

Theoretische Informatik I



Einheit 3.3

Eigenschaften kontextfreier Sprachen



1. Abschlußeigenschaften
2. Normalformen
3. Prüfen von Eigenschaften / Syntaxanalyse
4. Wann sind Sprachen nicht kontextfrei?

ABSCHLUSSEIGENSCHAFTEN KONTEXTFREIER SPRACHEN

Typ-2 Sprachen sind komplizierter als Typ-3 Sprachen

Typ-2 Sprachen sind komplizierter als Typ-3 Sprachen

● Abgeschlossenheit gilt nur für 6 Operationen

- Vereinigung zweier kontextfreier Sprachen $L_1 \cup L_2$
- Spiegelung einer kontextfreien Sprache L^R
- Hülle einer kontextfreien Sprache L^*
- Verkettung zweier kontextfreier Sprachen $L_1 \circ L_2$
- Substitution/Homomorphismus einer kontextfreien Sprache $\sigma(L)$
- Inverse Homomorphismus einer kontextfreien Sprache $h^{-1}(L)$

Typ-2 Sprachen sind komplizierter als Typ-3 Sprachen

● Abgeschlossenheit gilt nur für 6 Operationen

- Vereinigung zweier kontextfreier Sprachen $L_1 \cup L_2$
- Spiegelung einer kontextfreien Sprache L^R
- Hülle einer kontextfreien Sprache L^*
- Verkettung zweier kontextfreier Sprachen $L_1 \circ L_2$
- Substitution/Homomorphismus einer kontextfreien Sprache $\sigma(L)$
- Inverse Homomorphismus einer kontextfreien Sprache $h^{-1}(L)$

● Keine Abgeschlossenheit für

- Komplement einer kontextfreien Sprache \bar{L}
- Durchschnitt zweier kontextfreier Sprachen $L_1 \cap L_2$
- Differenz zweier kontextfreier Sprachen $L_1 - L_2$

Typ-2 Sprachen sind komplizierter als Typ-3 Sprachen

● Abgeschlossenheit gilt nur für 6 Operationen

- Vereinigung zweier kontextfreier Sprachen $L_1 \cup L_2$
- Spiegelung einer kontextfreien Sprache L^R
- Hülle einer kontextfreien Sprache L^*
- Verkettung zweier kontextfreier Sprachen $L_1 \circ L_2$
- Substitution/Homomorphismus einer kontextfreien Sprache $\sigma(L)$
- Inverse Homomorphismus einer kontextfreien Sprache $h^{-1}(L)$

● Keine Abgeschlossenheit für

- Komplement einer kontextfreien Sprache \bar{L}
- Durchschnitt zweier kontextfreier Sprachen $L_1 \cap L_2$
- Differenz zweier kontextfreier Sprachen $L_1 - L_2$

● Nachweis mit Grammatiken und PDAs

- Modelle sind ineinander umwandelbar – wähle das passendste

Typ-2 Sprachen sind komplizierter als Typ-3 Sprachen

● Abgeschlossenheit gilt nur für 6 Operationen

- Vereinigung zweier kontextfreier Sprachen $L_1 \cup L_2$
- Spiegelung einer kontextfreien Sprache L^R
- Hülle einer kontextfreien Sprache L^*
- Verkettung zweier kontextfreier Sprachen $L_1 \circ L_2$
- Substitution/Homomorphismus einer kontextfreien Sprache $\sigma(L)$
- Inverse Homomorphismus einer kontextfreien Sprache $h^{-1}(L)$

● Keine Abgeschlossenheit für

- Komplement einer kontextfreien Sprache \bar{L}
- Durchschnitt zweier kontextfreier Sprachen $L_1 \cap L_2$
- Differenz zweier kontextfreier Sprachen $L_1 - L_2$

● Nachweis mit Grammatiken und PDAs

- Modelle sind ineinander umwandelbar – wähle das passendste
- Negative Nachweise mit einem Typ-2 Pumping Lemma

SUBSTITUTIONEN VON SPRACHEN

Verallgemeinerung von Homomorphismen

SUBSTITUTIONEN VON SPRACHEN

Verallgemeinerung von Homomorphismen

- **Abbildung σ von Wörtern in Sprachen**

$\sigma: \Sigma^* \rightarrow \mathcal{L}$ ist **Substitution**, wenn $\sigma(v_1..v_n) = \sigma(v_1) \circ .. \circ \sigma(v_n)$ für alle $v_i \in \Sigma$

SUBSTITUTIONEN VON SPRACHEN

Verallgemeinerung von Homomorphismen

- **Abbildung σ von Wörtern in Sprachen**

$\sigma: \Sigma^* \rightarrow \mathcal{L}$ ist **Substitution**, wenn $\sigma(v_1..v_n) = \sigma(v_1) \circ .. \circ \sigma(v_n)$ für alle $v_i \in \Sigma$

$\sigma(L) = \bigcup \{ \sigma(w) \mid w \in L \}$ ist das Abbild der Wörter von L unter σ

SUBSTITUTIONEN VON SPRACHEN

Verallgemeinerung von Homomorphismen

- **Abbildung σ von Wörtern in Sprachen**

$\sigma: \Sigma^* \rightarrow \mathcal{L}$ ist **Substitution**, wenn $\sigma(v_1..v_n) = \sigma(v_1) \circ .. \circ \sigma(v_n)$ für alle $v_i \in \Sigma$

$\sigma(L) = \bigcup \{ \sigma(w) \mid w \in L \}$ ist das Abbild der Wörter von L unter σ

- **Beispiel: $\sigma(0) = \{a^n b^n \mid n \in \mathbb{N}\}$, $\sigma(1) = \{aa, bb\}$**

– $\sigma: \{0, 1\}^* \rightarrow \mathcal{L}$ ist eindeutig definiert durch $\sigma(0)$ und $\sigma(1)$

SUBSTITUTIONEN VON SPRACHEN

Verallgemeinerung von Homomorphismen

- **Abbildung σ von Wörtern in Sprachen**

$\sigma: \Sigma^* \rightarrow \mathcal{L}$ ist **Substitution**, wenn $\sigma(v_1..v_n) = \sigma(v_1) \circ .. \circ \sigma(v_n)$ für alle $v_i \in \Sigma$

$\sigma(L) = \bigcup \{ \sigma(w) \mid w \in L \}$ ist das Abbild der Wörter von L unter σ

- **Beispiel: $\sigma(0) = \{a^n b^n \mid n \in \mathbb{N}\}$, $\sigma(1) = \{aa, bb\}$**

– $\sigma: \{0, 1\}^* \rightarrow \mathcal{L}$ ist eindeutig definiert durch $\sigma(0)$ und $\sigma(1)$

– $\sigma(01) = \{a^n b^n \mid n \in \mathbb{N}\} \circ \{aa, bb\}$

$= \{w \in \{a, b\}^* \mid w = a^n b^{n+2} \vee w = a^n b^n aa \text{ für ein } n \in \mathbb{N}\}$

Verallgemeinerung von Homomorphismen

● Abbildung σ von Wörtern in Sprachen

$\sigma: \Sigma^* \rightarrow \mathcal{L}$ ist **Substitution**, wenn $\sigma(v_1..v_n) = \sigma(v_1) \circ .. \circ \sigma(v_n)$ für alle $v_i \in \Sigma$

$\sigma(L) = \bigcup \{ \sigma(w) \mid w \in L \}$ ist das Abbild der Wörter von L unter σ

● Beispiel: $\sigma(0) = \{a^n b^n \mid n \in \mathbb{N}\}$, $\sigma(1) = \{aa, bb\}$

– $\sigma: \{0, 1\}^* \rightarrow \mathcal{L}$ ist eindeutig definiert durch $\sigma(0)$ und $\sigma(1)$

– $\sigma(01) = \{a^n b^n \mid n \in \mathbb{N}\} \circ \{aa, bb\}$
 $= \{w \in \{a, b\}^* \mid w = a^n b^{n+2} \vee w = a^n b^n aa \text{ für ein } n \in \mathbb{N}\}$

– $\sigma(\{0\}^*) = \{a^n b^n \mid n \in \mathbb{N}\}^*$
 $= \{w \in \{a, b\}^* \mid w = a^{n_1} b^{n_1} a^{n_2} b^{n_2} .. a^{n_k} b^{n_k} \text{ für ein } k \text{ und } n_i \in \mathbb{N}\}$

Verallgemeinerung von Homomorphismen

- **Abbildung σ von Wörtern in Sprachen**

$\sigma: \Sigma^* \rightarrow \mathcal{L}$ ist **Substitution**, wenn $\sigma(v_1..v_n) = \sigma(v_1) \circ .. \circ \sigma(v_n)$ für alle $v_i \in \Sigma$

$\sigma(L) = \bigcup \{ \sigma(w) \mid w \in L \}$ ist das Abbild der Wörter von L unter σ

- **Beispiel: $\sigma(0) = \{a^n b^n \mid n \in \mathbb{N}\}$, $\sigma(1) = \{aa, bb\}$**

– $\sigma: \{0, 1\}^* \rightarrow \mathcal{L}$ ist eindeutig definiert durch $\sigma(0)$ und $\sigma(1)$

– $\sigma(01) = \{a^n b^n \mid n \in \mathbb{N}\} \circ \{aa, bb\}$
 $= \{w \in \{a, b\}^* \mid w = a^n b^{n+2} \vee w = a^n b^n aa \text{ für ein } n \in \mathbb{N}\}$

– $\sigma(\{0\}^*) = \{a^n b^n \mid n \in \mathbb{N}\}^*$
 $= \{w \in \{a, b\}^* \mid w = a^{n_1} b^{n_1} a^{n_2} b^{n_2} .. a^{n_k} b^{n_k} \text{ für ein } k \text{ und } n_i \in \mathbb{N}\}$

- **Extrem ausdrucksstarker Mechanismus**

– $L_1 \cup L_2 = \sigma(\{1, 2\})$ für $\sigma(1) = L_1$, $\sigma(2) = L_2$

Verallgemeinerung von Homomorphismen

● Abbildung σ von Wörtern in Sprachen

$\sigma: \Sigma^* \rightarrow \mathcal{L}$ ist **Substitution**, wenn $\sigma(v_1..v_n) = \sigma(v_1) \circ .. \circ \sigma(v_n)$ für alle $v_i \in \Sigma$

$\sigma(L) = \bigcup \{ \sigma(w) \mid w \in L \}$ ist das Abbild der Wörter von L unter σ

● Beispiel: $\sigma(0) = \{a^n b^n \mid n \in \mathbb{N}\}$, $\sigma(1) = \{aa, bb\}$

– $\sigma: \{0, 1\}^* \rightarrow \mathcal{L}$ ist eindeutig definiert durch $\sigma(0)$ und $\sigma(1)$

– $\sigma(01) = \{a^n b^n \mid n \in \mathbb{N}\} \circ \{aa, bb\}$
 $= \{w \in \{a, b\}^* \mid w = a^n b^{n+2} \vee w = a^n b^n aa \text{ für ein } n \in \mathbb{N}\}$

– $\sigma(\{0\}^*) = \{a^n b^n \mid n \in \mathbb{N}\}^*$
 $= \{w \in \{a, b\}^* \mid w = a^{n_1} b^{n_1} a^{n_2} b^{n_2} .. a^{n_k} b^{n_k} \text{ für ein } k \text{ und } n_i \in \mathbb{N}\}$

● Extrem ausdrucksstarker Mechanismus

– $L_1 \cup L_2 = \sigma(\{1, 2\})$ für $\sigma(1) = L_1$, $\sigma(2) = L_2$

– $L_1 \circ L_2 = \sigma(\{12\})$ für $\sigma(1) = L_1$, $\sigma(2) = L_2$

Verallgemeinerung von Homomorphismen

● Abbildung σ von Wörtern in Sprachen

$\sigma: \Sigma^* \rightarrow \mathcal{L}$ ist **Substitution**, wenn $\sigma(v_1..v_n) = \sigma(v_1) \circ .. \circ \sigma(v_n)$ für alle $v_i \in \Sigma$

$\sigma(L) = \bigcup \{ \sigma(w) \mid w \in L \}$ ist das Abbild der Wörter von L unter σ

● Beispiel: $\sigma(0) = \{a^n b^n \mid n \in \mathbb{N}\}$, $\sigma(1) = \{aa, bb\}$

– $\sigma: \{0, 1\}^* \rightarrow \mathcal{L}$ ist eindeutig definiert durch $\sigma(0)$ und $\sigma(1)$

– $\sigma(01) = \{a^n b^n \mid n \in \mathbb{N}\} \circ \{aa, bb\}$
 $= \{w \in \{a, b\}^* \mid w = a^n b^{n+2} \vee w = a^n b^n aa \text{ für ein } n \in \mathbb{N}\}$

– $\sigma(\{0\}^*) = \{a^n b^n \mid n \in \mathbb{N}\}^*$
 $= \{w \in \{a, b\}^* \mid w = a^{n_1} b^{n_1} a^{n_2} b^{n_2} .. a^{n_k} b^{n_k} \text{ für ein } k \text{ und } n_i \in \mathbb{N}\}$

● Extrem ausdrucksstarker Mechanismus

– $L_1 \cup L_2 = \sigma(\{1, 2\})$ für $\sigma(1) = L_1$, $\sigma(2) = L_2$

– $L_1 \circ L_2 = \sigma(\{12\})$ für $\sigma(1) = L_1$, $\sigma(2) = L_2$

– $L^* = \sigma(\{1\}^*)$ für $\sigma(1) = L$

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2$, $\sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2$, $\sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

- **Beweis mit Grammatiken**

“Ersetze $a \in T$ durch Startsymbol der kontextfreien Grammatik für $\sigma(a)$ ”

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2$, $\sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

● Beweis mit Grammatiken

“Ersetze $a \in T$ durch Startsymbol der kontextfreien Grammatik für $\sigma(a)$ ”

Seien L und $\sigma(a)$ kontextfrei für alle $a \in T$

Dann gibt es Typ-2 Grammatiken $G = (V, T, P, S)$ mit $L = L(G)$

und $G_a = (V_a, T_a, P_a, S_a)$ mit $\sigma(a) = L(G_a)$

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2, \sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

● Beweis mit Grammatiken

“Ersetze $a \in T$ durch Startsymbol der kontextfreien Grammatik für $\sigma(a)$ ”

Seien L und $\sigma(a)$ kontextfrei für alle $a \in T$

Dann gibt es Typ-2 Grammatiken $G = (V, T, P, S)$ mit $L = L(G)$

und $G_a = (V_a, T_a, P_a, S_a)$ mit $\sigma(a) = L(G_a)$

Dann ist $\sigma(L) = \sigma(L(G)) = \bigcup \{ \sigma(a_1) \circ \dots \circ \sigma(a_n) \mid S \xrightarrow{*} a_1 \dots a_n \}$
 $= \{ w_1 \dots w_n \mid \exists a_1 \dots a_n. S \xrightarrow{*} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*} w_i \}$

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2$, $\sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

• Beweis mit Grammatiken

“Ersetze $a \in T$ durch Startsymbol der kontextfreien Grammatik für $\sigma(a)$ ”

Seien L und $\sigma(a)$ kontextfrei für alle $a \in T$

Dann gibt es Typ-2 Grammatiken $G = (V, T, P, S)$ mit $L = L(G)$

und $G_a = (V_a, T_a, P_a, S_a)$ mit $\sigma(a) = L(G_a)$

Dann ist $\sigma(L) = \sigma(L(G)) = \bigcup \{ \sigma(a_1) \circ \dots \circ \sigma(a_n) \mid S \xrightarrow{*} a_1 \dots a_n \}$
 $= \{ w_1 \dots w_n \mid \exists a_1 \dots a_n. S \xrightarrow{*} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*} w_i \}$

Sei $P_\sigma = \{ A \rightarrow \alpha_\sigma \mid A \rightarrow \alpha \in P \} \cup \bigcup_{a \in T} P_a$, wobei α_σ aus $\alpha \in (V \cup T)^*$ entsteht, indem jedes $a \in T$ durch S_a ersetzt wird

und $G_\sigma = (V_\sigma, T_\sigma, P_\sigma, S)$ wobei $V_\sigma = V \cup \bigcup_{a \in T} V_a$ und $T_\sigma = \bigcup_{a \in T} T_a$

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2, \sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

• Beweis mit Grammatiken

“Ersetze $a \in T$ durch Startsymbol der kontextfreien Grammatik für $\sigma(a)$ ”

Seien L und $\sigma(a)$ kontextfrei für alle $a \in T$

Dann gibt es Typ-2 Grammatiken $G = (V, T, P, S)$ mit $L = L(G)$

und $G_a = (V_a, T_a, P_a, S_a)$ mit $\sigma(a) = L(G_a)$

Dann ist $\sigma(L) = \sigma(L(G)) = \bigcup \{ \sigma(a_1) \circ \dots \circ \sigma(a_n) \mid S \xrightarrow{*} a_1 \dots a_n \}$
 $= \{ w_1 \dots w_n \mid \exists a_1 \dots a_n. S \xrightarrow{*} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*} w_i \}$

Sei $P_\sigma = \{ A \rightarrow \alpha_\sigma \mid A \rightarrow \alpha \in P \} \cup \bigcup_{a \in T} P_a$, wobei α_σ aus $\alpha \in (V \cup T)^*$ entsteht, indem jedes $a \in T$ durch S_a ersetzt wird

und $G_\sigma = (V_\sigma, T_\sigma, P_\sigma, S)$ wobei $V_\sigma = V \cup \bigcup_{a \in T} V_a$ und $T_\sigma = \bigcup_{a \in T} T_a$

Dann gilt $w_1 \dots w_n \in L(G_\sigma) \Leftrightarrow S \xrightarrow{*}_{G_\sigma} w_1 \dots w_n$

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2$, $\sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

• Beweis mit Grammatiken

“Ersetze $a \in T$ durch Startsymbol der kontextfreien Grammatik für $\sigma(a)$ ”

Seien L und $\sigma(a)$ kontextfrei für alle $a \in T$

Dann gibt es Typ-2 Grammatiken $G = (V, T, P, S)$ mit $L = L(G)$

und $G_a = (V_a, T_a, P_a, S_a)$ mit $\sigma(a) = L(G_a)$

Dann ist $\sigma(L) = \sigma(L(G)) = \bigcup \{ \sigma(a_1) \circ \dots \circ \sigma(a_n) \mid S \xrightarrow{*} a_1 \dots a_n \}$
 $= \{ w_1 \dots w_n \mid \exists a_1 \dots a_n. S \xrightarrow{*} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*} w_i \}$

Sei $P_\sigma = \{ A \rightarrow \alpha_\sigma \mid A \rightarrow \alpha \in P \} \cup \bigcup_{a \in T} P_a$, wobei α_σ aus $\alpha \in (V \cup T)^*$ entsteht, indem jedes $a \in T$ durch S_a ersetzt wird

und $G_\sigma = (V_\sigma, T_\sigma, P_\sigma, S)$ wobei $V_\sigma = V \cup \bigcup_{a \in T} V_a$ und $T_\sigma = \bigcup_{a \in T} T_a$

Dann gilt $w_1 \dots w_n \in L(G_\sigma) \Leftrightarrow S \xrightarrow{*}_{G_\sigma} w_1 \dots w_n$
 $\Leftrightarrow \exists a_1 \dots a_n \in T^*. S \xrightarrow{*}_{G} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*}_{G_{a_i}} w_i$

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2$, $\sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

• Beweis mit Grammatiken

“Ersetze $a \in T$ durch Startsymbol der kontextfreien Grammatik für $\sigma(a)$ ”

Seien L und $\sigma(a)$ kontextfrei für alle $a \in T$

Dann gibt es Typ-2 Grammatiken $G = (V, T, P, S)$ mit $L = L(G)$

und $G_a = (V_a, T_a, P_a, S_a)$ mit $\sigma(a) = L(G_a)$

Dann ist $\sigma(L) = \sigma(L(G)) = \bigcup \{ \sigma(a_1) \circ \dots \circ \sigma(a_n) \mid S \xrightarrow{*} a_1 \dots a_n \}$
 $= \{ w_1 \dots w_n \mid \exists a_1 \dots a_n. S \xrightarrow{*} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*} w_i \}$

Sei $P_\sigma = \{ A \rightarrow \alpha_\sigma \mid A \rightarrow \alpha \in P \} \cup \bigcup_{a \in T} P_a$, wobei α_σ aus $\alpha \in (V \cup T)^*$ entsteht, indem jedes $a \in T$ durch S_a ersetzt wird

und $G_\sigma = (V_\sigma, T_\sigma, P_\sigma, S)$ wobei $V_\sigma = V \cup \bigcup_{a \in T} V_a$ und $T_\sigma = \bigcup_{a \in T} T_a$

Dann gilt $w_1 \dots w_n \in L(G_\sigma) \Leftrightarrow S \xrightarrow{*}_{G_\sigma} w_1 \dots w_n$
 $\Leftrightarrow \exists a_1 \dots a_n \in T^*. S \xrightarrow{*}_{G} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*}_{G_{a_i}} w_i$
 $\Leftrightarrow w_1 \dots w_n \in \sigma(L)$

ABGESCHLOSSENHEIT UNTER SUBSTITUTIONEN

$L \in \mathcal{L}_2$, $\sigma: T^* \rightarrow \mathcal{L}_2$ Substitution, $\Rightarrow \sigma(L)$ kontextfrei

• Beweis mit Grammatiken

“Ersetze $a \in T$ durch Startsymbol der kontextfreien Grammatik für $\sigma(a)$ ”

Seien L und $\sigma(a)$ kontextfrei für alle $a \in T$

Dann gibt es Typ-2 Grammatiken $G = (V, T, P, S)$ mit $L = L(G)$

und $G_a = (V_a, T_a, P_a, S_a)$ mit $\sigma(a) = L(G_a)$

Dann ist $\sigma(L) = \sigma(L(G)) = \bigcup \{ \sigma(a_1) \circ \dots \circ \sigma(a_n) \mid S \xrightarrow{*} a_1 \dots a_n \}$
 $= \{ w_1 \dots w_n \mid \exists a_1 \dots a_n. S \xrightarrow{*} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*} w_i \}$

Sei $P_\sigma = \{ A \rightarrow \alpha_\sigma \mid A \rightarrow \alpha \in P \} \cup \bigcup_{a \in T} P_a$, wobei α_σ aus $\alpha \in (V \cup T)^*$ entsteht, indem jedes $a \in T$ durch S_a ersetzt wird

und $G_\sigma = (V_\sigma, T_\sigma, P_\sigma, S)$ wobei $V_\sigma = V \cup \bigcup_{a \in T} V_a$ und $T_\sigma = \bigcup_{a \in T} T_a$

Dann gilt $w_1 \dots w_n \in L(G_\sigma) \Leftrightarrow S \xrightarrow{*}_{G_\sigma} w_1 \dots w_n$
 $\Leftrightarrow \exists a_1 \dots a_n \in T^*. S \xrightarrow{*}_{G} a_1 \dots a_n \wedge S_{a_i} \xrightarrow{*}_{G_{a_i}} w_i$
 $\Leftrightarrow w_1 \dots w_n \in \sigma(L)$

Also ist $\sigma(L)$ kontextfrei

VEREINIGUNG, VERKETTUNG, HÜLLE, HOMOMORPHISMEN

Verwende Abgeschlossenheit unter Substitutionen

VEREINIGUNG, VERKETTUNG, HÜLLE, HOMOMORPHISMEN

Verwende Abgeschlossenheit unter Substitutionen

- L_1, L_2 kontextfrei $\Rightarrow L_1 \cup L_2$ kontextfrei

Verwende Abgeschlossenheit unter Substitutionen

- L_1, L_2 kontextfrei $\Rightarrow L_1 \cup L_2$ kontextfrei
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1,2\}\rightarrow\mathcal{L}_2$ Substitution und $L_1 \cup L_2 = \sigma(\{1,2\}) \in \mathcal{L}_2$

Verwende Abgeschlossenheit unter Substitutionen

- L_1, L_2 kontextfrei $\Rightarrow L_1 \cup L_2$ kontextfrei
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1,2\}\rightarrow\mathcal{L}_2$ Substitution und $L_1 \cup L_2 = \sigma(\{1,2\}) \in \mathcal{L}_2$
- L_1, L_2 kontextfrei $\Rightarrow L_1 \circ L_2$ kontextfrei

Verwende Abgeschlossenheit unter Substitutionen

- L_1, L_2 kontextfrei $\Rightarrow L_1 \cup L_2$ kontextfrei
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1,2\}\rightarrow\mathcal{L}_2$ Substitution und $L_1 \cup L_2 = \sigma(\{1,2\}) \in \mathcal{L}_2$
- L_1, L_2 kontextfrei $\Rightarrow L_1 \circ L_2$ kontextfrei
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1,2\}\rightarrow\mathcal{L}_2$ Substitution und $L_1 \circ L_2 = \sigma(\{12\}) \in \mathcal{L}_2$

Verwende Abgeschlossenheit unter Substitutionen

- **L_1, L_2 kontextfrei $\Rightarrow L_1 \cup L_2$ kontextfrei**
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1, 2\} \rightarrow \mathcal{L}_2$ Substitution und $L_1 \cup L_2 = \sigma(\{1,2\}) \in \mathcal{L}_2$
- **L_1, L_2 kontextfrei $\Rightarrow L_1 \circ L_2$ kontextfrei**
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1, 2\} \rightarrow \mathcal{L}_2$ Substitution und $L_1 \circ L_2 = \sigma(\{12\}) \in \mathcal{L}_2$
- **L kontextfrei $\Rightarrow L^*$ kontextfrei**
 - Für $\sigma(1)=L$ ist $\sigma:\{1\} \rightarrow \mathcal{L}_2$ Substitution und $L^* = \sigma(\{1\}^*) \in \mathcal{L}_2$

Verwende Abgeschlossenheit unter Substitutionen

- **L_1, L_2 kontextfrei $\Rightarrow L_1 \cup L_2$ kontextfrei**
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1, 2\} \rightarrow \mathcal{L}_2$ Substitution und $L_1 \cup L_2 = \sigma(\{1,2\}) \in \mathcal{L}_2$
- **L_1, L_2 kontextfrei $\Rightarrow L_1 \circ L_2$ kontextfrei**
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1, 2\} \rightarrow \mathcal{L}_2$ Substitution und $L_1 \circ L_2 = \sigma(\{12\}) \in \mathcal{L}_2$
- **L kontextfrei $\Rightarrow L^*$ kontextfrei**
 - Für $\sigma(1)=L$ ist $\sigma:\{1\} \rightarrow \mathcal{L}_2$ Substitution und $L^* = \sigma(\{1\}^*) \in \mathcal{L}_2$
- **L kontextfrei $\Rightarrow L^+$ kontextfrei**
 - Für $\sigma(1)=L$ ist $\sigma:\{1\} \rightarrow \mathcal{L}_2$ Substitution und $L^+ = \sigma(\{1\}^+) \in \mathcal{L}_2$

Verwende Abgeschlossenheit unter Substitutionen

- L_1, L_2 kontextfrei $\Rightarrow L_1 \cup L_2$ kontextfrei
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1, 2\} \rightarrow \mathcal{L}_2$ Substitution und $L_1 \cup L_2 = \sigma(\{1,2\}) \in \mathcal{L}_2$
- L_1, L_2 kontextfrei $\Rightarrow L_1 \circ L_2$ kontextfrei
 - Sei $\sigma(1)=L_1$ und $\sigma(2)=L_2$
 - Dann ist $\sigma:\{1, 2\} \rightarrow \mathcal{L}_2$ Substitution und $L_1 \circ L_2 = \sigma(\{12\}) \in \mathcal{L}_2$
- L kontextfrei $\Rightarrow L^*$ kontextfrei
 - Für $\sigma(1)=L$ ist $\sigma:\{1\} \rightarrow \mathcal{L}_2$ Substitution und $L^* = \sigma(\{1\}^*) \in \mathcal{L}_2$
- L kontextfrei $\Rightarrow L^+$ kontextfrei
 - Für $\sigma(1)=L$ ist $\sigma:\{1\} \rightarrow \mathcal{L}_2$ Substitution und $L^+ = \sigma(\{1\}^+) \in \mathcal{L}_2$
- $L \in \mathcal{L}_2, h$ Homomorphismus $\Rightarrow h(L)$ kontextfrei
 - Für $\sigma(a)=\{h(a)\}$ ist $\sigma:T \rightarrow \mathcal{L}_2$ Substitution und $h(L) = \sigma(L) \in \mathcal{L}_2$

ABSCHLUSS UNTER SPIEGELUNG

L kontextfrei $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$ kontextfrei

ABSCHLUSS UNTER SPIEGELUNG

L kontextfrei $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$ kontextfrei

- **Beweis mit Grammatiken**

- Bilde **Spiegelgrammatik** zu $G = (V, T, P, S)$ mit $L = L(G)$

ABSCHLUSS UNTER SPIEGELUNG

L kontextfrei $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$ kontextfrei

● Beweis mit Grammatiken

- Bilde Spiegelgrammatik zu $G = (V, T, P, S)$ mit $L = L(G)$
 - Setze $G_R = (V, T, P_R, S)$ mit $P_R = \{A \rightarrow \alpha^R \mid A \rightarrow \alpha \in P\}$

ABSCHLUSS UNTER SPIEGELUNG

L kontextfrei $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$ kontextfrei

● Beweis mit Grammatiken

- Bilde Spiegelgrammatik zu $G = (V, T, P, S)$ mit $L = L(G)$
 - Setze $G_R = (V, T, P_R, S)$ mit $P_R = \{A \rightarrow \alpha^R \mid A \rightarrow \alpha \in P\}$
- Dann gilt für alle $A \in V, w \in (V \cup T)^*$: $A \vdash_G^* w \Leftrightarrow A \vdash_{G_R}^* w^R$
 - Beweis durch Induktion über Länge der Ableitung

ABSCHLUSS UNTER SPIEGELUNG

L kontextfrei $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$ kontextfrei

● Beweis mit Grammatiken

- Bilde Spiegelgrammatik zu $G = (V, T, P, S)$ mit $L = L(G)$
 - Setze $G_R = (V, T, P_R, S)$ mit $P_R = \{A \rightarrow \alpha^R \mid A \rightarrow \alpha \in P\}$
- Dann gilt für alle $A \in V, w \in (V \cup T)^*$: $A \vdash_G^* w \Leftrightarrow A \vdash_{G_R}^* w^R$
 - Beweis durch Induktion über Länge der Ableitung
- Also $L(G_R) = \{w \in T^* \mid S \vdash_{G_R}^* w\} = \{v^R \in T^* \mid S \vdash_G^* v\} = (L(G))^R$

ABSCHLUSS UNTER SPIEGELUNG

L kontextfrei $\Rightarrow L^R = \{w_n..w_1 \mid w_1..w_n \in L\}$ kontextfrei

● Beweis mit Grammatiken

- Bilde **Spiegelgrammatik** zu $G = (V, T, P, S)$ mit $L = L(G)$
 - Setze $G_R = (V, T, P_R, S)$ mit $P_R = \{A \rightarrow \alpha^R \mid A \rightarrow \alpha \in P\}$
- Dann gilt für alle $A \in V, w \in (V \cup T)^*$: $A \vdash_G^* w \Leftrightarrow A \vdash_{G_R}^* w^R$
 - Beweis durch Induktion über Länge der Ableitung
- Also $L(G_R) = \{w \in T^* \mid S \vdash_{G_R}^* w\} = \{v^R \in T^* \mid S \vdash_G^* v\} = (L(G))^R$

● Beweis mit PDAs ähnlich wie bei Typ-3 Sprachen

- Bilde **Umkehrautomaten** zu $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ mit $L = L_F(P)$

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

- **Beweis mit Pushdown Automaten**

“Berechnung von h vor Abarbeitung der Wörter im Automaten”

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

● Beweis mit Pushdown Automaten

“Berechnung von h vor Abarbeitung der Wörter im Automaten”

Sei L kontextfrei und $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ ein PDA

mit $L = L_F(P) = \{ v \in \Sigma^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, v, Z_0) \vdash^* (q, \epsilon, \beta) \}$

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

● Beweis mit Pushdown Automaten

“Berechnung von h vor Abarbeitung der Wörter im Automaten”

Sei L kontextfrei und $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ ein PDA

mit $L = L_F(P) = \{ v \in \Sigma^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, v, Z_0) \vdash^* (q, \epsilon, \beta) \}$

Dann ist $h^{-1}(L) = \{ w \in \Sigma'^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta) \}$

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

● Beweis mit Pushdown Automaten

“Berechnung von h vor Abarbeitung der Wörter im Automaten”

Sei L kontextfrei und $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ ein PDA

mit $L = L_F(P) = \{ v \in \Sigma^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, v, Z_0) \vdash^* (q, \epsilon, \beta) \}$

Dann ist $h^{-1}(L) = \{ w \in \Sigma'^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta) \}$

Konstruiere PDA $P_h = (Q_h, \Sigma', \Gamma, \delta_h, q_{0_h}, Z_0, F_h)$ mit der Eigenschaft

$(q_{0_h}, w, Z_0) \vdash^* (q_h, \epsilon, \beta) \Leftrightarrow (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta)$ für Endzustände

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

● Beweis mit Pushdown Automaten

“Berechnung von h vor Abarbeitung der Wörter im Automaten”

Sei L kontextfrei und $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ ein PDA

mit $L = L_F(P) = \{ v \in \Sigma^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, v, Z_0) \vdash^* (q, \epsilon, \beta) \}$

Dann ist $h^{-1}(L) = \{ w \in \Sigma'^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta) \}$

Konstruiere PDA $P_h = (Q_h, \Sigma', \Gamma, \delta_h, q_{0_h}, Z_0, F_h)$ mit der Eigenschaft

$(q_{0_h}, w, Z_0) \vdash^* (q_h, \epsilon, \beta) \Leftrightarrow (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta)$ für Endzustände

Ein Ansatz wie $\delta_h(q, a, X) = \hat{\delta}(q, h(a), X)$ funktioniert nicht!

Wie bei DEAs muß $h(a)$ schrittweise in den Zuständen abgearbeitet werden

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

● Beweis mit Pushdown Automaten

“Berechnung von h vor Abarbeitung der Wörter im Automaten”

Sei L kontextfrei und $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ ein PDA

mit $L = L_F(P) = \{ v \in \Sigma^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, v, Z_0) \vdash^* (q, \epsilon, \beta) \}$

Dann ist $h^{-1}(L) = \{ w \in \Sigma'^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta) \}$

Konstruiere PDA $P_h = (Q_h, \Sigma', \Gamma, \delta_h, q_{0_h}, Z_0, F_h)$ mit der Eigenschaft

$(q_{0_h}, w, Z_0) \vdash^* (q_h, \epsilon, \beta) \Leftrightarrow (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta)$ für Endzustände

Ein Ansatz wie $\delta_h(q, a, X) = \hat{\delta}(q, h(a), X)$ funktioniert nicht!

Wie bei DEAs muß $h(a)$ schrittweise in den Zuständen abgearbeitet werden

Setze $Q_h = Q \times \{ v \in \Sigma^* \mid v \text{ Suffix von } h(a) \text{ für ein } a \in \Sigma' \}$

$$\delta_h((q, \epsilon), a, X) = \{ ((q, h(a)), X) \} \quad a \in \Sigma', X \in \Gamma$$

$$\delta_h((q, bv), \epsilon, X) = \{ ((p, v), \alpha) \mid (p, \alpha) \in \delta(q, b, X) \} \quad b \in \Sigma \cup \{ \epsilon \}, v \in \Sigma^*, X \in \Gamma$$

$$q_{0_h} = (q_0, \epsilon) \quad F_h = \{ (q, \epsilon) \mid q \in F \}$$

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

• Beweis mit Pushdown Automaten

“Berechnung von h vor Abarbeitung der Wörter im Automaten”

Sei L kontextfrei und $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ ein PDA

mit $L = L_F(P) = \{ v \in \Sigma^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, v, Z_0) \vdash^* (q, \epsilon, \beta) \}$

Dann ist $h^{-1}(L) = \{ w \in \Sigma'^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta) \}$

Konstruiere PDA $P_h = (Q_h, \Sigma', \Gamma, \delta_h, q_{0_h}, Z_0, F_h)$ mit der Eigenschaft

$(q_{0_h}, w, Z_0) \vdash^* (q_h, \epsilon, \beta) \Leftrightarrow (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta)$ für Endzustände

Ein Ansatz wie $\delta_h(q, a, X) = \hat{\delta}(q, h(a), X)$ funktioniert nicht!

Wie bei DEAs muß $h(a)$ schrittweise in den Zuständen abgearbeitet werden

Setze $Q_h = Q \times \{ v \in \Sigma^* \mid v \text{ Suffix von } h(a) \text{ für ein } a \in \Sigma' \}$

$\delta_h((q, \epsilon), a, X) = \{ ((q, h(a)), X) \} \quad a \in \Sigma', X \in \Gamma$

$\delta_h((q, bv), \epsilon, X) = \{ ((p, v), \alpha) \mid (p, \alpha) \in \delta(q, b, X) \} \quad b \in \Sigma \cup \{ \epsilon \}, v \in \Sigma^*, X \in \Gamma$

$q_{0_h} = (q_0, \epsilon) \quad F_h = \{ (q, \epsilon) \mid q \in F \}$

Dann gilt $((q, \epsilon), a, X) \vdash_{P_h}^* ((p, \epsilon), \epsilon, \beta) \Leftrightarrow (q, h(a), X) \vdash_P^* (p, \epsilon, \beta)$

ABSCHLUSS UNTER INVERSEN HOMOMORPHISMEN

$L \in \mathcal{L}_2$, h Homomorphismus $\Rightarrow h^{-1}(L)$ kontextfrei

• Beweis mit Pushdown Automaten

“Berechnung von h vor Abarbeitung der Wörter im Automaten”

Sei L kontextfrei und $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ ein PDA

mit $L = L_F(P) = \{ v \in \Sigma^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, v, Z_0) \vdash^* (q, \epsilon, \beta) \}$

Dann ist $h^{-1}(L) = \{ w \in \Sigma'^* \mid \exists q \in F. \exists \beta \in \Gamma^*. (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta) \}$

Konstruiere PDA $P_h = (Q_h, \Sigma', \Gamma, \delta_h, q_{0_h}, Z_0, F_h)$ mit der Eigenschaft

$(q_{0_h}, w, Z_0) \vdash^* (q_h, \epsilon, \beta) \Leftrightarrow (q_0, h(w), Z_0) \vdash^* (q, \epsilon, \beta)$ für Endzustände

Ein Ansatz wie $\delta_h(q, a, X) = \hat{\delta}(q, h(a), X)$ funktioniert nicht!

Wie bei DEAs muß $h(a)$ schrittweise in den Zuständen abgearbeitet werden

Setze $Q_h = Q \times \{ v \in \Sigma^* \mid v \text{ Suffix von } h(a) \text{ für ein } a \in \Sigma' \}$

$\delta_h((q, \epsilon), a, X) = \{ ((q, h(a)), X) \} \quad a \in \Sigma', X \in \Gamma$

$\delta_h((q, bv), \epsilon, X) = \{ ((p, v), \alpha) \mid (p, \alpha) \in \delta(q, b, X) \} \quad b \in \Sigma \cup \{ \epsilon \}, v \in \Sigma^*, X \in \Gamma$

$q_{0_h} = (q_0, \epsilon) \quad F_h = \{ (q, \epsilon) \mid q \in F \}$

Dann gilt $((q, \epsilon), a, X) \vdash_{P_h}^* ((p, \epsilon), \epsilon, \beta) \Leftrightarrow (q, h(a), X) \vdash_P^* (p, \epsilon, \beta)$

Also ist $h^{-1}(L) = L(P_h)$ und damit kontextfrei

DURCHSCHNITT, KOMPLEMENT UND DIFFERENZ

Abgeschlossenheit gilt **nicht** für diese Operationen

- **Durchschnitt:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 \cap L_2 \in \mathcal{L}_2$
 - $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei (Beweis später)

DURCHSCHNITT, KOMPLEMENT UND DIFFERENZ

Abgeschlossenheit gilt **nicht** für diese Operationen

- **Durchschnitt:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 \cap L_2 \in \mathcal{L}_2$

- $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei (Beweis später)

- Aber $L = \{0^n 1^n 2^m \mid n, m \in \mathbb{N}\} \cap \{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$

- und $\{0^n 1^n 2^m \mid n, m \in \mathbb{N}\}$ und $\{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$ sind kontextfrei

- (Regeln für erste Sprache: $S \rightarrow AB, A \rightarrow 0A1, A \rightarrow 01, B \rightarrow 2B, B \rightarrow 2$)

DURCHSCHNITT, KOMPLEMENT UND DIFFERENZ

Abgeschlossenheit gilt **nicht** für diese Operationen

- **Durchschnitt:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 \cap L_2 \in \mathcal{L}_2$

- $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei (Beweis später)

- Aber $L = \{0^n 1^n 2^m \mid n, m \in \mathbb{N}\} \cap \{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$

- und $\{0^n 1^n 2^m \mid n, m \in \mathbb{N}\}$ und $\{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$ sind kontextfrei

- (Regeln für erste Sprache: $S \rightarrow AB, A \rightarrow 0A1, A \rightarrow 01, B \rightarrow 2B, B \rightarrow 2$)

Der Durchschnitt kontextfreier und regulärer Sprachen ist kontextfrei

(HMU Satz 7.27)

DURCHSCHNITT, KOMPLEMENT UND DIFFERENZ

Abgeschlossenheit gilt **nicht** für diese Operationen

- **Durchschnitt:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 \cap L_2 \in \mathcal{L}_2$

- $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei (Beweis später)

- Aber $L = \{0^n 1^n 2^m \mid n, m \in \mathbb{N}\} \cap \{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$

- und $\{0^n 1^n 2^m \mid n, m \in \mathbb{N}\}$ und $\{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$ sind kontextfrei

- (Regeln für erste Sprache: $S \rightarrow AB, A \rightarrow 0A1, A \rightarrow 01, B \rightarrow 2B, B \rightarrow 2$)

Der Durchschnitt kontextfreier und regulärer Sprachen ist kontextfrei

(HMU Satz 7.27)

- **Komplement** $L \in \mathcal{L}_2 \not\Rightarrow \overline{L} \in \mathcal{L}_2$

- Es ist $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

DURCHSCHNITT, KOMPLEMENT UND DIFFERENZ

Abgeschlossenheit gilt **nicht** für diese Operationen

- **Durchschnitt:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 \cap L_2 \in \mathcal{L}_2$

- $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei (Beweis später)

- Aber $L = \{0^n 1^n 2^m \mid n, m \in \mathbb{N}\} \cap \{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$

- und $\{0^n 1^n 2^m \mid n, m \in \mathbb{N}\}$ und $\{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$ sind kontextfrei

- (Regeln für erste Sprache: $S \rightarrow AB, A \rightarrow 0A1, A \rightarrow 01, B \rightarrow 2B, B \rightarrow 2$)

Der Durchschnitt kontextfreier und regulärer Sprachen ist kontextfrei

(HMU Satz 7.27)

- **Komplement** $L \in \mathcal{L}_2 \not\Rightarrow \overline{L} \in \mathcal{L}_2$

- Es ist $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

- Bei Abgeschlossenheit unter Komplementbildung würde Abgeschlossenheit unter Durchschnitt folgen

DURCHSCHNITT, KOMPLEMENT UND DIFFERENZ

Abgeschlossenheit gilt **nicht** für diese Operationen

- **Durchschnitt:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 \cap L_2 \in \mathcal{L}_2$

- $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei (Beweis später)

- Aber $L = \{0^n 1^n 2^m \mid n, m \in \mathbb{N}\} \cap \{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$

- und $\{0^n 1^n 2^m \mid n, m \in \mathbb{N}\}$ und $\{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$ sind kontextfrei

- (Regeln für erste Sprache: $S \rightarrow AB, A \rightarrow 0A1, A \rightarrow 01, B \rightarrow 2B, B \rightarrow 2$)

Der Durchschnitt kontextfreier und regulärer Sprachen ist kontextfrei

(HMU Satz 7.27)

- **Komplement** $L \in \mathcal{L}_2 \not\Rightarrow \bar{L} \in \mathcal{L}_2$

- Es ist $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

- Bei Abgeschlossenheit unter Komplementbildung würde Abgeschlossenheit unter Durchschnitt folgen

- **Differenz:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 - L_2 \in \mathcal{L}_2$

- Es ist $\bar{L} = \Sigma^* - L$

Abgeschlossenheit gilt **nicht** für diese Operationen

- **Durchschnitt:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 \cap L_2 \in \mathcal{L}_2$

- $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ ist nicht kontextfrei (Beweis später)

- Aber $L = \{0^n 1^n 2^m \mid n, m \in \mathbb{N}\} \cap \{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$

- und $\{0^n 1^n 2^m \mid n, m \in \mathbb{N}\}$ und $\{0^m 1^n 2^n \mid n, m \in \mathbb{N}\}$ sind kontextfrei

- (Regeln für erste Sprache: $S \rightarrow AB, A \rightarrow 0A1, A \rightarrow 01, B \rightarrow 2B, B \rightarrow 2$)

Der Durchschnitt kontextfreier und regulärer Sprachen ist kontextfrei

(HMU Satz 7.27)

- **Komplement** $L \in \mathcal{L}_2 \not\Rightarrow \bar{L} \in \mathcal{L}_2$

- Es ist $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

- Bei Abgeschlossenheit unter Komplementbildung würde Abgeschlossenheit unter Durchschnitt folgen

- **Differenz:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 - L_2 \in \mathcal{L}_2$

- Es ist $\bar{L} = \Sigma^* - L$

- Aus Abschluß unter Differenz folgt Abschluß unter Komplement

TESTS FÜR EIGENSCHAFTEN KONTEXTFREIER SPRACHEN

Welche Eigenschaften sind automatisch prüfbar?

Welche Eigenschaften sind automatisch prüfbar?

- **Ist eine kontextfreie Sprache leer?**
 - Entspricht Test auf Erreichbarkeit von Endzuständen
 - Nicht ganz so einfach, da Stackinhalt die Erreichbarkeit beeinflusst

Welche Eigenschaften sind automatisch prüfbar?

- **Ist eine kontextfreie Sprache leer?**
 - Entspricht Test auf Erreichbarkeit von Endzuständen
 - Nicht ganz so einfach, da Stackinhalt die Erreichbarkeit beeinflusst
- **Zugehörigkeit: gehört ein Wort zur Sprache?**
 - Verarbeitung durch Pushdown-Automaten ist nichtdeterministisch
 - Deterministische Pushdown-Automaten sind nicht mächtig genug
 - Frage nach Zugehörigkeit beinhaltet oft Frage nach Ableitungsbaum

Welche Eigenschaften sind automatisch prüfbar?

- **Ist eine kontextfreie Sprache leer?**
 - Entspricht Test auf Erreichbarkeit von Endzuständen
 - Nicht ganz so einfach, da Stackinhalt die Erreichbarkeit beeinflusst
- **Zugehörigkeit: gehört ein Wort zur Sprache?**
 - Verarbeitung durch Pushdown-Automaten ist nichtdeterministisch
 - Deterministische Pushdown-Automaten sind nicht mächtig genug
 - Frage nach Zugehörigkeit beinhaltet oft Frage nach Ableitungsbaum
- **Äquivalenz: sind zwei Typ-2 Sprachen identisch?**
 - Zusammenfassen äquivalenter Zustände im PDA kaum durchführbar

Welche Eigenschaften sind automatisch prüfbar?

- **Ist eine kontextfreie Sprache leer?**
 - Entspricht Test auf Erreichbarkeit von Endzuständen
 - Nicht ganz so einfach, da Stackinhalt die Erreichbarkeit beeinflusst
- **Zugehörigkeit: gehört ein Wort zur Sprache?**
 - Verarbeitung durch Pushdown-Automaten ist nichtdeterministisch
 - Deterministische Pushdown-Automaten sind nicht mächtig genug
 - Frage nach Zugehörigkeit beinhaltet oft Frage nach Ableitungsbaum
- **Äquivalenz: sind zwei Typ-2 Sprachen identisch?**
 - Zusammenfassen äquivalenter Zustände im PDA kaum durchführbar
- **Kontextfreie Grammatiken sind zu kompliziert**
 - Analyse braucht einfachere Versionen von Typ-2 Grammatiken
 - Bringe Grammatik auf “Normalform” (äquivalente einfachere Struktur)

DIE CHOMSKY NORMALFORM

Trenne Variablen von Terminalsymbolen

- Grammatik in **Chomsky-Normalform**

- Grammatik $G = (V, T, P, S)$, bei der jede Produktion die Form $A \rightarrow BC$ oder $A \rightarrow a$ hat $(A, B, C \in V, a \in T)$

DIE CHOMSKY NORMALFORM

Trenne Variablen von Terminalsymbolen

- **Grammatik in Chomsky-Normalform**

- Grammatik $G = (V, T, P, S)$, bei der jede Produktion die Form $A \rightarrow BC$ oder $A \rightarrow a$ hat $(A, B, C \in V, a \in T)$
- Grammatiken in Chomsky Normalform sind auch kontextsensitiv

DIE CHOMSKY NORMALFORM

Trenne Variablen von Terminalsymbolen

- **Grammatik in Chomsky-Normalform**
 - Grammatik $G = (V, T, P, S)$, bei der jede Produktion die Form $A \rightarrow BC$ oder $A \rightarrow a$ hat $(A, B, C \in V, a \in T)$
 - Grammatiken in Chomsky Normalform sind auch kontextsensitiv
- **Jede kontextfreie Grammatik G mit $\epsilon \notin L(G)$ ist in Chomsky-Normalform transformierbar**

DIE CHOMSKY NORMALFORM

Trenne Variablen von Terminalsymbolen

- **Grammatik in Chomsky-Normalform**

- Grammatik $G = (V, T, P, S)$, bei der jede Produktion die Form $A \rightarrow BC$ oder $A \rightarrow a$ hat $(A, B, C \in V, a \in T)$
- Grammatiken in Chomsky Normalform sind auch kontextsensitiv

- **Jede kontextfreie Grammatik G mit $\epsilon \notin L(G)$ ist in Chomsky-Normalform transformierbar**

1. Eliminierung von ϵ -Produktionen $A \rightarrow \epsilon$
2. Eliminierung von Einheitsproduktionen $A \rightarrow B$
3. Eliminierung unnützer Symbole

DIE CHOMSKY NORMALFORM

Trenne Variablen von Terminalsymbolen

- **Grammatik in Chomsky-Normalform**

- Grammatik $G = (V, T, P, S)$, bei der jede Produktion die Form $A \rightarrow BC$ oder $A \rightarrow a$ hat $(A, B, C \in V, a \in T)$
- Grammatiken in Chomsky Normalform sind auch kontextsensitiv

- **Jede kontextfreie Grammatik G mit $\epsilon \notin L(G)$ ist in Chomsky-Normalform transformierbar**

1. Eliminierung von ϵ -Produktionen $A \rightarrow \epsilon$
2. Eliminierung von Einheitsproduktionen $A \rightarrow B$
3. Eliminierung unnützer Symbole
4. Separieren von Terminalsymbolen und Variablen in Produktionen
5. Aufspalten von Produktionen $A \rightarrow \alpha$ mit $|\alpha| > 2$

DIE CHOMSKY NORMALFORM

Trenne Variablen von Terminalsymbolen

- **Grammatik in Chomsky-Normalform**

- Grammatik $G = (V, T, P, S)$, bei der jede Produktion die Form $A \rightarrow BC$ oder $A \rightarrow a$ hat $(A, B, C \in V, a \in T)$
- Grammatiken in Chomsky Normalform sind auch kontextsensitiv

- **Jede kontextfreie Grammatik G mit $\epsilon \notin L(G)$ ist in Chomsky-Normalform transformierbar**

1. Eliminierung von ϵ -Produktionen $A \rightarrow \epsilon$
2. Eliminierung von Einheitsproduktionen $A \rightarrow B$
3. Eliminierung unnützer Symbole
4. Separieren von Terminalsymbolen und Variablen in Produktionen
5. Aufspalten von Produktionen $A \rightarrow \alpha$ mit $|\alpha| > 2$

Aufblähung/Transformationszeit quadratisch relativ zur Größe von G

ϵ -PRODUKTIONEN ELIMINIEREN

- ϵ -Produktionen sind überflüssig, falls $\epsilon \notin L(G)$
 - Variablen $A \in V$ mit $A \xrightarrow{*} \epsilon$ sind **eliminierbar**

ϵ -PRODUKTIONEN ELIMINIEREN

- ϵ -Produktionen sind überflüssig, falls $\epsilon \notin L(G)$
 - Variablen $A \in V$ mit $A \xrightarrow{*} \epsilon$ sind **eliminierbar**
 - Menge eliminierbarer Symbole kann iterativ bestimmt werden
 - Ist $A \rightarrow \epsilon \in P$ dann ist A eliminierbar
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i eliminierbar, dann ist A eliminierbar

ϵ -PRODUKTIONEN ELIMINIEREN

- ϵ -Produktionen sind überflüssig, falls $\epsilon \notin L(G)$
 - Variablen $A \in V$ mit $A \xrightarrow{*} \epsilon$ sind **eliminierbar**
 - Menge eliminierbarer Symbole kann iterativ bestimmt werden
 - Ist $A \rightarrow \epsilon \in P$ dann ist A eliminierbar
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i eliminierbar, dann ist A eliminierbar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen

ϵ -PRODUKTIONEN ELIMINIEREN

- ϵ -Produktionen sind überflüssig, falls $\epsilon \notin L(G)$
 - Variablen $A \in V$ mit $A \xrightarrow{*} \epsilon$ sind **eliminierbar**
 - Menge eliminierbarer Symbole kann iterativ bestimmt werden
 - Ist $A \rightarrow \epsilon \in P$ dann ist A eliminierbar
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i eliminierbar, dann ist A eliminierbar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Erzeuge Grammatik ohne eliminierbare Symbole**
 - Für $G = (V, T, P, S)$ bestimme alle eliminierbare Variablen

ϵ -PRODUKTIONEN ELIMINIEREN

- ϵ -Produktionen sind überflüssig, falls $\epsilon \notin L(G)$
 - Variablen $A \in V$ mit $A \xrightarrow{*} \epsilon$ sind **eliminierbar**
 - Menge eliminierbarer Symbole kann iterativ bestimmt werden
 - Ist $A \rightarrow \epsilon \in P$ dann ist A eliminierbar
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i eliminierbar, dann ist A eliminierbar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Erzeuge Grammatik ohne eliminierbare Symbole**
 - Für $G = (V, T, P, S)$ bestimme alle eliminierbare Variablen
 - Für $A \rightarrow \alpha \in P$ mit eliminierbaren Symbolen X_1, \dots, X_m in α erzeuge 2^m Regeln $A \rightarrow \alpha_{i_1, \dots, i_k}$ (Streiche jeweils die Symbole $X_{i_1}..X_{i_k}$ aus α)

ϵ -PRODUKTIONEN ELIMINIEREN

- **ϵ -Produktionen sind überflüssig, falls $\epsilon \notin L(G)$**
 - Variablen $A \in V$ mit $A \xrightarrow{*} \epsilon$ sind **eliminierbar**
 - Menge eliminierbarer Symbole kann iterativ bestimmt werden
 - Ist $A \rightarrow \epsilon \in P$ dann ist A eliminierbar
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i eliminierbar, dann ist A eliminierbar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Erzeuge Grammatik ohne eliminierbare Symbole**
 - Für $G = (V, T, P, S)$ bestimme alle eliminierbare Variablen
 - Für $A \rightarrow \alpha \in P$ mit eliminierbaren Symbolen X_1, \dots, X_m in α erzeuge 2^m Regeln $A \rightarrow \alpha_{i_1, \dots, i_k}$ (Streiche jeweils die Symbole $X_{i_1}..X_{i_k}$ aus α)
 - Entferne alle Regeln der Form $A \rightarrow \epsilon$ (auch neu erzeugte)

ϵ -PRODUKTIONEN ELIMINIEREN

- **ϵ -Produktionen sind überflüssig, falls $\epsilon \notin L(G)$**
 - Variablen $A \in V$ mit $A \xrightarrow{*} \epsilon$ sind **eliminierbar**
 - Menge eliminierbarer Symbole kann iterativ bestimmt werden
 - Ist $A \rightarrow \epsilon \in P$ dann ist A eliminierbar
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i eliminierbar, dann ist A eliminierbar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Erzeuge Grammatik ohne eliminierbare Symbole**
 - Für $G = (V, T, P, S)$ bestimme alle eliminierbare Variablen
 - Für $A \rightarrow \alpha \in P$ mit eliminierbaren Symbolen X_1, \dots, X_m in α erzeuge 2^m Regeln $A \rightarrow \alpha_{i_1, \dots, i_k}$ (Streiche jeweils die Symbole $X_{i_1}..X_{i_k}$ aus α)
 - Entferne alle Regeln der Form $A \rightarrow \epsilon$ (auch neu erzeugte)
 - Wenn S eliminierbar ist, kann $S' \rightarrow S$ und $S' \rightarrow \epsilon$ ergänzt werden

ϵ -PRODUKTIONEN ELIMINIEREN

- **ϵ -Produktionen sind überflüssig, falls $\epsilon \notin L(G)$**
 - Variablen $A \in V$ mit $A \xrightarrow{*} \epsilon$ sind **eliminierbar**
 - Menge eliminierbarer Symbole kann iterativ bestimmt werden
 - Ist $A \rightarrow \epsilon \in P$ dann ist A eliminierbar
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i eliminierbar, dann ist A eliminierbar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Erzeuge Grammatik ohne eliminierbare Symbole**
 - Für $G = (V, T, P, S)$ bestimme alle eliminierbare Variablen
 - Für $A \rightarrow \alpha \in P$ mit eliminierbaren Symbolen X_1, \dots, X_m in α erzeuge 2^m Regeln $A \rightarrow \alpha_{i_1, \dots, i_k}$ (Streiche jeweils die Symbole $X_{i_1}..X_{i_k}$ aus α)
 - Entferne alle Regeln der Form $A \rightarrow \epsilon$ (auch neu erzeugte)
 - Wenn S eliminierbar ist, kann $S' \rightarrow S$ und $S' \rightarrow \epsilon$ ergänzt werden
- **Erzeugte Grammatik ist äquivalent**
 - Zeige $A \xrightarrow{*}_{G'} w \Leftrightarrow A \xrightarrow{*}_G w \wedge (w \neq \epsilon \vee A = S')$
durch Induktion über Länge der Ableitung

ELIMINATION VON ϵ -PRODUKTIONEN AM BEISPIEL

$$P = \{ S \rightarrow AB, A \rightarrow aAA \mid \epsilon, B \rightarrow bBB \mid \epsilon \}$$

- **Ermittlung eliminierbarer Symbole**

1.: A und B sind eliminierbar

ELIMINATION VON ϵ -PRODUKTIONEN AM BEISPIEL

$$P = \{ S \rightarrow AB, A \rightarrow aAA \mid \epsilon, B \rightarrow bBB \mid \epsilon \}$$

- **Ermittlung eliminierbarer Symbole**

- 1.: A und B sind eliminierbar
- 2.: S ist ebenfalls eliminierbar

ELIMINATION VON ϵ -PRODUKTIONEN AM BEISPIEL

$$P = \{ S \rightarrow AB, A \rightarrow aAA \mid \epsilon, B \rightarrow bBB \mid \epsilon \}$$

- **Ermittlung eliminierbarer Symbole**

1.: A und B sind eliminierbar

2.: S ist ebenfalls eliminierbar

- **Verändere Regeln der Grammatik**

– Aus $S \rightarrow AB$ wird $S \rightarrow AB \mid A \mid B$

ELIMINATION VON ϵ -PRODUKTIONEN AM BEISPIEL

$$P = \{ S \rightarrow AB, A \rightarrow aAA \mid \epsilon, B \rightarrow bBB \mid \epsilon \}$$

- **Ermittlung eliminierbarer Symbole**

1.: A und B sind eliminierbar

2.: S ist ebenfalls eliminierbar

- **Verändere Regeln der Grammatik**

– Aus $S \rightarrow AB$ wird $S \rightarrow AB \mid A \mid B$

– Aus $A \rightarrow aAA \mid \epsilon$ wird $A \rightarrow aAA \mid aA \mid a$

ELIMINATION VON ϵ -PRODUKTIONEN AM BEISPIEL

$$P = \{ S \rightarrow AB, A \rightarrow aAA \mid \epsilon, B \rightarrow bBB \mid \epsilon \}$$

- **Ermittlung eliminierbarer Symbole**

- 1.: A und B sind eliminierbar
- 2.: S ist ebenfalls eliminierbar

- **Verändere Regeln der Grammatik**

- Aus $S \rightarrow AB$ wird $S \rightarrow AB \mid A \mid B$
- Aus $A \rightarrow aAA \mid \epsilon$ wird $A \rightarrow aAA \mid aA \mid a$
- Aus $B \rightarrow bBB \mid \epsilon$ wird $B \rightarrow bBB \mid bB \mid b$

ELIMINATION VON ϵ -PRODUKTIONEN AM BEISPIEL

$$P = \{ S \rightarrow AB, A \rightarrow aAA \mid \epsilon, B \rightarrow bBB \mid \epsilon \}$$

- **Ermittlung eliminierbarer Symbole**

- 1.: A und B sind eliminierbar
- 2.: S ist ebenfalls eliminierbar

- **Verändere Regeln der Grammatik**

- Aus $S \rightarrow AB$ wird $S \rightarrow AB \mid A \mid B$
- Aus $A \rightarrow aAA \mid \epsilon$ wird $A \rightarrow aAA \mid aA \mid a$
- Aus $B \rightarrow bBB \mid \epsilon$ wird $B \rightarrow bBB \mid bB \mid b$

Grammatik erzeugt $L(G) - \{\epsilon\}$ ohne ϵ -Produktionen

ELIMINATION VON ϵ -PRODUKTIONEN AM BEISPIEL

$$P = \{ S \rightarrow AB, A \rightarrow aAA \mid \epsilon, B \rightarrow bBB \mid \epsilon \}$$

- **Ermittlung eliminierbarer Symbole**

- 1.: A und B sind eliminierbar
- 2.: S ist ebenfalls eliminierbar

- **Verändere Regeln der Grammatik**

- Aus $S \rightarrow AB$ wird $S \rightarrow AB \mid A \mid B$
- Aus $A \rightarrow aAA \mid \epsilon$ wird $A \rightarrow aAA \mid aA \mid a$
- Aus $B \rightarrow bBB \mid \epsilon$ wird $B \rightarrow bBB \mid bB \mid b$

Grammatik erzeugt $L(G) - \{\epsilon\}$ ohne ϵ -Produktionen

- **Ergänze neues Startsymbol**

- S war eliminierbar: ergänze Produktionen $S' \rightarrow S \mid \epsilon$

ELIMINATION VON ϵ -PRODUKTIONEN AM BEISPIEL

$$P = \{ S \rightarrow AB, A \rightarrow aAA \mid \epsilon, B \rightarrow bBB \mid \epsilon \}$$

- **Ermittlung eliminierbarer Symbole**

- 1.: A und B sind eliminierbar
- 2.: S ist ebenfalls eliminierbar

- **Verändere Regeln der Grammatik**

- Aus $S \rightarrow AB$ wird $S \rightarrow AB \mid A \mid B$
- Aus $A \rightarrow aAA \mid \epsilon$ wird $A \rightarrow aAA \mid aA \mid a$
- Aus $B \rightarrow bBB \mid \epsilon$ wird $B \rightarrow bBB \mid bB \mid b$

Grammatik erzeugt $L(G) - \{\epsilon\}$ ohne ϵ -Produktionen

- **Ergänze neues Startsymbol**

- S war eliminierbar: ergänze Produktionen $S' \rightarrow S \mid \epsilon$

Grammatik erzeugt $L(G)$ mit initialer ϵ -Produktion

EINHEITSPRODUKTIONEN ELIMINIEREN

Einheitsproduktionen verlängern Ableitungen
und verkomplizieren technische Beweise

EINHEITSPRODUKTIONEN ELIMINIEREN

Einheitsproduktionen verlängern Ableitungen und verkomplizieren technische Beweise

- Bestimme alle **Einheitspaare** (A, B) mit $A \xrightarrow{*} B$
 - Wie üblich ... iteratives Verfahren:
 - Alle Paare (A, A) für $A \in V$ sind Einheitspaare

Einheitsproduktionen verlängern Ableitungen und verkomplizieren technische Beweise

- Bestimme alle **Einheitspaare** (A, B) mit $A \xrightarrow{*} B$
 - Wie üblich ... iteratives Verfahren:
 - Alle Paare (A, A) für $A \in V$ sind Einheitspaare
 - Ist (A, B) Einheitspaar und $B \rightarrow C \in P$ dann ist (A, C) Einheitspaar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen

Einheitsproduktionen verlängern Ableitungen und verkomplizieren technische Beweise

- **Bestimme alle Einheitspaare (A, B) mit $A \xrightarrow{*} B$**
 - Wie üblich ... iteratives Verfahren:
 - Alle Paare (A, A) für $A \in V$ sind Einheitspaare
 - Ist (A, B) Einheitspaar und $B \rightarrow C \in P$ dann ist (A, C) Einheitspaar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Erzeuge Grammatik ohne Einheitsproduktionen**
 - Bestimme alle Einheitspaare in G
 - Für jedes Einheitspaar (A, B) erzeuge Produktionen
 $\{A \rightarrow \alpha \mid B \rightarrow \alpha \in P \text{ keine Einheitsproduktion}\}$

Einheitsproduktionen verlängern Ableitungen und verkomplizieren technische Beweise

- **Bestimme alle Einheitspaare (A, B) mit $A \xrightarrow{*} B$**
 - Wie üblich ... iteratives Verfahren:
 - Alle Paare (A, A) für $A \in V$ sind Einheitspaare
 - Ist (A, B) Einheitspaar und $B \rightarrow C \in P$ dann ist (A, C) Einheitspaar
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Erzeuge Grammatik ohne Einheitsproduktionen**
 - Bestimme alle Einheitspaare in G
 - Für jedes Einheitspaar (A, B) erzeuge Produktionen
 $\{A \rightarrow \alpha \mid B \rightarrow \alpha \in P \text{ keine Einheitsproduktion}\}$
- **Erzeugte Grammatik ist äquivalent**
 - Ableitungen in G' sind “Kurzformen” von Ableitungen in G
 - Beweis, wie immer, durch Induktion über Länge der Ableitung

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$
1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$
 - 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
 - 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$
 - 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
 - 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare
 - 3.: (E,F) und (T,I) sind ebenfalls Einheitspaare

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$
 - 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
 - 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare
 - 3.: (E,F) und (T,I) sind ebenfalls Einheitspaare
 - 4.: (E,I) ist ebenfalls Einheitspaar

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$
 - 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
 - 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare
 - 3.: (E,F) und (T,I) sind ebenfalls Einheitspaare
 - 4.: (E,I) ist ebenfalls Einheitspaar
 - 5.: Keine weiteren Einheitspaare möglich

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$
 - 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
 - 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare
 - 3.: (E,F) und (T,I) sind ebenfalls Einheitspaare
 - 4.: (E,I) ist ebenfalls Einheitspaar
 - 5.: Keine weiteren Einheitspaare möglich
- Erzeuge Grammatik ohne Einheitsproduktionen

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$
 - 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
 - 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare
 - 3.: (E,F) und (T,I) sind ebenfalls Einheitspaare
 - 4.: (E,I) ist ebenfalls Einheitspaar
 - 5.: Keine weiteren Einheitspaare möglich
- Erzeuge Grammatik ohne Einheitsproduktionen
 - Einheitspaare mit E : $\{E \rightarrow E+T \mid T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$

- 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
- 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare
- 3.: (E,F) und (T,I) sind ebenfalls Einheitspaare
- 4.: (E,I) ist ebenfalls Einheitspaar
- 5.: Keine weiteren Einheitspaare möglich

- Erzeuge Grammatik ohne Einheitsproduktionen

- Einheitspaare mit E : $\{E \rightarrow E+T \mid T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$
- Einheitspaare mit T : $\{T \rightarrow T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$

- 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
- 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare
- 3.: (E,F) und (T,I) sind ebenfalls Einheitspaare
- 4.: (E,I) ist ebenfalls Einheitspaar
- 5.: Keine weiteren Einheitspaare möglich

- Erzeuge Grammatik ohne Einheitsproduktionen

- Einheitspaare mit E : $\{E \rightarrow E+T \mid T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$
- Einheitspaare mit T : $\{T \rightarrow T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$
- Einheitspaare mit F : $\{F \rightarrow (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$

ELIMINATION VON EINHEITSPRODUKTIONEN AM BEISPIEL

$$P' = \{ E \rightarrow T \mid E+T, T \rightarrow F \mid T*F, F \rightarrow I \mid (E) \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- Bestimme alle **Einheitspaare** (A,B) mit $A \xrightarrow{*} B$

- 1.: (E,E) , (T,T) , (F,F) und (I,I) sind Einheitspaare
- 2.: (E,T) , (T,F) und (F,I) sind ebenfalls Einheitspaare
- 3.: (E,F) und (T,I) sind ebenfalls Einheitspaare
- 4.: (E,I) ist ebenfalls Einheitspaar
- 5.: Keine weiteren Einheitspaare möglich

- Erzeuge Grammatik ohne Einheitsproduktionen

- Einheitspaare mit E : $\{E \rightarrow E+T \mid T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$
- Einheitspaare mit T : $\{T \rightarrow T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$
- Einheitspaare mit F : $\{F \rightarrow (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$
- Einheitspaare mit I : $\{I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1\}$

UNNÜTZE SYMBOLE ELIMINIEREN

- **X nützlich**, falls $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w \in T^*$
 - **Erzeugend** ($X \xrightarrow{*} v \in T^*$) und **erreichbar** ($S \xrightarrow{*} \alpha X \beta$)

UNNÜTZE SYMBOLE ELIMINIEREN

- **X nützlich**, falls $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w \in T^*$
 - **Erzeugend** ($X \xrightarrow{*} v \in T^*$) und **erreichbar** ($S \xrightarrow{*} \alpha X \beta$)
- **Beispiel:** $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$

UNNÜTZE SYMBOLE ELIMINIEREN

- **X nützlich**, falls $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w \in T^*$
 - **Erzeugend** ($X \xrightarrow{*} v \in T^*$) und **erreichbar** ($S \xrightarrow{*} \alpha X \beta$)
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erreichbar: S, A, B, a , und b erzeugend: S, A, a , und b

UNNÜTZE SYMBOLE ELIMINIEREN

- **X nützlich**, falls $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w \in T^*$
 - **Erzeugend** ($X \xrightarrow{*} v \in T^*$) und **erreichbar** ($S \xrightarrow{*} \alpha X \beta$)
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erreichbar: S, A, B, a , und b erzeugend: S, A, a , und b
 - Nach Elimination von B : $\{ S \rightarrow a, A \rightarrow b \}$
 - Erreichbar: S und a erzeugend: S, A, a , und b

UNNÜTZE SYMBOLE ELIMINIEREN

- **X nützlich**, falls $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w \in T^*$
 - **Erzeugend** ($X \xrightarrow{*} v \in T^*$) und **erreichbar** ($S \xrightarrow{*} \alpha X \beta$)
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erreichbar: S, A, B, a , und b erzeugend: S, A, a , und b
 - Nach Elimination von B : $\{ S \rightarrow a, A \rightarrow b \}$
 - Erreichbar: S und a erzeugend: S, A, a , und b
 - Nach Elimination von A : $\{ S \rightarrow a \}$
 - Erreichbar: S und a erzeugend: S und a

UNNÜTZE SYMBOLE ELIMINIEREN

- **X nützlich**, falls $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w \in T^*$
 - **Erzeugend** ($X \xrightarrow{*} v \in T^*$) und **erreichbar** ($S \xrightarrow{*} \alpha X \beta$)
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erreichbar: S, A, B, a , und b erzeugend: S, A, a , und b
 - Nach Elimination von B : $\{ S \rightarrow a, A \rightarrow b \}$
 - Erreichbar: S und a erzeugend: S, A, a , und b
 - Nach Elimination von A : $\{ S \rightarrow a \}$
 - Erreichbar: S und a erzeugend: S und a

Erzeugte Produktionenmenge ist äquivalent zu P

UNNÜTZE SYMBOLE ELIMINIEREN

- **X nützlich**, falls $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w \in T^*$
 - **Erzeugend** ($X \xrightarrow{*} v \in T^*$) und **erreichbar** ($S \xrightarrow{*} \alpha X \beta$)
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erreichbar: S, A, B, a , und b erzeugend: S, A, a , und b
 - Nach Elimination von B : $\{ S \rightarrow a, A \rightarrow b \}$
 - Erreichbar: S und a erzeugend: S, A, a , und b
 - Nach Elimination von A : $\{ S \rightarrow a \}$
 - Erreichbar: S und a erzeugend: S und a

Erzeugte Produktionenmenge ist äquivalent zu P

- **Eliminationsverfahren für G mit $L(G) \neq \emptyset$**
 - Eliminiere nichterzeugende Symbole und Produktionen, die sie enthalten
 - Eliminiere unerreichbare Symbole und Produktionen, die sie enthalten

UNNÜTZE SYMBOLE ELIMINIEREN

- **X nützlich**, falls $S \xrightarrow{*} \alpha X \beta \xrightarrow{*} w \in T^*$
 - **Erzeugend** ($X \xrightarrow{*} v \in T^*$) und **erreichbar** ($S \xrightarrow{*} \alpha X \beta$)
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erreichbar: S, A, B, a , und b erzeugend: S, A, a , und b
 - Nach Elimination von B : $\{ S \rightarrow a, A \rightarrow b \}$
 - Erreichbar: S und a erzeugend: S, A, a , und b
 - Nach Elimination von A : $\{ S \rightarrow a \}$
 - Erreichbar: S und a erzeugend: S und a

Erzeugte Produktionenmenge ist äquivalent zu P

- **Eliminationsverfahren für G mit $L(G) \neq \emptyset$**
 - Eliminiere nichterzeugende Symbole und Produktionen, die sie enthalten
 - Eliminiere unerreichbare Symbole und Produktionen, die sie enthalten
- Resultierende Grammatik G' erzeugt dieselbe Sprache wie G**
- G' enthält nur nützliche Symbole und $S \in V'$
- Also $w \in L(G) \Leftrightarrow S \xrightarrow{*}_G w \Leftrightarrow S \xrightarrow{*}_{G'} w \Leftrightarrow w \in L(G')$

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
 - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
 - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erzeugende Symbole:

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
 - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erzeugende Symbole: 1.: a und b sind erzeugend

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
 - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erzeugende Symbole: 1.: a und b sind erzeugend
2.: S und A sind ebenfalls erzeugend

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
 - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erzeugende Symbole:
 - 1.: a und b sind erzeugend
 - 2.: S und A sind ebenfalls erzeugend
 - 3.: Keine weiteren Symbole sind erzeugend

BERECHNUNG ERZEUGENDER / ERREICHBARER SYMBOLE

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
 - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erzeugende Symbole:
 - 1.: a und b sind erzeugend
 - 2.: S und A sind ebenfalls erzeugend
 - 3.: Keine weiteren Symbole sind erzeugend
 - Erreichbare Symbole:
 - 1.: S ist erreichbar

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
 - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erzeugende Symbole:
 - 1.: a und b sind erzeugend
 - 2.: S und A sind ebenfalls erzeugend
 - 3.: Keine weiteren Symbole sind erzeugend
 - Erreichbare Symbole:
 - 1.: S ist erreichbar
 - 2.: A, B und a sind ebenfalls erreichbar

- **Generiere Menge erzeugender Symbole iterativ**
 - Alle Terminalsymbole $a \in T$ sind erzeugend
 - Ist $A \rightarrow X_1..X_n \in P$ und alle X_i erzeugend, dann ist A erzeugend
 - Verfahren terminiert nach maximal $|V| + 1$ Iterationen
- **Generiere Menge erreichbarer Symbole iterativ**
 - S ist erreichbar
 - Ist $A \rightarrow X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
 - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen
- **Beispiel: $P = \{ S \rightarrow AB \mid a, A \rightarrow b \}$**
 - Erzeugende Symbole:
 - 1.: a und b sind erzeugend
 - 2.: S und A sind ebenfalls erzeugend
 - 3.: Keine weiteren Symbole sind erzeugend
 - Erreichbare Symbole:
 - 1.: S ist erreichbar
 - 2.: A, B und a sind ebenfalls erreichbar
 - 3.: b ist ebenfalls erreichbar

ERZEUGUNG DER CHOMSKY-NORMALFORM

Nur Produktionen der Form $A \rightarrow BC$ oder $A \rightarrow a$

ERZEUGUNG DER CHOMSKY-NORMALFORM

Nur Produktionen der Form $A \rightarrow BC$ oder $A \rightarrow a$

- Jede kontextfreie Grammatik G ist umwandelbar in eine äquivalente Grammatik ohne unnütze Symbole, (echte) ϵ -Produktionen und Einheitsproduktionen
 - Falls $L(G) = \emptyset$, wähle $G' = (V, T, \emptyset, S)$ (Test auf \emptyset später)
 - Sonst eliminiere ϵ -Produktionen, Einheitsproduktionen, unnütze Symbole

ERZEUGUNG DER CHOMSKY-NORMALFORM

Nur Produktionen der Form $A \rightarrow BC$ oder $A \rightarrow a$

- Jede kontextfreie Grammatik G ist umwandelbar in eine äquivalente Grammatik ohne unnütze Symbole, (echte) ϵ -Produktionen und Einheitsproduktionen
 - Falls $L(G) = \emptyset$, wähle $G' = (V, T, \emptyset, S)$ (Test auf \emptyset später)
 - Sonst eliminiere ϵ -Produktionen, Einheitsproduktionen, unnütze Symbole
- **Separiere Terminalsymbole von Variablen**
 - Für jedes Terminalsymbol $a \in T$ erzeuge neue Variable X_a
 - Ersetze jede Produktion $A \rightarrow \alpha$ mit $|\alpha| \geq 2$ durch $A \rightarrow X_a$
 - Ergänze Produktionen $X_a \rightarrow a$ für alle $a \in T$

ERZEUGUNG DER CHOMSKY-NORMALFORM

Nur Produktionen der Form $A \rightarrow BC$ oder $A \rightarrow a$

- **Jede kontextfreie Grammatik G ist umwandelbar in eine äquivalente Grammatik ohne unnütze Symbole, (echte) ϵ -Produktionen und Einheitsproduktionen**
 - Falls $L(G) = \emptyset$, wähle $G' = (V, T, \emptyset, S)$ (Test auf \emptyset später)
 - Sonst eliminiere ϵ -Produktionen, Einheitsproduktionen, unnütze Symbole
- **Separiere Terminalsymbole von Variablen**
 - Für jedes Terminalsymbol $a \in T$ erzeuge neue Variable X_a
 - Ersetze jede Produktion $A \rightarrow \alpha$ mit $|\alpha| \geq 2$ durch $A \rightarrow X_a$
 - Ergänze Produktionen $X_a \rightarrow a$ für alle $a \in T$
- **Spalte Produktionen $A \rightarrow \alpha$ mit $|\alpha| > 2$**
 - Ersetze jede Produktion $A \rightarrow X_1..X_k$ durch $k-1$ Produktionen
 $A \rightarrow X_1Y_1, Y_1 \rightarrow X_2Y_2, \dots, Y_{k-2} \rightarrow X_{k-1}X_k$, wobei alle Y_i neue Variablen

ERZEUGUNG DER CHOMSKY-NORMALFORM AM BEISPIEL

$$P = \{ E \rightarrow E + T \mid T * F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\ T \rightarrow T * F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\ F \rightarrow (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

ERZEUGUNG DER CHOMSKY-NORMALFORM AM BEISPIEL

$$\begin{aligned}
 P = \{ & E \rightarrow E+T \mid T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\
 & T \rightarrow T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\
 & F \rightarrow (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\
 & I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}
 \end{aligned}$$

- **Separiere Terminalsymbole von Variablen**

$$\begin{aligned}
 P' = \{ & E \rightarrow EX_+T \mid TX_*F \mid X(EX) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\
 & T \rightarrow TX_*F \mid X(EX) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\
 & F \rightarrow X(EX) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\
 & I \rightarrow a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\
 & X_a \rightarrow a, X_b \rightarrow b, X_c \rightarrow c, X_0 \rightarrow 0, X_1 \rightarrow 1, X_+ \rightarrow +, X_* \rightarrow *, X_{(} \rightarrow (, X_{)} \rightarrow) \}
 \end{aligned}$$

ERZEUGUNG DER CHOMSKY-NORMALFORM AM BEISPIEL

$$P = \{ E \rightarrow E+T \mid T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\ T \rightarrow T*F \mid (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\ F \rightarrow (E) \mid a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \\ I \rightarrow a \mid b \mid c \mid Ia \mid Ib \mid Ic \mid I0 \mid I1 \}$$

- **Separiere Terminalsymbole von Variablen**

$$P' = \{ E \rightarrow EX_+T \mid TX_*F \mid X(EX) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\ T \rightarrow TX_*F \mid X(EX) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\ F \rightarrow X(EX) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\ I \rightarrow a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\ X_a \rightarrow a, X_b \rightarrow b, X_c \rightarrow c, X_0 \rightarrow 0, X_1 \rightarrow 1, X_+ \rightarrow +, X_* \rightarrow *, X_{(} \rightarrow (, X_{)} \rightarrow) \}$$

- **Spalte Produktionen $A \rightarrow \alpha$ mit $|\alpha| > 2$**

$$P' = \{ E \rightarrow EY_1 \mid TY_2 \mid X(Y_3) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\ T \rightarrow TY_2 \mid X(Y_3) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\ F \rightarrow X(Y_3) \mid a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\ I \rightarrow a \mid b \mid c \mid IX_a \mid IX_b \mid IX_c \mid IX_0 \mid IX_1 \\ Y_1 \rightarrow X_+T, Y_2 \rightarrow X_*F, Y_3 \rightarrow EX) \\ X_a \rightarrow a, X_b \rightarrow b, X_c \rightarrow c, X_0 \rightarrow 0, X_1 \rightarrow 1, X_+ \rightarrow +, X_* \rightarrow *, X_{(} \rightarrow (, X_{)} \rightarrow) \}$$

TESTS FÜR EIGENSCHAFTEN KONTEXTFREIER SPRACHEN

- Ist eine kontextfreie Sprache leer?

- **Ist eine kontextfreie Sprache leer?**

- Für $G = (V, T, P, S)$ gilt

- $L(G)$ ist leer genau dann wenn S nicht erzeugend ist

- Menge erzeugender Variablen kann iterativ bestimmt werden

- **Ist eine kontextfreie Sprache leer?**

- Für $G = (V, T, P, S)$ gilt

- $L(G)$ ist leer genau dann wenn S nicht erzeugend ist

- Menge erzeugender Variablen kann iterativ bestimmt werden

- Mit speziellen Datenstrukturen ist Test in linearer Zeit durchführbar

(Details ins HMU §7.4.3)

● Ist eine kontextfreie Sprache leer?

– Für $G = (V, T, P, S)$ gilt

$L(G)$ ist leer genau dann wenn S nicht erzeugend ist

– Menge erzeugender Variablen kann iterativ bestimmt werden

– Mit speziellen Datenstrukturen ist Test in linearer Zeit durchführbar

(Details ins HMU §7.4.3)

● Gehört ein Wort zu einer kontextfreien Sprache?

– Naive Methode für den Test $w \in L(G)$:

● Ist eine kontextfreie Sprache leer?

– Für $G = (V, T, P, S)$ gilt

$L(G)$ ist leer genau dann wenn S nicht erzeugend ist

– Menge erzeugender Variablen kann iterativ bestimmt werden

– Mit speziellen Datenstrukturen ist Test in linearer Zeit durchführbar

(Details ins HMU §7.4.3)

● Gehört ein Wort zu einer kontextfreien Sprache?

– Naive Methode für den Test $w \in L(G)$:

1. Erzeuge Chomsky-Normalform G' von G

2. In G' erzeuge alle Ableitungsbäume mit $2|w| - 1$ Variablenknoten

3. Teste, ob einer dieser Bäume das Wort w erzeugt

● Ist eine kontextfreie Sprache leer?

– Für $G = (V, T, P, S)$ gilt

$L(G)$ ist leer genau dann wenn S nicht erzeugend ist

– Menge erzeugender Variablen kann iterativ bestimmt werden

– Mit speziellen Datenstrukturen ist Test in linearer Zeit durchführbar

(Details ins HMU §7.4.3)

● Gehört ein Wort zu einer kontextfreien Sprache?

– Naive Methode für den Test $w \in L(G)$:

1. Erzeuge Chomsky-Normalform G' von G

2. In G' erzeuge alle Ableitungsbäume mit $2|w| - 1$ Variablenknoten

3. Teste, ob einer dieser Bäume das Wort w erzeugt

– Hochgradig ineffizient, da exponentiell viele Bäume zu erzeugen

– Iterative Analyseverfahren sind besser

SYNTAXANALYSE: COCKE-YOUNGER-KASAMI ALGORITHMUS

Bestimme Variablenmengen, aus denen $w_i..w_j$ ableitbar

- **Eingabe:** Grammatik $G = (V, T, P, S)$ in Chomsky-NF, $w \in T^*$

SYNTAXANALYSE: COCKE-YOUNGER-KASAMI ALGORITHMUS

Bestimme Variablenmengen, aus denen $w_i..w_j$ ableitbar

- **Eingabe:** Grammatik $G = (V, T, P, S)$ in Chomsky-NF, $w \in T^*$
- Berechne Mengen $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i...w_j\}$ iterativ

$w_1 \quad w_2 \quad \dots \quad w_{n-1} \quad w_n$

SYNTAXANALYSE: COCKE-YOUNGER-KASAMI ALGORITHMUS

Bestimme Variablenmengen, aus denen $w_i..w_j$ ableitbar

- **Eingabe:** Grammatik $G = (V, T, P, S)$ in Chomsky-NF, $w \in T^*$
- Berechne Mengen $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i..w_j\}$ iterativ
 $j=i: V_{i,i} = \{A \in V \mid A \rightarrow w_i \in P\}$

$V_{1,1}$	$V_{2,2}$	\dots	$V_{n-1,n-1}$	$V_{n,n}$
w_1	w_2	\dots	w_{n-1}	w_n

SYNTAXANALYSE: COCKE-YOUNGER-KASAMI ALGORITHMUS

Bestimme Variablenmengen, aus denen $w_i..w_j$ ableitbar

- **Eingabe:** Grammatik $G = (V, T, P, S)$ in Chomsky-NF, $w \in T^*$
- Berechne Mengen $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i...w_j\}$ iterativ

$$j=i: V_{i,i} = \{A \in V \mid A \rightarrow w_i \in P\}$$

$$j>i: V_{i,j} = \{A \in V \mid$$

$$\exists i \leq k < j.$$

$$\exists A \rightarrow BC \in P.$$

$$B \in V_{i,k} \wedge C \in V_{k+1,j}\}$$

$V_{1,2}$	$V_{2,3}$	\dots	$V_{n-1,n}$	
$V_{1,1}$	$V_{2,2}$	\dots	$V_{n-1,n-1}$	$V_{n,n}$
w_1	w_2	\dots	w_{n-1}	w_n

SYNTAXANALYSE: COCKE-YOUNGER-KASAMI ALGORITHMUS

Bestimme Variablenmengen, aus denen $w_i..w_j$ ableitbar

- **Eingabe:** Grammatik $G = (V, T, P, S)$ in Chomsky-NF, $w \in T^*$
- **Berechne Mengen** $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i...w_j\}$ iterativ

$$j=i: V_{i,i} = \{A \in V \mid A \rightarrow w_i \in P\}$$

$$j>i: V_{i,j} = \{A \in V \mid$$

$$\exists i \leq k < j.$$

$$\exists A \rightarrow BC \in P.$$

$$B \in V_{i,k} \wedge C \in V_{k+1,j}\}$$

$V_{1,n}$	
$V_{1,n-1}$	$V_{2,n}$
\vdots	\vdots
$V_{1,2}$	$V_{2,3} \dots V_{n-1,n}$
$V_{1,1}$	$V_{2,2} \dots V_{n-1,n-1} V_{n,n}$
w_1	$w_2 \dots w_{n-1} w_n$

SYNTAXANALYSE: COCKE-YOUNGER-KASAMI ALGORITHMUS

Bestimme Variablenmengen, aus denen $w_i..w_j$ ableitbar

- **Eingabe:** Grammatik $G = (V, T, P, S)$ in Chomsky-NF, $w \in T^*$
- Berechne Mengen $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i...w_j\}$ iterativ

$$j=i: V_{i,i} = \{A \in V \mid A \rightarrow w_i \in P\}$$

$$j>i: V_{i,j} = \{A \in V \mid$$

$$\exists i \leq k < j.$$

$$\exists A \rightarrow BC \in P.$$

$$B \in V_{i,k} \wedge C \in V_{k+1,j}\}$$

$V_{1,n}$	
$V_{1,n-1}$	$V_{2,n}$
\vdots	$\vdots \quad \vdots$
$V_{1,2}$	$V_{2,3} \dots V_{n-1,n}$
$V_{1,1}$	$V_{2,2} \dots V_{n-1,n-1} \quad V_{n,n}$
w_1	$w_2 \dots w_{n-1} \quad w_n$

- Akzeptiere w genau dann, wenn $S \in V_{1,|w|}$

SYNTAXANALYSE: COCKE-YOUNGER-KASAMI ALGORITHMUS

Bestimme Variablenmengen, aus denen $w_i..w_j$ ableitbar

• **Eingabe:** Grammatik $G = (V, T, P, S)$ in Chomsky-NF, $w \in T^*$

• Berechne Mengen $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i...w_j\}$ iterativ

$$j=i: V_{i,i} = \{A \in V \mid A \rightarrow w_i \in P\}$$

$$j>i: V_{i,j} = \{A \in V \mid$$

$$\exists i \leq k < j.$$

$$\exists A \rightarrow BC \in P.$$

$$B \in V_{i,k} \wedge C \in V_{k+1,j}\}$$

$V_{1,n}$	
$V_{1,n-1}$	$V_{2,n}$
\vdots	\vdots
$V_{1,2}$	$V_{2,3} \dots V_{n-1,n}$
$V_{1,1}$	$V_{2,2} \dots V_{n-1,n-1} V_{n,n}$
w_1	$w_2 \dots w_{n-1} w_n$

• Akzeptiere w genau dann, wenn $S \in V_{1,|w|}$

Entscheidet $w \in L(G)$ in kubischer Zeit relativ zur Größe von w
 Konstruiert gleichzeitig den Syntaxbaum von w

DER CYK-ALGORITHMUS AM BEISPIEL

$$\{ S \rightarrow AB|BC, A \rightarrow BA|a, B \rightarrow CC|b, C \rightarrow AB|a \}$$

- Prüfe $w = baaba \in L(G)$

DER CYK-ALGORITHMUS AM BEISPIEL

$\{ S \rightarrow AB|BC, A \rightarrow BA|a, B \rightarrow CC|b, C \rightarrow AB|a \}$

- Prüfe $w = baaba \in L(G)$
- Berechne $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i \dots w_j\}$

$b \quad a \quad a \quad b \quad a$

DER CYK-ALGORITHMUS AM BEISPIEL

$\{ S \rightarrow AB|BC, A \rightarrow BA|a, B \rightarrow CC|b, C \rightarrow AB|a \}$

- Prüfe $w = baaba \in L(G)$
- Berechne $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i \dots w_j\}$

$\{B\}$	$\{A, C\}$	$\{A, C\}$	$\{B\}$	$\{A, C\}$
b	a	a	b	a

DER CYK-ALGORITHMUS AM BEISPIEL

$$\{ S \rightarrow AB|BC, A \rightarrow BA|a, B \rightarrow CC|b, C \rightarrow AB|a \}$$

- Prüfe $w = baaba \in L(G)$
- Berechne $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i \dots w_j\}$

$\{S, A\}$	$\{B\}$	$\{S, C\}$	$\{S, A\}$	
$\{B\}$	$\{A, C\}$	$\{A, C\}$	$\{B\}$	$\{A, C\}$
b	a	a	b	a

DER CYK-ALGORITHMUS AM BEISPIEL

$$\{ S \rightarrow AB|BC, A \rightarrow BA|a, B \rightarrow CC|b, C \rightarrow AB|a \}$$

- Prüfe $w = baaba \in L(G)$
- Berechne $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i \dots w_j\}$

—	{B}	{B}		
{S, A}	{B}	{S, C}	{S, A}	
{B}	{A, C}	{A, C}	{B}	{A, C}
<i>b</i>	<i>a</i>	<i>a</i>	<i>b</i>	<i>a</i>

DER CYK-ALGORITHMUS AM BEISPIEL

$\{ S \rightarrow AB|BC, A \rightarrow BA|a, B \rightarrow CC|b, C \rightarrow AB|a \}$

- Prüfe $w = baaba \in L(G)$
- Berechne $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i \dots w_j\}$

—	$\{S, A, C\}$			
—	$\{B\}$	$\{B\}$		
$\{S, A\}$	$\{B\}$	$\{S, C\}$	$\{S, A\}$	
$\{B\}$	$\{A, C\}$	$\{A, C\}$	$\{B\}$	$\{A, C\}$
b	a	a	b	a

DER CYK-ALGORITHMUS AM BEISPIEL

$\{ S \rightarrow AB|BC, A \rightarrow BA|a, B \rightarrow CC|b, C \rightarrow AB|a \}$

- Prüfe $w = baaba \in L(G)$
- Berechne $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i \dots w_j\}$

$\{S, A, C\}$				
—	$\{S, A, C\}$			
—	$\{B\}$	$\{B\}$		
$\{S, A\}$	$\{B\}$	$\{S, C\}$	$\{S, A\}$	
$\{B\}$	$\{A, C\}$	$\{A, C\}$	$\{B\}$	$\{A, C\}$
b	a	a	b	a

DER CYK-ALGORITHMUS AM BEISPIEL

$\{ S \rightarrow AB|BC, A \rightarrow BA|a, B \rightarrow CC|b, C \rightarrow AB|a \}$

- Prüfe $w = baaba \in L(G)$
- Berechne $V_{i,j} = \{A \in V \mid A \xrightarrow{*} w_i \dots w_j\}$

$\{S, A, C\}$					
—	$\{S, A, C\}$				
—	$\{B\}$	$\{B\}$			
$\{S, A\}$	$\{B\}$	$\{S, C\}$	$\{S, A\}$		
$\{B\}$	$\{A, C\}$	$\{A, C\}$	$\{B\}$	$\{A, C\}$	
b	a	a	b	a	

- $S \in V_{1,5}$, also $w \in L(G)$

UNENTSCHEIDBARE PROBLEME FÜR TYP-2 SPRACHEN

Die folgenden Probleme können nicht getestet werden

- $L(G) = T^*$ Welche Menge beschreibt G ?
- $L(G_1) = L(G_2)$ Äquivalenz von Grammatiken
- $L(G_1) \subseteq L(G_2)$
- $L(G_1) \cap L(G_2) = \emptyset$
- $L(G) \in \mathcal{L}_3$
- $\overline{L(G)} \in \mathcal{L}_2$ kontextfreies Komplement?
- $L(G_1) \cap L(G_2) \in \mathcal{L}_2$ kontextfreier Schnitt ?

Beweise brauchen Berechenbarkeitstheorie / TI-2

GRENZEN KONTEXTFREIER SPRACHEN

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

GRENZEN KONTEXTFREIER SPRACHEN

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

- **Typ-2 Grammatiken arbeiten lokal**
 - Anwendbarkeit einer Produktion hängt nur von einer Variablen ab (der Kontext der Variablen ist irrelevant)

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

- **Typ-2 Grammatiken arbeiten lokal**

- Anwendbarkeit einer Produktion hängt nur von einer Variablen ab (der Kontext der Variablen ist irrelevant)
- Eine Regel kann nur an einer Stelle im Wort etwas erzeugen

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

- **Typ-2 Grammatiken arbeiten lokal**

- Anwendbarkeit einer Produktion hängt nur von einer Variablen ab (der Kontext der Variablen ist irrelevant)
- Eine Regel kann nur an einer Stelle im Wort etwas erzeugen
- Eine Typ-2 Grammatik kann entweder 0/1 oder 1/2 simultan erhöhen aber nicht beides gleichzeitig

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

- **Typ-2 Grammatiken arbeiten lokal**

- Anwendbarkeit einer Produktion hängt nur von einer Variablen ab (der Kontext der Variablen ist irrelevant)
- Eine Regel kann nur an einer Stelle im Wort etwas erzeugen
- Eine Typ-2 Grammatik kann entweder 0/1 oder 1/2 simultan erhöhen aber nicht beides gleichzeitig
- Grammatik müßte die Anzahl der 0/1 oder 1/2 im Voraus bestimmen und diese Anzahl für die 2 bzw. 0 im Namen der Variablen codieren

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

- **Typ-2 Grammatiken arbeiten lokal**

- Anwendbarkeit einer Produktion hängt nur von einer Variablen ab (der Kontext der Variablen ist irrelevant)
- Eine Regel kann nur an einer Stelle im Wort etwas erzeugen
- Eine Typ-2 Grammatik kann entweder 0/1 oder 1/2 simultan erhöhen aber nicht beides gleichzeitig
- Grammatik müßte die Anzahl der 0/1 oder 1/2 im Voraus bestimmen und diese Anzahl für die 2 bzw. 0 im Namen der Variablen codieren

- **Grammatiken sind endlich**

- Es gibt nur endlich viele Variablen

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

- **Typ-2 Grammatiken arbeiten lokal**

- Anwendbarkeit einer Produktion hängt nur von einer Variablen ab (der Kontext der Variablen ist irrelevant)
- Eine Regel kann nur an einer Stelle im Wort etwas erzeugen
- Eine Typ-2 Grammatik kann entweder 0/1 oder 1/2 simultan erhöhen aber nicht beides gleichzeitig
- Grammatik müßte die Anzahl der 0/1 oder 1/2 im Voraus bestimmen und diese Anzahl für die 2 bzw. 0 im Namen der Variablen codieren

- **Grammatiken sind endlich**

- Es gibt nur endlich viele Variablen
- Für $n > |V|$ muß eine Variable X doppelt benutzt worden sein zur Codierung von $0^n 1^n$ und $0^i 1^i$ mit $i < n$

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

● Typ-2 Grammatiken arbeiten lokal

- Anwendbarkeit einer Produktion hängt nur von einer Variablen ab (der Kontext der Variablen ist irrelevant)
- Eine Regel kann nur an einer Stelle im Wort etwas erzeugen
- Eine Typ-2 Grammatik kann entweder 0/1 oder 1/2 simultan erhöhen aber nicht beides gleichzeitig
- Grammatik müßte die Anzahl der 0/1 oder 1/2 im Voraus bestimmen und diese Anzahl für die 2 bzw. 0 im Namen der Variablen codieren

● Grammatiken sind endlich

- Es gibt nur endlich viele Variablen
- Für $n > |V|$ muß eine Variable X doppelt benutzt worden sein zur Codierung von $0^n 1^n$ und $0^i 1^i$ mit $i < n$
- Grammatik würde auch $0^n 1^n 2^i$ und $0^i 1^i 2^n$ generieren

Warum ist $L = \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$ nicht kontextfrei?

● Typ-2 Grammatiken arbeiten lokal

- Anwendbarkeit einer Produktion hängt nur von einer Variablen ab (der Kontext der Variablen ist irrelevant)
- Eine Regel kann nur an einer Stelle im Wort etwas erzeugen
- Eine Typ-2 Grammatik kann entweder 0/1 oder 1/2 simultan erhöhen aber nicht beides gleichzeitig
- Grammatik müßte die Anzahl der 0/1 oder 1/2 im Voraus bestimmen und diese Anzahl für die 2 bzw. 0 im Namen der Variablen codieren

● Grammatiken sind endlich

- Es gibt nur endlich viele Variablen
- Für $n > |V|$ muß eine Variable X doppelt benutzt worden sein zur Codierung von $0^n 1^n$ und $0^i 1^i$ mit $i < n$
- Grammatik würde auch $0^n 1^n 2^i$ und $0^i 1^i 2^n$ generieren

● Genaues Argument ist etwas komplizierter

- Allgemeine Version: **Pumping Lemma** für kontextfreie Sprachen

DAS PUMPING LEMMA FÜR KONTEXTFREIE SPRACHEN

Wie zeigt man, daß eine Sprache nicht kontextfrei ist?

DAS PUMPING LEMMA FÜR KONTEXTFREIE SPRACHEN

Wie zeigt man, daß eine Sprache nicht kontextfrei ist?

- Für jede kontextfreie Sprache $L \in \mathcal{L}_2$ gibt es eine Zahl $n \in \mathbb{N}$, so daß jedes Wort $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit den Eigenschaften
 - (1) $v \neq \epsilon$,
 - (2) $|v w x| \leq n$ und
 - (3) für alle $i \in \mathbb{N}$ ist $u v^i w x^i y \in L$

DAS PUMPING LEMMA FÜR KONTEXTFREIE SPRACHEN

Wie zeigt man, daß eine Sprache nicht kontextfrei ist?

- Für jede kontextfreie Sprache $L \in \mathcal{L}_2$ gibt es eine Zahl $n \in \mathbb{N}$, so daß jedes Wort $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit den Eigenschaften
 - (1) $v \neq \epsilon$,
 - (2) $|v w x| \leq n$ und
 - (3) für alle $i \in \mathbb{N}$ ist $u v^i w x^i y \in L$
- Aussage ist wechselseitig konstruktiv
 - Die Zahl n kann zu jeder kontextfreien Sprache L bestimmt werden
 - Die Zerlegung $z = u v w x y$ kann zu jedem Wort $z \in L$ bestimmt werden

DAS PUMPING LEMMA FÜR KONTEXTFREIE SPRACHEN

Wie zeigt man, daß eine Sprache nicht kontextfrei ist?

- Für jede kontextfreie Sprache $L \in \mathcal{L}_2$ gibt es eine Zahl $n \in \mathbb{N}$, so daß jedes Wort $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit den Eigenschaften
 - (1) $v \neq \epsilon$,
 - (2) $|v w x| \leq n$ und
 - (3) für alle $i \in \mathbb{N}$ ist $u v^i w x^i y \in L$
- Aussage ist wechselseitig konstruktiv
 - Die Zahl n kann zu jeder kontextfreien Sprache L bestimmt werden
 - Die Zerlegung $z = u v w x y$ kann zu jedem Wort $z \in L$ bestimmt werden
- Beweis benötigt Chomsky-Normalform
 - Ableitungen der Länge k können maximal Wörter der Länge 2^k generieren

DAS PUMPING LEMMA FÜR KONTEXTFREIE SPRACHEN

Wie zeigt man, daß eine Sprache nicht kontextfrei ist?

- Für jede kontextfreie Sprache $L \in \mathcal{L}_2$ gibt es eine Zahl $n \in \mathbb{N}$, so daß jedes Wort $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit den Eigenschaften
 - (1) $v \circ x \neq \epsilon$,
 - (2) $|v w x| \leq n$ und
 - (3) für alle $i \in \mathbb{N}$ ist $u v^i w x^i y \in L$
- Aussage ist wechselseitig konstruktiv
 - Die Zahl n kann zu jeder kontextfreien Sprache L bestimmt werden
 - Die Zerlegung $z = u v w x y$ kann zu jedem Wort $z \in L$ bestimmt werden
- Beweis benötigt Chomsky-Normalform
 - Ableitungen der Länge k können maximal Wörter der Länge 2^k generieren
 - Ableitungen der Länge $k > |V|$ benutzen ein Hilfssymbol X doppelt

DAS PUMPING LEMMA FÜR KONTEXTFREIE SPRACHEN

Wie zeigt man, daß eine Sprache nicht kontextfrei ist?

- Für jede kontextfreie Sprache $L \in \mathcal{L}_2$ gibt es eine Zahl $n \in \mathbb{N}$, so daß jedes Wort $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit den Eigenschaften
 - (1) $v \circ x \neq \epsilon$,
 - (2) $|v w x| \leq n$ und
 - (3) für alle $i \in \mathbb{N}$ ist $u v^i w x^i y \in L$
- Aussage ist wechselseitig konstruktiv
 - Die Zahl n kann zu jeder kontextfreien Sprache L bestimmt werden
 - Die Zerlegung $z = u v w x y$ kann zu jedem Wort $z \in L$ bestimmt werden
- Beweis benötigt Chomsky-Normalform
 - Ableitungen der Länge k können maximal Wörter der Länge 2^k generieren
 - Ableitungen der Länge $k > |V|$ benutzen ein Hilfssymbol X doppelt
 - Die Schleife der Ableitung von X aus X kann beliebig wiederholt werden

BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
(1) $v \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
(1) $v \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

– Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise

BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = uvwx y$ mit
(1) $v \neq \epsilon$, (2) $|vwx| \leq n$ (3) $uv^iwx^iy \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$

BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
(1) $v \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

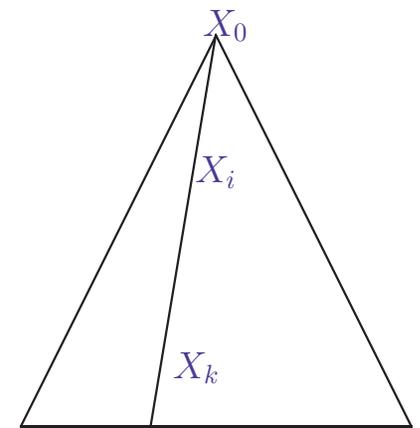
- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$
- Wähle $n = 2^{|V|}$ und betrachte $z = z_1 \dots z_m$ mit $|z| \geq n$

BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
(1) $v \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$
- Wähle $n = 2^{|V|}$ und betrachte $z = z_1 \dots z_m$ mit $|z| \geq n$
- Dann hat jeder Ableitungsbaum für z eine Tiefe von mindestens $|V| + 1$

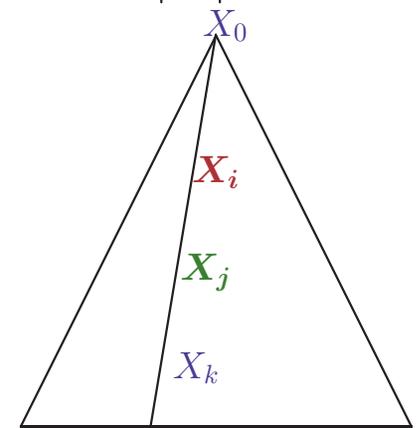


BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = uvwx y$ mit
(1) $v \neq \epsilon$, (2) $|vwx| \leq n$ (3) $uv^iwx^iy \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$
- Wähle $n = 2^{|V|}$ und betrachte $z = z_1 \dots z_m$ mit $|z| \geq n$
- Dann hat jeder Ableitungsbaum für z eine Tiefe von mindestens $|V| + 1$
- Sei X_0, \dots, X_k die Folge der verarbeiteten Variablen auf dem längsten Pfad
Dann erscheint eine Variable zweimal: $X_i = X_j$ für ein $i < j$ mit $k - |V| < i$

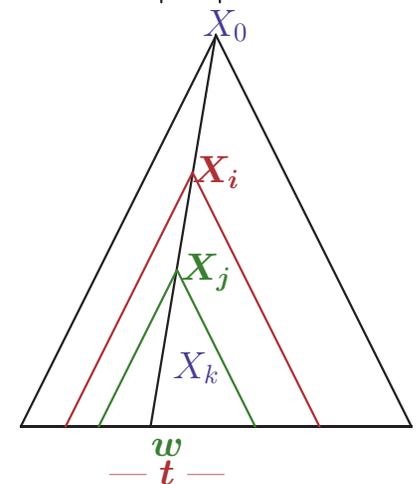


BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
(1) $v \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$
- Wähle $n = 2^{|V|}$ und betrachte $z = z_1 \dots z_m$ mit $|z| \geq n$
- Dann hat jeder Ableitungsbaum für z eine Tiefe von mindestens $|V| + 1$
- Sei X_0, \dots, X_k die Folge der verarbeiteten Variablen auf dem längsten Pfad
Dann erscheint eine Variable zweimal: $X_i = X_j$ für ein $i < j$ mit $k - |V| < i$
- Seien w und t die aus X_j bzw. X_i abgeleiteten Teilwörter

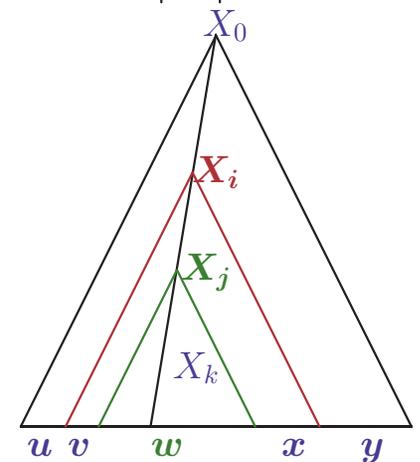


BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
(1) $v \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$
- Wähle $n = 2^{|V|}$ und betrachte $z = z_1 \dots z_m$ mit $|z| \geq n$
- Dann hat jeder Ableitungsbaum für z eine Tiefe von mindestens $|V| + 1$
- Sei X_0, \dots, X_k die Folge der verarbeiteten Variablen auf dem längsten Pfad
Dann erscheint eine Variable zweimal: $X_i = X_j$ für ein $i < j$ mit $k - |V| < i$
- Seien w und t die aus X_j bzw. X_i abgeleiteten Teilwörter
- Dann gilt $t = v w x$ und $z = u t y$ für Wörter u, v, x und y

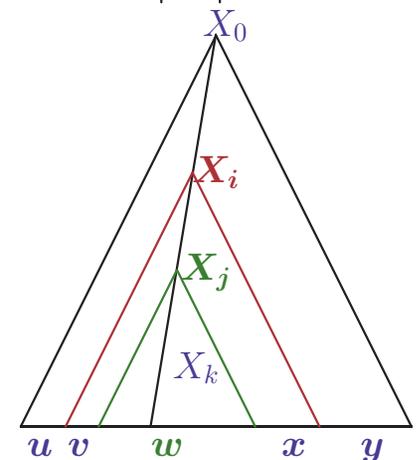


BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
 (1) $v \circ x \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$
- Wähle $n = 2^{|V|}$ und betrachte $z = z_1 \dots z_m$ mit $|z| \geq n$
- Dann hat jeder Ableitungsbaum für z eine Tiefe von mindestens $|V| + 1$
- Sei X_0, \dots, X_k die Folge der verarbeiteten Variablen auf dem längsten Pfad
 Dann erscheint eine Variable zweimal: $X_i = X_j$ für ein $i < j$ mit $k - |V| < i$
- Seien w und t die aus X_j bzw. X_i abgeleiteten Teilwörter
- Dann gilt $t = v w x$ und $z = u t y$ für Wörter u, v, x und y
- Da G in Chomsky-Normalform ist, gilt $v \circ x \neq \epsilon$

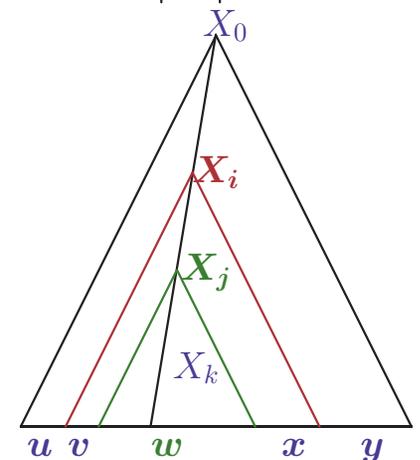


BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
(1) $v \circ x \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$
- Wähle $n = 2^{|V|}$ und betrachte $z = z_1 \dots z_m$ mit $|z| \geq n$
- Dann hat jeder Ableitungsbaum für z eine Tiefe von mindestens $|V| + 1$
- Sei X_0, \dots, X_k die Folge der verarbeiteten Variablen auf dem längsten Pfad
Dann erscheint eine Variable zweimal: $X_i = X_j$ für ein $i < j$ mit $k - |V| < i$
- Seien w und t die aus X_j bzw. X_i abgeleiteten Teilwörter
- Dann gilt $t = v w x$ und $z = u t y$ für Wörter u, v, x und y
- Da G in Chomsky-Normalform ist, gilt $v \circ x \neq \epsilon$
- Wegen $k - |V| < i$ gilt $|v w x| = |t| \leq n$

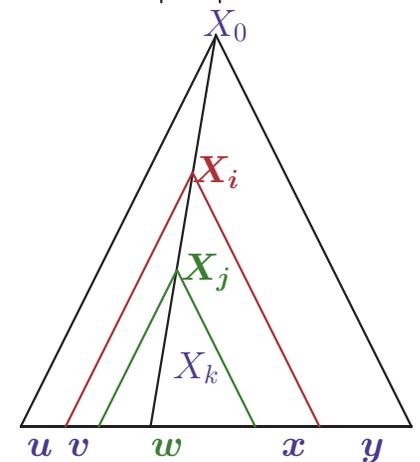


BEWEIS DES PUMPING LEMMAS

Für jede Sprache $L \in \mathcal{L}_2$ gibt es ein $n \in \mathbb{N}$, so daß jedes $z \in L$ mit Länge $|z| \geq n$ zerlegt werden kann in $z = u v w x y$ mit
 (1) $v \circ x \neq \epsilon$, (2) $|v w x| \leq n$ (3) $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$

Beweis mit Grammatiken in Chomsky-Normalform

- Für $L = \emptyset$ oder $L = \{\epsilon\}$ gilt die Behauptung trivialerweise
- Andernfalls sei $G = (V, T, P, S)$ in Chomsky-Normalform mit $L = L(G)$
- Wähle $n = 2^{|V|}$ und betrachte $z = z_1 \dots z_m$ mit $|z| \geq n$
- Dann hat jeder Ableitungsbaum für z eine Tiefe von mindestens $|V| + 1$
- Sei X_0, \dots, X_k die Folge der verarbeiteten Variablen auf dem längsten Pfad
 Dann erscheint eine Variable zweimal: $X_i = X_j$ für ein $i < j$ mit $k - |V| < i$
- Seien w und t die aus X_j bzw. X_i abgeleiteten Teilwörter
- Dann gilt $t = v w x$ und $z = u t y$ für Wörter u, v, x und y
- Da G in Chomsky-Normalform ist, gilt $v \circ x \neq \epsilon$
- Wegen $k - |V| < i$ gilt $|v w x| = |t| \leq n$
- Wegen $X_i = X_j$ kann die Ableitung von X_i bis X_j beliebig wiederholt werden und es gilt $u v^i w x^i y \in L$ für alle $i \in \mathbb{N}$



ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei
 - Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^iwx^iy \notin L) \Rightarrow L \notin \mathcal{L}_2$$

ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei

– Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^iwx^iy \notin L) \Rightarrow L \notin \mathcal{L}_2$$

– Sei $n \in \mathbb{N}$ beliebig. Wir wählen $z = 0^m 1^m 2^m$ für ein $m > n$

ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei

- Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v \circ x \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^iwx^iy \notin L) \Rightarrow L \notin \mathcal{L}_2$$

- Sei $n \in \mathbb{N}$ beliebig. Wir wählen $z = 0^m 1^m 2^m$ für ein $m > n$

- Sei $u, v, w, x, y \in T^*$ beliebig mit $z = uvwxy$,
und (1) $v \circ x \neq \epsilon$ und (2) $|vwx| \leq n$

ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei

- Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v \circ x \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^iwx^iy \notin L) \Rightarrow L \notin \mathcal{L}_2$$

- Sei $n \in \mathbb{N}$ beliebig. Wir wählen $z = 0^m 1^m 2^m$ für ein $m > n$

- Sei $u, v, w, x, y \in T^*$ beliebig mit $z = uvwxy$,
und (1) $v \circ x \neq \epsilon$ und (2) $|vwx| \leq n$

- Wir wählen $i = 0$ und zeigen $uw y = uv^iwx^iy \notin L$

ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei

- Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v \circ x \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^iwx^iy \notin L) \Rightarrow L \notin \mathcal{L}_2$$

- Sei $n \in \mathbb{N}$ beliebig. Wir wählen $z = 0^m 1^m 2^m$ für ein $m > n$
- Sei $u, v, w, x, y \in T^*$ beliebig mit $z = uvwxy$,
und (1) $v \circ x \neq \epsilon$ und (2) $|vwx| \leq n$
- Wir wählen $i = 0$ und zeigen $uw y = uv^iwx^iy \notin L$
- Wegen (2) enthält vwx keine Nullen oder keine Zweien

ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei

- Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v \circ x \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^iwx^iy \notin L) \Rightarrow L \notin \mathcal{L}_2$$

- Sei $n \in \mathbb{N}$ beliebig. Wir wählen $z = 0^m 1^m 2^m$ für ein $m > n$

- Sei $u, v, w, x, y \in T^*$ beliebig mit $z = uvwxy$,
und (1) $v \circ x \neq \epsilon$ und (2) $|vwx| \leq n$

- Wir wählen $i = 0$ und zeigen $uw y = uv^iwx^iy \notin L$

- Wegen (2) enthält vwx keine Nullen oder keine Zweien

- Falls vwx keine Null enthält, dann enthält $uw y$ genau m Nullen aber wegen (1) weniger Einsen und/oder Zweien
- Falls vwx keine Zwei enthält, dann enthält $uw y$ genau m Zweien aber wegen (1) weniger Nullen und/oder Einsen

ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei

- Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v^0 x \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^i wx^i y \notin L) \Rightarrow L \notin \mathcal{L}_2$$

- Sei $n \in \mathbb{N}$ beliebig. Wir wählen $z = 0^m 1^m 2^m$ für ein $m > n$

- Sei $u, v, w, x, y \in T^*$ beliebig mit $z = uvwxy$,
und (1) $v^0 x \neq \epsilon$ und (2) $|vwx| \leq n$

- Wir wählen $i = 0$ und zeigen $uw y = uv^i wx^i y \notin L$

- Wegen (2) enthält vwx keine Nullen oder keine Zweien

- Falls vwx keine Null enthält, dann enthält $uw y$ genau m Nullen
aber wegen (1) weniger Einsen und/oder Zweien

- Falls vwx keine Zwei enthält, dann enthält $uw y$ genau m Zweien
aber wegen (1) weniger Nullen und/oder Einsen

- Damit kann $uw y = uv^0 wx^0 y$ nicht zu L gehören

ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei

- Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v \circ x \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^iwx^iy \notin L) \Rightarrow L \notin \mathcal{L}_2$$

- Sei $n \in \mathbb{N}$ beliebig. Wir wählen $z = 0^m 1^m 2^m$ für ein $m > n$

- Sei $u, v, w, x, y \in T^*$ beliebig mit $z = uvwxy$,
und (1) $v \circ x \neq \epsilon$ und (2) $|vwx| \leq n$

- Wir wählen $i = 0$ und zeigen $uw y = uv^iwx^iy \notin L$

- Wegen (2) enthält vwx keine Nullen oder keine Zweien

- Falls vwx keine Null enthält, dann enthält $uw y$ genau m Nullen
aber wegen (1) weniger Einsen und/oder Zweien

- Falls vwx keine Zwei enthält, dann enthält $uw y$ genau m Zweien
aber wegen (1) weniger Nullen und/oder Einsen

- Damit kann $uw y = uv^0wx^0y$ nicht zu L gehören

- Mit dem Pumping Lemma folgt nun, daß L nicht kontextfrei ist

ANWENDUNGEN DES PUMPING LEMMAS

- $L = \{0^m 1^m 2^m \mid m \in \mathbb{N}\}$ ist nicht kontextfrei

- Verwende Kontraposition des Pumping Lemmas

$$(\forall n \in \mathbb{N}. \exists z \in L. |z| \geq n \wedge \forall u, v, w, x, y \in T^*. (z = uvwxy \wedge v \circ x \neq \epsilon \wedge |vwx| \leq n) \Rightarrow \exists i \in \mathbb{N}. uv^iwx^iy \notin L) \Rightarrow L \notin \mathcal{L}_2$$

- Sei $n \in \mathbb{N}$ beliebig. Wir wählen $z = 0^m 1^m 2^m$ für ein $m > n$
- Sei $u, v, w, x, y \in T^*$ beliebig mit $z = uvwxy$,
und (1) $v \circ x \neq \epsilon$ und (2) $|vwx| \leq n$
- Wir wählen $i = 0$ und zeigen $uw y = uv^iwx^iy \notin L$
- Wegen (2) enthält vwx keine Nullen oder keine Zweien
 - Falls vwx keine Null enthält, dann enthält $uw y$ genau m Nullen aber wegen (1) weniger Einsen und/oder Zweien
 - Falls vwx keine Zwei enthält, dann enthält $uw y$ genau m Zweien aber wegen (1) weniger Nullen und/oder Einsen
- Damit kann $uw y = uv^0wx^0y$ nicht zu L gehören
- Mit dem Pumping Lemma folgt nun, daß L nicht kontextfrei ist

- $L' = \{ww \mid w \in \{0, 1\}^*\} \notin \mathcal{L}_2$

- Ähnliches Argument mit Wörtern der Form $0^m 1^m 0^m 1^m$

RÜCKBLICK: EIGENSCHAFTEN KONTEXTFREIER SPRACHEN

Kontextfreie Sprachen sind deutlich komplizierter

RÜCKBLICK: EIGENSCHAFTEN KONTEXTFREIER SPRACHEN

Kontextfreie Sprachen sind deutlich komplizierter

- **Abschlußeigenschaften**

- Operationen \cup , R , \circ , $*$, σ , h^{-1} erhalten Kontextfreiheit von Sprachen

Kontextfreie Sprachen sind deutlich komplizierter

● **Abschlußeigenschaften**

- Operationen \cup , R , \circ , $*$, σ , h^{-1} erhalten Kontextfreiheit von Sprachen
- Keine Abgeschlossenheit unter \cap , $\bar{}$, $-$

Kontextfreie Sprachen sind deutlich komplizierter

● **Abschlußeigenschaften**

- Operationen \cup , R , \circ , $*$, σ , h^{-1} erhalten Kontextfreiheit von Sprachen
- Keine Abgeschlossenheit unter \cap , $\bar{}$, $-$

● **Automatische Prüfungen**

- Man kann testen ob eine kontextfreie Sprache leer ist
- Man kann testen ob ein Wort zu einer kontextfreien Sprache gehört

Kontextfreie Sprachen sind deutlich komplizierter

● **Abschlußeigenschaften**

- Operationen \cup , R , \circ , $*$, σ , h^{-1} erhalten Kontextfreiheit von Sprachen
- Keine Abgeschlossenheit unter \cap , $\bar{}$, $-$

● **Automatische Prüfungen**

- Man kann testen ob eine kontextfreie Sprache leer ist
 - Man kann testen ob ein Wort zu einer kontextfreien Sprache gehört
 - Man kann **nicht** testen ob zwei kontextfreie Sprachen gleich sind
- Viele wichtige Fragen sind nicht automatisch prüfbar

Kontextfreie Sprachen sind deutlich komplizierter

● **Abschlußeigenschaften**

- Operationen \cup , R , \circ , $*$, σ , h^{-1} erhalten Kontextfreiheit von Sprachen
- Keine Abgeschlossenheit unter \cap , $\bar{}$, $-$

● **Automatische Prüfungen**

- Man kann testen ob eine kontextfreie Sprache leer ist
- Man kann testen ob ein Wort zu einer kontextfreien Sprache gehört
- Man kann **nicht** testen ob zwei kontextfreie Sprachen gleich sind
- Viele wichtige Fragen sind nicht automatisch prüfbar

● **Pumping Lemma**

- Wiederholt man bestimmte Teile ausreichend großer Wörter einer kontextfreien Sprache beliebig oft, so erhält man immer ein Wort der Sprache

Kontextfreie Sprachen sind deutlich komplizierter

● **Abschlußeigenschaften**

- Operationen \cup , R , \circ , $*$, σ , h^{-1} erhalten Kontextfreiheit von Sprachen
- Keine Abgeschlossenheit unter \cap , $\bar{}$, $-$

● **Automatische Prüfungen**

- Man kann testen ob eine kontextfreie Sprache leer ist
- Man kann testen ob ein Wort zu einer kontextfreien Sprache gehört
- Man kann **nicht** testen ob zwei kontextfreie Sprachen gleich sind
- Viele wichtige Fragen sind nicht automatisch prüfbar

● **Pumping Lemma**

- Wiederholt man bestimmte Teile ausreichend großer Wörter einer kontextfreien Sprache beliebig oft, so erhält man immer ein Wort der Sprache
- Konsequenz: viele einfache Sprachen sind nicht kontextfrei
- Für diese sind aufwendigere Mechanismen erforderlich

↪ TI-2