

## 8b

**Entscheiden des Wortproblems für Typ 1-Grammatiken**

Prof. Dr. Jasmin Blanchette

Lehr- und Forschungseinheit für  
Theoretische Informatik und Theorembeweisen

Stand: 15. April 2024

Basiert auf Folien von PD Dr. David Sabel



## Definition

Eine Sprache  $L$  ist **entscheidbar**, wenn es einen Algorithmus gibt, der bei Eingabe eines Wortes  $w$  in endlicher Zeit feststellt, ob  $w \in L$  gilt oder nicht.

# Wortproblem für Typ 1-Grammatiken

---

## Definition

Das **Wortproblem** für Typ  $i$ -Grammatiken ist die Frage, ob für eine gegebene Typ  $i$ -Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$   $w \in L(G)$  gilt oder nicht.

# Wortproblem für Typ 1-Grammatiken

## Definition

Das **Wortproblem** für Typ  $i$ -Grammatiken ist die Frage, ob für eine gegebene Typ  $i$ -Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$   $w \in L(G)$  gilt oder nicht.

## Satz

Das Wortproblem für Typ 1-Grammatiken ist entscheidbar:  
Es gibt einen Algorithmus, der bei Eingabe von Typ 1-Grammatik  $G$  und Wort  $w$  nach endlicher Zeit entscheidet, ob  $w \in L(G)$  gilt oder nicht.

# Wortproblem für Typ 1-Grammatiken

## Definition

Das **Wortproblem** für Typ  $i$ -Grammatiken ist die Frage, ob für eine gegebene Typ  $i$ -Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$   $w \in L(G)$  gilt oder nicht.

## Satz

Das Wortproblem für Typ 1-Grammatiken ist entscheidbar:  
Es gibt einen Algorithmus, der bei Eingabe von Typ 1-Grammatik  $G$  und Wort  $w$  nach endlicher Zeit entscheidet, ob  $w \in L(G)$  gilt oder nicht.

**Beweis** Algorithmus 2 auf späterer Folie bewerkstelligt dies. □

# Ein naiver Ansatz

---

Seien eine Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$ .

Schritte:

1. Beginne mit der Menge  $L_0 := \{S\}$ .
2. Wiederhole für  $i = 0, 1, 2, \dots$ :
  - 2.1 Wende die Produktionen von  $P$  auf die Satzformen in  $L_i$  an.  
Sei  $N$  das Ergebnis. Setze  $L_{i+1} := L_i \cup N$ .
  - 2.2 Falls  $w \in L_{i+1}$ , stoppe und gib ja aus.
  - 2.3 Falls  $L_{i+1} = L_i$ , stoppe und gib nein aus.

# Ein naiver Ansatz

Seien eine Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$ .

Schritte:

1. Beginne mit der Menge  $L_0 := \{S\}$ .
2. Wiederhole für  $i = 0, 1, 2, \dots$ :
  - 2.1 Wende die Produktionen von  $P$  auf die Satzformen in  $L_i$  an.  
Sei  $N$  das Ergebnis. Setze  $L_{i+1} := L_i \cup N$ .
  - 2.2 Falls  $w \in L_{i+1}$ , stoppe und gib **ja** aus.
  - 2.3 Falls  $L_{i+1} = L_i$ , stoppe und gib **nein** aus.

Problem: Der Ansatz **terminiert nicht zwingend**.

# Ein naiver Ansatz

Seien eine Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$ .

Schritte:

1. Beginne mit der Menge  $L_0 := \{S\}$ .
2. Wiederhole für  $i = 0, 1, 2, \dots$ :
  - 2.1 Wende die Produktionen von  $P$  auf die Satzformen in  $L_i$  an.  
Sei  $N$  das Ergebnis. Setze  $L_{i+1} := L_i \cup N$ .
  - 2.2 Falls  $w \in L_{i+1}$ , stoppe und gib **ja** aus.
  - 2.3 Falls  $L_{i+1} = L_i$ , stoppe und gib **nein** aus.

Problem: Der Ansatz **terminiert nicht zwingend**.

Bei Nichtterminierung ist  $w \notin L$  (aber das erfahren wir nie).



## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$

1.  $L_1 := L_0 \cup \{aSBc, abc\}$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aS Bc, abc\}$
2.  $L_2 := L_1 \cup \{aaS Bc Bc, aabc Bc\}$



## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$

## Beispiel für den naiven Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$

## Beispiel für den naiven Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBcBc, aabBcc\}$

## Beispiel für den naiven Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$

## Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$
4.  $L_4 := L_3 \cup \{aaaaSBcBcBcBc, aaaabcBcBcBc, \dots, aabbcc\}$

## Beispiel für den naiven Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$
4.  $L_4 := L_3 \cup \{aaaaSBcBcBcBc, aaaabcBcBcBc, \dots, aabbcc\}$

## Beispiel für den naiven Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$
4.  $L_4 := L_3 \cup \{aaaaSBcBcBcBc, aaaabcBcBcBc, \dots, aabbcc\}$



## Beispiel für den naiven Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$
4.  $L_4 := L_3 \cup \{aaaaSBcBcBcBc, aaaabcBcBcBc, \dots, aabbcc\}$

## Beispiel für den naiven Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$
4.  $L_4 := L_3 \cup \{aaaaSBcBcBcBc, aaaabcBcBcBc, \dots, aabbcc\}$

Aus  $aabbcc \in L_4$  folgt  $aabbcc \in L(G)$ .

## Weiteres Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort *cababa*

## Weiteres Beispiel für den naiven Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $cababa$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$
- $\vdots$

## Weiteres Beispiel für den naiven Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $cababa$

Schritte:

0.  $L_0 := \{S\}$
1.  $L_1 := L_0 \cup \{aSBc, abc\}$
2.  $L_2 := L_1 \cup \{aaSBcBc, aabcBc\}$
3.  $L_3 := L_2 \cup \{aaaSBcBcBc, aaabcBcBc, aaSBBcc, aabBcc\}$
- $\vdots$

Der Ansatz **terminiert nicht**.

D.h.  $cababa \notin L(G)$  (aber das erfahren wir nie).

# Ein besserer Ansatz

Seien eine Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$  der Länge  $n$ .  
Schritte:

1. Beginne mit der Menge  $L_0^n := \{S\}$ .  
(Hier ist  $n$  ein Index und keine Potenz.)
2. Wiederhole für  $i = 0, 1, 2, \dots$ :
  - 2.1 Wende die Produktionen von  $P$  auf die Satzformen in  $L_i^n$  an.  
Sei  $N$  das Ergebnis. Setze  $L_{i+1}^n := L_i^n \cup N'$ ,  
**wobei  $N'$  aus den Satzformen der Länge  $\leq n$  aus  $N$  besteht.**
  - 2.2 Falls  $w \in L_{i+1}^n$ , stoppe und gib **ja** aus.
  - 2.3 Falls  $L_{i+1}^n = L_i^n$ , stoppe und gib **nein** aus.

## Ein besserer Ansatz

Seien eine Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$  der Länge  $n$ .  
Schritte:

1. Beginne mit der Menge  $L_0^n := \{S\}$ .  
(Hier ist  $n$  ein Index und keine Potenz.)
2. Wiederhole für  $i = 0, 1, 2, \dots$ :
  - 2.1 Wende die Produktionen von  $P$  auf die Satzformen in  $L_i^n$  an.  
Sei  $N$  das Ergebnis. Setze  $L_{i+1}^n := L_i^n \cup N'$ ,  
**wobei  $N'$  aus den Satzformen der Länge  $\leq n$  aus  $N$  besteht.**
  - 2.2 Falls  $w \in L_{i+1}^n$ , stoppe und gib **ja** aus.
  - 2.3 Falls  $L_{i+1}^n = L_i^n$ , stoppe und gib **nein** aus.

Der Ansatz **terminiert immer** (Beweis später),  
hat aber **exponentielle Laufzeitkomplexität**.

# Ein besserer Ansatz

Seien eine Grammatik  $G = (V, \Sigma, P, S)$  und ein Wort  $w \in \Sigma^*$  der Länge  $n$ .  
Schritte:

1. Beginne mit der Menge  $L_0^n := \{S\}$ .  
(Hier ist  $n$  ein Index und keine Potenz.)
2. Wiederhole für  $i = 0, 1, 2, \dots$ :
  - 2.1 Wende die Produktionen von  $P$  auf die Satzformen in  $L_i^n$  an.  
Sei  $N$  das Ergebnis. Setze  $L_{i+1}^n := L_i^n \cup N'$ ,  
**wobei  $N'$  aus den Satzformen der Länge  $\leq n$  aus  $N$  besteht.**
  - 2.2 Falls  $w \in L_{i+1}^n$ , stoppe und gib **ja** aus.
  - 2.3 Falls  $L_{i+1}^n = L_i^n$ , stoppe und gib **nein** aus.

Der Ansatz **terminiert immer** (Beweis später),  
hat aber **exponentielle Laufzeitkomplexität**.

Er ist auch **inkorrekt für Typ 0-Grammatiken**.

Deshalb **fokussieren wir uns auf Typ 1-Grammatiken** (mit 1. Sonderregel).



## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

$$0. L_0^6 := \{S\}$$

## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$

1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$

## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

$$0. L_0^6 := \{S\}$$

$$1. L_1^6 := L_0^6 \cup \{aSBc, abc\}$$

## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

$$0. L_0^6 := \{S\}$$

$$1. L_1^6 := L_0^6 \cup \{aSBc, abc\}$$

## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$

## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aS Bc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$

## Beispiel für den besseren Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$   ~~$aSBc \Rightarrow aaSBcBc$~~



## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$
3.  $L_3^6 := L_2^6 \cup \{aabBcc\}$

## Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aab**c**Bc\}$
3.  $L_3^6 := L_2^6 \cup \{aab**B**cc\}$

## Beispiel für den besseren Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$
3.  $L_3^6 := L_2^6 \cup \{aabBcc\}$
4.  $L_4^6 := L_3^6 \cup \{aabbcc\}$

## Beispiel für den besseren Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$
3.  $L_3^6 := L_2^6 \cup \{aabBcc\}$
4.  $L_4^6 := L_3^6 \cup \{aabbcc\}$

## Beispiel für den besseren Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$
3.  $L_3^6 := L_2^6 \cup \{aabBcc\}$
4.  $L_4^6 := L_3^6 \cup \{aabbcc\}$
5.  $L_5^6 := L_4^6$

## Beispiel für den besseren Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $aabbcc$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$
3.  $L_3^6 := L_2^6 \cup \{aabBcc\}$
4.  $L_4^6 := L_3^6 \cup \{aabbcc\}$
5.  $L_5^6 := L_4^6$

Aus  $aabbcc \in L_4^6$  folgt  $aabbcc \in L(G)$ .

## Weiteres Beispiel für den besseren Ansatz

---

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $cababa$

## Weiteres Beispiel für den besseren Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $cababa$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$
3.  $L_3^6 := L_2^6 \cup \{aabBcc\}$
4.  $L_4^6 := L_3^6 \cup \{aabbcc\}$
5.  $L_5^6 := L_4^6$



## Weiteres Beispiel für den besseren Ansatz

Grammatik  $G = (\{S, B\}, \{a, b, c\}, P, S)$  mit  
 $P = \{S \rightarrow aSBc, S \rightarrow abc, cB \rightarrow Bc, bB \rightarrow bb\}$  und  
Wort  $cababa$

Schritte:

0.  $L_0^6 := \{S\}$
1.  $L_1^6 := L_0^6 \cup \{aSBc, abc\}$
2.  $L_2^6 := L_1^6 \cup \{aabcBc\}$
3.  $L_3^6 := L_2^6 \cup \{aabBcc\}$
4.  $L_4^6 := L_3^6 \cup \{aabbcc\}$
5.  $L_5^6 := L_4^6$

Aus  $cababa \notin L_4^6$  folgt  $cababa \notin L(G)$ .

# Gegenbeispiel für eine Typ 0-Grammatik

---

Grammatik  $G = (\{S\}, \{a, b\}, P, S)$  mit  
 $P = \{S \rightarrow aaaa, aaaa \rightarrow bb\}$  und  
Wort  $bb$

Schritte:

# Gegenbeispiel für eine Typ 0-Grammatik

---

Grammatik  $G = (\{S\}, \{a, b\}, P, S)$  mit  
 $P = \{S \rightarrow aaaa, aaaa \rightarrow bb\}$  und  
Wort  $bb$

Schritte:

0.  $L_0^2 := \{S\}$

# Gegenbeispiel für eine Typ 0-Grammatik

Grammatik  $G = (\{S\}, \{a, b\}, P, S)$  mit  
 $P = \{S \rightarrow aaaa, aaaa \rightarrow bb\}$  und  
Wort  $bb$

Schritte:

0.  $L_0^2 := \{S\}$
1.  $L_1^2 := L_0^2$

# Gegenbeispiel für eine Typ 0-Grammatik

Grammatik  $G = (\{S\}, \{a, b\}, P, S)$  mit  
 $P = \{S \rightarrow aaaa, aaaa \rightarrow bb\}$  und  
Wort  $bb$

Schritte:

0.  $L_0^2 := \{S\}$

1.  $L_1^2 := L_0^2$

$bb \notin L_0^2$  und jedoch  $bb \in L(G)$ .

# Formale Definition der Menge $L_i^n$

## Definition

Sei  $G = (V, \Sigma, P, S)$  eine Typ 1-Grammatik. Für  $i, n \in \mathbb{N}$  sei

$$L_i^n := \{w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ und } S \Rightarrow_G^k w \text{ mit } k \leq i\}$$

# Formale Definition der Menge $L_i^n$

## Definition

Sei  $G = (V, \Sigma, P, S)$  eine Typ 1-Grammatik. Für  $i, n \in \mathbb{N}$  sei

$$L_i^n := \{w \in (V \cup \Sigma)^* \mid |w| \leq n \text{ und } S \Rightarrow_G^k w \text{ mit } k \leq i\}$$

Informell:

$L_i^n$  = Menge aller Satzformen der Länge höchstens  $n$ ,  
die in höchstens  $i$  Schritten von  $S$  aus ableitbar sind

# Formale Definition der Prozedur

Seien eine Typ 1-Grammatik  $G = (V, \Sigma, P, S)$  (mit 1. Sonderregel) und ein Wort  $w \in \Sigma^*$ .

Sei

$$\text{next}(L, n) := L \cup \{v \mid u \in L, u \Rightarrow_G v \text{ und } |v| \leq n\}$$

Schritte:

1. Beginne mit  $L_0^n := \{S\}$  wenn  $n > 0$ , sonst  $L_0^n := \emptyset$ .
2. Wiederhole für  $i = 0, 1, 2, \dots$ :
  - 2.1 Setze  $L_{i+1}^n := \text{next}(L_i^n, n)$ .
  - 2.2 Falls  $w \in L_{i+1}^n$ , stoppe und gib **ja** aus.
  - 2.3 Falls  $L_{i+1}^n = L_i^n$ , stoppe und gib **nein** aus.



# Terminierung der Prozedur

---

## **Satz**

Die Prozedur terminiert.

# Terminierung der Prozedur

---

## **Satz**

Die Prozedur terminiert.

**Beweis** Durch Widerspruch:

# Terminierung der Prozedur

---

## **Satz**

Die Prozedur terminiert.

**Beweis** Durch Widerspruch:

Wir nehmen an, dass die Berechnung nicht stoppt.

## Satz

Die Prozedur terminiert.

**Beweis** Durch Widerspruch:

Wir nehmen an, dass die Berechnung nicht stoppt.

Da  $L_i^n \subseteq L_{i+1}^n$  und  $L_i^n \neq L_{i+1}^n$ , muss für alle  $i \in \mathbb{N}$  gelten:  $L_i^n \subset L_{i+1}^n$ .

# Terminierung der Prozedur

## Satz

Die Prozedur terminiert.

**Beweis** Durch Widerspruch:

Wir nehmen an, dass die Berechnung nicht stoppt.

Da  $L_i^n \subseteq L_{i+1}^n$  und  $L_i^n \neq L_{i+1}^n$ , muss für alle  $i \in \mathbb{N}$  gelten:  $L_i^n \subset L_{i+1}^n$ .

Daher gilt  $|L_0^n| < |L_1^n| < |L_2^n| < \dots$ . Die Mengen werden also beliebig groß.

# Terminierung der Prozedur

## Satz

Die Prozedur terminiert.

**Beweis** Durch Widerspruch:

Wir nehmen an, dass die Berechnung nicht stoppt.

Da  $L_i^n \subseteq L_{i+1}^n$  und  $L_i^n \neq L_{i+1}^n$ , muss für alle  $i \in \mathbb{N}$  gelten:  $L_i^n \subset L_{i+1}^n$ .

Daher gilt  $|L_0^n| < |L_1^n| < |L_2^n| < \dots$ . Die Mengen werden also beliebig groß.

Gleichzeitig gibt es eine obere Schranke für die Mächtigkeit der Mengen:

$|L_i^n| \leq (|V \cup \Sigma| + 1)^n$  für alle  $i \in \mathbb{N}$ . Widerspruch. □

## Satz

Die Prozedur ist korrekt: Sei  $w$  ein Wort der Länge  $\leq n$ .  
Wenn  $L_{i+1}^n = L_i^n$ , dann ist  $w \in L_i^n$  g.d.w.  $w \in L(G)$ .

## Satz

Die Prozedur ist korrekt: Sei  $w$  ein Wort der Länge  $\leq n$ .  
Wenn  $L_{i+1}^n = L_i^n$ , dann ist  $w \in L_i^n$  g.d.w.  $w \in L(G)$ .

**Beweis** Zur Erinnerung:

$L_i^n$  = Menge aller Satzformen der Länge höchstens  $n$ ,  
die in höchstens  $i$  Schritten von  $S$  aus ableitbar sind



## Satz

Die Prozedur ist korrekt: Sei  $w$  ein Wort der Länge  $\leq n$ .  
Wenn  $L_{i+1}^n = L_i^n$ , dann ist  $w \in L_i^n$  g.d.w.  $w \in L(G)$ .

**Beweis** Zur Erinnerung:

$L_i^n$  = Menge aller Satzformen der Länge höchstens  $n$ ,  
die in höchstens  $i$  Schritten von  $S$  aus ableitbar sind

Wenn  $L_{i+1}^n = L_i^n$ , dann gilt  $L_{i+k}^n = L_i^n$  für alle  $k \in \mathbb{N}$  und daher enthält  $L_i^n$  alle aus  $S$  ableitbaren Wörter der Länge höchstens  $n$  und keine anderen.  $\square$

## Algorithmus 2: Entscheiden des Wortproblems für Typ 1-Grammatiken

**Eingabe:** Typ 1-Grammatik  $G = (V, \Sigma, P, S)$  (mit 1. Sonderregel) und  $w \in \Sigma^*$

**Ausgabe:** *Ja*, wenn  $w \in L(G)$  und *Nein*, wenn  $w \notin L(G)$

**Beginn**

$n := |w|;$

$L := \{S\};$

**wiederhole**

$L_{old} := L;$

$L := next(L_{old}, n);$

**bis**  $w \in L$  oder  $L_{old} = L;$

**wenn**  $w \in L$  **dann**

**return** *Ja*;

**sonst**

**return** *Nein*;

**Ende**

**Ende**

# Wortproblem für Typ 2- (und 3-)Grammatiken

---

## **Korollar**

Das Wortproblem für Typ 2- (und 3-)Grammatiken ist entscheidbar.

# Wortproblem für Typ 2- (und 3-)Grammatiken

---

## Korollar

Das Wortproblem für Typ 2- (und 3-)Grammatiken ist entscheidbar.

Bemerkungen:

- ▶ Der Algorithmus für Typ 1-Grammatiken hat **exponentielle Laufzeitkomplexität**.

# Wortproblem für Typ 2- (und 3-)Grammatiken

## Korollar

Das Wortproblem für Typ 2- (und 3-)Grammatiken ist entscheidbar.

Bemerkungen:

- ▶ Der Algorithmus für Typ 1-Grammatiken hat **exponentielle Laufzeitkomplexität**.
- ▶ Das Wortproblem für Typ 2- (und 3-)Grammatiken ist in **polynomieller Zeit** lösbar (durch den CYK-Algorithmus).

# Wortproblem für Typ 2- (und 3-)Grammatiken

## Korollar

Das Wortproblem für Typ 2- (und 3-)Grammatiken ist entscheidbar.

Bemerkungen:

- ▶ Der Algorithmus für Typ 1-Grammatiken hat **exponentielle Laufzeitkomplexität**.
- ▶ Das Wortproblem für Typ 2- (und 3-)Grammatiken ist in **polynomieller Zeit** lösbar (durch den CYK-Algorithmus).
- ▶ Das Wortproblem für Typ 0-Grammatiken ist **unentscheidbar**.