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

- **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 $\overline{L}$
  - 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

Verallgemeinerung von Homomorphismen

- **Abbildung $\boldsymbol{\sigma}$ von Worten in Sprachen**

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

  $\boldsymbol{\sigma(L)}{=}\bigcup\{\sigma(w)\,|\,w\in L\}$ ist das Abbild der Worte von L unter σ

- **Beispiel: $\boldsymbol{\sigma(0)=\{a^nb^n\,|\,n\in\mathbb{N}\},\;\; \sigma(1)=\{aa,bb\}}$**
  - $\sigma{:}\{0,1\}^*{\to}\mathcal{L}$ ist eindeutig definiert durch $\sigma(0)$ und $\sigma(1)$
  - $\sigma(01) = \{a^nb^n\,|\,n\in\mathbb{N}\}\circ\{aa,bb\}$

    $= \{w\in\{a,b\}^*\,|\,w = a^nb^{n+2} \vee w = a^nb^naa$ für ein $n\in\mathbb{N}\}$
  - $\sigma(\{0\}^*) = \{a^nb^n\,|\,n\in\mathbb{N}\}^*$

    $= \{w\in\{a,b\}^*\,|\,w = a^{n_1}b^{n_1}a^{n_2}b^{n_2}..a^{n_k}b^{n_k}$ für ein k und $n_i\in\mathbb{N}\}$

- **Extrem ausdrucksstarker Mechanismus**
  - $\boldsymbol{L_1 \cup L_2 = \sigma(\{1{,}2\})}$ für $\sigma(1){=}L_1$, $\sigma(2){=}L_2$
  - $\boldsymbol{L_1\circ L_2 = \sigma(\{12\})}$ für $\sigma(1){=}L_1$, $\sigma(2){=}L_2$
  - $\boldsymbol{L^* = \sigma(\{1\}^*)}$ für $\sigma(1){=}L$

    ⋮

Abgeschlossenheit unter Substitutionen

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

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

  Sei $P_\sigma = \{A \to \alpha_\sigma \mid A \to \alpha \in P\} \cup \bigcup_{a \in T} P_a$, 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 $w_1..w_n \in L(G_\sigma) \Leftrightarrow S \xrightarrow{*}_{G_\sigma} w_1..w_n$
  $\Leftrightarrow \exists a_1..a_n \in T^*.\ S \xrightarrow{*}_G a_1..a_n \wedge S_{a_i} \xrightarrow{*}_{G_{a_i}} w_i$
  $\Leftrightarrow w_1..w_n \in \sigma(L)$

  Also ist $\sigma(L)$ **kontextfrei**

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

Abschluss unter Spiegelung

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

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

- **Beweis mit Pushdown Automaten**

  Berechnung von h vor Abarbeitung der Worte 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

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

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

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

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

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

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

  Dann gilt $((\boldsymbol{q},\boldsymbol{\epsilon}),\boldsymbol{a},\boldsymbol{X}) \vdash^*_{P_h} ((\boldsymbol{p},\boldsymbol{\epsilon}),\boldsymbol{\epsilon},\boldsymbol{\beta}) \;\Leftrightarrow\; (\boldsymbol{q},\boldsymbol{h(a)},\boldsymbol{X}) \vdash^*_P (\boldsymbol{p},\boldsymbol{\epsilon},\boldsymbol{\beta})$

  Also ist $\boldsymbol{h^{-1}(L) = L(P_h)}$ 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)
  - 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: $S \to AB$, $A \to 0A1$, $A \to 01$, $B \to 2B$, $B \to 2$)

  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 Durchsschnitt folgen

- **Differenz:** $L_1, L_2 \in \mathcal{L}_2 \not\Rightarrow L_1 - L_2 \in \mathcal{L}_2$
  - Es ist $\overline{L} = \Sigma^* - L$
  - Aus Abschluß unter Differenz folgt Abschluß unter Komplement

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)

Trenne Variablen von Terminalsymbolen

- **Grammatik in Chomsky-Normalform**
  - Grammatik $G = (V, T, P, S)$, bei der jede Produktion die Form $A \to B\,C$ oder $A \to a$ hat $(A, B, C \in V,\ a \in T)$
- **Jede kontextfreie Grammatik G mit $\epsilon \notin L(G)$ ist in Chomsky-Normalform transformierbar**
  - Eliminierung unnützer Symbole
  - Eliminierung von ϵ-Produktionen $A \to \epsilon$
  - Eliminierung von Einheitsproduktionen $A \to B$
  - Aufspalten von Produktionen $A \to \alpha$ mit $|\alpha|>2$
  - Separieren von Terminalsymbolen und Variablen in Produktionen

  Aufblähung/Transformationszeit quadratisch relativ zur Größe von G

Unnütze Symbole eliminieren

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

  Erzeugte Produktionenmenge ist äquivalent zu P
- **Eliminationsverfahren für** $\boldsymbol{G}$ **mit** $\boldsymbol{L(G) \neq \emptyset}$
  - Eliminiere nichterzeugende Symbole und Produktionen, die sie enthalten
  - Eliminiere unerreichbare Symbole und Produktionen, die sie enthalten

  Resultierende Grammatik $\boldsymbol{G'}$ **erzeugt dieselbe Sprache wie** $\boldsymbol{G}$

  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
  - Ist $A \to X_1..X_n \in P$ 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 $A \to X_1..X_n \in P$ und A erreichbar dann sind alle X_i erreichbar
  - Verfahren terminiert nach maximal $|V| + |T|$ Iterationen
- **Beispiel:** $\boldsymbol{P = \{\ S \to AB \mid a,\ A \to b\ \}}$
  - 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

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 $A \to \epsilon \in P$ dann ist A eliminierbar
    - · Ist $A \to X_1..X_n \in P$ 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 $A \to \alpha \in P$ mit eliminierbaren Symbolen $X_1, .., X_m$ in α erzeuge 2^m Regeln $A \to \alpha_{i_1,..,i_k}$ (Streiche jeweils die Symbole $X_{i_1}..X_{i_k}$ aus α)
  - Entferne alle Regeln der Form $A \to \epsilon$ (auch neu erzeugte)
  - Wenn S eliminierbar ist, kann $S' \to S$ und $S' \to \epsilon$ ergänzt werden
- **Erzeugte Grammatik ist äquivalent**
  - Zeige $\boldsymbol{A \xrightarrow{*}_{G'} w \;\Leftrightarrow\; A \xrightarrow{*}_{G} w \wedge (w \neq \epsilon \vee A = S')}$ durch Induktion über Länge der Ableitung

Elimination von ϵ-Produktionen am Beispiel

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

- **Ermittelung eliminierbarer Symbole**

  1.: A und B sind eliminierbar

  2.: S ist ebenfalls eliminierbar

- **Verändere Regeln der Grammatik**
  - Aus $S \to AB$ wird $S \to AB \mid A \mid B$
  - Aus $A \to aAA \mid \epsilon$ wird $A \to aAA \mid aA \mid a$
  - Aus $B \to bBB \mid \epsilon$ wird $B \to bBB \mid bB \mid b$

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

- **Ergänze neues Startsymbol**
  - S war eliminierbar: ergänze Produktionen $S' \to S \mid \epsilon$

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

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
    - Ist (A,B) Einheitspaar und $B \to C \in P$ 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
    $\{A \to \alpha \mid B \to \alpha \in P \text{ keine Einheitsproduktion}\}$
- **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

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

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

Nur Produktionen der Form $A \to B\,C$ oder $A \to a$

- **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 $\boldsymbol{X_a}$
  - Ersetze jede Produktion $A \to \alpha$ ($|\alpha| \geq 2$) durch $\boldsymbol{A \to h(\alpha)}$, wobei $h(a) = X_a$
  - Ergänze Produktionen $\boldsymbol{X_a \to a}$ für alle $a \in T$
- **Spalte Produktionen $A \to \alpha$ mit $|\alpha| > 2$**
  - Ersetze jede Produktion $A \to X_1..X_k$ durch $k-1$ Produktionen $A \to X_1 Y_1,\ Y_1 \to X_2 Y_2,\ ...Y_{k-2} \to X_{k-1} X_k$, wobei alle Y_i neue Variablen

Erzeugung der Chomsky-Normalform am Beispiel

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

- **Separiere Terminalsymbole von Variablen**

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

- **Spalte Produktionen $\boldsymbol{A \to \alpha}$ mit $\boldsymbol{|\alpha|>2}$**

$$\begin{aligned} P' = \{ & E \to EY_1 \mid TY_2 \mid X_(Y_3 \mid a \mid b \mid IX_a \mid IX_b \mid IX_0 \mid IX_1 \\ & T \to TY_2 \mid X_(Y_3 \mid a \mid b \mid IX_a \mid IX_b \mid IX_0 \mid IX_1 \\ & F \to X_(Y_3 \mid a \mid b \mid IX_a \mid IX_b \mid IX_0 \mid IX_1 \\ & I \to a \mid b \mid IX_a \mid IX_b \mid IX_0 \mid IX_1 \\ & Y_1 \to X_+T,\ Y_2 \to X_*F,\ Y_3 \to EX_) \\ & X_a \to a,\ X_b \to b,\ X_0 \to 0,\ X_1 \to 1,\ X_+ \to +,\ X_* \to *,\ X_(\to (,\ X_) \to)\ \} \end{aligned}$$

Tests für Eigenschaften kontextfreier Sprachen

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

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

$$j{>}i\text{:}\; V_{i,j} = \{A \in V \mid \exists i{\leq}k{<}j.\ \exists A{\to}BC \in P.\ B \in V_{i,k} \wedge C \in V_{k+1,j}\}$$

$$\begin{array}{|lllll}
V_{1,n} & & & & \\
V_{1,n-1} & V_{2,n} & & & \\
\vdots & \vdots & \vdots & & \\
V_{1,2} & V_{2,3} & \dots & V_{n-1,n} & \\
V_{1,1} & V_{2,2} & \dots & V_{n-1,n-1} & V_{n,n} \\
\hline
w_1 & w_2 & \dots & w_{n-1} & w_n
\end{array}$$

- **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

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

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

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

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 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?

- **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 Worte 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) $v \circ x \neq \epsilon$, (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 $z = z_1..z_m$ mit $|z| \geq n$

– Dann hat jeder Ableitungsbaum für z eine Tiefe von mindestens $|V|+1$

– Sei $X_0, \ldots X_k$ die Folge der verarbeiteten Variablen auf dem längsten Pfad
 Dann erscheint eine Variable zweimal: $X_i = X_j$ für ein $i < j$ mit $k - |V| < i$

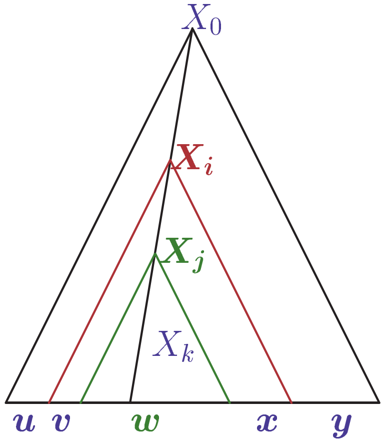
– Seien w und t die aus X_j bzw. X_i abgeleiteten Teilworte

– Dann gilt $t = v\,w\,x$ und $z = u\,t\,y$ für Worte 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**
  - Wir nehmen an L sei kontextfrei
    und zeigen, daß die Aussage des Pumping Lemmas verletzt wird
  - Wähle n entsprechend des Pumping Lemmas und $m>n$
  - Dann kann $z = 0^m 1^m 2^m$ zerlegt werden in $z = u\,v\,w\,x\,y$
    mit (1) $v\circ x \neq \epsilon$ und (2) $|v\,w\,x| \leq n$ und (3) $u\,w\,y = u\,v^0\,w\,x^0\,y \in L$
  - Wegen (2) enthält $v\,w\,x$ keine Nullen oder keine Zweien
  - Falls $v\,w\,x$ keine Null enthält, dann enthält $u\,w\,y$ genau m Nullen
    aber wegen (1) weniger Einsen und/oder Zweien
  - Falls $v\,w\,x$ keine Zwei enthält, dann enthält $u\,w\,y$ genau m Zweien
    aber wegen (1) weniger Nullen und/oder Einsen
  - Damit kann $v\,w\,x$ nicht zu L gehören
  - Dies ist ein Widerspruch, also ist L nicht kontextfrei

- **$L' = \{\, ww \mid w \in \{0,1\}^* \} \notin \mathcal{L}_2$**
  - Ähnliches Argument mit Worten der Form $0^m 1^m 0^m 1^m$

Kontextfreie Sprachen sind deutlich komplizierter

- **Abschlußeigenschaften**
  - Operationen $\cup$, ^{R}, $\circ$, *, σ, h^{-1} erhalten Kontextfreiheit von Sprachen
  - Keine Abgeschlossenheit unter $\cap$, $\overline{}$, -
- **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 Worte 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 $\mapsto$ TI-2