Theoretische Grundlagen der Informatik
Tutorium XII & XIII (SR 301)
Tut Nr. 4 – Üb3, Chomsky Typ 0 & 1, TM

David Münch

Karlsruher Institut für Technologie
Institut für Informatik
IKS Müller-Quade

19. November 2009

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

Organisatorisches

Email: muenchdavid@gmail.com

`https://www.stud.uni-karlsruhe.de/~uhbro/`

Tutorium 12: Donnerstags 8:00 Uhr - Raum 301
Tutorium 13: Donnerstags 9:45 Uhr - Raum 301

Übungsblattabgabe Mittwochs 12:00 Uhr.

Organisatorisches

Deckblatt benutzen: `http://www.stud.uni-karlsruhe.de/~unbdh/deckblatt/index.php?course=6`

Gruppenarbeit erwünscht, aber jeder muss handschriftliche Lösung mit Namen aller Gruppenteilnehmer abgeben.

50% der Punkte sind notwendig für den Schein.

Organisatorisches

Nicht abgeholte Übungsblätter können ab sofort bei Nico Döttling R274 abgeholt werden.

`https://puck.iaks.uka.de/eiss/?id=167`

Was wollen wir heute erreichen?

Was wollen wir heute erreichen?

- Übungsblatt 3 besprechen.

Was wollen wir heute erreichen?

- Übungsblatt 3 besprechen.
- Pumping-Lemma für kontextfreie Sprachen.

Was wollen wir heute erreichen?

- Übungsblatt 3 besprechen.
- Pumping-Lemma für kontextfreie Sprachen.
- Chomsky Typ 1 und Typ 0.

Was wollen wir heute erreichen?

- Übungsblatt 3 besprechen.
- Pumping-Lemma für kontextfreie Sprachen.
- Chomsky Typ 1 und Typ 0.
- Einführung in Turingmaschinen.

Aufgabe 1

Gegeben sei folgenden Sprache

$$\mathcal{L} = \{a^i b^j | i \geq j\}$$

a) Zeigen Sie: $\mathcal{L}$ ist nicht regulär.

Aufgabe 1

Gegeben sei folgenden Sprache

$$\mathcal{L} = \{a^i b^j | i \geq j\}$$

a) Zeigen Sie: $\mathcal{L}$ ist nicht regulär.

b) Geben Sie einen Kellerautomaten an, der $\mathcal{L}$ erkennt.

Wiederholung

Satz: Pumping-Lemma

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, sodass für jedes Wort $w \in L$ mit $|w| > n$ eine Darstellung

$$w = uvx \text{ mit } |uv| \leq n, v \neq \epsilon,$$

existiert, bei der auch $uv^ix \in L$ ist für alle $i \in \mathbb{N}_0$.

Wiederholung

Satz: Pumping-Lemma

Sei L eine reguläre Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, sodass für jedes Wort $w \in L$ mit $|w| > n$ eine Darstellung

$$w = uvx \text{ mit } |uv| \leq n, v \neq \epsilon,$$

existiert, bei der auch $uv^ix \in L$ ist für alle $i \in \mathbb{N}_0$.

Hinweis:
Man verwendet das Pumping-Lemma um zu zeigen, dass eine Sprache nicht regulär ist.

Aufgabe 1a

$$\mathcal{L} = \{a^i b^j | i \geq j\}$$

Aufgabe 1a

$$\mathcal{L} = \{a^i b^j | i \geq j\}$$

- Annahme: $\mathcal{L}$ ist regulär.

Aufgabe 1a

$$\mathcal{L} = \{a^i b^j | i \geq j\}$$

- Annahme: $\mathcal{L}$ ist regulär.
- Sei $n \in \mathbb{N}$ die Pumping-Lemma Zahl.

Aufgabe 1a

$$\mathcal{L} = \{a^i b^j | i \geq j\}$$

- Annahme: $\mathcal{L}$ ist regulär.
- Sei $n \in \mathbb{N}$ die Pumping-Lemma Zahl.
- Betrachte $w = a^n b^n \in \mathcal{L}$.

Aufgabe 1a

$$\mathcal{L} = \{a^i b^j | i \geq j\}$$

- Annahme: $\mathcal{L}$ ist regulär.
- Sei $n \in \mathbb{N}$ die Pumping-Lemma Zahl.
- Betrachte $w = a^n b^n \in \mathcal{L}$.
- Es sei dann $w = uvx$ eine Zerlegung wie im PL mit $|uv| \leq n, |v| \geq 1$. uv besteht nur aus a's.

Aufgabe 1a

$$\mathcal{L} = \{a^i b^j | i \geq j\}$$

- Annahme: $\mathcal{L}$ ist regulär.
- Sei $n \in \mathbb{N}$ die Pumping-Lemma Zahl.
- Betrachte $w = a^n b^n \in \mathcal{L}$.
- Es sei dann $w = uvx$ eine Zerlegung wie im PL mit $|uv| \leq n, |v| \geq 1$. uv besteht nur aus a's.
- Pumpen. Wähle z.B. $i = 0$: $uv^0x = a^{n-|v|}b^n \notin \mathcal{L}$.

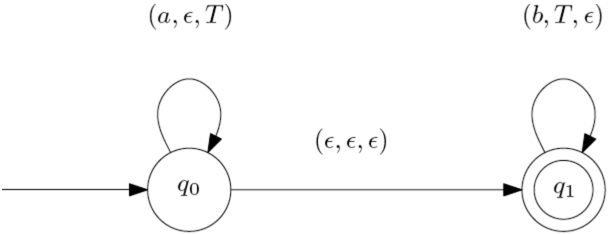
Aufgabe 1a

$\mathcal{L} = \{a^i b^j | i \geq j\}$

- Annahme: $\mathcal{L}$ ist regulär.
- Sei $n \in \mathbb{N}$ die Pumping-Lemma Zahl.
- Betrachte $w = a^n b^n \in \mathcal{L}$.
- Es sei dann $w = uvx$ eine Zerlegung wie im PL mit $|uv| \leq n, |v| \geq 1$. uv besteht nur aus a's.
- Pumpen. Wähle z.B. $i = 0$: $uv^0x = a^{n-|v|}b^n \notin \mathcal{L}$.
- Dies ist ein Widerspruch zum PL $\Rightarrow \mathcal{L}$ ist nicht regulär.

Aufgabe 1b

Wir definieren einen nichtdeterministischen Kellerautomaten $\mathcal{K} = (\mathcal{Q}, \Sigma, \Gamma, \delta, q_0, \mathcal{F})$ mit $\mathcal{Q} = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $\Gamma = \{T\}$ und $\mathcal{F} = \{q_1\}$. Der Zustandsübergangsgraph δ sei gegeben durch



Idee: Für jedes a lege ein Symbol T auf den Kellerspeicher. Für jedes folgende b entferne ein T vom Kellerspeicher. Bei $\#b \leq \#T = \#a$ akzeptiert $\mathcal{K}$. Beachte: $\varepsilon \in \mathcal{L}$

Aufgabe 2

Gegeben sei die Sprache

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

a) Konstruieren Sie für $\mathcal{L}$ eine kontextfreie Grammatik in Chomsky Normalform.
b) Überprüfen Sie mit dem Cocke-Younger-Kasami Algorithmus ob $w = abaabaaba$ in $\mathcal{L}$ liegt.

Aufgabe 2a

Wir geben zuerst eine intuitiv einfache Grammatik an:
$G = (\{S\}, \{a, b\}, S, \{S \to \varepsilon | a | b | aSa | bSb\})$.
Anschließend überführen wir G in Chomsky-Normalform.

Definition: Chomsky-Normalform

Eine kontextfreie Grammatik G heißt in CNF, falls alle Regeln eine der folgenden Formen haben:

- $A \to BC$
- $A \to a$
- $S \to \varepsilon$.

Dabei sind A, B, C, S Variablen, S Startsymbol und a Terminal.

Vorgehen: Chomsky-Normalform

1. $V \cup S_0$, $P \cup \{S_0 \rightarrow S\}$ und falls $\varepsilon \in L$, dann $P \cup \{S_0 \rightarrow \varepsilon\}$.
2. Beseitigung der ε-Produktionen $A \rightarrow \varepsilon$ $(A \neq S_0)$:
   Für jedes Vorkommen von A auf der rechten Seite einer Produktion füge neue Produktion ohne A hinzu.
3. Beseitige Produktionen der Form $A \rightarrow B$ für $B \rightarrow \alpha$ füge $A \rightarrow \alpha$ hinzu, es sei denn $A \rightarrow \alpha$ wurde schon einmal entfernt.
4. Überführe alle Regeln in die Form $A \rightarrow BC$ oder $A \rightarrow a$. Falls $A \rightarrow U_1 .... U_k$ und $k \geq 3$, dann $A \rightarrow U_1 Y_1, Y_1 \rightarrow U_2 Y_2, ...$.

$$G = (\{S\}, \{a, b\}, S, P)$$
$$P = \{S \to \varepsilon | a | b | aSa | bSb\}.$$

$G = (\{S\}, \{a, b\}, S, P)$
$P = \{S \to \varepsilon|a|b|aSa|bSb\}.$
Schritt 1:
$G_{CNF} = (\{S, S_0\}, \{a, b\}, S_0, P)$
$P = \{S_0 \to \varepsilon|S,$
$S \to \varepsilon|a|b|aSa|bSb\}.$

$G = (\{S\}, \{a, b\}, S, P)$
$P = \{S \rightarrow \varepsilon|a|b|aSa|bSb\}.$

Schritt 1:

$G_{CNF} = (\{S, S_0\}, \{a, b\}, S_0, P)$
$P = \{S_0 \rightarrow \varepsilon|S,$
$S \rightarrow \varepsilon|a|b|aSa|bSb\}.$

Schritt 2:

$G_{CNF} = (\{S, S_0\}, \{a, b\}, S_0, P)$
$P = \{S_0 \rightarrow \varepsilon|S,$
$S \rightarrow a|b|aSa|bSb|aa|bb\}.$

$G = (\{S\}, \{a, b\}, S, P)$
$P = \{S \to \varepsilon|a|b|aSa|bSb\}.$

Schritt 1:

$G_{CNF} = (\{S, S_0\}, \{a, b\}, S_0, P)$
$P = \{S_0 \to \varepsilon|S,$
$S \to \varepsilon|a|b|aSa|bSb\}.$

Schritt 2:

$G_{CNF} = (\{S, S_0\}, \{a, b\}, S_0, P)$
$P = \{S_0 \to \varepsilon|S,$
$S \to a|b|aSa|bSb|aa|bb\}.$

Schritt 3:

$G_{CNF} = (\{S, S_0\}, \{a, b\}, S_0, P)$
$P = \{S_0 \to \varepsilon|a|b|aSa|bSb|aa|bb,$
$S \to a|b|aSa|bSb|aa|bb\}.$

Schritt 4.1:

$G_{CNF} = (\{S, S_0, Y_1, Y_2\}, \{a, b\}, S_0, P)$
$P = \{S_0 \rightarrow \varepsilon | a | b | aY_1 | bY_2 | aa | bb,$
$S \rightarrow a | b | aY_1 | bY_2 | aa | bb$
$Y_1 \rightarrow Sa,$
$Y_2 \rightarrow Sb\}.$

Schritt 4.1:

$G_{CNF} = (\{S, S_0, Y_1, Y_2\}, \{a, b\}, S_0, P)$

$P = \{S_0 \to \varepsilon|a|b|aY_1|bY_2|aa|bb,$

$S \to a|b|aY_1|bY_2|aa|bb$

$Y_1 \to Sa,$

$Y_2 \to Sb\}.$

Schritt 4.2:

$G_{CNF} = (\{S, S_0, Y_1, Y_2\}, \{a, b\}, S_0, P)$

$P = \{S_0 \to \varepsilon|a|b|X_aY_1|X_bY_2|X_aX_a|X_bX_b,$

$S \to a|b|X_aY_1|X_bY_2|X_aX_a|X_bX_b\}'$

$Y_1 \to SX_a,$

$Y_2 \to SX_b,$

$X_a \to a,$

$X_b \to b\}.$

Pumping-Lemma für kf. Sprachen

Definition: Pumping-Lemma für kontextfreie Sprachen

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 $uv^iwx^iy \in L$ ist.

Beispiel

Zeige mithilfe des Pumping-Lemmas, dass $L = \{a^i b^i c^i | i \geq 1\}$ nicht kontextfrei ist.

Beispiel

Zeige mithilfe des Pumping-Lemmas, dass $L = \{a^i b^i c^i | i \geq 1\}$ nicht kontextfrei ist.

Aufgabe

Gib eine Grammatik G mit $L(G) = L = \{a^i b^i c^i | i \geq 1\}$ von höchstmöglichem Chomsky Typ an.

Aufgabe

Zeige mithilfe des Pumping-Lemmas, dass $L = \{a^i b^j c^i d^j | i, j \geq 1\}$ nicht kontextfrei ist.

Definition: Chomsky-Hierarchie

- Grammatiken ohne weitere Einschränkungen heißen Grammatiken vom **Typ 0**.

Definition: Chomsky-Hierarchie

- Grammatiken ohne weitere Einschränkungen heißen Grammatiken vom **Typ 0**.
- Grammatiken, bei denen alle Ableitungsregeln die Form

  $u \to v$ mit $u \in V^+, v \in ((V \cup \Sigma)\setminus\{S\})^+$ und $|u| \leq |v|$, oder
  $S \to \varepsilon$

  haben, heißen **kontextsensitiv** od. Grammatiken vom **Typ 1**.

Definition: Chomsky-Hierarchie

- Grammatiken ohne weitere Einschränkungen heißen Grammatiken vom **Typ 0**.
- Grammatiken, bei denen alle Ableitungsregeln die Form

  $u \to v$ mit $u \in V^+, v \in ((V \cup \Sigma)\backslash\{S\})^+$ und $|u| \leq |v|$, oder
  $S \to \varepsilon$

  haben, heißen **kontextsensitiv** od. Grammatiken vom **Typ 1**.
- Grammatiken, bei denen alle Ableitungsregeln die Form
  $A \to v$ mit $A \in V$ und $v \in (V \cup \Sigma)^*$
  haben, heißen **kontextfrei** oder Grammatiken vom **Typ 2**.

Definition: Chomsky-Hierarchie

- Grammatiken ohne weitere Einschränkungen heißen Grammatiken vom **Typ 0**.
- Grammatiken, bei denen alle Ableitungsregeln die Form

  $u \rightarrow v$ mit $u \in V^+, v \in ((V \cup \Sigma) \setminus \{S\})^+$ und $|u| \leq |v|$, oder
  $S \rightarrow \varepsilon$

  haben, heißen **kontextsensitiv** od. Grammatiken vom **Typ 1**.
- Grammatiken, bei denen alle Ableitungsregeln die Form
  $A \rightarrow v$ mit $A \in V$ und $v \in (V \cup \Sigma)^*$
  haben, heißen **kontextfrei** oder Grammatiken vom **Typ 2**.
- Grammatiken, bei denen alle Ableitungsregeln die Form
  $A \rightarrow v$ mit $A \in V$ und $v = \varepsilon$ oder $v = aB$ mit $a \in \Sigma, B \in V$
  haben, heißen **rechtslinear** oder Grammatiken vom **Typ 3**.

Übersicht Maschinenmodelle

Maschinenmodell für Typ 3:
Deterministischer und nichtdeterministischer endlicher Automat.

Übersicht Maschinenmodelle

Maschinenmodell für Typ 3:
Deterministischer und nichtdeterministischer endlicher Automat.
Maschinenmodell für Typ 2:
Deterministischer und nichtdeterministischer Kellerautomat.

Übersicht Maschinenmodelle

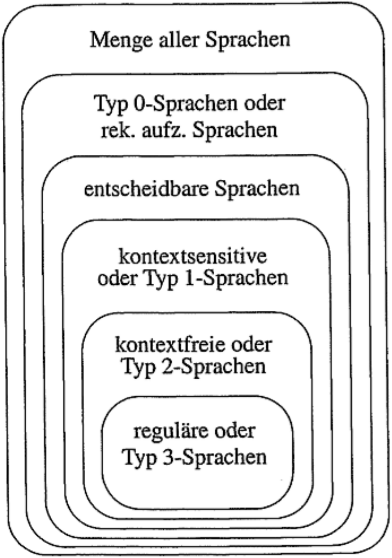
Maschinenmodell für Typ 3:
Deterministischer und nichtdeterministischer endlicher Automat.
Maschinenmodell für Typ 2:
Deterministischer und nichtdeterministischer Kellerautomat.
Maschinenmodell für Typ 1:
Nichtdeterministische linear beschränkte Turingmaschine (NLBA)

Übersicht Maschinenmodelle

Maschinenmodell für Typ 3:
Deterministischer und nichtdeterministischer endlicher Automat.
Maschinenmodell für Typ 2:
Deterministischer und nichtdeterministischer Kellerautomat.
Maschinenmodell für Typ 1:
Nichtdeterministische linear beschränkte Turingmaschine (NLBA)
Offenes Problem: NLBA äquivalent Det. LBA

Übersicht Maschinenmodelle

Maschinenmodell für Typ 3:
Deterministischer und nichtdeterministischer endlicher Automat.
Maschinenmodell für Typ 2:
Deterministischer und nichtdeterministischer Kellerautomat.
Maschinenmodell für Typ 1:
Nichtdeterministische linear beschränkte Turingmaschine (NLBA)
Offenes Problem: NLBA äquivalent Det. LBA
Maschinenmodell für Typ 0:
Turingmaschine


Menge aller Sprachen
Typ 0-Sprachen oder
rek. aufz. Sprachen
entscheidbare Sprachen
kontextsensitive
oder Typ 1-Sprachen
kontextfreie oder
Typ 2-Sprachen
reguläre oder
Typ 3-Sprachen

Definition:

- Eine Sprache $L \subseteq \Sigma^*$ heisst **rekursiv** oder **entscheidbar**, wenn es eine TM gibt, die auf allen Eingaben stoppt und eine Eingabe w genau dann akzeptiert, wenn $w \in L$ gilt.

Definition:

- Eine Sprache $L \subseteq \Sigma^*$ heisst **rekursiv** oder **entscheidbar**, wenn es eine TM gibt, die auf allen Eingaben stoppt und eine Eingabe w genau dann akzeptiert, wenn $w \in L$ gilt.
- Eine Sprache $L \subseteq \Sigma^*$ heisst **rekursiv-aufzählbar** oder **semi-entscheidbar** oder **Chomsky Typ 0**, wenn es eine TM gibt, die genau die Eingaben w akzeptiert für die $w \in L$. (Das Verhalten der TM für $w \notin L$ ist nicht defniniert.)

Turingmaschine

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

Q, endliche Zustandsmenge

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

Q, endliche Zustandsmenge

Σ, endliches Bandalphabet

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

Q, endliche Zustandsmenge

Σ, endliches Bandalphabet

$\sqcup$, Blanksymbol, beachte $\sqcup \notin \Sigma$

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

Q, endliche Zustandsmenge

Σ, endliches Bandalphabet

$\sqcup$, Blanksymbol, beachte $\sqcup \notin \Sigma$

Γ, endliches Bandalphabet mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

Q, endliche Zustandsmenge

Σ, endliches Bandalphabet

$\sqcup$, Blanksymbol, beachte $\sqcup \notin \Sigma$

Γ, endliches Bandalphabet mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$

$q_0 \in Q$, einem Startzustand

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

Q, endliche Zustandsmenge

Σ, endliches Bandalphabet

$\sqcup$, Blanksymbol, beachte $\sqcup \notin \Sigma$

Γ, endliches Bandalphabet mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$

$q_0 \in Q$, einem Startzustand

$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, N\}$, einer Überführungsfunktion.

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine
$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

- Q, endliche Zustandsmenge
- Σ, endliches Bandalphabet
- $\sqcup$, Blanksymbol, beachte $\sqcup \notin \Sigma$
- Γ, endliches Bandalphabet mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$
- $q_0 \in Q$, einem Startzustand
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$, einer Überführungsfunktion.
- $q_{accept} \in Q$, akzeptierender Zustand. Auch: $\mathcal{F}$ Menge von akzeptierenden Zuständen.

Definition: deterministische Turingmaschine (DTM)

Eine deterministische Turingmaschine $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ besteht aus:

- Q, endliche Zustandsmenge
- Σ, endliches Bandalphabet
- $\sqcup$, Blanksymbol, beachte $\sqcup \notin \Sigma$
- Γ, endliches Bandalphabet mit $\Sigma \cup \{\sqcup\} \subseteq \Gamma$
- $q_0 \in Q$, einem Startzustand
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$, einer Überführungsfunktion.
- $q_{accept} \in Q$, akzeptierender Zustand. Auch: $\mathcal{F}$ Menge von akzeptierenden Zuständen.
- $q_{reject} \in Q$, ablehnender Zustand.

Aufgabe

Entwerfe eine DTM, die als Eingabe eine (zusammenhängende) Folge von Nullen erhält und als Ausgabe eine Folge von Einsen der doppelten Länge auf das Band schreibt. Beschreibe zunächst kurz in Worten, wie die Turingmaschine funktioniert.

Aufgabe

Entwerfe eine NTM, welche die Sprache $L = \{ww | w \in \{a, b\}^*\}$ erkennt. Beschreibe zunächst kurz in Worten, wie die Turingmaschine funktioniert.

Hinweis zum Übungsblatt Nr. 4

Bitte die Turingmaschinen als Programmcode an mich per Email kompatibel zu `http://ironphoenix.org/tril/tm/` abgeben.

Zum Beispiel:

$1, \_ \; 2, 1, >$
$1, 1 \; 3, \_, <$
$2, \_ \; 3, 1, >$
$2, 1 \; 4, 1, >$

Reflexion

Was haben wir heute gelernt?

Reflexion

Was haben wir heute gelernt?

- Pumping-Lemma

Reflexion

Was haben wir heute gelernt?

- Pumping-Lemma
- Chomsky-Normalform wiederholt

Reflexion

Was haben wir heute gelernt?

- Pumping-Lemma
- Chomsky-Normalform wiederholt
- CYK-Algorithmus wiederholt

Reflexion

Was haben wir heute gelernt?

- Pumping-Lemma
- Chomsky-Normalform wiederholt
- CYK-Algorithmus wiederholt
- kompletten Überblick über die Chomsky-Hierarchie erhalten

Reflexion

Was haben wir heute gelernt?

- Pumping-Lemma
- Chomsky-Normalform wiederholt
- CYK-Algorithmus wiederholt
- kompletten Überblick über die Chomsky-Hierarchie erhalten
- alle Maschinenmodelle dazu kennen gelernt

Reflexion

Was haben wir heute gelernt?

- Pumping-Lemma
- Chomsky-Normalform wiederholt
- CYK-Algorithmus wiederholt
- kompletten Überblick über die Chomsky-Hierarchie erhalten
- alle Maschinenmodelle dazu kennen gelernt
- Turingmaschine

Noch Fragen?

Vorschau

Vorschau

- Abschlusseigenschaften

Vorschau

- Abschlusseigenschaften
- Entscheidbarkeit

Vorschau

- Abschlusseigenschaften
- Entscheidbarkeit
- Wortproblem

Vorschau

- Abschlusseigenschaften
- Entscheidbarkeit
- Wortproblem
- Halteproblem

Vorschau

- Abschlusseigenschaften
- Entscheidbarkeit
- Wortproblem
- Halteproblem
- Postsches Korrespondenzproblem

Bis zum nächsten Mal