



Automatentheorie

kontextfreie Grammatiken

Prof. Dr. Franz-Karl Schmatzer
schmatzf@dhbw-loerrach.de

Literatur

- C.Wagenknecht, M.Hielscher; Formale Sprachen, abstrakte Automaten und Compiler; 3.Aufl. Springer Vieweg 2022;
- Sipser M.; Introduction to the Theory of Computation; 2.Aufl.; Thomson Course Technology 2006
- Hopcroft, T. et al; Introduction to Automata Theory, Language, and Computation; 3. Aufl. Pearson Verlag 2006
- Vossen,G. Witt K.; Grundkurs Theoretische Informatik; 4.Aufl.; Vieweg Verlag 2006
- Cohen, D; Introduction to Computer Theory; John Wiley 1990

Agenda

- kontextfreie Grammatiken
 - Ableitungsbäume
 - Eindeutig und mehrdeutige Grammatiken
- Normalformen kontextfreier Grammatiken
- Erweiterte Backus-Naur-Form
- Eigenschaften kontextfreier Grammatiken
- CYK-Algorithmus

Kontextfreie Sprache

Einführung

- Die Sprache $L(P)$ mit

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

gehört nicht zu einer Typ-3-Grammatik

- Bei einer Typ-3-Sprache kann höchstens 1 Terminalsymbol pro Produktionsregel erzeugt werden.
- Aufheben dieser Beschränkung:
 - $S \rightarrow aSb$
 - (Anwenden der Regel: $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow \dots$)
 - d.h. $S \Rightarrow^* a^n S b^n$

Wir brauchen noch eine Terminierung $S \rightarrow \varepsilon$ und haben $L(P)$

- $G = (\{S\}, \{a,b\}, \{S \rightarrow aSb, S \rightarrow \varepsilon\}, S)$ erzeugt die Sprache $L(P)$

Kontextfreie Sprache

Definition

- Eine Grammatik $G = (N, \Sigma, P, S)$ heißt Typ-2-Grammatik, wenn
 - N eine Menge von Nicht-Terminalsymbolen, die zu Σ disjunkt ist.
 - Σ eine Menge von Terminalsymbolen
 - P eine Relation $P \subseteq N \times (\Sigma \cup N)^*$ und $|P| < \infty$
 - $S \in N$, dem Startsymbol
 - Ein Element $p \in P$ mit $p = (l, r)$ heißt Produktion oder Regel mit $l \in N$ und $r \in (\Sigma \cup N)^*$ und der
Notation: $l \rightarrow r$ (l geht über in r oder l wird durch r ersetzt)
- Eine Sprache L heißt kontextfrei über Σ , falls es eine kontextfreie Grammatik G über Σ gibt mit $L = L(G)$

Kontextfreie Sprache

Beispiel 2 Palindrome

- $L_P = \{ww^r \mid w \in \Sigma^*\}$ (Palindrome)
 - L gehört nicht zu den regulären Sprachen, sondern zu den kontextfreien Sprachen.
 - Die Grammatik dazu:
 - $G_P = (\{S\}, \{0,1\}, \{S \rightarrow 0S0 \mid 1S1 \mid \varepsilon\}, S)$
 - Erstellen Sie die Grammatik mit Flaci und produzieren sie alle Worte der Länge $l < 5$

Kontextfreie Sprache

Beispiel 2 Palindrome

- $L_P = \{ww^r \mid w \in \Sigma^*\}$ (Palindrome) FLACI
- $L = \{\epsilon, 00, 11, 0110, 1001, 0000, 1111, \dots\}$

$G = (N, T, P, s)$

$N = \{S\}$

$T = \{0, 1\}$

$P = \{$

$S \rightarrow \epsilon \mid 0S0 \mid 1S1$

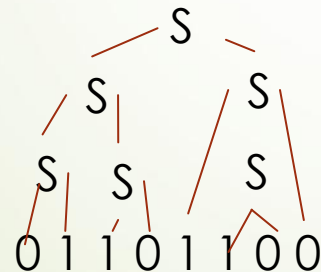
$\}$

$s = S$

Aufgaben

kontextfreie Grammatiken

- Konstruieren Sie eine kontextfreie Grammatik für Sprachen L mit
 - $L = \{ 0^n 1^m 2^n \mid n, m \geq 0 \}$
 - $L = \{ 0^n 1^m \mid n, m \geq 0, n < m \}$
 - $L = \{ w \in \Sigma^* \mid \text{Anzahl der 0-Ziffern} = \text{Anzahl der 1-Ziffern in } w \}$
 - Die Sprache L der ausgewogenen Klammerausdrücke. D.h. jede öffnende Klammer muss auch eine schließende Klammer haben.
- Nutzen Sie FLACI und prüfen Sie ihre Implementierung auf Korrektheit



Lösung Aufgaben

kontextfreie Grammatiken

- Konstruieren Sie eine kontextfreie Grammatik für Sprachen L mit

- $L = \{ 0^n 1^m 2^n \mid n, m \geq 0 \}$

$$G = (\{S, B\}, \{0, 1\}, \{S \rightarrow 0S2 \mid 1B \mid \varepsilon, B \rightarrow 1B \mid \varepsilon\}, S)$$

- $L = \{ 0^n 1^m \mid n, m \geq 0, n < m \}$

Idee umschreiben in $L = \{ 0^n 1^k 1^n \mid k > 0, n \geq 0 \}$

$$G = (\{S, B\}, \{0, 1\}, \{S \rightarrow 0S1 \mid 1B, B \rightarrow 1B \mid \varepsilon\}, S)$$

- Die Sprache L der ausgewogenen Klammerausdrücke. D.h. jede öffnende Klammer muss auch eine schließende Klammer haben.

$$G = (\{S, B\}, \{0, 1\}, \{S \rightarrow (S) \mid \varepsilon \mid SS\}, S)$$

- $L = \{ w \in \Sigma^* \mid \text{Anzahl der 0-Ziffern} = \text{Anzahl der 1-Ziffern in } w \}$

$$G = (\{S, B\}, \{0, 1\}, \{S \rightarrow 0S1 \mid 1S0 \mid \varepsilon \mid SS\}, S)$$

- Nutzen Sie FLACI und prüfen Sie ihre Implementierung auf Korrektheit

Kontextfreie Sprache

Ableitungsbaum

- $L_P = \{ww^r \mid w \in \Sigma^*\}$ (Palindrom)
- L gehört nicht zu den regulären Sprachen, sondern zu den kontextfreien Sprachen.
- Die Grammatik dazu: $G_P = (\{S\}, \{a,b\}, \{S \rightarrow aSa \mid bSb \mid \varepsilon\}, S)$
- Ableitung abbbba
- $S \Rightarrow aSa \Rightarrow abSba \Rightarrow abbSbba \Rightarrow abbbba$
- Dazu kann man auch einen Ableitungsbaum erstellen

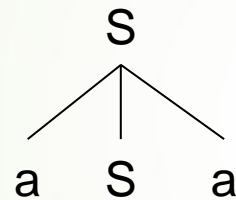
Kontextfreie Sprache

Beispiel Ableitungsbaum

➤ Dazu kann man einen Ableitungsbaum erstellen

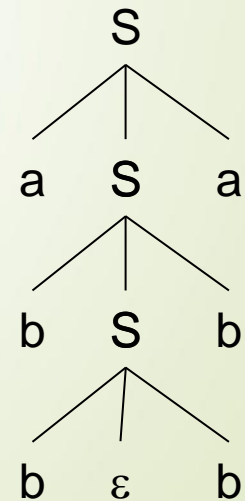
➤ $w = abbbbba$

➤ Regel: $S \rightarrow aSa$ ergibt zum Beispiel:



➤ und damit folgenden Ableitungsbaum

➤ Erstellen Sie Ableitungsbäume mit FLACI



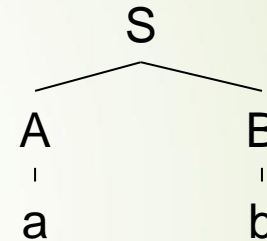
Kontextfreie Sprache

Grammatik: eindeutig

- Die Worte einer Grammatik G sind nicht immer in eindeutiger Weise anhand der Regeln ableitbar.

Beispiel: $S \rightarrow AB$, $A \rightarrow a$, $B \rightarrow b$

- Das Wort $w=ab$ lässt sich ableiten als
 - $S \Rightarrow AB \Rightarrow aB \Rightarrow ab$ (1) oder
 - $S \Rightarrow AB \Rightarrow Ab \Rightarrow ab$ (2)
- Der Ableitungsbaum für beide Ableitungen (1) oder (2) ist jedoch identisch.
- Die Grammatik ist daher **eindeutig**.



Kontextfreie Sprache

Grammatik: mehrdeutig

- Die Worte einer Grammatik G sind nicht immer in eindeutiger Weise anhand der Regeln ableitbar und können auch verschiedene Ableitungsbäume ergeben.

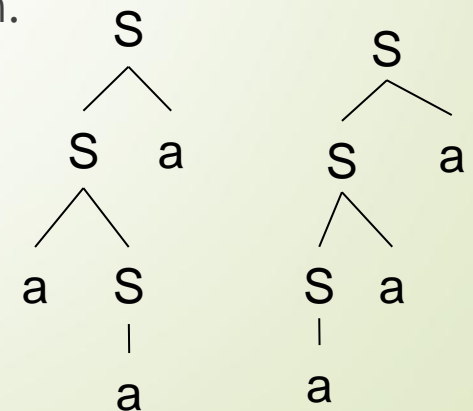
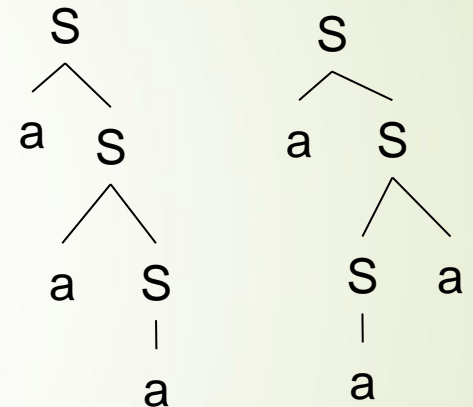
Beispiel: $S \rightarrow aS \mid Sa \mid a$

- Das Wort $w=aaa$ lässt sich ableiten als

- $S \Rightarrow aS \Rightarrow aaS \Rightarrow aaa$ (1) oder
- $S \Rightarrow aS \Rightarrow aSa \Rightarrow aaa$ (2) oder
- $S \Rightarrow Sa \Rightarrow Saa \Rightarrow aaa$ (3) oder
- $S \Rightarrow Sa \Rightarrow aSa \Rightarrow aaa$ (4)

- Der Ableitungsbäume sind alle verschieden.

- Die Grammatik ist somit **mehrdeutig**.



Aufgabe

Ableitungsbäume

Erstellen Sie einen Ableitungsbaum für Worte der Länge 4 zu folgenden Grammatiken $G = (\{S, A, B\}, \{0,1\}, P, S)$. Welche Grammatiken sind mehrdeutig?

1. $P = \{S \rightarrow 0S \mid 1S \mid 0\}$
2. $P = \{S \rightarrow 0S0S \mid 1\}$
3. $P = \{S \rightarrow 0S1 \mid 1A, A \rightarrow 1A \mid 1\}$
4. $P = \{S \rightarrow 1A \mid 0B, A \rightarrow 1AA \mid 0S \mid 0, B \rightarrow 0BB \mid 1S \mid 1\}$

Aufgabe Lösung

Ableitungsbäume

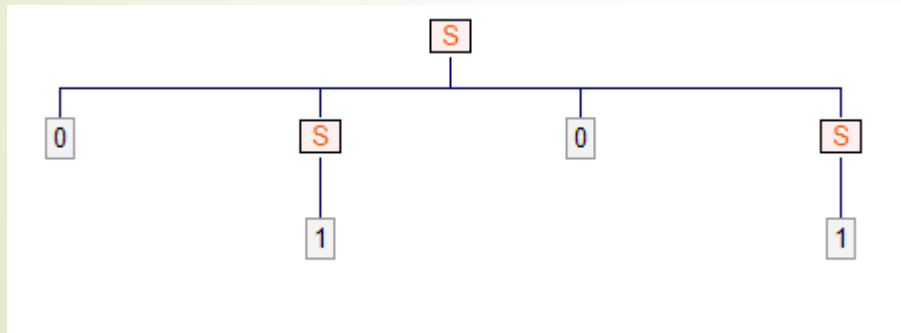
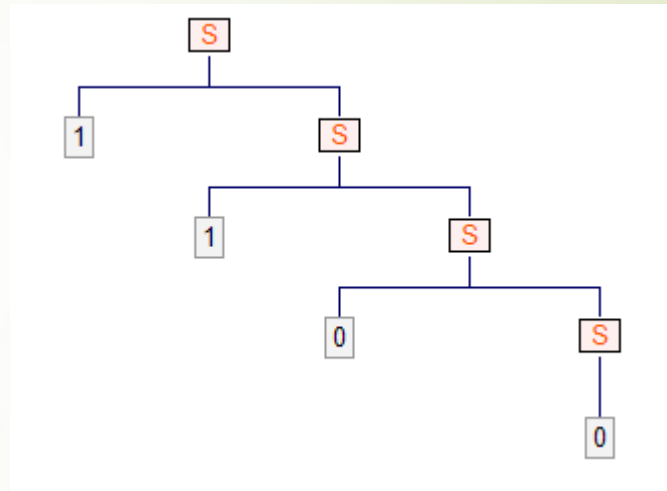
Erstellen Sie einen Ableitungsbaum für Worte der Länge 4 zu folgenden Grammatiken $G = (\{S, A, B\}, \{0,1\}, P, S)$. Welche Grammatiken sind mehrdeutig?

1. $P = \{S \rightarrow 0S \mid 1S \mid 0\}$

$w := 1100$

1. $P = \{S \rightarrow 0S0S \mid 1\}$

$W := 0101$



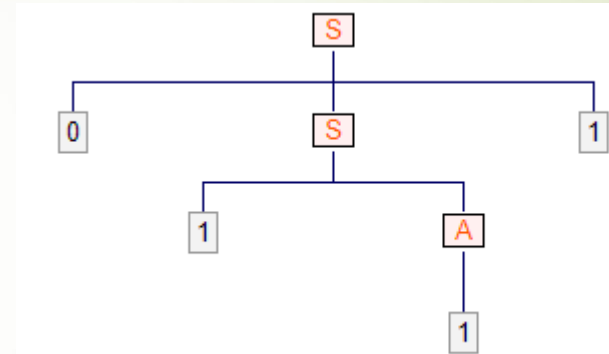
Aufgabe Lösung

Ableitungsbäume

Erstellen Sie einen Ableitungsbaum für Worte der Länge 4 zu folgenden Grammatiken $G = (\{S, A, B\}, \{0,1\}, P, S)$. Welche Grammatiken sind mehrdeutig?

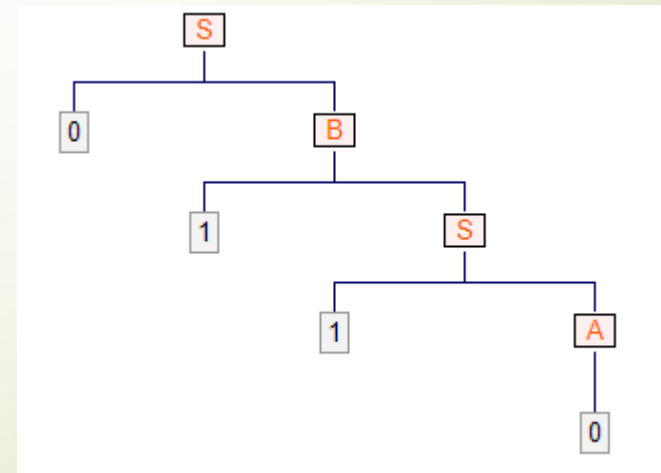
3. $P = \{S \rightarrow 0S1 \mid 1A, A \rightarrow 1A \mid 1\}$

$W := 0111$



4. $P = \{S \rightarrow 1A \mid 0B, A \rightarrow 1AA \mid 0S \mid 0, B \rightarrow 0BB \mid 1S \mid 1\}$

$W := 0110$



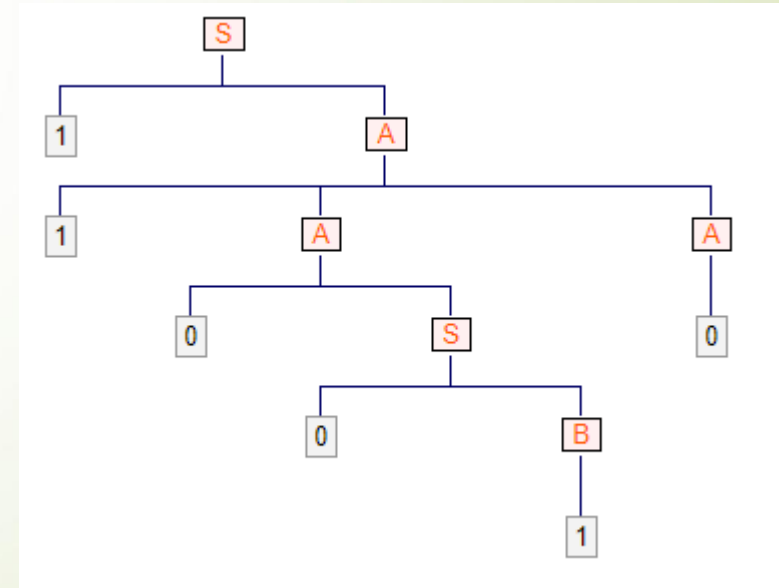
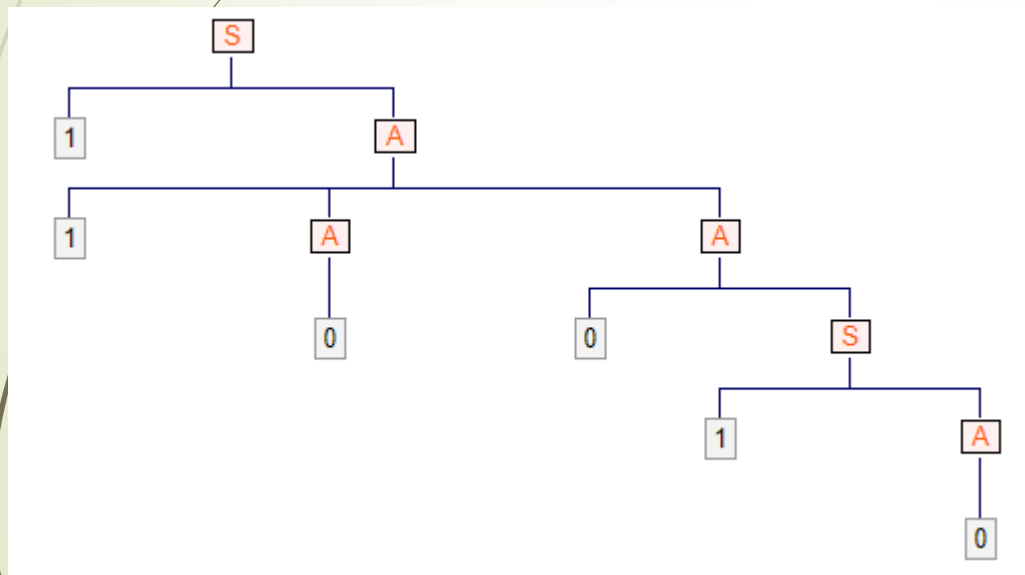
Aufgabe Lösung

Ableitungsbäume

Grammatik 4 ist mehrdeutig?

4. $P = \{S \rightarrow 1A \mid 0B, A \rightarrow 1AA \mid 0S \mid 0, B \rightarrow 0BB \mid 1S \mid 1\}$

Für das Wort $w := 110010$ gibt es z.B. 2 Ableitungsbäume



Kontextfreie Sprache

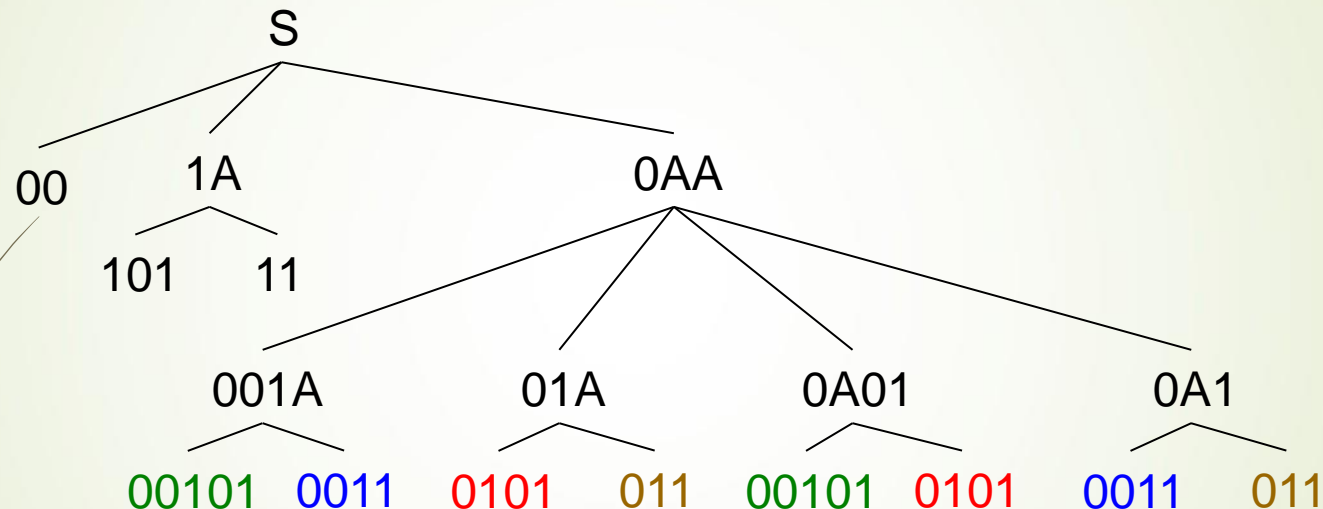
Grammatik: mehrdeutig

- Die Grammatik $G_m = (\{S\}, \{0\}, \{S \rightarrow aS \mid Sa \mid a\}, S)$ ist mehrdeutig, aber sie lässt sich leicht durch eine eindeutige Grammatik ersetzen.
- $G = (\{S\}, \{0\}, \{S \rightarrow aS \mid a\}, S)$
- Eine mehrdeutigen Grammatik impliziert noch lange nicht, dass die Sprache mehrdeutig ist.
- Ziel
 - für eine eindeutige Sprache L auch eine eindeutige Grammatik G mit $L(G) = L$ zu erhalten.

Kontextfreie Sprache

Grammatik: eindeutig/mehrdeutig

- Beispiel: $G = (\{S, A\}, \{0, 1\}, \{S \rightarrow 00 \mid 1A \mid 0AA, A \rightarrow 01 \mid 1\}, S)$



- Die Sprache $L(G)$ hat nur 7 Worte: $L(G) = \{00, 101, 11, 00101, 0011, 0101, 011\}$
- Die Worte: 00101, 0011, 0101 und 011 können auf zwei verschiedene Arten abgeleitet werden, aber es gibt nur einen Ableitungsbaum.

⇒ die Grammatik G ist daher eindeutig.

Kontextfreie Sprache

Grammatik: eindeutig, aber Ersetzungen mehrdeutig

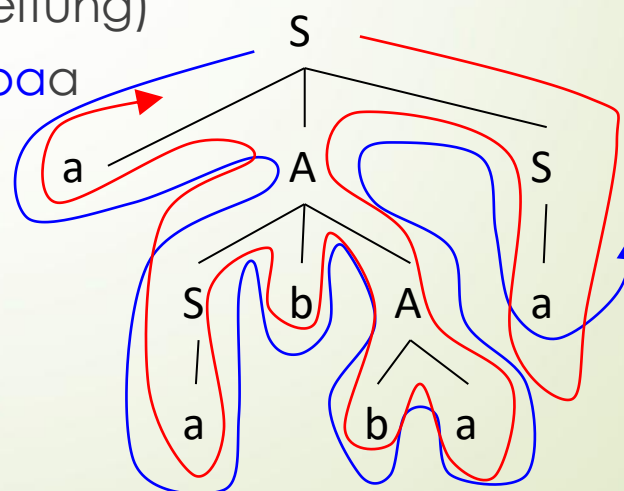
- Mehrdeutigkeit in der Ersetzung ist möglich
- daher erzielt man Eindeutigkeit dadurch, dass:
 - Nichtterminale, die ganz links stehen, werden zuerst ersetzt oder
 - Nichtterminale, die ganz rechts stehen, werden zuerst ersetzt.
- Beispiel: $G_2 = (\{S, A\}, \{a, b\}, \{S \rightarrow aAS \mid a, A \rightarrow SbA \mid SS \mid ba\}, S)$
- Ableiten des Wortes: aabbbaa

$S \Rightarrow aAS \Rightarrow aSbAS \Rightarrow aabAS$

$\Rightarrow aabbaS \Rightarrow aabbbaa$ (Linksableitung)

$S \Rightarrow aAS \Rightarrow aAa \Rightarrow aSbAa \Rightarrow aSbbaa$

$\Rightarrow aabbbaa$ (Rechtsableitung)



Rechtsableitung

Linksableitung

Ableitungsbaum

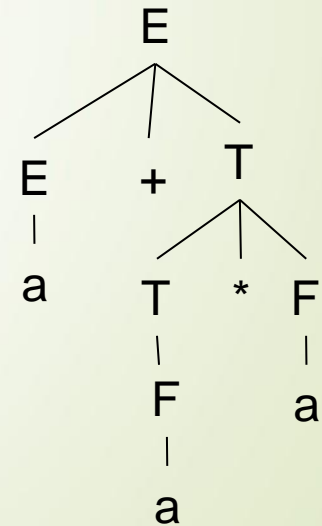
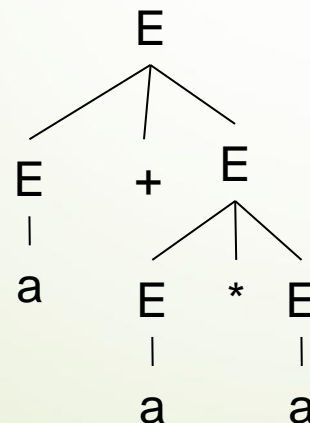
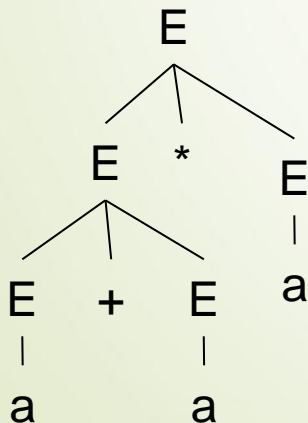
Aufgabe Mehrdeutigkeit

- Zeigen Sie, dass die Grammatik G mehrdeutig ist und die dazugehörige äquivalente Grammatik G_e eindeutig.
 - $G = (\{E\}, \{a, +, *, (,)\}, \{E \rightarrow E+E \mid E^*E \mid (E) \mid a\}, E)$
 - $G_e = (\{E, T, F\}, \{a, +, *, (,)\}, \{E \rightarrow E+T \mid T, T \rightarrow T^*F \mid F, F \rightarrow (E) \mid a\}, E)$
- Beispiel einer inhärent mehrdeutige Sprache
 - $L = \{a^i b^j c^k \mid i=j \text{ oder } j=k \text{ mit } i, j, k > 0\}$

Aufgabe Mehrdeutigkeit

Lösung

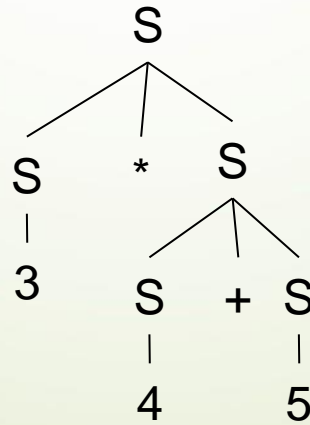
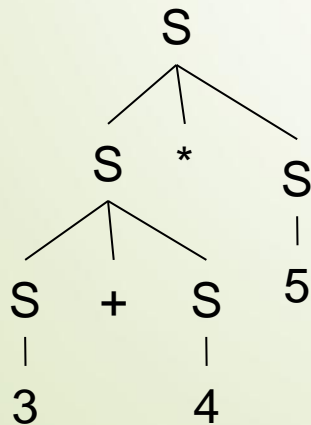
- Zeigen Sie, dass die Grammatik G mehrdeutig ist und die dazugehörige äquivalente Grammatik G_e eindeutig.
- $G = (\{E\}, \{a, +, *, (,)\}, \{E \rightarrow E+E \mid E * E \mid (E) \mid a\}, E)$
- $G_e = (\{E, T, F\}, \{a, +, *, (,)\}, \{E \rightarrow E+T \mid T, T \rightarrow T * F \mid F, F \rightarrow (E) \mid a\}, E)$
- Beispiel
- $a+a*a$



Kontextfreie Sprache

Ableitungsbäume

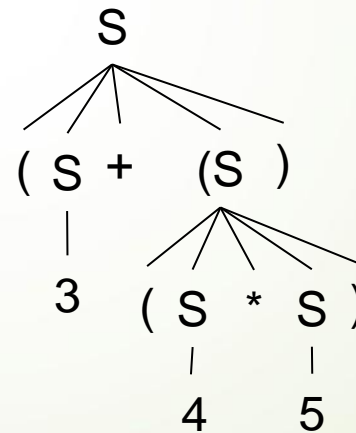
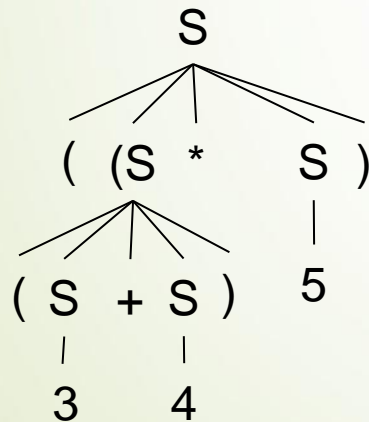
- Neben der Generierung von Worten einer Sprache, können auch Ableitungsbäume zur Berechnung von Ausdrücken verwandt werden
- Beispiel:
 - $G = (\{S\}, \{a \in \mathbb{N}\}, \{S \rightarrow S+S \mid S*S \mid a\}, S)$
 - Mögliche Ausdrücke $3+4*5$, $1+7+8$
 - Was heißt $3+4*5$?
 - $3+(4*5)$ oder $(3+4)*5$
 - Wie bekomme ich die Mehrdeutigkeit weg?



Kontextfreie Sprache

Ableitungsbäume

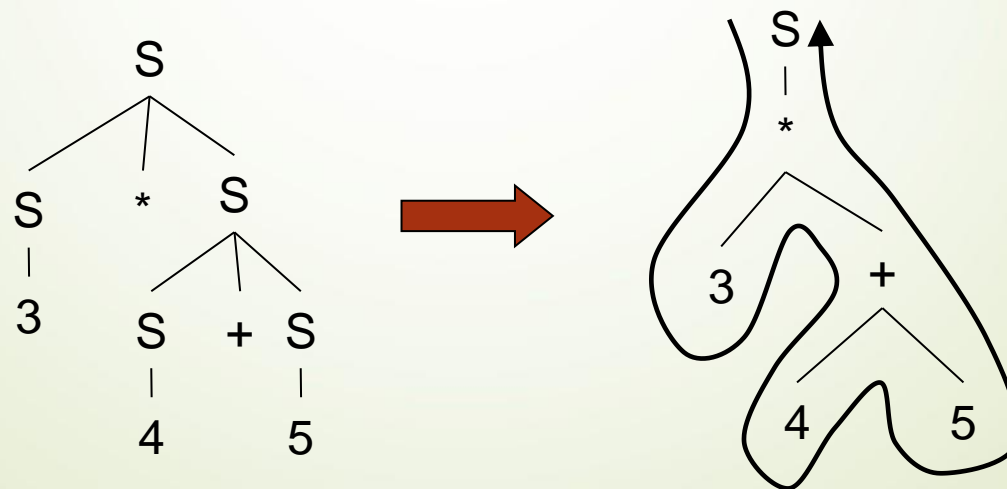
- Durch Klammerung
- $G = (\{S\}, \{ (,), a \in \mathbb{N} \}, \{ S \rightarrow (S+S) \mid (S*S) \mid a \}, S)$
- Was heißt $((3+4)*5)$ oder $(3+(4*5))$?
- Es gibt jedoch noch einen anderen Weg



Kontextfreie Sprache

Berechnungsbäume

- Die Ableitungsbäume erlauben arithmetische Ausdrücke zu berechnen
- Wenn man den Baum durchwandert erhält man $*3+4\ 5$
 - Dies nennt man Prefix-Notation (oder Polnische Notation)
(Operation steht vor den beiden Operanden)
 - erlaubt die Klammerfreie Berechnung von Ausdrücken



Kontextfreie Sprache

Berechnungsbäume

- Durchwandern von Bäumen
 - **Prefix Notation** (Operation wird zuerst ausgegeben, dann der linke Ast rekursiv durchwandert und dann der rechte Ast rekursiv durchwandert)

Nennt man auch Polnische Notation

- **Infix Notation** (zuerst der linke Ast durchwandert dann die Operation ausgegeben und dann der rechte Ast durchwandert)

- **Postfix Notation** (Der linke Ast wird rekursiv durchwandert und dann der rechte Ast rekursiv durchwandert dann wird die Operation ausgegeben)

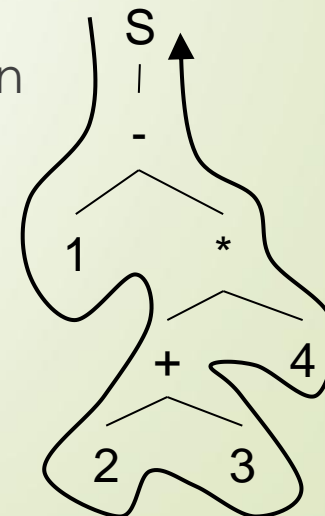
Nennt man auch Umgekehrte Polnische Notation

Beispiel: $(1-(2+3)*4)$

➤ Prefix: $- 1 * + 2 3 4$

➤ Infix: $1 - 2+3*4$

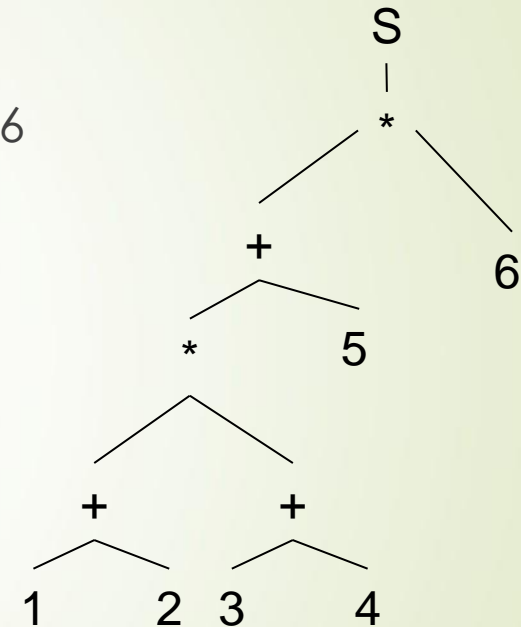
➤ Postfix: $1 2 3 + 4 * -$



Kontextfreie Sprache

Beispiel: Berechnungsbäume

- Beispiel: $((1+2)*(3+4)+5)*6$
- Polnische Notation: $* + * + 1 2 + 3 4 5 6$
- Berechnung:
 - $+ * + 1 2 + 3 4 5 6 \Rightarrow * + * 3 + 3 4 5 6$
 - $+ * 3 + 3 4 5 6 \Rightarrow * + * 3 7 5 6$
 - $* + * 3 7 5 6 \Rightarrow * + 21 5 6$
 - $+ 21 5 6 \Rightarrow * 26 6$
 - $* 26 6 \Rightarrow 156$



Aufgabe Polnische Notation

➤ Wandeln Sie folgende Infix-Notation in Polnische Notation um.

1. $1*2*3$

2. $1*2+3$

3. $1*(2+3)$

4. $((1+2)*3)+4$

5. $1+2*3+4$

Aufgabe Polnische Notation

Lösung Berechnungsbäume

► Wandeln Sie folgende Infix-Notation in Polnische Notation um.

1. $1*2*3$

2. $1*2+3$

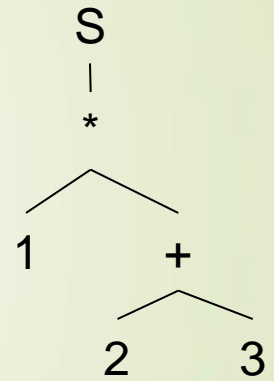
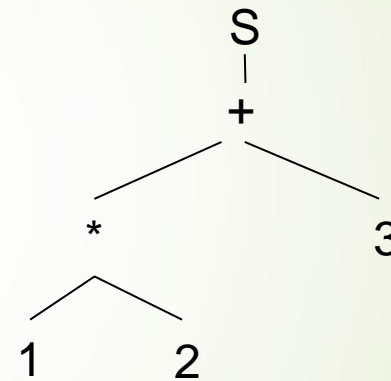
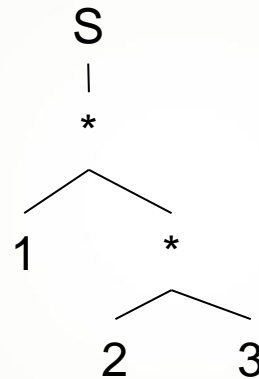
3. $1*(2+3)$

Polnische Notation

1. $*1*23$

2. $+*123$

3. $*1+23$



Aufgabe Polnische Notation

Lösung 2

➤ Wandeln Sie folgende Infix-Notation in Polnische Notation um.

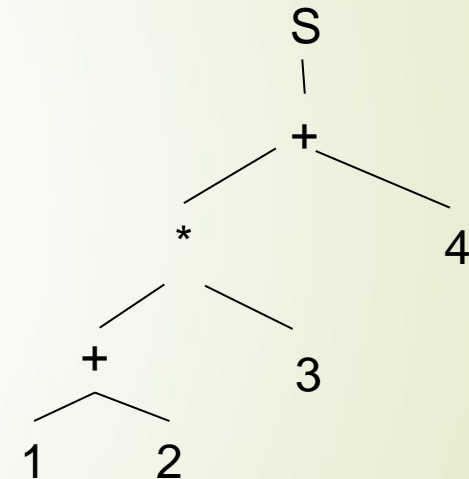
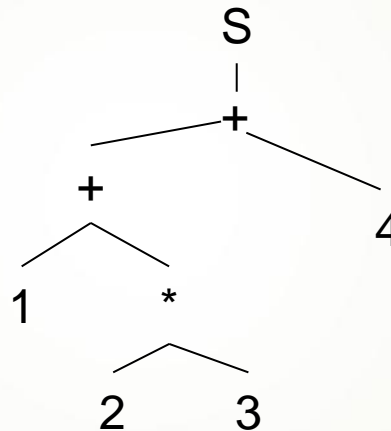
1. $((1+2)*3)+4$

2. $1+2*3+4$

➤ Polnische Notation

1. $++1*234$

2. $+*+1234$



Kontextfreie Sprache

Beispiel arithmetische Ausdrücke

- Erzeugen von arithmetischen Ausdrücken
 Als Stellvertreter für Variablen und Ausdrücken wählen wir a
 Ausdrücke wäre: a , $a+a$, $a*a$, $a+(a+a)$, $a*((a-a)/a)-((a+a)*a)$.
- Was ist das Terminalalphabet?
 - a als Bezeichner für Konstanten und Variablen
 - $()$ Klammersymbolen
 - $+, -, *, /$ als Operatoren
 - Hilfssymbole E für Ausdrücke und O für Operationen
- D.h. wir haben
 - $E \rightarrow a$ und $O \rightarrow + \mid - \mid * \mid /$
 - $E \rightarrow E O E$ zum Verknüpfen von Ausdrücken
 - $E \rightarrow (E)$ zum Einklammern
- $G_A = (\{E, O\}, \{a, (,), +, -, *, /\}, P, E)$
 $P = \{E \rightarrow a \mid E O E \mid (E) \mid , O \rightarrow + \mid - \mid * \mid /\}$

Kontextfreie Sprache

Beispiel 2 Ableitung von Wörter

$$G_A = (\{E, O\}, \{a, (,), +, -, *, /\}, \{E \rightarrow a \mid E O E \mid (E) \mid , O \rightarrow + \mid - \mid * \mid / \}, E)$$

► Ableiten des Ausdrucks: $a^*(((a-a)/a)-((a+a)*a))$

$$E \Rightarrow EOE$$

$$\Rightarrow E^*E$$

$$\Rightarrow E^*(E)$$

$$\Rightarrow E^*(EOE)$$

$$\Rightarrow E^*(E-E)$$

$$\Rightarrow E^*(E-(E))$$

$$\Rightarrow E^*(E-(EOE))$$

$$\Rightarrow E^*(E-(E^*E))$$

$$\Rightarrow E^*(E-((E)^*E))$$

$$\Rightarrow E^*(E-((EOE)^*E))$$

$$\Rightarrow E^*(E-((E+E)^*E))$$

$$E \Rightarrow E^*((E)-((E+E)^*E))$$

$$\Rightarrow E^*((EOE)-((E+E)^*E))$$

$$\Rightarrow E^*((E/E)-((E+E)^*E))$$

$$\Rightarrow E^*(((E)/E)-((E+E)^*E))$$

$$\Rightarrow E^*(((EOE)/E)-((E+E)^*E))$$

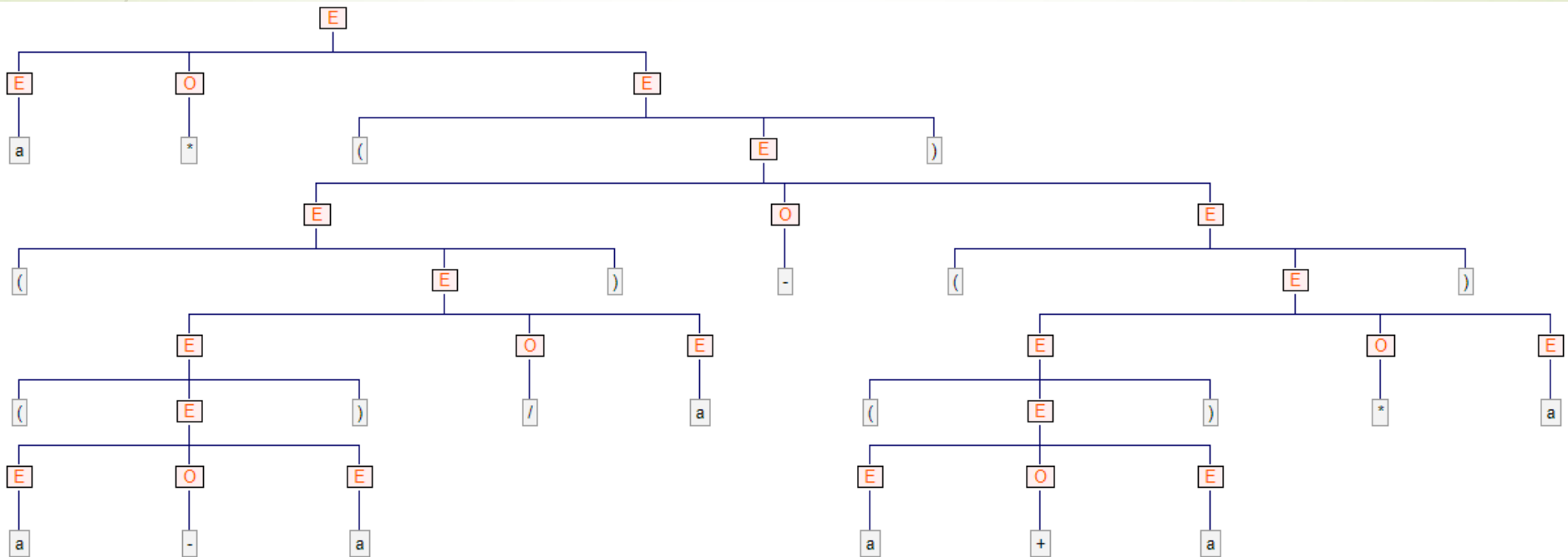
$$\Rightarrow E^*(((E-E)/E)-((E+E)^*E))$$

$$\Rightarrow^* a^*(((a-a)/a)-((a+a)^*a))$$

Kontextfreie Sprache

Beispiel2 Ableitungsbaum

➤ Ableitungsbaum für: $a^*((a-a)/a)-((a+a)*a)$



Kontextfreie Grammatiken

Normalformen

- Jede kontextfreie Grammatik lässt sich vereinfachen und in eine Normalform überführen.
 - Elimination der ε -Regeln
 - Elimination von Kettenregeln
 - Elimination von nutzlosen Variablen
- Gängige Normalformen sind:
 - Chomsky-Normalform
(Produktionen P mit $A, B, C \in N$ und $a \in \Sigma$ nur in der Form:
 $A \rightarrow BC$ oder $A \rightarrow a$)
 - Greibach-Normalform
(Produktionen P mit $a \in \Sigma$ und $X \in N^*$ nur in der Form: $A \rightarrow aX$)

Kontextfreie Grammatiken

Elimination der ε -Regeln

- In jeder kontextfreie Grammatik G lassen sich alle ε -Regeln mit Ausnahme der Regel $S \rightarrow \varepsilon$ eliminieren.
 - Die Grammatiken sind dabei äquivalent.
- Verfahren:
 1. Sei $X \rightarrow vAw$ mit $A \rightarrow \varepsilon$ (A eine ε -Variable), dann füge $X \rightarrow vw$ als Regel hinzu.
 2. Mache das iterativ bis keine neue ε -Variable mehr hinzu kommt.
 3. Lösche alle ε -Regeln bis auf $S \rightarrow \varepsilon$

Kontextfreie Grammatiken

Elimination der ε -Regeln

- Beispiel: $G = (\{S,A,B,C\}, \{0,1,2\}, \{S \rightarrow 0A \mid 1B, A \rightarrow BC, B \rightarrow B2 \mid \varepsilon, C \rightarrow \varepsilon\}, S)$
- 1.Schritt (neue Regeln für die ε -Variablen B und C)
 - $S \rightarrow 0A \mid 1B \mid 1$
 - $A \rightarrow BC \mid B \mid C \mid \varepsilon$
 - $B \rightarrow B2 \mid 2 \mid \varepsilon$
 - $C \rightarrow \varepsilon$
- Nun ist A eine ε -Variablen (Iteration über A)
 - $S \rightarrow 0A \mid 0 \mid 1B \mid 1$
 - $A \rightarrow BC \mid B \mid C \mid \varepsilon$
 - $B \rightarrow B2 \mid 2 \mid \varepsilon$
 - $C \rightarrow \varepsilon$
- Löschen aller ε -Regeln
 - $S \rightarrow 0A \mid 0 \mid 1B \mid 1$
 - $A \rightarrow BC \mid B \mid C$
 - $B \rightarrow B2 \mid 2$

Kontextfreie Grammatiken

Elimination nutzloser Variablen

- Eine Variable heißt **nützlich**, wenn sie in der Ableitung eines Terminalwortes vorkommt, d.h. $S \Rightarrow^* uAv \Rightarrow^* w$ mit $w \in \Sigma^*$ und $u, v \in (N \cup \Sigma)^*$
- A ist nutzlos,
 - wenn sich daraus kein Terminalwort ableiten lässt oder
 - wenn A von Startsymbol aus nicht erreichbar ist.
- Beispiel: Grammatik $G = (\{S, A, B, C\}, \{0, 1, 2\}, P, S)$ mit P:

$S \rightarrow 0A \mid 0 \mid 1B \mid 1$

$A \rightarrow BC \mid B \mid C$

$B \rightarrow B2 \mid 2$

- die Variable C ist nutzlos. D.h. eliminieren alle Produktionen, wo C vorkommt.

$S \rightarrow 0A \mid 0 \mid 1B \mid 1$

$A \rightarrow B$

$B \rightarrow B2 \mid 2$

Kontextfreie Grammatiken

Kettenregel entfernen

- Beispiel: $G = (\{S, A, B, C\}, \{0, 1, 2\}, \{S \rightarrow 0A \mid 1B, A \rightarrow BC, B \rightarrow B2 \mid \varepsilon, C \rightarrow \varepsilon\}, S)$
 - $S \rightarrow 0A \mid 0 \mid 1B \mid 1$
 - $A \rightarrow B$
 - $B \rightarrow B2 \mid 2$
- Variable A durch B ersetzen (da Regel $A \rightarrow B$)
 - $S \rightarrow 0B \mid 0 \mid 1B \mid 1$
 - $B \rightarrow B2 \mid 2$

Kontextfreie Grammatiken

Elimination von Kettenregeln

- In jeder kontextfreie Grammatik G lassen sich alle Kettenregeln eliminieren.
- Eine Kettenregel ist eine Regeln von der Form $A \rightarrow B$ mit $B \in N$
- Def: $[A]^* = \{B \in N \mid \text{mit } A \rightarrow^* B\}$
- Verfahren:
 1. Zunächst Zyklen eliminieren. Ein Zyklus ist eine Menge A_1, \dots, A_k von Variablen mit: $A_1 \rightarrow A_2, A_2 \rightarrow A_3, \dots, A_{k-1} \rightarrow A_k, A_k \rightarrow A_1$. Der Zyklus wird entfernt, indem die zyklische Regel gelöscht wird und alle Variablen A_1, \dots, A_k durch eine Variable B ersetzt werden.
 2. Für alle $A \in N$
 1. bestimme $[A]^*$. Lösche anschließend alle Kettenregeln
 2. Für jedes $B \in [A]^*$ und jede Regel $B \rightarrow w$: Füge die Regel $A \rightarrow w$ hinzu.

Kontextfreie Grammatiken

Elimination von Kettenregeln

- Beispiel: $G = (\{S, A, B, C\}, \{a, b, c, d\}, P, S)$ mit P :
 - $S \rightarrow A \mid bB$
 - $A \rightarrow B \mid C$
 - $B \rightarrow A \mid Cc$
 - $C \rightarrow c \mid d$
- 1. Schritt Zyklus $A \rightarrow B, B \rightarrow A$ entfernen. A, B werden durch X ersetzt.
 - $S \rightarrow X \mid bX$
 - $X \rightarrow C \mid Cc$
 - $C \rightarrow c \mid d$
- 2. Schritt $[S]^*$, $[X]^*$ und $[C]^*$ bestimmen
 - $[S]^* = \{X, C\}$, $[X]^* = \{C\}$, $[C]^* = \emptyset$
 - $X \rightarrow C$ und $S \rightarrow X$ löschen.
- 3. Schritt Regeln hinzufügen ($S \rightarrow c \mid d \mid Cc$ und $X \rightarrow c \mid d$)
 - $S \rightarrow c \mid d \mid Cc \mid bX$
 - $X \rightarrow c \mid d \mid Cc$
 - $C \rightarrow c \mid d$

Aufgabe Vereinfachung

- Falls möglich, vereinfachen Sie folgende Grammatiken G
- $G = (\{S, A, B, C, D\}, \{0, 1\},$
 $\{S \rightarrow 00B \mid 1A, A \rightarrow B \mid C, B \rightarrow 1B \mid 0 \mid AD, C \rightarrow BD \mid AD, D \rightarrow \varepsilon\}, S)$

Aufgabe Vereinfachung

Lösung

➤ $G = (\{S, A, B, C, D\}, \{0, 1\}, \{S \rightarrow 00B \mid 1A, A \rightarrow B \mid C, B \rightarrow 1B \mid 0 \mid AD, C \rightarrow BD \mid AD, D \rightarrow \varepsilon\}, S)$

➤ Epsilon-Regel entfernen

~~$S \rightarrow 00B$~~
 ~~$S \rightarrow 1A$~~
 ~~$A \rightarrow B$~~
 ~~$A \rightarrow C,$~~
 ~~$B \rightarrow 1B$~~
 ~~$B \rightarrow 0$~~
 ~~$B \rightarrow AD$~~
 ~~$C \rightarrow BD$~~
 ~~$C \rightarrow AD$~~
 ~~$D \rightarrow \varepsilon$~~

$S \rightarrow 00B$
 $S \rightarrow 1A$
 $A \rightarrow B$
 $A \rightarrow C,$
 $B \rightarrow 1B$
 $B \rightarrow 0$
 $B \rightarrow AD \mid A$
 $C \rightarrow BD \mid B$
 $C \rightarrow AD \mid A$

Zyklen ersetzen A,B,C durch X

$S \rightarrow 00B$
 $S \rightarrow 1A$
 $A \rightarrow B$
 $A \rightarrow C,$
 $B \rightarrow 1B$
 $B \rightarrow 0$
 $B \rightarrow AD \mid A$
 $C \rightarrow BD \mid B$
 $C \rightarrow AD \mid A$

$S \rightarrow 00X$
 $S \rightarrow 1X$
 $X \rightarrow X$
 $X \rightarrow X,$
 $X \rightarrow 1X$
 $X \rightarrow 0$
 $X \rightarrow XD \mid X$
 $X \rightarrow XD \mid X$
 $X \rightarrow XD \mid X$

$S \rightarrow 00X$
 $S \rightarrow 1X$
 $X \rightarrow 1X$
 $X \rightarrow 0$
 $X \rightarrow XD$

$S \rightarrow 00X$
 $S \rightarrow 1X$
 $X \rightarrow 1X$
 $X \rightarrow 0$
 $X \rightarrow XD$

Nutzlose Variable D entfernen

$S \rightarrow 00X$
 $S \rightarrow 1X$
 $X \rightarrow 1X$
 $X \rightarrow 0$

Chomsky Normalform

- Definition: Chomsky Normalform

Eine kontextfreie Grammatik mit $\varepsilon \notin L(G)$ heißt in Chomsky-Normalform, wenn alle ihre Regeln von einer der beiden Formen sind:

$$A \rightarrow BC \text{ oder}$$

$$A \rightarrow a$$

mit $A, B, C \in N$ und $a \in \Sigma$

- Jede kontextfreie Grammatik G mit $\varepsilon \notin L(G)$ lässt sich in Chomsky-Normalform transformieren.**

Chomsky Normalform

Transformation 1: Beispiel

Beispiel: Grammatik $G = (\{S, A, B, C\}, \{0, 1, 2, 3\}, P, S)$ mit P

$$S \rightarrow 0A1 \mid 0C$$

$$A \rightarrow B0B \mid 0 \mid C$$

$$B \rightarrow BA \mid 1$$

$$C \rightarrow C2 \mid 3C$$

■ Elimination von C (nutzlos)

$$S \rightarrow 0A1$$

$$A \rightarrow B0B \mid 0$$

$$B \rightarrow BA \mid 1$$

■ Für jedes Terminalsymbol $\{0, 1\}$ eine Regel $X_0 \rightarrow 0$, $X_1 \rightarrow 1$ einführen und entsprechend in der Regelmenge ersetzen.

$$S \rightarrow X_0AX_1$$

$$A \rightarrow BX_0B \mid 0$$

$$B \rightarrow BA \mid 1$$

$$X_0 \rightarrow 0$$

$$X_1 \rightarrow 1$$

Chomsky Normalform

Transformation 2: Beispiel

$$\begin{aligned} S &\rightarrow X_0 A X_1 \\ A &\rightarrow B X_0 B \mid 0 \\ B &\rightarrow B A \mid 1 \\ X_0 &\rightarrow 0 \\ X_1 &\rightarrow 1 \end{aligned}$$

➤ Schließlich mehrfach Produktionen: $A \rightarrow B_1 B_2 \dots B_k$ ersetzen durch
 $A \rightarrow B_1 W_1, W_1 \rightarrow B_2 W_2 \dots W_{k-2} \rightarrow B_{k-1} B_k$

$$\begin{aligned} S &\rightarrow X_0 W_1 \\ W_1 &\rightarrow A X_1 \\ A &\rightarrow B W_2 \mid 0 \\ W_2 &\rightarrow X_0 B \\ B &\rightarrow B A \mid 1 \\ X_0 &\rightarrow 0 \\ X_1 &\rightarrow 1 \end{aligned}$$

Aufgabe Chomsky-Normalform

- Überführen Sie folgende Grammatiken $G = (\{S, A, B\}, \{0,1\}, P, S)$ in Chomsky-Normalform.
 1. $P = \{S \rightarrow OSO \mid SS0 \mid 0\}$
 2. $P = \{S \rightarrow SOS \mid SOS1S \mid S1SOS \mid \varepsilon\}$
 3. $P = \{S \rightarrow ABABAB, A \rightarrow 0 \mid \varepsilon, B \rightarrow 1\}$
 4. $P = \{S \rightarrow 0A \mid B1, A \rightarrow S, B \rightarrow 1B \mid 1\}$

Aufgabe Chomsky-Normalform

Lösungen 1 und 2

- Grammatiken $G = (\{S, A, B\}, \{0,1\}, P, S)$ in Chomsky-Normalform.

1. $P = \{S \rightarrow OSO \mid SS0 \mid 0\}$

- $S \rightarrow TW \mid SW \mid 0$

- $T \rightarrow 0$

- $W \rightarrow ST$

2. $P = \{S \rightarrow SOS \mid SOS1S \mid S1SOS \mid \epsilon\}$

1. Terminale ersetzen

1. $S \rightarrow SXS \mid SXSYS \mid SYSXS \mid \epsilon$

2. $X \rightarrow 0$

3. $Y \rightarrow 1$

2. Doppelregeln $W \rightarrow XS, T \rightarrow YS$

1. $S \rightarrow SW \mid SWT \mid STW \mid \epsilon$

2. $W \rightarrow XS$

3. $T \rightarrow YS$

4. $X \rightarrow 0$

5. $Y \rightarrow 1$

- Doppelregeln $A \rightarrow WT, B \rightarrow TW$

1. $S \rightarrow SW \mid SA \mid SB \mid \epsilon$

2. $W \rightarrow XS$

3. $T \rightarrow YS$

4. $X \rightarrow 0$

5. $Y \rightarrow 1$

6. $A \rightarrow WT$

7. $B \rightarrow TW$

Aufgabe Chomsky-Normalform

Lösungen 3

■ Grammatiken Chomsky-Normalform

■ $G = (\{S, A, B\}, \{0,1\}, P, S)$ $P = \{S \rightarrow ABABAB, A \rightarrow 0 \mid \varepsilon, B \rightarrow 1\}$

1. ε -Regel beseitigen ($A \rightarrow \varepsilon$)

$S \rightarrow ABABAB \mid BABAB \mid ABBAB \mid ABABB$

$S \rightarrow BBAB \mid BABB \mid ABBB \mid BBB$

$A \rightarrow 0$

$B \rightarrow 1$

2. Doppelregel $W \rightarrow AB$ einführen

$S \rightarrow WWW \mid BWW \mid WBW \mid WWB$

$S \rightarrow BBW \mid BWB \mid WBB \mid BBB$

2. Doppelregel $X \rightarrow BW, Y \rightarrow WB$ einführen

$S \rightarrow WWW \mid XW \mid WX \mid WY$

$S \rightarrow BX \mid XB \mid YB \mid BBB$

3. Doppelregel $U \rightarrow WW, Z \rightarrow BB$ einführen

$S \rightarrow UW \mid XW \mid WX \mid WY$

$S \rightarrow BX \mid XB \mid YB \mid ZB$

fertig

■ Chomsky-Normalform

$S \rightarrow UW \mid XW \mid WX \mid WY$

$S \rightarrow BX \mid XB \mid YB \mid ZB$

$W \rightarrow AB$

$X \rightarrow BW,$

$Y \rightarrow WB$

$U \rightarrow WW$

$Z \rightarrow BB$

$A \rightarrow 0$

$B \rightarrow 1$

Aufgabe Chomsky-Normalform

Lösungen 4

► Grammatiken Chomsky-Normalform

$P = \{S \rightarrow 0A \mid B1, A \rightarrow S, B \rightarrow 1B \mid 1\}$

► $S \rightarrow XS \mid BY$

► $B \rightarrow YB$

► $B \rightarrow 1$

► $X \rightarrow 0$

► $Y \rightarrow 1$

Greibach-Normalform

- Neben der Chomsky-Normalform gibt es auch Greibach-Normalform. Die Regeln sind alle vom Typ:
 - $L \rightarrow aN^*$ (N: Nichtterminale, a die Terminale der Grammatik)
 - In jeden Schritt lässt sich ein Zeichen ableiten. Um ein Wort der Länge n zu erkennen braucht man höchstens n-Schritte.
 - Die Umwandlung einer kontextfreien Grammatik in die Greibach-Normalform ist recht aufwändig. Daher belassen wir es mit zwei Beispielen für Grammatiken in der Greibach-Normalform
 - Beispiele
 - $G_1 = (\{S, B\}, \{a, b\}, \{S \rightarrow aB \mid aSB, B \rightarrow b\}, S)$
 - $G_2 = (\{S, A, B\}, \{a, b\}, \{S \rightarrow aA \mid bB \mid aSA \mid bSB, A \rightarrow a, B \rightarrow b\}, S)$
- Welche Sprachen erzeugen diese Grammatiken?

Darstellung kontextfreier Grammatiken

- Die erweiterte Backus-Naur-Form (EBNF) ist eine effiziente Darstellungsform für kontextfreie Grammatiken.
- Verwendet für die Darstellung von Programmier- und Dialogsprachen.
- Entwickelt durch Backus und Naur.
 - Diese waren in den 50 und 60 Jahren wesentlich an der Entwicklung von Fortran und Algol beteiligt.

Erweiterterte Backus-Naur-Form (EBNF)

Definition

- Seien Σ und N zwei Alphabete mit Σ terminales und N nicht terminales Alphabet. Dann gilt:
 - Jedes $a \in \Sigma$ und jedes $A \in N$ sowie ε sind Elemente von EBNF
 - Sind a_1, a_2, \dots, a_k , $k \geq 1 \in \text{EBNF}$ dann auch $(a_1 a_2 \dots a_k)$ sowie $(a_1 \mid a_2 \mid \dots \mid a_k)$
 - Sind a, b und $c \in \text{EBNF}$, dann auch $a\{b\}^*c$ oder alternativ $a\{b\}c \in \text{EBNF}$
 - Sind a, b und $c \in \text{EBNF}$, dann auch $a\{b\}^0 \text{ oder } 1 c$ oder alternativ $a[b]c \in \text{EBNF}$
- Sei $A \in N$ und x eine EBNF, dann heißt $A ::= x$ eine erweiterte Backus-Naur-Regel
- Eine Grammatik $G = \{N, \Sigma, P, S\}$, wobei $S \in N$ das Startsymbol und P eine endliche Menge von erweiterten Backus-Naur-Regeln ist, heißt erweiterte Backus-Naur Grammatik.

Definition der Sprache

- Festlegen der Ableitungsregeln ($A \in N, a, b, c \in \Sigma$)
 - Bisher wurde ein nicht-terminals Symbol A durch ein Wort w über $N \cup \Sigma$ ersetzt. (Expansionsschritt)
 $aAc \Rightarrow abc$, falls eine Regel $A \rightarrow b$ existierte mit $A \in N$ und $a, b, c \in \Sigma$
- Außer diesen Regeln gibt es noch (Reduktionsschritt):
 - Ist ein Wort $w = a(b_1 \mid b_2 \mid \dots \mid b_k)c$ abgeleitet, dann gilt $a(b_1 \mid b_2 \mid \dots \mid b_k) \Rightarrow ab_i c$ für ein $1 \leq i \leq k$
 - Ist ein Wort $w = a\{b\}c$ abgeleitet, dann gilt $a\{b\}c \Rightarrow ab^i c$ für ein $i \geq 0$
 - Ist ein Wort $w = a[b]c$ abgeleitet, dann gilt $a[b]c \Rightarrow ac$ oder $a[b]c \Rightarrow abc$
- Die Symbole \mid und $*$ werden wie bei den regulären Ausdrücken interpretiert und $[a]$ entspricht dem regulären Ausdruck $(\epsilon \mid a)$
- Damit lässt sich die Sprache der erweiterten Backus-Naur-Grammatik definieren: $L(G) = \{ w \in \Sigma^* \mid S \Rightarrow^* w \}$

EBNF

Beispiel: Ableitung

■ Gegeben $G_1 = (\{int, v_z, z_f, z_1, z_0\}, \{+, -, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}, P, int)$

$$\begin{aligned}
 P = \{ & int ::= v_z z_f, \\
 & v_z ::= [+ \mid -], \\
 & z_f ::= (0 \mid z_1 \{z_0\}), \\
 & z_1 ::= (1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9), \\
 & z_0 ::= (0 \mid z_1) \}
 \end{aligned}$$

Ableiten der Zahl: 7305:

$$\begin{aligned}
 int &\Rightarrow v_z z_f \Rightarrow [+ \mid -] z_f \Rightarrow z_f \\
 &\Rightarrow (0 \mid z_1 \{z_0\}) \Rightarrow z_1 \{z_0\} \Rightarrow z_1 z_0 z_0 z_0 z_0 \\
 &\Rightarrow (1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9) z_0 z_0 z_0 z_0 \Rightarrow 7 z_0 z_0 z_0 z_0 \\
 &\Rightarrow 7(0 \mid z_1) z_0 z_0 \Rightarrow 7 z_1 z_0 z_0 \Rightarrow 7(1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9) z_0 z_0 \Rightarrow 73 z_0 z_0 \\
 &\Rightarrow 73(0 \mid z_1) z_0 \Rightarrow 730 z_0 \\
 &\Rightarrow 730(0 \mid z_1) \Rightarrow 730 z_1 \Rightarrow 730(1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9) \Rightarrow 7305
 \end{aligned}$$

EBNG ist äquivalent zu den kontextfreien Grammatiken

- Die Sprache einer erweiterten Backus-Naur-Grammatik L_{BN} ist äquivalent der Sprache einer kontextfreien Grammatik L_{kf}

Beweis durch Konstruktion:

1) Jedes $A \rightarrow a \Rightarrow A ::= a$

d.h. jede kontextfreie Sprache lässt sich durch eine EBNG beschreiben

2) Jede Backus-Naur-Regel lässt sich in eine kontextfreie Regel transformieren.

$A ::= (a_1 a_2 \dots a_k) \Rightarrow A \rightarrow a_1 a_2 \dots a_k$

$A ::= (a_1 \mid a_2 \mid \dots \mid a_k) \Rightarrow A \rightarrow a_1, A \rightarrow a_2, \dots, A \rightarrow a_k$

$A ::= a\{b\}c \Rightarrow A \rightarrow aBc, B \rightarrow bB, B \rightarrow \varepsilon$

mit B ein neues nicht-terminals Symbol

$A ::= a[b]c \Rightarrow A \rightarrow ac, A \rightarrow abc$

Mit diesem Verfahren alle Regeln transformieren.

EBNF

Beispiel: EBNG ist äquivalent zu den kontextfreien Grammatiken

- Gegeben $G_1 = (\{int, v_z, z_f, z_1, z_0\}, \{+, -, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}, P, int)$

$$P = \{ int ::= v_z z_f,$$

$$v_z ::= [+ \mid -],$$

$$z_f ::= (0 \mid z_1 \{z_0\}),$$

$$z_1 ::= (1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9),$$

$$z_0 ::= (0 \mid z_1) \}$$

$$int ::= v_z z_f \quad \Rightarrow \quad int \rightarrow v_z z_f$$

$$v_z ::= [+ \mid -] \quad \Rightarrow \quad v_z \rightarrow + \mid - \mid \varepsilon$$

$$z_f ::= (0 \mid z_1 \{z_0\}) \Rightarrow z_f \rightarrow 0 \text{ und } z_f \rightarrow z_1 \{z_0\}$$

$$z_f \rightarrow z_1 \{z_0\} \text{ wird transformiert in } z_f \rightarrow z_1 B, B \rightarrow z_0, B \rightarrow \varepsilon$$

$$z_1 ::= (1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9) \Rightarrow z_1 \rightarrow 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

$$z_0 ::= (0 \mid z_1) \quad \Rightarrow \quad z_0 \rightarrow 0 \mid z_1$$

EBNF

vereinfachte Regel Darstellung

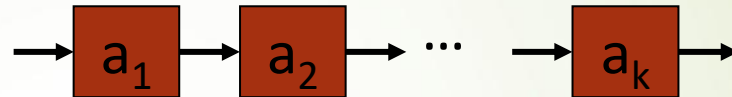
- Wenn man sich EBNF genauer ansieht, lassen sich mit der dortigen Notation die Regeln regulärer Grammatiken kompakter schreiben
 - $A \rightarrow a_1, A \rightarrow a_2, \dots, A \rightarrow a_k$ kompakter $A \rightarrow a_1 \mid a_2 \mid \dots \mid a_k$
 - $A \rightarrow aBc, B \rightarrow bB, B \rightarrow \varepsilon$ kompakter $A \rightarrow a\{b\}c$
 - $A \rightarrow ac, A \rightarrow abc$ kompakter $A \rightarrow a[b]c$
- und graphisch darstellen (Syntaxdiagramme).

EBNF

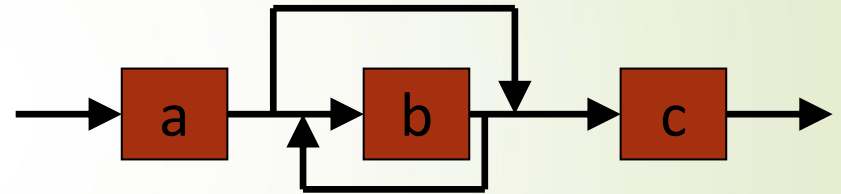
Syntaxdiagramme

➤ Grafische Darstellung der Regeln

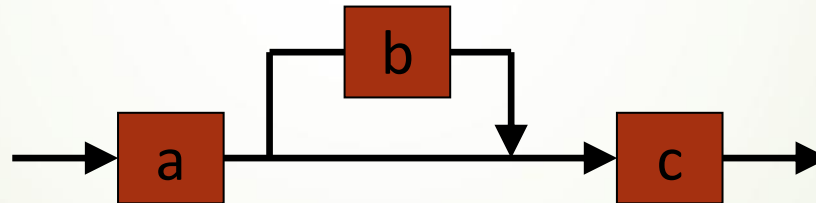
➤ $A \rightarrow a_1 a_2 \dots a_k$



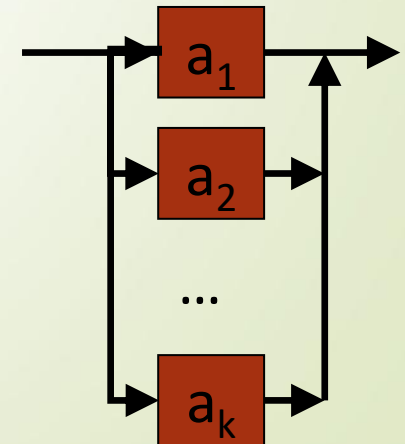
➤ $A \rightarrow a\{b\}c$



➤ $A \rightarrow a[b]c$



➤ $A \rightarrow a_1 \mid a_2 \mid \dots \mid a_k$



➤ Darstellung kontextfreien Grammatiken durch Syntaxdiagramme

➤ Ein Pfeil markiert das Startsymbol

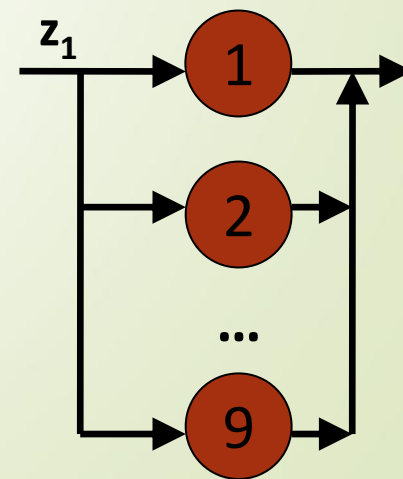
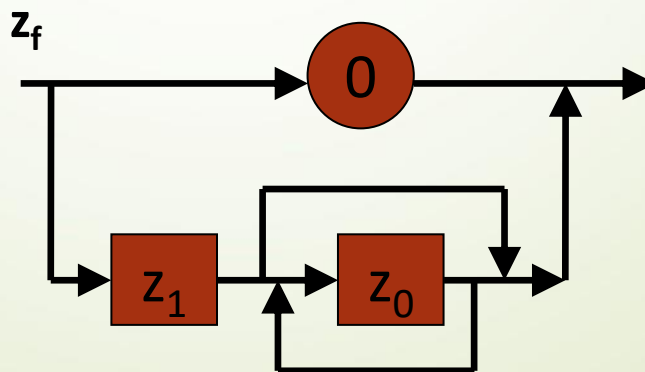
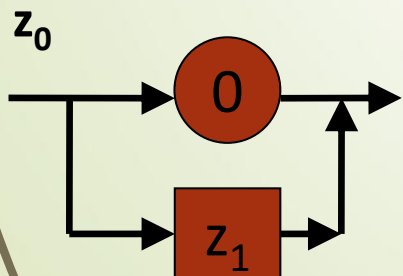
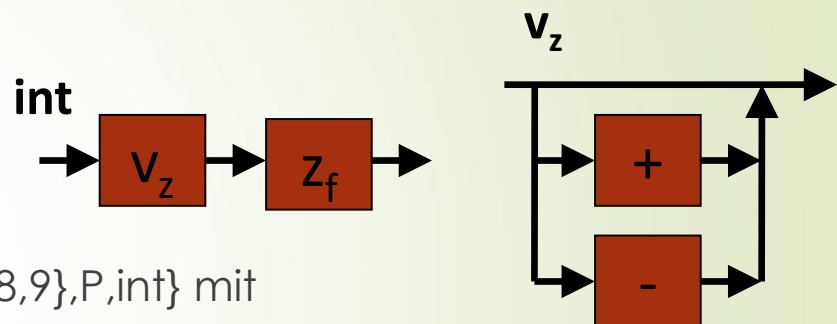
➤ Terminale sind Kreise

➤ Nicht-Terminale sind Rechtecke

➤ Beispiel $G_1 = (\{int, v_z, z_f, z_1, z_0\}, \{+, -, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}, P, int)$ mit

$P = \{ int ::= v_z z_f, v_z ::= [+ | -], z_f ::= (0 | z_1 \{z_0\}), z_1 ::= (1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9), z_0 ::= (0 | z_1) \}$

Bauen Sie die Grammatik in FLACI nach!



- $G_1 = (\{int, v_z, z_f, z_1, z_0\}, \{+, -, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}, P, int)$ mit
 $P = \{ int ::= v_z z_f, v_z ::= [+ | -], z_f ::= (0 | z_1 \{z_0\}), z_1 ::= (1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9), z_0 ::= (0 | z_1) \}$

Produktionsregeln in FLACI

Int \rightarrow VZ ZF

VZ $\rightarrow + \mid - \mid \text{EPSILON}$

ZF $\rightarrow 0 \mid Z1 ZE$

ZE $\rightarrow Z0 ZE \mid \text{EPSILON}$

Z0 $\rightarrow 0 \mid Z1$

Z1 $\rightarrow 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

Eigenschaften kontextfreier Sprachen

- Es gibt ein Pumping-Lemma für diese Sprachen
 - Sei L eine kontextfreie Sprache. Dann existiert eine Zahl $p \geq 0$ mit $p \in \mathbb{N}$, so dass sich jedes Wort $w \in L$ mit $|w| \geq p$ in der folgenden Form schreiben lässt:

$$w = uvwxy \text{ mit } |vx| \geq 1 \text{ und } |vwx| \leq p$$

und $\forall i \geq 0$ gilt uv^iwx^iy gehört auch zu L .

- Die kontextfreien Sprachen sind abgeschlossen unter:
 - Vereinigung
 - Konkatenation
 - Kleene-Stern
- Die kontextfreien Sprachen sind **nicht** abgeschlossen unter:
 - Durchschnittsbildung
 - Komplement

Kontextfreier Sprachen

Abgeschlossenheit

- Seien $L_1 = L(G_1)$ und $L_2 = L(G_2)$ zwei kontextfreie Sprachen, die durch die Grammatiken
 - $G_1 = (N_1, \Sigma, P_1, S_1)$ und
 - $G_2 = (N_2, \Sigma, P_2, S_2)$ definiert sind

Kontextfreie Sprachen sind abgeschlossen unter

- Vereinigung** $L(G) = L_1 \cup L_2$
 - $G = (N_1 \cup N_2 \cup \{S\}, \Sigma, P_1 \cup P_2 \cup \{S \rightarrow S_1 \mid S_2\}, S)$
 - ist eine kontextfreie Sprache.
- Konkatenation** $L(G) = L_1 L_2$
 - $G = (N_1 \cup N_2 \cup \{S\}, \Sigma, P_1 \cup P_2 \cup \{S \rightarrow S_1 S_2\}, S)$
 - ist eine kontextfreie Sprache.
- Kleene-Stern** $L(G) = L_1^*$
 - $G = (N_1 \cup \{S\}, \Sigma, P_1 \cup \{S \rightarrow SS_1 \mid \varepsilon\}, S)$
 - ist eine kontextfreie Sprache.

Kontextfreier Sprachen

Nichtabgeschlossenheit

Kontextfreie Sprachen sind nicht abgeschlossen unter

➤ Durchschnittsbildung

Gegenbeispiel:

- $L_1 = \{a^i b^k c^k \mid i, k \geq 1\}$ und $L_2 = \{a^i b^k c^k \mid i, k \geq 1\}$
- Sowohl L_1 als auch L_2 gehören zu den kontextfreien Sprachen
 - mit $G_1 = (\{S, A, B\}, \{a, b, c\}, \{S \rightarrow AB, A \rightarrow aAb \mid ab, B \rightarrow cB \mid c\}, S)$
 - und $G_2 = (\{S, C, D\}, \{a, b, c\}, \{S \rightarrow CD, C \rightarrow aC \mid a, D \rightarrow bDc \mid bc\}, S)$
- Die Schnittmenge $L = L_1 \cap L_2 = \{a^i b^i c^i \mid i \geq 1\}$ ist nicht kontextfrei.
- und Komplement.

Da $L_1 \cap L_2 = (L_1^c \cup L_2^c)^c \Rightarrow L^c$ kann auch nicht kontextfrei sein.

Entscheidungsprobleme

- Der Zweck eines endlichen Automaten ist die Prüfung, ob ein gegebenes Wort zu seiner Sprache gehört. (Wortproblem)
- Auch andere Entscheidungsprobleme sind hier relevant.
- Für kontextfreie Sprachen können diese fast alle mit ja beantwortet werden.

Problem	Gegeben	Gefragt	Entscheidbar
Wortproblem	L und $w \in \Sigma^*$	Gilt $w \in L$?	Ja
Leerheitsproblem	L	Gilt $L = \emptyset$?	Ja
Endlichkeitsproblem	L	Gilt $ L < \infty$?	Ja
Äquivalenzproblem	L_1 und L_2	$L_1 = L_2$?	nein

Entscheidungsprobleme

- Das **Wortproblem** ist entscheidbar
 - Z.B. lösen mit Hilfe des CYK-Algorithmus
- Das **Leerheitsproblem** ist entscheidbar
 - Die Sprache $L(G)$ ist genau dann leer, wenn S eine nutzlose Variable ist. (Siehe dazu Kapitel Vereinfachung der Grammatik von kontextfreien Sprachen)
- Das **Endlichkeitsproblem** ist entscheidbar
 - Transformation in Chomsky-Normalform
 - Aufstellen eines gerichteten Graphen.
 - Die Knoten sind die Nichtterminalen.
 - Die gerichtete Kantenmenge V wird gebildet anhand der Regeln. Jede Regel $S \rightarrow AB$ ergibt zwei Kanten $S \rightarrow A$ und $S \rightarrow B$.
 - Enthält der gerichtete Graph keine Zyklen, dann ist die Sprache endlich.
- Das **Äquivalenzproblem** ist nicht entscheidbar

Das Endlichkeitsproblem

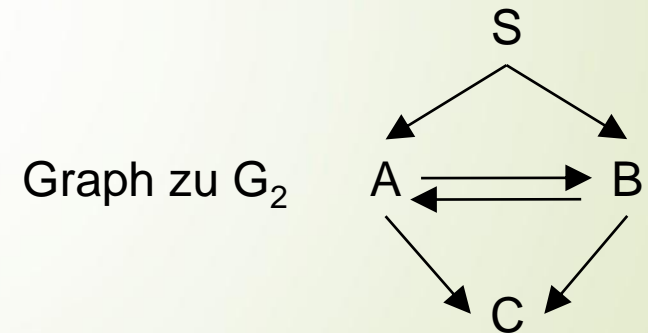
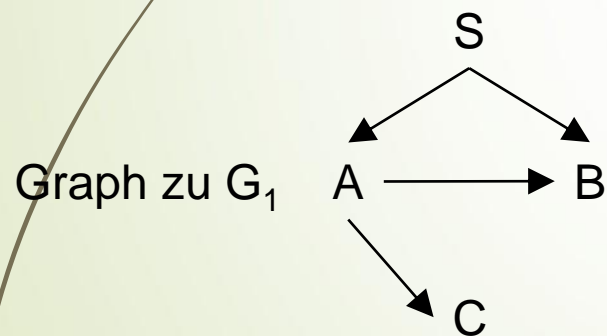
Beispiel

- Gegeben die Grammatiken:

$$G_1 = (\{S, A, B, C\}, \{a, b, c\}, \{S \rightarrow AB, A \rightarrow BC \mid a, B \rightarrow b, C \rightarrow c\}, S)$$

$$G_2 = (\{S, A, B, C\}, \{a, b, c\}, \{S \rightarrow AB, A \rightarrow BC, B \rightarrow CA \mid b, A \rightarrow a, C \rightarrow c\}, S)$$

- Die zugehörigen Graphen sind:



- G_1 nicht zyklisch $\Rightarrow L(G_1)$ endlich
- G_2 zyklisch $\Rightarrow L(G_2)$ unendlich

Das Wortproblem

CYK Algorithmus

$w \in L(G)$ und G eine kontextfreie Grammatik ?

- Diese Frage lässt sich mit ja beantworten.
- Ein einfacher Algorithmus mit Komplexität $O(n^3)$ stammt von Cocke, Younger und Kasami (CYK Algorithmus)
- Methode dynamische Programmierung
 - In der Chomsky-Normalform sind alle Regeln ($A, B, C \in N$ und $a \in \Sigma$) entweder
 - $A \rightarrow a$ oder
 - $A \rightarrow BC$
 - Sei $w \in \Sigma^*$ und es soll geprüft werden ob $w \in L(G)$ ist.
 - Ist $|w| = 1$ so muss es eine Regel $S \rightarrow w$ geben, andernfalls $w \notin L(G)$
 - Ist $|w| > 1$ so ist die erste Regel $S \Rightarrow XY \Rightarrow^* w$.
d.h w lässt sich zerlegen in $w = uv$ mit $X \Rightarrow^* u$ und $Y \Rightarrow^* v$
Dies wird nun rekursiv auf die Teilworte u und v von w angewandt bis $|u| = 1$ und $|v| = 1$
- Dieser rekursive Algorithmus lässt sich auch in einen iterativen Algorithmus überführen

Das Wortproblem

CYK Algorithmus: iterative Variante

- Methode der Tabellierung
 - Für jedes Teilwort des Eingabeworts w wird notiert aus welchem Nichtterminal es sich ableiten lässt. Dies wird sukzessiv angewandt bis man beim Startsymbol endet.
- Sei $w = a_1a_2\dots a_n$ das zu erkennende Wort.
- Man erstellt eine Tabelle $T \in (n+1) \times n$
 - in $T[i,j]$ wird notiert aus welchem Nichtterminal das Teilwort $u = a_ja_{j+1}\dots a_{j+1-i}$ das an Position j beginnt und Länge i hat ableiten lässt.
 - Das Ergebnis ist dann in $T[n,1]$ abzulesen.
 - Enthält $T[n,1]$ das Startsymbol S ist w aus dem S ableitbar und damit $w \in L(G)$

Das Wortproblem

CYK Algorithmus: Beispiel

- Beispiel $w_1=0011$ und $w_2=1001$ Frage $w_1 \in L(G)$, $w_2 \in L(G)$?
mit $G=(\{S,A,B,C\},\{0,1\}, S \rightarrow BC \mid BA, A \rightarrow SC \mid BS, B \rightarrow 0, C \rightarrow 1\}, S)$
- Aufstellen der Tabelle

w_1	j=1	j=2	j=3	j=4
i=0	0	0	1	1
i=1	B	B	C	C
i=2		S		
i=3	A	A		
i=4	S			

$\Rightarrow w_1=0011 \in L(G)$ und

$\Rightarrow w_2=1001 \notin L(G)$

w_2	j=1	j=2	j=3	j=4
i=0	1	0	0	1
i=1	C	B	B	C
i=2			S	
i=3		A		
i=4				

Aufgabe CYK Algorithmus

Lösung

- Überprüfen Sie mit Hilfe des CYK-Algorithmus, ob folgende Worte w zur Sprache $L(G)$ mit der Grammatik
- $G = (\{S, A, B, C, D, \{a,b,c,S\}, P, S\}$ mit
 $P = \{S \rightarrow AB \mid AC \mid \epsilon, C \rightarrow DB, D \rightarrow AB \mid AC, A \rightarrow a, B \rightarrow b \mid c\}$
gehören. (Überprüfen Sie ihr Ergebnis mit FLACI).
 - $w = aacbc$
 - $w = aaabcb$

Aufgabe CYK Algorithmus

➤ $G = (\{S, A, B, C, D, \{a,b,c,S\}, P, S\}$ mit

$P = \{S \rightarrow AB \mid AC \mid \varepsilon, C \rightarrow DB, D \rightarrow AB \mid AC, A \rightarrow a, B \rightarrow b \mid c\}$
gehören. (Überprüfen Sie ihr Ergebnis mit FLACI).

➤ $w = aacbc$

➤ gehört nicht dazu

w_1	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
$i=0$	a	a	c	b	c
$i=1$	A	A	B	B	B
$i=2$		S,D			
$i=3$		C			
$i=4$	S				
$j=5$					

Aufgabe CYK Algorithmus

➤ $G = (\{S, A, B, C, D, \{a,b,c,S\}, P, S\}$ mit

$P = \{S \rightarrow AB \mid AC \mid \varepsilon, C \rightarrow DB, D \rightarrow AB \mid AC, A \rightarrow a, B \rightarrow b \mid c\}$
gehören. (Überprüfen Sie ihr Ergebnis mit FLACI).

➤ $w = aaabcb$

➤ Gehört dazu

w_1	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$
$i=0$	a	a	a	b	c	b
$i=1$	A	A	A	B	B	B
$i=2$			S,D			
$i=3$			C			
$i=4$		S,D				
$j=5$		C				
$j=6$	S					