Theoretische Informatik für Wirtschaftsinformatik und Lehramt

Kontextfreie Sprachen

Priv.-Doz. Dr. Stefan Milius
stefan.milius@fau.de

Theoretische Informatik
Friedrich-Alexander Universität Erlangen-Nürnberg

SS 2016

Gliederung

Worum geht es in diesem Abschnitt? (I)

- **Vorher:** reguläre Sprachen und ihre Eigenschaften

  $\{ a^n \mid n \in \mathbb{N} \}$ regulär

  $\{ a^n b^n \mid n \in \mathbb{N} \}$ nicht regulär

- **Jetzt:** kontextfreie Sprachen

  (Anwendung: Beschreibung der Syntax von Programmiersprachen)

  - Erzeugung durch: kontextfreie Grammatiken

    Beispiel: $\{a^n b^n \mid n \in \mathbb{N}\}$ kontextfrei
    (ähnlich korrekter Klammerung von Ausdrücken)

  - Akzeptierung durch Maschinenmodell:
    Kellerautomaten = endliche Automaten + Stackspeicher

Lernziele

- kontextfreie Sprachen durch kontextfreie Grammatiken beschreiben können
- bestimmen können, welche Sprache eine kontextfreie Grammatik erzeugt
- durch Angabe einer kontextfreien Grammatik oder eines Kellerautomaten, nachweisen können, dass eine Sprache kontextfrei ist

Kontextfreie Grammatiken

Kontextfreie Grammatiken (I)

Anwendung: **Syntaxdefinitionen** von Programmiersprachen

Beispiel

Festlegung der syntaktisch zulässigen Gestalt von „arithmetischen Ausdrücken“ der folgenden Art durch eine Grammatik:

$x + y$

$y * (x + y) - z$

usw.

Kontextfreie Grammatiken (II)

Beispiel (Fortsetzung)

Grammatik (in Backus-Naur-Form (BNF))

$$\langle Ausdruck \rangle ::= \langle Term \rangle \mid \langle Ausdruck \rangle + \langle Term \rangle \mid \langle Ausdruck \rangle - \langle Term \rangle$$

$$\langle Term \rangle ::= \langle Faktor \rangle \mid \langle Term \rangle * \langle Faktor \rangle \mid \langle Term \rangle / \langle Faktor \rangle$$

$$\langle Faktor \rangle ::= \langle Identifikator \rangle \mid (\langle Ausdruck \rangle)$$

$$\langle Identifikator \rangle ::= x \mid y \mid z$$

Bemerkung:

- realistische Syntaxdefinitionen lassen z.B. kompliziertere „Identifikatoren“(= Variablenbezeichner) zu
- Tool für kontextfreie Grammatiken (Parsergenerator): yacc („yet another compiler compiler“)

Kontextfreie Grammatiken (III)

- Zeilen sind Beispiele für Regeln einer Grammatik:
  - Variablen von der Form $\langle \cdots \rangle$ und
  - „::=“ ersetzt „$\rightarrow$“

Beispiel (Fortsetzung)

Grammatik $G = (V, \Sigma, P, A)$ mit

$$V = \{A, T, F, I\}, \qquad \Sigma = \{(,), +, -, *, /, x, y, z\},$$

und den Regeln

$$\begin{array}{rrcl} P: & A & \rightarrow & T \mid A + T \mid A - T \\ & T & \rightarrow & F \mid T * F \mid T/F \\ & F & \rightarrow & I \mid (A) \\ & I & \rightarrow & x \mid y \mid z \end{array}$$

Bemerkung

Die Grammatik G ist nicht regulär **und** $L(G)$ ist nicht regulär.

Kontextfreie Grammatiken (IV)

Beispiel (Fortsetzung)

Warum ist $L(G)$ nicht regulär?

- in syntaktisch korrekten arithmetischen Ausdrücken gilt:

  Anzahl „(“ $=$ Anzahl „)“

- ähnliche Eigenschaft wie die Wörter von $\{ a^n b^n \mid n \in \mathbb{N} \}$
- Beweis der Nichtregularität:
  - mit Pumping-Lemma für reguläre Sprachen
  - mit Nerode-Äquivalenz
- **Fazit:** reguläre Grammatiken reichen nicht aus, um Syntax einer Programmiersprache zu beschreiben

Kontextfreie Grammatiken (V)

Wiederholung Typ-2 Grammatiken:

Definition (kontextfreie Grammatik)

Eine Grammatik $G = (V, \Sigma, P, S)$ heißt **kontextfrei**, falls alle Regeln in P von der folgenden Form sind:

$$A \to w \qquad \text{mit} \qquad A \in V, \quad w \in (\Sigma \cup V)^*.$$

Eine formale Sprache $L \subset \Sigma^*$ heißt **kontextfrei** falls es eine kontextfreie Grammatik G gibt, die L erzeugt, d.h. mit

$$L = L(G) = \{w \in \Sigma^* \mid S \Rightarrow^* w\}.$$

Kontextfreie Grammatiken (VI)

Beispiel 1

$G_{ab} = (\{S\}, \{a, b\}, P, S)$ mit

$$\begin{array}{rlll} P: & S & \to & \varepsilon \\ & S & \to & aSb \end{array} \qquad \text{(oder kürzer: } S \to aSb \mid \varepsilon\text{)}$$

Die von G erzeugte Sprache ist kontextfrei:

$$L(G) = \{ a^n b^n \mid n \in \mathbb{N} \}$$

Mögliche Ableitung:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$$

Kontextfreie Grammatiken (VII)

Beispiel 2

Kontextfreie Grammatik $G = (V, \Sigma, P, A)$ mit

$$V = \{A, T, F, I\}, \qquad \Sigma = \{(,), +, -, *, /, x, y, z\},$$

und den Regeln

$$\begin{array}{rrcl} P: & A & \to & T \mid A + T \mid A - T \\ & T & \to & F \mid T * F \mid T/F \\ & F & \to & I \mid (A) \\ & I & \to & x \mid y \mid z \end{array}$$

Die von G erzeugte Sprache ist kontextfrei:

$$L(G) = \text{ alle arithmetischen Ausdrücke über } \{x, y, z\}$$

Kontextfreie Grammatiken (VIII)

Beispiel 2 (Fortsetzung)

Mögliche Ableitung:

$$
\begin{aligned}
A \quad & \Rightarrow A - T \Rightarrow T - T \Rightarrow T * F - T \Rightarrow F * F - T \Rightarrow \\
& \Rightarrow I * F - T \Rightarrow y * F - T \Rightarrow y * (A) - T \Rightarrow \\
& \Rightarrow y * (A + T) - T \Rightarrow y * (T + T) - T \Rightarrow \\
& \Rightarrow y * (F + T) - T \Rightarrow y * (I + T) - T \Rightarrow \\
& \Rightarrow y * (x + T) - T \Rightarrow y * (x + F) - T \Rightarrow \\
& \Rightarrow y * (x + I) - T \Rightarrow y * (x + y) - T \Rightarrow \\
& \Rightarrow y * (x + y) - F \Rightarrow y * (x + y) - I \Rightarrow \\
& \Rightarrow y * (x + y) - z
\end{aligned}
$$

Kontextfreie Grammatiken (IX)

Erinnerung: Regelformen bei regulären Grammatiken:

$$A \to \varepsilon \qquad A \to a \qquad A \to aB \qquad (A, B \in V, a \in \Sigma).$$

Daher gilt trivialerweise:

Satz 4.1

Jede reguläre Grammatik ist auch eine kontextfreie Grammatik, und daher ist jede reguläre Sprache auch kontextfrei.

Wegen der nicht regulären aber kontextfreien Sprache

$$\{ a^n b^n \mid n \in \mathbb{N} \}$$

Satz 4.2

Es gibt kontextfreie Sprachen, die nicht regulär sind.

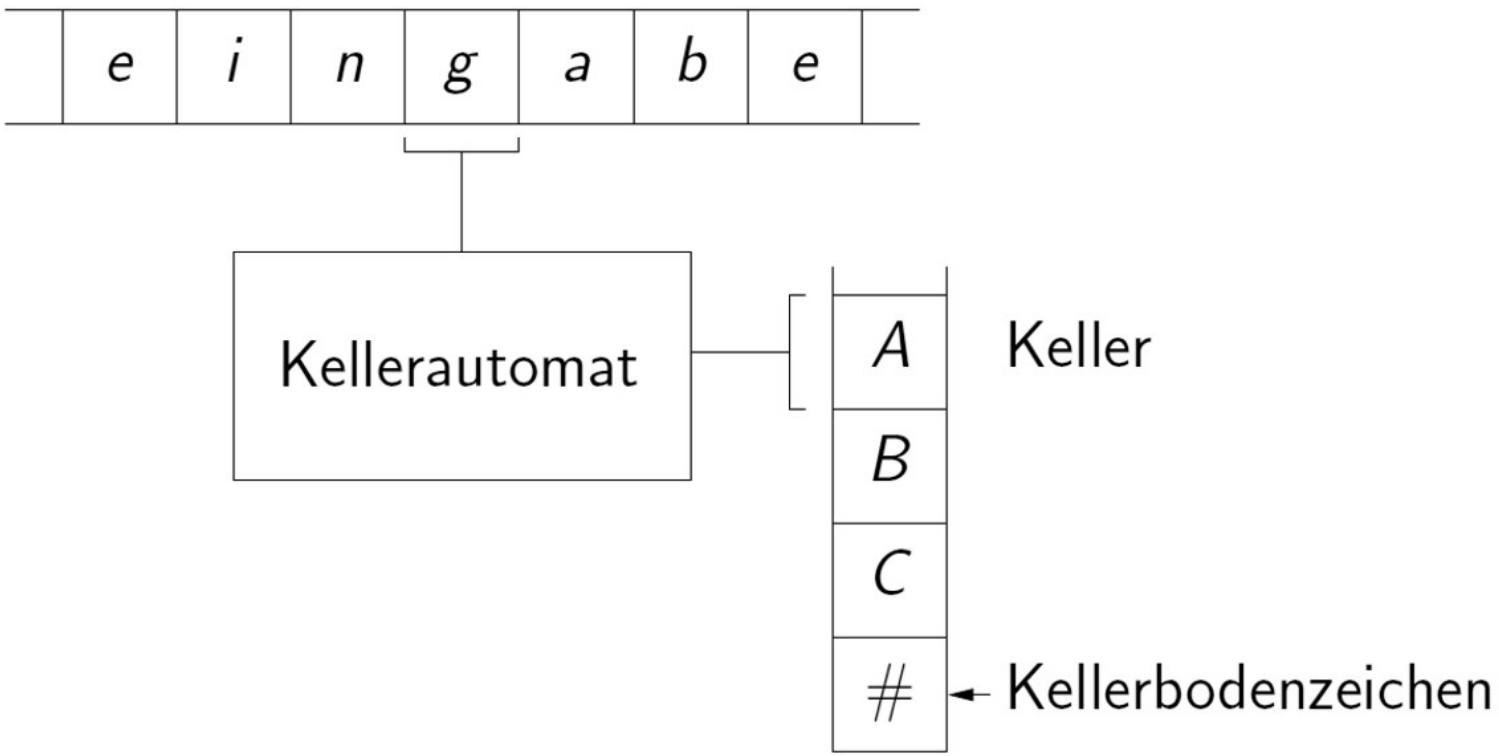
Kellerautomaten

Kellerautomaten (I)

- **Jetzt:** Maschinenmodell für kontextfreie Sprachen
- Erweiterung von endlichen Automaten
  aber Einschränkung von Turing-Maschinen (später)
- **Benötigt wird:**
  Möglichkeit unbeschränkt Informationen zu speichern
  Z.B. für $\{ a^n b^n \mid n \in \mathbb{N} \}$
- **Grundidee:** Endliche Automaten werden um einen unbeschränkten Speicher erweitert
  (genauer: ein Stack- bzw. Kellerspeicher genügt)

Kellerautomaten (III)

Schematische Darstellung eines Kellerautomaten:



Kellerautomaten (IV)

Betrachte

$$\Sigma = \{a, b, c, d\} \quad \text{und} \quad L = \{ x_1 x_2 \cdots x_n \$ x_n \cdots x_2 x_1 \mid x_i \in \Sigma \}.$$

Ein Kellerautomat akzeptiert diese Sprache folgendermaßen:

- Ein Wort w wird von links nach rechts gelesen.
- Der Automat hat zwei Zustände:

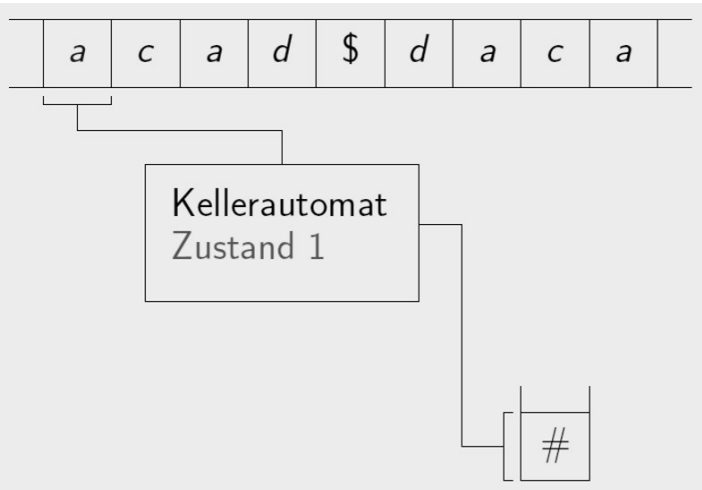
  Zustand 1: ersten Teil des Wortes auf dem Stack speichern

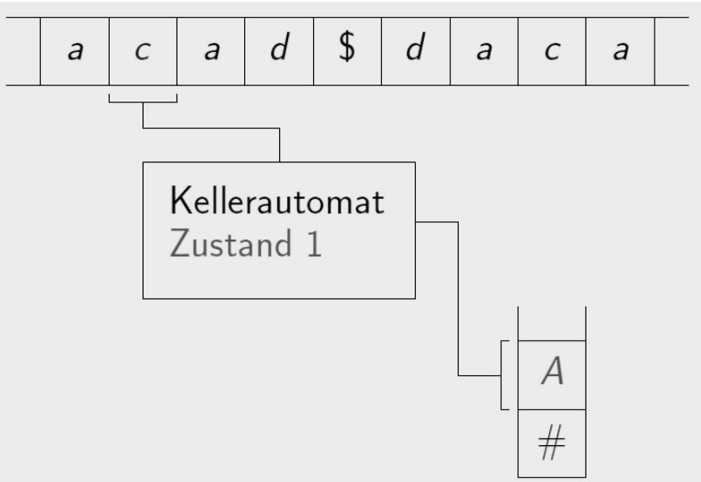
  Zustand 2: zweiten Teil des Wortes auf „Gleichheit“ überprüfen

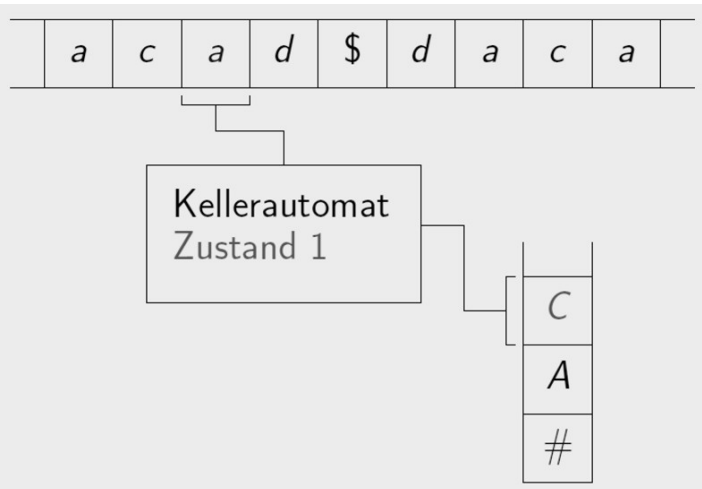
Kellerautomaten (V)

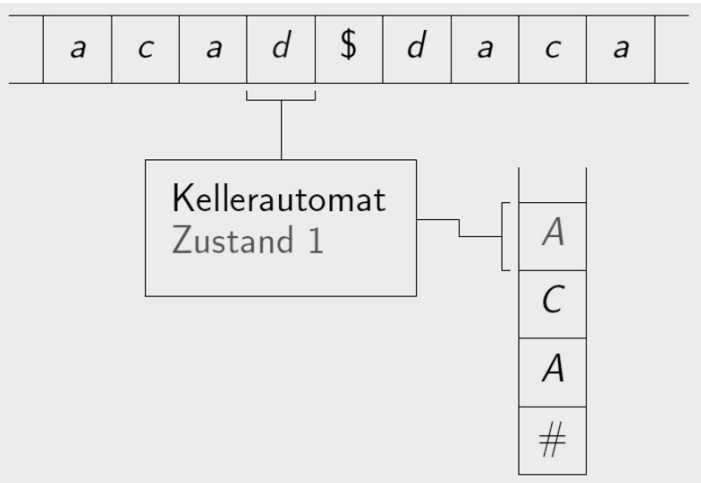
- **Zustand 1:** Solange \$ noch nicht erreicht:
  jedes gelesene Symbol wird als Großbuchstabe auf den Keller gelegt $(a \rightsquigarrow A, b \rightsquigarrow B, \ldots)$
- Wenn \$ gelesen wird:
  Keller bleibt unverändert und nach Zustand 2 wechseln
- **Zustand 2:** für jedes gelesene Zeichen x:
  - falls passender Großbuchstabe X oben im Keller steht:
    X entfernen
  - anderenfalls:
    halten und nicht akzeptieren
- Am Schluss (falls „Gleichheitsprüfung“ immer erfolgreich):
  Kellerbodenzeichen entfernen, halten und akzeptieren

Kellerautomaten (VI)

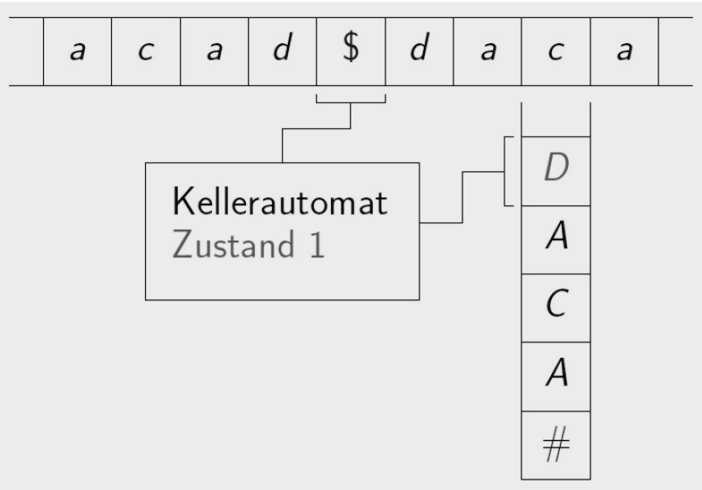


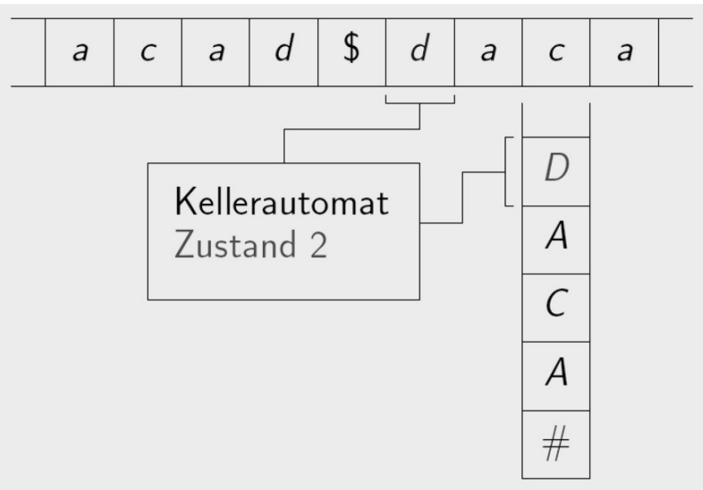


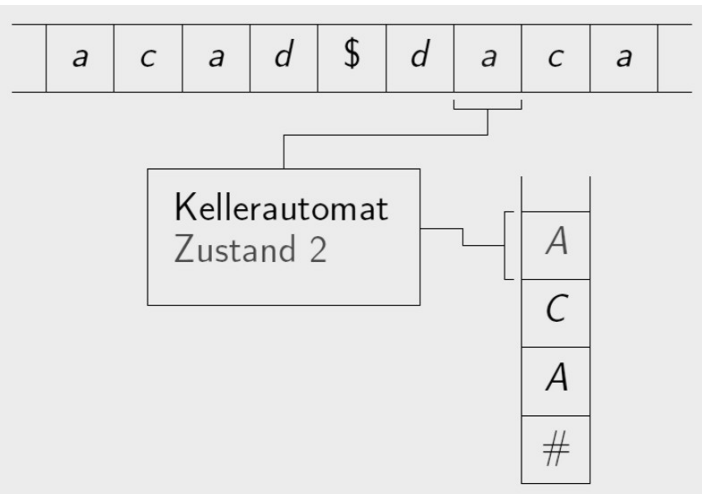


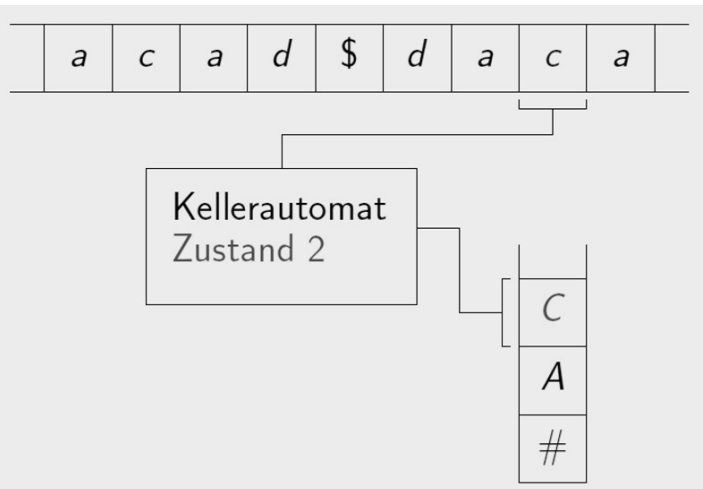


Kellerautomaten (VII)

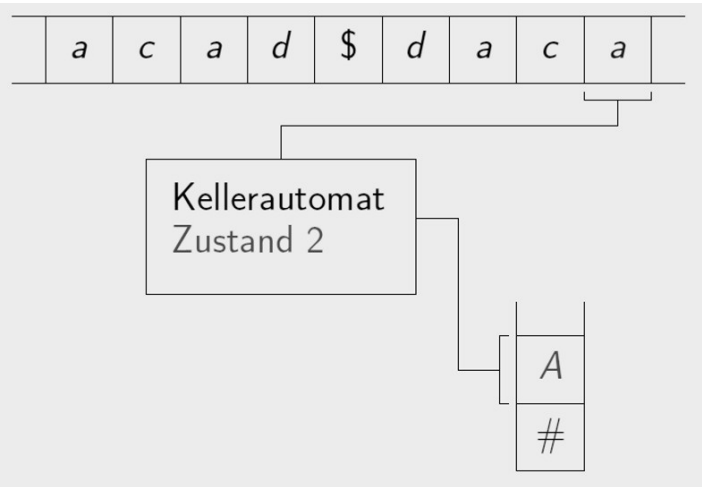


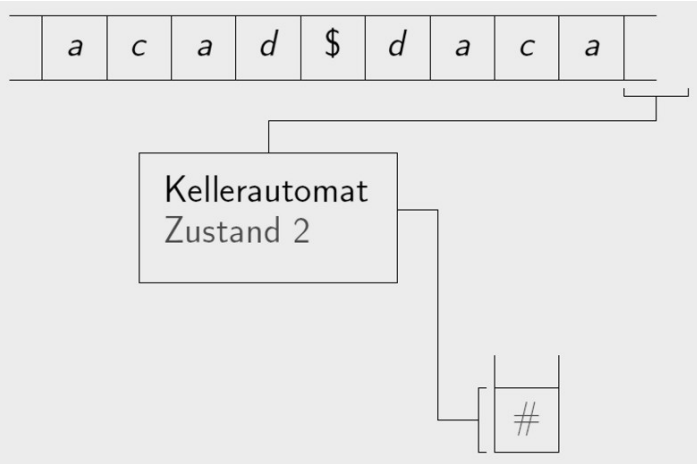


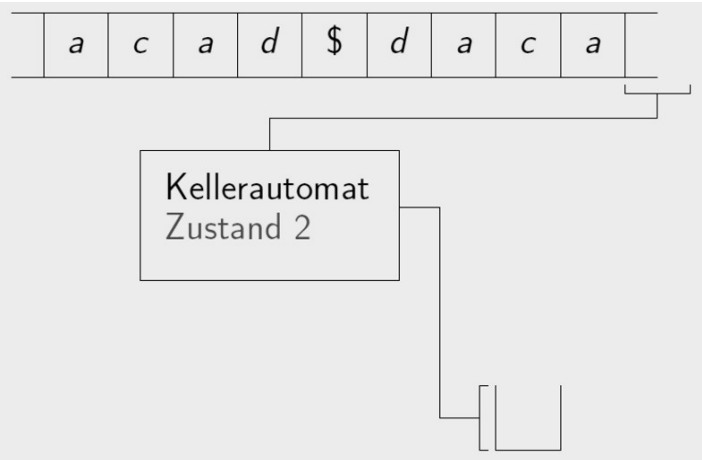




Kellerautomaten (VIII)







Kellerautomaten (IX)

Definition (Kellerautomat)

Ein **(nichtdeterministischer) Kellerautomat (pushdown automaton, PDA)** $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ ist gegeben durch:

- eine endliche Menge Z von **Zuständen**,
- ein **Eingabealphabet** Σ,
- ein **Kelleralphabet** Γ,
- eine totale **Überführungsfunktion**

$$\delta : Z \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to \wp(Z \times \Gamma^*)$$

  mit: $\delta(z, a, A)$ *endlich* für jedes $(z, a, A) \in Z \times (\Sigma \cup \{\varepsilon\}) \times \Gamma$.
- einen **Startzustand** $z_0 \in Z$,
- ein **Kellerstartsymbol** $\# \in \Gamma$,

Kellerautomaten (X)

Bemerkungen:

- Kellerautomaten haben **spontane Übergänge**:
  $(z', w) \in \delta(z, \varepsilon, A)$ bedeutet:

  M wechselt bei oberstem Kellersymbol A von z nach z'
  (ohne ein Eingabezeichen zu lesen)
- Kellerautomaten akzeptieren nach Verarbeitung der Eingabe durch einen leeren Keller (d.h. mit Kellerinhalt ε)

Das wird jetzt formalisiert: ...

Kellerautomaten (XI)

Definition (Kellerautomat (Fortsetzung))

Eine **Konfiguration** von M ist ein Tripel

$$(z, w, \alpha) \in Z \times \Sigma^* \times \Gamma^*.$$

Bemerkung: Eine Konfiguration (z, w, α) beschreibt eine „Momentaufnahme“ einer Berechnung:

- Kellerautomat befindet sich im Zustand z
- w ist die noch zu verarbeitende Rest-Eingabe
- α ist der aktuelle Inhalt des Kellers
  (erstes Zeichen von α = oberstes Kellerzeichen)

Konfigurationsänderungen bewirkt durch δ $\rightsquigarrow$ „Rechenschritte“...

Kellerautomaten (XII)

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ ein PDA.
Die binäre Relation $\vdash_M$ auf der Menge der Konfigurationen von M ist definiert durch

$$(z, ax, A\beta) \vdash_M (z', x, \gamma\beta) \quad \text{gdw.} \quad (z', \gamma) \in \delta(z, a, A)$$

$$(\text{für } a \in \Sigma \cup \{\varepsilon\}, \quad x \in \Sigma^*, \quad A \in \Gamma \quad \text{und} \quad \beta, \gamma \in \Gamma^*).$$

Es ist $\vdash_M^*$ sei die reflexive, transitive Hülle von $\vdash_M$.

Bemerkungen:

- Schreibe: $\vdash$ statt $\vdash_M$
- $\vdash$ beschreibt einen „Rechenschritt“
- $\vdash^*$ beschreibt $0, 1, 2, 3, \ldots$ „Rechenschritte“

Arbeitsweise von Kellerautomaten (I)

- M arbeitet ein Eingabewort zeichenweise von links nach rechts ab.
- wenn (z, au, Av) momentate Konfiguration

  (M im Zustand z, liest Eingabezeichen a und hat A oben im Keller):
  - M wählt nichtdeterministisch ein

    $(z', w) \in \delta(z, a, A)$ oder $(z', w) \in \delta(z, \varepsilon, A)$
  - löscht A vom Keller
  - wechselt nach Zustand z'
  - schreibt w oben auf den Keller

    (neue Konfiguration (z', u, wv) oder (z', au, wv))
- Dann betrachtet M das nächste Zeichen der Eingabe.

  (bei spontanem Übergang:

  aktuelles Eingabezeichen = nächstes Eingabezeichen)
- M hält an falls:
  - $\delta(z, a, A) = \emptyset = \delta(z, \varepsilon, A)$ (kein Übergang möglich)
  - Konfiguration (z, w, ε) erreicht (immer bei leerem Keller)

Akzeptierte Sprache eines Kellerautomaten (I)

Definition

Sei $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$ ein PDA.

M **akzeptiert** ein Wort $w \in \Sigma^*$, falls

$$(z_0, w, \#) \vdash^* (z, \varepsilon, \varepsilon)$$

(d.h. es gibt eine akzeptierende Rechnung).

Die **von M akzeptierte Sprache** ist

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

Akzeptierte Sprache eines Kellerautomaten (II)

Beispiel

$M = (\{z_0, z_1, z_2, z_3\}, \{a, b\}, \{A, B, \#\}, \delta, z_0, \#)$ mit

$$
\begin{array}{lcl}
\delta(z_0, \varepsilon, \#) & = & \{(z_0, \varepsilon)\} \\
\delta(z_0, a, \#) & = & \{(z_1, A\#)\} \\
\delta(z_0, b, \#) & = & \{(z_1, B\#)\} \\
\delta(z_1, a, B) & = & \{(z_1, AB)\} \\
\delta(z_1, b, A) & = & \{(z_1, BA)\} \\
\delta(z_1, a, A) & = & \{(z_1, AA), (z_2, \varepsilon)\} \\
\delta(z_1, b, B) & = & \{(z_1, BB), (z_2, \varepsilon)\}
\end{array}
\qquad
\begin{array}{lcl}
\delta(z_2, a, A) & = & \{(z_2, \varepsilon)\} \\
\delta(z_2, b, B) & = & \{(z_2, \varepsilon)\} \\
\delta(z_2, \varepsilon, \#) & = & \{(z_3, \varepsilon)\}
\end{array}
$$

$(\delta(\dots) = \emptyset$ sonst$)$

M akzeptiert die Sprache $\{ ww^R \mid w \in \{a, b\}^* \}$

Akzeptierte Sprache eines Kellerautomaten (III)

Beispiel (Fortsetzung)

Eine akzeptierende Rechnung für *abbbba*:

$$\begin{aligned}(z_0, abbbba, \#) &\vdash (z_1, bbbba, A\#)\\ &\vdash (z_1, bbba, BA\#)\\ &\vdash (z_1, bba, BBA\#)\\ &\vdash (z_2, ba, BA\#)\\ &\vdash (z_2, a, A\#)\\ &\vdash (z_2, \varepsilon, \#)\\ &\vdash (z_3, \varepsilon, \varepsilon)\end{aligned}$$

Eine nicht akzeptierende Rechnung für *abbbba*:

$$\begin{aligned}(z_0, abbbba, \#) &\vdash (z_1, bbbba, A\#)\\ &\vdash (z_1, bbba, BA\#) \vdash (z_2, bba, A\#)\end{aligned}$$

Akzeptierte Sprache eines Kellerautomaten (IV)

Beispiel (Fortsetzung)

Ablauf der akzeptierenden Rechung:

1. $w = abb$ (kodiert als ABB) im Keller speichern
2. Übergang zu w^R nichtdeterministisch raten (Übergang in z_2)
3. Überprüfung durch zeichenweises „Abstreichen“ von B, B und A im Keller
4. am Schluss: Keller leeren und nach z_3 wechseln

Kellerautomaten und kontextfreie Sprachen

Kellerautomaten und kontextfreie Sprachen (I)

- **Jetzt:**
  Kellerautomaten gleichmächtig wie kontextfreie Grammatiken
- **Zwei Konstruktionen:**

$$\text{kontextfreie Grammatik} \rightsquigarrow \text{Kellerautomat}$$

$$\text{Kellerautomat} \rightsquigarrow \text{kontextfreie Grammatik}$$

Kellerautomaten und kontextfreie Sprachen (I)

- **Gegeben:** kontextfreie Grammatik $G = (V, \Sigma, P, S)$.
- Idee der Konstruktion
  - Ableitungen der Grammatik werden auf dem Stack „simuliert“
  - Nichtterminale werden gemäß der Regeln ersetzt
  - Terminale werden mit Eingabe verglichen und gelöscht
- **Konstruktion** des PDA M_G :

  $M_G = (\{z\}, \Sigma, V \cup \Sigma, \delta, z, S)$ mit

$$\begin{array}{rcll} \delta(z, \varepsilon, A) & = & \{\,(z, w) \mid A \to w \text{ Regel in } P\,\} & \text{Regelanwendung} \\ \delta(z, s, s) & = & \{\,(z, \varepsilon)\,\} & \text{Terminale vergleichen \& löschen} \\ \delta(\cdots) & = & \emptyset & \text{sonst} \end{array}$$

Kellerautomaten und kontextfreie Sprachen (II)

Beispiel

Grammatik $G_{ab} = (\{S\}, \{a, b\}, P, S)$ für $L(G) = \{a^n b^n \mid n \in \mathbb{N}\}$ mit

$$P : S \to \varepsilon \mid aSb$$

Konstruierter Kellerautomat:

$$M_{G_{ab}} = (\{z\}, \{a, b\}, \{S, a, b\}, \delta, z, S) \quad \text{mit}$$

$$\begin{aligned} \delta(z, \varepsilon, S) &= \{(z, \varepsilon), (z, aSb)\} \\ \delta(z, a, a) &= \{(z, \varepsilon)\} \\ \delta(z, b, b) &= \{(z, \varepsilon)\} \end{aligned}$$

und $\delta(\cdots) = \emptyset$ sonst.

Kellerautomaten und kontextfreie Sprachen (III)

Beispiel (Fortsetzung)

Eine akzeptierende Rechnung für *aaabbb*:

$$
\begin{aligned}
(z, aaabbb, S) &\vdash (z, aaabbb, aSb) \\
&\vdash (z, aabbb, Sb) \\
&\vdash (z, aabbb, aSbb) \\
&\vdash (z, abbb, Sbb) \\
&\vdash (z, abbb, aSbbb) \\
&\vdash (z, bbb, Sbbb) \\
&\vdash (z, bbb, bbb) \\
&\vdash (z, bb, bb) \\
&\vdash (z, b, b) \\
&\vdash (z, \varepsilon, \varepsilon)
\end{aligned}
$$

Kellerautomaten und kontextfreie Sprachen (IV)

Lemma 4.3

Für jede kontextfreie Grammatik G ist $L(M_G) = L(G)$.

Jetzt: umgekehrte Konstruktion:

$$\text{Kellerautomat} \quad \rightsquigarrow \quad \text{kontextfreie Grammatik}$$

Kellerautomaten und kontextfreie Sprachen (V)

- **Gegeben:** PDA $M = (Z, \Sigma, \Gamma, \delta, z_0, \#)$
- **Konstruiere:** kontextfreie Grammatik G_M mit $L(G_M) = L(M)$:

$$G_M = (\underbrace{Z \times \Gamma \times Z \cup \{S\}}_{V}, \Sigma, P, S)$$

- Startsymbol S wird neu hinzugefügt
- Variablen der Form (z, X, z') repräsentieren:
  Folge von Rechenschritten von M, die
  - im Zustand z beginnt mit X oben im Stack und
  - im Zustand z' endet, der X vom Stack entfernt.

Kellerautomaten und kontextfreie Sprachen (VI)

Regel in P:

1. $S \to (z_0, \#, z)$ für alle $z \in Z$
   (Entfernen des Kellerbodenzeichens)
2. für $(z', \varepsilon) \in \delta(z, a, A)$ die Regel

   $$(z, A, z') \to a$$

   (Symbol A kann – bei Einlesen von a – sofort entfernt werden)
3. für $(z_1, B_1 \cdots B_k) \in \delta(z, a, A)$ die Regeln

   $$(z, A, z') \to a(z_1, B_1, z_2)(z_2, B_2, z_3) \cdots (z_k, B_k, z')$$

   für alle Kombinationen von $z_2, \ldots, z_k, z' \in Z$
   (Symbol A durch $B_1 \cdots B_k$ ersetzt,
   $B_1, \ldots, B_k$ werden durch Zustände $z_1, \ldots, z_k$ vom Stack entfernt)

Kellerautomaten und kontextfreie Sprachen (VII)

Beispiel

$M_{ab} = (\{z_0, z_1\}, \{a, b\}, \{A, \#\}, \delta, z_0, \#)$ mit

$$\begin{aligned}
\delta(z_0, \varepsilon, \#) &= \{(z_1, \varepsilon)\} \\
\delta(z_0, a, \#) &= \{(z_0, A\#)\} \\
\delta(z_0, a, A) &= \{(z_0, AA)\} \\
\delta(z_0, b, A) &= \{(z_1, \varepsilon)\} \\
\delta(z_1, b, A) &= \{(z_1, \varepsilon)\} \\
\delta(z_1, \varepsilon, \#) &= \{(z_1, \varepsilon)\}
\end{aligned}$$

und $\delta(\cdots) = \emptyset$ sonst.

$$L(M_{ab}) = \{\, a^n b^n \mid n \in \mathbb{N} \,\}$$

Kellerautomaten und kontextfreie Sprachen (VIII)

Beispiel (Fortsetzung)

Beispiel einer akzeptierenden Rechnung für *aaabbb*:

$$
\begin{aligned}
(z_0, aaabbb, \#) &\vdash (z_0, aabbb, A\#) \\
&\vdash (z_0, abbb, AA\#) \\
&\vdash (z_0, bbb, AAA\#) \\
&\vdash (z_1, bb, AA\#) \\
&\vdash (z_1, b, A\#) \\
&\vdash (z_1, \varepsilon, \#) \\
&\vdash (z_2, \varepsilon, \varepsilon)
\end{aligned}
$$

Kellerautomaten und kontextfreie Sprachen (IX)

Beispiel (Fortsetzung)

Grammatik $G_{M_{ab}} = (V, \{a, b\}, P, S)$ konstruieren:

Variablenmenge

$$V = \{(z_0, A, z_0), (z_0, A, z_1), (z_1, A, z_0), (z_1, A, z_1)$$
$$(z_0, \#, z_0), (z_0, \#, z_1), (z_1, \#, z_0), (z_1, \#, z_1), S\}$$

Regeln in P:

1) $S \to (z_0, \#, z)$ für jedes $z \in Z$, also:

$S \to (z_0, \#, z_0)$
$S \to (z_0, \#, z_1)$

Kellerautomaten und kontextfreie Sprachen (X)

Beispiel (Fortsetzung)

Regeln in P (Fortsetzung):

2) für jedes $(z', \varepsilon) \in \delta(z, a, A)$ die Regel $(z, A, z') \to a$, also:

$$
\begin{array}{lcll}
(z_0, \#, z_1) & \to & \varepsilon & \text{aus } \delta(z_0, \varepsilon, \#) = \{(z_1, \varepsilon)\} \\
(z_0, A, z_1) & \to & b & \text{aus } \delta(z_0, b, A) = \{(z_1, \varepsilon)\} \\
(z_1, A, z_1) & \to & b & \text{aus } \delta(z_1, b, A) = \{(z_1, \varepsilon)\} \\
(z_1, \#, z_1) & \to & \varepsilon & \text{aus } \delta(z_1, \varepsilon, \#) = \{(z_1, \varepsilon)\}
\end{array}
$$

3) Regeln für $(z_1, B_1 \cdots B_k) \in \delta(z, a, A)$:

$$
\begin{array}{lcl}
(z_0, \#, z_0) & \to & a(z_0, A, z_0)(z_0, \#, z_0) \mid a(z_0, A, z_1)(z_1, \#, z_0) \\
(z_0, \#, z_1) & \to & a(z_0, A, z_0)(z_0, \#, z_1) \mid a(z_0, A, z_1)(z_1, \#, z_1) \\
 & & \text{beide aus } \delta(z_0, a, \#) = \{(z_0, A\#)\} \\
(z_0, A, z_0) & \to & a(z_0, A, z_0)(z_0, A, z_0) \mid a(z_0, A, z_1)(z_1, A, z_0) \\
(z_0, A, z_1) & \to & a(z_0, A, z_0)(z_0, A, z_1) \mid a(z_0, A, z_1)(z_1, A, z_1) \\
 & & \text{beide aus } \delta(z_0, a, A) = \{(z_0, AA)\}
\end{array}
$$

Kellerautomaten und kontextfreie Sprachen (XI)

Ergebnisgrammatik enthält unnötige Regeln und kann noch vereinfacht werden:

- lösche Variablen, die auf keiner linken Seite vorkommen:

  $(z_1, \#, z_0), \quad (z_1, A, z_0)$

- lösche alle Regeln, die solche Variablen (rechts) enthalten:

  $$\begin{aligned} (z_0, \#, z_0) &\rightarrow a(z_0, A, z_1)(z_1, \#, z_0) \\ (z_0, A, z_0) &\rightarrow a(z_0, A, z_1)(z_1, A, z_0) \end{aligned}$$

- lösche Variablen, die dann durch keinen Ableitungsschritt verschwinden:

  $(z_0, \#, z_0), \quad (z_0, A, z_0)$

Kellerautomaten und kontextfreie Sprachen (XII)

Beispiel (Fortsetzung)

Lösche: $(z_1, \#, z_0), (z_1, A, z_0), (z_0, \#, z_0), (z_0, A, z_0)$

vereinfachte Regelmenge:

$$
\begin{array}{rcl}
S & \to & (z_0, \#, z_1) \\
(z_0, \#, z_1) & \to & \varepsilon \\
(z_0, A, z_1) & \to & b \\
(z_1, A, z_1) & \to & b \\
(z_1, \#, z_1) & \to & \varepsilon \\
(z_0, \#, z_1) & \to & a(z_0, A, z_1)(z_1, \#, z_1) \\
(z_0, A, z_1) & \to & a(z_0, A, z_1)(z_1, A, z_1)
\end{array}
$$

Kellerautomaten und kontextfreie Sprachen (XIII)

Beispiel (Fortsetzung)

Eine Ableitung von *aaabbb* ist:

$$
\begin{aligned}
S &\Rightarrow (z_0, \#, z_1) \\
&\Rightarrow a(z_0, A, z_1)(z_1, \#, z_1) \\
&\Rightarrow aa(z_0, A, z_1)(z_1, A, z_1)(z_1, \#, z_1) \\
&\Rightarrow aaa(z_0, A, z_1)(z_1, A, z_1)(z_1, A, z_1)(z_1, \#, z_1) \\
&\Rightarrow aaab(z_1, A, z_1)(z_1, A, z_1)(z_1, \#, z_1) \\
&\Rightarrow aaabb(z_1, A, z_1)(z_1, \#, z_1) \\
&\Rightarrow aaabbb(z_1, \#, z_1) \\
&\Rightarrow aaabbb
\end{aligned}
$$

Kellerautomaten und kontextfreie Sprachen (XIV)

Lemma 4.4

Für jeden PDA M ist $L(G_M) = L(M)$.

Insgesamt:

Satz 4.5

Eine Sprache ist genau dann kontextfrei, wenn sie von einem PDA akzeptiert wird.

Später noch zu klären: sind kontextfreie Sprachen entscheidbar?

D.h. existiert ein Algorithmus für das Wortproblem jeder kontextfreien Sprache L:
Eingabe: $w \in \Sigma^*$
Aufgabe: entscheide, ob $w \in L$

Folgt noch nicht aus den bisherigen Ergebnissen!

Deterministisch kontextfreie Sprachen

Deterministisch kontextfreie Sprachen (I)

Kellerautomaten sind i. Allg. nichtdeterministisch.

Deterministische Kellerautomaten:

Definition (deterministischer Kellerautomat (DPDA))

Ein **deterministischer Kellerautomat (DPDA)** ist ein 6-Tupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \#, E)$ mit Z, Σ, Γ, z_0 und $\#$ wie beim PDA, Endzuständen E und einer *partiellen Funktion*

$$\delta : Z \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to Z \times \Gamma^*,$$

so dass für alle $z \in Z, a \in \Sigma, A \in \Gamma$ gilt:

entweder $\delta(z, a, A)$ oder $\delta(z, \varepsilon, A)$ ist definiert.

Deterministisch kontextfreie Sprachen (II)

Definition

Die **von einem DPDA M akzeptierte Sprache** ist

$$L(M) = \{ w \in \Sigma^* \mid (z_0, w, \#) \vdash^* (z, \varepsilon, v), \quad v \in \Gamma^*, z \in E \}$$

Bemerkung:

In einem DPDA ist in jeder Konfiguration nur höchstens ein (möglicherweise spontaner) Übergang möglich.

Deterministisch kontextfreie Sprachen (III)

- DPDAs sind „mächtiger“ als endliche Automaten, jedoch weniger mächtig als PDAs.
- Es gibt einen DPDA für die nicht reguläre Sprache

$$L = \{ w\$w^R \mid w \in \{a, b\}^* \}$$

- Für die folgende kontextfreie Sprache gibt es keinen DPDA:

$$L = \{ww^R \mid w \in \{a, b\}^*\}$$

Deterministisch kontextfreie Sprachen (IV)

DPDAs ergeben neue Sprachklasse:

Definition

Eine Sprache L heißt **deterministisch kontextfrei**, wenn es einen DPDA M gibt mit $L(M) = L$.

- Diese Sprachen spielen in der Praxis eine große Rolle; z.B. sind sie in linearer Rechenzeit entscheidbar.
- Es gibt auch spezielle kontextfreie Grammatiken – **LR(k)-Grammatiken** – die genau solche Sprachen erzeugen.

Zusammenfassung