Grundlagen der Theoretischen Informatik

4. Kellerautomaten und kontextfreie Sprachen (II)

16.05.2018

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

Bis jetzt

Erinnerung: kontextfreie Grammatiken

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

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

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

Bis jetzt

Ableitungsbäume

- Ablesen eines Wortes vom Ableitungsbaum

- Front eines Ableitungsbäumes

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

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

Bis jetzt

Links- und Rechtsableitung

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.

Bis jetzt

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

  $$S_{neu} \rightarrow S$$

  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 $x \in (V \cup T)$ heißt

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

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

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

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:

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

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

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

$S' = S$

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:

  - $V'' :=$ 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:
  - $V' :=$ Neu2 $\cap V''$
  - $T' =$ 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** $Q = A_1 \dots A_n$ **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

$$\text{für alle } A \in V \;\; \text{für alle } w \in (V \cup T)^* - \{\varepsilon\}$$

$$\Big((A \Longrightarrow^*_G w) \quad \underline{\text{gdw}} \quad (A \Longrightarrow^*_{G'} w)\Big),$$

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

Elimination von ε-Regeln: Beispiel

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

Elimination von ε-Regeln: Beispiel

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

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

Elimination von ε-Regeln: Beispiel

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

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: $B \to AC \mid C$.

Elimination von ε-Regeln

Beweis (Forts.) Wir beweisen die Behauptung

$$\text{für alle } A \in V \;\; \text{für alle } w \in (V \cup T)^* - \{\varepsilon\}$$

$$\big((A \Longrightarrow_G^* w) \quad \underline{\text{gdw}} \quad (A \Longrightarrow_{G'}^* w)\big),$$

"$\Rightarrow$" Wir zeigen: Aus $A \Longrightarrow_G^* w$ folgt $A \Longrightarrow_{G'}^* w$ (Induktion über Länge einer Ableitung von A nach w in G).

Induktionsanfang: Länge $= 0$.

Dann ist $w = A$, und $A \Longrightarrow_{G'}^* A$ gilt immer.

Induktionsschritt: Es sei schon gezeigt: Wenn in G in n Schritten eine Ableitung $B \Longrightarrow_G^* u$ durchgeführt werden kann, dann folgt, dass in G' die Ableitung $B \Longrightarrow_{G'}^* u$ möglich ist.

Elimination von ε-Regeln

Beweis (Forts.)

Außerdem gelte in der Ausgangsgrammatik G: $A \Longrightarrow_G^* w \neq \varepsilon$ in $n+1$ Schritten.

Dann gilt:

- $A \Longrightarrow_G w' \Longrightarrow_G^* w$,
- $w' = A_1 \dots A_\ell \Longrightarrow_G^* w_1 \dots w_\ell = w$,
- und es wird jeweils A_i zu w_i in höchstens n Schritten für geeignete $w', A_1, \dots, A_\ell, w_1, \dots, w_\ell$.
- Für $1 \leq i \leq \ell$ gilt:
  - Entweder $w_i \neq \varepsilon$ und $A_i \Longrightarrow_G^* w_i$
    also (per Induktionsvoraussetzung) $A_i \Longrightarrow_{G'}^* w_i$
  - oder $w_i = \varepsilon$ und $A_i \Longrightarrow_G^* w_i$.

Elimination von ε-Regeln

Beweis (Forts.)

Fall 1: $w_i = \varepsilon$, A_i ist nullbar.

Dann gibt es in G' eine Regel $A \to A_1 \ldots A_{i-1}A_{i+1} \ldots A_\ell$ nach der obigen Konstruktionsvorschrift für G', falls $A_1 \ldots A_{i-1}A_{i+1} \ldots A_\ell \neq \varepsilon$. Das ist der Fall, denn sonst hätten wir: $A \Longrightarrow w' = \varepsilon \Longrightarrow^* w = \varepsilon$ (aus nichts wird nichts), aber $w = \varepsilon$ ist ausgeschlossen.

Fall 2: $w_i \neq \varepsilon$. Dann gilt nach Induktionsvoraussetzung

$A_i \Longrightarrow^*_{G'} w_i$.

Elimination von ε-Regeln

Beweis (Forts.)

Wir haben also folgendes gezeigt:

Sei $I = \{i \in \{1 \ldots \ell\} \mid w_i \neq \varepsilon\} \neq \emptyset$.

Dann gibt es in R' eine Regel $A \to A_{i_1} \ldots A_{i_m}$ mit $I = \{i_1, \ldots, i_m\}$, und die A_i sind so angeordnet wie in der ursprünglichen Regel $A \to A_1 \ldots A_\ell$.

Mit dieser neuen Regel können wir w so ableiten:

$$A \Longrightarrow_{G'} A_{i_1} \ldots A_{i_m} \Longrightarrow^*_{G'} w_{i_1} \ldots w_{i_m} = w$$

Elimination von ε-Regeln

Beweis (Forts.)

"$\Leftarrow$" Wir zeigen: Aus $A \Longrightarrow^*_{G'} w$ folgt $A \Longrightarrow^*_G w$ (Induktion über Länge einer Ableitung von A nach w in G'):

Induktionsanfang: Länge $= 0$. Dann ist $w = A$, und $A \Longrightarrow^*_G A$ gilt immer.

Induktionsschritt: Es gelte für alle Ableitungen $A \Longrightarrow^*_{G'} w$ einer Länge von höchstens n, dass $A \Longrightarrow^*_G w$.

Ist $A \Longrightarrow^*_{G'} w$ eine Ableitung der Länge $n+1$, so gibt es ein ℓ, Wörter $w_1, \ldots, w_\ell$ und Variablen $A_1, \ldots, A_\ell$ mit $A \Longrightarrow_{G'} A_1 \ldots A_\ell \Longrightarrow^*_{G'} w = w_1 \ldots w_\ell$. Es gilt jeweils $A_i \Longrightarrow^*_{G'} w_i$ in höchstens n Schritten, und $w_i \neq \varepsilon$.

Elimination von ε-Regeln

Beweis (Forts.)

Nach der Induktionsvoraussetzung folgt daraus:

- für die Originalgrammatik G gibt es Ableitungen $A_i \Longrightarrow_G^* w_i$
- damit gibt es auch eine Ableitung $A_1 \dots A_\ell \Longrightarrow_G^* w$.

Elimination von ε-Regeln

Beweis (Forts.)

Nach der Induktionsvoraussetzung folgt daraus:

- für die Originalgrammatik G gibt es Ableitungen $A_i \Longrightarrow_G^* w_i$
- damit gibt es auch eine Ableitung $A_1 \ldots A_\ell \Longrightarrow_G^* w$.

Da es in G' eine Ableitung $A \Longrightarrow_{G'} A_1 \ldots A_\ell$ gibt, gibt es in R' eine Regel $A \to A_1 \ldots A_\ell$. Wie ist diese Regel aus R entstanden?

Elimination von ε-Regeln

Beweis (Forts.)

Nach der Induktionsvoraussetzung folgt daraus:

- für die Originalgrammatik G gibt es Ableitungen $A_i \Longrightarrow_G^* w_i$
- damit gibt es auch eine Ableitung $A_1 \dots A_\ell \Longrightarrow_G^* w$.

Da es in G' eine Ableitung $A \Longrightarrow_{G'} A_1 \dots A_\ell$ gibt, gibt es in R' eine Regel $A \to A_1 \dots A_\ell$. Wie ist diese Regel aus R entstanden?

Eine Regel in R' entsteht aus einer Regel in R, indem einige nullbare Variablen gestrichen werden. Es gab also in G nullbare Variablen B_1 bis B_m, so dass R die Regel

$$A \to A_1 \dots A_{\ell_1} B_1 A_{\ell_1+1} \dots A_{\ell_2} B_2 \dots A_m B_m A_{m+1} \dots A_\ell$$

enthält. (m kann auch 0 sein, dann war die Regel selbst schon in R.)

Elimination von ε-Regeln

Beweis (Forts.)

Also gilt in G:

$$\begin{aligned} A &\Longrightarrow_G A_1 \dots A_{\ell_1} B_1 A_{\ell_1+1} \dots A_{\ell_2} B_2 \dots A_m B_m A_{m+1} \dots A_\ell \\ &\Longrightarrow_G^* A_1 \dots A_{\ell_1} A_{\ell_1+1} \dots A_{\ell_2} \dots A_m A_{m+1} \dots A_\ell \Longrightarrow_G^* w \end{aligned}$$

da ja $B_i \Longrightarrow_G^* \varepsilon$ möglich ist. $\square$

Elimination von ε-Regeln: Beispiel

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

Elimination von ε-Regeln: Beispiel

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

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

Elimination von ε-Regeln: Beispiel

$$R:\qquad\qquad\qquad\qquad R':$$

$$\begin{array}{rcl@{\qquad\qquad}rcl}
S & \to & ABD & S & \to & ABD \mid AD \mid BD \mid D \\
A & \to & ED \mid BB & A & \to & ED \mid BB \mid B \\
B & \to & AC \mid \varepsilon & B & \to & AC \mid A \mid C \\
C & \to & \varepsilon & & & \\
D & \to & d & D & \to & d \\
E & \to & e & E & \to & e
\end{array}$$

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: $B \to AC \mid C$.

Elimination von ε-Regeln: Beispiel

Nach Elimination der nutzlosen Variablen/Regeln

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

Elimination von ε-Regeln

Korollar.

$$\mathcal{L}_2 \subseteq \mathcal{L}_1$$

Das heißt, jede kontextfreie Sprache ist auch kontextsensitiv

Elimination von ε-Regeln

Korollar.

$$\mathcal{L}_2 \subseteq \mathcal{L}_1$$

Das heißt, jede kontextfreie Sprache ist auch kontextsensitiv

Beweis. Regeln einer kontextsensitiven Grammatik müssen folgende Form haben:

- entweder $uAv \to u\alpha v$
  mit $u, v, \alpha \in (V \cup T)^*, |\alpha| \geq 1, A \in V$
- oder $S \to \varepsilon$
  und S kommt in keiner Regelconclusio vor.

Diesen Bedingungen genügt die kontextfreie Grammatik nach Elimination der ε-Regeln.