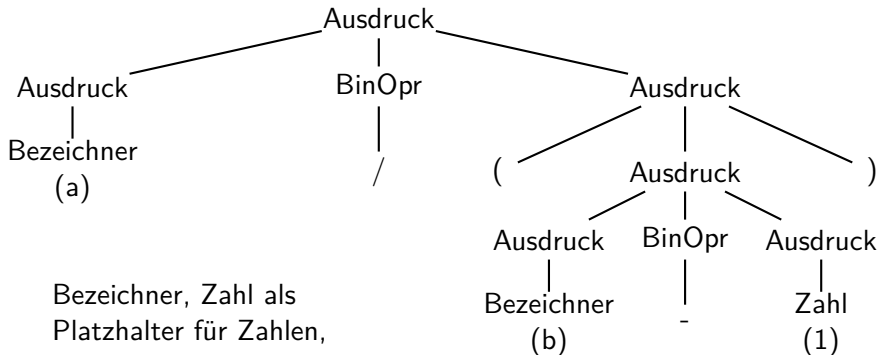
BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'

Ableitungsbaum zum Ausdruck  $a/(b - 1)$ :



Bezeichner, Zahl als  
Platzhalter für Zahlen,  
Variablen

# Mehrdeutigkeit

## Definition 5

Eine Grammatik  $G = (T, N, P, S)$  heißt mehrdeutig, wenn es Elemente in  $L(G)$  gibt, für die mehrere Ableitungsbäume existieren.

# Mehrdeutigkeit

## Definition 5

Eine Grammatik  $G = (T, N, P, S)$  heißt mehrdeutig, wenn es Elemente in  $L(G)$  gibt, für die mehrere Ableitungsbäume existieren.

## Mehrdeutigkeit

- kann zu Verwirrungen über die Bedeutung von Ausdrücken führen,

# Mehrdeutigkeit

## Definition 5

Eine Grammatik  $G = (T, N, P, S)$  heißt mehrdeutig, wenn es Elemente in  $L(G)$  gibt, für die mehrere Ableitungsbäume existieren.

## Mehrdeutigkeit

- kann zu Verwirrungen über die Bedeutung von Ausdrücken führen,
- kann den Nachweis von Eigenschaften der von einer Grammatik definierten Sprache erschweren.

# Arithmetische Ausdrücke und Mehrdeutigkeit

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= '('Ausdruck')'

Startsymbol : Ausdruck

BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'



# Arithmetische Ausdrücke und Mehrdeutigkeit

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= '('Ausdruck')'

Startsymbol : Ausdruck

BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'

Ableitungsbäume zum Ausdruck  $a * b - 1$ :

# Arithmetische Ausdrücke und Mehrdeutigkeit

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= '('Ausdruck')'

Startsymbol : Ausdruck

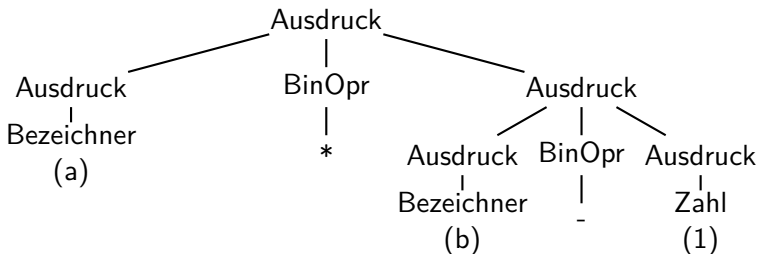
BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'

Ableitungsbäume zum Ausdruck  $a * b - 1$ :



# Arithmetische Ausdrücke und Mehrdeutigkeit

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= ('Ausdruck')

Startsymbol : Ausdruck

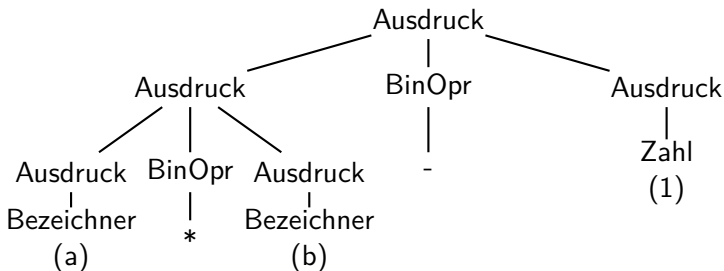
BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'

Ableitungsbäume zum Ausdruck  $a * b - 1$ :



## Klammerausdrücke und Mehrdeutigkeit

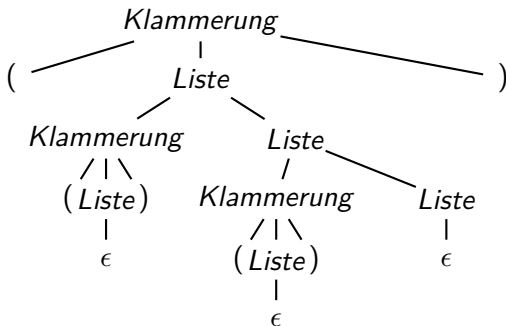
$$P = \{ \textit{Klammerung} ::= \textit{'('Liste')'}, \\ \textit{Liste} ::= \textit{Klammerung Liste}, \\ \textit{Liste} ::= \epsilon \}$$

**Ableitungsbaum für  $\textit{Klammerung} \rightarrow^* (( ) ( ))$ :**

# Klammerausdrücke und Mehrdeutigkeit

$$P = \{ \text{Klammerung} ::= '(' \text{Liste}' , \\ \text{Liste} ::= \text{Klammerung} \text{Liste} , \\ \text{Liste} ::= \epsilon \}$$

**Ableitungsbaum für  $\text{Klammerung} \rightarrow^* (( ) ( ))$ :**



## Klammerausdrücke und Mehrdeutigkeit

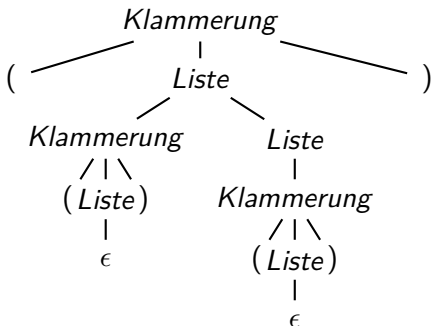
$$P = \{ \textit{Klammerung} ::= \textit{'('Liste')'}, \\ \textit{Liste} ::= \textit{Klammerung Liste}, \\ \textbf{Liste} ::= \textbf{Klammerung}, \\ \textit{Liste} ::= \epsilon \}$$

**Ableitungsbaum für  $\textit{Klammerung} \rightarrow^* (( ) ( ))$ :**

# Klammerausdrücke und Mehrdeutigkeit

$$P = \{ \text{Klammerung} ::= \text{'(' Liste ')'}, \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \text{Klammerung}, \\ \text{Liste} ::= \epsilon \}$$

Ableitungsbaum für  $\text{Klammerung} \rightarrow^* (( ) ( ))$ :



# Klammerausdrücke und Mehrdeutigkeit

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = \text{Klammerung}$$

$$P = \{ \text{Klammerung} ::= '(Liste)', \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \epsilon \\ \}$$



# Klammerausdrücke und Mehrdeutigkeit

$G_2 = (T, N, P, S)$  mit

$T = \{(, )\}$

$N = \{Klammerung, Liste\}$

$S = Klammerung$

$P = \{Klammerung ::= '(Liste)'$ ,

$Liste ::= Klammerung Liste$ ,

$Liste ::= \epsilon$

}

## Satz 6

*Die Grammatik  $G_2$  ist nicht mehrdeutig.*

## Spezielle Ableitungen

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

## Spezielle Ableitungen

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

### Linksableitungen

Eine Linksableitung ist eine Folge von Ableitungsschritten, bei der stets das am weitesten links stehende Nichtterminal durch Anwendung einer Produktion ersetzt wird.

# Spezielle Ableitungen

## Aritmetische Ausdrücke

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= '('Ausdruck)'

BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'

# Spezielle Ableitungen

## Aritmetische Ausdrücke

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::=  $(\text{'Ausdruck'})$

BinOpr ::=  $+$

BinOpr ::=  $-$

BinOpr ::=  $*$

BinOpr ::=  $/$

## Beispiel einer Linksableitung:

Ausdruck  $\rightarrow$  Ausdruck BinOpr Ausdruck

$\rightarrow$  Bezeichner (a) BinOpr Ausdruck

$\rightarrow$  Bezeichner (a) \* Ausdruck

$\rightarrow$  Bezeichner (a) \* Ausdruck BinOpr Ausdruck

$\rightarrow$  Bezeichner (a) \* Bezeichner (b) BinOpr Ausdruck

$\rightarrow$  Bezeichner (a) \* Bezeichner (b) - Ausdruck

$\rightarrow$  Bezeichner (a) \* Bezeichner (b) - Zahl (1)

$(= a * b - 1)$

## Spezielle Ableitungen

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

### Rechtsableitungen

Eine Rechtsableitung ist eine Folge von Ableitungsschritten, bei der stets das am weitesten rechts stehende Nichtterminal durch Anwendung einer Produktion ersetzt wird.

# Spezielle Ableitungen

## Aritmetische Ausdrücke

Ausdruck	::=	Ausdruck BinOpr Ausdruck	BinOpr	::=	'+'
Ausdruck	::=	Zahl	BinOpr	::=	'-'
Ausdruck	::=	Bezeichner	BinOpr	::=	'*'
Ausdruck	::=	('Ausdruck')	BinOpr	::=	'/'

## Beispiel einer Rechtsableitung:

Ausdruck  $\rightarrow$  Ausdruck BinOpr Ausdruck  
 $\rightarrow$  Ausdruck BinOpr Ausdruck BinOpr Ausdruck  
 $\rightarrow$  Ausdruck BinOpr Ausdruck BinOpr Zahl (1)  
 $\rightarrow$  Ausdruck BinOpr Ausdruck - Zahl (1)  
 $\rightarrow$  Ausdruck BinOpr Bezeichner (b) - Zahl (1)  
 $\rightarrow$  Ausdruck \* Bezeichner (b) - Zahl (1)  
 $\rightarrow$  Bezeichner (a) \* Bezeichner (b) - Zahl (1)  
(=  $a * b - 1$ )

## Spezielle Ableitungen und Mehrdeutigkeit

### Definition 5

Eine Grammatik  $G = (T, N, P, S)$  heißt mehrdeutig, wenn es Elemente in  $L(G)$  gibt, für die mehrere Ableitungsbäume existieren.



# Spezielle Ableitungen und Mehrdeutigkeit

## Definition 5

Eine Grammatik  $G = (T, N, P, S)$  heißt mehrdeutig, wenn es Elemente in  $L(G)$  gibt, für die mehrere Ableitungsbäume existieren.

## Satz 7

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

- ➊  $G$  ist mehrdeutig genau dann, wenn es Elemente in  $L(G)$  gibt, für die mehrere Linksableitungen existieren.
- ➋  $G$  ist mehrdeutig genau dann, wenn es Elemente in  $L(G)$  gibt, für die mehrere Rechtsableitungen existieren.

# Spezielle Ableitungen und Mehrdeutigkeit

## Definition 5

Eine Grammatik  $G = (T, N, P, S)$  heißt mehrdeutig, wenn es Elemente in  $L(G)$  gibt, für die mehrere Ableitungsbäume existieren.

## Satz 7

Sei  $G = (T, N, P, S)$  eine kontextfreie Grammatik.

- ➊  $G$  ist mehrdeutig genau dann, wenn es Elemente in  $L(G)$  gibt, für die mehrere Linksableitungen existieren.
- ➋  $G$  ist mehrdeutig genau dann, wenn es Elemente in  $L(G)$  gibt, für die mehrere Rechtsableitungen existieren.

## Satz 6

Die Grammatik  $G_2$  ist nicht mehrdeutig.

# Klammerausdrücke

$G_2 = (T, N, P, S)$  mit

$$T = \{(, )\}$$

$$N = \{Klammerung, Liste\}$$

$$S = \text{Klammerung}$$

$$P = \{ \text{Klammerung} ::= '(Liste)', \\ \text{Liste} ::= \text{Klammerung Liste}, \\ \text{Liste} ::= \epsilon \\ \}$$

# Normalformen - Backus-Naur-Form

## Backus-Naur-Form (BNF)

- ist eine Kurzschreibweise zur Darstellung einer kontextfreien Grammatik  $G = (T, N, P, S)$

# Normalformen - Backus-Naur-Form

## Backus-Naur-Form (BNF)

- ist eine Kurzschreibweise zur Darstellung einer kontextfreien Grammatik  $G = (T, N, P, S)$
- ist  $A \in N$  und sind  $A ::= x_i, x_i \in V^*, i = 1, \dots, k$  die Produktionen in  $P$  mit linker Seite  $A$ , so werden diese zu

$$A ::= x_1 \mid x_2 \mid \dots \mid x_k$$

zusammengefasst

# Normalformen - Backus-Naur-Form

## Backus-Naur-Form (BNF)

- ist eine Kurzschreibweise zur Darstellung einer kontextfreien Grammatik  $G = (T, N, P, S)$
- ist  $A \in N$  und sind  $A ::= x_i, x_i \in V^*, i = 1, \dots, k$  die Produktionen in  $P$  mit linker Seite  $A$ , so werden diese zu

$$A ::= x_1 \mid x_2 \mid \dots \mid x_k$$

zusammengefasst

## Arithmetische Ausdrücke

Ausdruck ::= Ausdruck BinOpr Ausdruck

Ausdruck ::= Zahl

Ausdruck ::= Bezeichner

Ausdruck ::= '('Ausdruck')'

BinOpr ::= '+'

BinOpr ::= '-'

BinOpr ::= '\*'

BinOpr ::= '/'

# Normalformen - Backus-Naur-Form

## Backus-Naur-Form (BNF)

- ist eine Kurzschreibweise zur Darstellung einer kontextfreien Grammatik  $G = (T, N, P, S)$
- ist  $A \in N$  und sind  $A ::= x_i, x_i \in V^*, i = 1, \dots, k$  die Produktionen in  $P$  mit linker Seite  $A$ , so werden diese zu

$$A ::= x_1 \mid x_2 \mid \dots \mid x_k$$

zusammengefasst

## Arithmetische Ausdrücke in BNF

Ausdruck  $::=$  Ausdruck BinOpr Ausdruck  $\mid$  Zahl  $\mid$  Bezeichner  $\mid$  ('Ausdruck')

BinOpr  $::=$  '+'  $\mid$  '-'  $\mid$  '\*'  $\mid$  '/'

## Normalformen - Chomsky-Normalform

### Definition 8

Eine kontextfreie Grammatik  $G = (T, N, P, S)$  ist in Chomsky-Normalform, wenn jede Regel in  $P$  von der Form

$$\begin{array}{ll} A ::= BC & B, C \in N \\ \text{oder } A ::= a & a \in T \end{array}$$

ist. Zusätzlich ist die Regel  $S ::= \epsilon$  erlaubt. Wenn sie vorkommt, dann darf  $S$  nicht auf der rechten Seite einer Produktion auftauchen.



# Normalformen - Chomsky-Normalform

## Definition 8

Eine kontextfreie Grammatik  $G = (T, N, P, S)$  ist in Chomsky-Normalform, wenn jede Regel in  $P$  von der Form

$$\begin{array}{ll} A ::= BC & B, C \in N \\ \text{oder } A ::= a & a \in T \end{array}$$

ist. Zusätzlich ist die Regel  $S ::= \epsilon$  erlaubt. Wenn sie vorkommt, dann darf  $S$  nicht auf der rechten Seite einer Produktion auftauchen.

## Satz 9

*Sei  $G$  eine kontextfreie Grammatik. Dann kann aus  $G$  eine kontextfreie Grammatik  $G'$  in Chomsky-Normalform konstruiert werden, die dieselbe Sprache wie  $G$  erzeugt.*

## Normalformen - Chomsky-Normalform

### Definition 8

Eine kontextfreie Grammatik  $G = (T, N, P, S)$  ist in Chomsky-Normalform, wenn jede Regel in  $P$  von der Form

$$\begin{array}{ll} A ::= BC & B, C \in N \\ \text{oder } A ::= a & a \in T \end{array}$$

ist. Zusätzlich ist die Regel  $S ::= \epsilon$  erlaubt. Wenn sie vorkommt, dann darf  $S$  nicht auf der rechten Seite einer Produktion auftauchen.

$G_3 = (T, N, P, S)$  mit

$$T = \{a\}, N = \{A\}$$

$$S = A$$

$$P = \{A ::= aA, A ::= a\}$$

ist nicht in Chomsky-Normalform.

## Normalformen - Chomsky-Normalform

### Definition 8

Eine kontextfreie Grammatik  $G = (T, N, P, S)$  ist in Chomsky-Normalform, wenn jede Regel in  $P$  von der Form

$$\begin{array}{ll} A ::= BC & B, C \in N \\ \text{oder } A ::= a & a \in T \end{array}$$

ist. Zusätzlich ist die Regel  $S ::= \epsilon$  erlaubt. Wenn sie vorkommt, dann darf  $S$  nicht auf der rechten Seite einer Produktion auftauchen.

$G_3 = (T, N, P, S)$  mit

$$T = \{a\}, N = \{A\}$$

$$S = A$$

$$P = \{A ::= aA, A ::= a\}$$

ist nicht in Chomsky-Normalform.

$G'_3 = (T, N', P', S)$  mit

$$T = \{a\}, N' = \{A, A_1\}$$

$$S = A$$

$$P' = \{A ::= A_1A \mid a, A_1 ::= a\}$$

ist in Chomsky-Normalform.

Außerdem ist  $L(G_3) = L(G'_3)$ .

## Normalformen - Greibach-Normalform

### Definition 10

Eine kontextfreie Grammatik  $G = (T, N, P, S)$  ist in Greibach-Normalform, wenn jede Regel in  $P$  von der Form

$$A ::= aB_1 \dots B_k$$

mit  $a \in T, k \geq 0, B_i \in N$  für  $i = 1, \dots, k$ , ist.

Zusätzlich ist die Regel  $S ::= \epsilon$  erlaubt. Wenn sie vorkommt, dann darf  $S$  nicht auf der rechten Seite einer Produktion auftauchen.

## Normalformen - Greibach-Normalform

### Definition 10

Eine kontextfreie Grammatik  $G = (T, N, P, S)$  ist in Greibach-Normalform, wenn jede Regel in  $P$  von der Form

$$A ::= aB_1 \dots B_k$$

mit  $a \in T, k \geq 0, B_i \in N$  für  $i = 1, \dots, k$ , ist.

Zusätzlich ist die Regel  $S ::= \epsilon$  erlaubt. Wenn sie vorkommt, dann darf  $S$  nicht auf der rechten Seite einer Produktion auftauchen.

### Satz 11

*Sei  $G$  eine kontextfreie Grammatik. Dann kann aus  $G$  eine kontextfreie Grammatik  $G'$  in Greibach-Normalform konstruiert werden, die dieselbe Sprache wie  $G$  erzeugt.*

## Normalformen - Greibach-Normalform

### Definition 10

Eine kontextfreie Grammatik  $G = (T, N, P, S)$  ist in Greibach-Normalform, wenn jede Regel in  $P$  von der Form

$$A ::= aB_1 \dots B_k$$

mit  $a \in T, k \geq 0, B_i \in N$  für  $i = 1, \dots, k$ , ist.

Zusätzlich ist die Regel  $S ::= \epsilon$  erlaubt. Wenn sie vorkommt, dann darf  $S$  nicht auf der rechten Seite einer Produktion auftauchen.

$G_3 = (T, N, P, S)$  mit

$$T = \{a\}$$

$$N = \{A\}$$

$$S = A$$

$$P = \{A ::= aA, A ::= a\}$$

ist in Greibach-Normalform.

# HTML

- HTML (Hypertext Markup Language) ist eine Sprache zur Darstellung von verzeigten Texten
- zusammen mit XML (Extensible Markup Language) verwendet insbesondere im WorldWideWeb (WWW)
- typisch für HTML sind geklammerte Strukturen:  $\langle x \rangle \dots \langle /x \rangle$

# HTML

- HTML (Hypertext Markup Language) ist eine Sprache zur Darstellung von verzeigerten Texten
- zusammen mit XML (Extensible Markup Language) verwendet insbesondere im WorldWideWeb (WWW)
- typisch für HTML sind geklammerte Strukturen:  $\langle x \rangle \dots \langle /x \rangle$

## Ausschnitt der HTML-Produktionen zur Erzeugung von Tabellen:

*Table ::=  $\langle table \rangle$  Rows  $\langle /table \rangle$*

*Rows ::= Rows Row*

*Rows ::= Row*

*Row ::=  $\langle tr \rangle$  Cells  $\langle /tr \rangle$*

*Cells ::= Cells Cell*

*Cells ::= Cell*

*Cell ::=  $\langle td \rangle$  Text  $\langle /td \rangle$*

*Cell ::=  $\langle td \rangle$  Table  $\langle /td \rangle$*



# HTML

- HTML (Hypertext Markup Language) ist eine Sprache zur Darstellung von verzeigerten Texten
- zusammen mit XML (Extensible Markup Language) verwendet insbesondere im WorldWideWeb (WWW)
- typisch für HTML sind geklammerte Strukturen:  $\langle x \rangle \dots \langle /x \rangle$

## Ausschnitt der HTML-Produktionen zur Erzeugung von Tabellen:

$Table ::= \langle table \rangle Rows \langle /table \rangle$

$Rows ::= Rows Row$

$Rows ::= Row$

$Row ::= \langle tr \rangle Cells \langle /tr \rangle$

$Cells ::= Cells Cell$

$Cells ::= Cell$

$Cell ::= \langle td \rangle Text \langle /td \rangle$

$Cell ::= \langle td \rangle Table \langle /td \rangle$

**Achtung:** Im Beispiel sind HTML-Strukturklammern nicht aufgeführt.

# HTML

## Ausschnitt der Produktionen von HTML zur Erzeugung von Tabellen:

$Table ::= \langle table \rangle Rows \langle /table \rangle$

$Rows ::= Rows Row$

$Rows ::= Row$

$Row ::= \langle tr \rangle Cells \langle /tr \rangle$

$Cells ::= Cells Cell$

$Cells ::= Cell$

$Cell ::= \langle td \rangle Text \langle /td \rangle$

$Cell ::= \langle td \rangle Table \langle /td \rangle$

# HTML

## Ausschnitt der Produktionen von HTML zur Erzeugung von Tabellen:

$Table ::= \langle table \rangle Rows \langle /table \rangle$

$Rows ::= Rows Row$

$Rows ::= Row$

$Row ::= \langle tr \rangle Cells \langle /tr \rangle$

$Cells ::= Cells Cell$

$Cells ::= Cell$

$Cell ::= \langle td \rangle Text \langle /td \rangle$

$Cell ::= \langle td \rangle Table \langle /td \rangle$

*Text* steht als Platzhalter für beliebigen Text.

# HTML

Ausschnitt der Produktionen von HTML zur Erzeugung von Tabellen:

$Table ::= \langle table \rangle Rows \langle /table \rangle$

$Rows ::= Rows Row$

$Rows ::= Row$

$Row ::= \langle tr \rangle Cells \langle /tr \rangle$

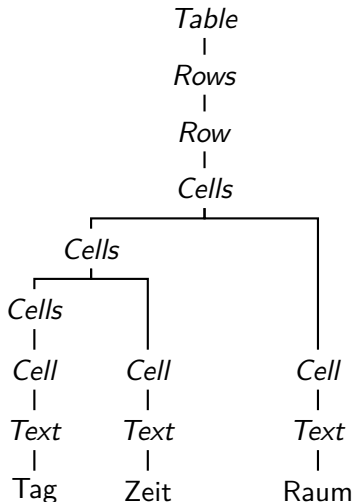
$Cells ::= Cells Cell$

$Cells ::= Cell$

$Cell ::= \langle td \rangle Text \langle /td \rangle$

$Cell ::= \langle td \rangle Table \langle /td \rangle$

Ableitungsbaum für Zeile  
Tag Zeit Raum:



# HTML Ableitung

Tag	Zeit	Raum
Mo	18:00 -19:30	AM
Fr	11:00 -13:00	AM

# HTML Ableitung

Tag	Zeit	Raum
Mo	18:00 -19:30	AM
Fr	11:00 -13:00	AM

