

Vorlesung

Grundlagen der Theoretischen Informatik / Einführung in die Theoretische Informatik I

Bernhard Beckert

Institut für Informatik



Sommersemester 2007

Dank

Diese Vorlesungsmaterialien basieren ganz wesentlich auf den Folien zu den Vorlesungen von

Katrin Erk (gehalten an der Universität Koblenz-Landau)

Jürgen Dix (gehalten an der TU Clausthal)

Ihnen beiden gilt mein herzlicher Dank.

– *Bernhard Beckert, April 2007*

Elimination von ε -Regeln

Korollar

$$L_2 \subseteq L_1$$

Das heißt, jede kontextfreie Sprache ist auch kontextsensitiv

Beweis

Regeln einer kontextsensitiven Grammatik müssen folgende Form haben:

- entweder $uAv \rightarrow u\alpha v$
mit $u, v, \alpha \in (V \cup T)^*$, $|\alpha| \geq 1$, $A \in V$
- oder $S \rightarrow \varepsilon$
und S kommt in keiner Regelconclusio vor.

Diesen Bedingungen genügt die kontextfreie Grammatik nach Elimination der ε -Regeln.

Elimination von ε -Regeln

Korollar

$$L_2 \subseteq L_1$$

Das heißt, jede kontextfreie Sprache ist auch kontextsensitiv

Beweis

Regeln einer kontextsensitiven Grammatik müssen folgende Form haben:

- entweder $uAv \rightarrow u\alpha v$
mit $u, v, \alpha \in (V \cup T)^*$, $|\alpha| \geq 1$, $A \in V$
- oder $S \rightarrow \varepsilon$
und S kommt in keiner Regelconclusio vor.

Diesen Bedingungen genügt die kontextfreie Grammatik nach Elimination der ε -Regeln.

Elimination von ε -Regeln

Korollar

$$L_2 \subseteq L_1$$

Das heißt, jede kontextfreie Sprache ist auch kontextsensitiv

Beweis

Regeln einer kontextsensitiven Grammatik müssen folgende Form haben:

- entweder $uAv \rightarrow u\alpha v$
mit $u, v, \alpha \in (V \cup T)^*$, $|\alpha| \geq 1$, $A \in V$
- oder $S \rightarrow \varepsilon$
und S kommt in keiner Regelconclusio vor.

Diesen Bedingungen genügt die kontextfreie Grammatik nach Elimination der ε -Regeln.

Definition 19.7 (Kettenproduktion)

Eine Regel der Form

$$A \rightarrow B \quad \text{mit } A, B \in V$$

heißt **Kettenproduktion**.

Theorem 19.8 (Kettenproduktionen sind eliminierbar)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik ohne Kettenproduktionen.

Definition 19.7 (Kettenproduktion)

Eine Regel der Form

$$A \rightarrow B \quad \text{mit } A, B \in V$$

heißt **Kettenproduktion**.

Theorem 19.8 (Kettenproduktionen sind eliminierbar)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik ohne Kettenproduktionen.

Elimination von Kettenproduktionen

Beweis

Sei $G = (V, T, R, S)$ eine kontextfreie Grammatik ohne ε -Regeln, außer ggf. $S \rightarrow \varepsilon$.

Konstruiere neue Grammatik wie folgt:

1 Für alle

- Variablenpaare $A, B \in V$, $A \neq B$ mit $A \implies^* B$
- Regeln $B \rightarrow \alpha \in R$, $\alpha \notin V$

füge zu R hinzu:

$$A \rightarrow \alpha$$

2 Lösche alle Kettenproduktionen

Elimination von Kettenproduktionen

Beweis

Sei $G = (V, T, R, S)$ eine kontextfreie Grammatik ohne ε -Regeln, außer ggf. $S \rightarrow \varepsilon$.

Konstruiere neue Grammatik wie folgt:

1 Für alle

- Variablenpaare $A, B \in V$, $A \neq B$ mit $A \Longrightarrow^* B$
- Regeln $B \rightarrow \alpha \in R$, $\alpha \notin V$

füge zu R hinzu:

$$A \rightarrow \alpha$$

2 Lösche alle Kettenproduktionen

Elimination von Kettenproduktionen

Beweis

Sei $G = (V, T, R, S)$ eine kontextfreie Grammatik ohne ε -Regeln, außer ggf. $S \rightarrow \varepsilon$.

Konstruiere neue Grammatik wie folgt:

1 Für alle

- Variablenpaare $A, B \in V$, $A \neq B$ mit $A \Longrightarrow^* B$
- Regeln $B \rightarrow \alpha \in R$, $\alpha \notin V$

füge zu R hinzu:

$$A \rightarrow \alpha$$

2 Lösche alle Kettenproduktionen

Elimination von Kettenproduktionen

Beweis

Sei $G = (V, T, R, S)$ eine kontextfreie Grammatik ohne ε -Regeln, außer ggf. $S \rightarrow \varepsilon$.

Konstruiere neue Grammatik wie folgt:

1 Für alle

- Variablenpaare $A, B \in V$, $A \neq B$ mit $A \Longrightarrow^* B$
- Regeln $B \rightarrow \alpha \in R$, $\alpha \notin V$

füge zu R hinzu:

$$A \rightarrow \alpha$$

2 Lösche alle Kettenproduktionen

Theorem 19.9 (Normalform für cf-Grammatiken)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik

- *ohne ε -Regeln*
(bis auf $S \rightarrow \varepsilon$, falls ε zur Sprache gehört;
in diesem Fall darf S in keiner Regelconclusio vorkommen),
- *ohne nutzlose Symbole,*
- *ohne Kettenproduktionen,*
- *so daß für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.*

Theorem 19.9 (Normalform für cf-Grammatiken)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik

- *ohne ε -Regeln*
(bis auf $S \rightarrow \varepsilon$, falls ε zur Sprache gehört;
in diesem Fall darf S in keiner Regelconclusio vorkommen),
- *ohne nutzlose Symbole,*
- *ohne Kettenproduktionen,*
- *so daß für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.*

Theorem 19.9 (Normalform für cf-Grammatiken)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik

- *ohne ε -Regeln*
(bis auf $S \rightarrow \varepsilon$, falls ε zur Sprache gehört;
in diesem Fall darf S in keiner Regelconclusio vorkommen),
- *ohne nutzlose Symbole,*
- *ohne Kettenproduktionen,*
- *so daß für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.*

Theorem 19.9 (Normalform für cf-Grammatiken)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik

- *ohne ε -Regeln*
(bis auf $S \rightarrow \varepsilon$, falls ε zur Sprache gehört;
in diesem Fall darf S in keiner Regelconclusio vorkommen),
- *ohne nutzlose Symbole,*
- *ohne Kettenproduktionen,*
- *so daß für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.*

Beweis

- 1 Man teste zunächst, ob S nullbar ist. Falls ja, dann verwende man S_{neu} als neues Startsymbol und füge die Regeln $S_{neu} \rightarrow S \mid \varepsilon$ zum Regelsatz hinzu.
- 2 Man eliminiere nutzlose Symbole.
- 3 Man eliminiere alle ε -Regeln außer $S_{neu} \rightarrow \varepsilon$.
- 4 Man bringe die Grammatik in die Normalform, bei der für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.
- 5 Man eliminiere Kettenproduktionen.
- 6 Zum Schluss eliminiere man noch einmal alle nutzlosen Symbole (wg. Schritt 3)

Beweis

- 1 Man teste zunächst, ob S nullbar ist. Falls ja, dann verwende man S_{neu} als neues Startsymbol und füge die Regeln $S_{neu} \rightarrow S \mid \varepsilon$ zum Regelsatz hinzu.
- 2 **Man eliminiere nutzlose Symbole.**
- 3 Man eliminiere alle ε -Regeln außer $S_{neu} \rightarrow \varepsilon$.
- 4 Man bringe die Grammatik in die Normalform, bei der für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.
- 5 Man eliminiere Kettenproduktionen.
- 6 Zum Schluss eliminiere man noch einmal alle nutzlosen Symbole (wg. Schritt 3)

Beweis

- 1 Man teste zunächst, ob S nullbar ist. Falls ja, dann verwende man S_{neu} als neues Startsymbol und füge die Regeln $S_{neu} \rightarrow S \mid \varepsilon$ zum Regelsatz hinzu.
- 2 Man eliminiere nutzlose Symbole.
- 3 **Man eliminiere alle ε -Regeln außer $S_{neu} \rightarrow \varepsilon$.**
- 4 Man bringe die Grammatik in die Normalform, bei der für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.
- 5 Man eliminiere Kettenproduktionen.
- 6 Zum Schluss eliminiere man noch einmal alle nutzlosen Symbole (wg. Schritt 3)

Normalform für cf-Grammatiken

Beweis

- 1 Man teste zunächst, ob S nullbar ist. Falls ja, dann verwende man S_{neu} als neues Startsymbol und füge die Regeln $S_{neu} \rightarrow S \mid \varepsilon$ zum Regelsatz hinzu.
- 2 Man eliminiere nutzlose Symbole.
- 3 Man eliminiere alle ε -Regeln außer $S_{neu} \rightarrow \varepsilon$.
- 4 **Man bringe die Grammatik in die Normalform, bei der für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.**
- 5 Man eliminiere Kettenproduktionen.
- 6 Zum Schluss eliminiere man noch einmal alle nutzlosen Symbole (wg. Schritt 3)

Normalform für cf-Grammatiken

Beweis

- 1 Man teste zunächst, ob S nullbar ist. Falls ja, dann verwende man S_{neu} als neues Startsymbol und füge die Regeln $S_{neu} \rightarrow S \mid \varepsilon$ zum Regelsatz hinzu.
- 2 Man eliminiere nutzlose Symbole.
- 3 Man eliminiere alle ε -Regeln außer $S_{neu} \rightarrow \varepsilon$.
- 4 Man bringe die Grammatik in die Normalform, bei der für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.
- 5 **Man eliminiere Kettenproduktionen.**
- 6 Zum Schluss eliminiere man noch einmal alle nutzlosen Symbole (wg. Schritt 3)

Normalform für cf-Grammatiken

Beweis

- 1 Man teste zunächst, ob S nullbar ist. Falls ja, dann verwende man S_{neu} als neues Startsymbol und füge die Regeln $S_{neu} \rightarrow S \mid \varepsilon$ zum Regelsatz hinzu.
- 2 Man eliminiere nutzlose Symbole.
- 3 Man eliminiere alle ε -Regeln außer $S_{neu} \rightarrow \varepsilon$.
- 4 Man bringe die Grammatik in die Normalform, bei der für jede Regel $P \rightarrow Q$ gilt: entweder $Q \in V^*$ oder $Q \in T$.
- 5 Man eliminiere Kettenproduktionen.
- 6 **Zum Schluss eliminiere man noch einmal alle nutzlosen Symbole (wg. Schritt 3)**

Teil IV

- 1 Ableitungsbäume
- 2 Umformung von Grammatiken**
- 3 Normalformen
- 4 Pumping-Lemma für kontextfreie Sprachen
- 5 Pushdown-Automaten (PDAs)
- 6 Determinierte PDAs
- 7 Abschlusseigenschaften
- 8 Wortprobleme
- 9 Der CYK-Algorithmus

Teil IV

- 1 Ableitungsbäume
- 2 Umformung von Grammatiken
- 3 Normalformen**
- 4 Pumping-Lemma für kontextfreie Sprachen
- 5 Pushdown-Automaten (PDAs)
- 6 Determinierte PDAs
- 7 Abschlusseigenschaften
- 8 Wortprobleme
- 9 Der CYK-Algorithmus

Unterschied: Grammatiktypen und Normalformen

Gemeinsamkeit: Sowohl Grammatiktypen als auch Normalformen schränken die Form von Grammatikregeln ein.

Unterschied:

- Grammatiktypen (rechtslinear, kontextfrei usw.) führen zu **unterschiedlichen Sprachklassen**
- Normalformeln führen zu **den selben Sprachklassen**

Unterschied: Grammatiktypen und Normalformen

Gemeinsamkeit: Sowohl Grammatiktypen als auch Normalformen schränken die Form von Grammatikregeln ein.

Unterschied:

- Grammatiktypen (rechtslinear, kontextfrei usw.)
führen zu **unterschiedlichen Sprachklassen**
- Normalformeln führen zu
den selben Sprachklassen

Unterschied: Grammatiktypen und Normalformen

Gemeinsamkeit: Sowohl Grammatiktypen als auch Normalformen schränken die Form von Grammatikregeln ein.

Unterschied:

- Grammatiktypen (rechtslinear, kontextfrei usw.) führen zu **unterschiedlichen Sprachklassen**
- **Normalformeln** führen zu **den selben Sprachklassen**

Wozu dann Normalformen?

- Weniger Fallunterscheidungen bei Algorithmen, die mit Grammatiken arbeiten.
- Struktur von Grammatiken einfacher zu „durchschauen“

Zwei Normalformen

Chomsky-Normalform: Baut auf den Umformungen des vorigen Teils auf.

Greibach-Normalform: Ähnlich den rechtslinearen Grammatiken.

Wozu dann Normalformen?

- Weniger Fallunterscheidungen bei Algorithmen, die mit Grammatiken arbeiten.
- **Struktur von Grammatiken einfacher zu „durchschauen“**

Zwei Normalformen

Chomsky-Normalform: Baut auf den Umformungen des vorigen Teils auf.

Greibach-Normalform: Ähnlich den rechtslinearen Grammatiken.

Wozu dann Normalformen?

- Weniger Fallunterscheidungen bei Algorithmen, die mit Grammatiken arbeiten.
- Struktur von Grammatiken einfacher zu „durchschauen“

Zwei Normalformen

Chomsky-Normalform: Baut auf den Umformungen des vorigen Teils auf.

Greibach-Normalform: Ähnlich den rechtslinearen Grammatiken.

Wozu dann Normalformen?

- Weniger Fallunterscheidungen bei Algorithmen, die mit Grammatiken arbeiten.
- Struktur von Grammatiken einfacher zu „durchschauen“

Zwei Normalformen

Chomsky-Normalform: Baut auf den Umformungen des vorigen Teils auf.

Greibach-Normalform: Ähnlich den rechtslinearen Grammatiken.

Definition 20.1 (Chomsky-Normalform)

Eine cf-Grammatik $G = (V, T, R, S)$ ist in **Chomsky-Normalform (CNF)**, wenn gilt:

- G hat nur Regeln der Form

$$A \rightarrow BC \quad \text{mit } A, B, C \in V \text{ und}$$

$$A \rightarrow a \quad \text{mit } A \in V, a \in T \quad (\text{nicht } \varepsilon!)$$

- Ist $\varepsilon \in L(G)$, so darf G zusätzlich die Regel $S \rightarrow \varepsilon$ enthalten. In diesem Fall darf S in keiner Regelconclusio vorkommen.
- G enthält keine nutzlosen Symbole.

Definition 20.1 (Chomsky-Normalform)

Eine cf-Grammatik $G = (V, T, R, S)$ ist in **Chomsky-Normalform (CNF)**, wenn gilt:

- G hat nur Regeln der Form

$$A \rightarrow BC \quad \text{mit } A, B, C \in V \text{ und}$$

$$A \rightarrow a \quad \text{mit } A \in V, a \in T \quad (\text{nicht } \varepsilon!)$$

- Ist $\varepsilon \in L(G)$, so darf G zusätzlich die Regel $S \rightarrow \varepsilon$ enthalten.
In diesem Fall darf S in keiner Regelconclusio vorkommen.
- G enthält keine nutzlosen Symbole.

Definition 20.1 (Chomsky-Normalform)

Eine cf-Grammatik $G = (V, T, R, S)$ ist in **Chomsky-Normalform (CNF)**, wenn gilt:

- G hat nur Regeln der Form

$$A \rightarrow BC \quad \text{mit } A, B, C \in V \text{ und}$$

$$A \rightarrow a \quad \text{mit } A \in V, a \in T \quad (\text{nicht } \varepsilon!)$$

- Ist $\varepsilon \in L(G)$, so darf G zusätzlich die Regel $S \rightarrow \varepsilon$ enthalten. In diesem Fall darf S in keiner Regelconclusio vorkommen.
- G enthält keine nutzlosen Symbole.

Theorem 20.2 (Chomsky-Normalform)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik in Chomsky-Normalform.

Beweis

Schritt 1: Wende auf G die Umformungen des letzten Abschnitts an.

Ergebnis:

- G hat keine nutzlosen Symbole
- Alle Regeln haben die Form
 - 1 $A \rightarrow \alpha$ mit $A \in V$ und $\alpha \in V^*$, $|\alpha| \geq 2$, und
 - 2 $A \rightarrow a$ mit $A \in V$, $a \in T$

Theorem 20.2 (Chomsky-Normalform)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik in Chomsky-Normalform.

Beweis

Schritt 1: Wende auf G die Umformungen des letzten Abschnitts an.

Ergebnis:

- G hat keine nutzlosen Symbole
- Alle Regeln haben die Form
 - 1 $A \rightarrow \alpha$ mit $A \in V$ und $\alpha \in V^*$, $|\alpha| \geq 2$, und
 - 2 $A \rightarrow a$ mit $A \in V$, $a \in T$

Theorem 20.2 (Chomsky-Normalform)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik in Chomsky-Normalform.

Beweis

Schritt 1: Wende auf G die Umformungen des letzten Abschnitts an.

Ergebnis:

- G hat keine nutzlosen Symbole
- Alle Regeln haben die Form
 - 1 $A \rightarrow \alpha$ mit $A \in V$ und $\alpha \in V^*$, $|\alpha| \geq 2$, und
 - 2 $A \rightarrow a$ mit $A \in V$, $a \in T$

Theorem 20.2 (Chomsky-Normalform)

Zu jeder cf-Grammatik existiert eine äquivalente cf-Grammatik in Chomsky-Normalform.

Beweis

Schritt 1: Wende auf G die Umformungen des letzten Abschnitts an.

Ergebnis:

- G hat keine nutzlosen Symbole
- Alle Regeln haben die Form
 - 1 $A \rightarrow \alpha$ mit $A \in V$ und $\alpha \in V^*$, $|\alpha| \geq 2$, und
 - 2 $A \rightarrow a$ mit $A \in V$, $a \in T$

Beweis (Forts.)

Schritt 2: Regeln so umformen, daß keine Conclusio eine Länge größer 2 hat.

Ersetze jede Regel

$$A \rightarrow A_1 \dots A_n \text{ mit } A, A_i \in V, n \geq 3$$

durch:

$$A \rightarrow A_1 C_1$$

$$C_1 \rightarrow A_2 C_2$$

$$\vdots$$

$$C_{n-2} \rightarrow A_{n-1} A_n$$

Dabei sind die C_j neue Variablen in V .



Beweis (Forts.)

Schritt 2: Regeln so umformen, daß keine Conclusio eine Länge größer 2 hat.

Ersetze jede Regel

$$A \rightarrow A_1 \dots A_n \text{ mit } A, A_i \in V, n \geq 3$$

durch:

$$A \rightarrow A_1 C_1$$

$$C_1 \rightarrow A_2 C_2$$

$$\vdots$$

$$C_{n-2} \rightarrow A_{n-1} A_n$$

Dabei sind die C_i neue Variablen in V .



Definition 20.3 (Greibach-Normalform)

Eine cf-Grammatik $G = (V, T, R, S)$ ist in **Greibach-Normalform (GNF)**, wenn gilt:

- G hat nur Regeln der Form

$$A \rightarrow a\alpha \text{ mit } A \in V \text{ und } a \in T \text{ und } \alpha \in V^*$$

- Ist $\varepsilon \in L(G)$, so darf G zusätzlich die Regel $S \rightarrow \varepsilon$ enthalten. In diesem Fall darf S in keiner Regelconclusio vorkommen.
- G enthält keine nutzlosen Symbole.

Definition 20.3 (Greibach-Normalform)

Eine cf-Grammatik $G = (V, T, R, S)$ ist in **Greibach-Normalform (GNF)**, wenn gilt:

- G hat nur Regeln der Form

$$A \rightarrow a\alpha \text{ mit } A \in V \text{ und } a \in T \text{ und } \alpha \in V^*$$

- Ist $\varepsilon \in L(G)$, so darf G zusätzlich die Regel $S \rightarrow \varepsilon$ enthalten. In diesem Fall darf S in keiner Regelconclusio vorkommen.
- G enthält keine nutzlosen Symbole.

Definition 20.3 (Greibach-Normalform)

Eine cf-Grammatik $G = (V, T, R, S)$ ist in **Greibach-Normalform (GNF)**, wenn gilt:

- G hat nur Regeln der Form

$$A \rightarrow a\alpha \text{ mit } A \in V \text{ und } a \in T \text{ und } \alpha \in V^*$$

- Ist $\varepsilon \in L(G)$, so darf G zusätzlich die Regel $S \rightarrow \varepsilon$ enthalten.
In diesem Fall darf S in keiner Regelconclusio vorkommen.
- G enthält keine nutzlosen Symbole.

Definition 20.3 (Greibach-Normalform)

Eine cf-Grammatik $G = (V, T, R, S)$ ist in **Greibach-Normalform (GNF)**, wenn gilt:

- G hat nur Regeln der Form

$$A \rightarrow a\alpha \text{ mit } A \in V \text{ und } a \in T \text{ und } \alpha \in V^*$$

- Ist $\varepsilon \in L(G)$, so darf G zusätzlich die Regel $S \rightarrow \varepsilon$ enthalten. In diesem Fall darf S in keiner Regelconclusio vorkommen.
- G enthält keine nutzlosen Symbole.

Teil IV

- 1 Ableitungsbäume
- 2 Umformung von Grammatiken
- 3 Normalformen**
- 4 Pumping-Lemma für kontextfreie Sprachen
- 5 Pushdown-Automaten (PDAs)
- 6 Determinierte PDAs
- 7 Abschlusseigenschaften
- 8 Wortprobleme
- 9 Der CYK-Algorithmus

Teil IV

- 1 Ableitungsbäume
- 2 Umformung von Grammatiken
- 3 Normalformen
- 4 Pumping-Lemma für kontextfreie Sprachen**
- 5 Pushdown-Automaten (PDAs)
- 6 Determinierte PDAs
- 7 Abschlusseigenschaften
- 8 Wortprobleme
- 9 Der CYK-Algorithmus

Wiederholung: Pumping-Lemma für reguläre Sprachen

Theorem 21.1 (Pumping-Lemma für L_3 -Sprachen)

Sei $L \in \mathbf{RAT}$.

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$x \in L \text{ mit } |x| \geq n$$

existiert eine Zerlegung

$$x = uvw \quad u, v, w \in \Sigma^*$$

mit

- $|v| \geq 1$
- $|v| < n$
- $uv^m w \in L$ für alle $m \in \mathbb{N}$

Wiederholung: Pumping-Lemma für reguläre Sprachen

Theorem 21.1 (Pumping-Lemma für L_3 -Sprachen)

Sei $L \in \mathbf{RAT}$.

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$x \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$x = uvw \quad u, v, w \in \Sigma^*$$

mit

- $|v| \geq 1$
- $|v| < n$
- $uv^m w \in L$ für alle $m \in \mathbb{N}$

Wiederholung: Pumping-Lemma für reguläre Sprachen

Theorem 21.1 (Pumping-Lemma für L_3 -Sprachen)

Sei $L \in \mathbf{RAT}$.

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$x \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$x = uvw \quad u, v, w \in \Sigma^*$$

mit

- $|v| \geq 1$
- $|v| < n$
- $uv^m w \in L$ für alle $m \in \mathbb{N}$

Wiederholung: Pumping-Lemma für reguläre Sprachen

Theorem 21.1 (Pumping-Lemma für L_3 -Sprachen)

Sei $L \in \mathbf{RAT}$.

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$x \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$x = uvw \quad u, v, w \in \Sigma^*$$

mit

- $|v| \geq 1$
- $|v| < n$
- $uv^m w \in L$ für alle $m \in \mathbb{N}$

Wiederholung: Pumping-Lemma für reguläre Sprachen

Theorem 21.1 (Pumping-Lemma für L_3 -Sprachen)

Sei $L \in \mathbf{RAT}$.

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$x \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$x = uvw \quad u, v, w \in \Sigma^*$$

mit

- $|v| \geq 1$

- $|v| < n$

- $uv^m w \in L$ für alle $m \in \mathbb{N}$

Wiederholung: Pumping-Lemma für reguläre Sprachen

Theorem 21.1 (Pumping-Lemma für L_3 -Sprachen)

Sei $L \in \mathbf{RAT}$.

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$x \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$x = uvw \quad u, v, w \in \Sigma^*$$

mit

- $|v| \geq 1$
- $|v| < n$
- $uv^m w \in L$ für alle $m \in \mathbb{N}$

Pumping-Lemma für kontextfreie Sprachen

Theorem 21.2 (Pumping-Lemma für kontextfreie Sprachen)

Sei L kontextfrei

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$z \in L \text{ mit } |x| \geq n$$

existiert eine Zerlegung

$$z = uvwxy \quad u, v, w, x, y \in \Sigma^*$$

mit

- $|vx| \geq 1$
- $|vwx| < n$
- $uv^mwx^my \in L$ für alle $m \in \mathbb{N}$

Pumping-Lemma für kontextfreie Sprachen

Theorem 21.2 (Pumping-Lemma für kontextfreie Sprachen)

Sei L kontextfrei

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$z \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$z = uvwxy \quad u, v, w, x, y \in \Sigma^*$$

mit

- $|vx| \geq 1$
- $|vwx| < n$
- $uv^mwx^my \in L$ für alle $m \in \mathbb{N}$

Pumping-Lemma für kontextfreie Sprachen

Theorem 21.2 (Pumping-Lemma für kontextfreie Sprachen)

Sei L kontextfrei

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$z \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$z = uvwxy \quad u, v, w, x, y \in \Sigma^*$$

mit

- $|vx| \geq 1$
- $|vwx| < n$
- $uv^mwx^my \in L$ für alle $m \in \mathbb{N}$

Pumping-Lemma für kontextfreie Sprachen

Theorem 21.2 (Pumping-Lemma für kontextfreie Sprachen)

Sei L kontextfrei

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$z \in L \text{ mit } |x| \geq n$$

existiert eine Zerlegung

$$z = uvwxy \quad u, v, w, x, y \in \Sigma^*$$

mit

- $|vx| \geq 1$
- $|vwx| < n$
- $uv^mwx^my \in L$ für alle $m \in \mathbb{N}$

Pumping-Lemma für kontextfreie Sprachen

Theorem 21.2 (Pumping-Lemma für kontextfreie Sprachen)

Sei L kontextfrei

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$z \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$z = uvwxy \quad u, v, w, x, y \in \Sigma^*$$

mit

- $|vx| \geq 1$
- $|vwx| < n$
- $uv^mwx^my \in L$ für alle $m \in \mathbb{N}$

Pumping-Lemma für kontextfreie Sprachen

Theorem 21.2 (Pumping-Lemma für kontextfreie Sprachen)

Sei L kontextfrei

Dann existiert ein $n \in \mathbb{N}$, so dass:

Für alle

$$z \in L \quad \text{mit} \quad |x| \geq n$$

existiert eine Zerlegung

$$z = uvwxy \quad u, v, w, x, y \in \Sigma^*$$

mit

- $|vx| \geq 1$
- $|vwx| < n$
- $uv^mwx^my \in L$ für alle $m \in \mathbb{N}$

Pumping-Lemma für kontextfreie Sprachen

Beweisidee

Bei der Ableitung eines hinreichend langen Wortes muss es eine Variable geben, die mehr als einmal auftaucht.

Dies führt zu einer Schleife in der Ableitung, die aufgepumpt werden kann.

Beweisidee

Bei der Ableitung eines hinreichend langen Wortes muss es eine Variable geben, die mehr als einmal auftaucht.

Dies führt zu einer Schleife in der Ableitung, die aufgepumpt werden kann.

Anwendung des Pumping-Lemmas für cf-Sprachen

Wenn das cf-Pumping-Lemma für eine Sprache nicht gilt, dann kann sie nicht kontextfrei sein.

Beispiel 21.3 (Sprachen, die nicht kontextfrei sind)

Für folgende Sprachen kann man mit Hilfe des cf-Pumping-Lemmas zeigen, dass sie nicht kontextfrei sind:

- $\{a^p \mid p \text{ prim}\}$
- $\{a^n b^n c^n \mid n \in \mathbb{N}\}$
- $\{zzz \mid z \in \{a, b\}^*\}d$.

Pumping-Lemma für kontextfreie Sprachen

Anwendung des Pumping-Lemmas für cf-Sprachen

Wenn das cf-Pumping-Lemma für eine Sprache nicht gilt, dann kann sie nicht kontextfrei sein.

Beispiel 21.3 (Sprachen, die nicht kontextfrei sind)

Für folgende Sprachen kann man mit Hilfe des cf-Pumping-Lemmas zeigen, dass sie nicht kontextfrei sind:

- $\{a^p \mid p \text{ prim}\}$
- $\{a^n b^n c^n \mid n \in \mathbb{N}\}$
- $\{zzz \mid z \in \{a, b\}^*\}$ d.