Kontextfreie Sprachen: Greibach-Normalform und Abschlusseigenschaften

Prof. Dr. Jasmin Blanchette

Lehr- und Forschungseinheit für
Theoretische Informatik

Stand: 22. Mai 2023
Folien ursprünglich von PD Dr. David Sabel

Links- bzw. rechts-rekursive Produktion

Definition (links- bzw. rechts-rekursive Produktion)

Eine Produktion nennt man **links-rekursiv**, wenn sie von der Form

$$A \to Au$$

ist, und **rechts-rekursiv**, wenn sie von der Form

$$A \to uA$$

ist, wobei in beiden Fällen u eine Satzform ist.

Elimination der Links-Rekursion

Lemma (Elimination der Links-Rekursion)

Sei $G = (V, \Sigma, P, S)$, $P = P' \dot\cup \underbrace{\{A \to Au_1 \mid \cdots \mid Au_n \mid w_1 \mid \cdots \mid w_m\}}_{P''}$ eine CFG mit

- P'' sind alle Produktionen in P mit A als linker Seite und
- die Satzformen $w_1, \ldots, w_m$ beginnen alle nicht mit A.

Elimination der Links-Rekursion

Lemma (Elimination der Links-Rekursion)

Sei $G = (V, \Sigma, P, S)$, $P = P' \dot\cup \underbrace{\{A \to Au_1 \mid \cdots \mid Au_n \mid w_1 \mid \cdots \mid w_m\}}_{P''}$ eine CFG mit

- P'' sind alle Produktionen in P mit A als linker Seite und
- die Satzformen $w_1, \ldots, w_m$ beginnen alle nicht mit A.

Es gilt $L(G') = L(G)$ für $G' = (V \dot\cup \{B\}, \Sigma, P' \cup P''', S)$ mit B neue Variable und

$$\begin{aligned} P''' = \{&A \to w_1 B \mid \cdots \mid w_m B \mid w_1 \mid \cdots \mid w_m, \\ &B \to u_1 \mid \ldots \mid u_n \mid u_1 B \mid \ldots \mid u_n B\} \end{aligned}$$

Beweis: im Skript.

Beispiel

Die CFG $G = (\{A, C\}, \{b, c, d\}, \{A \to ACA \mid bb,\ C \to Ccc \mid d\}, A)$ hat links-rekursive Produktionen.

Beispiel

Die CFG $G = (\{A, C\}, \{b, c, d\}, \{A \to ACA \mid bb,\ C \to Ccc \mid d\}, A)$ hat links-rekursive Produktionen.

Entfernen der Links-Rekursion für A ergibt die CFG

$$G' = (\{A, B, C\}, \{b, c, d\}, \{A \to bbB \mid bb,\ B \to CA \mid CAB,\ C \to Ccc \mid d\}, A)$$

Beispiel

Die CFG $G = (\{A, C\}, \{b, c, d\}, \{A \to ACA \mid bb,\ C \to Ccc \mid d\}, A)$ hat links-rekursive Produktionen.

Entfernen der Links-Rekursion für A ergibt die CFG

$$G' = (\{A, B, C\}, \{b, c, d\}, \{A \to bbB \mid bb,\ B \to CA \mid CAB,\ C \to Ccc \mid d\}, A)$$

Anschließendes Entfernen der Links-Rekursion für C ergibt die CFG

$$G'' = (\{A, B, C, D\}, \{b, c, d\},$$
$$\{A \to bbB \mid bb,\ B \to CA \mid CAB,\ C \to dD \mid d,\ D \to cc \mid ccD\}, A)$$

Greibach-Normalform

Definition (Greibach-Normalform)

Ein CFG $G = (V, \Sigma, P, S)$ ist in *Greibach-Normalform*, falls alle Produktionen in P von der Form $A \to aB_1B_2 \dots B_j$ mit $j \geq 0$, $A, B_1, \dots, B_j \in V$ und $a \in \Sigma$ sind.

Bemerkungen:

- benannt nach Sheila A. Greibach
- Reguläre Grammatiken sind Spezialfall der Greibach-NF:
  Dort ist nur $j = 0$ oder $j = 1$ erlaubt.
- Die Greibach-Normalform wird u.a. verwendet, um zu zeigen, dass kontextfreie Sprachen genau von den nichtdeterministischen Kellerautomaten erkannt werden (später).

Algorithmus 7: Herstellen der Greibach-Normalform

Eingabe: CFG $G = (\{A_1, \ldots, A_n\}, \Sigma, P, A_i)$ in Chomsky-Normalform mit $\varepsilon \notin L(G)$
Ausgabe: CFG G' in Greibach-Normalform mit $L(G) = L(G')$
Beginn

- **für** $i = 1$ *bis* n **tue**
  - **für** $j = 1$ *bis* $i - 1$ **tue**
    - **für alle** $A_i \to A_j u \in P$ **tue**
      - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
      - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \ldots \mid w_m u$ in P;
  - **wenn** $A_i \to A_i u \in P$ **dann**
    - Eliminiere die Regel mit der Operation „Elimination der Links-Rekursion“;
    - Sei B_i die dabei neu erzeugte Variable;
- **für** $i = n - 1$ *bis* 1 **tue**
  - **für alle** $A_i \to A_j u \in P$, $j > i$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \cdots \mid w_m u$ in P;
- **für** $i = 1$ *bis* n **tue**
  - **für alle** $B_i \to A_j u \in P$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $B_i \to A_j u$ durch $B_i \to w_1 u \mid \cdots \mid w_m u$ in P;

Algorithmus 7: Herstellen der Greibach-Normalform

Eingabe: CFG $G = (\{A_1, \ldots, A_n\}, \Sigma, P, A_i)$ in Chomsky-Normalform mit $\varepsilon \notin L(G)$
Ausgabe: CFG G' in Greibach-Normalform mit $L(G) = L(G')$

Ziel der geschachtelten für-Schleife: Es gibt $A_i \to A_j u$ nur für $i < j$

Beginn
- **für** $i = 1$ *bis* n **tue**
  - **für** $j = 1$ *bis* $i - 1$ **tue**
    - **für alle** $A_i \to A_j u \in P$ **tue**
      - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
      - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \ldots \mid w_m u$ in P;
  - **wenn** $A_i \to A_i u \in P$ **dann**
    - Eliminiere die Regel mit der Operation „Elimination der Links-Rekursion“;
    - Sei B_i die dabei neu erzeugte Variable;
- **für** $i = n - 1$ *bis* 1 **tue**
  - **für alle** $A_i \to A_j u \in P$, $j > i$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \cdots \mid w_m u$ in P;
- **für** $i = 1$ *bis* n **tue**
  - **für alle** $B_i \to A_j u \in P$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $B_i \to A_j u$ durch $B_i \to w_1 u \mid \cdots \mid w_m u$ in P;

Algorithmus 7: Herstellen der Greibach-Normalform

Eingabe: CFG $G = (\{A_1, \ldots, A_n\}, \Sigma, P, A_i)$ in Chomsky-Normalform mit $\varepsilon \notin L(G)$
Ausgabe: CFG G' in Greibach-Normalform mit $L(G) = L(G')$

Ziel der geschachtelten für-Schleife: Es gibt $A_i \to A_j u$ nur für $i < j$

Ersetzen der Regeln $A_i \to A_j u$ mit $i > j$

Beginn
- **für** $i = 1$ *bis* n **tue**
  - **für** $j = 1$ *bis* $i - 1$ **tue**
    - **für alle** $A_i \to A_j u \in P$ **tue**
      - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
      - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \ldots \mid w_m u$ in P;
  - **wenn** $A_i \to A_i u \in P$ **dann**
    - Eliminiere die Regel mit der Operation „Elimination der Links-Rekursion“;
    - Sei B_i die dabei neu erzeugte Variable;
- **für** $i = n - 1$ *bis* 1 **tue**
  - **für alle** $A_i \to A_j u \in P$, $j > i$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \cdots \mid w_m u$ in P;
- **für** $i = 1$ *bis* n **tue**
  - **für alle** $B_i \to A_j u \in P$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $B_i \to A_j u$ durch $B_i \to w_1 u \mid \cdots \mid w_m u$ in P;

Algorithmus 7: Herstellen der Greibach-Normalform

Eingabe: CFG $G = (\{A_1, \ldots, A_n\}, \Sigma, P, A_i)$ in Chomsky-Normalform mit $\varepsilon \notin L(G)$
Ausgabe: CFG G' in Greibach-Normalform mit $L(G) = L(G')$
Beginn

Ziel der geschachtelten für-Schleife: Es gibt $A_i \to A_j u$ nur für $i < j$

- **für** $i = 1$ *bis* n **tue**
  - **für** $j = 1$ *bis* $i - 1$ **tue**
    - **für alle** $A_i \to A_j u \in P$ **tue**
      - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
      - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \ldots \mid w_m u$ in P;

    Ersetzen der Regeln $A_i \to A_j u$ mit $i > j$
  - **wenn** $A_i \to A_i u \in P$ **dann**
    - Eliminiere die Regel mit der Operation „Elimination der Links-Rekursion“;
    - Sei B_i die dabei neu erzeugte Variable;

    Ersetzen der Regeln $A_i \to A_i u$
- **für** $i = n - 1$ *bis* 1 **tue**
  - **für alle** $A_i \to A_j u \in P$, $j > i$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \cdots \mid w_m u$ in P;
- **für** $i = 1$ *bis* n **tue**
  - **für alle** $B_i \to A_j u \in P$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $B_i \to A_j u$ durch $B_i \to w_1 u \mid \cdots \mid w_m u$ in P;

Algorithmus 7: Herstellen der Greibach-Normalform

Eingabe: CFG $G = (\{A_1, \ldots, A_n\}, \Sigma, P, A_i)$ in Chomsky-Normalform mit $\varepsilon \notin L(G)$
Ausgabe: CFG G' in Greibach-Normalform mit $L(G) = L(G')$
Beginn

Ziel der geschachtelten für-Schleife: Es gibt $A_i \to A_j u$ nur für $i < j$

für $i = 1$ *bis* n **tue**
  für $j = 1$ *bis* $i - 1$ **tue**
    für alle $A_i \to A_j u \in P$ **tue**
      Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
      Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \ldots \mid w_m u$ in P;

Ersetzen der Regeln $A_i \to A_j u$ mit $i > j$

  wenn $A_i \to A_i u \in P$ **dann**
    Eliminiere die Regel mit der Operation „Elimination der Links-Rekursion";
    Sei B_i die dabei neu erzeugte Variable;

Ersetzen der Regeln $A_i \to A_i u$

Nun gilt für $A_i \to A_j u$ stets $i < j$.
Damit gilt für $A_n \to u$: u beginnt mit Zeichen aus Σ.
Nächste Schleife: Ersetze alle $A_i \to A_j u$ mit $i < j$

für $i = n - 1$ *bis* 1 **tue**
  für alle $A_i \to A_j u \in P$, $j > i$ **tue**
    Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \cdots \mid w_m u$ in P;

für $i = 1$ *bis* n **tue**
  für alle $B_i \to A_j u \in P$ **tue**
    Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    Ersetze $B_i \to A_j u$ durch $B_i \to w_1 u \mid \cdots \mid w_m u$ in P;

Algorithmus 7: Herstellen der Greibach-Normalform

Eingabe: CFG $G = (\{A_1, \ldots, A_n\}, \Sigma, P, A_i)$ in Chomsky-Normalform mit $\varepsilon \notin L(G)$
Ausgabe: CFG G' in Greibach-Normalform mit $L(G) = L(G')$
Beginn

- **für** $i = 1$ *bis* n **tue**
  - **für** $j = 1$ *bis* $i - 1$ **tue**
    - **für alle** $A_i \to A_j u \in P$ **tue**
      - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
      - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \ldots \mid w_m u$ in P;
  - **wenn** $A_i \to A_i u \in P$ **dann**
    - Eliminiere die Regel mit der Operation „Elimination der Links-Rekursion“;
    - Sei B_i die dabei neu erzeugte Variable;
- **für** $i = n - 1$ *bis* 1 **tue**
  - **für alle** $A_i \to A_j u \in P$, $j > i$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \cdots \mid w_m u$ in P;
- **für** $i = 1$ *bis* n **tue**
  - **für alle** $B_i \to A_j u \in P$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $B_i \to A_j u$ durch $B_i \to w_1 u \mid \cdots \mid w_m u$ in P;

Ziel der geschachtelten für-Schleife: Es gibt $A_i \to A_j u$ nur für $i < j$

Ersetzen der Regeln $A_i \to A_j u$ mit $i > j$

Ersetzen der Regeln $A_i \to A_i u$

Nun gilt für $A_i \to A_j u$ stets $i < j$.
Damit gilt für $A_n \to u$: u beginnt mit Zeichen aus Σ.
Nächste Schleife: Ersetze alle $A_i \to A_j u$ mit $i < j$

$w_1, \ldots, w_m$ fangen mit Zeichen aus Σ an, da Schleife absteigend läuft.

Algorithmus 7: Herstellen der Greibach-Normalform

Eingabe: CFG $G = (\{A_1, \ldots, A_n\}, \Sigma, P, A_i)$ in Chomsky-Normalform mit $\varepsilon \notin L(G)$
Ausgabe: CFG G' in Greibach-Normalform mit $L(G) = L(G')$
Beginn

> Ziel der geschachtelten für-Schleife: Es gibt $A_i \to A_j u$ nur für $i < j$

- **für** $i = 1$ *bis* n **tue**
  - **für** $j = 1$ *bis* $i - 1$ **tue**
    - **für alle** $A_i \to A_j u \in P$ **tue**
      - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
      - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \ldots \mid w_m u$ in P;

    > Ersetzen der Regeln $A_i \to A_j u$ mit $i > j$

  - **wenn** $A_i \to A_i u \in P$ **dann**
    - Eliminiere die Regel mit der Operation „Elimination der Links-Rekursion“;
    - Sei B_i die dabei neu erzeugte Variable;

    > Ersetzen der Regeln $A_i \to A_i u$

> Nun gilt für $A_i \to A_j u$ stets $i < j$.
> Damit gilt für $A_n \to u$: u beginnt mit Zeichen aus Σ.
> Nächste Schleife: Ersetze alle $A_i \to A_j u$ mit $i < j$

- **für** $i = n - 1$ *bis* 1 **tue**
  - **für alle** $A_i \to A_j u \in P$, $j > i$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $A_i \to A_j u$ durch $A_i \to w_1 u \mid \cdots \mid w_m u$ in P;

  > $w_1, \ldots, w_m$ fangen mit Zeichen aus Σ an, da Schleife absteigend läuft.

- **für** $i = 1$ *bis* n **tue**
  - **für alle** $B_i \to A_j u \in P$ **tue**
    - Seien $A_j \to w_1 \mid \cdots \mid w_m$ alle Regeln in P mit A_j als linker Seite;
    - Ersetze $B_i \to A_j u$ durch $B_i \to w_1 u \mid \cdots \mid w_m u$ in P;

  > Behandle die neuen Regeln mit B_i als linker Seite.

Korrektheit

Satz

Zu jeder CFG G mit $\varepsilon \notin L(G)$ gibt es eine CFG G' in Greibach-Normalform, sodass $L(G) = L(G')$ gilt.

Korrektheit

Satz

Zu jeder CFG G mit $\varepsilon \notin L(G)$ gibt es eine CFG G' in Greibach-Normalform, sodass $L(G) = L(G')$ gilt.

Korrektheit folgt durch:

- Prüfen der genannten Invarianten
- Korrektheit der Elimination der Links-Rekursion
- Korrektheit der Operation „Inlining von Produktionen".

Korrektheit

Satz

Zu jeder CFG G mit $\varepsilon \notin L(G)$ gibt es eine CFG G' in Greibach-Normalform, sodass $L(G) = L(G')$ gilt.

Korrektheit folgt durch:

- Prüfen der genannten Invarianten
- Korrektheit der Elimination der Links-Rekursion
- Korrektheit der Operation „Inlining von Produktionen".

Ein Beispiel zur Erstellung der Greibach-Normalform ist im Skript.

Abschlusseigenschaften der kontextfreien Sprachen

Theorem

Die kontextfreien Sprachen sind abgeschlossen bezüglich Vereinigung, Produkt und Kleeneschem Abschluss.

Abschlusseigenschaften der kontextfreien Sprachen

Theorem

Die kontextfreien Sprachen sind abgeschlossen bezüglich Vereinigung, Produkt und Kleeneschem Abschluss.

Beweis:

- Seien L_1,L_2 CFLs und $G_i = (V_i, \Sigma_i, P_i, S_i)$ CFGs mit $L(G_i) = L_i$ für $i = 1, 2$. O.B.d.A. sei $V_1 \cap V_2 = \emptyset$.
- Seien S, S' neue Variablen (d.h. $\{S, S'\} \cap (V_1 \cup V_2) = \emptyset$).

Abschlusseigenschaften der kontextfreien Sprachen

Theorem

Die kontextfreien Sprachen sind abgeschlossen bezüglich Vereinigung, Produkt und Kleeneschem Abschluss.

Beweis:

- Seien L_1,L_2 CFLs und $G_i = (V_i, \Sigma_i, P_i, S_i)$ CFGs mit $L(G_i) = L_i$ für $i = 1, 2$. O.B.d.A. sei $V_1 \cap V_2 = \emptyset$.
- Seien S, S' neue Variablen (d.h. $\{S, S'\} \cap (V_1 \cup V_2) = \emptyset$).

Vereinigung: Sei

$$G_\cup = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 \mid S_2\}, S)$$

Dann gilt: $L(G_\cup) = L(G_1) \cup L(G_2) = L_1 \cup L_2$.

Abschlusseigenschaften der kontextfreien Sprachen (2)

Beweis (Fortsetzung):

Produkt: Sei

$$G_\circ = (V_1 \cup V_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 S_2\}, S)$$

Dann gilt $L(G_\circ) = L(G_1)L(G_2) = L_1 L_2$.

Abschlusseigenschaften der kontextfreien Sprachen (3)

Beweis (Fortsetzung):

Kleenescher Abschluss: Sei

$$G_* = (V_1 \cup \{S', S\}, \Sigma, P', S')$$

mit

$$P' = (P \setminus \{S_1 \to \varepsilon\}) \cup \{S' \to \varepsilon, S' \to S, S \to SS, S \to S_1\}$$

Dann gilt $L(G_*) = L(G_1)^*$. □

Abschlusseigenschaften der kontextfreien Sprachen (4)

Theorem

Die kontextfreien Sprachen sind **nicht abgeschlossen** unter Schnitt- und Komplementbildung.

Einschub: Eine nichtkontextfreie Sprache

Satz

Die Sprache $L = \{a^i b^i c^i \mid i \in \mathbb{N}\}$ ist nicht kontextfrei.

Ein Beweis hierfür sehen wir später mit dem Pumping-Lemma
für kontextfreie Sprachen.

Wir verwenden die Sprache und die Eigenschaft jedoch im Folgenden.

Abschlusseigenschaften der kontextfreien Sprachen (5)

Theorem

Die kontextfreien Sprachen sind **nicht abgeschlossen** unter Schnitt- und Komplementbildung.

Abschlusseigenschaften der kontextfreien Sprachen (5)

Theorem

Die kontextfreien Sprachen sind **nicht abgeschlossen** unter Schnitt- und Komplementbildung.

Beweis:

Schnittbildung:

- Sei $L_1 = \{a^n b^m c^m \mid m, n \in \mathbb{N}\}$ und sei $L_2 = \{a^m b^m c^n \mid m, n \in \mathbb{N}\}$
- $G_1 = (\{A, D, S\}, \{a, b, c\}, \{S \to AD, A \to \varepsilon \mid aA, D \to bDc \mid \varepsilon\}, S)$
  $G_2 = (\{C, D, S\}, \{a, b, c\}, \{S \to DC, C \to cC \mid \varepsilon, D \to aDb \mid \varepsilon\}, S)$
  Für i = 1, 2: $L(G_i) = L_i$, daher: L_1 und L_2 sind beide kontextfrei.
- $L_1 \cap L_2 = \{a^n b^n c^n \mid n \in \mathbb{N}_{>0}\}$ ist nicht kontextfrei.
- Daher sind die CFLs nicht abgeschlossen bezüglich Schnittbildung.

Abschlusseigenschaften der kontextfreien Sprachen (6)

Beweis (Fortsetzung):

Komplement:

- Beweis durch Widerspruch: Wir nehmen an, es gilt
$$L \text{ ist CFL} \implies \overline{L} \text{ ist CFL}$$
- Seien L_1, L_2 CFLs. Dann ist auch $\overline{\overline{L_1} \cup \overline{L_2}}$ CFL (da CFLs abgeschlossen bezüglich $\overline{}$ und $\cup$).
- Aber: $\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2$. Widerspruch. □