Script generated by TTT

Title: seidl: Theoretische_informatik (21.05.2012)

Date: Mon May 21 10:20:08 CEST 2012

Duration: 83:51 min

Pages: 93

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

$A \rightarrow BC$

$A \rightarrow a$

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Eliminiere alle ϵ-Produktionen.

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Eliminiere alle ϵ-Produktionen.
2. Eliminiere alle Kettenproduktionen.

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Eliminiere alle ϵ-Produktionen.
2. Eliminiere alle Kettenproduktionen.
3. Füge für jedes $a \in \Sigma$, das in einer rechten Seite der Länge ≥ 2 vorkommt, ein neues Nichtterminal A_a zu V hinzu,

$A \to B A_a C$

$A_a \to a$

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Eliminiere alle ϵ-Produktionen.
2. Eliminiere alle Kettenproduktionen.
3. Füge für jedes $a \in \Sigma$, das in einer rechten Seite der Länge ≥ 2 vorkommt, ein neues Nichtterminal A_a zu V hinzu,
   ersetze a in allen rechten Seiten der Länge ≥ 2 durch A_a,

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Eliminiere alle ϵ-Produktionen.
2. Eliminiere alle Kettenproduktionen.
3. Füge für jedes $a \in \Sigma$, das in einer rechten Seite der Länge ≥ 2 vorkommt, ein neues Nichtterminal A_a zu V hinzu,
   ersetze a in allen rechten Seiten der Länge ≥ 2 durch A_a,
   und füge $A_a \to a$ zu P hinzu.

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Eliminiere alle ϵ-Produktionen.
2. Eliminiere alle Kettenproduktionen.
3. Füge für jedes $a \in \Sigma$, das in einer rechten Seite der Länge ≥ 2 vorkommt, ein neues Nichtterminal A_a zu V hinzu,
   ersetze a in allen rechten Seiten der Länge ≥ 2 durch A_a,
   und füge $A_a \to a$ zu P hinzu.
4. Ersetze jede Produktion der Form

   $$A \to B_1 B_2 \cdots B_k \ (k \geq 3)$$

   durch

   $$A \to B_1 C_2,\ C_2 \to B_2 C_3,\ \ldots,\ C_{k-1} \to B_{k-1} B_k$$

   wobei $C_2, \ldots, C_{k-1}$ neue Nichtterminale sind.

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Eliminiere alle ϵ-Produktionen.
2. Eliminiere alle Kettenproduktionen.
3. Füge für jedes $a \in \Sigma$, das in einer rechten Seite der Länge ≥ 2 vorkommt, ein neues Nichtterminal A_a zu V hinzu, ersetze a in allen rechten Seiten der Länge ≥ 2 durch A_a, und füge $A_a \to a$ zu P hinzu.
4. Ersetze jede Produktion der Form

$$A \to B_1 B_2 \cdots B_k \ (k \geq 3)$$

durch

$$A \to B_1 C_2, \ C_2 \to B_2 C_3, \ \ldots, \ C_{k-1} \to B_{k-1} B_k$$

wobei $C_2, \ldots, C_{k-1}$ neue Nichtterminale sind.

Konstruktion einer Chomsky-Normalform

Eingabe: Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$

1. Eliminiere alle ϵ-Produktionen.
2. Eliminiere alle Kettenproduktionen.
3. Füge für jedes $a \in \Sigma$, das in einer rechten Seite der Länge ≥ 2 vorkommt, ein neues Nichtterminal A_a zu V hinzu, ersetze a in allen rechten Seiten der Länge ≥ 2 durch A_a, und füge $A_a \to a$ zu P hinzu.
4. Ersetze jede Produktion der Form

$$A \to B_1 B_2 \cdots B_k \ (k \geq 3)$$

durch

$$A \to B_1 C_2, \ C_2 \to B_2 C_3, \ \ldots, \ C_{k-1} \to B_{k-1} B_k$$

wobei $C_2, \ldots, C_{k-1}$ neue Nichtterminale sind.

Definition 3.30

Eine CFG ist in Greibach-Normalform (benannt nach Sheila Greibach, UCLA), falls jede Produktion von der Form

$$A \to a A_1 \ldots A_n$$

ist.

Definition 3.30

Eine CFG ist in Greibach-Normalform (benannt nach Sheila Greibach, UCLA), falls jede Produktion von der Form

$$A \to a A_1 \ldots A_n$$

ist.

Satz 3.31

Zu jeder CFG G gibt es eine CFG G' in Greibach-Normalform mit $L(G') = L(G) \setminus \{\epsilon\}$.

3.4 Das Pumping-Lemma für kontextfreie Sprachen

Zur Erinnerung: Das Pumping-Lemma für reguläre Sprachen: Für jede reguläre Sprache L gibt es ein $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in $z = uvw$ mit $|uv| \leq n$, $v \neq \epsilon$ und $uv^*w \subseteq L$.

Definition 3.30

Eine CFG ist in Greibach-Normalform (benannt nach Sheila Greibach, UCLA), falls jede Produktion von der Form

$$A \to aA_1 \dots A_n$$

$n \geq 0$

ist.

Satz 3.31

Zu jeder CFG G gibt es eine CFG G' in Greibach-Normalform mit $L(G') = L(G) \setminus \{\epsilon\}$.

3.4 Das Pumping-Lemma für kontextfreie Sprachen

Zur Erinnerung: Das Pumping-Lemma für reguläre Sprachen: Für jede reguläre Sprache L gibt es ein $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in $z = uvw$ mit $|uv| \leq n$, $v \neq \epsilon$ und $uv^*w \subseteq L$.

3.4 Das Pumping-Lemma für kontextfreie Sprachen

Zur Erinnerung: Das Pumping-Lemma für reguläre Sprachen: Für jede reguläre Sprache L gibt es ein $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in $z = uvw$ mit $|uv| \leq n$, $v \neq \epsilon$ und $uv^*w \subseteq L$.

Zum Beweis haben wir $n = |Q|$ gewählt, wobei Q die Zustandsmenge eines L erkennenden DFA war. Das Argument war dann, dass beim Erkennen von z (mindestens) ein Zustand zweimal besucht werden muss und damit der dazwischen liegende Weg im Automaten beliebig oft wiederholt werden kann.

Ein ähnliches Argument kann auch auf kontextfreie Grammatiken angewendet werden.

Satz 3.32 (Pumping-Lemma)

Für jede kontextfreie Sprache L gibt es ein $n \geq 1$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in

$$z = uvwxy,$$

mit

Satz 3.32 (Pumping-Lemma)

Für jede kontextfreie Sprache L gibt es ein $n \geq 1$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in

$$z = uvwxy,$$

mit

- $vx \neq \epsilon$,
- $|vwx| \leq n$, *und*

Satz 3.32 (Pumping-Lemma)

Für jede kontextfreie Sprache L gibt es ein $n \geq 1$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in

$$z = uvwxy,$$

mit

- $vx \neq \epsilon$,
- $|vwx| \leq n$, *und*
- $\forall i \in \mathbb{N}.\ uv^i wx^i y \in L.$

Beweis:

Sei $G = (V, \Sigma, P, S)$ eine CFG in Chomsky-Normalform mit $L(G) = L \setminus \{\epsilon\}$.

Beweis:

Sei $G = (V, \Sigma, P, S)$ eine CFG in Chomsky-Normalform mit $L(G) = L \setminus \{\epsilon\}$. Wähle $n = 2^{|V|}$.

Beweis:

Sei $G = (V, \Sigma, P, S)$ eine CFG in Chomsky-Normalform mit $L(G) = L \setminus \{\epsilon\}$. Wähle $n = 2^{|V|}$. Sei $z \in L(G)$ mit $|z| \geq n$.

Beweis:

Sei $G = (V, \Sigma, P, S)$ eine CFG in Chomsky-Normalform mit $L(G) = L \setminus \{\epsilon\}$. Wähle $n = 2^{|V|}$. Sei $z \in L(G)$ mit $|z| \geq n$.

Jeder Binärbaum mit $\geq 2^k$ Blättern hat die Höhe $\geq k$

Dann hat der Syntaxbaum für z (ohne die letzte Stufe für die Terminale) mindestens einen Pfad der Länge $\geq |V|$, da er wegen der Chomsky-Normalform ein Binärbaum ist:

Beweis:

Sei $G = (V, \Sigma, P, S)$ eine CFG in Chomsky-Normalform mit $L(G) = L \setminus \{\epsilon\}$. Wähle $n = 2^{|V|}$. Sei $z \in L(G)$ mit $|z| \geq n$.

Jeder Binärbaum mit $\geq 2^k$ Blättern hat die Höhe $\geq k$

Dann hat der Syntaxbaum für z (ohne die letzte Stufe für die Terminale) mindestens einen Pfad der Länge $\geq |V|$, da er wegen der Chomsky-Normalform ein Binärbaum ist: Wir betrachten einen solchen Pfad maximaler Länge. Auf diesem Pfad der Länge $\geq |V|$ kommen $\geq |V| + 1$ Nichtterminale vor, also mindestens eins doppelt

Beweis:

Sei $G = (V, \Sigma, P, S)$ eine CFG in Chomsky-Normalform mit $L(G) = L \setminus \{\epsilon\}$. Wähle $n = 2^{|V|}$. Sei $z \in L(G)$ mit $|z| \geq n$.

Jeder Binärbaum mit $\geq 2^k$ *Blättern hat die Höhe* $\geq k$

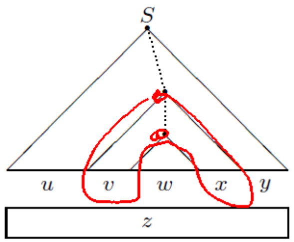
Dann hat der Syntaxbaum für z (ohne die letzte Stufe für die Terminale) mindestens einen Pfad der Länge $\geq |V|$, da er wegen der Chomsky-Normalform ein Binärbaum ist: Wir betrachten einen solchen Pfad maximaler Länge. Auf diesem Pfad der Länge $\geq |V|$ kommen $\geq |V| + 1$ Nichtterminale vor, also mindestens eins doppelt — nennen wir es A.

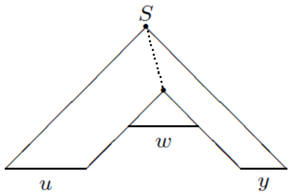
Beweis:

Sei $G = (V, \Sigma, P, S)$ eine CFG in Chomsky-Normalform mit $L(G) = L \setminus \{\epsilon\}$. Wähle $n = 2^{|V|}$. Sei $z \in L(G)$ mit $|z| \geq n$.

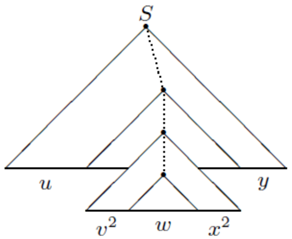
Jeder Binärbaum mit $\geq 2^k$ *Blättern hat die Höhe* $\geq k$

Dann hat der Syntaxbaum für z (ohne die letzte Stufe für die Terminale) mindestens einen Pfad der Länge $\geq |V|$, da er wegen der Chomsky-Normalform ein Binärbaum ist: Wir betrachten einen solchen Pfad maximaler Länge. Auf diesem Pfad der Länge $\geq |V|$ kommen $\geq |V| + 1$ Nichtterminale vor, also mindestens eins doppelt — nennen wir es A. Die zwischen zwei Vorkommen von A liegende Teilableitung kann nun beliebig oft wiederholt werden.





Dieser Ableitungsbaum zeigt $uwy \in L$



Dieser Ableitungsbaum zeigt $uv^2wx^2y \in L$

Beweis (Forts.):

Die Bedingung $vx \neq \epsilon$ folgt, da das obere der beiden A's mit $A \to BC$ abgeleitet werden muss.

Beweis (Forts.):

Die Bedingung $vx \neq \epsilon$ folgt, da das obere der beiden A's mit $A \to BC$ abgeleitet werden muss.

Um $|vwx| \leq n$ zu erreichen, darf das obere A maximal $|V| + 1$ Schritte von einem Blatt entfernt sein. Da der ausgewählte Pfad maximale Länge hat, genügt es, dass das obere A vom Blatt dieses Pfads $|V| + 1$ Schritte entfernt ist. Da es nur $|V|$ Nichtterminale gibt, muss ein solches doppeltes A existieren. □

Beispiel 3.33

Wir zeigen, dass die Sprache

$$L := \{a^i b^i c^i \mid i \in \mathbb{N}\}$$

nicht kontextfrei ist.

Beispiel 3.33

Wir zeigen, dass die Sprache

$$L := \{a^i b^i c^i \mid i \in \mathbb{N}\}$$

nicht kontextfrei ist.

Wäre L kontextfrei, so gäbe es eine dazugehörige Pumping-Lemma Zahl n und wir könnten jedes $z \in L$ mit $|z| \geq n$, insb. $z = a^n b^n c^n$, zerlegen und aufpumpen, ohne aus der Sprache herauszufallen.

Beispiel 3.33

Wir zeigen, dass die Sprache

$$L := \{a^i b^i c^i \mid i \in \mathbb{N}\}$$

nicht kontextfrei ist.

Wäre L kontextfrei, so gäbe es eine dazugehörige Pumping-Lemma Zahl n und wir könnten jedes $z \in L$ mit $|z| \geq n$, insb. $z = a^n b^n c^n$, zerlegen und aufpumpen, ohne aus der Sprache herauszufallen.
Eine Zerlegung $z = u\boxed{vwx}y$ mit $|vwx| \leq n$ bedeutet aber, dass vwx nicht a's und c's enthalten kann.

Beispiel 3.33

Wir zeigen, dass die Sprache

$$L := \{a^i b^i c^i \mid i \in \mathbb{N}\}$$

nicht kontextfrei ist.

Wäre L kontextfrei, so gäbe es eine dazugehörige Pumping-Lemma Zahl n und wir könnten jedes $z \in L$ mit $|z| \geq n$, insb. $z = a^n b^n c^n$, zerlegen und aufpumpen, ohne aus der Sprache herauszufallen. Eine Zerlegung $z = uvwxy$ mit $|vwx| \leq n$ bedeutet aber, dass vwx nicht a's und c's enthalten kann. OE nehmen wir an, vwx enhält nur a's und b's.

Beispiel 3.33

Wir zeigen, dass die Sprache

$$L := \{a^i b^i c^i \mid i \in \mathbb{N}\}$$

nicht kontextfrei ist.

Wäre L kontextfrei, so gäbe es eine dazugehörige Pumping-Lemma Zahl n und wir könnten jedes $z \in L$ mit $|z| \geq n$, insb. $z = a^n b^n c^n$, zerlegen und aufpumpen, ohne aus der Sprache herauszufallen. Eine Zerlegung $z = uvwxy$ mit $|vwx| \leq n$ bedeutet aber, dass vwx nicht a's und c's enthalten kann. OE nehmen wir an, vwx enhält nur a's und b's.
 Da $vx \neq \epsilon$, muss vx mindestens ein a oder ein b enthalten.

Beispiel 3.33

Wir zeigen, dass die Sprache

$$L := \{a^i b^i c^i \mid i \in \mathbb{N}\}$$

nicht kontextfrei ist.

Wäre L kontextfrei, so gäbe es eine dazugehörige Pumping-Lemma Zahl n und wir könnten jedes $z \in L$ mit $|z| \geq n$, insb. $z = a^n b^n c^n$, zerlegen und aufpumpen, ohne aus der Sprache herauszufallen. Eine Zerlegung $z = uvwxy$ mit $|vwx| \leq n$ bedeutet aber, dass vwx nicht a's und c's enthalten kann. OE nehmen wir an, vwx enhält nur a's und b's. Da $vx \neq \epsilon$, muss vx mindestens ein a oder ein b enthalten.
 Damit gilt aber $\#_a(uv^2wx^2y) > \#_c(uv^2wx^2y)$ oder $\#_b(uv^2wx^2y) > \#_c(uv^2wx^2y)$. Also $uv^2wx^2y \notin L$. ↯

Beispiel 3.34

Mit dem Pumping-Lemma kann man zeigen, dass die Sprache $\{ww \mid w \in \{a,b\}^*\}$ nicht kontextfrei ist.

Beispiel 3.34

Mit dem Pumping-Lemma kann man zeigen, dass die Sprache $\{ww \mid w \in \{a,b\}^*\}$ nicht kontextfrei ist. Dies zeigt, dass Kontextbedingungen in Programmiersprachen, etwa „*Deklaration vor Benutzung*“, nicht durch kontextfreie Grammatiken ausgedrückt werden können.

3.5 Abschlusseigenschaften

3.5 Abschlusseigenschaften

Satz 3.35

Seien kontextfreie Grammatiken $G_1 = (V_1, \Sigma_1, P_1, S_1)$ und $G_2 = (V_2, \Sigma_2, P_2, S_2)$ gegeben. Dann kann man in linearer Zeit kontextfreie Grammatiken für

konstruieren.

3.5 Abschlusseigenschaften

Satz 3.35

Seien kontextfreie Grammatiken $G_1 = (V_1, \Sigma_1, P_1, S_1)$ und $G_2 = (V_2, \Sigma_2, P_2, S_2)$ gegeben. Dann kann man in linearer Zeit kontextfreie Grammatiken für

1. $L(G_1) \cup L(G_2)$

konstruieren.

3.5 Abschlusseigenschaften

Satz 3.35

Seien kontextfreie Grammatiken $G_1 = (V_1, \Sigma_1, P_1, S_1)$ *und* $G_2 = (V_2, \Sigma_2, P_2, S_2)$ *gegeben. Dann kann man in linearer Zeit kontextfreie Grammatiken für*

1. $L(G_1) \cup L(G_2)$
2. $L(G_1)L(G_2)$
3. $(L(G_1))^*$

konstruieren.

3.5 Abschlusseigenschaften

Satz 3.35

Seien kontextfreie Grammatiken $G_1 = (V_1, \Sigma_1, P_1, S_1)$ *und* $G_2 = (V_2, \Sigma_2, P_2, S_2)$ *gegeben. Dann kann man in linearer Zeit kontextfreie Grammatiken für*

1. $L(G_1) \cup L(G_2)$
2. $L(G_1)L(G_2)$
3. $(L(G_1))^*$
4. $(L(G_1))^R$

konstruieren.

3.5 Abschlusseigenschaften

Satz 3.35

Seien kontextfreie Grammatiken $G_1 = (V_1, \Sigma_1, P_1, S_1)$ *und* $G_2 = (V_2, \Sigma_2, P_2, S_2)$ *gegeben. Dann kann man in linearer Zeit kontextfreie Grammatiken für*

1. $L(G_1) \cup L(G_2)$
2. $L(G_1)L(G_2)$
3. $(L(G_1))^*$
4. $(L(G_1))^R$

konstruieren.

Die Klasse der kontextfreien Sprachen ist also unter Vereinigung, Konkatenation, Stern und Spiegelung abgeschlossen.

Beweis:

OE nehmen wir an, dass $V_1 \cap V_2 = \emptyset$.

$\square$

3.5 Abschlusseigenschaften

Satz 3.35

Seien kontextfreie Grammatiken $G_1 = (V_1, \Sigma_1, P_1, S_1)$ *und* $G_2 = (V_2, \Sigma_2, P_2, S_2)$ *gegeben. Dann kann man in linearer Zeit kontextfreie Grammatiken für*

1. $L(G_1) \cup L(G_2)$
2. $L(G_1)L(G_2)$
3. $(L(G_1))^*$
4. $(L(G_1))^R$

konstruieren.

Die Klasse der kontextfreien Sprachen ist also unter Vereinigung, Konkatenation, Stern und Spiegelung abgeschlossen.

Beweis:

OE nehmen wir an, dass $V_1 \cap V_2 = \emptyset$.

1. $V = V_1 \cup V_2 \cup \{S\}$; S neu
   $P = P_1 \cup P_2 \cup \{S \to S_1 \mid S_2\}$

$\square$

Beweis:

OE nehmen wir an, dass $V_1 \cap V_2 = \emptyset$.

1. $V = V_1 \cup V_2 \cup \{S\}$; S neu
   $P = P_1 \cup P_2 \cup \{S \to S_1 \mid S_2\}$
2. $V = V_1 \cup V_2 \cup \{S\}$; S neu
   $P = P_1 \cup P_2 \cup \{S \to S_1 S_2\}$

$\square$

Beweis:

OE nehmen wir an, dass $V_1 \cap V_2 = \emptyset$.

1. $V = V_1 \cup V_2 \cup \{S\}$; S neu
   $P = P_1 \cup P_2 \cup \{S \to S_1 \mid S_2\}$
2. $V = V_1 \cup V_2 \cup \{S\}$; S neu
   $P = P_1 \cup P_2 \cup \{S \to S_1 S_2\}$
3. $V = V_1 \cup \{S\}$; S neu
   $P = P_1 \cup \{S \to \epsilon \mid S_1 S\}$

$\square$

Beweis:

OE nehmen wir an, dass $V_1 \cap V_2 = \emptyset$.

1. $V = V_1 \cup V_2 \cup \{S\}$; S neu
   $P = P_1 \cup P_2 \cup \{S \to S_1 \mid S_2\}$
2. $V = V_1 \cup V_2 \cup \{S\}$; S neu
   $P = P_1 \cup P_2 \cup \{S \to S_1 S_2\}$
3. $V = V_1 \cup \{S\}$; S neu
   $P = P_1 \cup \{S \to \epsilon \mid S_1 S\}$
4. $P = \{A \to \alpha^R \mid (A \to \alpha) \in P_1\}$

$\square$

Satz 3.36

Die Menge der kontextfreien Sprachen ist nicht abgeschlossen unter Durchschnitt oder Komplement.

Satz 3.36

Die Menge der kontextfreien Sprachen ist nicht abgeschlossen unter Durchschnitt oder Komplement.

Beweis:

Satz 3.36

Die Menge der kontextfreien Sprachen ist nicht abgeschlossen unter Durchschnitt oder Komplement.

Beweis:

$$L_1 := \{a^i b^i c^j \mid i, j \in \mathbb{N}\} \text{ ist kontextfrei}$$
$$L_2 := \{a^i b^j c^j \mid i, j \in \mathbb{N}\} \text{ ist kontextfrei}$$
$$L_1 \cap L_2 = \{a^i b^i c^i \mid i \in \mathbb{N}\} \text{ ist nicht kontextfrei}$$

Satz 3.36
Die Menge der kontextfreien Sprachen ist nicht abgeschlossen unter Durchschnitt oder Komplement.

Beweis:

$$L_1 := \{a^i b^i c^j \mid i, j \in \mathbb{N}\} \text{ ist kontextfrei}$$
$$L_2 := \{a^i b^j c^j \mid i, j \in \mathbb{N}\} \text{ ist kontextfrei}$$
$$L_1 \cap L_2 = \{a^i b^i c^i \mid i \in \mathbb{N}\} \text{ ist nicht kontextfrei}$$

Wegen de Morgan (1806–1871) können die CFLs daher auch nicht unter Komplement abgeschlossen sein. □

3.6 Algorithmen für kontextfreie Grammatiken

Wie üblich: $G = (V, \Sigma, P, S)$ ist eine CFG.

3.6 Algorithmen für kontextfreie Grammatiken

Wie üblich: $G = (V, \Sigma, P, S)$ ist eine CFG.

Ein Symbol $X \in V \cup \Sigma$ ist

nützlich gdw es eine Ableitung $S \to_G^* w \in \Sigma^*$ gibt, in der X vorkommt.

3.6 Algorithmen für kontextfreie Grammatiken

Wie üblich: $G = (V, \Sigma, P, S)$ ist eine CFG.

Ein Symbol $X \in V \cup \Sigma$ ist

nützlich gdw es eine Ableitung $S \to_G^* w \in \Sigma^*$ gibt, in der X vorkommt.

erzeugend gdw es eine Ableitung $X \to_G^* w \in \Sigma^*$ gibt.

3.6 Algorithmen für kontextfreie Grammatiken

Wie üblich: $G = (V, \Sigma, P, S)$ ist eine CFG.

Ein Symbol $X \in V \cup \Sigma$ ist

- **nützlich** gdw es eine Ableitung $S \to_G^* w \in \Sigma^*$ gibt, in der X vorkommt.
- **erzeugend** gdw es eine Ableitung $X \to_G^* w \in \Sigma^*$ gibt.
- **erreichbar** gdw es eine Ableitung $S \to_G^* \alpha X \beta$ gibt.

3.6 Algorithmen für kontextfreie Grammatiken

Wie üblich: $G = (V, \Sigma, P, S)$ ist eine CFG.

Ein Symbol $X \in V \cup \Sigma$ ist

- **nützlich** gdw es eine Ableitung $S \to_G^* w \in \Sigma^*$ gibt, in der X vorkommt.
- **erzeugend** gdw es eine Ableitung $X \to_G^* w \in \Sigma^*$ gibt.
- **erreichbar** gdw es eine Ableitung $S \to_G^* \alpha X \beta$ gibt.

Fakt 3.37

Nützliche Symbole sind erzeugend und erreichbar.

3.6 Algorithmen für kontextfreie Grammatiken

Wie üblich: $G = (V, \Sigma, P, S)$ ist eine CFG.

Ein Symbol $X \in V \cup \Sigma$ ist

- **nützlich** gdw es eine Ableitung $S \to_G^* w \in \Sigma^*$ gibt, in der X vorkommt.
- **erzeugend** gdw es eine Ableitung $X \to_G^* w \in \Sigma^*$ gibt.
- **erreichbar** gdw es eine Ableitung $S \to_G^* \alpha X \beta$ gibt.

Fakt 3.37

Nützliche Symbole sind erzeugend und erreichbar.

Aber nicht notwendigerweise umgekehrt:

$$S \to AB \mid a, \quad A \to b$$

3.6 Algorithmen für kontextfreie Grammatiken

Wie üblich: $G = (V, \Sigma, P, S)$ ist eine CFG.

Ein Symbol $X \in V \cup \Sigma$ ist

- **nützlich** gdw es eine Ableitung $S \to_G^* w \in \Sigma^*$ gibt, in der X vorkommt.
- **erzeugend** gdw es eine Ableitung $X \to_G^* w \in \Sigma^*$ gibt.
- **erreichbar** gdw es eine Ableitung $S \to_G^* \alpha X \beta$ gibt.

Fakt 3.37

Nützliche Symbole sind erzeugend und erreichbar.

Aber nicht notwendigerweise umgekehrt:

$$S \to AB \mid a, \quad A \to b$$

Ziel: Elimination der unnützen Symbole und der Produktionen, in denen sie vorkommen.

Beispiel 3.38

$$S \to AB \mid a, \quad A \to b$$

1. Elimination nicht erzeugender Symbole:

$$S \to a, \quad A \to b$$

Beispiel 3.38

$$S \to AB \mid a, \quad A \to b$$

1. Elimination nicht erzeugender Symbole:

$$S \to a, \quad A \to b$$

2. Elimination unerreichbarer Symbole:

$$S \to a$$

Beispiel 3.38

$$S \to AB \mid a, \quad A \to b$$

1. Elimination nicht erzeugender Symbole:

$$S \to a, \quad A \to b$$

2. Elimination unerreichbarer Symbole:

$$S \to a$$

Umgekehrte Reihenfolge:

1. Elimination unerreichbarer Symbole:

$$S \to AB \mid a, \quad A \to b$$

2. Elimination nicht erzeugender Symbole:

$$S \to a, \quad A \to b$$

Satz 3.39

Eliminiert man aus einer Grammatik G

1. *alle nicht erzeugenden Symbole, mit Resultat G_1,*
2. *aus G_1 alle unerreichbaren Symbole, mit Resultat G_2,*

Satz 3.39

Eliminiert man aus einer Grammatik G

1. *alle nicht erzeugenden Symbole, mit Resultat G_1,*
2. *aus G_1 alle unerreichbaren Symbole, mit Resultat G_2,*

dann enthält G_2 nur noch nützliche Symbole und $L(G_2) = L(G)$.

Satz 3.39

Eliminiert man aus einer Grammatik G

1. *alle nicht erzeugenden Symbole, mit Resultat G_1,*
2. *aus G_1 alle unerreichbaren Symbole, mit Resultat G_2,*

dann enthält G_2 nur noch nützliche Symbole und $L(G_2) = L(G)$.

Beweis:

Wir zeigen zuerst $L(G_2) = L(G)$.

Satz 3.39

Eliminiert man aus einer Grammatik G

1. *alle nicht erzeugenden Symbole, mit Resultat G_1,*
2. *aus G_1 alle unerreichbaren Symbole, mit Resultat G_2,*

dann enthält G_2 nur noch nützliche Symbole und $L(G_2) = L(G)$.

Beweis:

Wir zeigen zuerst $L(G_2) = L(G)$.

Da $P_2 \subseteq P$ gilt $L(G_2) \subseteq L(G)$.

Satz 3.39

Eliminiert man aus einer Grammatik G

1. *alle nicht erzeugenden Symbole, mit Resultat G_1,*
2. *aus G_1 alle unerreichbaren Symbole, mit Resultat G_2,*

dann enthält G_2 nur noch nützliche Symbole und $L(G_2) = L(G)$.

Beweis:

Wir zeigen zuerst $L(G_2) = L(G)$.

Da $P_2 \subseteq P$ gilt $L(G_2) \subseteq L(G)$.

Umgekehrt, sei $w \in L(G)$, dh $S \rightarrow_G^* w$.

Satz 3.39

Eliminiert man aus einer Grammatik G

1. *alle nicht erzeugenden Symbole, mit Resultat G_1,*
2. *aus G_1 alle unerreichbaren Symbole, mit Resultat G_2,*

dann enthält G_2 nur noch nützliche Symbole und $L(G_2) = L(G)$.

Beweis:

Wir zeigen zuerst $L(G_2) = L(G)$.

Da $P_2 \subseteq P$ gilt $L(G_2) \subseteq L(G)$.

Umgekehrt, sei $w \in L(G)$, dh $S \rightarrow_G^* w$.
Jedes Symbol in dieser Ableitung ist erreichbar und erzeugend.
Also gilt auch $S \rightarrow_{G_2}^* w$, dh $w \in L(G_2)$.

Beweis (Forts.): $[G_1$: erzeugend in G, G_2 : erreichbar in $G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \rightarrow_{G_2}^* \quad \ldots X \ldots \quad \rightarrow_{G_2}^* \quad \ldots \in \Sigma_2^*$$

Beweis (Forts.): $[G_1$: erzeugend in G, G_2 : erreichbar in $G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \rightarrow_{G_2}^* \quad \ldots X \ldots \quad \rightarrow_{G_2}^* \quad \ldots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \rightarrow_{G_1}^* \alpha X \beta$.

Beweis (Forts.): $[G_1$: erzeugend in G, G_2 : erreichbar in $G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \rightarrow_{G_2}^* \quad \ldots X \ldots \quad \rightarrow_{G_2}^* \quad \ldots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \rightarrow_{G_1}^* \alpha X \beta$.
Da alle Symbole in der Ableitung erreichbar sind: $S \rightarrow_{G_2}^* \alpha X \beta$.

Beweis (Forts.): $[G_1 : \text{erzeugend in } G,\ G_2 : \text{erreichbar in } G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \to^*_{G_2} \quad \dots X \dots \quad \to^*_{G_2} \quad \dots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \to^*_{G_1} \alpha X \beta$.
Da alle Symbole in der Ableitung erreichbar sind: $S \to^*_{G_2} \alpha X \beta$.

Alle Symbole in $\alpha X \beta$ müssen in G erzeugend sein:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma^*.\ Y \to^*_G u$

Beweis (Forts.): $[G_1 : \text{erzeugend in } G,\ G_2 : \text{erreichbar in } G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \to^*_{G_2} \quad \dots X \dots \quad \to^*_{G_2} \quad \dots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \to^*_{G_1} \alpha X \beta$.
Da alle Symbole in der Ableitung erreichbar sind: $S \to^*_{G_2} \alpha X \beta$.

Alle Symbole in $\alpha X \beta$ müssen in G erzeugend sein:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma^*.\ Y \to^*_G u$
Da alle Symbole in den Ableitungen $Y \to^*_G u$ erzeugend sind:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma_1^*.\ Y \to^*_{G_1} u$

Beweis (Forts.): $[G_1 : \text{erzeugend in } G,\ G_2 : \text{erreichbar in } G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \to^*_{G_2} \quad \dots X \dots \quad \to^*_{G_2} \quad \dots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \to^*_{G_1} \alpha X \beta$.
Da alle Symbole in der Ableitung erreichbar sind: $S \to^*_{G_2} \alpha X \beta$.

Alle Symbole in $\alpha X \beta$ müssen in G erzeugend sein:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma^*.\ Y \to^*_G u$
Da alle Symbole in den Ableitungen $Y \to^*_G u$ erzeugend sind:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma_1^*.\ Y \to^*_{G_1} u$
Also gibt es $w \in \Sigma_1^*$ mit $\alpha X \beta \to^*_{G_1} w$.

Beweis (Forts.): $[G_1 : \text{erzeugend in } G,\ G_2 : \text{erreichbar in } G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \to^*_{G_2} \quad \dots X \dots \quad \to^*_{G_2} \quad \dots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \to^*_{G_1} \alpha X \beta$.
Da alle Symbole in der Ableitung erreichbar sind: $S \to^*_{G_2} \alpha X \beta$.

Alle Symbole in $\alpha X \beta$ müssen in G erzeugend sein:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma^*.\ Y \to^*_G u$
Da alle Symbole in den Ableitungen $Y \to^*_G u$ erzeugend sind:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma_1^*.\ Y \to^*_{G_1} u$
Also gibt es $w \in \Sigma_1^*$ mit $\alpha X \beta \to^*_{G_1} w$.

Da $S \to^*_{G_1} \alpha X \beta$: Die Ableitung $\alpha X \beta \to^*_{G_1} w$ enthält nur Symbole, die in G_1 erreichbar sind.

Beweis (Forts.): $[G_1 : \text{erzeugend in } G,\ G_2 : \text{erreichbar in } G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \rightarrow^*_{G_2} \quad \ldots X \ldots \quad \rightarrow^*_{G_2} \quad \ldots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \rightarrow^*_{G_1} \alpha X \beta$.
Da alle Symbole in der Ableitung erreichbar sind: $S \rightarrow^*_{G_2} \alpha X \beta$.

Alle Symbole in $\alpha X \beta$ müssen in G erzeugend sein:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma^*.\ Y \rightarrow^*_G u$
Da alle Symbole in den Ableitungen $Y \rightarrow^*_G u$ erzeugend sind:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma_1^*.\ Y \rightarrow^*_{G_1} u$
Also gibt es $w \in \Sigma_1^*$ mit $\alpha X \beta \rightarrow^*_{G_1} w$.

Da $S \rightarrow^*_{G_1} \alpha X \beta$: Die Ableitung $\alpha X \beta \rightarrow^*_{G_1} w$ enthält nur Symbole, die in G_1 erreichbar sind. Daher auch $\alpha X \beta \rightarrow^*_{G_2} w \in \Sigma_2^*$.

Beweis (Forts.): $[G_1 : \text{erzeugend in } G,\ G_2 : \text{erreichbar in } G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \rightarrow^*_{G_2} \quad \ldots X \ldots \quad \rightarrow^*_{G_2} \quad \ldots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \rightarrow^*_{G_1} \alpha X \beta$.
Da alle Symbole in der Ableitung erreichbar sind: $S \rightarrow^*_{G_2} \alpha X \beta$.

Alle Symbole in $\alpha X \beta$ müssen in G erzeugend sein:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma^*.\ Y \rightarrow^*_G u$
Da alle Symbole in den Ableitungen $Y \rightarrow^*_G u$ erzeugend sind:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma_1^*.\ Y \rightarrow^*_{G_1} u$
Also gibt es $w \in \Sigma_1^*$ mit $\alpha X \beta \rightarrow^*_{G_1} w$.

Da $S \rightarrow^*_{G_1} \alpha X \beta$: Die Ableitung $\alpha X \beta \rightarrow^*_{G_1} w$ enthält nur Symbole, die in G_1 erreichbar sind. Daher auch $\alpha X \beta \rightarrow^*_{G_2} w \in \Sigma_2^*$.

$$S \quad \rightarrow^*_{G_2} \quad \alpha X \beta \quad \rightarrow^*_{G_2} \quad w \in \Sigma_2^*$$

□

Beweis (Forts.): $[G_1 : \text{erzeugend in } G,\ G_2 : \text{erreichbar in } G_1]$

Wir zeigen: Alle $X \in V_2 \cup \Sigma_2$ sind nützlich in G_2, dh

$$S \quad \rightarrow^*_{G_2} \quad \ldots X \ldots \quad \rightarrow^*_{G_2} \quad \ldots \in \Sigma_2^*$$

X muss in G_1 erreichbar sein, dh $S \rightarrow^*_{G_1} \alpha X \beta$.
Da alle Symbole in der Ableitung erreichbar sind: $S \rightarrow^*_{G_2} \alpha X \beta$.

Alle Symbole in $\alpha X \beta$ müssen in G erzeugend sein:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma^*.\ Y \rightarrow^*_G u$
Da alle Symbole in den Ableitungen $Y \rightarrow^*_G u$ erzeugend sind:
$\forall Y \in \alpha X \beta.\ \exists u \in \Sigma_1^*.\ Y \rightarrow^*_{G_1} u$
Also gibt es $w \in \Sigma_1^*$ mit $\alpha X \beta \rightarrow^*_{G_1} w$.

Da $S \rightarrow^*_{G_1} \alpha X \beta$: Die Ableitung $\alpha X \beta \rightarrow^*_{G_1} w$ enthält nur Symbole, die in G_1 erreichbar sind. Daher auch $\alpha X \beta \rightarrow^*_{G_2} w \in \Sigma_2^*$.

Satz 3.39
Eliminiert man aus einer Grammatik G

1. *alle nicht erzeugenden Symbole, mit Resultat G_1,*
2. *aus G_1 alle unerreichbaren Symbole, mit Resultat G_2,*

dann enthält G_2 nur noch nützliche Symbole und $L(G_2) = L(G)$.

Beweis:
Wir zeigen zuerst $L(G_2) = L(G)$.

Da $P_2 \subseteq P$ gilt $L(G_2) \subseteq L(G)$.

Umgekehrt, sei $w \in L(G)$, dh $S \rightarrow^*_G w$.
Jedes Symbol in dieser Ableitung ist erreichbar und erzeugend.
Also gilt auch $S \rightarrow^*_{G_2} w$, dh $w \in L(G_2)$.

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

- Alle Symbole in Σ sind erzeugend.

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

- Alle Symbole in Σ sind erzeugend.
- Falls $(A \to \alpha) \in P$ und alle Symbole in α sind erzeugend,

---

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

- Alle Symbole in Σ sind erzeugend.
- Falls $(A \to \alpha) \in P$ und alle Symbole in α sind erzeugend, dann ist auch A erzeugend. $\square$

Beispiel 3.41

$$S \to SAB, \quad A \to BC, \quad B \to C, \quad C \to c$$

Erzeugend: {

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

- Alle Symbole in Σ sind erzeugend.
- Falls $(A \to \alpha) \in P$ und alle Symbole in α sind erzeugend, dann ist auch A erzeugend. $\square$

Beispiel 3.41

$$S \to SAB, \quad A \to BC, \quad B \to C, \quad C \to c$$

Erzeugend: $\{c$

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

- Alle Symbole in Σ sind erzeugend.
- Falls $(A \to \alpha) \in P$ und alle Symbole in α sind erzeugend, dann ist auch A erzeugend. $\square$

Beispiel 3.41

$$S \to SAB, \quad A \to BC, \quad B \to C, \quad C \to c$$

Erzeugend: $\{c, C$

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

- Alle Symbole in Σ sind erzeugend.
- Falls $(A \to \alpha) \in P$ und alle Symbole in α sind erzeugend, dann ist auch A erzeugend. $\square$

Beispiel 3.41

$$S \to SAB, \quad A \to BC, \quad B \to C, \quad C \to c$$

Erzeugend: $\{c, C, B$

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

- Alle Symbole in Σ sind erzeugend.
- Falls $(A \to \alpha) \in P$ und alle Symbole in α sind erzeugend, dann ist auch A erzeugend. $\square$

Beispiel 3.41

$$S \to SAB, \quad A \to BC, \quad B \to C, \quad C \to c$$

Erzeugend: $\{c, C, B, A$

Satz 3.40

Die Menge der erzeugenden Symbole einer CFG sind berechenbar.

Beweis:

Wir berechnen die erzeugenden Symbole induktiv:

- Alle Symbole in Σ sind erzeugend.
- Falls $(A \to \alpha) \in P$ und alle Symbole in α sind erzeugend, dann ist auch A erzeugend. $\square$

Beispiel 3.41

$$S \to SAB, \quad A \to BC, \quad B \to C, \quad C \to c$$

Erzeugend: $\{c, C, B, A\}$.

Korollar 3.42

Für eine kontextfreie Grammatik G ist entscheidbar, ob $L(G) = \emptyset$.

Satz 3.43

Die Menge der erreichbaren Symbole einer CFG ist berechenbar.

Beweis:

Satz 3.43

Die Menge der erreichbaren Symbole einer CFG ist berechenbar.

Beweis:

Wir berechnen die erreichbaren Symbole induktiv:

Satz 3.43

Die Menge der erreichbaren Symbole einer CFG ist berechenbar.

Beweis:

Wir berechnen die erreichbaren Symbole induktiv:

- S ist erreichbar.
- Ist A erreichbar und $(A \to \alpha X \beta) \in P$, so ist auch X erreichbar.

$\square$

Satz 3.44

Das Wortproblem ($w \in L(G)$?) ist für eine CFG G entscheidbar.

Satz 3.44

Das Wortproblem ($w \in L(G)$?) ist für eine CFG G entscheidbar.