Teil IV

Kontextfreie Sprachen, Teil 4: Kellerautomaten

Kellerautomaten (Idee)

Ziel

Automatenmodell für CFL

Kellerautomaten (Idee)

Ziel

Automatenmodell für CFL

Ansatz

Endlicher Automat mit unendlichem Speicher.

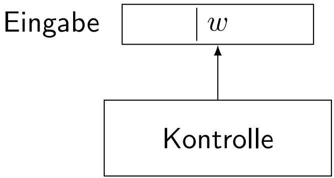
Kellerautomaten (Idee)

Ziel

Automatenmodell für CFL

Ansatz

Endlicher Automat mit unendlichem Speicher.



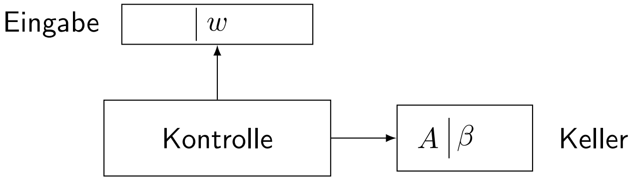
Kellerautomaten (Idee)

Ziel

Automatenmodell für CFL

Ansatz

Endlicher Automat mit unendlichem Speicher.



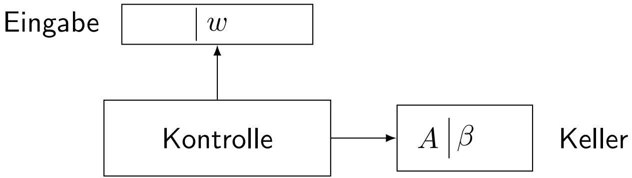
Kellerautomaten (Idee)

Ziel

Automatenmodell für CFL

Ansatz

Endlicher Automat mit unendlichem Speicher.



Kellerautomat (PDA)

- ε-NFA mit Keller
- Keller: unendlich, aber LIFO
- liest Eingabe und oberstes Kellersymbol

Kellerautomaten (formal)

Definition

Ein **Kellerautomat (PDA)** A über einem Alphabet Σ wird definiert durch:

1. ein Alphabet Γ von **Kellersymbolen**
2. eine nicht-leere, endliche Menge Q von **Zuständen**,
3. eine Funktion $\delta : (Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma) \to \mathcal{P}_F(Q \times \Gamma^*)$ (**Übergangsrelation**),
4. einen Zustand $q_0 \in Q$ (**Startzustand**),
5. einem Kellersymbol $K_0 \in \Gamma$ (**Startsymbol**),
6. eine Menge $F \subseteq Q$ von **akzeptierenden Zuständen**.

Wir schreiben dies als $A := (\Sigma, \Gamma, Q, \delta, q_0, K_0, F)$.

Berechnungen

Konfiguration

- **Konfiguration** von A: 3-Tupel aus $Q \times \Sigma^* \times \Gamma^*$
- Zustand, verbleibende Eingabe, Kellerinhalt
- Startkonfiguration mit Eingabe w: (q_0, w, K_0)

Berechnungen

Konfiguration

- **Konfiguration** von A: 3-Tupel aus $Q \times \Sigma^* \times \Gamma^*$
- Zustand, verbleibende Eingabe, Kellerinhalt
- Startkonfiguration mit Eingabe w: (q_0, w, K_0)

Erreichbarkeitsrelation

Für alle $q \in Q$, $a \in \Sigma$, $w \in \Sigma^*$, $A \in \Gamma$, $\beta \in \Gamma^*$ gilt:

1. $(q, aw, A\beta) \vdash_A (p, w, \gamma\beta)$, wenn $(p, \gamma) \in \delta(q, a, A)$, und

Berechnungen

Konfiguration

- **Konfiguration** von A: 3-Tupel aus $Q \times \Sigma^* \times \Gamma^*$
- Zustand, verbleibende Eingabe, Kellerinhalt
- Startkonfiguration mit Eingabe w: (q_0, w, K_0)

Erreichbarkeitsrelation

Für alle $q \in Q$, $a \in \Sigma$, $w \in \Sigma^*$, $A \in \Gamma$, $\beta \in \Gamma^*$ gilt:

1. $(q, aw, A\beta) \vdash_A (p, w, \gamma\beta)$, wenn $(p, \gamma) \in \delta(q, a, A)$, und
2. $(q, w, A\beta) \vdash_A (p, w, \gamma\beta)$, wenn $(p, \gamma) \in \delta(q, \varepsilon, A)$.

Berechnungen

Konfiguration

- **Konfiguration** von A: 3-Tupel aus $Q \times \Sigma^* \times \Gamma^*$
- Zustand, verbleibende Eingabe, Kellerinhalt
- Startkonfiguration mit Eingabe w: (q_0, w, K_0)

Erreichbarkeitsrelation

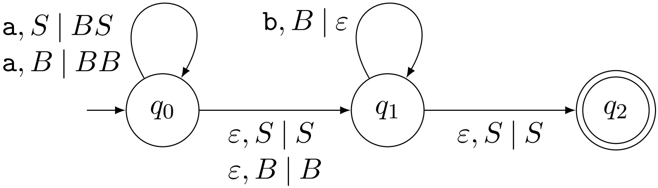
Für alle $q \in Q$, $a \in \Sigma$, $w \in \Sigma^*$, $A \in \Gamma$, $\beta \in \Gamma^*$ gilt:

1. $(q, aw, A\beta) \vdash_A (p, w, \gamma\beta)$, wenn $(p, \gamma) \in \delta(q, a, A)$, und
2. $(q, w, A\beta) \vdash_A (p, w, \gamma\beta)$, wenn $(p, \gamma) \in \delta(q, \varepsilon, A)$.

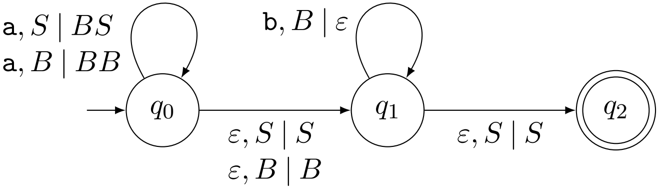
Sprache(n) eines PDA

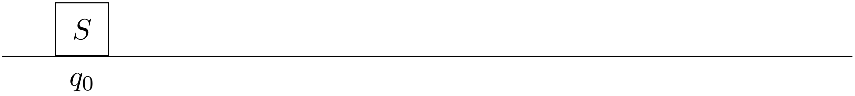
$$\mathcal{L}_Z(A) := \{w \in \Sigma^* \mid (q_0, w, K_0) \vdash_A^* (q, \varepsilon, \gamma), q \in F, \gamma \in \Gamma^*\},$$
$$\mathcal{L}_K(A) := \{w \in \Sigma^* \mid (q_0, w, K_0) \vdash_A^* (q, \varepsilon, \varepsilon), q \in Q\}.$$

Beispiel (Akzeptanz mit akzeptierenden Zuständen)

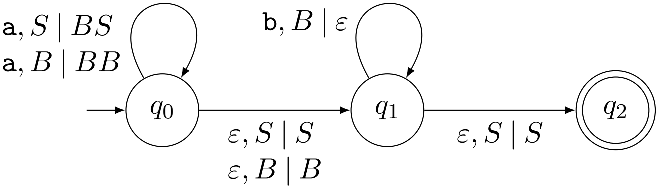


Beispiel (Akzeptanz mit akzeptierenden Zuständen)

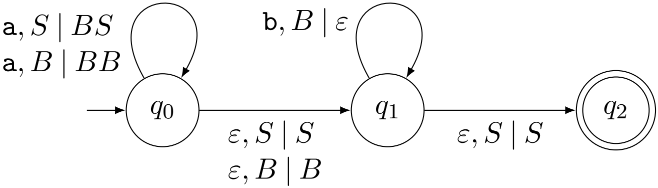




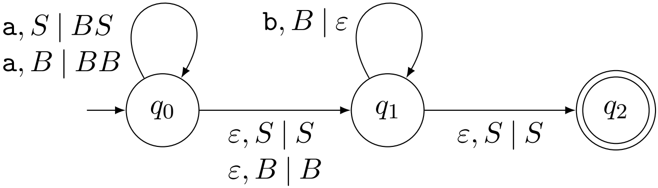
Beispiel (Akzeptanz mit akzeptierenden Zuständen)

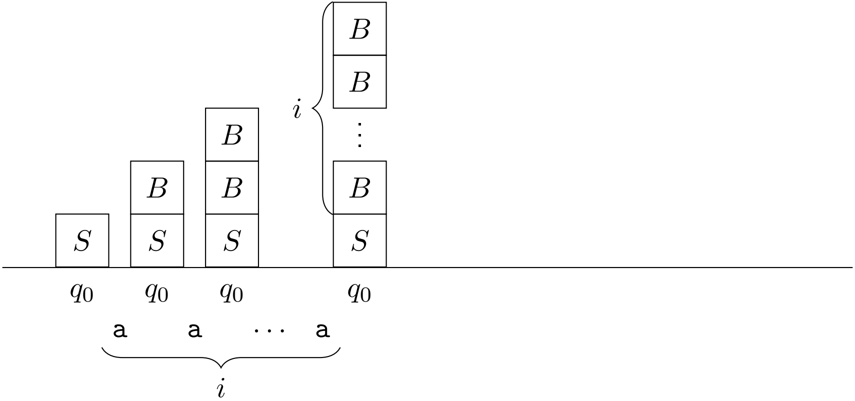


Beispiel (Akzeptanz mit akzeptierenden Zuständen)

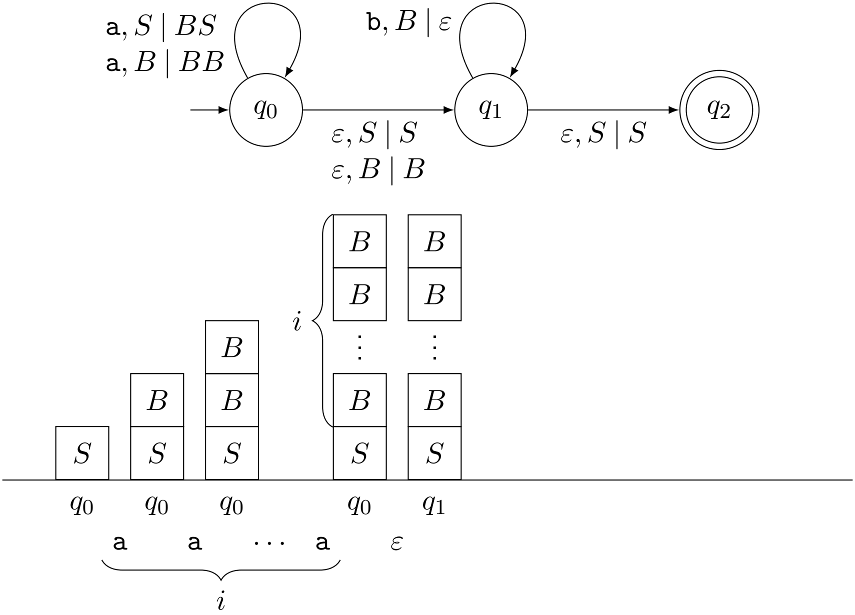


Beispiel (Akzeptanz mit akzeptierenden Zuständen)

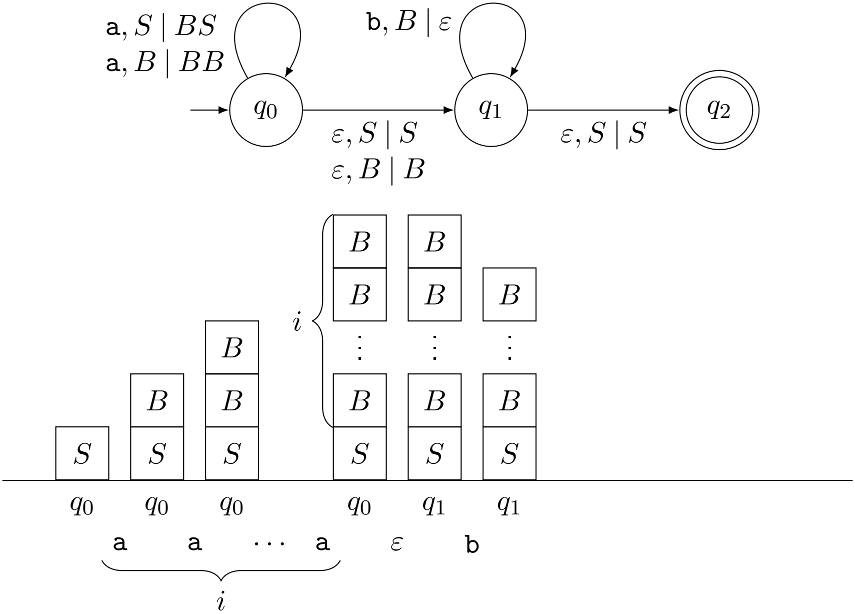




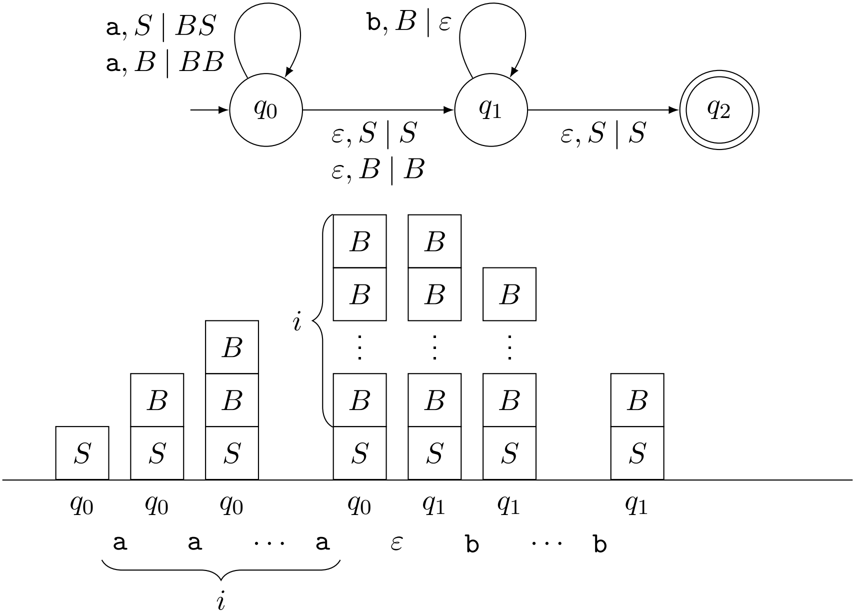
Beispiel (Akzeptanz mit akzeptierenden Zuständen)



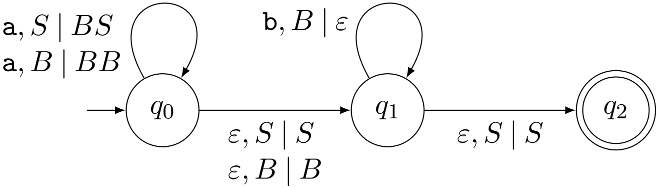
Beispiel (Akzeptanz mit akzeptierenden Zuständen)

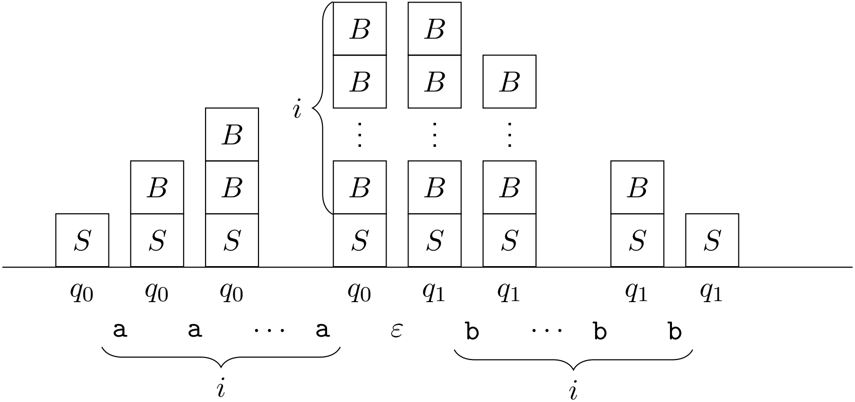


Beispiel (Akzeptanz mit akzeptierenden Zuständen)

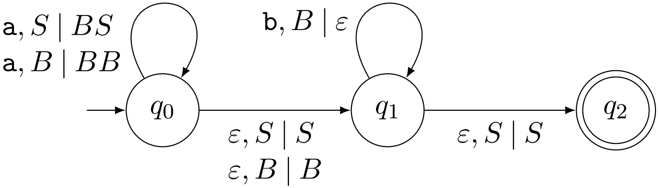


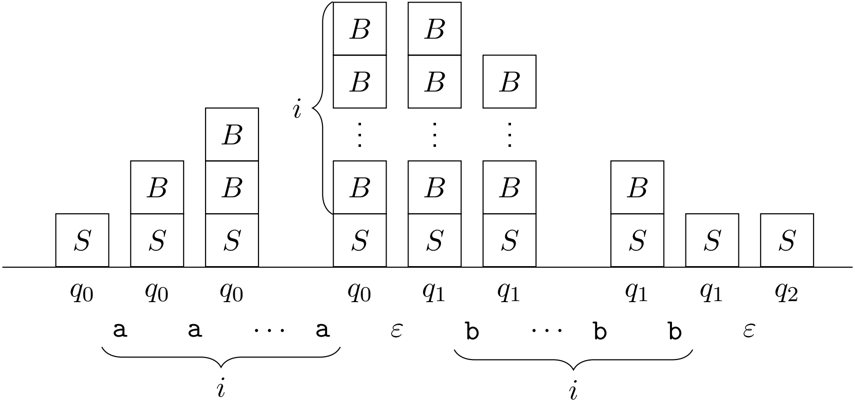
Beispiel (Akzeptanz mit akzeptierenden Zuständen)



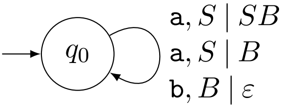


Beispiel (Akzeptanz mit akzeptierenden Zuständen)

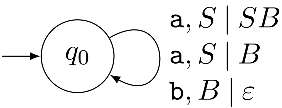


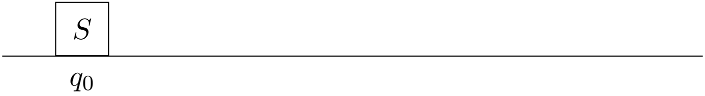


Beispiel (Akzeptanz mit leerem Keller)

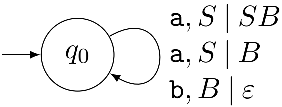


Beispiel (Akzeptanz mit leerem Keller)

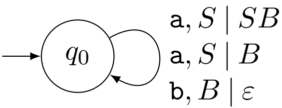


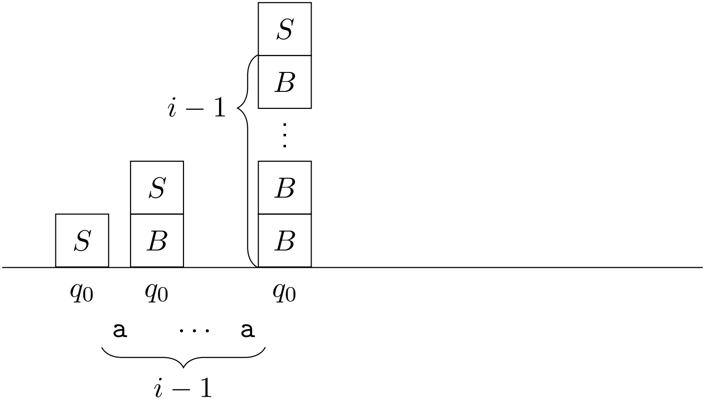


Beispiel (Akzeptanz mit leerem Keller)

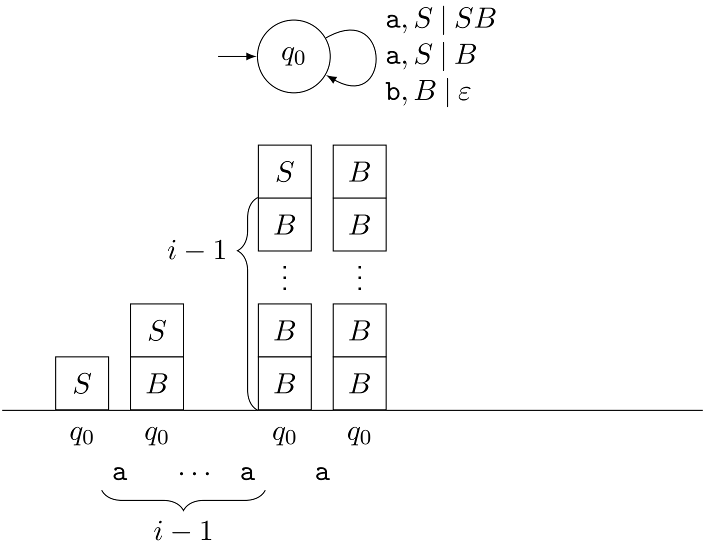


Beispiel (Akzeptanz mit leerem Keller)

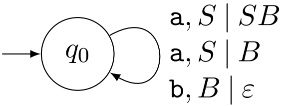


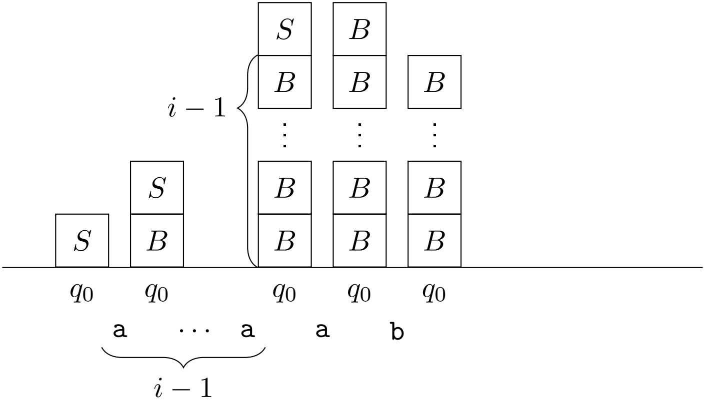


Beispiel (Akzeptanz mit leerem Keller)

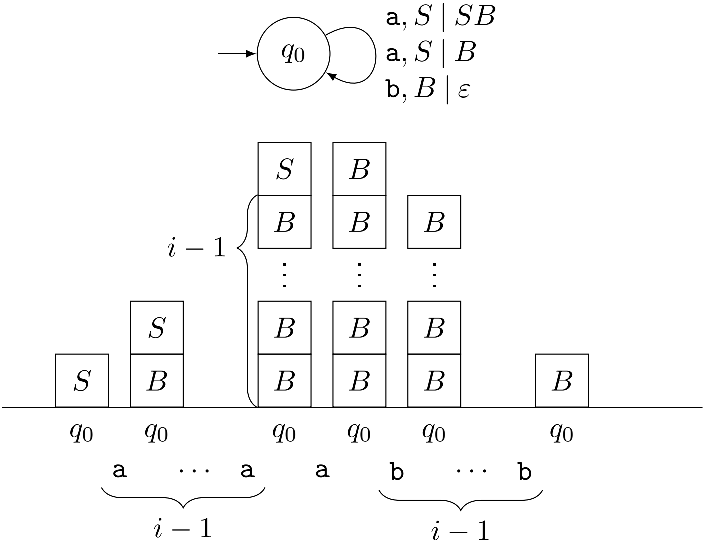


Beispiel (Akzeptanz mit leerem Keller)

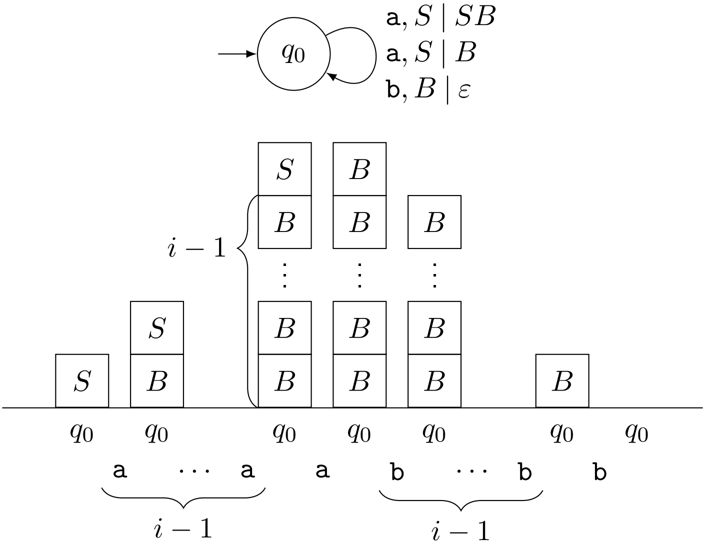




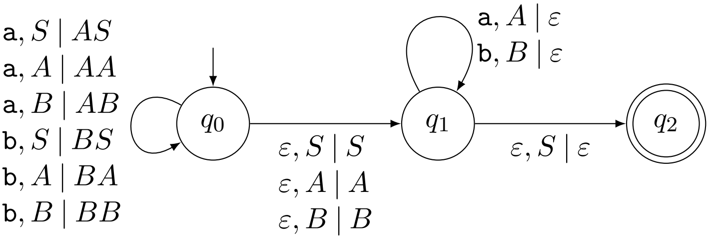
Beispiel (Akzeptanz mit leerem Keller)



Beispiel (Akzeptanz mit leerem Keller)



Noch ein Beispiel



Vergleich der Akzeptanzverhalten

Satz

Sei A ein PDA.
Dann existieren PDAs A_K und A_Z mit:

- $\mathcal{L}_Z(A) = \mathcal{L}_K(A_K)$, und
- $\mathcal{L}_K(A) = \mathcal{L}_Z(A_Z)$.

Vergleich der Akzeptanzverhalten

Satz

Sei A ein PDA.
Dann existieren PDAs A_K und A_Z mit:

- $\mathcal{L}_Z(A) = \mathcal{L}_K(A_K)$, und
- $\mathcal{L}_K(A) = \mathcal{L}_Z(A_Z)$.

Von Z zu K

verbinde akzeptierende Zustände mit „Leer-Räum-Zustand“

Vergleich der Akzeptanzverhalten

Satz

Sei A ein PDA.
Dann existieren PDAs A_K und A_Z mit:

- $\mathcal{L}_Z(A) = \mathcal{L}_K(A_K)$, und
- $\mathcal{L}_K(A) = \mathcal{L}_Z(A_Z)$.

Von Z zu K

verbinde akzeptierende Zustände mit „Leer-Räum-Zustand“

Von K zu Z

- lege neues Symbol K_L unter K_0
- wenn K_L zu sehen: wechsle in akzeptierenden Zustand

Satz

L ist kontextfrei genau dann,
wenn ein PDA A existiert mit $\mathcal{L}(A) = L$.

Kellerautomaten und kontextfreie Sprachen

Satz

L ist kontextfrei genau dann,
wenn ein PDA A existiert mit $\mathcal{L}(A) = L$.

Beweisidee $\Rightarrow$

konstruiere PDA direkt
aus CFG in CNF

Kellerautomaten und kontextfreie Sprachen

Satz

L ist kontextfrei genau dann,
wenn ein PDA A existiert mit $\mathcal{L}(A) = L$.

Beweisidee $\Rightarrow$

konstruiere PDA direkt aus CFG in CNF

Beweisidee $\Leftarrow$

simuliere PDA in CFG: Tripelkonstruktion

Satz

L ist kontextfrei genau dann,
wenn ein PDA A existiert mit $\mathcal{L}(A) = L$.

Beweisidee $\Rightarrow$

konstruiere PDA direkt
aus CFG in CNF

Beweisidee $\Leftarrow$

simuliere PDA in CFG:
Tripelkonstruktion

Gemeinsame Idee

Schritte des PDA entsprechen
Linksableitungsschritten der Grammatik.

PDA aus CFG

- $G = (\Sigma, V, P, S)$ in CNF
- $A := (\Sigma, \Gamma, Q, \delta, q, S, \emptyset)$
  (Akzeptanz durch Keller)
- $\Gamma := V$,
- $Q := \{q\}$.

PDA aus CFG

- $G = (\Sigma, V, P, S)$ in CNF
- $A := (\Sigma, \Gamma, Q, \delta, q, S, \emptyset)$ (Akzeptanz durch Keller)
- $\Gamma := V$,
- $Q := \{q\}$.

Definition von δ

Für alle $a \in \Sigma$, $A \in \Gamma$:

$$\delta(q, a, A) := \{(q, \varepsilon) \mid A \to a \in P\}$$

PDA aus CFG

- $G = (\Sigma, V, P, S)$ in CNF
- $A := (\Sigma, \Gamma, Q, \delta, q, S, \emptyset)$ (Akzeptanz durch Keller)
- $\Gamma := V$,
- $Q := \{q\}$.

Definition von δ

Für alle $a \in \Sigma$, $A \in \Gamma$:

$$\delta(q, a, A) := \{(q, \varepsilon) \mid A \to a \in P\}$$

Für alle $A \in \Gamma$ ist

$$\delta(q, \varepsilon, A) := \{BC \mid A \to BC \in P\}$$

Ggf. noch $(q, \varepsilon) \in \delta(q, \varepsilon, S)$

Hauptteil des Beweises

PDA aus CFG

- $G = (\Sigma, V, P, S)$ in CNF
- $A := (\Sigma, \Gamma, Q, \delta, q, S, \emptyset)$ (Akzeptanz durch Keller)
- $\Gamma := V$,
- $Q := \{q\}$.

Definition von δ

Für alle $a \in \Sigma$, $A \in \Gamma$:

$$\delta(q, a, A) := \{(q, \varepsilon) \mid A \to a \in P\}$$

Für alle $A \in \Gamma$ ist

$$\delta(q, \varepsilon, A) := \{BC \mid A \to BC \in P\}$$

Ggf. noch $(q, \varepsilon) \in \delta(q, \varepsilon, S)$

Hauptteil des Beweises

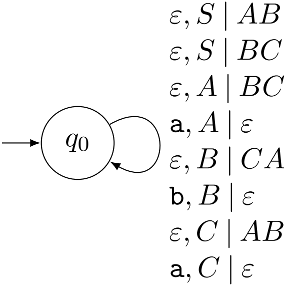
Für alle $w \in \Sigma^+$, alle $A \in \Gamma$ und alle $n \in \mathbb{N}_{>0}$ gilt

$$(q, w, A) \vdash_A^n (q, \varepsilon, \varepsilon) \text{ genau dann, wenn } A \Rightarrow_G^n w.$$

Beispiel für PDA aus CFG ($\Rightarrow$)

$$
\begin{aligned}
G &:= (\Sigma, V, P, S),\\
V &:= \{S, A, B, C\},\\
P := \{S &\to AB \mid BC,\\
A &\to BC \mid \mathtt{a},\\
B &\to CA \mid \mathtt{b},\\
C &\to AB \mid \mathtt{a}\}.
\end{aligned}
$$

$$
\begin{aligned}
G &:= (\Sigma, V, P, S),\\
V &:= \{S, A, B, C\},\\
P &:= \{S \to AB \mid BC,\\
&\quad A \to BC \mid \mathtt{a},\\
&\quad B \to CA \mid \mathtt{b},\\
&\quad C \to AB \mid \mathtt{a}\}.
\end{aligned}
$$



Tripelkonstruktion ($\Leftarrow$)

- $A = (\Sigma, \Gamma, Q, \delta, q_0, K_0, \emptyset)$
- $G := (\Sigma, V, P, S)$
- $V := \{S\} \cup \{[p, K, q] \mid p, q \in Q, K \in \Gamma\}$

Tripelkonstruktion ($\Leftarrow$)

- $A = (\Sigma, \Gamma, Q, \delta, q_0, K_0, \emptyset)$
- $G := (\Sigma, V, P, S)$
- $V := \{S\} \cup \{[p, K, q] \mid p, q \in Q, K \in \Gamma\}$

Idee

$$[p, K, q] \Rightarrow_G^* w$$
genau dann, wenn
$$(p, w, K) \vdash_A^* (q, \varepsilon, \varepsilon)$$

Tripelkonstruktion ($\Leftarrow$)

- $A = (\Sigma, \Gamma, Q, \delta, q_0, K_0, \emptyset)$
- $G := (\Sigma, V, P, S)$
- $V := \{S\} \cup \{[p, K, q] \mid p, q \in Q, K \in \Gamma\}$

Idee

$$[p, K, q] \Rightarrow_G^* w$$

genau dann, wenn

$$(p, w, K) \vdash_A^* (q, \varepsilon, \varepsilon)$$

Regeln

1. $S \to [q_0, K_0, p]$ für alle $p \in Q$.

Tripelkonstruktion ($\Leftarrow$)

- $A = (\Sigma, \Gamma, Q, \delta, q_0, K_0, \emptyset)$
- $G := (\Sigma, V, P, S)$
- $V := \{S\} \cup \{[p, K, q] \mid p, q \in Q, K \in \Gamma\}$

Idee

$$[p, K, q] \Rightarrow_G^* w$$

genau dann, wenn

$$(p, w, K) \vdash_A^* (q, \varepsilon, \varepsilon)$$

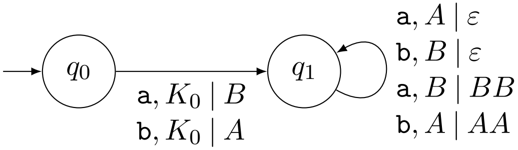
Regeln

1. $S \to [q_0, K_0, p]$ für alle $p \in Q$.
2. Für alle $p, q \in Q$, $a \in (\Sigma \cup \{\varepsilon\})$, $K \in \Gamma$ und jedes $(p_1, K_1 \cdots K_n) \in \delta(p, a, K)$ erzeugen wir die Regeln

$$[p, K, q] \to a[p_1, K_1, p_2][p_2, K_2, p_3] \cdots [p_n K_n p_{n+1}]$$

mit $p_2, \ldots, p_{n+1} \in Q$ und $p_{n+1} = q$.

Beispiel Tripelkonstruktion



Eine Anwendung

Lemma

Die Klasse CFL ist abgeschlossen unter inversem Homomorphismus.

Eine Anwendung

Lemma

Die Klasse CFL ist abgeschlossen unter inversem Homomorphismus.

Idee

- gegeben: PDA A, Hom. h
- konstruiere PDA A_I
- A_I simuliert A bei Eingabe von $h(a)$

Eine Anwendung

Lemma

Die Klasse CFL ist abgeschlossen unter inversem Homomorphismus.

Idee

- gegeben: PDA A, Hom. h
- konstruiere PDA A_I
- A_I simuliert A bei Eingabe von $h(a)$

Problem

- A_I kann pro Schritt nur ein Kellersymbol entfernen

Eine Anwendung

Lemma

Die Klasse CFL ist abgeschlossen unter inversem Homomorphismus.

Idee

- gegeben: PDA A, Hom. h
- konstruiere PDA A_I
- A_I simuliert A bei Eingabe von $h(a)$

Problem

- A_I kann pro Schritt nur ein Kellersymbol entfernen

Lösung

- A_I benutzt Zustände als **Zwischenspeicher**
- Produktkonstruktion
- Zustände $[q, x]$, $q \in Q$, $x \in \mathrm{suffix}(h(a))$