Automatentheorie und formale Sprachen
Chomsky-Hierarchie
Kellerautomaten

Dozentin: Wiebke Petersen

8.7.2009

Formale Grammatik

Definition

Eine formale Grammatik ist ein 4-Tupel $G = (N, T, S, P)$ aus

- *einem Alphabet von Terminalsymbolen T (häufig auch Σ)*
- *einem Alphabet von Nichtterminalsymbolen N mit $N \cap T = \emptyset$*
- *einem Startsymbol $S \in N$*
- *einer Menge von Regeln/Produktionen*
  $P \subseteq \{\langle\alpha, \beta\rangle \mid \alpha, \beta \in (N \cup T)^* \text{ und } \alpha \notin T^*\}$.

Für eine Regel $\langle\alpha, \beta\rangle$ schreiben wir auch $\alpha \rightarrow \beta$.
Formale Grammatiken werden auch Typ0- oder allgemeine Regelgrammatiken genannt.

Chomsky-Hierarchie

- Wenn man die Form der Regeln einschränkt erhält man Teilmengen der Menge aller durch eine Grammatik erzeugten Sprachen.

Chomsky-Hierarchie

- Wenn man die Form der Regeln einschränkt erhält man Teilmengen der Menge aller durch eine Grammatik erzeugten Sprachen.
- Die Chomsky-Hierarchie ist eine Hierarchie über die Regelbedingungen (den verschiedenen Sprachklassen entsprechen Einschränkungen über die rechten und linken Regelseiten).

Chomsky-Hierarchie

- Wenn man die Form der Regeln einschränkt erhält man Teilmengen der Menge aller durch eine Grammatik erzeugten Sprachen.
- Die Chomsky-Hierarchie ist eine Hierarchie über die Regelbedingungen (den verschiedenen Sprachklassen entsprechen Einschränkungen über die rechten und linken Regelseiten).
- Die Chomsky Hierarchie reflektiert eine spezielle Form der Komplexität, andere Kriterien sind denkbar und führen zu anderen Hierarchien.

Chomsky-Hierarchie

- Wenn man die Form der Regeln einschränkt erhält man Teilmengen der Menge aller durch eine Grammatik erzeugten Sprachen.
- Die Chomsky-Hierarchie ist eine Hierarchie über die Regelbedingungen (den verschiedenen Sprachklassen entsprechen Einschränkungen über die rechten und linken Regelseiten).
- Die Chomsky Hierarchie reflektiert eine spezielle Form der Komplexität, andere Kriterien sind denkbar und führen zu anderen Hierarchien.
- Die Sprachklassen der Chomsky Hierarchie sind in der Informatik intensiv untersucht worden (Berechnungskomplexität, effektive Parser).

Chomsky-Hierarchie

- Wenn man die Form der Regeln einschränkt erhält man Teilmengen der Menge aller durch eine Grammatik erzeugten Sprachen.
- Die Chomsky-Hierarchie ist eine Hierarchie über die Regelbedingungen (den verschiedenen Sprachklassen entsprechen Einschränkungen über die rechten und linken Regelseiten).
- Die Chomsky Hierarchie reflektiert eine spezielle Form der Komplexität, andere Kriterien sind denkbar und führen zu anderen Hierarchien.
- Die Sprachklassen der Chomsky Hierarchie sind in der Informatik intensiv untersucht worden (Berechnungskomplexität, effektive Parser).
- Für Linguisten ist die Chomsky Hierarchie besonders interessant, da sie die Form der Regeln zentral stellt, und somit Aussagen über Grammatikformalismen zuläßt.

Noam Chomsky

Noam Chomsky
($*$ 7.12.1928, Philadelphia)
Noam Chomsky, *Three Models for the Description of Language*,
IRE Transactions on Information Theory (1956).

Chomsky-hierarchy (details)

A grammar (N, T, S, P) is a

(right-linear) regular grammar (REG): iff every production is of the form
$A \longrightarrow bB$ or $A \longrightarrow b$ with $A, B \in N$ and $b \in T$ (or $S \longrightarrow \epsilon$, , in which case S does not occur on any right-hand side of a production).

context-free grammar (CFG): iff every production is of the form
$A \longrightarrow \beta$ with $A \in N$ and $\beta \in (N \cup T)^*$.

Chomsky-hierarchy (details)

A grammar (N, T, S, P) is a

(right-linear) regular grammar (REG): iff every production is of the form
$A \rightarrow bB$ or $A \rightarrow b$ with $A, B \in N$ and $b \in T$ (or $S \rightarrow \epsilon$, , in which case S does not occur on any right-hand side of a production).

context-free grammar (CFG): iff every production is of the form
$A \rightarrow \beta$ with $A \in N$ and $\beta \in (N \cup T)^*$.

context-sensitive grammar (CS): iff every production is of the form
$\gamma A \delta \rightarrow \gamma \beta \delta$ with $\gamma, \delta, \beta \in (N \cup T)^*, A \in N$ and $\beta \neq \epsilon$; or of the form $S \rightarrow \epsilon$, in which case S does not occur on any right-hand side of a production.

recursively enumerable grammar (RE): if it is an arbitrary formal grammar.

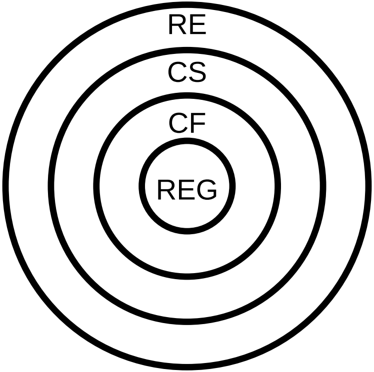
Chomsky-Hierarchie (grober Überblick)

reguläre Sprachen (regular languages)	Typ 3, REG	$A \rightarrow bA$ $A \rightarrow a$	a^*b^*
kontextfreie Sprachen (context-free languages)	Typ 2, CF	$A \rightarrow \beta$	a^nb^n, w^Rw
kontextsensitive Sprachen context-sensitive languages	Typ 1, CS	$\alpha A\nu \rightarrow \alpha\beta\nu$	$a^nb^nc^n$, ww, $a^nb^mc^nd^m$
allgemeine Regelsprachen recursively enumerable languages	Typ 0, RE	$\alpha \rightarrow \beta$	

$a \in T$, $A \in N$, $\alpha, \beta, \ldots \in (N \cup T)^*$, S Startsymbol

Main theorem

REG $\subset$ CF $\subset$ CS $\subset$ RE



Abschlusseigenschaften formaler Sprachen

	Typ3	Typ2	Typ1	Typ0
Vereinigung	+	+	+	+
Schnittmenge	+	-	+	+
Komplement	+	-	+	-
Konkatenation	+	+	+	+
Stern von Kleene	+	+	+	+
Schnitt mit regulärer Sprache	+	+	+	+

Abschlusseigenschaften kontextfreier Sprachen

	Typ3	Typ2	Typ1	Typ0
Vereinigung	+	+	+	+
Schnittmenge	+	-	+	+
Komplement	+	-	+	-
Konkatenation	+	+	+	+
Stern von Kleene	+	+	+	+
Schnitt mit regulärer Sprache	+	+	+	+

Abschlusseigenschaften kontextfreier Sprachen

	Typ3	Typ2	Typ1	Typ0
Vereinigung	+	+	+	+
Schnittmenge	+	-	+	+
Komplement	+	-	+	-
Konkatenation	+	+	+	+
Stern von Kleene	+	+	+	+
Schnitt mit regulärer Sprache	+	+	+	+

Vereinigung: $G = (N_1 \uplus N_2 \cup \{S\}, T_1 \cup T_2, S, P)$ mit
$P = P_1 \cup_{\uplus} P_2 \cup \{S \rightarrow S_1, S \rightarrow S_2\}$

Abschlusseigenschaften kontextfreier Sprachen

	Typ3	Typ2	Typ1	Typ0
Vereinigung	+	+	+	+
Schnittmenge	+	-	+	+
Komplement	+	-	+	-
Konkatenation	+	+	+	+
Stern von Kleene	+	+	+	+
Schnitt mit regulärer Sprache	+	+	+	+

Vereinigung: $G = (N_1 \uplus N_2 \cup \{S\}, T_1 \cup T_2, S, P)$ mit
$P = P_1 \cup_{\uplus} P_2 \cup \{S \rightarrow S_1, S \rightarrow S_2\}$

Schnittmenge: $L_1 = \{a^n b^n a^k\}$, $L_2 = \{a^n b^k a^k\}$, aber $L_1 \cap L_2 = \{a^n b^n a^n\}$

Abschlusseigenschaften kontextfreier Sprachen

	Typ3	Typ2	Typ1	Typ0
Vereinigung	+	+	+	+
Schnittmenge	+	-	+	+
Komplement	+	-	+	-
Konkatenation	+	+	+	+
Stern von Kleene	+	+	+	+
Schnitt mit regulärer Sprache	+	+	+	+

Vereinigung: $G = (N_1 \uplus N_2 \cup \{S\}, T_1 \cup T_2, S, P)$ mit
$P = P_1 \cup_{\uplus} P_2 \cup \{S \rightarrow S_1, S \rightarrow S_2\}$

Schnittmenge: $L_1 = \{a^n b^n a^k\}$, $L_2 = \{a^n b^k a^k\}$, aber $L_1 \cap L_2 = \{a^n b^n a^n\}$

Komplement: *de Morgan*

Abschlusseigenschaften kontextfreier Sprachen

	Typ3	Typ2	Typ1	Typ0
Vereinigung	+	+	+	+
Schnittmenge	+	-	+	+
Komplement	+	-	+	-
Konkatenation	+	+	+	+
Stern von Kleene	+	+	+	+
Schnitt mit regulärer Sprache	+	+	+	+

Vereinigung: $G = (N_1 \uplus N_2 \cup \{S\}, T_1 \cup T_2, S, P)$ mit
$P = P_1 \cup_{\uplus} P_2 \cup \{S \to S_1, S \to S_2\}$

Schnittmenge: $L_1 = \{a^n b^n a^k\}$, $L_2 = \{a^n b^k a^k\}$, aber $L_1 \cap L_2 = \{a^n b^n a^n\}$

Komplement: *de Morgan*

Konkatenation: $G = (N_1 \uplus N_2 \cup \{S\}, T_1 \cup T_2, S, P)$ with
$P = P_1 \cup_{\uplus} P_2 \cup \{S \to S_1 S_2\}$

Abschlusseigenschaften kontextfreier Sprachen

	Typ3	Typ2	Typ1	Typ0
Vereinigung	+	+	+	+
Schnittmenge	+	-	+	+
Komplement	+	-	+	-
Konkatenation	+	+	+	+
Stern von Kleene	+	+	+	+
Schnitt mit regulärer Sprache	+	+	+	+

Vereinigung: $G = (N_1 \uplus N_2 \cup \{S\}, T_1 \cup T_2, S, P)$ mit
$P = P_1 \cup_{\uplus} P_2 \cup \{S \rightarrow S_1, S \rightarrow S_2\}$

Schnittmenge: $L_1 = \{a^n b^n a^k\}$, $L_2 = \{a^n b^k a^k\}$, aber $L_1 \cap L_2 = \{a^n b^n a^n\}$

Komplement: *de Morgan*

Konkatenation: $G = (N_1 \uplus N_2 \cup \{S\}, T_1 \cup T_2, S, P)$ with
$P = P_1 \cup_{\uplus} P_2 \cup \{S \rightarrow S_1 S_2\}$

Stern von Kleene: $G = (N_1 \cup \{S\}, T_1, S, P)$ with $P = P_1 \cup \{S \rightarrow S_1 S, S \rightarrow \epsilon\}$

13 Kellerautomaten (Pushdown-Automaten)

Ziel:

Einführung eines Automatenmodells, mit dem genau die kontextfreien Sprachen akzeptiert werden können.

Wir benötigen eine echte Erweiterung des Modells des endlichen Automaten.

Ansatz: Hinzunahme eines unbeschränkten Speichers.

Speicherinhalt ist jeweils ein Wort.

Zugriff auf Speicherinhalt über die Spitze: Der Wort**anfang** kann jeweils gelesen und modifiziert werden.

Wir sprechen von **Kellerspeicher** oder **Stack** oder **Pushdown Stack**.

Informelles Beispiel

Wir betrachten die Sprache $\{a^i b^i \mid i > 0\}$.

Das Akzeptieren eines Eingabewortes geschieht wie folgt:

1. Aufbau des Kellers: für jedes gelesene a lege ein Symbol auf dem Keller ab (wir nehmen das Symbol Z)
2. Abbau des Kellers: für jedes gelesene b nehme ein Symbol Z vom Keller herunter
3. durch zwei Kontrollzustände sorge dafür, dass Aufbau und Abbau nur in dieser Reihenfolge möglich sind (Aufbau mit q_0, Abbau mit q_1, keine Rückkehr nach q_0)
4. Akzeptiere, wenn am Ende des Eingabewortes der Kellerboden erreicht ist.

Der Keller realisiert in diesem Fall eine Zählervariable.

Kellerautomaten

Kellerautomaten entstehen aus endlichen Automaten durch

- Hinzunahme eines Kelleralphabets
- Erweiterung der Transitionen (es muss das Lesen und Ersetzen der Kellerspitze realisiert werden)

Definition

Ein *Kellerautomat* *ist ein 6-Tupel* $\langle \Phi, \Sigma, \Gamma, \Delta, q_0, F \rangle$ *bestehend aus:*

1. *einem* *Zustandsalphabet* Φ
2. *einem* *Eingabealphabet* Σ
3. *einem* *Kelleralphabet* Γ
4. *einer* *Übergangsrelation / Transitionsrelation*
   $\Delta \subseteq \Phi \times \Sigma \cup \{\epsilon\} \times \Gamma \cup \{\epsilon\} \times \Gamma^* \times \Phi$
5. *einem* *Startzustand* $q_0 \in \Phi$ *und*
6. *einer Menge von* *Endzuständen* $F \subseteq \Phi$.

Transition

$$(p, a, Z, v, q)$$

Transition

(p, a, Z, v, q)

aktueller Zustand

Transition

$$(\mathbf{p}, \mathbf{a}, \mathbf{Z}, \mathbf{v}, \mathbf{q})$$

aktueller Zustand

zu lesendes Eingabesymbol

Transition

oberstes Symbol auf dem Stack
(wird entfernt)
- pop up -

(p, a, Z, v, q)

aktueller Zustand

zu lesendes Eingabesymbol

Transition

oberstes Symbol auf dem Stack
(wird entfernt)
- pop up -

$$(p, a, Z, v, q)$$

aktueller Zustand

zu lesendes Eingabesymbol

Wort, das auf den Stack gelegt wird
- push down -

Transition

oberstes Symbol auf dem Stack
(wird entfernt)
- pop up -

neuer Zustand

(p, a, Z, v, q)

aktueller Zustand

zu lesendes Eingabesymbol

Wort, das auf den Stack gelegt wird
- push down -

Transition

oberstes Symbol auf dem Stack
(wird entfernt)
- pop up -

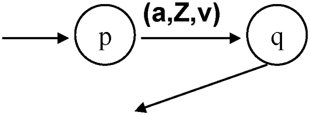
neuer Zustand

(p, a, Z, v, q)

aktueller Zustand

zu lesendes Eingabesymbol

Wort, das auf den Stack gelegt wird
- push down -



Transition

oberstes Symbol auf dem Stack
(wird entfernt)
- pop up -

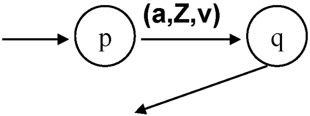
neuer Zustand

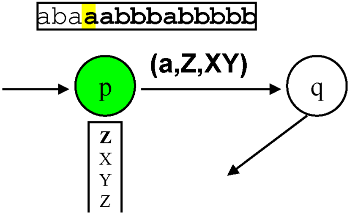
(p, a, Z, v, q)

aktueller Zustand

zu lesendes Eingabesymbol

Wort, das auf den Stack gelegt wird
- push down -





Transition

oberstes Symbol auf dem Stack
(wird entfernt)
- pop up -

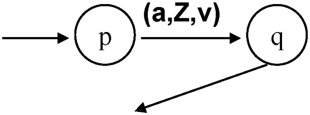
neuer Zustand

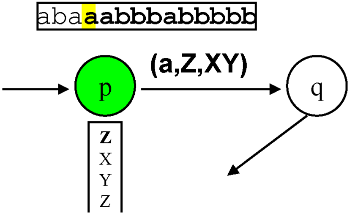
(p, a, Z, v, q)

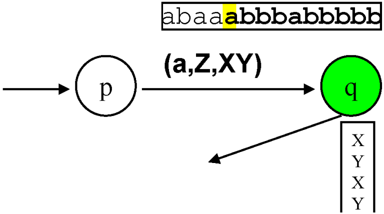
aktueller Zustand

zu lesendes Eingabesymbol

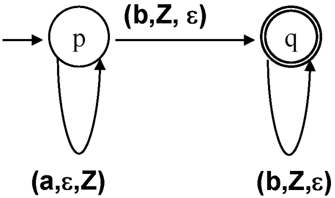
Wort, das auf den Stack gelegt wird
- push down -





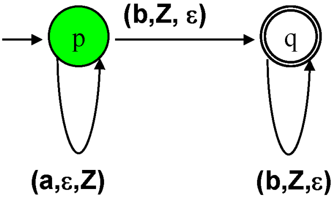


Beispiel eines Kellerautomaten



dieser Kellerautomat akzeptiert
die Sprache $a^n b^n$

Arbeitsweise eines Kellerautomaten

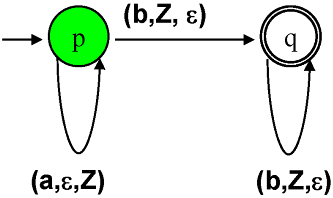


(noch) zu lesendes Wort:

a a a b b b

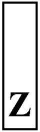
aktueller Stack:

Arbeitsweise eines Kellerautomaten

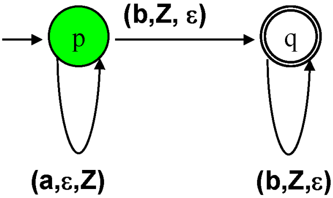


(noch) zu lesendes Wort:

aktueller Stack:



Arbeitsweise eines Kellerautomaten



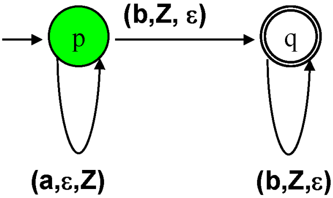
(noch) zu lesendes Wort:

a b b b

aktueller Stack:

Z
Z

Arbeitsweise eines Kellerautomaten



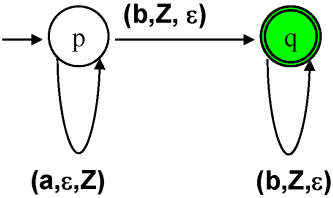
(noch) zu lesendes Wort:

b b b

aktueller Stack:

Z
Z
Z

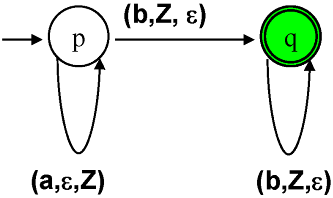
Arbeitsweise eines Kellerautomaten



(noch) zu lesendes Wort:

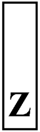
aktueller Stack:

Arbeitsweise eines Kellerautomaten

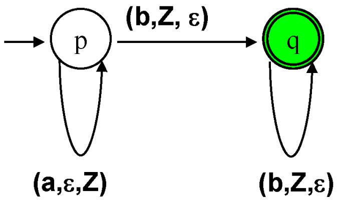


(noch) zu lesendes Wort:

aktueller Stack:

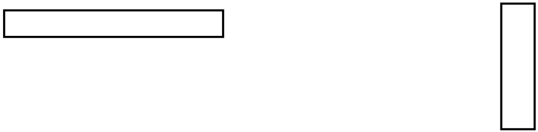


Arbeitsweise eines Kellerautomaten

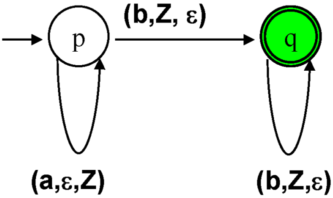


(noch) zu lesendes Wort:

aktueller Stack:



Arbeitsweise eines Kellerautomaten

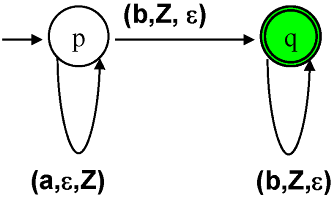


(noch) zu lesendes Wort:

Das Wort ist abgearbeitet!

aktueller Stack:

Arbeitsweise eines Kellerautomaten



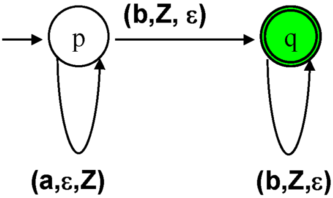
(noch) zu lesendes Wort:

Das Wort ist abgearbeitet!

aktueller Stack:

Der Stack ist leer!

Arbeitsweise eines Kellerautomaten

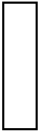


Der Automat befindet sich in einem Endzustand!

(noch) zu lesendes Wort:

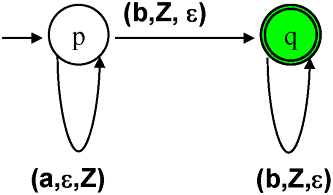
Das Wort ist abgearbeitet!

aktueller Stack:



Der Stack ist leer!

Arbeitsweise eines Kellerautomaten



Der Automat befindet
sich in einem Endzustand!

Darum akzeptiert der
Kellerautomat das Wort!

(noch) zu lesendes Wort:

Das Wort ist abgearbeitet!

aktueller Stack:

Der Stack ist leer!

Von Grammatiken zu Kellerautomaten

Theorem

Zu einer kontextfreien Grammatik G kann man einen Kellerautomaten A konstruieren, der die von G generierte Sprache akzeptiert.

Von Grammatiken zu Kellerautomaten

Theorem

Zu einer kontextfreien Grammatik G kann man einen Kellerautomaten A konstruieren, der die von G generierte Sprache akzeptiert.

Konstruktionsvorschrift:

Sei $G = (N, T, S, P)$ die gegebene Grammatik. Konstruiere den Automaten $A = \langle \Phi, \Sigma, \Gamma, \Delta, q_0, F \rangle$ wie folgt:

- $\Phi = \{q_0, q_1\}, \quad \Sigma = T, \quad \Gamma = T \uplus N, \quad F = \{q_1\},$

Von Grammatiken zu Kellerautomaten

Theorem

Zu einer kontextfreien Grammatik G kann man einen Kellerautomaten A konstruieren, der die von G generierte Sprache akzeptiert.

Konstruktionsvorschrift:

Sei $G = (N, T, S, P)$ die gegebene Grammatik. Konstruiere den Automaten $A = \langle \Phi, \Sigma, \Gamma, \Delta, q_0, F \rangle$ wie folgt:

- $\Phi = \{q_0, q_1\}, \quad \Sigma = T, \quad \Gamma = T \uplus N, \quad F = \{q_1\},$
- Δ enthält die folgenden Transitionen:
  - $(q_0, \epsilon, \epsilon, S, q_1)$

Von Grammatiken zu Kellerautomaten

Theorem

Zu einer kontextfreien Grammatik G kann man einen Kellerautomaten A konstruieren, der die von G generierte Sprache akzeptiert.

Konstruktionsvorschrift:

Sei $G = (N, T, S, P)$ die gegebene Grammatik. Konstruiere den Automaten $A = \langle \Phi, \Sigma, \Gamma, \Delta, q_0, F \rangle$ wie folgt:

- $\Phi = \{q_0, q_1\}$, $\quad \Sigma = T$, $\quad \Gamma = T \uplus N$, $\quad F = \{q_1\}$,
- Δ enthält die folgenden Transitionen:
  - $(q_0, \epsilon, \epsilon, S, q_1)$
  - für jede Regel $B \to \beta$ der Grammatik eine Transition $(q_1, \epsilon, B, \beta, q_1)$

Von Grammatiken zu Kellerautomaten

Theorem

Zu einer kontextfreien Grammatik G kann man einen Kellerautomaten A konstruieren, der die von G generierte Sprache akzeptiert.

Konstruktionsvorschrift:
Sei $G = (N, T, S, P)$ die gegebene Grammatik. Konstruiere den Automaten $A = \langle \Phi, \Sigma, \Gamma, \Delta, q_0, F \rangle$ wie folgt:

- $\Phi = \{q_0, q_1\}, \quad \Sigma = T, \quad \Gamma = T \uplus N, \quad F = \{q_1\},$
- Δ enthält die folgenden Transitionen:
  - $(q_0, \epsilon, \epsilon, S, q_1)$
  - für jede Regel $B \rightarrow \beta$ der Grammatik eine Transition $(q_1, \epsilon, B, \beta, q_1)$
  - für jedes Symbol $a \in T$ eine Transition $(q_1, a, a, \epsilon, q_1)$.

Wirkungsweise der Konstruktion

- Für das Akzeptieren einer kontextfreien Sprache genügt ein Kellerautomat mit nur zwei Zuständen, wobei die einzige Aufgabe des Startzustands darin besteht, das Startsymbol S der Grammatik in den Keller zu legen.

Wirkungsweise der Konstruktion

- Für das Akzeptieren einer kontextfreien Sprache genügt ein Kellerautomat mit nur zwei Zuständen, wobei die einzige Aufgabe des Startzustands darin besteht, das Startsymbol S der Grammatik in den Keller zu legen.
- Während der eigentlichen Rechnung befindet sich der Automat permanent in dem selben Zustand, die Rechnung findet nur in dem Keller statt.

Wirkungsweise der Konstruktion

- Für das Akzeptieren einer kontextfreien Sprache genügt ein Kellerautomat mit nur zwei Zuständen, wobei die einzige Aufgabe des Startzustands darin besteht, das Startsymbol S der Grammatik in den Keller zu legen.
- Während der eigentlichen Rechnung befindet sich der Automat permanent in dem selben Zustand, die Rechnung findet nur in dem Keller statt.
- Der konstruierte Automat vollzieht zwei verschiedene Arbeitsschritte:
  - Nicht-Leseschritt mit Bezug auf Grammatikregel $B \to \beta$:
    Ersetze die Kellerspitze B mit β
  - Leseschritte:
    Lies ein $a \in T$ der Eingabekette und entferne a von der Kellerspitze.

Chomsky-Hierarchie & Automaten

Sprache	*Automat*	*Grammatik*	*Erkennung*	*Abhängigkeit*
rekursiv aufzählbar	Turing Maschine	unbeschränkt Baa $\rightarrow \varepsilon$	unentscheidbar	beliebig
kontext-sensitiv	linear gebunden	kontext-sensitiv $\gamma A \delta \rightarrow \gamma \beta \delta$	NP-vollständig	überkreuzt
kontext-frei	Kellerautomat (Stapel)	kontextfrei C $\rightarrow$ bABa	polynomiell	eingebettet
regulär	endlicher Automat	regulär A $\rightarrow$ bA	linear	strikt lokal

Hausaufgabe

1. Gegeben seien die beiden kontextfreien Grammatiken
   $G_1 = (\{S\}, \{a, b\}, S, \{S \rightarrow aSb, S \rightarrow \epsilon\})$ und
   $G_2 = (\{S\}, \{a, b\}, S, \{S \rightarrow aSb, S \rightarrow bSa, S \rightarrow \epsilon\})$.
   Geben sie eine kontextfreie Grammatik für die folgenden Sprachen an:
   - $L(G_1) \cup L(G_2)$
   - $L(G_1) \circ L(G_2)$
   - $L(G_1)^*$.
2. Geben sie zur Sprache $L(G_1) \circ L(G_2)$ eine Grammatik in Chomsky-Normalform an.
3. Konstruieren sie einen Kellerautomaten zur Grammatik G_2. Verwenden sie das Verfahren, das auf den vorangegangenen Folien beschrieben wurde.
4. Zeigen sie, wie der von ihnen konstruierte Kellerautomat das Wort *aabb* abarbeitet, indem sie in einer Tabelle für jeden Schritt die folgenden Dinge erfassen: (1) gelesene Kette, (2) aktueller Zustand, (3) aktueller Stack, (4) noch zu lesende Kette, (5) anzuwendende Regel.