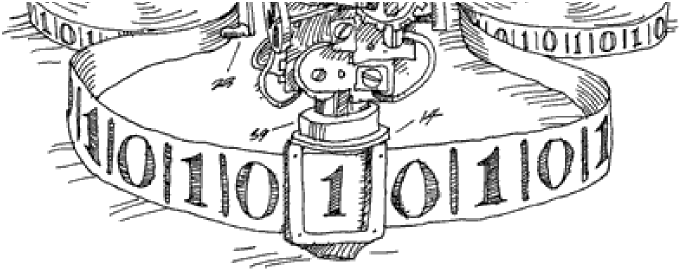
Theoretische Grundlagen der Informatik

Tutorium 12

Institut für Theoretische Informatik



Pumping-Lemma für kontextfreie Sprachen

Karlsruher Institut für Technologie

Lemma

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ so als

$$z = uvwxy$$

schreiben lässt, dass

- $|vx| \geq 1$,
- $|vwx| \leq n$ und
- für alle $i \geq 0$ das Wort $uv^i wx^i y \in L$ ist.

Ogdens Lemma für kontextfreie Sprachen

Karlsruher Institut für Technologie

Lemma

Für jede kontextfreie Sprache L gibt es eine Konstante $n \in \mathbb{N}$, so dass für jedes Wort $z \in L$ mit $|z| \geq n$ gilt:
Wenn wir in z mindestens n Buchstaben markieren, so lässt sich z so als $z = uvwxy$ schreiben, dass

- von den mindestens n markierten Buchstaben
  - mindestens einer zu vx gehört und
  - höchstens n zu vwx gehören und
- für alle $i \geq 0$ das Wort $uv^iwx^iy \in L$ ist.

Aufgabe

1. Zeige, dass die Sprache

$$L = \{a^j b^k c^l d^m \mid j = 0 \vee k = l = m\}$$

   die notwendige Bedingung des Pumping-Lemmas für Kontextfreiheit erfüllt.
2. Zeige, dass L nicht kontextfrei ist.

Lösung zu 1.

Setze $n := 1$, denn für jedes Wort $z \in L$ mit $|w| \geq 1$ gilt:

Fall $j = 0$: Dann ist $z = b^k c^l d^m$ mit $k, l, m \in \mathbb{N}_0$. Zerlege $z = uvwxy$ mit $u, x, y = \varepsilon, |v| = 1$, denn jedes $z' = uv^i xy^i z = v^i w$ ist in L, da keine a darin vorkommen ($j = 0$).

Fall $j \neq 0$: Dann ist $z = a^j b^k c^k d^k$. Zerlege $z = uvwxy$ mit $u, w, x = \varepsilon, v = a \Rightarrow y = a^{j-1} b^k c^k d^k$. Dann ist für alle $i \in \mathbb{N}_0$ $uv^i wx^i y = a^{j+i} b^k c^k d^k \in L$.

Lösung zu 2.

Sei $L = \{a^i b^j c^k d^l \mid i = 0 \lor j = k = l\}$. Wir zeigen mit Ogdens Lemma, dass L nicht kontextfrei ist.

Annahme: L sei kontextfrei. Für gegebenes n müsste Ogdens Lemma also für jedes Wort mit n markierten Buchstaben erfüllt sein.

Betrachte aber $z = ab^n c^n d^n \in L$ und markiere b^n.

Zu vx muss also immer ein b gehören und in vwx dürfen höchsten n markierte Buchstaben vorkommen. Offensichtlich dürfen in v und x jeweils nur eine Art von Symbol vorkommen. Da aber in v oder x mindestens ein b vorkommen muss, und die Anzahl der b und c und d gleich bleiben muss, ist das Lemma nicht erfüllt. Daher kann L nicht kontextfrei sein.

Berechnungsmodelle für Grammatiken

- **Typ-0-Grammatiken** / **semientscheidbar:** Turingmaschinen
- **Typ-1-Grammatiken** / **kontextsensitiv:** linear beschränkte Turingmaschinen (LBA)
- **Typ-2-Grammatiken** / **kontextfrei:** ?
- **Typ-3-Grammatiken** / **regulär:** Endliche Automaten

Berechnungsmodelle für Grammatiken

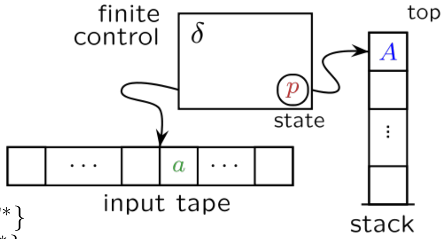
Karlsruher Institut für Technologie

- **Typ-0-Grammatiken** / **semientscheidbar:** Turingmaschinen
- **Typ-1-Grammatiken** / **kontextsensitiv:** linear beschränkte Turingmaschinen (LBA)
- **Typ-2-Grammatiken** / **kontextfrei:** Kellerautomaten!
- **Typ-3-Grammatiken** / **regulär:** Endliche Automaten

Definition

Ein (nichtdeterministischer) **Kellerautomat** (NPDA bzw PDA, Pushdown Automaton) besteht aus $(Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$, wobei

- Q endliche Zustandsmenge
- Σ endliches Eingabealphabet
- Γ endliches Stack-Alphabet
- $q_0 \in Q$ Anfangszustand
- $Z_0 \in \Gamma$ Initialisierung des Stacks
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$
  - $\delta(q, a, Z) \subseteq \{(q, \gamma) : q \in Q, \gamma \in \Gamma^*\}$
  - $\delta(q, \varepsilon, Z) \subseteq \{(q, \gamma) : q \in Q, \gamma \in \Gamma^*\}$
- $F \subseteq Q$ Menge der akzeptierenden Endzustände, $F = \emptyset$ ist möglich.



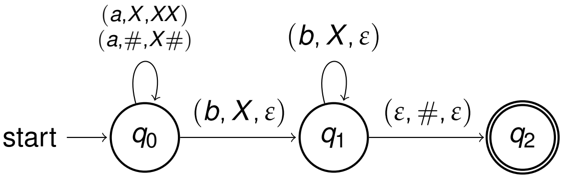
Zu Kellerautomaten

- Akzeptieren nach Eingabeende, wenn
  - der Stack leer ist *oder*
  - der Automat in einen akzeptierenden Zustand kommt.
- Sind im Allgemeinen nichtdeterministisch
- Man kann Endzustände auch aus der Definition weglassen und alternativ verlangen, dass der Automat genau bei leerem Keller akzeptiert.
- Man kann sogar alle Zustände bis auf einen weglassen und alles in die Kellerbelegung kodieren

Beispiel

$M = (Q, \Sigma, \Gamma, q_0, Z_0, \delta, F)$

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{a, b\}$
- $\Gamma = \{\#, X\}$
- $Z_0 = \#$
- $F = \{q_2\}$



- Welche Sprache akzeptiert dieser Automat?

Motivation zur Greibach-Normalform

- Reguläre Grammatiken: $A \to a$ oder $A \to aB$
- Passendes Modell: Endliche Automaten
  - Sind in einem Zustand
  - Lesen ein Symbol von der Eingabe
  - Wechseln in den nächsten Zustand

- Kontextfreie Grammatiken:
- Passendes Modell: Kellerautomaten
- Hier: Kellerautomat mit nur einem Zustand
  - Lesen ein Symbol vom Stack
  - Lesen ein Symbol von der Eingabe
  - Schreiben beliebig viele Symbole auf den Stack

$\Rightarrow$ Greibach-Normalform!

Motivation zur Greibach-Normalform

- Reguläre Grammatiken: $A \rightarrow a$ oder $A \rightarrow aB$
- Passendes Modell: Endliche Automaten
  - Sind in einem Zustand
  - Lesen ein Symbol von der Eingabe
  - Wechseln in den nächsten Zustand

- Kontextfreie Grammatiken:
- Passendes Modell: Kellerautomaten
- Hier: Kellerautomat mit nur einem Zustand
  - Lesen ein Symbol vom Stack
  - Lesen ein Symbol von der Eingabe
  - Schreiben beliebig viele Symbole auf den Stack

$\Rightarrow$ Greibach-Normalform!

Motivation zur Greibach-Normalform

- Reguläre Grammatiken: $A \to a$ oder $A \to aB$
- Passendes Modell: Endliche Automaten
  - Sind in einem Zustand
  - Lesen ein Symbol von der Eingabe
  - Wechseln in den nächsten Zustand

- Kontextfreie Grammatiken: $A \to a\alpha$ mit $\alpha \in V^*$
- Passendes Modell: Kellerautomaten
- Hier: Kellerautomat mit nur einem Zustand
  - Lesen ein Symbol vom Stack
  - Lesen ein Symbol von der Eingabe
  - Schreiben beliebig viele Symbole auf den Stack

$\Rightarrow$ Greibach-Normalform!

Greibach Normalform

Definition

Eine kontextfreie Grammatik ist in **Greibach-Normalform**, wenn alle Ableitungsregeln von der Form

$$A \rightarrow a\alpha \text{ mit } A \in V, a \in \Sigma \text{ und } \alpha \in V^*$$

sind.

Zur Greibach-Normalform

- Weitere Normalform für CH-2 Grammatiken, d.h. jede Grammatik kann in Greibach-Normalform gebracht werden
- Zur Konstruktion von Kellerautomaten aus Grammatiken
- Es kann stärker, aber äquivalent, verlangt werden, dass auf der rechten Seite höchstens zwei Variablen vorkommen.

Umwandlung in Greibach-Normalform

Nebenbemerkung

Im folgenden stehen Kleinbuchstaben für Terminale, Großbuchstaben für einzelne Nichtterminale und griechische Buchstaben für (eventuell) mehrere Nichtterminale

Die Grammatik sei zunächst in Chomsky-Normalform.

Los geht's!

Annahmen

- Wir gehen davon aus, dass die Grammatik G in Chomsky-Normalform ist, mit $V = \{A_1, \dots, A_m\}$ und $\Sigma = \{a_1, \dots, a_n\}$
- Folglich sind alle Regeln von der Form $A_i \rightarrow A_j A_k$ oder $A_i \rightarrow a_j$

Schritt 1

Ziel: Alle Variablen haben am Beginn der rechten Seite keine Variablen mit gleicher oder niedrigerer Nummer.

- Für alle Variablen V_i
  - Für alle Variablen V_j mit $j < i$
    - Simuliere alle Regeln für A_j bei Produktionen der Form $A_i \rightarrow A_j \alpha$.
  - Für Produktionen der Form $A_i \rightarrow A_i \alpha$ führe eine neue Variable ein (wie: siehe nächste Folie).

$A_i \to A_i\alpha$

Für Regeln der Form

$$A \to A\alpha_1 \mid \dots \mid A\alpha_r$$

$$A \to \beta_1 \mid \dots \mid \beta_s$$

(wobei β_i nicht mit A beginnt) führe ein neues Nichtterminal B ein.
Ersetze nun die Regeln

$$A \to A\alpha_1 \mid \dots \mid A\alpha_r$$

durch

$$A \to \beta_1 B \mid \dots \mid \beta_s B$$

$$B \to \alpha_1 \mid \dots \mid \alpha_r$$

$$B \to \alpha_1 B \mid \dots \mid \alpha_r B$$

Schritt 2

Gehe nun die Produktionen absteigend nach k sortiert durch und simuliere bei alle Regeln mit $A_k \to A_j\alpha$ die Produktionen für A_j auf der rechten Seite.

Da alle Regeln mit einem A_i als linker Seite der Greibach-Normalform genügen, kann man dieses Verfahren nun bei den neuen Regeln für $B_1, \ldots$ auch anwenden.

Danach ist die Grammatik in Greibach-Normalform.

Aufgabe zur Greibach-Normalform

Karlsruher Institut für Technologie

Sei die Grammatik G gegeben durch

- $\Sigma = \{0, 1\}$
- $V = \{A_1, A_2, A_3\}$
- $S = A_1$
- $R = \{$

$$
\begin{aligned}
A_1 &\to A_2 A_3, \\
A_2 &\to A_3 A_1, \\
A_2 &\to 1, \\
A_3 &\to A_1 A_2, \\
A_3 &\to 0
\end{aligned}
$$

$\}$

Bringe G in Greibach-Normalform.

Lösung

- $V = \{A_1, A_2, A_3, B_3\}$
- $\Sigma = A_1$
- $S = A_1$
- $R = \{A_1 \to 1A_3, A_1 \to 0B_3A_1A_3, A_1 \to 1A_3A_2A_1A_3, A_1 \to 0A_1A_3, A_1 \to 1A_3A_2B_3A_1A_3, A_2 \to 0B_3, A_2 \to 1A_3A_2A_1, A_2 \to 0A_1, A_2 \to 1A_3A_2B_3A_1, A_2 \to 1, A_3 \to 0B_3, A_3 \to 1A_3A_2B_3, A_3 \to 1A_3A_2, A_3 \to 0, B_3 \to 1A_3A_2A_2, B_3 \to 0B_3A_1A_3A_3A_2, B_3 \to 1A_3A_2A_1A_3A_3A_2, B_3 \to 0A_1A_3A_3A_2, B_3 \to 1A_3A_2B_3A_1A_3A_3A_2, B_3 \to 1A_3A_3A_2B_3, B_3 \to 0B_3A_1A_3A_3A_2B_3, B_3 \to 1A_3A_2A_1A_3A_3A_2B_3B_3 \to 0A_1A_3A_3A_2B_3, B_3 \to 1A_3A_2B_3A_1A_3A_3A_2B_3\}$

Die ursprüngliche Grammatik hatte nur fünf Regeln.

Lösung

- $V = \{A_1, A_2, A_3, B_3\}$
- $\Sigma = A_1$
- $S = A_1$
- $R = \{A_1 \to 1A_3, A_1 \to 0B_3A_1A_3, A_1 \to 1A_3A_2A_1A_3, A_1 \to 0A_1A_3, A_1 \to 1A_3A_2B_3A_1A_3, A_2 \to 0B_3, A_2 \to 1A_3A_2A_1, A_2 \to 0A_1, A_2 \to 1A_3A_2B_3A_1, A_2 \to 1, A_3 \to 0B_3, A_3 \to 1A_3A_2B_3, A_3 \to 1A_3A_2, A_3 \to 0, B_3 \to 1A_3A_2A_2, B_3 \to 0B_3A_1A_3A_3A_2, B_3 \to 1A_3A_2A_1A_3A_3A_2, B_3 \to 0A_1A_3A_3A_2, B_3 \to 1A_3A_2B_3A_1A_3A_3A_2, B_3 \to 1A_3A_3A_2B_3, B_3 \to 0B_3A_1A_3A_3A_2B_3, B_3 \to 1A_3A_2A_1A_3A_3A_2B_3B_3 \to 0A_1A_3A_3A_2B_3, B_3 \to 1A_3A_2B_3A_1A_3A_3A_2B_3\}$

Die ursprüngliche Grammatik hatte nur fünf Regeln.

Kellerautomat aus Greibach-Normalform

Karlsruher Institut für Technologie

Erinnerung: Greibach-Normalform

Eine kontextfreie Grammatik ist in **Greibach-Normalform**, wenn alle Ableitungsregeln von der Form

$$A \to a\alpha \text{ mit } A \in V, a \in \Sigma \text{ und } \alpha \in V^*$$

sind.

Erinnerung: Übergangsfunktion des Kellerautomaten

Die Eingabe enthält einen Zustand, ein $a \in \Sigma \cup \{\varepsilon\}$ und ein Zeichen des Stacks.

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$$

Wie könnte man mit einer Grammatik G in Greibach-Normalform einen Kellerautomaten konstruieren, der $L(G)$ erkennt?

Kellerautomat aus Greibach-Normalform

Erinnerung: Greibach-Normalform

Eine kontextfreie Grammatik ist in **Greibach-Normalform**, wenn alle Ableitungsregeln von der Form

$$A \to a\alpha \text{ mit } A \in V, a \in \Sigma \text{ und } \alpha \in V^*$$

sind.

Erinnerung: Übergangsfunktion des Kellerautomaten

Die Eingabe enthält einen Zustand, ein $a \in \Sigma \cup \{\varepsilon\}$ und ein Zeichen des Stacks.

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$$

Wie könnte man mit einer Grammatik G in Greibach-Normalform einen Kellerautomaten konstruieren, der $L(G)$ erkennt?

Kellerautomat aus Greibach-Normalform

Erinnerung: Greibach-Normalform

Eine kontextfreie Grammatik ist in **Greibach-Normalform**, wenn alle Ableitungsregeln von der Form

$$A \to a\alpha \text{ mit } A \in V, a \in \Sigma \text{ und } \alpha \in V^*$$

sind.

Erinnerung: Übergangsfunktion des Kellerautomaten

Die Eingabe enthält einen Zustand, ein $a \in \Sigma \cup \{\varepsilon\}$ und ein Zeichen des Stacks.

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$$

Wie könnte man mit einer Grammatik G in Greibach-Normalform einen Kellerautomaten konstruieren, der $L(G)$ erkennt?

Kellerautomat aus Greibach-Normalform

Konstruktion des Kellerautomaten

Gegeben sei eine kontextfreie Grammatik $G = (\Sigma, V, S, R)$ in Greibach-Normalform.

Konstruiere einen Kellerautomaten $PDA = (Q, \Sigma', \Gamma, q_0, Z_0, \delta, F)$ mit:

- $Q := \{q_0\}$
- $F := \emptyset$
- $\Sigma' := \Sigma$
- $\Gamma := V$
- $Z_0 := S$
- $\delta(q_0, a, A) := \{(q_0, \alpha) | (A \to a\alpha) \in R\}$

Der Automat akzeptiert durch leeren Stack.

Kellerautomat aus Greibach-Normalform

Karlsruher Institut für Technologie

Konstruktion des Kellerautomaten

Gegeben sei eine kontextfreie Grammatik $G = (\Sigma, V, S, R)$ in Greibach-Normalform.

Konstruiere einen Kellerautomaten $PDA = (Q, \Sigma', \Gamma, q_0, Z_0, \delta, F)$ mit:

- $Q := \{q_0\}$
- $F := \emptyset$
- $\Sigma' := \Sigma$
- $\Gamma := V$
- $Z_0 := S$
- $\delta(q_0, a, A) := \{(q_0, \alpha) | (A \rightarrow a\alpha) \in R\}$

Der Automat akzeptiert durch leeren Stack.

Kellerautomat

$PDA = (Q, \Sigma', \Gamma, q_0, Z_0, \delta, F)$ mit:

- $Q := \{q_0\}$
- $F := \emptyset$
- $\Sigma' := \Sigma$
- $\Gamma := V$
- $Z_0 := A_1$
- δ siehe nächste Folie

Umwandlung

Aus $A_1 \rightarrow 1A_3, A_1 \rightarrow 1A_3A_2A_1A_3, A_1 \rightarrow 1A_3A_2B_3A_1A_3$ wird
$\delta(q_0, A_1, 1) = \{(q_0, A_3), (q_0, A_3A_2A_1A_3), (q_0, A_3A_2B_3A_1A_3)\}$

δ

- $\delta(q_0, A_1, 0) = \{(q_0, B_3 A_1 A_3), (q_0, A_1 A_3)\}$
- $\delta(q_0, A_1, 1) = \{(q_0, A_3), (q_0, A_3 A_2 A_1 A_3), (q_0, A_3 A_2 B_3 A_1 A_3)\}$
- $\delta(q_0, A_2, 0) = \{(q_0, B_3), (q_0, A_1)\}$
- $\delta(q_0, A_2, 1) = \{(q_0, A_3 A_2 A_1), (q_0, A_3 A_2 B_3 A_1), (q_0, \varepsilon)\}$
- $\delta(q_0, A_3, 0) = \{(q_0, B_3), (q_0, \varepsilon)\}$
- $\delta(q_0, A_3, 1) = \{(q_0, A_3 A_2 B_3), (q_0, A_3 A_2, A_3)\}$
- $\delta(q_0, B_3, 0) =$ $\{(q_0, B_3 A_1 A_3 A_3 A_2), (q_0, A_1 A_3 A_3 A_2), (q_0, B_3 A_1 A_3 A_3 A_2 B_3),$ $(q_0, A_1 A_3 A_3 A_2 B_3)\}$
- $\delta(q_0, B_3, 1) =$ $\{(q_0, A_3 A_2 A_2), (q_0, A_3 A_2 A_1 A_3 A_3 A_2), (q_0, A_3 A_2 B_3 A_1 A_3 A_3 A_2),$ $(q_0, A_3 A_3 A_2 B_3), (q_0, A_3 A_2 A_1 A_3 A_3 A_2 B_3), (q_0, A_3 A_2 B_3 A_1 A_3 A_3 A_2 B_3)\}$

Tripelkonstruktion

- Umkehrung der Konstruktionsrichtung
- Aus einem PDA $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$, der durch leeren Stack akzeptiert, wird eine Grammatik G mit $L_\mathcal{A} = L(G)$ erzeugt.

- $V := \{[q, X, p] \mid p, q \in Q, X \in \Gamma\} \cup \{S\}$
- $R :=$
  - $S \to [q_0, Z_0, q]$ für alle $q \in Q$
  - $[q, X, q_{m+1}] \to a[q_1, Y_1, q_2]...[q_m, Y_m, q_{m+1}]$ für alle q_2, ..., $q_{m+1} \in Q$, falls $(q_1, Y_1, ..., Y_m) \in \delta(q, a, X)$

Aufgabe

Über dem Alphabet $\Sigma = \{(,)\}$ ist die Sprache $L_{()}$ der korrekten Klammerausdrücke gegeben. Ferner ist die Grammatik $G_{()}$ gegeben. Dabei ist $G_{()} = (\{(,)\}, \{S\}, S, R)$ mit

$$R = \{S \to \epsilon | SS | (S)\}$$

1. Konstruiere einen Kellerautomaten, der die Sprache $L_{()}$ durch leeren Stack erkennt. Modifiziere diesen Kellerautomaten so, dass er $L_{()}$ durch akzeptierenden Endzustand erkennt.
2. Dokumentiere eine akzeptierende Berechnung der Wortes $(()())$.

Aufgabe

Über dem Alphabet $\Sigma = \{(,)\}$ ist die Sprache $L_{()}$ der korrekten Klammerausdrücke gegeben. Ferner ist die Grammatik $G_{()}$ gegeben. Dabei ist $G_{()} = (\{(,)\}, \{S\}, S, R)$ mit

$$R = \{S \to \epsilon | SS | (S)\}$$

3. Zeige, dass $G_{()}$ genau $L_{()}$ erzeugt.
4. Was ist das maximale k, so dass $G_{()}$ Chomsky-Typ k hat?
5. Gibt es eine Grammatik mit Chomsky-Typ $k+1$, die $L_{()}$ erzeugt? Begründe deine Antwort.
6. Bestimme eine Grammatik G' für $L_{()} \setminus \{\epsilon\}$ in Greibach-Normalform und zeige exemplarisch, wie man daraus einen Kellerautomaten für $L_{()}$ ableiten kann.

Wiederholung Chomsky-Hierarchie

$u \in V^+$, $v \in (\Sigma \cup V)^*$, $A \in V$, $a \in \Sigma$.
Ausnahme: S kommt bei Chomsky-1-Grammatiken nicht auf rechten Seiten vor.

Typ	Bezeichnung	Regeln	Abgeschlossen				Modell
			$\cup$	$\cap$	$\cdot$	L^c	
0							
1							
2							
3							

Wiederholung Chomsky-Hierarchie

$u \in V^+$, $v \in (\Sigma \cup V)^*$, $A \in V$, $a \in \Sigma$.
Ausnahme: S kommt bei Chomsky-1-Grammatiken nicht auf rechten Seiten vor.

			Abgeschlossen				
Typ	Bezeichnung	Regeln	$\cup$	$\cap$	$\cdot$	L^c	Modell
0	semientscheidbar	beliebig	ja	ja	ja	nein	TM
1							
2							
3							

Wiederholung Chomsky-Hierarchie

$u \in V^+$, $v \in (\Sigma \cup V)^*$, $A \in V$, $a \in \Sigma$.
Ausnahme: S kommt bei Chomsky-1-Grammatiken nicht auf rechten Seiten vor.

Typ	Bezeichnung	Regeln	Abgeschlossen $\cup$	Abgeschlossen $\cap$	Abgeschlossen $\cdot$	Abgeschlossen L^c	Modell
0	semientscheidbar	beliebig	ja	ja	ja	nein	TM
1	kontextsensitiv	$u \rightarrow v$ $\lvert u\rvert \leq \lvert v\rvert$	ja	ja	ja	ja	LBA
2							
3							

Wiederholung Chomsky-Hierarchie

Karlsruher Institut für Technologie

$u \in V^+$, $v \in (\Sigma \cup V)^*$, $A \in V$, $a \in \Sigma$.
Ausnahme: S kommt bei Chomsky-1-Grammatiken nicht auf rechten Seiten vor.

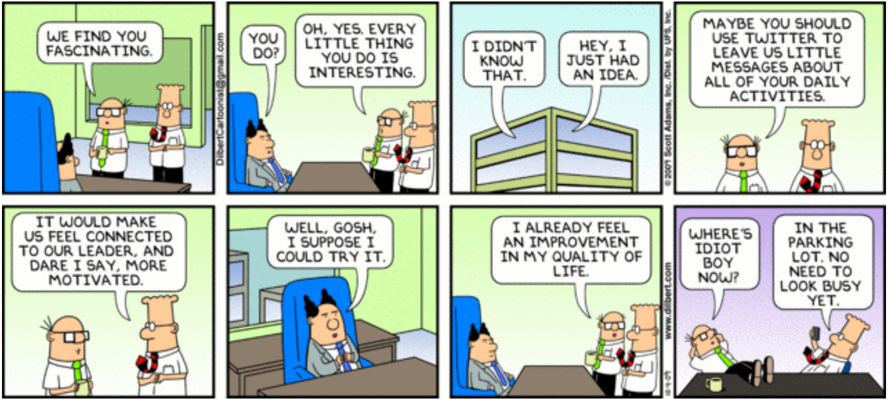
			Abgeschlossen				
Typ	Bezeichnung	Regeln	$\cup$	$\cap$	$\cdot$	L^c	Modell
0	semientscheidbar	beliebig	ja	ja	ja	nein	TM
1	kontextsensitiv	$u \rightarrow v$ $\lvert u\rvert \leq \lvert v\rvert$	ja	ja	ja	ja	LBA
2	kontextfrei	$A \rightarrow v$	ja	nein	ja	nein	PDA
3							

Wiederholung Chomsky-Hierarchie

$u \in V^+$, $v \in (\Sigma \cup V)^*$, $A \in V$, $a \in \Sigma$.
Ausnahme: S kommt bei Chomsky-1-Grammatiken nicht auf rechten Seiten vor.

Typ	Bezeichnung	Regeln	Abgeschlossen $\cup$	$\cap$	$\cdot$	L^c	Modell
0	semientscheidbar	beliebig	ja	ja	ja	nein	TM
1	kontextsensitiv	$u \to v$ $\lvert u\rvert \leq \lvert v\rvert$	ja	ja	ja	ja	LBA
2	kontextfrei	$A \to v$	ja	nein	ja	nein	PDA
3	regulär	$A \to a$ $A \to aB$	ja	ja	ja	ja	NEA

Bis zum nächsten Mal!



Lizenzen