Einführung in die Theoretische Informatik

Johannes Köbler

Institut für Informatik
Humboldt-Universität zu Berlin

WS 2016/17

Kontextsensitive Sprachen

Definition

Sei $G = (V, \Sigma, P, S)$ eine Grammatik.

1. G heißt vom Typ 3 oder regulär, falls für alle Regeln $u \to v$ gilt:

   $u \in V$ und $v \in \Sigma V \cup \Sigma \cup \{\varepsilon\}$.

   (d.h. alle Regeln haben die Form $A \to aB$, $A \to a$ oder $A \to \varepsilon$)

2. G heißt vom Typ 2 oder kontextfrei, falls für alle Regeln $u \to v$ gilt:

   $u \in V$. (d.h. alle Regeln haben die Form $A \to \alpha$)

3. G heißt vom Typ 1 oder kontextsensitiv, falls für alle Regeln $u \to v$ gilt:

   $|v| \geq |u|$. (mit Ausnahme der ε-Sonderregel, s. unten)

4. Jede Grammatik ist automatisch vom Typ 0.

Die ε-Sonderregel

In einer kontextsensitiven Grammatik ist auch die Regel $S \to \varepsilon$ zulässig. Aber nur, wenn das Startsymbol S nur links vorkommt.

CFL ist echt in CSL enthalten

Bemerkung

- Wie wir gesehen haben, ist CFL in CSL enthalten.
- Zudem ist folgende Sprache nicht kontextfrei:

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

- L kann jedoch von einer kontextsensitiven Grammatik erzeugt werden.
- Daher ist CFL echt in CSL enthalten.

Eine kontextsensitive Grammatik für $\{a^n b^n c^n \mid n \geq 1\}$

Beispiel

- Betrachte die kontextsensitive Grammatik $G = (V, \Sigma, P, S)$ mit $V = \{S, B\}$, $\Sigma = \{a, b, c\}$ und den Regeln

$$P\colon S \to aSBc, abc \ (1,2) \quad cB \to Bc \ (3) \quad bB \to bb \ (4)$$

- In G lässt sich beispielsweise das Wort $w = aabbcc$ ableiten:

$$\underline{S} \underset{(1)}{\Rightarrow} a\underline{S}Bc \underset{(2)}{\Rightarrow} aab\underline{cB}c \underset{(3)}{\Rightarrow} aa\underline{bB}cc \underset{(4)}{\Rightarrow} aabbcc$$

- Allgemein gilt für alle $n \geq 1$:

$$S \underset{(1)}{\Rightarrow}^{n-1} a^{n-1}S(Bc)^{n-1} \underset{(2)}{\Rightarrow} a^n bc(Bc)^{n-1} \underset{(3)}{\Rightarrow}^{\binom{n}{2}} a^n bB^{n-1}c^n \underset{(4)}{\Rightarrow}^{n-1} a^n b^n c^n$$

- Also gilt $a^n b^n c^n \in L(G)$ für alle $n \geq 1$.

Eine kontextsensitive Grammatik für $\{a^n b^n c^n \mid n \geq 1\}$

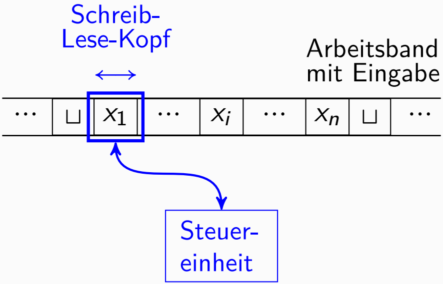
Beispiel (Schluss)

- Betrachte die kontextsensitive Grammatik $G = (V, \Sigma, P, S)$ mit $V = \{S, B\}$, $\Sigma = \{a, b, c\}$ und den Regeln

$$P\colon S \to aSBc, abc\ (1,2) \quad cB \to Bc\ (3) \quad bB \to bb\ (4)$$

- Umgekehrt folgt durch Induktion über die Ableitungslänge m, dass jede Satzform $\alpha \in (V \cup \Sigma)^*$ mit $S \Rightarrow^m \alpha$ die folgenden Bedingungen erfüllt:
  - $\#_a(\alpha) = \#_b(\alpha) + \#_B(\alpha) = \#_c(\alpha)$,
  - links von S kommen nur a's vor,
  - links von einem a kommen ebenfalls nur a's vor,
  - links von einem b kommen nur a's oder b's vor.
- Daraus ergibt sich, dass in G nur Wörter $w \in \Sigma^*$ der Form $w = a^n b^n c^n$ ableitbar sind, d.h. $L(G) = \{a^n b^n c^n \mid n \geq 1\} \in \mathrm{CSL}$. $\triangleleft$

Die Turingmaschine



- Um ein geeignetes Maschinenmodell für die kontextsensitiven Sprachen zu finden, führen wir zunächst das Rechenmodell der nichtdeterministischen Turingmaschine (NTM) ein.
- Eine NTM erhält ihre Eingabe auf einem nach links und rechts unbegrenzten Band.
- Während ihrer Rechnung kann sie den Schreib-Lese-Kopf auf dem Band in beide Richtungen bewegen und dabei die besuchten Bandfelder lesen sowie die gelesenen Zeichen gegebenenfalls überschreiben.

Das Rechenmodell der Turingmaschine

Definition

- Sei $k \geq 1$. Eine nichtdeterministische k-Band-Turingmaschine (k-NTM oder einfach NTM) wird durch ein 6-Tupel $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ beschrieben, wobei
  - Z eine endliche Menge von Zuständen,
  - Σ das Eingabealphabet (mit $\sqcup \notin \Sigma$),
  - Γ das Arbeitsalphabet (mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$),
  - δ: $Z \times \Gamma^k \to \mathcal{P}(Z \times \Gamma^k \times \{L, R, N\}^k)$ die Überführungsfunktion,
  - q_0 der Startzustand und
  - $E \subseteq Z$ die Menge der Endzustände ist.
- Eine k-NTM M heißt deterministisch (kurz: M ist eine k-DTM oder einfach DTM), falls für alle $(q, a_1, \dots a_k) \in Z \times \Gamma^k$ gilt:

$$\|\delta(q, a_1, \dots a_k)\| \leq 1.$$

- Für $(q, b_1, \dots, b_k, D_1, \dots, D_k) \in \delta(p, a_1, \dots a_k)$ schreiben wir auch

$$(p, a_1, \dots, a_k) \to (q, b_1, \dots, b_k, D_1, \dots, D_k).$$

- Eine solche Anweisung ist ausführbar, falls
  - p der aktuelle Zustand von M ist und
  - sich für $i = 1, \dots, k$ der Lesekopf des i-ten Bandes auf einem mit a_i beschrifteten Feld befindet.
- Ihre Ausführung bewirkt, dass M
  - vom Zustand p in den Zustand q übergeht,
  - auf Band i das Symbol a_i durch b_i ersetzt und
  - den Kopf gemäß D_i bewegt (L: ein Feld nach links, R: ein Feld nach rechts, N: keine Bewegung).

Das Rechenmodell der Turingmaschine

Definition

- Eine Konfiguration ist ein $(3k+1)$-Tupel

  $$K = (q, u_1, a_1, v_1, \ldots, u_k, a_k, v_k) \in Z \times (\Gamma^* \times \Gamma \times \Gamma^*)^k$$

  und besagt, dass
  - q der momentane Zustand ist und
  - das i-te Band mit $\ldots \sqcup u_i a_i v_i \sqcup \ldots$ beschriftet ist, wobei sich der Kopf auf dem Zeichen a_i befindet.
- Im Fall $k = 1$ schreiben wir für eine Konfiguration (q, u, a, v) auch kurz $uqav$.

Definition

Seien $K = (p, u_1, a_1, v_1, \ldots, u_k, a_k, v_k)$ und $K' = (q, u'_1, a'_1, v'_1, \ldots, u'_k, a'_k, v'_k)$ Konfigurationen. K' heißt Folgekonfiguration von K (kurz $K \vdash K'$), falls eine Anweisung $(p, a_1, \ldots, a_k) \to (q, b_1, \ldots, b_k, D_1, \ldots, D_k)$ existiert, so dass für $i = 1, \ldots, k$ gilt:

im Fall $D_i = \mathrm{N}$:	$D_i = \mathrm{R}$:	$D_i = \mathrm{L}$:
K: $u_i \; \boxed{a_i} \; v_i$	K: $u_i \; \boxed{a_i} \; v_i$	K: $u_i \; \boxed{a_i} \; v_i$
K': $u_i \; \boxed{b_i} \; v_i$	K': $u_i \; b_i \; \boxed{a'_i} \; v'_i$	K': $u'_i \; \boxed{a'_i} \; b_i \; v_i$
$u'_i = u_i$, $a'_i = b_i$ und $v'_i = v_i$.	$u'_i = u_i b_i$ und $a'_i v'_i = \begin{cases} v_i, & v_i \neq \varepsilon, \\ \sqcup, & \text{sonst.} \end{cases}$	$u'_i a'_i = \begin{cases} u_i, & u_i \neq \varepsilon, \\ \sqcup, & \text{sonst} \end{cases}$ und $v'_i = b_i v_i$.

Das Rechenmodell der Turingmaschine

Definition

- Die Startkonfiguration von M bei Eingabe $x = x_1 \dots x_n \in \Sigma^*$ ist

$$K_x = \begin{cases} (q_0, \varepsilon, x_1, x_2 \dots x_n, \varepsilon, \sqcup, \varepsilon, \dots, \varepsilon, \sqcup, \varepsilon), & x \neq \varepsilon, \\ (q_0, \varepsilon, \sqcup, \varepsilon, \dots, \varepsilon, \sqcup, \varepsilon), & x = \varepsilon. \end{cases}$$

- Eine Rechnung von M bei Eingabe x ist eine (endliche oder unendliche) Folge von Konfigurationen $K_0, K_1, K_2 \dots$ mit $K_0 = K_x$ und $K_0 \vdash K_1 \vdash K_2 \cdots$.
- Die von M akzeptierte oder erkannte Sprache ist

$$L(M) = \{x \in \Sigma^* \mid \exists\, K \in E \times (\Gamma^* \times \Gamma \times \Gamma^*)^k : K_x \vdash^* K\}.$$

- Ein Wort x wird also genau dann von M akzeptiert (kurz: $M(x)$ akzeptiert), wenn es eine Rechnung von M bei Eingabe x gibt, bei der ein Endzustand erreicht wird.

Beispiel

Betrachte die 1-DTM $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ mit $Z = \{q_0, \dots q_4\}$, $\Sigma = \{a, b\}$, $\Gamma = \Sigma \cup \{A, B, \sqcup\}$, $E = \{q_4\}$ und den Anweisungen

δ: $q_0 a \to q_1 AR$	(1)	Anfang der Schleife: Ersetze das erste a durch A
$q_1 a \to q_1 aR$	(2)	Bewege den Kopf nach rechts bis zum ersten b
$q_1 B \to q_1 BR$	(3)	und ersetze dies durch ein B (falls kein b mehr
$q_1 b \to q_2 BL$	(4)	vorhanden ist, dann halte ohne zu akzeptieren).
$q_2 a \to q_2 aL$	(5)	Bewege den Kopf nach links bis ein A kommt,
$q_2 B \to q_2 BL$	(6)	gehe ein Feld nach rechts zurück und wiederhole
$q_2 A \to q_0 AR$	(7)	die Schleife.
$q_0 B \to q_3 BR$	(8)	Falls kein a am Anfang der Schleife, dann teste,
$q_3 B \to q_3 BR$	(9)	ob noch ein b vorhanden ist. Wenn ja, dann halte
$q_3 \sqcup \to q_4 \sqcup N$	(10)	ohne zu akzeptieren. Andernfalls akzeptiere.

Beispiel (Fortsetzung)

$$\begin{array}{llllllll}
\delta\colon q_0 a \to q_1 AR & (1) & q_1 a \to q_1 aR & (2) & q_2 a \to q_2 aL & (5) & q_0 B \to q_3 BR & (8) \\
 & & q_1 B \to q_1 BR & (3) & q_2 B \to q_2 BL & (6) & q_3 B \to q_3 BR & (9) \\
 & & q_1 b \to q_2 BL & (4) & q_2 A \to q_0 AR & (7) & q_3 \sqcup \to q_4 \sqcup N & (10)
\end{array}$$

- Dann akzeptiert M die Eingabe *aabb* wie folgt:

$$\begin{array}{lllll}
q_0 aabb \underset{(1)}{\vdash} Aq_1 abb & \underset{(2)}{\vdash} Aaq_1 bb & \underset{(4)}{\vdash} Aq_2 aBb & \underset{(5)}{\vdash} q_2 AaBb \\
\underset{(7)}{\vdash} Aq_0 aBb & \underset{(1)}{\vdash} AAq_1 Bb & \underset{(3)}{\vdash} AABq_1 b & \underset{(4)}{\vdash} AAq_2 BB \\
\underset{(6)}{\vdash} Aq_2 ABB & \underset{(7)}{\vdash} AAq_0 BB & \underset{(8)}{\vdash} AABq_3 B & \underset{(9)}{\vdash} AABBq_3 \sqcup \\
\underset{10}{\vdash} AABBq_4 \sqcup & & &
\end{array}$$

- Ähnlich lässt sich für ein beliebiges $n \geq 1$ zeigen, dass $a^n b^n \in L(M)$ ist.

Beispiel (Schluss)

$$\delta\colon \begin{array}{llllllll} q_0 a \to q_1 A R & (1) & q_1 a \to q_1 a R & (2) & q_2 a \to q_2 a L & (5) & q_0 B \to q_3 B R & (8) \\ & & q_1 B \to q_1 B R & (3) & q_2 B \to q_2 B L & (6) & q_3 B \to q_3 B R & (9) \\ & & q_1 b \to q_2 B L & (4) & q_2 A \to q_0 A R & (7) & q_3 \sqcup \to q_4 \sqcup N & (10) \end{array}$$

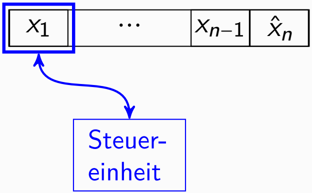
- Andererseits führt die Eingabe *abb* auf die Rechnung

$$q_0 abb \underset{(1)}{\vdash} A q_1 bb \underset{(4)}{\vdash} q_2 ABb \underset{(7)}{\vdash} A q_0 Bb \underset{(8)}{\vdash} AB q_3 b$$

- Da diese nicht fortsetzbar ist und da M deterministisch ist, kann $M(abb)$ nicht den Endzustand q_4 erreichen, d.h. *abb* gehört nicht zu $L(M)$.
- Tatsächlich lässt sich zeigen, dass $L(M) = \{a^n b^n \mid n \geq 1\}$ ist.
- In den Übungen werden wir eine 1-DTM für die Sprache $L = \{a^n b^n c^n \mid n \geq 1\}$ konstruieren. $\triangleleft$

Ein Maschinenmodell für CSL

- Es ist leicht zu sehen, dass jede Typ-0 Sprache von einer NTM M erkannt wird, die ausgehend von der Eingabe x eine Rückwärtsableitung (Reduktion) auf das Startsymbol sucht.
- Im Fall einer Typ-1 Sprache und $x \neq \varepsilon$ ist die linke Seite jeder benutzten Regel höchstens so lang wie die rechte Seite.
- Daher muss M in diesem Fall nur deshalb den Bereich der Eingabe $x = x_1 \cdots x_n$ verlassen, um das Ende von x erkennen zu können.
- Falls wir jedoch das letzte Zeichen x_n von x markieren, kann M die Rechnung auf den Bereich der (markierten) Eingabe beschränken.



- NTMs mit dieser Eigenschaft werden auch als LBAs bezeichnet.

Linear beschränkte Automaten

Definition

- Für ein Alphabet Σ sei $\hat{\Sigma} = \Sigma \cup \{\hat{a} \mid a \in \Sigma\}$.
- Für $x = x_1 \dots x_n \in \Sigma^*$ sei $\hat{x} = x_1 \dots x_{n-1}\hat{x}_n$.
- Eine 1-NTM $M = (Z, \hat{\Sigma}, \Gamma, \delta, q_0, E)$ heißt LBA, falls gilt:

  $$\forall x \in \Sigma^+ : K_{\hat{x}} \vdash^* uqav \Rightarrow |uav| \le |x|$$

- Die von einem LBA M akzeptierte oder erkannte Sprache ist

  $$L(M) = \{x \in \Sigma^* \mid M(\hat{x}) \text{ akzeptiert}\}$$

- Ein deterministischer LBA wird auch als DLBA bezeichnet.
- Die Klasse der deterministisch kontextsensitiven Sprachen ist $\mathrm{DCSL} = \{L(M) \mid M \text{ ist ein DLBA}\}$.

Bemerkung

Jede k-NTM, die bei Eingaben der Länge n höchstens linear viele (also $cn + c$ für eine Konstante c) Bandfelder besucht, kann von einem LBA simuliert werden. LBA steht also für linear beschränkter Automat.

Linear beschränkte Automaten

Beispiel

- Es ist nicht schwer, die 1-DTM $M = (Z, \Sigma, \Gamma, \delta, q_0, E)$ mit

$$\begin{array}{llllllll} \delta\colon q_0 a \to q_1 AR & (1) & q_1 a \to q_1 aR & (2) & q_2 a \to q_2 aL & (5) & q_0 B \to q_3 BR & (8) \\ & & q_1 B \to q_1 BR & (3) & q_2 B \to q_2 BL & (6) & q_3 B \to q_3 BR & (9) \\ & & q_1 b \to q_2 BL & (4) & q_2 A \to q_0 AR & (7) & q_3 \sqcup \to q_4 \sqcup N & (10) \end{array}$$

  in einen DLBA $M' = (Z,\ \hat{\Sigma},\ \Gamma',\ \delta',\ q_0,\ E)$ für die Sprache $\{a^n b^n \mid n \geq 1\}$ umzuwandeln.

- Ersetze hierzu
  - $\Sigma = \{a, b\}$ durch $\hat{\Sigma} = \{a, b, \hat{a}, \hat{b}\}$,
  - $\Gamma = \Sigma \cup \{A, B, \sqcup\}$ durch $\Gamma' = \hat{\Sigma} \cup \{A, B, \hat{B}, \sqcup\}$ sowie
  - die Anweisung $q_3 \sqcup \to q_4 \sqcup N$ (10) durch $q_3 \hat{B} \to q_4 \hat{B} N$ (10′) und

  füge die Anweisungen $q_1 \hat{b} \to q_2 \hat{B} L$ (4a) und $q_0 \hat{B} \to q_4 \hat{B} N$ (8a) hinzu:

Linear beschränkte Automaten

Beispiel

- Ersetze die Anweisung $q_3\sqcup \to q_4\sqcup N$ (10) durch $q_3\hat{B} \to q_4\hat{B}N$ $(10')$ und füge die Anweisungen $q_1\hat{b} \to q_2\hat{B}L$ $(4a)$ und $q_0\hat{B} \to q_4\hat{B}N$ $(8a)$ hinzu:

$$\begin{array}{llllll} \delta'\colon q_0a \to q_1AR & (1) & q_1\hat{b} \to q_2\hat{B}L & (4a) & q_0B \to q_3BR & (8) \\ q_1a \to q_1aR & (2) & q_2a \to q_2aL & (5) & q_0\hat{B} \to q_4\hat{B}N & (8a) \\ q_1B \to q_1BR & (3) & q_2B \to q_2BL & (6) & q_3B \to q_3BR & (9) \\ q_1b \to q_2BL & (4) & q_2A \to q_0AR & (7) & q_3\hat{B} \to q_4\hat{B}N & (10') \end{array}$$

- Dann akzeptiert M' die Eingabe $aab\hat{b}$ wie folgt (d.h. $aabb \in L(M')$):

$$\begin{array}{lllllllll} q_0aab\hat{b} & \underset{(1)}{\vdash} & Aq_1ab\hat{b} & \underset{(2)}{\vdash} & Aaq_1b\hat{b} & \underset{(4)}{\vdash} & Aq_2aB\hat{b} & \underset{(5)}{\vdash} & q_2AaB\hat{b} \\ & \underset{(7)}{\vdash} & Aq_0aB\hat{b} & \underset{(1)}{\vdash} & AAq_1B\hat{b} & \underset{(3)}{\vdash} & AABq_1\hat{b} & \underset{(4a)}{\vdash} & AAq_2B\hat{B} \\ & \underset{(6)}{\vdash} & Aq_2AB\hat{B} & \underset{(7)}{\vdash} & AAq_0B\hat{B} & \underset{(8)}{\vdash} & AABq_3\hat{B} & \underset{(10')}{\vdash} & AABq_4\hat{B} \end{array}$$

Linear beschränkte Automaten

Beispiel

- Ersetze die Anweisung $q_3 \sqcup \to q_4 \sqcup N$ (10) durch $q_3 \hat{B} \to q_4 \hat{B} N$ (10′) und füge die Anweisungen $q_1 \hat{b} \to q_2 \hat{B} L$ (4a) und $q_0 \hat{B} \to q_4 \hat{B} N$ (8a) hinzu:

$$\delta': \begin{array}{llllll} q_0 a \to q_1 AR & (1) & q_1 \hat{b} \to q_2 \hat{B} L & (4a) & q_0 B \to q_3 BR & (8) \\ q_1 a \to q_1 aR & (2) & q_2 a \to q_2 aL & (5) & q_0 \hat{B} \to q_4 \hat{B} N & (8a) \\ q_1 B \to q_1 BR & (3) & q_2 B \to q_2 BL & (6) & q_3 B \to q_3 BR & (9) \\ q_1 b \to q_2 BL & (4) & q_2 A \to q_0 AR & (7) & q_3 \hat{B} \to q_4 \hat{B} N & (10') \end{array}$$

- Die Eingabe $a\hat{b}$ wird ebenfalls akzeptiert:

$$q_0 a\hat{b} \underset{(1)}{\vdash} A q_1 \hat{b} \underset{(4a)}{\vdash} q_2 A \hat{B} \underset{(7)}{\vdash} A q_0 \hat{B} \underset{(8a)}{\vdash} A q_4 \hat{B}$$

◁

Das LBA-Problem

Bemerkung

- Der DLBA M' für die Sprache $\{a^n b^n \mid n \geq 1\}$ aus obigem Beispiel lässt sich leicht in einen DLBA für die kontextsensitive Sprache $\{a^n b^n c^n \mid n \geq 1\}$ transformieren (siehe Übungen).
- Die Sprache $\{a^n b^n c^n \mid n \geq 1\}$ liegt also in DCSL $\setminus$ CFL.
- Bis heute ungelöst ist die Frage, ob die Klasse DCSL eine echte Teilklasse von CSL ist oder nicht?
- Diese Fragestellung ist als LBA-Problem bekannt.

Charakterisierung von CSL mittels LBAs

Als nächstes zeigen wir, dass LBAs genau die kontextsensitiven Sprachen erkennen.

Satz

CSL = $\{L(M) \mid M$ ist ein LBA$\}$.

Beweis von CSL $\subseteq \{L(M) \mid M \text{ ist ein LBA}\}$

Sei $G = (V, \Sigma, P, S)$ eine kontextsensitive Grammatik. Dann wird $L(G)$ von folgendem LBA M akzeptiert (o.B.d.A. sei $\varepsilon \notin L(G)$):

Arbeitsweise von M bei Eingabe $\hat{x} = x_1 \ldots x_{n-1}\hat{x}_n$ mit $n > 0$:

1 Markiere das erste Eingabezeichen x_1 mittels $\tilde{x}_1$ (bzw. $\hat{x}_1$ mittels $\tilde{\hat{x}}_1$)
2 Wähle (nichtdeterministisch) eine Regel $\alpha \to \beta$ aus P
3 Wähle ein beliebiges Vorkommen von β auf dem Band
(falls β nicht vorkommt, halte ohne zu akzeptieren)
4 Ersetze die ersten $|\alpha|$ Zeichen von β durch α
5 Falls das erste (oder letzte) Zeichen von β markiert war,
markiere auch das erste (letzte) Zeichen von α
6 Verschiebe die Zeichen rechts von β um $|\beta| - |\alpha|$ Positionen nach
links und überschreibe die frei werdenden Felder mit Blanks
7 Enthält das Band nur noch das (doppelt markierte) Startsymbol
gefolgt von Blanks, so halte in einem Endzustand
8 Gehe zurück zu Schritt 2

Beweis von CSL $\subseteq \{L(M) \mid M$ ist ein LBA$\}$

- Nun ist leicht zu sehen, dass M wegen $|\beta| \geq |\alpha|$ tatsächlich ein LBA ist.
- M akzeptiert x, falls es gelingt, eine Ableitung für x in G zu finden (in umgekehrter Reihenfolge).
- Da sich genau für die Wörter in $L(G)$ eine Ableitung finden lässt, folgt $L(M) = L(G)$. $\square$

Beweis von $\{L(M) \mid M \text{ ist ein LBA}\} \subseteq \mathrm{CSL}$

- Sei $M = (Z, \hat{\Sigma}, \Gamma, \delta, q_0, E)$ ein LBA (o.B.d.A. sei $\varepsilon \notin L(M)$).
- Betrachte die kontextsensitive Grammatik $G = (V, \Sigma, P, S)$ mit

$$V = \{S, A\} \cup (Z\Gamma \cup \Gamma) \times \Sigma = \{S, A, (qc, a), (c, a) \mid q \in Z, c \in \Gamma, a \in \Sigma\},$$

die für alle $a, b \in \Sigma$ und $c, d \in \Gamma$ folgende Regeln enthält:

$$\begin{array}{lrll}
P: & S \to A(\hat{a}, a),\ (q_0\hat{a}, a) & & \text{(S)} \quad \text{„Startregeln“} \\
& A \to A(a, a),\ (q_0 a, a) & & \text{(A)} \quad \text{„A-Regeln“} \\
& (c, a) \to a & & \text{(F)} \quad \text{„Finale Regeln“} \\
& (qc, a) \to a, & \text{falls } q \in E & \text{(E)} \quad \text{„E-Regeln“} \\
& (qc, a) \to (q'c', a), & \text{falls } qc \to_M q'c'N & \text{(N)} \quad \text{„N-Regeln“} \\
& (qc, a)(d, b) \to (c', a)(q'd, b), & \text{falls } qc \to_M q'c'R & \text{(R)} \quad \text{„R-Regeln“} \\
& (d, a)(qc, b) \to (q'd, a)(c', b), & \text{falls } qc \to_M q'c'L & \text{(L)} \quad \text{„L-Regeln“}
\end{array}$$