Vorlesung

Grundlagen der Theoretischen Informatik / Einführung in die Theoretische Informatik I

Bernhard Beckert

Institut für Informatik

UNIVERSITÄT KOBLENZ · LANDAU

Sommersemester 2007

Dank

Diese Vorlesungsmaterialien basieren ganz wesentlich auf den Folien zu den Vorlesungen von

Katrin Erk (gehalten an der Universität Koblenz-Landau)

Jürgen Dix (gehalten an der TU Clausthal)

Ihnen beiden gilt mein herzlicher Dank.

– Bernhard Beckert, April 2007

Inhalt von Teil IV

- Die von **Kellerautomaten** (Push-Down-Automaten, PDAs) erkannten Sprachen sind genau die vom Typ 2 (**kontextfrei**).
- **Normalformen** für kontextfreie Grammatiken.
- **Pumping-Lemma** für kontextfreie Sprachen.
- Effiziente Algorithmen für Probleme über PDAs

Teil IV

Kellerautomaten und kontextfreie Sprachen

Zur Erinnerung: kontextfreie Grammatiken

Kontextfreie Grammatiken

- Kontextfreie Regel:
  Eine Variable wird durch ein Wort ersetzt,
  (egal in welchem Kontext die Variable steht)
- Es wird eine **einzelne** Variable ersetzt.
- Das Wort in der Conclusio kann Variablen und Terminale in **beliebiger Mischung** enthalten.

Zur Erinnerung: kontextfreie Sprachen

Beispiel 18.1 (kontextfreie Sprachen)

- $\{a^n b^n \mid n \in \mathbb{N}_0\}$
- $\{a^n b a^n \mid n \in \mathbb{N}_0\}$
- $\{ww^R \mid w \in \{a,b\}^*\}$

Ableitungsbäume

Definition 18.2 (Ableitungsbaum zu einer Grammatik)

Sei

$$G = (V, T, R, S)$$

eine kontextfreie Grammatik.

Ein **Ableitungsbaum (parse tree)** zu G ist ein angeordneter Baum

$$B = (W, E, v_0)$$

Ableitungsbäume

Definition 18.3 (Ableitungsbaum zu einer Grammatik, Fortsetzung)

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 $v_1, \dots, v_k$ in dieser Anordnung und ist A die Markierung von v und A_i die Markierung von v_i, dann ist $A \to A_1 \dots A_k \in R$.
- 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.

Ableitungsbäume

Definition 18.4

Seien b_1, b_2 Blätter. Dann:

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

Definition 18.5

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

Ableitungsbäume

Theorem 18.6

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

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

Beweis.

Einfach aus den Definitionen. □

Ableitungsbäume: Beispiel

Beispiel 18.7

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

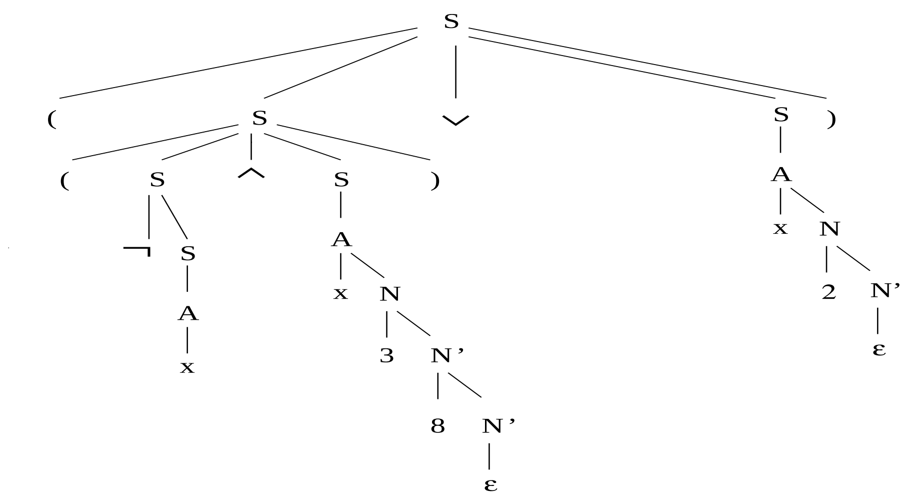
$$G = (\{S, A, N, N'\}, \{x, 0, \dots, 9, (,), \wedge, \vee, \neg\}, R, S)$$

mit der Regelmenge

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

Ableitungsbäume: Beispiel

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



Ableitungsbäume: Beispiel

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

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

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

Links- und Rechtsableitung

Definition 18.8 (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 18.9 (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

Beispiel 18.10 (Mehrdeutigkeit)

Eindeutige Grammatik für aussagenlogische Formeln:

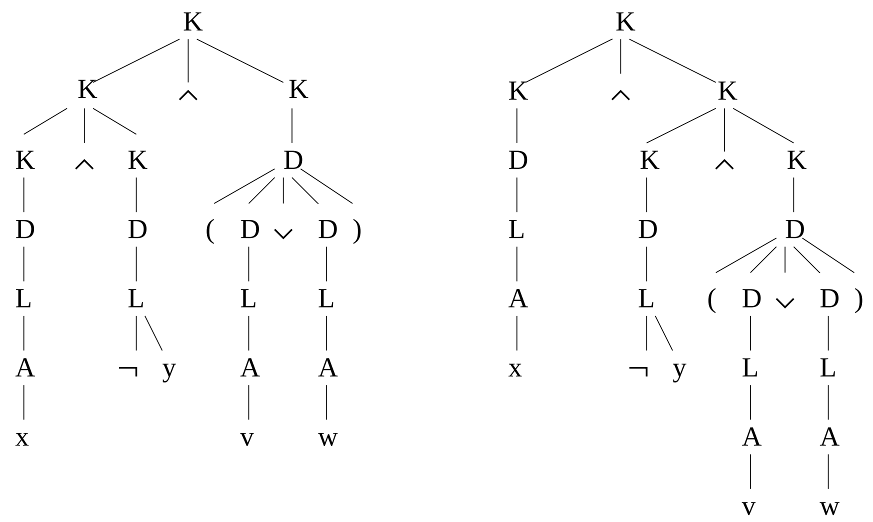
$$
\begin{aligned}
S &\rightarrow (S \wedge S) \mid (S \vee S) \mid \neg S \mid A \\
A &\rightarrow x \mid xN \\
N &\rightarrow 1N' \mid 2N' \mid \ldots \mid 9N' \mid 0 \\
N' &\rightarrow 0N' \mid 1N' \mid \ldots \mid 9N' \mid \varepsilon\}
\end{aligned}
$$

Mehrdeutige Grammatik für aussagenlogische Formeln:

$$
\begin{aligned}
K &\rightarrow K \wedge K \mid D \\
D &\rightarrow (D \vee D) \mid L \\
L &\rightarrow \neg A \mid A \\
A &\rightarrow v \mid w \mid x \mid y \mid z
\end{aligned}
$$

Regel mit Klammer-Ersparnis!

Mehrdeutigkeit: Beispiele



Mehrdeutigkeit: Beispiele

Beispiel 18.11 (Inhärente Mehrdeutigkeit)

Die Sprache

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

ist **inhärent mehrdeutig**.

Teil IV

Kellerautomaten und kontextfreie Sprachen

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 19.1 ((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 19.2 (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

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

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

output Neu

Nutzlose Symbole

Algorithmus zur Berechnung der erreichbaren Symbole

Input: Grammatik $G = (V, T, R, S)$
Output: erreichbare Symbole

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

output Neu

Normalform für Regeln

Theorem 19.3 (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

Definition 19.4 (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 19.5 (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 $1 \le i \le n$ **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 $P \to A_1 \dots A_n$ generiere alle möglichen Kombinationen

$$P \to \alpha_1 \dots \alpha_n$$

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
  $$\text{für alle } A \in V \text{ für alle } w \in (V \cup T)^* - \{\varepsilon\}$$
  $$\big((A \Longrightarrow_G^* w) \ \underline{\text{gdw}} \ (A \Longrightarrow_{G'}^* w)\big),$$
- Daraus folgt sofort $L(G') = L(G)$.

Elimination von ε-Regeln

Beweis (Forts.)

"⇒" 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:

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

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

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

Mit dieser neuen Regel können wir w so ableiten:

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

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 $w_1, \dots, w_\ell$ und Variablen $A_1, \dots, A_\ell$ mit $A \Longrightarrow_{G'} A_1 \dots A_\ell \Longrightarrow^*_{G'} w = w_1 \dots w_\ell$. 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 $A_1 \dots A_\ell \Longrightarrow^*_G w$.

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. □

Elimination von ε-Regeln: Beispiel

Beispiel 19.6

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

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: $B \longrightarrow AC \mid C$.