Kontextfreie Sprachen

Hans U. Simon (RUB)

Email: simon@lmi.rub.de

Homepage: http://www.ruhr–uni–bochum.de/lmi

Bekannte Beispiele

Die Regeln

$$E \to T \mid E + T \ , \ T \to F \mid T * F \ , \ F \to a \mid (E)$$

erzeugen die korrekt geklammerten arithmetischen Ausdrücke.

Die Regeln

$$S \to ab \mid aSb$$

erzeugen die einfache Klammersprache

$$L = \{a^n b^n \mid n \geq 1\} \ .$$

Kontextfreie Regeln und Rekursion

Die Variablen in kontextfreien Regeln repräsentieren rekursiv definierbare Konzepte. Zum Beipiel:

- Lies Regel $E \rightarrow T \mid E + T$ wie folgt: Ein Expression ist ein Term oder die Summe aus einem Expression und einem Term.
- Lies Regel $S \rightarrow ab \mid aSb$ wie folgt: ein Mitglied der einfachen Klammersprache hat die Form ab oder ist ein von a und b eingeklammertes Mitglied der einfachen Klammersprache.

Kontextfreie Regeln und Programmiersprachen

Programmiersprachen enthalten

- eine Vielzahl rekursiv definierbarer Konzepte
- und diverse Klammerstrukturen

Betrachte zum Beispiel folgendes Fragment einer kontextfreien Grammatik für MODULA:

$$\begin{array}{rcl}
\langle \text{Anw} \rangle & \rightarrow & \langle \text{While–Anw} \rangle \mid \langle \text{If–Anw} \rangle \\
\langle \text{While–Anw} \rangle & \rightarrow & \text{WHILE } \langle \text{Bedingung} \rangle \text{ DO } \langle \text{Anw} \rangle \text{ END} \\
\langle \text{If–Anw} \rangle & \rightarrow & \text{IF } \langle \text{Bedingung} \rangle \text{ THEN } \langle \text{Anw} \rangle \text{ END} \\
\langle \text{Bedingung} \rangle & \rightarrow & \cdots
\end{array}$$

Erläuterungen dazu in der Vorlesung!

Chomsky Normalform (CNF)

Definition: Eine kontextfreie Grammatik ist in Chomsky Normalform, wenn sie nur Regeln der Form

$$A \rightarrow BC \ , A \rightarrow a \text{ mit } A, B, C \in V, a \in \Sigma$$

besitzt.

Effekt: Binäre Syntaxbäume !

Transformation in Chomsky Normalform

Satz: Jede kontextfreie Grammatik G mit $\varepsilon \notin L(G)$ kann in eine äquivalente kontextfreie Grammatik G' in Chomsky Normalform transformiert werden.

Der Beweis erfolgt in 4 Phasen, die folgendes erreichen:

1. Die rechte Seite einer Regel besteht entweder nur aus einem Terminalzeichen oder aus einem String, der nur Variable (aber keine Terminalzeichen) enthält. Die Grammatik heißt dann separiert.
2. Die rechte Seite einer Regel hat maximal die Länge 2.
3. Es gibt keine ε–Regeln mehr.
4. Es gibt keine „Kettenregeln“ (der Form $A \to B$) mehr.

Danach ist die Grammatik in Chomsky Normalform.

Ziel 1: Separierte Grammatik

- Führe für jedes Terminalzeichen a eine neue Variable X_a ein sowie die neuen Regeln $X_a \rightarrow a$.
  (Jedes Terminalzeichen bekommt eine „große Schwester“ in V.)
- Bei einer Regel, deren rechte Seite nicht nur aus einem Terminalzeichen bzw. einem String über V besteht, wird jedes Terminalzeichen a durch seine „große Schwester“ $X_a \in V$ ersetzt.

Danach ist die Grammatik separiert (ohne dass die Sprache abgeändert wurde).

Ziel 2: Verkürzung rechter Seiten von Regeln

Ersetze eine Regel R der Form

$$A \rightarrow B_1 B_2 \cdots B_k,\ \ k \geq 3$$

unter Verwendung neuer Variablen $R_1, \ldots, R_{k-2}$ durch das Regelsystem

$$A \rightarrow B_1 R_1,\ R_1 \rightarrow B_2 R_2,\ \cdots\ ,\ R_{k-3} \rightarrow B_{k-2} R_{k-2},\ R_{k-2} \rightarrow B_{k-1} B_k\ \ .$$

Danach hat jede rechte Seite einer Regel maximal die Länge 2 (ohne dass die Sprache abgeändert wurde).

Ziel 3: Elimination von ε–Regeln

- Bestimme (nach dem früher einmal besprochenen Schema) die Menge V_ε aller Variablen, aus denen ε ableitbar ist.
- Für jede Regel der Form $A \rightarrow BC$ mache folgendes:
  - Falls $B \in V_\varepsilon$, dann kreiere die zusätzliche Regel $A \rightarrow C$.
  - Falls $C \in V_\varepsilon$, dann kreiere die zusätzliche Regel $A \rightarrow B$.
- Eliminiere alle ε–Regeln.

Danach enthält die Grammatik keine ε–Regeln mehr (ohne das die Sprache abgeändert wurde).

Ziel 4: Elimination von Kettenregeln

- Betrachte die „Kettenregel–Relation“ K bestehend aus allen Paaren (A, B) zu denen eine Kettenregel $A \rightarrow B$ existiert und berechne ihre transitive Hülle K^+.
- Eliminiere alle Kettenregeln.
- Falls $(A, C) \in K^+$, dann kreiere für jede C–Regel $C \rightarrow \beta$ die zusätzliche Regel $A \rightarrow \beta$.
  A „erbt“ gewissermaßen alle rechten Seiten von C.

Danach enthält die Grammatik keine Kettenregeln mehr. Durch den Trick mit der „Vererbung“ rechter Seiten wurden die Kettenregeln überflüssig gemacht. Die Sprache wird durch die beschriebenen Transformationsschritte nicht abgeändert.

Details zur Berechnung der transitiven Hülle

Bemerkungen:

- $(A, C) \in K^+$ **gdw** A und C sind durch eine „Kette von Kettenregeln“ der Form

$$A \to B_1 \to B_2 \to \cdots \to B_{k-1} \to B_k \to C$$

(inklusive des Grenzfalles $A \to C$) miteinander verbunden.

- Berechnung von K^+ entspricht algorithmisch der Berechnung der transitiven Hülle in dem „Kettenregel–Hilfsgraphen“ (Algorithmus von Warshall).

Beispiel

Die kontextfreie Grammatik mit den Regeln

$$S \rightarrow aOb\ ,\ O \rightarrow P \mid OO \mid aOb\ ,\ P \rightarrow x \mid E\ ,\ E \rightarrow \varepsilon$$

wird in Chomsky Normalform gebracht wie folgt:

1. Mit Hilfe der neuen Variablen A, B (die „großen Schwestern“ von a, b) erhalten wir die separierte Grammatik

$$S \rightarrow AOB\ ,\ O \rightarrow P \mid OO \mid AOB\ ,\ P \rightarrow x \mid E\ ,\ E \rightarrow \varepsilon\ , A \rightarrow a\ ,\ B \rightarrow b\ .$$

2. Mit Hilfe der neuen Variablen S', O' vermeiden wir zu lange rechte Seiten, indem die Regeln $S \rightarrow AOB$ bzw. $O \rightarrow AOB$ ersetzt werden durch

$$S \rightarrow AS'\ ,\ S' \rightarrow OB \text{ bzw. } O \rightarrow AO'\ ,\ O' \rightarrow OB\ .$$

Beispiel (fortgesetzt)

Zwischenergebnis nach Phase 2:

$$S \to AS' \;,\; S' \to OB \;,\; O \to P \mid OO \mid AO' \;,\; O' \to OB$$

$$P \to x \mid E \;,\; E \to \varepsilon \;,\; A \to a \;,\; B \to b$$

3. $V_\varepsilon = \{E, P, O\}$. Die Technik zur Elimination von ε–Regeln führt zu

$$S \to AS' \;,\; S' \to OB \mid B \;,\; O \to P \mid OO \mid O \mid AO' \;,\; O' \to OB \mid B \;,$$

$$P \to x \mid E \;, A \to a \;,\; B \to b \;.$$

4. Es gilt $K^+ = \{(S', B), (O, P), (O, O), (O', B), (P, E), (O, E)\}$. Die Technik zur Elimination von Kettenregeln führt zu

$$S \to AS' \;,\; S' \to OB \mid b \;,\; O \to OO \mid AO' \mid x \;,\; O' \to OB \mid b \;,$$

$$P \to x \;,\; A \to a \;,\; B \to b \;.$$

Greibach Normalform (GNF)

Definition: Eine kontextfreie Grammatik ist in Greibach Normalform, wenn sie nur Regeln der Form

$$A \rightarrow aB_1 \cdots B_k \ , \ A \rightarrow a \text{ mit } k \geq 1, A, B_1, \ldots, B_k \in V, a \in \Sigma$$

besitzt (einzelnes Terminalzeichen gefolgt von einem String aus Variablen).

Bemerkung:

- Die Einschränkung auf $k = 1$ würde die regulären Grammatiken liefern.
- Greibach Normalform führt später zu Kellerautomaten, die in „Realzeit“ arbeiten (in jedem Schritt ein Eingabesymbol verarbeiten).

Transformation in Greibach Normalform

Satz: Jede kontextfreie Grammatik G mit $\varepsilon \notin L(G)$ kann in eine äquivalente kontextfreie Grammatik G' in Greibach Normalform transformiert werden.

Der Beweis geht **oBdA** davon aus, dass G separiert ist und keine ε–Regeln besitzt (was zum Beispiel bei Chomsky Normalform der Fall ist). Er besteht aus vier Teilen:

1. Technik zur Vermeidung von „Linksrekursion“
2. Erzwingen einer „Monotonie–Eigenschaft“ unter Verwendung von Hilfsvariablen
3. Transformation in Greibach–Normalform abgesehen von B–Regeln für eine Hilfsvariable B
4. Transformation der B–Regeln in Greibach Normalform für jede Hilfsvariable B

Eine Invarianzbedingung

Im Beweis wird das Regelsystem P dynamisch modifiziert werden. Dabei wird es jedoch stets nur Regeln enthalten, deren rechte Seiten

- zu V^+ gehören (String aus Variablen)
- oder der GNF genügen (einzelnes Terminalzeichen gefolgt von einem String aus Variablen).

Bemerkungen:

1. Anfangs ist dies der Fall, da wir von einer separierten kontextfreien Grammatik ohne ε–Regeln ausgehen.
2. Dass die Eigenschaft erhalten bleibt, könnte im Prinzip imduktiv gezeigt werden.

Es foglt eine Beschreibung der vier Beweisteile.

Vermeidung von Linksrekursion

Die A–Regeln

$$A \to \overbrace{A\alpha_1 \mid \cdots \mid A\alpha_k}^{linksrekursiv} \mid \beta_1 \mid \cdots \mid \beta_l$$

erzeugen gerade die durch folgenden regulären Ausdruck beschreibbaren Satzformen:

$$(\beta_1|\cdots|\beta_l) \cdot (\alpha_1|\cdots|\alpha_k)^*$$

Diese Satzformen lassen sich genau so gut (ohne Linksrekursion) erzeugen durch

$$\begin{aligned} A &\to \beta_1 \mid \cdots \mid \beta_l \mid \beta_1 B \mid \cdots \mid \beta_l B \\ B &\to \alpha_1 \mid \cdots \mid \alpha_k \mid \alpha_1 B \mid \cdots \mid \alpha_k B \end{aligned}$$

B ist dabei eine neue Variable.

Erzwingen einer Monotonie–Eigenschaft

Ziel:

Für $V = \{A_1, \ldots, A_m\}$ und Regeln der Form $A_i \rightarrow A_j\alpha$ erzwinge, dass $i < j$.

Dies leistet folgende „Bootstrapping“–Methode:

FOR $i := 1$ TO m DO
    FOR $j := 1$ TO $i - 1$ DO
        FOR all $A_i \rightarrow A_j\alpha \in P$ DO
            Seien $A_j \rightarrow \beta_1|\cdots|\beta_k$ alle A_j–Regeln
            Streiche $A_i \rightarrow A_j\alpha$ aus P
            Füge $A_i \rightarrow \beta_1\alpha|\cdots|\beta_k\alpha$ hinzu
        ENDFOR
    ENDFOR
    Vermeide linksrekursive Regeln der Form $A_i \rightarrow A_i\alpha$ (sofern vorhanden)
    mit der besprochenen Technik unter Einsatz der Hilfsvariable B_i
ENDFOR

Greibach Normalform für A_i–Regeln

Die Monotonie–Eigenschaft impliziert, dass die rechten Seiten der A_m–Regeln mit einem Terminalzeichen beginnen. Folgende Methode erreicht diese Eigenschaft iterativ auch für die Variablen $A_{m-1}, \ldots, A_1$:

```
FOR i := m − 1 DOWNTO 1 DO
    FOR j := i + 1 TO m DO
        FOR all A_i → A_j α ∈ P DO
            Seien A_j → β_1 | ··· | β_k alle A_j–Regeln
            Streiche A_i → A_j α aus P
            Füge A_i → β_1 α | ··· | β_k α hinzu
        ENDFOR
    ENDFOR
ENDFOR
```

Greibach Normalform für B_i–Regeln

Die rechte Seite einer B_i–Regel beginnt

- mit einem Terminalzeichen (gut)
- oder einer Variable A_j (weniger gut)

Da alle A_j–Regeln bereits der Greibach–Normalform genügen, ist die folgende Maßnahme nun ausreichend:

FOR $i := 1$ TO m DO
    FOR $j := 1$ TO m DO
        FOR all $B_i \rightarrow A_j\alpha \in P$ DO
            Seien $A_j \rightarrow \beta_1|\cdots|\beta_k$ alle A_j–Regeln
            Streiche $B_i \rightarrow A_j\alpha$ aus P
            Füge $B_i \rightarrow \beta_1\alpha|\cdots|\beta_k\alpha$ hinzu
        ENDFOR
    ENDFOR
ENDFOR

Beispiel

Aufgabe: Bringe die durch

$$A_1 \rightarrow A_2A_3 \ , \ A_2 \rightarrow A_3A_1 \mid b \ , \ A_3 \rightarrow A_1A_2 \mid a$$

gegebene Grammatik in Greibach Normalform.

Beim Erzwingen der Monotonie–Eigenschaft wüsssen nur die A_3–Regeln modifiziert werden. Es entstehen folgende Zwischenergebnisse:

$$\begin{array}{lcl}
(1)\ A_3 & \rightarrow & A_2A_3A_2 \mid a \\
(2)\ A_3 & \rightarrow & A_3A_1A_3A_2 \mid bA_3A_2 \mid a \\
(3)\ A_3 & \rightarrow & bA_3A_2 \mid a \mid bA_3A_2B_3 \mid aB_3 \\
(3)\ B_3 & \rightarrow & A_1A_3A_2 \mid A_1A_3A_2B_3
\end{array}$$

Beim Übergang von (2) nach (3) wurde die Technik zur Vermeidung von Linksrekursion eingesetzt.

Beispiel (fortgesetzt)

$$
\begin{array}{lcl}
A_1 & \rightarrow & A_2 A_3 \\
A_2 & \rightarrow & A_3 A_1 \mid b \\
A_3 & \rightarrow & bA_3A_2 \mid a \mid bA_3A_2B_3 \mid aB_3 \\
B_3 & \rightarrow & A_1A_3A_2 \mid A_1A_3A_2B_3
\end{array}
$$

hat bereits A_3–Regeln in GNF. Die Technik zur Vererbung dieser Eigenschaft auf die Variablen A_2, A_1 liefert:

$$
\begin{array}{lcl}
A_3 & \rightarrow & bA_3A_2 \mid a \mid bA_3A_2B_3 \mid aB_3 \\
A_2 & \rightarrow & bA_3A_2A_1 \mid aA_1 \mid bA_3A_2B_3A_1 \mid aB_3A_1 \mid b \\
A_1 & \rightarrow & bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3A_2B_3A_1A_3 \mid aB_3A_1A_3 \mid bA_3 \\
B_3 & \rightarrow & A_1A_3A_2 \mid A_1A_3A_2B_3
\end{array}
$$

Beispiel (fortgesetzt)

$$
\begin{array}{lcl}
A_3 & \to & bA_3A_2 \mid a \mid bA_3A_2B_3 \mid aB_3 \\
A_2 & \to & bA_3A_2A_1 \mid aA_1 \mid bA_3A_2B_3A_1 \mid aB_3A_1 \mid b \\
A_1 & \to & bA_3A_2A_1A_3 \mid aA_1A_3 \mid bA_3A_2B_3A_1A_3 \mid aB_3A_1A_3 \mid bA_3 \\
B_3 & \to & A_1A_3A_2 \mid A_1A_3A_2B_3
\end{array}
$$

hat bereits A_i–Regeln in GNF. Die Technik zur Vererbung dieser Eigenschaft auf die B_i–Regeln liefert dieselben A_i–Regeln und die folgenden B_3–Regeln:

$$
\begin{array}{lcl}
B_3 & \to & bA_3A_2A_1A_3A_3A_2 \mid aA_1A_3A_3A_2 \mid bA_3A_2B_3A_1A_3A_3A_2 \\
 & & \mid aB_3A_1A_3A_3A_2 \mid bA_3A_3A_2 \\
 & & \mid bA_3A_2A_1A_3A_3A_2B_3 \mid aA_1A_3A_3A_2B_3 \mid bA_3A_2B_3A_1A_3A_3A_2B_3 \\
 & & \mid aB_3A_1A_3A_3A_2B_3 \mid bA_3A_3A_2B_3
\end{array}
$$