

Grundbegriffe der Theoretischen Informatik

Sommersemester 2018 - Thomas Schwentick

Teil B: Kontextfreie Sprachen

9: Kellerautomaten

Version von: 17. Mai 2018 (14:12)

Ein Beispiel: Klammersausdrücke

- Sei $L_{\langle 2 \rangle}$ die Sprache der korrekt geklammerten „Tag-Ausdrücke“ mit zwei Tag-Paaren $\langle b \rangle \langle /b \rangle$ und $\langle a \rangle \langle /a \rangle$
 - Also über dem Alphabet $\Sigma = \{\langle a \rangle, \langle /a \rangle, \langle b \rangle \langle /b \rangle\}$

- $\langle a \rangle \langle a \rangle \langle b \rangle \langle /b \rangle \langle a \rangle \langle /a \rangle \langle /a \rangle \langle b \rangle \langle /b \rangle \langle /a \rangle$ ist korrekt
- $\langle a \rangle \langle a \rangle \langle b \rangle \langle /b \rangle \langle a \rangle \langle /b \rangle \langle /a \rangle \langle b \rangle \langle /b \rangle \langle /a \rangle$ ist nicht korrekt

- $L_{\langle 2 \rangle}$ ist kontextfrei und wird von der folgenden Grammatik erzeugt:

$$K \rightarrow KK \mid \langle b \rangle K \langle /b \rangle \mid \langle a \rangle K \langle /a \rangle \mid \epsilon$$

- Klar: $L_{\langle 2 \rangle}$ ist nicht regulär:
 - die Strings $\langle a \rangle^n$, $n \geq 0$, sind paarweise nicht äquivalent bezüglich $\sim_{L_{\langle 2 \rangle}}$
- Wie lässt sich algorithmisch testen, ob ein gegebener Klammerausdruck korrekt ist?
- Idee:
 - Versuche zusammengehörige Klammern zu finden
 - Geeignete Datenstruktur:
 - * Keller („Last In First Out“)

Erkennen von Klammerausdrücken mit Hilfe eines Kellers

Beispiel



$\langle b \rangle \langle a \rangle \langle b \rangle \langle a \rangle \langle a \rangle \langle /a \rangle \langle b \rangle \langle /b \rangle \langle /a \rangle \langle /b \rangle \langle a \rangle$
 $\langle /a \rangle \langle /a \rangle \langle b \rangle \langle /b \rangle \langle /b \rangle$

- $\langle a \rangle$ und $\langle b \rangle$ werden jeweils auf den Keller gelegt
- Wenn ein $\langle /a \rangle$ gelesen wird, muss ein $\langle a \rangle$ auf dem Keller sein
☞ und wird gelöscht
- Wenn ein $\langle /b \rangle$ gelesen wird, muss ein $\langle b \rangle$ auf dem Keller sein
☞ und wird gelöscht
- Am Schluss muss der Keller leer sein

Kellerautomaten: informell (1/2)

- Die Vorgehensweise dieses Algorithmus ähnelt einem endlichen Automaten:
 - Zeichenweises Lesen der Eingabe von links nach rechts
 - Am Ende Entscheidung, ob die Eingabe akzeptiert wird
- Allerdings verwendet der Algorithmus zusätzlich einen Keller
- Solche Algorithmen modellieren wir im Folgenden durch **Kellerautomaten**
- Wir werden sehen, dass Kellerautomaten genau die kontextfreien Sprachen entscheiden können
- Bevor wir die formale Definition von Kellerautomaten geben, betrachten wir noch ein Beispiel

Zweites Beispiel

Beispiel

- $L = \{a^i b^j \mid i \geq j\}$



- Idee:

- 1.Phase: Lege jedes a auf den Keller
 - 2.Phase: Lösche für jedes b ein a vom Keller
 - Falls auf diese Weise der String ganz gelesen wird, akzeptiere
- ☞ auch wenn noch etwas im Keller steht



aaabb wird akzeptiert
aaabbbb wird nicht akzeptiert

Kellerautomaten: informell (2/2)

- Die beiden betrachteten Beispiele waren nur sehr einfache Kellerautomaten
- Im Allgemeinen erlauben wir zusätzlich:
 - Zustände  endlich viele
 - Nichtdeterminismus
 - ϵ -Übergänge
 - Zusätzliche Symbolmenge für Keller
 - Zusätzliches unterstes Kellersymbol
 - Schreiben mehrerer Kellersymbole in einem Schritt
- Das erste Beispiel hat illustriert, dass es hilfreich sein kann, wenn die Akzeptierbedingung besagt, dass der Keller am Ende der Berechnung leer sein soll
- Das zweite Beispiel hat illustriert, dass es hilfreich sein kann, wenn die Akzeptierbedingung vom Zustand abhängt  Phase 2
- Wir erlauben in der Definition von Kellerautomaten beides

Inhalt

▷ 9.1 Kellerautomaten: Definitionen


9.2 Leerer Keller vs. akzeptierende Zustände

9.3 Grammatiken vs. Kellerautomaten

9.4 Kellerautomaten: Korrektheitsbeweise

9.5 Anhang: Beweisdetails

Eine kleine Komplikation


- Wenn der erste Beispiel„automat“ den String $\langle a \rangle \langle b \rangle \langle /b \rangle \langle /a \rangle \langle a \rangle \langle b \rangle \langle a \rangle \langle /a \rangle \langle /b \rangle \langle b \rangle \langle /b \rangle \langle /a \rangle$ liest, ist der Keller nach dem Lesen des vierten Zeichens leer  Vor Lesen des ersten Zeichens auch...
- Die Transitionen von Kellerautomaten sollen aber immer vom obersten Symbol des Kellers abhängen (und möglicherweise dem nächsten Eingabezeichen)
- Dass der Keller vor dem Ende der Berechnung leer wird, soll also vermieden werden
- Deshalb definieren wir für jeden Kellerautomaten ein *unterstes Kellersymbol*, das zu Beginn der Berechnung schon auf dem Keller liegt

Kellerautomaten: Definition



Definition (Kellerautomat, PDA)

- Ein **Kellerautomat (PDA)** \mathcal{A} besteht aus
 - einer Zustandsmenge Q ,
 - einem Eingabealphabet Σ ,
 - einem **Kellularphabet** Γ
(nicht notwendigerweise disjunkt zu Σ),
 - einer endlichen **Transitionsrelation** $\delta \subseteq (Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma) \times (Q \times \Gamma^*)$,
 - einem Startzustand s ,
 - einem **untersten Kellersymbol** $\tau_0 \in \Gamma$, und
 - einer Menge F akzeptierender Zustände

- Also: $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, F)$

 Englische Bezeichnung:
pushdown automaton

- Deshalb Abkürzung: PDA

- Warum ist δ so kompliziert?
- Das Verhalten des Automaten im nächsten Schritt darf abhängen von:
 - dem aktuellen Zustand p
 - dem nächsten Eingabesymbol σ
 - dem obersten Kellersymbol τ
- In einem Schritt:
 - kann sich der Zustand ändern  q
 - kann ein Eingabesymbol gelesen werden  muss aber nicht: ϵ
 - kann sich der Kellerinhalt verändern:
 - * τ kann durch (möglicherweise) mehrere Symbole ersetzt werden

 String z

- Transitionen sind deshalb von der Form
 - (p, σ, τ, q, z) mit $\sigma \in \Sigma$ oder
 - $(p, \epsilon, \tau, q, z)$

PDA für $L_{\langle 2 \rangle}$: formal

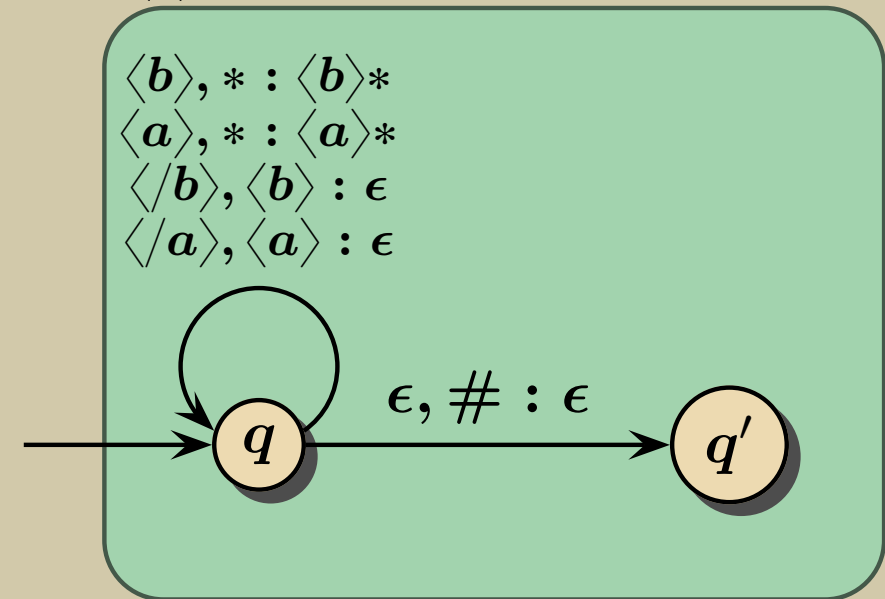
Beispiel

- Ein PDA $\mathcal{A}_{\langle 2 \rangle}$ für die Sprache $L_{\langle 2 \rangle}$, der dem vorgestellten Algorithmus entspricht, lässt sich wie folgt definieren:
 $(\{q, q'\}, \{\langle b \rangle, \langle a \rangle, \langle /a \rangle, \langle /b \rangle\},$
 $\{\langle b \rangle, \langle a \rangle, \#\}, \delta, q, \#, \emptyset),$
wobei δ die folgenden Transitionen enthält:
 - $(q, \langle a \rangle, \tau, q, \langle a \rangle \tau)$, für alle $\tau \in \Gamma$
 - $(q, \langle b \rangle, \tau, q, \langle b \rangle \tau)$, für alle $\tau \in \Gamma$
 - $(q, \langle /a \rangle, \langle a \rangle, q, \epsilon)$
 - $(q, \langle /b \rangle, \langle b \rangle, q, \epsilon)$
 - $(q, \epsilon, \#, q', \epsilon)$

- Dabei ist $\#$ das unterste Kellersymbol, das zu Beginn der Berechnung schon im Keller liegt und am Ende der Berechnung „anzeigt“, ob alle Klammern wieder vom Keller gelöscht wurden

Beispiel

- $\mathcal{A}_{\langle 2 \rangle}$ als Diagramm:



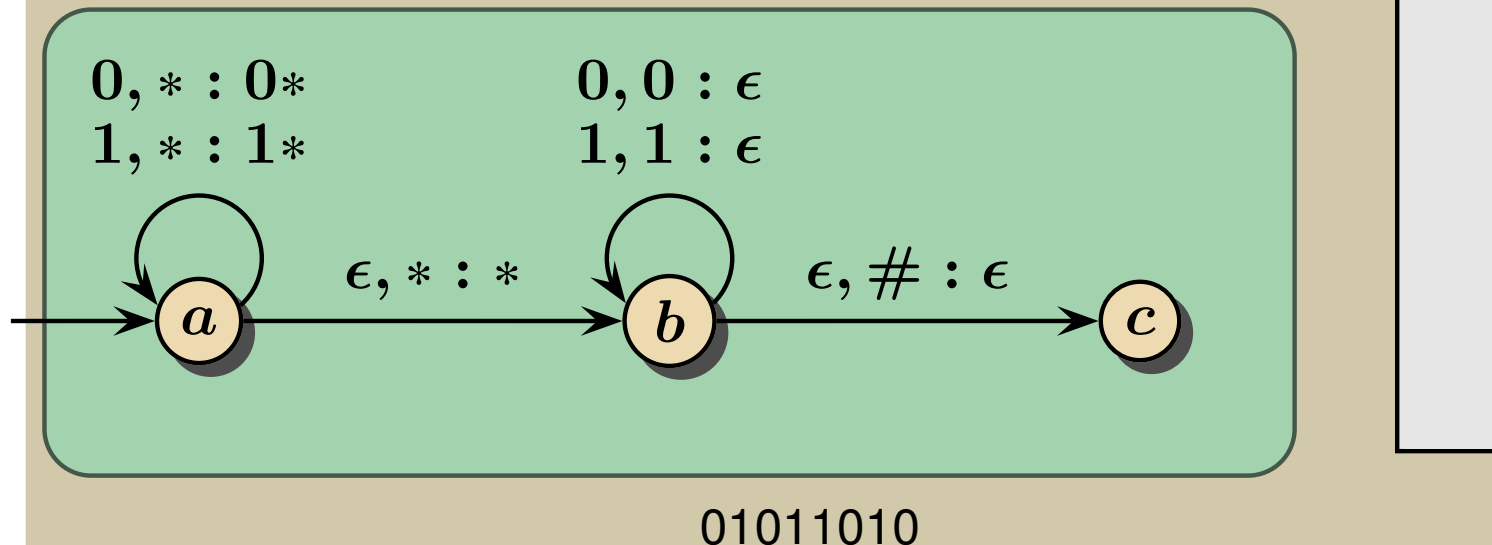
- $p \xrightarrow{\sigma, \tau : w} q$
steht für $(p, \sigma, \tau, q, w) \in \delta$

- Die abkürzende Schreibweise $\langle b \rangle, * : \langle b \rangle^*$ bedeutet, dass alle Übergänge der Art $\langle b \rangle, \tau : \langle b \rangle \tau$, mit $\tau \in \Gamma$ möglich sind

Noch ein PDA

Beispiel

- Kellerautomat \mathcal{A}_{rev} für $L_{\text{rev}} \stackrel{\text{def}}{=} \{ww^R \mid w \in \{0, 1\}^*\}$
- **Konstruktionsidee:**
 - „Rate“ die Stelle, an der w zu Ende ist
 - Kopiere bis zu dieser Stelle alles auf den Keller
 - Nach dieser Stelle vergleiche immer das nächste Eingabesymbol mit dem obersten Kellersymbol (und lösche dieses)



Kellerautomaten: Konfigurationen


- Das zukünftige Verhalten eines endlichen Automaten hängt jeweils ab von:
 - dem aktuellen Zustand,
 - den noch zu lesenden Eingabezeichen

- Das zukünftige Verhalten eines Kellerautomaten hängt jeweils ab von:
 - dem aktuellen Zustand,
 - den noch zu lesenden Eingabezeichen,
 - **dem Kellerinhalt**

→ der Kellerinhalt muss für die Definition der Semantik von Kellerautomaten berücksichtigt werden

- Läufe (Berechnungen) bestehen bei PDAs also nicht nur aus Folgen von Zuständen und gelesenen Zeichen
- Stattdessen werden wir Folgen von **Konfigurationen** betrachten, die jeweils die aktuelle „Situation“ beschreiben

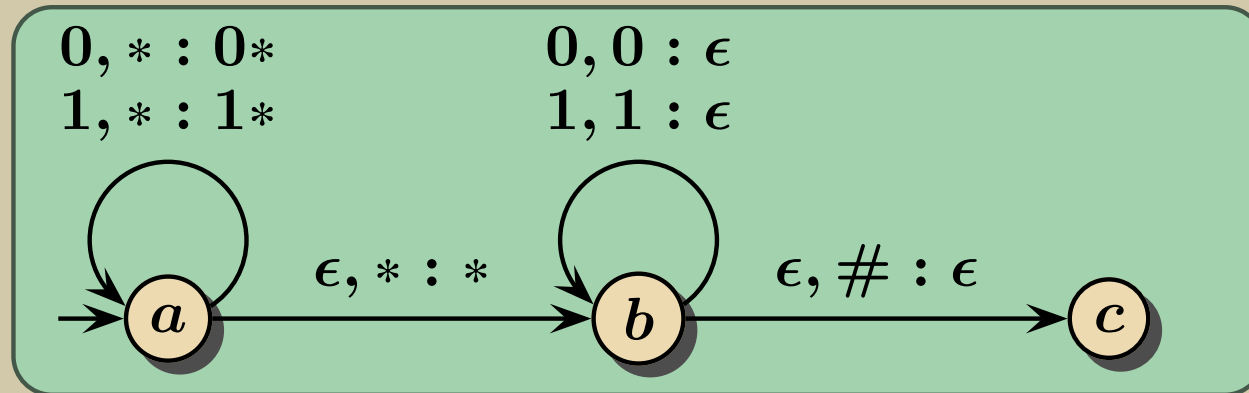
Definition (Konfiguration eines PDA)

- Sei $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, F)$ ein Kellerautomat
- Eine **Konfiguration** (q, u, v) von \mathcal{A} besteht aus:
 - einem Zustand $q \in Q$
 - der noch zu lesenden Eingabe $u \in \Sigma^*$
 - dem Kellerinhalt $v \in \Gamma^*$
 -  Das erste Zeichen von v ist das oberste Kellerzeichen!

- **Startkonfiguration** bei Eingabe w :
 (s, w, τ_0)

Konfigurationen: Beispiel

Beispiel




$(a, 01011010, \#) \vdash (a, 1011010, 0\#)$
 $\vdash (a, 011010, 10\#)$
 $\vdash (a, 11010, 010\#)$
 $\vdash (a, 1010, 1010\#)$
 $\vdash (b, 1010, 1010\#)$
 $\vdash (b, 010, 010\#)$
 $\vdash (b, 10, 10\#)$
 $\vdash (b, 0, 0\#)$
 $\vdash (b, \epsilon, \#)$
 $\vdash (c, \epsilon, \epsilon)$

Kellerautomaten: Konfigurationen und Berechnungen

Definition (Nachfolgekonfiguration)

- Sei $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, F)$ ein PDA
- Die Nachfolgekonfigurationsrelation $\vdash_{\mathcal{A}}$ ist wie folgt definiert
- Für alle $p, q \in Q, \sigma \in \Sigma, \tau \in \Gamma, u \in \Sigma^*, z, v \in \Gamma^*$ gilt:
- $(p, \sigma u, \tau v) \vdash_{\mathcal{A}} (q, u, zv)$, falls $(p, \sigma, \tau, q, z) \in \delta$
- $(p, u, \tau v) \vdash_{\mathcal{A}} (q, u, zv)$, falls $(p, \epsilon, \tau, q, z) \in \delta$
- Wenn $\underline{K \vdash_{\mathcal{A}} K'}$ gilt heißt K' (eine) Nachfolgekonfiguration von K

 Wenn der Automat \mathcal{A} durch den Kontext klar ist, lassen wir das Subskript \mathcal{A} meist weg

Definition (Berechnung eines PDA)

- Eine Berechnung (oder: ein Lauf) eines PDA \mathcal{A} ist eine Folge K_1, \dots, K_n von Konfigurationen mit
 - $K_i \vdash K_{i+1}$, für alle $i \in \{1, \dots, n-1\}$
- Schreibweise: $K_1 \vdash_{\mathcal{A}}^* K_n$

 Zu beachten:

- Wenn die Eingabe schon vollständig gelesen wurde, ist es immer noch möglich, ϵ -Übergänge auszuführen,
 - * aber: bei leerem Keller gibt es keine Nachfolgekonfiguration!

Kellerautomaten: Akzeptieren

- Wann akzeptiert \mathcal{A} die Eingabe?
- Bei den bisherigen Beispielen galten am Ende der Berechnung die beiden folgenden Aussagen:
 - der Keller ist leer
 - der Automat ist in einem „speziellen“ Zustand
- Wir definieren zwei Varianten von PDAs, deren Akzeptieren jeweils auf **einer** dieser beiden Bedingungen basiert
- Denn: mal ist das eine praktischer, mal das andere
- Dann werden wir sehen:
 - Beide Modelle sind äquivalent

Definition (Sprache eines PDA)

- Sei $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, F)$ ein Kellerautomat
- \mathcal{A} akzeptiert einen String $w \in \Sigma^*$, falls
 - $F = \emptyset$ und $(s, w, \tau_0) \vdash^* (q, \epsilon, \epsilon)$ für ein $q \in Q$
☞ „Akzeptieren mit leerem Keller“
 - oder
 - $F \neq \emptyset$ und $(s, w, \tau_0) \vdash^* (q, \epsilon, u)$ für ein $u \in \Gamma^*$ und $q \in F$
☞ „Akzeptieren mit akzeptierenden Zuständen“
- $\underline{L(\mathcal{A})} \stackrel{\text{def}}{=} \{w \mid \mathcal{A} \text{ akzeptiert } w\}$
- Wir sagen, dass \mathcal{A} die Sprache $L(\mathcal{A})$ entscheidet

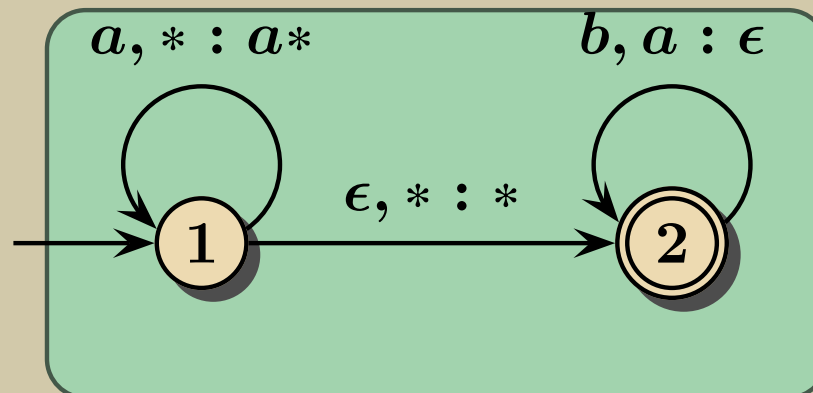
Akzeptierende Zustände: Beispielautomat

Beispiel

- $L = \{a^i b^j \mid i \geq j\}$

- Idee:

- 1.Phase: Lege jedes a auf den Keller ☞ Zustand 1
- 2.Phase: Lösche für jedes b ein a vom Keller ☞ Zustand 2
- Falls auf diese Weise der String ganz gelesen wird, akzeptiere
☞ auch wenn noch etwas im Keller steht



#

aaabb wird akzeptiert
aaabbbb wird nicht akzeptiert

Inhalt

9.1 Kellerautomaten: Definitionen

▷ **9.2 Leerer Keller vs. akzeptierende Zustände**

9.3 Grammatiken vs. Kellerautomaten

9.4 Kellerautomaten: Korrektheitsbeweise

9.5 Anhang: Beweisdetails

Leerer Keller vs. akzeptierende Zustände

- Dass PDAs mit leerem Keller oder mit akzeptierenden Zuständen definiert werden können, gibt uns bei der Konstruktion von PDAs eine gewisse Freiheit
- Der folgende Satz besagt aber, dass PDAs mit leerem Keller und PDAs mit akzeptierenden Zuständen genau dieselben Sprachen entscheiden können

Satz 9.1

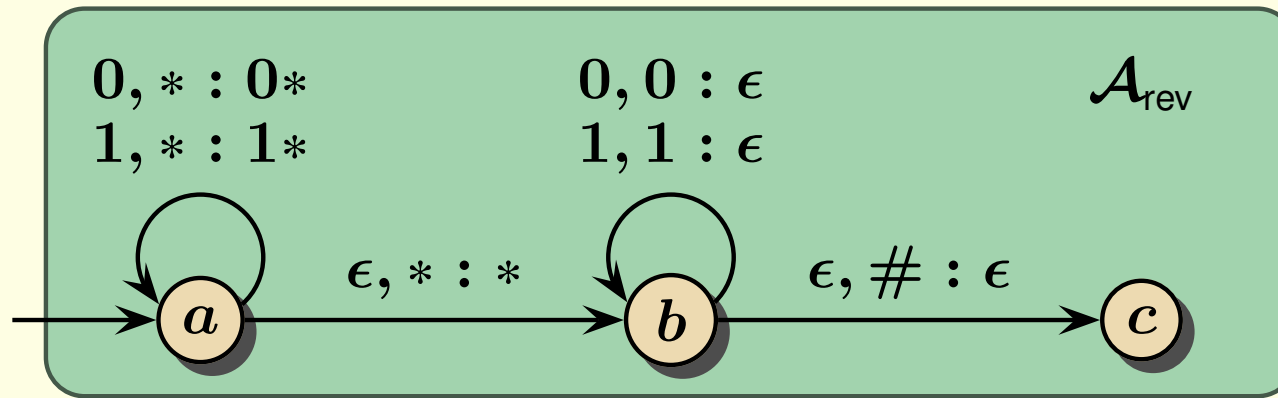
- (a) Für jeden Kellerautomaten \mathcal{A} , der mit leerem Keller akzeptiert, gibt es einen Kellerautomaten \mathcal{B} , der mit akzeptierenden Zuständen akzeptiert und $L(\mathcal{A}) = L(\mathcal{B})$ erfüllt
 - (b) Für jeden Kellerautomaten \mathcal{A} , der mit akzeptierenden Zuständen akzeptiert, gibt es einen Kellerautomaten \mathcal{B} , der mit leerem Keller akzeptiert und $L(\mathcal{A}) = L(\mathcal{B})$ erfüllt
- Beide Beweise verwenden die Methode der **Simulation**:
 - Der Automat \mathcal{B} ahmt jeweils das Verhalten von \mathcal{A} nach
 - Kurz: „ \mathcal{B} simuliert \mathcal{A} “

Leerer Keller \rightarrow akzeptierende Zustände: Idee

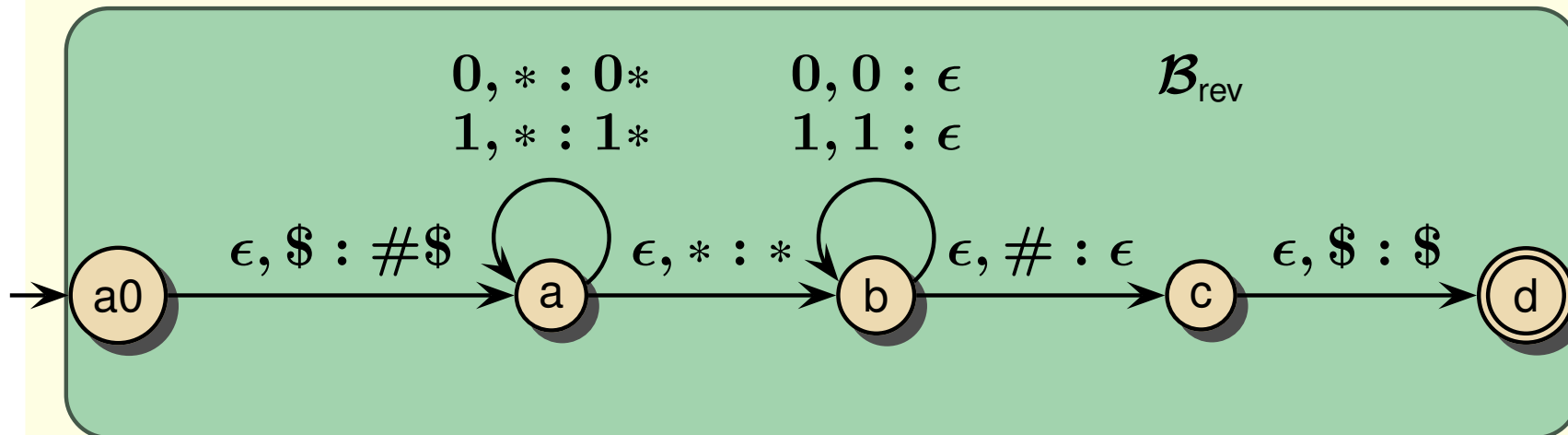
Beweisidee zu Satz 9.1 (a)

- „Leerer Keller \rightarrow akzeptierende Zustände“
- **Herausforderung:** wenn in \mathcal{A} der Keller leer wird, ist keine weitere Transition in einen akzeptierenden Zustand möglich
- **Idee:**
 - \mathcal{B} simuliert \mathcal{A}
 - \mathcal{B} verwendet gegenüber \mathcal{A} ein neues unterstes Kellersymbol $\$$, das zu Beginn der Simulation unter das unterste Kellersymbol τ_0 von \mathcal{A} gelegt wird
 - Wenn bei der Simulation in \mathcal{B} das Zeichen $\$$ „sichtbar“ wird, wäre in \mathcal{A} der Keller leer
 - Falls bei der Simulation von \mathcal{A} das Symbol $\$$ auf dem Keller zum Vorschein kommt, kann \mathcal{B} deshalb in den akzeptierenden Zustand übergehen

Leerer Keller \rightarrow akzeptierende Zustände: Beispiel



01011010

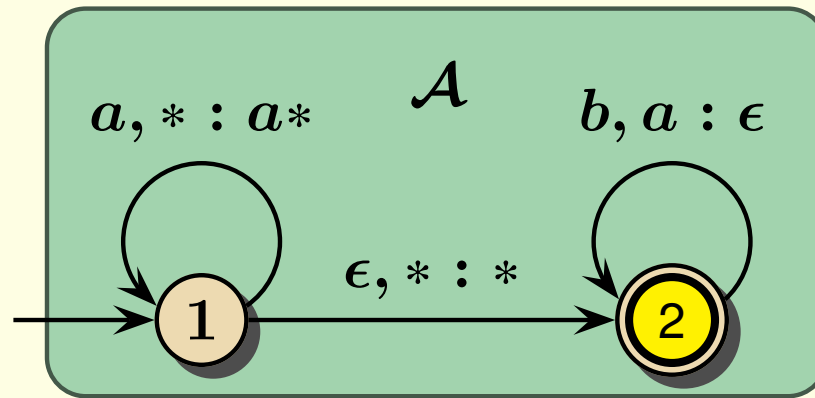


Akzeptierende Zustände → Leerer Keller: Idee

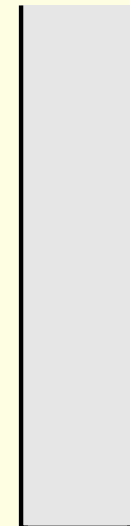
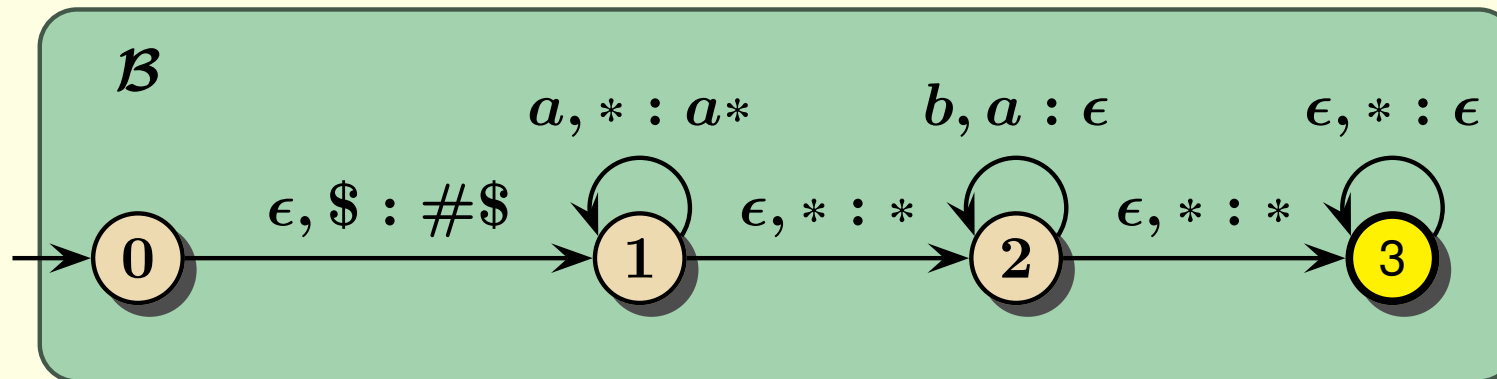
Beweisidee zu Satz 9.1 (b)

- „Akzeptierende Zustände → leerer Keller“
- **Herausforderungen:**
 - Wenn \mathcal{A} am Ende einen akzeptierenden Zustanderreicht, muss \mathcal{B} den Keller noch leeren
 - Wenn es eine Berechnung von \mathcal{A} gibt, die am Ende einen leeren Keller hat, aber keinen akzeptierenden Zustand, so soll diese Berechnung in \mathcal{B} *nicht* den Keller leeren
- **Idee:**
 - \mathcal{B} simuliert \mathcal{A}
 - Von jedem Zustand in F aus kann \mathcal{B} in den „Aufräumzustand“ q_a übergehen und dann den Keller mit Hilfe von ϵ -Übergängen leeren
 - Wenn die Eingabe vollständig gelesen war, führt das zum Akzeptieren mit leerem Keller
 - Damit Berechnungen von \mathcal{A} , die den Keller lehren ohne in einen akzeptierenden Zustand überzugehen, nicht fälschlich zum Akzeptieren von \mathcal{B} führen, verwendet \mathcal{B} wieder ein neues unterstes Kellersymbol $\$$

Akzeptierende Zustände \rightarrow Leerer Keller: Beispiel



aaabb|



Inhalt

9.1 Kellerautomaten: Definitionen

9.2 Leerer Keller vs. akzeptierende Zustände

▷ **9.3 Grammatiken vs. Kellerautomaten**

9.4 Kellerautomaten: Korrektheitsbeweise

9.5 Anhang: Beweisdetails

Äquivalenz von Grammatiken und Kellerautomaten

- Ziel: Nachweis, dass Kellerautomaten genau die kontextfreien Sprachen entscheiden

Satz 9.2

- Zu jeder kontextfreien Grammatik G gibt es einen Kellerautomaten \mathcal{A} mit $L(\mathcal{A}) = L(G)$
- Der Beweis von Satz 9.2 ist nicht sehr schwierig und folgt einer einfachen Idee:
 - Der Kellerautomat versucht eine Linksableitung zu finden

Satz 9.3

- Zu jedem Kellerautomaten \mathcal{A} gibt es eine kontextfreie Grammatik G mit $L(G) = L(\mathcal{A})$
- Der Beweis von Satz 9.3 ist deutlich komplizierter

Grammatik → Kellerautomat: Idee

Satz 9.2



- Zu jeder kontextfreien Grammatik G gibt es einen Kellerautomaten \mathcal{A} mit
$$L(\mathcal{A}) = L(G)$$

Beweisidee

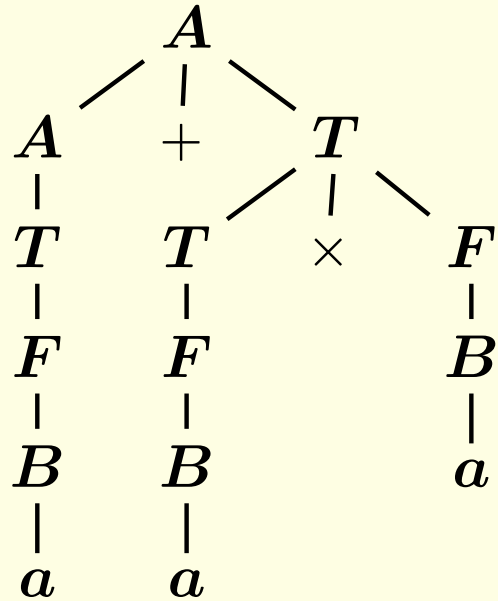
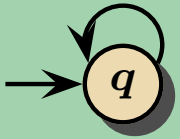
- Sei $G = (V, \Sigma, S, P)$
- **Idee:**
 - Der Kellerautomat \mathcal{A} erzeugt bei Eingabe w eine Linksableitung für ein Wort v und testet, dass $w = v$ gilt
 - Erzeugen und Testen sind dabei ineinander verschränkt:
 - * Wenn die aktuelle Satzform mit einem Terminalsymbol anfängt, wird dieses gleich mit w verglichen

Beweisidee (Forts.)

- Ein einzelner Schritt einer Linksableitung
 - ersetzt in einer Satzform der Art $uX\alpha$
 - die Variable X durch einen String β
 - gemäß einer Regel $X \rightarrow \beta$

 mit $u \in \Sigma^*$, $X \in V$,
 $\alpha \in (\Sigma \cup V)^*$
- In \mathcal{A} soll dies der folgenden Situation entsprechen:
 - u ist schon gelesen, $X\alpha$ ist der Kellerinhalt
 - Zur Umsetzung des Ableitungsschrittes geht \mathcal{A} wie folgt vor:
 1. „Rate“ Regel $X \rightarrow \beta \in P$
 2. Ersetze auf dem Keller X durch β
 3. Vergleiche die führenden Terminalsymbole von $\beta\alpha$ mit den nächsten Zeichen der Eingabe und reduziere sie  d.h., lösche sie vom Keller

Grammatik \rightarrow Kellerautomat: Beispiel

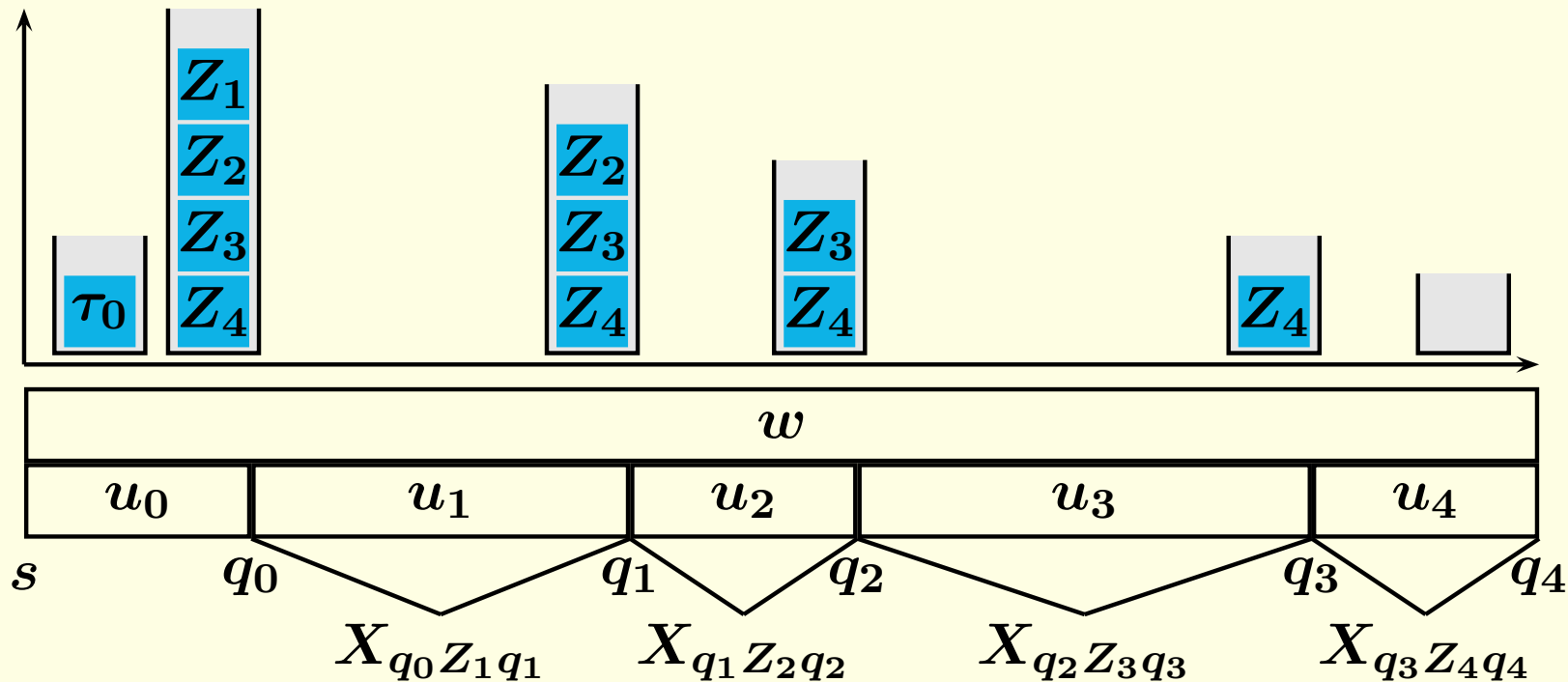
$$\begin{array}{l} A \rightarrow A + T \mid T \\ T \rightarrow T \times F \mid F \\ F \rightarrow (A) \mid B \\ B \rightarrow a \mid b \mid Ba \mid \\ \quad Bb \mid B0 \mid B1 \end{array}$$
$$\begin{array}{l} a, a : \epsilon \\ b, b : \epsilon \\ 0, 0 : \epsilon \\ 1, 1 : \epsilon \\ \times, \times : \epsilon \\ +, + : \epsilon \\ (, (: \epsilon \\),) : \epsilon \\ \epsilon, A : A + T \\ \epsilon, A : T \\ \epsilon, T : T \times F \\ \epsilon, T : F \\ \epsilon, F : (A) \\ \epsilon, F : B \\ \epsilon, B : a \\ \epsilon, B : b \\ \epsilon, B : Ba \\ \epsilon, B : Bb \\ \epsilon, B : B0 \\ \epsilon, B : B1 \end{array}$$

$$a + a \times a$$
$$\begin{array}{l}
(q, a + a \times a, A) \\
\vdash (q, a + a \times a, A + T) \\
\vdash (q, a + a \times a, T + T) \\
\vdash (q, a + a \times a, F + T) \\
\vdash (q, a + a \times a, B + T) \\
\vdash (q, a + a \times a, a + T) \\
\vdash (q, + a \times a, + T) \\
\vdash (q, a \times a, T) \\
\vdash (q, a \times a, T \times F) \\
\vdash (q, a \times a, F \times F) \\
\vdash (q, a \times a, B \times F) \\
\vdash (q, a \times a, a \times F) \\
\vdash (q, \times a, \times F) \\
\vdash (q, a, F) \\
\vdash (q, a, B) \\
\vdash (q, a, a) \\
\vdash (q, \epsilon, \epsilon)
\end{array}$$

Grammatik \rightarrow Kellerautomat: Konstruktion

Beweis von Satz 9.2

- Sei $G = (V, \Sigma, S, P)$
- $\mathcal{A} \stackrel{\text{def}}{=} (\{q\}, \Sigma, V \cup \Sigma, \delta, q, S, \emptyset)$,
 - $\delta \stackrel{\text{def}}{=} \{(q, \sigma, \sigma, q, \epsilon) \mid \sigma \in \Sigma\} \cup \{(q, \epsilon, X, q, \alpha) \mid X \rightarrow \alpha \in P\}$
- Eine detaillierte Beweisskizze findet sich im Anhang

Kellerautomat → Grammatik: Idee

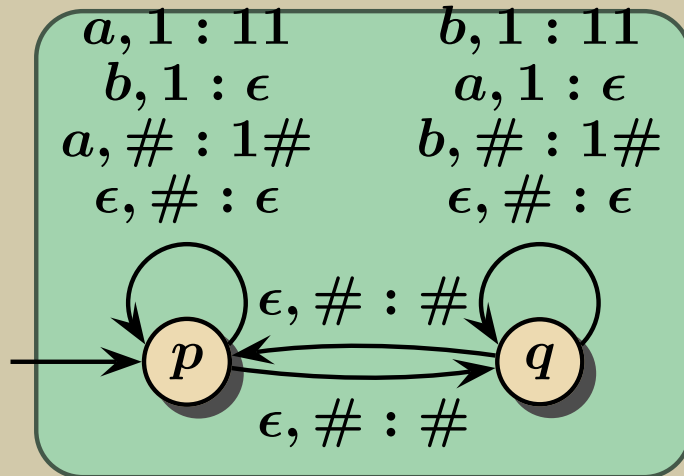


- Im ersten Schritt ersetzt \mathcal{A} das unterste Kellersymbol durch einen String $Z_1 \cdots Z_k$ und liest ein Präfix u_0 der Eingabe w $\Rightarrow u_0 \in \Sigma \cup \{\epsilon\}$
- Die Zeichen Z_1, \dots, Z_k werden dann im Rest der Berechnung nach und nach wieder vom Keller gelöscht
- Dabei werden Teilstrings u_1, \dots, u_k der Eingabe gelesen
- Idee für die Grammatik:
 - Für jede Kombination $p, p' \in Q, \tau \in \Gamma$ enthält G eine Variable $X_{p, \tau, p'}$, die alle Strings erzeugt, für die \mathcal{A} eine Teilberechnung von Zustand p in Zustand p' hat, die insgesamt das Zeichen τ vom Keller löscht

Kellerautomat → Grammatik: Beispiel (1/3)

Beispiel

- Kellerautomat für $L_{a=b} = \{w \in \{a, b\}^* \mid \#_a(w) = \#_b(w)\}$:



- Zustand p :
 - * Mindestens so viele a wie b gelesen
 - * Anzahl der Einsen auf dem Keller entspricht Überschuss an a 's
- Zustand q : analog, aber mindestens so viele b wie a gelesen

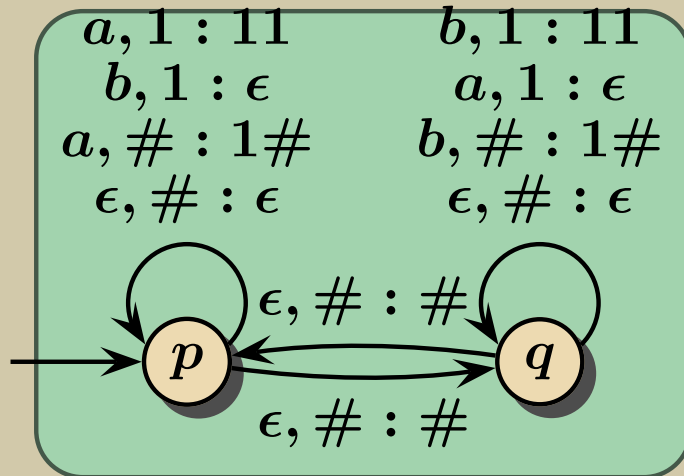
- Der PDA akzeptiert einen String w , wenn er bei Lesen von w insgesamt das Kellersymbol $\#$ löscht und dabei von p nach p oder q übergeht
- Das entspricht den „Start-Regeln“: $S \rightarrow X_{p\#p} \mid X_{p\#q}$

- Die Regeln für $X_{p\#p}$ ergeben sich aus den drei Transitionen, die für den PDA möglich sind, wenn $\#$ das oberste Kellersymbol ist
 - Er kann ein a lesen, eine 1 auf den Keller legen und im Zustand p bleiben
 - * Um aus der entstehenden Situation den Keller zu leeren, muss zuerst 1 und dann $\#$ vom Keller gelöscht werden
 - * Die Teilberechnung, die die 1 löscht, kann in p oder in q enden
 - * Die entsprechenden Regeln sind: $X_{p\#p} \rightarrow aX_{p1p}X_{p\#p} \mid aX_{p1q}X_{q\#p}$

Kellerautomat → Grammatik: Beispiel (2/3)

Beispiel

- Kellerautomat für $L_{a=b} = \{w \in \{a, b\}^* \mid \#_a(w) = \#_b(w)\}$:



- Zustand p :
 - * Mindestens so viele a wie b gelesen
 - * Anzahl der Einsen auf dem Keller entspricht Überschuss an a 's
- Zustand q : analog, aber mindestens so viele b wie a gelesen

- Die Regeln für $X_{p\#p}$ ergeben sich aus den drei Transitionen, die für den PDA möglich sind, wenn $\#$ das oberste Kellersymbol ist

(2) Er kann ohne etwas zu lesen, eine 1 auf den Keller legen und $\#$ vom Keller löschen und im Zustand p bleiben

* Die entsprechende Regel ist: $X_{p\#p} \rightarrow \epsilon$

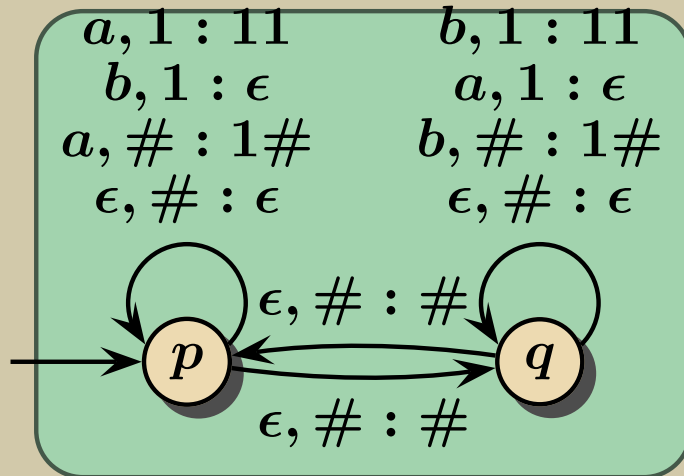
(3) Er kann ohne etwas zu lesen und ohne den Keller zu verändern in Zustand q übergehen

* Die entsprechende Regel ist: $X_{p\#p} \rightarrow X_{q\#p}$

Kellerautomat → Grammatik: Beispiel (3/3)

Beispiel

- Kellerautomat für $L_{a=b} = \{w \in \{a, b\}^* \mid \#_a(w) = \#_b(w)\}$:



- Zustand p :
 - * Mindestens so viele a wie b gelesen
 - * Anzahl der Einsen auf dem Keller entspricht Überschuss an a 's
- Zustand q : analog, aber mindestens so viele b wie a gelesen

- Die daraus entstehende Grammatik:

$$\begin{aligned}
 S &\rightarrow X_{p\#p} \mid X_{p\#q} \\
 X_{p\#p} &\rightarrow aX_{p1p}X_{p\#p} \mid aX_{p1q}X_{q\#p} \mid \epsilon \mid X_{q\#p} \\
 X_{p\#q} &\rightarrow aX_{p1p}X_{p\#q} \mid aX_{p1q}X_{q\#q} \mid X_{q\#q} \\
 X_{p1p} &\rightarrow aX_{p1p}X_{p1p} \mid aX_{p1q}X_{q1p} \mid b \\
 X_{p1q} &\rightarrow aX_{p1p}X_{p1q} \mid aX_{p1q}X_{q1q} \\
 X_{q\#q} &\rightarrow bX_{q1q}X_{q\#q} \mid bX_{q1p}X_{p\#q} \mid \epsilon \mid X_{p\#q} \\
 X_{q\#p} &\rightarrow bX_{q1q}X_{q\#p} \mid bX_{q1p}X_{p\#p} \mid X_{p\#p} \\
 X_{q1q} &\rightarrow bX_{q1q}X_{q1q} \mid bX_{q1p}X_{p1q} \mid a \\
 X_{q1p} &\rightarrow bX_{q1q}X_{q1p} \mid bX_{q1p}X_{p1p}
 \end{aligned}$$

Kellerautomat \rightarrow Grammatik: Beweis (1/2)

Satz 9.3

- Zu jedem Kellerautomaten \mathcal{A} gibt es eine kontextfreie Grammatik G mit $L(G) = L(\mathcal{A})$

Beweisidee

- Sei $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, \emptyset)$
 - (oBdA: \mathcal{A} akzeptiert mit leerem Keller)
- Weitere Annahme (oBdA): \mathcal{A} legt bei jeder Transition maximal zwei Zeichen auf den Keller
- Wir konstruieren eine Grammatik $G_{\mathcal{A}}$ mit Variablen $X_{p,\tau,p'}$, für alle $p, p' \in Q$ und $\tau \in \Gamma$ mit der folgenden Intention:
 - $X_{p,\tau,p'} \Rightarrow^* w$ soll gelten, falls \mathcal{A} durch Lesen von w vom Zustand p in den Zustand p' kommen und dabei insgesamt τ vom Keller löschen kann
 - Formal soll also gelten:
$$X_{p,\tau,p'} \Rightarrow^* w \iff (p, w, \tau) \vdash_{\mathcal{A}}^* (p', \epsilon, \epsilon)$$

Kellerautomat → Grammatik: Beweis (2/2)

Beweis von Satz 9.3

- P enthält pro Tupel in δ eine oder mehrere Regeln, jeweils für alle möglichen Zustände p_1, p_2 :

δ	P
$(p, \alpha, \tau, q, \epsilon)$	$X_{p,\tau,q} \rightarrow \alpha$
$(p, \alpha, \tau, q, \tau_1)$	$X_{p,\tau,p_1} \rightarrow \alpha X_{q,\tau_1,p_1}$
$(p, \alpha, \tau, q, \tau_1 \tau_2)$	$X_{p,\tau,p_2} \rightarrow \alpha X_{q,\tau_1,p_1} X_{p_1,\tau_2,p_2}$

- Dabei ist $\alpha \in \Sigma \cup \{\epsilon\}$  also: Zeichen oder Leerstring

- Zusätzlich hat $G_{\mathcal{A}}$ das Startsymbol S und Regeln $S \rightarrow X_{s,\tau_0,q}$, für jedes $q \in Q$

- **Behauptung:**

$$X_{p,\tau,p'} \Rightarrow^* w \iff (p, w, \tau) \vdash_{\mathcal{A}}^* (p', \epsilon, \epsilon)$$

- „ \Leftarrow “: Induktion nach der Anzahl der Berechnungsschritte
- „ \Rightarrow “: Induktion nach der Anzahl der Ableitungsschritte
- Die Beweisdetails finden sich im Anhang

Kellerautomaten und Grammatiken: Fazit

Satz 9.4

- Für eine Sprache L sind äquivalent:
 - L ist kontextfrei
 - L wird von einem Kellerautomaten mit akzeptierenden Zuständen entschieden
 - L wird von einem Kellerautomaten mit leerem Keller entschieden

- Wie groß werden die bei der Umwandlung konstruierten Objekte?
 - Grammatik \rightarrow Kellerautomat: $\mathcal{O}(n)$
 - Kellerautomat \rightarrow Grammatik: $\mathcal{O}(n^4)$
 - Zwischen Kellerautomaten: $\mathcal{O}(n)$

- Aus Satz 9.3 und dem Beweis von Satz 9.2 folgt außerdem folgende Normalform für Kellerautomaten:

Folgerung

- Zu jedem Kellerautomaten gibt es einen äquivalenten Kellerautomaten mit nur einem Zustand

Inhalt

9.1 Kellerautomaten: Definitionen

9.2 Leerer Keller vs. akzeptierende Zustände

9.3 Grammatiken vs. Kellerautomaten

▷ **9.4 Kellerautomaten: Korrektheitsbeweise**

9.5 Anhang: Beweisdetails

Intervall-Notation für Teilstrings

- Wir verwenden zukünftig die folgende Notation, um über Teilstrings und einzelne Zeichen von Strings zu sprechen
- Sei $w = \sigma_1 \cdots \sigma_n$ ein String
(der Länge n)
- Dann sei, für alle $i, j \in \{1, \dots, n\}$ mit $i \leq j$:
 - $\underline{w[i]} \stackrel{\text{def}}{=} \sigma_i$
 - $\underline{w[i, j]} \stackrel{\text{def}}{=} \sigma_i \cdots \sigma_j$
 - $\underline{w[i, j)} \stackrel{\text{def}}{=} \sigma_i \cdots \sigma_{j-1}$
 - $\underline{w(i, j]} \stackrel{\text{def}}{=} \sigma_{i+1} \cdots \sigma_j$
 - $\underline{w(i, j)} \stackrel{\text{def}}{=} \sigma_{i+1} \cdots \sigma_{j-1}$
(nur für $i < j$)
 - $\underline{w[* , j]} \stackrel{\text{def}}{=} \sigma_1 \cdots \sigma_j$
 - $\underline{w[i, *]} \stackrel{\text{def}}{=} \sigma_i \cdots \sigma_n$
- Für $i > j$ sei $w[i, j] \stackrel{\text{def}}{=} \epsilon$

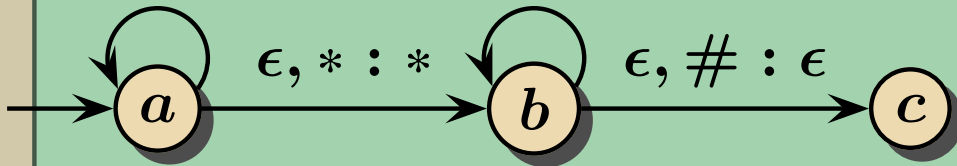
Beispiel

- Sei $w = acbbcabba$
- Dann ist
 - $w[3] = b$
 - $w[4, 6] = bca$
 - $w[4, 6) = bc$
 - $w(4, 6] = ca$
 - $w(4, 6) = c$
 - $w(4, 5) = \epsilon$
 - $w[* , 3] = acb$
 - $w[5, *] = cabba$

Kellerautomaten: Korrektheitsbeweise (1/2)

Beispiel

$0, * : 0^*$ $0, 0 : \epsilon$
 $1, * : 1^*$ $1, 1 : \epsilon$



Proposition 9.5

$$L(\mathcal{A}_{\text{rev}}) = L_{\text{rev}}$$

- Zur Erinnerung:

$$L_{\text{rev}} = \{ww^R \mid w \in \{0, 1\}^*\}$$

☞ Palindrome gerader Länge

Beweisskizze

- Wir beweisen:

- $L_{\text{rev}} \subseteq L(\mathcal{A}_{\text{rev}})$ ☞ Vollständigkeit
- $L(\mathcal{A}_{\text{rev}}) \subseteq L_{\text{rev}}$ ☞ Korrektheit

Beweisskizze für „ $L_{\text{rev}} \subseteq L(\mathcal{A}_{\text{rev}})$ “

- Wir zeigen, dass für beliebige $w \in \Sigma^*$ der String ww^R von \mathcal{A}_{rev} akzeptiert wird
- Dazu lässt sich durch Induktion nach der Länge von u bzw. v für beliebige Strings $u, v, x \in \{0, 1\}^*$ beweisen:
 - (a) $(a, ux, \#) \vdash^* (a, x, u^R \#)$ und
 - (b) $(b, v, v\#) \vdash^* (b, \epsilon, \#)$

- Dann folgt:

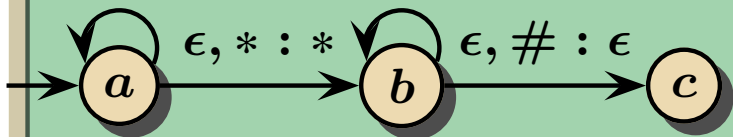
$$\begin{aligned}
 (a, ww^R, \#) &\vdash^* (a, w^R, w^R \#) \\
 &\vdash (b, w^R, w^R \#) \\
 &\vdash^* (b, \epsilon, \#) \\
 &\vdash (c, \epsilon, \epsilon)
 \end{aligned}$$

➡ $ww^R \in L(\mathcal{A}_{\text{rev}})$

Kellerautomaten: Korrektheitsbeweise (2/2)

Beispiel

$0, * : 0*$ $0, 0 : \epsilon$
 $1, * : 1*$ $1, 1 : \epsilon$



Beweisskizze: „ $L(\mathcal{A}_{\text{rev}}) \subseteq L_{\text{rev}}$ “

- Klar: ein String v ist genau dann in L_{rev} , wenn
 - er gerade Länge $n = 2k$ hat und
 - für jedes $i \leq k$ gilt:
 $v[i] = v[n - i + 1]$

Beweisskizze (Forts.)

- Sei v ein von \mathcal{A}_{rev} akzeptierter String mit n Zeichen
 ➡ $(a, v, \#) \vdash^* (b, \epsilon, \#) \vdash (c, \epsilon, \epsilon)$
- Nach Konstruktion von \mathcal{A} verläuft diese Berechnung in drei Phasen
 1. \mathcal{A}_{rev} liest ein Präfix $v[1, k]$ der Eingabe (für ein $k \leq n$) und schreibt es zeichenweise auf den Keller,
 2. dann geht \mathcal{A}_{rev} in den Zustand b über
 3. schließlich liest \mathcal{A}_{rev} die restliche Eingabe $v[k + 1, n]$ und vergleicht sie mit den zuvor auf den Keller geschriebenen Zeichen
- Durch Induktion lässt sich zeigen:
 - Die Konfiguration nach Phase 1 ist
 $(a, v[k + 1, n], v[1, k]^R \#)$
 - Damit Phase 3 erfolgreich ist, muss gelten:
 - * $n - k = k \Rightarrow n = 2k$
 - * für jedes $i \leq k$ ist $v[i] = v[n - i + 1]$
- ➡ $v \in L_{\text{rev}}$

Zusammenfassung

- Kellerautomaten entstehen durch Erweiterung von ϵ -NFAs um einen Keller (LIFO)
- Kellerautomaten, die durch leeren Keller akzeptieren, sind genauso mächtig wie Kellerautomaten, die mit akzeptierenden Zuständen akzeptieren
- Mit Kellerautomaten und kontextfreien Grammatiken lassen sich genau dieselben Sprachen beschreiben: die kontextfreien Sprachen
- Der Kellerautomat zu einer Grammatik versucht eine Linksableitung zu finden
- Die Konstruktion der Grammatik zu einem Kellerautomaten ist erheblich komplizierter

Inhalt

9.1 Kellerautomaten: Definitionen

9.2 Leerer Keller vs. akzeptierende Zustände

9.3 Grammatiken vs. Kellerautomaten

9.4 Kellerautomaten: Korrektheitsbeweise

▷ **9.5 Anhang: Beweisdetails**

Leerer Keller vs. akzeptierende Zustände: Beweisideen (1/4)

- Der Beweis der Äquivalenz der beiden Akzeptiermethoden von PDAs verwendet mehrfach die folgende einfache Erkenntnis:
 - Eine Berechnung wird nur von den wirklich gelesenen Zeichen der Eingabe und des Kellers beeinflusst:
- (a) Deshalb können die Schritte einer Berechnung immer noch ausgeführt werden, wenn hinter der Eingabe und unter dem Keller etwas hinzugefügt wird
- (b) Andererseits können Zeichen der Eingabe, die während einer (partiellen) Berechnung (noch) nicht gelesen wurden und Zeichen des Kellers, die niemals sichtbar werden, entfernt werden, ohne die Berechnung zu beeinflussen

• Notation: $\underline{K \vdash_{(\gamma)}^* K'} \stackrel{\text{def}}{\iff}$

es gibt eine Berechnung $K \vdash \dots \vdash K'$, in der jede Konfiguration **vor** K' einen String $u\gamma$ mit $u \neq \epsilon$ im Keller stehen hat

 In K' kann γ „alleine“ im Keller stehen

Lemma 9.6



- Sei $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, F)$ ein Kellerautomat
- Seien
 - $x, y, w \in \Sigma^*$,
 - $\alpha \in \Gamma^+, \beta, \gamma \in \Gamma^*$,
 - $p, q \in Q$
- Dann sind äquivalent:
 - (a) $(p, x, \alpha) \vdash^* (q, y, \beta)$
 - (b) $(p, xw, \alpha\gamma) \vdash_{(\gamma)}^* (q, yw, \beta\gamma)$
- Der Beweis kann leicht durch Induktion nach der Berechnungslänge geführt werden

Leerer Keller vs. akzeptierende Zustände: Beweisideen (2/4)

Beweisidee zu Satz 9.1 (a)

- „Leerer Keller \rightarrow akzeptierende Zustände“
- **Idee:**
 - \mathcal{B} simuliert \mathcal{A}
 - \mathcal{B} verwendet gegenüber \mathcal{A} ein neues unterstes Kellersymbol $\$$, das zu Beginn der Simulation unter das unterste Kellersymbol τ_0 von \mathcal{A} gelegt wird
 - Wenn bei der Simulation in \mathcal{B} das Zeichen $\$$ „sichtbar“ wird, wäre in \mathcal{A} der Keller leer
 - Falls bei der Simulation von \mathcal{A} das Symbol $\$$ auf dem Keller zum Vorschein kommt, kann \mathcal{B} deshalb in den akzeptierenden Zustand übergehen

Beweisansatz zu Satz 9.1 (a)

- Sei $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, \emptyset)$
- Sei $\mathcal{B} \stackrel{\text{def}}{=} (Q \cup \{q_0, q_a\}, \Sigma, \Gamma \cup \{\$\}, \delta', q_0, \$, \{q_a\})$
- Dabei sind:
 - $q_0, q_a \notin Q$ neue Zustände, und
 - $\$ \notin \Gamma$ ein neues Kellersymbol
- δ' enthält:
 - alle Transitionen von δ
 - $(q_0, \epsilon, \$, s, \tau_0 \$)$  Initialisierung
 - $(q, \epsilon, \$, q_a, \$)$, für alle $q \in Q$
 entspricht leerem Keller in \mathcal{A}

Leerer Keller vs. akzeptierende Zustände: Beweisideen (3/4)

Beweisdetails zu Satz 9.1 (a)

- Zur Erinnerung:
 - $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, \emptyset)$
 - $\mathcal{B} \stackrel{\text{def}}{=} (Q \cup \{q_0, q_a\}, \Sigma, \Gamma \cup \{\$, \delta', q_0, \$, \{q_a\})$
 - δ' enthält:
 - * alle Transitionen von δ
 - * $(q_0, \epsilon, \$, s, \tau_0 \$)$
 - * $(q, \epsilon, \$, q_a, \$)$,
 für alle $q \in Q$
- **Behauptung:** $L(\mathcal{B}) = L(\mathcal{A})$
- Ausnahmsweise zeigen wir nicht zwei Inklusionen sondern direkt, dass für alle Strings $w \in \Sigma^*$ gilt:

$$w \in L(\mathcal{B}) \iff w \in L(\mathcal{A})$$
- Für den Beweis sei $w \in \Sigma^*$ beliebig

Beweisdetails zu Satz 9.1 (a) (Forts.)

- Der erste Schritt von \mathcal{B} bei Eingabe w ist auf jeden Fall $(q_0, w, \$) \vdash_{\mathcal{B}} (s, w, \tau_0 \$)$
- Da \mathcal{B} genau dann in q_a übergeht, wenn $\$$ oberstes Kellersymbols ist, gilt für alle $u \in \Gamma^*$:

$$(s, w, \tau_0 \$) \vdash_{\mathcal{B}}^* (q_a, \epsilon, u) \Rightarrow u = \$$$
 Und: $(s, w, \tau_0 \$) \vdash_{\mathcal{B}}^* (q_a, \epsilon, \$) \Rightarrow$
 es gibt ein $q: (s, w, \tau_0 \$) \vdash_{\mathcal{B},(\$)}^* (q, \epsilon, \$)$
- Wegen Lemma 9.6 gilt, für alle q :

$$(s, w, \tau_0 \$) \vdash_{\mathcal{B},(\$)}^* (q, \epsilon, \$) \iff (s, w, \tau_0) \vdash_{\mathcal{B}}^* (q, \epsilon, \epsilon)$$
- Da \mathcal{A} und \mathcal{B} identisch arbeiten, solange $\$$ nicht zu sehen ist, gilt, für alle q :

$$(s, w, \tau_0) \vdash_{\mathcal{B}}^* (q, \epsilon, \epsilon) \iff (s, w, \tau_0) \vdash_{\mathcal{A}}^* (q, \epsilon, \epsilon)$$
- Insgesamt haben wir also:



$$(s, w, \tau_0 \$) \vdash_{\mathcal{B}}^* (q_a, \epsilon, u) \iff \text{es gibt ein } q: (s, w, \tau_0) \vdash_{\mathcal{A}}^* (q, \epsilon, \epsilon)$$
- ➔ $w \in L(\mathcal{B}) \iff w \in L(\mathcal{A})$

Leerer Keller vs. akzeptierende Zustände: Beweiseideen (4/4)

Beweisidee zu Satz 9.1 (b)

- „Akzeptierende Zustände \rightarrow leerer Keller“
- **Idee:**
 - \mathcal{B} simuliert \mathcal{A}
 - Von jedem Zustand in F aus kann \mathcal{B} in den „Aufräumzustand“ q_a übergehen und dann den Keller mit Hilfe von ϵ -Übergängen leeren
 - Wenn die Eingabe vollständig gelesen war, führt das zum Akzeptieren mit leerem Keller
 - Damit keine Berechnung von \mathcal{A} fälschlich zum Akzeptieren von \mathcal{B} führt, indem \mathcal{A} den Keller selbst leert (ohne in einen akzeptierenden Zustand zu gehen), verwendet \mathcal{B} wieder ein neues unterstes Kellersymbol $\$$

Beweisansatz zu Satz 9.1 (b) (Forts.)

- Sei $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, s, \tau_0, F)$
- Sei $\mathcal{B} \stackrel{\text{def}}{=} (Q \cup \{q_0, q_a\}, \Sigma, \Gamma \cup \{\$, \}, \delta', q_0, \$, \emptyset)$
 - $q_0, q_a \notin Q, \$ \notin \Gamma$ (wie zuvor)
- δ' enthält:
 - alle Transitionen aus δ
 - $(q_0, \epsilon, \$, s, \tau_0 \$)$  Initialisierung
 - $(q, \epsilon, \tau, q_a, \tau)$, für alle $q \in F$
 - * Aus akzeptierenden Zuständen ist ein Übergang nach q_a möglich
 - $(q_a, \epsilon, \tau, q_a, \epsilon)$  zum Leeren des Kellers
- **Behauptung:** $L(\mathcal{B}) = L(\mathcal{A})$
(ohne Beweis)

Grammatik → Kellerautomat: Beweisdetails (1/3)

Beweis von Satz 9.2

- Sei $G = (V, \Sigma, S, P)$
- $\mathcal{A} \stackrel{\text{def}}{=} (\{q\}, \Sigma, V \cup \Sigma, \delta, q, S, \emptyset)$,
 - $\delta \stackrel{\text{def}}{=} \{(q, \sigma, \sigma, q, \epsilon) \mid \sigma \in \Sigma\} \cup \{(q, \epsilon, X, q, \alpha) \mid X \rightarrow \alpha \in P\}$

- **Behauptung:** $L(\mathcal{A}) = L(G)$
 - Wir führen den Beweis nur für Grammatiken in Chomsky-Normalform

- **Wir zeigen zuerst:** $L(G) \subseteq L(\mathcal{A})$
 - Sei $w \in L(G)$ und sei

$$S \Rightarrow \gamma_1 \Rightarrow \gamma_2 \Rightarrow \dots \Rightarrow \gamma_n = w$$
 eine Linksableitung für w mit:
 - * $\gamma_i = u_i X_i \alpha_i$
 - * $z_i \stackrel{\text{def}}{=} \text{Suffix von } w \text{ mit } w = u_i z_i$
 - Dabei sind:
 - * $u_i, z_i \in \Sigma^*, X_i \in V, \alpha_i \in V^*$,
für $i < n$
 - * $u_n = w, X_n \alpha_n = \epsilon, z_n = \epsilon$

Beweis (Forts.)

- X_i ist also die am weitesten links stehende Variable der i -ten Satzform
- u_i ist der String aus Terminalzeichen links davon
- α_i ist der String rechts davon, der nur aus Variablen besteht, da dies eine Linksableitung zu einer CNF-Grammatik ist

- Die im $(i + 1)$ -ten Schritt angewendete Regel sei
 - $X_i \rightarrow Y_i Z_i$ oder
 - $X_i \rightarrow \sigma_i$
- Dabei sind:
 - $\sigma_i \in \Sigma$
 - $Y_i, Z_i \in V$

- Wir zeigen durch Induktion nach i , dass für alle $i \geq 0$ gilt:

$$(q, w, S) \vdash_{\mathcal{A}}^* (q, z_i, X_i \alpha_i)$$

Grammatik → Kellerautomat: Beweisdetails (2/3)

Beweisdetails für $L(G) \subseteq L(\mathcal{A})$

- Ind.-Beh.: $(q, w, S) \vdash_{\mathcal{A}}^* (q, z_i, X_i \alpha_i)$
 - $i = 0: \checkmark$ ($z_0 = w, \alpha_0 = \epsilon$)
 - Von i zu $i + 1$:
 - Nach Induktion gilt:

$$(q, w, S) \vdash_{\mathcal{A}}^* (q, z_i, X_i \alpha_i)$$
 - Wir unterscheiden nach der Art der im $(i + 1)$ -ten Schritt verwendeten Regel
 - 1. Fall: $X_i \rightarrow Y_i Z_i$
 - * Dann gelten:
 - $z_{i+1} = z_i$
 - $X_{i+1} = Y_i$ und
 - $\alpha_{i+1} = Z_i \alpha_i$
 - * Nach Definition von \mathcal{A} gilt dann:

$$(q, z_i, X_i \alpha_i) \vdash (q, z_i, Y_i Z_i \alpha_i) = (q, z_{i+1}, X_{i+1} \alpha_{i+1})$$
- ➡ Induktionsbehauptung

Beweisdetails (Forts.)

- 2. Fall: $X_i \rightarrow \sigma_i$
 - Da wir eine Linksableitung einer CNF-Grammatik haben, ist das erste Symbol von α_i eine Variable, also X_{i+1} in der von uns gewählten Notation
 - Es gilt: $\alpha_i = X_{i+1} \alpha_{i+1}$
 - Es folgt:

$$(q, z_i, X_i \alpha_i) \vdash (q, z_i, \sigma_i X_{i+1} \alpha_{i+1})$$
 - Da die Ableitung insgesamt w erzeugt, ist σ_i das erste Zeichen von z_i und es gilt $z_i = \sigma_i z_{i+1}$
 - Dann folgt: $(q, z_i, \sigma_i X_{i+1} \alpha_{i+1}) \vdash (q, z_{i+1}, X_{i+1} \alpha_{i+1})$
- ➡ Induktionsbehauptung
- Der 2. Fall findet insbesondere im letzten Ableitungsschritt Anwendung und führt damit zur Konfiguration (q, ϵ, ϵ)
- ➡ $w \in L(\mathcal{A})$

Grammatik → Kellerautomat: Beweisdetails (3/3)

Beweis (Forts.)

- Zu zeigen: $L(\mathcal{A}) \subseteq L(G)$
- Wir beweisen durch Induktion nach der Berechnungslänge n :
 - Für alle $X \in V$ und $w \in \Sigma^*$:
 wenn $(q, w, X) \vdash^n (q, \epsilon, \epsilon)$,
 dann $X \Rightarrow^* w$
- $n = 1$:
 - $(q, w, X) \vdash (q, \epsilon, \epsilon)$
 - ➔ $X = S$ und $w = \epsilon$ und es gibt die Regel $S \rightarrow \epsilon$ in G
 ☞ wegen CNF
 - ➔ $X \Rightarrow^* w$
- $n = 2$:
 - $(q, w, X) \vdash (q, w, \sigma)$
 $\vdash (q, \epsilon, \epsilon)$
 - ➔ $w = \sigma$ und es gibt die Regel $X \rightarrow \sigma$ in G
 - ➔ $X \Rightarrow^* w$

Beweis (Forts.)

- $n + 1$: $(q, w, X) \vdash^{n+1} (q, \epsilon, \epsilon)$
 - Sei $(q, w, X) \vdash (q, w, Z_1 Z_2)$ der erste Schritt der Berechnung, für gewisse $Z_1, Z_2 \in V$
 - ➔ $(q, w, Z_1 Z_2) \vdash^n (q, \epsilon, \epsilon)$ und in dieser Berechnung werden Z_1 und Z_2 nach und nach vom Keller entfernt
 - ➔ Es gibt eine Zerlegung $w = u_1 u_2$, so dass
 $(q, u_1 u_2, Z_1 Z_2) \vdash_{(Z_2)}^{m_1} (q, u_2, Z_2)$
 $\vdash^{m_2} (q, \epsilon, \epsilon)$
 - Backstage-Lemma:
 - * $(q, u_1, Z_1) \vdash^{m_1} (q, \epsilon, \epsilon)$
 - Induktion: $Z_1 \Rightarrow^* u_1$ und $Z_2 \Rightarrow^* u_2$
 ☞ $m_1, m_2 \leq n$
 - ➔ $X \Rightarrow Z_1 Z_2 \Rightarrow^* u_1 u_2 = w$
- Die Anwendung auf $X = S$ liefert dann $L(\mathcal{A}) \subseteq L(G)$

Kellerautomat → Grammatik: Beweisdetails (1/2)

Beweis von Satz 9.3 (Forts.)

- Wir zeigen zuerst durch Induktion nach n :
 - falls $(p, w, \tau) \vdash_{\mathcal{A}}^n (p', \epsilon, \epsilon)$
 - so gilt: $X_{p, \tau, p'} \Rightarrow^* w$
- $n = 1$:
 - Dann gilt:
 - * $w = \epsilon$ und $(p, \epsilon, \tau, p', \epsilon) \in \delta$ oder
 - * $w = \sigma$ und $(p, \sigma, \tau, p', \epsilon) \in \delta$
 - Im ersten Fall enthält P die Regel $X_{p, \tau, p'} \rightarrow \epsilon$ im zweiten Fall $X_{p, \tau, p'} \rightarrow \sigma$
- $n > 1$: Wir betrachten zuerst den Fall, dass der erste Schritt der Berechnung ein Zeichen σ liest, also:

$$(p, w, \tau) \vdash (q, u, \tau_1 \tau_2)$$

mit $w = \sigma u$
- Dann gilt: $(p, \sigma, \tau, q, \tau_1 \tau_2) \in \delta$
- Nach Konstruktion von G gibt es also für alle p_1 und p' eine Regel

$$X_{p, \tau, p'} \rightarrow \sigma X_{q, \tau_1, p_1} X_{p_1, \tau_2, p'}$$

Beweis (Forts.)

- Sei p_1 der Zustand nach dem Entfernen von τ_1 vom Keller in der Berechnung

$$(q, u, \tau_1 \tau_2) \vdash^{n-1} (p', \epsilon, \epsilon)$$
- Seien $u = u_1 u_2$, so dass u_1 bis zum Entfernen von τ_1 gelesen wird
- ➔ Es gibt i, j mit $i + j = n - 1$, so dass:

$$(p, w, \tau) \vdash (q, u_1 u_2, \tau_1 \tau_2) \vdash_{(\tau_2)}^i (p_1, u_2, \tau_2) \vdash^j (p', \epsilon, \epsilon)$$
- Mit Lemma 10.3 gelten also:
 - $(q, u_1, \tau_1) \vdash^i (p_1, \epsilon, \epsilon)$ und
 - $(p_1, u_2, \tau_2) \vdash^j (p', \epsilon, \epsilon)$
- Nach Induktion folgt:
 - $X_{q, \tau_1, p_1} \Rightarrow^* u_1$
 - $X_{p_1, \tau_2, p'} \Rightarrow^* u_2$
- ➔
$$X_{p, \tau, p'} \Rightarrow \sigma X_{q, \tau_1, p_1} X_{p_1, \tau_2, p'} \Rightarrow^* \sigma u_1 u_2 = w$$
- Der Fall, dass der erste Schritt ein ϵ -Übergang ist, lässt sich analog beweisen

Kellerautomat → Grammatik: Beweisdetails (2/2)

Beweis von Satz 9.3 (Forts.)

- Wir zeigen jetzt durch Induktion nach der Ableitungslänge n :
 - falls $X_{p,\tau,p'} \Rightarrow^n w$
 - so gilt: $(p, w, \tau) \vdash_{\mathcal{A}}^* (p', \epsilon, \epsilon)$
- $n = 1$: Die einzigen Regeln von G , die keine Variablen erzeugen, sind von der Form
 - $X_{p,\tau,p'} \rightarrow \alpha$, mit
 $(p, \alpha, \tau, p', \epsilon) \in \delta$
 mit $\alpha \in \Sigma \cup \{\epsilon\}$
- Die Behauptung folgt direkt
- $n > 1$: In diesem Fall gibt es zwei verschiedene Typen des ersten Ableitungsschrittes

Beweis (Forts.)

- Wir betrachten den ersten Fall:

$$X_{p,\tau,p'} \Rightarrow \alpha X_{q,\tau_1,p_1} X_{p_1,\tau_2,p'}$$
- Es gilt dann:

$$\alpha X_{q,\tau_1,p_1} X_{p_1,\tau_2,p'} \Rightarrow^{n-1} w$$
- Also gibt es u_1, u_2 mit $w = \alpha u_1 u_2$ und i, j mit $i + j = n - 1$, so dass gilt:
 - $X_{q,\tau_1,p_1} \Rightarrow^i u_1$
 - $X_{p_1,\tau_2,p'} \Rightarrow^j u_2$
- Nach Induktion folgt:
 - $(q, u_1, \tau_1) \vdash^* (p_1, \epsilon, \epsilon)$
 - $(p_1, u_2, \tau_2) \vdash^* (p', \epsilon, \epsilon)$
- Mit Lemma 10.3 gelten dann auch:
 - $(q, u_1 u_2, \tau_1 \tau_2) \vdash^* (p_1, u_2, \tau_2)$
 - $(p_1, u_2, \tau_2) \vdash^* (p', \epsilon, \epsilon)$
- Zusammen ergibt sich

$$(p, w, \tau) \vdash (q, u_1 u_2, \tau_1 \tau_2) \vdash_{(\tau_2)}^* (p', \epsilon, \epsilon)$$
- Die anderen Fällen sind analog