Grundlagen der Theoretischen Informatik

4. Kellerautomaten und kontextfreie Sprachen (II)

18.05.2017

Viorica Sofronie-Stokkermans

e-mail: sofronie@uni-koblenz.de

Übersicht

1. Motivation
2. Terminologie
3. Endliche Automaten und reguläre Sprachen
4. Kellerautomaten und kontextfreie Sprachen
5. Turingmaschinen und rekursiv aufzählbare Sprachen
6. Berechenbarkeit, (Un-)Entscheidbarkeit
7. Komplexitätsklassen P und NP

Zur Erinnerung: kontextfreie Grammatiken

Definition (Kontextfreie Grammatik)
Eine Grammatik $G = (V, T, R, S)$ heißt **kontextfrei** gdw

$$\forall (P \to Q) \in R \ \left(P \in V \text{ und } Q \in (V \cup T)^*\right)$$

Das heißt, bei jeder Regelanwendung:

- Links eine **einzelne Variable**
- Die Prämisse macht keine Aussage, was der Kontext dieser Variablen ist („kontextfrei“)
- Rechts steht etwas beliebiges

Ableitungsbäume

Definition (Ableitungsbaum zu einer Grammatik)

Sei $G = (V, T, R, S)$ eine kontextfreie Grammatik.

Ein **Ableitungsbaum (parse tree)** zu G ist ein angeordneter Baum $B = (W, E, v_0)$

Zudem muss gelten:

- Jeder Knoten $v \in W$ ist mit einem Symbol aus $V \cup T \cup \{\varepsilon\}$ markiert.
- Die Wurzel v_0 ist mit S markiert.
- Jeder innere Knoten ist mit einer Variablen aus V markiert.
- Jedes Blatt ist mit einem Symbol aus $T \cup \{\varepsilon\}$ markiert.
- Ist $v \in W$ ein innerer Knoten mit Söhnen $v_1, \ldots, v_k$ in dieser Anordnung und ist A die Markierung von v und A_i die Markierung von v_i, dann ist $A \to A_1 \ldots A_k \in R$.
- Ein mit ε markiertes Blatt hat keinen Bruder (denn das entspräche einer Ableitung wie $A \to ab\varepsilon Bc$).

Ableitungsbäume

Ablesen eines Wortes vom Ableitungsbaum

Wenn Wort w von Grammatik G erzeugt wird,
dann gibt es einen Ableitungsbaum mit den Buchstaben von w
als Blätter von links nach rechts.

Merke:

Die Blätter eines Ableitungsbaumes sind angeordnet.

Es gibt eine Ordnung unter den Söhnen eines Knotens.

Ableitungsbäum

Definition

Seien b_1, b_2 Knoten. Dann:

$$b_1 < b_2 \text{ gdw } b_1, b_2 \text{ sind Brüder, und } b_1 \text{ liegt "links" von } b_2,$$
$$\text{oder } \exists v, v_1, v_2 \in W \quad v \to v_1,\ v \to v_2,\ v_1 < v_2$$
$$\text{und } v_i \text{ ist Vorfahre von } b_i \text{ für } i \in \{1, 2\}.$$

Definition

Sei $\{b_1, \ldots, b_k\}$ die Menge aller Blätter in B mit $b_1 < \ldots < b_k$, und sei A_i die Markierung von b_i.
Dann heißt das Wort $A_1 \ldots A_k$ die **Front** von B.

Ableitungsbäume

Theorem.
Sei $G = (V, T, R, S)$ eine kontextfreie Grammatik.
Dann gilt für $w \in T^*$:

$\left(S \Longrightarrow_G^* w\right)$ gdw Es existiert ein Ableitungsbaum zu G mit Front w.

Beweis. Einfach aus den Definitionen.

Ableitungsbäume: Beispiel

Grammatik für die Menge aller aussagenlogischen Formeln über den Variablen $\{x, x_0, x_1, x_2, \ldots\}$:

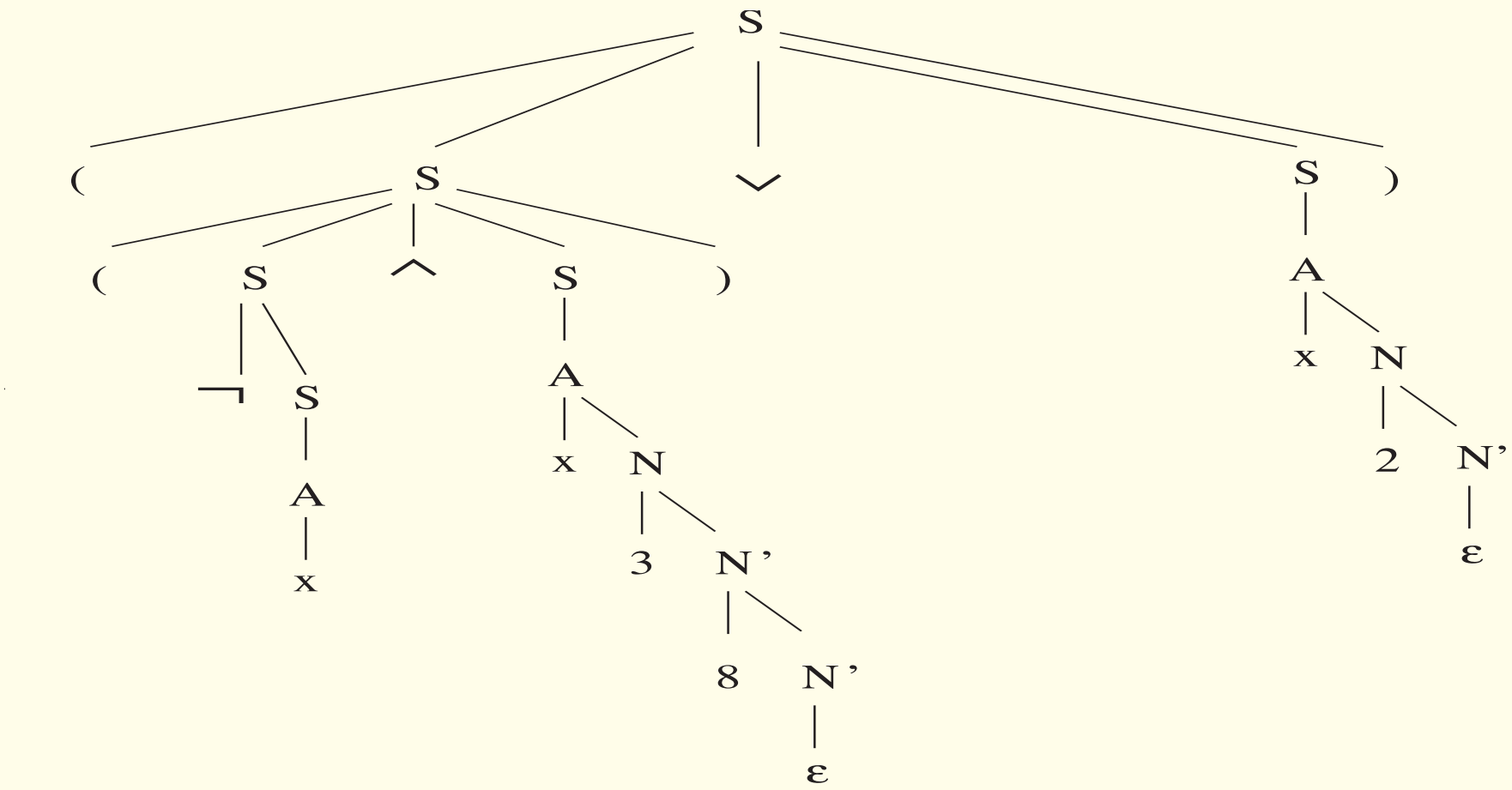
$$G = (\{S, A, N, N'\}, \{x, 0, \ldots, 9, (,), \wedge, \vee, \neg\}, R, S)$$

mit der Regelmenge

$$\begin{aligned} R = \{S &\to (S \wedge S) \mid (S \vee S) \mid \neg S \mid A \\ A &\to x \mid xN \\ N &\to 1N' \mid 2N' \mid \ldots \mid 9N' \mid 0 \\ N' &\to 0N' \mid 1N' \mid \ldots \mid 9N' \mid \varepsilon\} \end{aligned}$$

Ableitungsbäume: Beispiel

Ableitungsbaum für $((\neg x \wedge x38) \vee x2)$



Ableitungsbäume: Beispiel

Der Ableitungsbaum steht für viele **äquivalente** Ableitungen, darunter diese:

$$
\begin{array}{rcrcr}
S \Rightarrow & & (S \vee S) & \Rightarrow \\
((S \wedge S) \vee S) & \Rightarrow & ((\neg S \wedge S) \vee S) & \Rightarrow \\
((\neg A \wedge S) \vee S) & \Rightarrow & ((\neg x \wedge S) \vee S) & \Rightarrow \\
((\neg x \wedge A) \vee S) & \Rightarrow & ((\neg x \wedge xN) \vee S) & \Rightarrow \\
((\neg x \wedge x3N') \vee S) & \Rightarrow & ((\neg x \wedge x38N') \vee S) & \Rightarrow \\
((\neg x \wedge x38) \vee S) & \Rightarrow & ((\neg x \wedge x38) \vee A) & \Rightarrow \\
((\neg x \wedge x38) \vee xN) & \Rightarrow & ((\neg x \wedge x38) \vee x2N') & \Rightarrow \\
((\neg x \wedge x38) \vee x2) & &
\end{array}
$$

Links- und Rechtsableitung

Definition (Linksableitung)

Eine Ableitung

$$w_1 \Longrightarrow_G w_2 \Longrightarrow_G \ldots \Longrightarrow_G w_n$$

heißt **Linksableitung** falls w_{i+1} durch Ersetzen der linkesten Variable in w_i entsteht für alle $i < n$.

Die **Rechtsableitung** ist analog definiert.

Mehrdeutigkeit

Definition (Mehrdeutigkeit)

Eine cf-Grammatik G heißt **mehrdeutig**

gdw

es gibt ein Wort $w \in L(G)$,
zu dem es in G **zwei verschiedene Linksableitungen** gibt.

Eine Sprache $L \in \mathbf{L}_2$ heißt **inhärent mehrdeutig**

gdw

alle kontextfreien Grammatiken für L sind mehrdeutig.

Bemerkung

Eine Grammatik G ist mehrdeutig gdw :

es gibt zwei verschiedene Ableitungsbäume in G mit gleicher Front.

Mehrdeutigkeit: Beispiele

Eindeutige Grammatik für aussagenlogische Formeln:

$$
\begin{aligned}
S &\to (S \ \wedge \ S) \mid (S \ \vee \ S) \mid \neg S \mid A \\
A &\to x \mid xN \\
N &\to 1N' \mid 2N' \mid \ldots \mid 9N' \mid 0 \\
N' &\to 0N' \mid 1N' \mid \ldots \mid 9N' \mid \varepsilon
\end{aligned}
$$

Mehrdeutigkeit: Beispiele

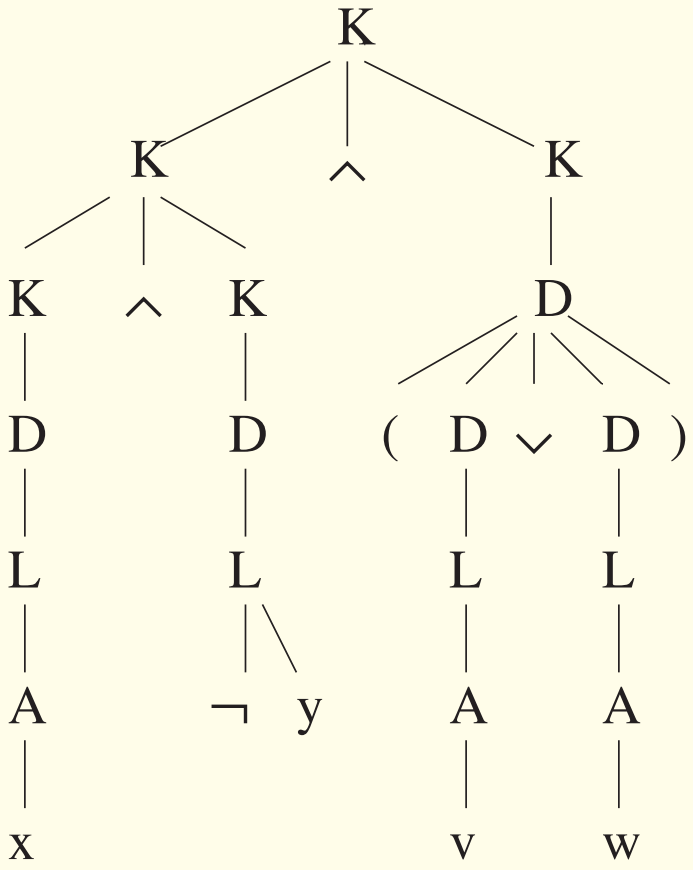
Eindeutige Grammatik für aussagenlogische Formeln:

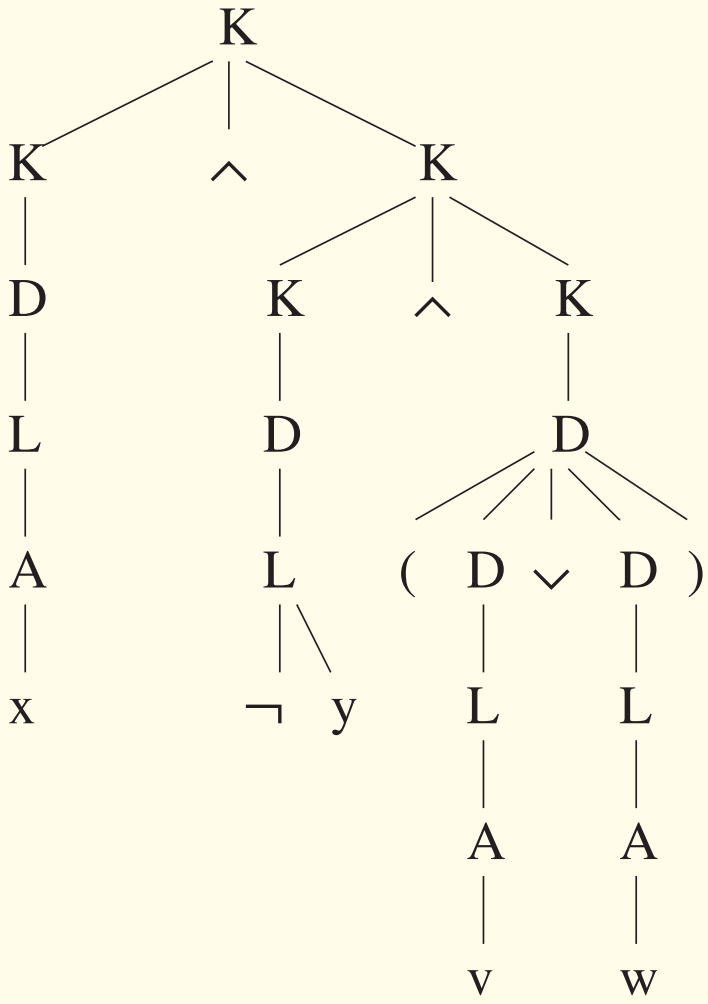
$$
\begin{aligned}
S &\to (S \ \wedge \ S) \mid (S \ \vee \ S) \mid \neg S \mid A \\
A &\to x \mid xN \\
N &\to 1N' \mid 2N' \mid \ldots \mid 9N' \mid 0 \\
N' &\to 0N' \mid 1N' \mid \ldots \mid 9N' \mid \varepsilon
\end{aligned}
$$

Mehrdeutige Grammatik für aussagenlogische Formeln in KNF:

$$
\begin{aligned}
K &\to K \wedge K \mid D \qquad \textbf{Regel mit Klammer-Ersparnis!} \\
D &\to (D \vee D) \mid L \\
L &\to \neg A \mid A \\
A &\to v \mid w \mid x \mid y \mid z
\end{aligned}
$$

Mehrdeutigkeit: Beispiele





Mehrdeutigkeit: Beispiele

Inhärente Mehrdeutigkeit

Die Sprache

$$L := \{a^i b^j c^k \mid i = j \text{ oder } j = k\}$$

ist **inhärent mehrdeutig**.

Für einen Beweis siehe [Wegener "Theoretische Informatik" 1993 S.168-170], cf. Beispiel 6.1.7 Erk, Priese "Theoretische Informatik".

Umformung von Grammatiken

Startsymbol nur links

Einfache Annahme:

Im folgenden soll für alle cf-Grammatiken gelten:

Das Startsymbol S kommt nie auf einer rechten Regelseite vor.

Umformung

Ist das bei einer Grammatik nicht gegeben, kann man es wie folgt erreichen:

- Führe ein neues Startsymbol S_{neu} ein
- Füge die Regel

  $$S_{neu} \to S$$

  hinzu.

Nutzlose Symbole

Nutzlose Symbole und Regeln: Intuition

- Variablen und Symbole, die vom Startsymbol aus unerreichbar sind.
- Variablen, von denen aus kein Terminalwort abgeleitet werden kann.
- Regeln, die solche Variablen und Symbole enthalten

Nutzlose Symbole

Definition ((co-)erreichbare, nutzlose Symbole)

Sei $G = (V, T, R, S)$ eine Grammatik.

Ein Symbol $x \in (V \cup T)$ heiÃt

erreichbar: Es gibt $\alpha, \beta \in (V \cup T)^*$: $S \Longrightarrow_G^* \alpha x \beta$

co-erreichbar: Es gibt $w \in T^*$: $x \Longrightarrow_G^* w$

nutzlos: x ist nicht erreichbar oder nicht co-erreichbar.

Nutzlose Symbole

Theorem (cf-Grammatik ohne nutzlose Symbole)

Ist $G = (V, T, R, S)$ eine cf-Grammatik mit $L(G) \neq \emptyset$,
dann existiert eine cf-Grammatik $G' = (V', T', R', S')$ mit:

- G' ist äquivalent zu G.
- Jedes $x \in (V \cup T)$ ist erreichbar und co-erreichbar.

Nutzlose Symbole

Theorem (cf-Grammatik ohne nutzlose Symbole)
Ist $G = (V, T, R, S)$ eine cf-Grammatik mit $L(G) \neq \emptyset$,
dann existiert eine cf-Grammatik $G' = (V', T', R', S')$ mit:

- G' ist äquivalent zu G.
- Jedes $x \in (V \cup T)$ ist erreichbar und co-erreichbar.

Beweis

Man kann G' aus G effektiv konstruieren:

- Wie im folgenden beschrieben, die nutzlosen Symbole bestimmen.
- Diese Symbole und alle Regeln, die sie enthalten, entfernen.

Nutzlose Symbole

Algorithmus zur Berechnung der co-erreichbaren Variablen

Input: Grammatik $G = (V, T, R, S)$
Output: co-erreichbare Variablen

$\mathsf{Alt} := \emptyset$
$\mathsf{Neu} := \{A \in V \mid \exists w \in T^* \ (A \to w \in R)\}$

while $\mathsf{Alt} \neq \mathsf{Neu}$
{
    $\mathsf{Alt} := \mathsf{Neu}$
    $\mathsf{Neu} := \mathsf{Alt} \cup \{A \in V \mid \exists \alpha \in (T \cup \mathsf{Alt})^* \ (A \to \alpha \in R)\}$
}

output Neu

Nutzlose Symbole

Bestimmung einer Grammatik $G'' = (V'', T'', R'', S)$ nur mit diesen co-erreichbaren Variablen.

if $S \in$ Neu /* S ist co-erreichbar */

{

$V'' :=$ Neu

$T'' := T$

$R'' = R \cap (V'' \times (V'' \cup T'')^*)$

}

else /* $L(G) = \emptyset$ */

Nutzlose Symbole

Algorithmus zur Berechnung der erreichbaren Variablen von G''

Input: Grammatik $G'' = (V'', T'', R'', S)$
Output: erreichbare Symbole

Alt $:= \emptyset$
Neu2 $:= \{S\}$
while Alt $\neq$ Neu2
{
    Alt := Neu2
    Neu2 := Alt $\cup \{x \in (V'' \cup T'') \mid \exists A \in$ Alt
    $\exists \alpha, \beta \in (V'' \cup T'')^*$
    $(A \to \alpha x \beta \in R'')\}$
}

output Neu2

Nutzlose Symbole

Bestimmung der Grammatik G' ohne nutzlose Symbole:

$G' = (V', T', R', S')$ mit:

$V' := \text{Neu2} \cap V''$

$T' = \text{Neu2} \cap T$

$R' = R'' \cap (V' \times (V' \cup T')^*)$

$S' = S$

Damit gilt dann: $L(G') = L(G)$ und G' enthält keine nutzlosen Symbole.

Nutzlose Symbole

- Algorithmus zur Berechnung der Menge Neu der co-erreichbaren Variablen
- Bestimmung einer Grammatik $G'' = (V'', T'', R'', S)$ nur mit diesen co-erreichbaren Variablen.

  Falls S ist co-erreichbar:
  - $V'' := \text{Neu}$
  - $T'' := T$
  - $R'' = R \cap (V'' \times (V'' \cup T'')^*)$

  sonst: $L(G) = \emptyset$
- Algorithmus zur Berechnung der Menge Neu2 der erreichbaren Symbole von G''
- Bestimmung der Grammatik $G' = (V', T', R', S')$ ohne nutzlose Symbole:
  - $V' := \text{Neu2} \cap V''$
  - $T' = \text{Neu2} \cap T$
  - $R' = R'' \cap (V' \times (V' \cup T')^*)$
  - $S' = S$

  Damit gilt dann: $L(G') = L(G)$ und G' enthält keine nutzlosen Symbole.

Normalform für Regeln

Normalform für Regeln

Theorem (Normalform)

Zu jeder Grammatik G (beliebigen Typs)
existiert eine äquivalente Grammatik G',
bei der für alle Regeln $P \to Q \in R'$ gilt:

- $Q \in V^*$ und P beliebig
- $Q \in T$ und $P \in V$

Für alle Typen außer den linearen hat G' denselben Typ wie G.

Normalform für Regeln

Beweis

Für jedes Terminal $t \in T$ erzeuge man eine neue Variable V_t.

- $V' = V \cup \{V_t \mid t \in T\}$
- R' entsteht aus R, indem für jede Regel $P \to Q \in R$ in Q alle Vorkommen eines Terminals t durch die zugehörige Variable V_t ersetzt werden. Außerdem enthält R' für jedes $t \in T$ eine neue Regel $V_t \to t$.

Also $L(G') = L(G)$,
und für alle Sprachklassen außer $\mathbf{L}_3$ hat G' denselben Typ wie G.

Elimination von ε-Regeln

Idee: Variablen, aus denen ε ableitbar ist, sollten eliminiert werden

Elimination von ε-Regeln

Idee: Variablen, aus denen ε ableitbar ist, sollten eliminiert werden

Definition (ε-Regel, nullbare Variablen)

Eine Regel der Form

$$P \to \varepsilon \qquad (P \text{ eine Variable})$$

heißt **ε-Regel**.

Eine Variable A heißt **nullbar**,
falls

$$A \Longrightarrow^* \varepsilon$$

Elimination von ε-Regeln

Theorem (ε-Regeln sind eliminierbar)

Zu jeder cf-Grammatik G existiert eine äquivalente cf-Grammatik G'

- ohne ε-Regeln und nullbare Variablen,
  falls $\varepsilon \notin L(G)$,
- mit der einzigen ε-Regel $S \to \varepsilon$ und der einzigen nullbaren Variablen S,
  falls $\varepsilon \in L(G)$ und S das Startsymbol ist.

Elimination von ε-Regeln

Algorithmus zur Berechnung der nullbaren Variablen

Input: Grammatik $G = (V, T, R, S)$ S o.B.d.A. in keiner Regel rechts
Output: nullbare Variablen

$Alt := \emptyset$
$Neu := \{A \in V \mid A \to \varepsilon \in R\}$
while $Alt \neq Neu$
{ $Alt := Neu$
für alle $(P \to Q) \in R$ **do**
{ **if** $Q = A_1 \ldots A_n$ **and** $A_i \in Neu$ für $1 \leq i \leq n$ **and** $P \notin Neu$,
then $Neu := Neu \cup \{P\}$
}
}

output Neu

Elimination von ε-Regeln

Beweis (Forts.)

Ausgangsgrammatik G habe die Normalform, bei der für jede Regel $P \to Q$: $Q \in V^*$ oder $Q \in T$.

Für jede Regel $P \to A_1 \ldots A_n$ generiere alle möglichen Kombinationen

$$P \to \alpha_1 \ldots \alpha_n$$

mit

- $\alpha_i \in \{\varepsilon, A_i\}$ falls A_i nullbar
- $\alpha_i = A_i$ falls A_i nicht nullbar

Dann

- Füge alle diese neuen Regeln zur Grammatik hinzu
- Entferne alle Regeln der Form $A \to \varepsilon$ mit $A \neq S$

Elimination von ε-Regeln

Beweis ((Forts.)

Zu zeigen:
Für die neue Grammatik G' gilt: $L(G') = L(G)$

Vorgehen:

- G hat die Normalform:

  Für jede Regel $P \to Q$ gilt $Q \in V^*$ oder $Q \in T$.

- Wir beweisen die etwas stärkere Behauptung

$$\text{für alle } A \in V \quad \text{für alle } w \in (V \cup T)^* - \{\varepsilon\}$$
$$\big((A \Longrightarrow_G^* w) \quad \underline{\text{gdw}} \quad (A \Longrightarrow_{G'}^* w)\big),$$

- Daraus folgt sofort $L(G') = L(G)$.

Elimination von ε-Regeln: Beispiel

R :

$$
\begin{aligned}
S &\to ABD \\
A &\to ED \mid BB \\
B &\to AC \mid \varepsilon \\
C &\to \varepsilon \\
D &\to d \\
E &\to e
\end{aligned}
$$

R' :

$$
\begin{aligned}
S &\to ABD \mid AD \mid BD \mid D \\
A &\to ED \mid BB \mid B \\
B &\to AC \mid A \mid C \\
D &\to d \\
E &\to e
\end{aligned}
$$

Elimination von ε-Regeln: Beispiel

R:			R':		
S	$\to$	ABD	S	$\to$	$ABD \mid AD \mid BD \mid D$
A	$\to$	$ED \mid BB$	A	$\to$	$ED \mid BB \mid B$
B	$\to$	$AC \mid \varepsilon$	B	$\to$	$AC \mid A \mid C$
C	$\to$	ε			
D	$\to$	d	D	$\to$	d
E	$\to$	e	E	$\to$	e

Für die Regelmenge R in der linken Spalte sind die Variablen A, B, C nullbar.

Elimination von ε-Regeln: Beispiel

R:			R':		
S	$\to$	ABD	S	$\to$	$ABD \mid AD \mid BD \mid D$
A	$\to$	$ED \mid BB$	A	$\to$	$ED \mid BB \mid B$
B	$\to$	$AC \mid \varepsilon$	B	$\to$	$AC \mid A \mid C$
C	$\to$	ε			
D	$\to$	d	D	$\to$	d
E	$\to$	e	E	$\to$	e

Für die Regelmenge R in der linken Spalte sind die Variablen A, B, C nullbar.

Der obige Algorithmus erzeugt aus R die rechts aufgeführte Regelmenge R'.

Elimination von ε-Regeln

Beobachtung

- Der Algorithmus lässt nutzlose Variablen zurück,
  die nicht in Prämissen auftauchen
  (und deshalb nicht co-erreichbar sind).
  Hier: C.
- Der Algorithmus lässt nutzlose Regeln zurück.
  Hier: $B \to AC \mid C$.