Kontextfreie Sprachen

- Bedeutung: Programmiersprachen (Compilerbau)
- Syntaxbäume
- Chomsky-Normalform
- effiziente Lösung des Wortproblems (CYK-Algorithmus)
- Grenzen kontextfreier Sprachen (Pumping Lemma)
- Charakterisierung durch Kellerautomaten
- Deterministisch kontextfreie Sprachen

Beispiel für eine kontextfreie Grammatik

$$G = (\{S, X, C\}, \{x, c, 0, +, -, ;\, , :=, \neq, \mathrm{LOOP}, \mathrm{WHILE}, \mathrm{DO}, \mathrm{END}\}, P, S)$$

mit folgenden Regeln in der Regelmenge P:

$$\begin{aligned}
S &\to S; S \\
S &\to \mathrm{LOOP}\ X\ \mathrm{DO}\ S\ \mathrm{END} \\
S &\to \mathrm{WHILE}\ X \neq 0\ \mathrm{DO}\ S\ \mathrm{END} \\
S &\to X := X + C \\
S &\to X := X - C \\
X &\to x \\
C &\to c
\end{aligned}$$

Weiteres Beispiel für eine kontextfreie Grammatik

$$G = (\{S\}, \{a_1, a_2, b_1, b_2\}, P, S)$$

mit der Regelmenge $P = \{S \to SS, S \to a_1Sb_1, S \to a_2Sb_2, S \to \varepsilon\}$.

G erzeugt die Sprache D_2, die sogenannte *Dyck-Sprache* über zwei Klammerpaaren. Induktive Definition von D_2:

1. $\varepsilon \in D_2$.
2. Aus $w_1 \in D_2, w_2 \in D_2$ folgt $w_1 w_2 \in D_2$.
3. Aus $w \in D_2$ folgt $a_1 w b_1 \in D_2$ und $a_2 w b_2 \in D_2$.
4. D_2 enthält keine weiteren Wörter.

Syntaxbäume

Jeder Ableitung eines Wortes w in einer kontextfreien Grammatik kann eindeutig ein Syntaxbaum zugeordnet werden.

Sei $w \in L(G)$ und $S = w_0 \Longrightarrow w_1 \Longrightarrow w_2 \Longrightarrow \cdots \Longrightarrow w_n = w$ eine Ableitung für w. Dann wird der Syntaxbaum folgendermaßen konstruiert:

- Die Wurzel hat die Beschriftung S.
- Nach dem i-ten Schritt ergeben die Beschriftungen der Blätter von links nach rechts gelesen das Wort w_i, $0 \leq i \leq n$.
- Wird bei der Ableitung eine Regel $A \to \alpha$ angewendet, so erhält das zugehörige Blatt (mit der Beschriftung A) $|\alpha|$ Söhne, deren Beschriftung von links nach rechts das Wort α ergibt.

Linksableitungen

Definition. Eine Ableitung in einer kontextfreien Grammatik heißt Linksableitung, wenn in jedem Schritt das am weitesten links stehende Nichtterminalsymbol ersetzt wird.

Jedem Syntaxbaum zu einer Ableitung kann eindeutig eine Linksableitung zugeordnet werden, d.h.:

> Satz
> Die von einer kontextfreien Grammatik G erzeugte Sprache ist die Menge der durch Linksableitungen in G erzeugbaren Wörter.

Chomsky Normalform

Definition

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist in Chomsky Normalform, falls jede Regel in P in einer der Formen (i)-(iii) ist:

(i) $A \to BC$ mit $A, B, C \in V$,

(ii) $A \to a$ mit $A \in V$ und $a \in \Sigma$,

(iii) $S \to \varepsilon$, wobei S auf keiner rechten Seite einer Regel vorkommt.

Satz

Zu jeder kontextfreien Grammatik G kann man eine kontextfreie Grammatik G' in Chomsky Normalform konstruieren mit $L(G) = L(G')$.

Der Algorithmus von Cocke, Younger und Kasami

Eingabe: kontextfreie Grammatik $G = (V, \Sigma, P, S)$ in Chomsky NF,
Wort $w \in \Sigma^*$

Ausgabe: *ja*, falls $w \in L(G)$; *nein*, sonst.

Grundidee des CYK-Algorithmus

- $w[s,t]$ sei das Teilwort von w von der Stelle s bis zur Stelle t, wobei $1 \leq s \leq t \leq n = |w|$.
- Bestimme für $1 \leq j \leq n$ und $1 \leq i \leq n - j + 1$ die Mengen

$$T[i,j] = \{A \in V \mid A \stackrel{*}{\Longrightarrow} w[i, i+j-1]\}.$$

- Es gilt $w \in L(G)$ genau dann, wenn $S \in T[1,n]$.

CYK-Algorithmus – Fortsetzung

Induktive Bestimmung der Mengen $T[i,j]$:

- Für $j = 1$: $T[i,1] = \{A \in V \mid A \to w[i,i] \in P\}$
- Für $j > 1$:

$$T[i,j] = \{A \in V \mid \text{es gibt } B, C \in V \text{ und } 1 \leq k < j \text{ mit}$$
$$A \to BC \in P, B \in T[i,k], C \in T[i+k, j-k]\}$$

- CYK-Algorithmus ist Beispiel für Dynamische Programmierung.

Beispiel für CYK-Algorithmus

Die Sprache $L = \{a^n b^n c^m \mid m,n \geq 1\}$ ist kontextfrei und wird von der Grammatik $G = (V, \Sigma, P, S)$ mit den Regeln

$$S \to AB, \qquad A \to aAb \mid ab, \qquad B \to cB \mid c$$

erzeugt. Eine äquivalente Grammatik in Chomsky Normalform für L hat folgende Menge von Regeln:

$$\begin{array}{llll} S \to AB, & B \to EB \mid c, & C \to a, & \\ A \to CD \mid CF, & F \to AD, & D \to b, & E \to c. \end{array}$$

Sei $x = aaabbbcc$. Dann erzeugt der Algorithmus folgende Tabelle.

Beispiel für CYK-Algorithmus – Tabelle

i

a	a	a	b	b	b	c	c
C	C	C	D	D	D	E, B	E, B
		A				B	
		F					
	A						
	F						
A							
S							
S							

$= x$

j

Beispiel für CYK-Algorithmus – Erklärung

In der Tabelle stellt jedes Feld eine Menge $T[i, j]$ von Nichtterminalen dar. Die Koordinatenachsen für i und j sind eingezeichnet.

Der Algorithmus besteht aus drei Teilen. Im ersten Teil wird die oberste Zeile ($T[i, 1]$) berechnet.

Im zweiten Teil werden zeilenweise die weiteren Mengen $T[i, j]$ berechnet, indem immer die zugehörige Spalte von oben und die nach rechts oben führende Diagonale betrachtet werden.

Im dritten Teil des Algorithmus schließlich wird die Menge $T[1, n]$ betrachtet, hier $T[1, 8]$. Ausgabe “ja” genau dann, wenn $S \in T[1, n]$.

Fortsetzung CYK-Algorithmus

Noch eine Bemerkung: aus der aufgestellten Tabelle kann man auch die Ableitung für das Wort x ablesen, indem wir rückwärts die Regeln anwenden, die auf S in $T[1,n]$ geführt haben. Hier sieht die Ableitung für $aaabbbc$ dann folgendermaßen aus (die Nonterminale, die ersetzt werden, sind jeweils unterstrichen).

$$\underline{S} \Longrightarrow \underline{A}B \Longrightarrow C\underline{F}B \Longrightarrow C\underline{A}DB \Longrightarrow CC\underline{F}DB \Longrightarrow CC\underline{A}DDB$$
$$\Longrightarrow \underline{C}CCDDDB \Longrightarrow a\underline{C}CDDDB \Longrightarrow aa\underline{C}DDDB \Longrightarrow aaa\underline{D}DDB$$
$$\Longrightarrow aaab\underline{D}DB \Longrightarrow aaabb\underline{D}B \Longrightarrow aaabbb\underline{B} \Longrightarrow aaabbbc$$

Pumping Lemma für kontextfreie Sprachen

Sei L eine kontextfreie Sprache. Dann gibt es eine Konstante $k \in \mathbb{N}$, so dass für alle Wörter $z \in L$ mit $|z| \geq k$ eine Zerlegung $z = uvwxy$ existiert, so dass gilt:

(i) $|vwx| \leq k$ und $|vx| \geq 1$,

(ii) für alle $i \in \mathbb{N}$ gilt $uv^iwx^iy \in L$.

Pumping Lemma – Anwendung

Mit Hilfe des Pumping Lemmas kann man zeigen, dass folgende Sprachen nicht kontextfrei sind:

- $\{a^n b^n c^n \mid n \geq 1\}$,
- $\{ww \mid w \in \{a,b\}^*\}$,
- $\{a^m b^n a^m b^n \mid m, n \geq 1\}$,
- $\{a^{2^n} \mid n \geq 0\}$,
- $\{a^m b^n \mid 0 \leq m, 1 \leq n \leq 2^m\}$.

Abschlusseigenschaften

Satz

Die Menge der kontextfreien Sprachen ist unter den Operationen

(i) Vereinigung,

(ii) Produkt (Konkatenation) und

(iii) Kleene-Stern

abgeschlossen. Sie ist unter den Operationen

(iv) Durchschnitt und

(v) Komplement

nicht abgeschlossen.

Weitere Entscheidungsprobleme

Satz

Das Leerheitsproblem und das Endlichkeitsproblem für kontextfreie Grammatiken sind entscheidbar.

Satz

Das Schnittproblem und das Äquivalenzproblem für kontextfreie Grammatiken sind unentscheidbar.

Kellerautomaten

- endliche Automaten besitzen nur begrenzten Speicherplatz (ihre Zustände).
- Turingmaschinen besitzen unbegrenzten Speicherplatz mit im Prinzip wahlfreiem Zugriff.
- Kellerautomaten besitzen unbegrenzten Speicherplatz in Form eines Kellerspeichers (Stack).

Definition Kellerautomat

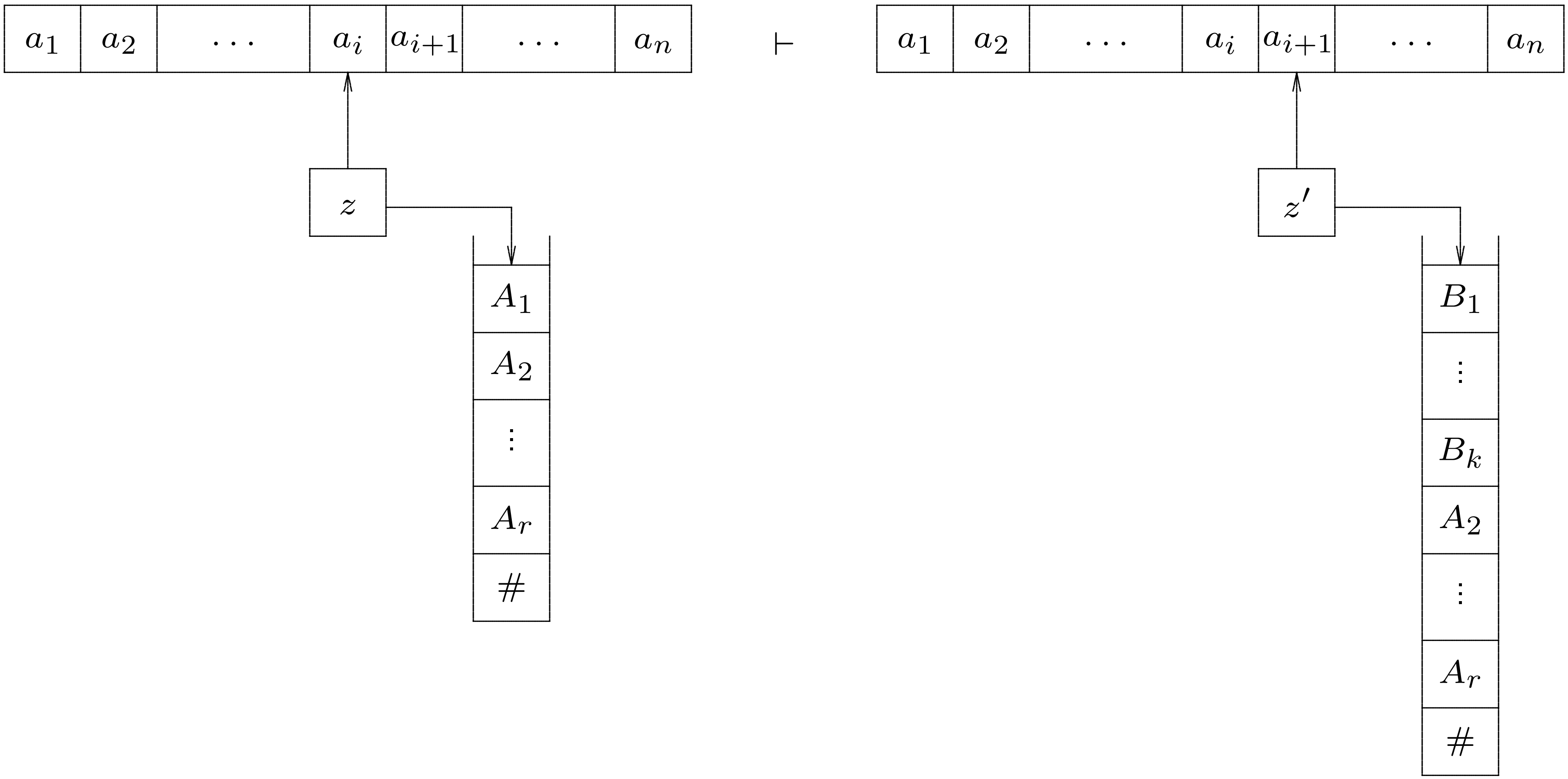
Definition Ein (nichtdeterministischer) Kellerautomat (kurz PDA von *push-down automaton*) M ist ein 6-Tupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$. Dabei ist

- Z das Zustandsalphabet,
- Σ das Eingabealphabet,
- Γ das Kelleralphabet,
- $z_0 \in Z$ der Anfangszustand,
- $\delta\colon Z \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2_E^{Z \times \Gamma^*}$ die Zustandsüberführungsfunktion (2_E^X bedeutet dabei die Menge der endlichen Teilmengen von X),
- $\# \in \Gamma$ das Kelleranfangssymbol.

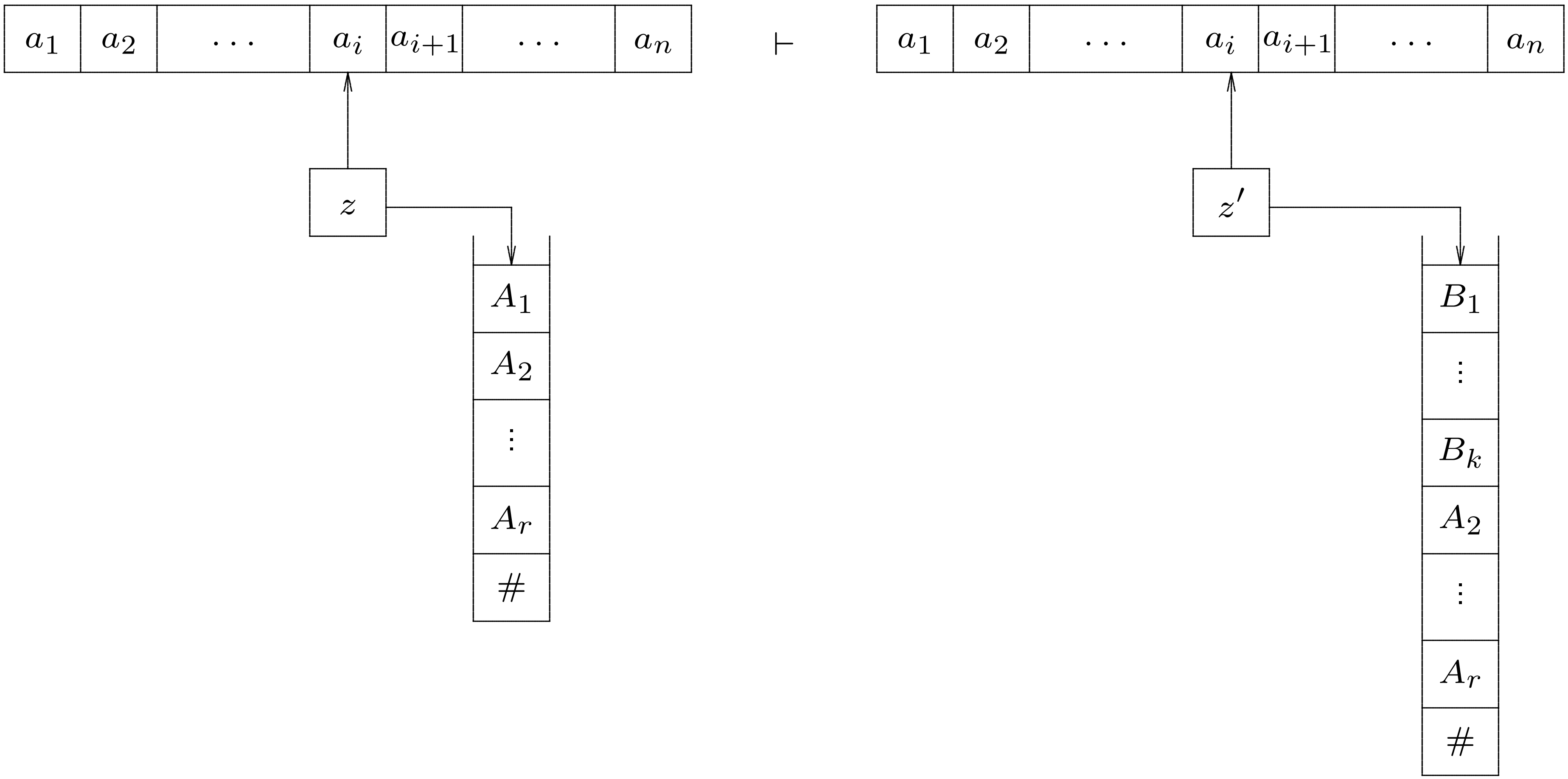
Arbeitsweise von Kellerautomaten

- Der PDA startet im Zustand z_0 und mit dem Kellerinhalt $\#$.
- Bei $(z', B_1B_2 \dots B_k) \in \delta(z, a, A)$ mit $a \in \Sigma$ befindet sich der PDA im Zustand z, liest auf dem Eingabeband ein a und im Keller ein A (das oberste Symbol). Er entfernt dieses A aus dem Keller, geht in den Zustand z' über, bewegt den Lesekopf auf dem Eingabeband einen Schritt nach rechts und schreibt auf den Keller das Wort $B_1B_2 \dots B_k$ derart, dass B_1 das neue oberste Symbol des Kellers ist.
- Bei $(z', B_1B_2 \dots B_k) \in \delta(z, \varepsilon, A)$ liest er im Gegensatz zur obigen Aktion auf dem Eingabeband nichts ein und bewegt den Lesekopf auf dem Eingabeband nicht.
- Der PDA akzeptiert die Eingabe, wenn er sie vollständig eingelesen hat und der Keller leer ist (auch $\#$ steht nicht mehr im Keller).

Kellerautomat: Arbeitsweise für $(z', B_1 B_2 \cdots B_k) \in \delta(z, a_i, A_1)$



Kellerautomat: Arbeitsweise für $(z', B_1 B_2 \cdots B_k) \in \delta(z, \varepsilon, A_1)$



Konfiguration von Kellerautomaten

Definition Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ ein Kellerautomat. Eine Konfiguration von M ist ein Tripel $k \in Z \times \Sigma^* \times \Gamma^*$.

Anschaulich beschreibt eine Konfiguration $k = (z, v, \gamma)$ folgende augenblickliche Situation des PDA:

- er befindet sich im Zustand z,
- v ist die noch nicht verarbeitete Eingabe auf dem Eingabeband und
- im Keller steht das Wort γ, wobei das erste Symbol von γ das oberste Kellersymbol sein soll.

PDA – Konfigurationsübergänge und akzeptierte Sprache

$k_1 \vdash k_2$ bedeutet, dass der Kellerautomat die Konfiguration k_1 in genau einem Schritt in die Konfiguration k_2 überführt.

$k_1 \vdash^* k_2$ bedeutet, dass der Kellerautomat k_1 in endlich vielen Schritten (auch null) in k_2 überführt.

Definition

Für einen PDA $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ sei die von ihm akzeptierte Sprache $T(M)$ definiert durch

$$T(M) = \{w \in \Sigma^* \mid (z_0, w, \#) \vdash^* (z, \varepsilon, \varepsilon) \text{ für ein } z \in Z\}.$$

Beispiel eines PDA

Es sei $M = (\{z_0, z_1\}, \{a, b\}, \{A, B, \#\}, \delta, z_0, \#)$ ein Kellerautomat, mit

$$\begin{aligned}
\delta(z_0, a, X) &= \{(z_0, AX)\} \qquad \text{für } X \in \{B, \#\},\\
\delta(z_0, a, A) &= \{(z_0, AA), (z_1, \varepsilon)\},\\
\delta(z_0, b, X) &= \{(z_0, BX)\} \qquad \text{für } X \in \{A, \#\},\\
\delta(z_0, b, B) &= \{(z_0, BB), (z_1, \varepsilon)\},\\
\delta(z_0, \varepsilon, \#) &= \{(z_1, \varepsilon)\},\\
\delta(z_1, a, A) &= \{(z_1, \varepsilon)\},\\
\delta(z_1, b, B) &= \{(z_1, \varepsilon)\},\\
\delta(z_1, \varepsilon, \#) &= \{(z_1, \varepsilon)\}.
\end{aligned}$$

Dann gilt $T(M) = \{ww^R \mid w \in \{a, b\}^*\}$.

Kellerautomaten und kontextfreie Sprachen

Satz
Eine Sprache L ist kontextfrei genau dann, wenn ein (nichtdeterministischer) Kellerautomat M mit $T(M) = L$ existiert.

Deterministisch kontextfreie Sprachen

Definition. Ein Kellerautomat $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ heißt deterministisch, wenn:

$$|\delta(z, a, A)| + |\delta(z, \varepsilon, A)| \leq 1 \text{ für alle } z \in Z, a \in \Sigma, A \in \Gamma.$$

Definition. Eine Sprache $L \subseteq \Sigma^*$ heißt deterministisch kontextfrei, wenn $L\$$ von einem deterministischen Kellerautomaten akzeptiert wird, wobei $\$ \notin \Sigma$ ein Sondersymbol ist.

Satz
Die Menge der deterministisch kontextfreien Sprachen ist eine echte Teilmenge der Menge der kontextfreien Sprachen.

Beispiel eines deterministischen PDA

$M = (\{z_0\}, \{a_1, a_2, b_1, b_2, \$\}, \{A_1, A_2, \#\}, \delta, z_0, \#)$ ein Kellerautomat mit

$$\begin{aligned}
\delta(z_0, a_1, X) &= \{(z_0, A_1X)\} && \text{für } X \in \{A_1, A_2, \#\}, \\
\delta(z_0, a_2, X) &= \{(z_0, A_2X)\} && \text{für } X \in \{A_1, A_2, \#\}, \\
\delta(z_0, b_1, A_1) &= \{(z_0, \varepsilon)\}, \\
\delta(z_0, b_2, A_2) &= \{(z_0, \varepsilon)\}, \\
\delta(z_0, \$, \#) &= \{(z_0, \varepsilon)\}.
\end{aligned}$$

$T(M) = \{w\$ \mid w \in D_2\}$, d.h.
die Dyck-Sprache D_2 ist deterministisch kontextfrei.

Weiteres Beispiel eines deterministischen PDA

Sei $M = (\{z_0, z_1\}, \{a, b, c, \$\}, \{A, B, \#\}, \delta, z_0, \#)$ ein Kellerautomat mit

$$
\begin{aligned}
\delta(z_0, a, X) &= \{(z_0, AX)\} \qquad \text{für } X \in \{A, B, \#\},\\
\delta(z_0, b, X) &= \{(z_0, BX)\} \qquad \text{für } X \in \{A, B, \#\},\\
\delta(z_0, c, X) &= \{(z_1, X)\} \qquad \text{für } X \in \{A, B, \#\},\\
\delta(z_1, a, A) &= \{(z_1, \varepsilon)\},\\
\delta(z_1, b, B) &= \{(z_1, \varepsilon)\},\\
\delta(z_1, \$, \#) &= \{(z_1, \varepsilon)\}.
\end{aligned}
$$

Dann gilt $T(M) = \{wcw^R\$ \mid w \in \{a, b\}^*\}$.

Bedeutung der deterministisch kontextfreien Sprachen

- Wortproblem entscheidbar in linearer Zeit
(gegenüber kubischer Laufzeit des CYK-Algorithmus)
- Ausdruckskraft hinreichend für syntaktische Analyse im Compilerbau
- Bei realen Programmiersprachen kommen $LR(k)$-Grammatiken und deren Varianten zum Einsatz.

Formale Sprachen – Übersichten

Chomsky-Hierarchie: $\boxed{\textbf{Typ 3} \subsetneq \textbf{Typ 2} \subsetneq \textbf{Typ 1} \subsetneq \textbf{Typ 0}.}$

Beispiele für die Echtheit der Inklusionen.

- $\{a^n b^n \mid n \geq 1\} \in$ **Typ 2** $\setminus$ **Typ 3**,
- $\{a^n b^n c^n \mid n \geq 1\} \in$ **Typ 1** $\setminus$ **Typ 2**,
- $K \in$ **Typ 0** $\setminus$ **Typ 1**,
  wobei $K \subseteq \{0,1\}^*$ das spezielle Halteproblem ist.

Charakterisierungen der Sprachfamilien der Chomsky Hierarchie

Sprache	Grammatik	Automat	Andere
Regulär	Typ 3	Endlicher Automat (EA)	Regulärer Ausdruck
Deterministisch kontextfrei	$LR(k)$	Deterministischer Kellerautomat (DPDA)	
Kontextfrei	Typ 2	(Nichtdeterministischer) Kellerautomat (PDA)	
Kontextabhängig	Typ 1	(Nichtdeterministischer) Linear beschränkter Automat (LBA)	
Rekursiv aufzählbar	Typ 0	Turingmaschine (TM)	

Determinismus vs. Nichtdeterminismus

Nichtdeterminismus	Determinismus	Äquivalenz
NEA	DEA	ja
PDA	DPDA	nein
LBA	DLBA	?
NTM	DTM	ja

Abschlusseigenschaften

	Typ 3	Det. kf.	Typ 2	Typ 1	Typ 0
Vereinigung	ja	nein	ja	ja	ja
Durchschnitt	ja	nein	nein	ja	ja
Komplement	ja	ja	nein	ja	nein
Konkatenation	ja	nein	ja	ja	ja
Kleene-Stern	ja	nein	ja	ja	ja

Entscheidbarkeitsprobleme

	Typ 3	Det. kf.	Typ 2	Typ 1	Typ 0
Wortproblem	ja	ja	ja	ja	nein
Leerheitsproblem	ja	ja	ja	nein	nein
Schnittproblem	ja	nein	nein	nein	nein
Äquivalenzproblem	ja	ja	nein	nein	nein