

# FORMALE SYSTEME

## ÜBUNG 6

---

Eric Kunze

`eric.kunze@tu-dresden.de`

TU Dresden, 26. November 2021

letzte Änderung:  
20.11.2021, 22:52

# Aufgabe 1:

## *Pumping-Lemma*

---

# NICHTREGULARITÄT DURCH PUMPEN

## Idee:

- ▶ Jeder DFA hat nur endlich viele Zustände  $n$
- ▶ Aber manche reguläre Sprachen enthalten beliebig lange Wörter

## Wie kann ein DFA Wörter mit mehr als $n$ Zeichen akzeptieren?

- ▶ Dann muss der DFA beim Einlesen einen Zustand mehr als einmal besuchen
- ▶ Dafür muss es in den Zustandsübergängen eine Schleife geben
- ▶ Diese Schleife kann man aber auch mehr als einmal durchlaufen

*Jedes akzeptierte Wort mit  $\geq n$  Zeichen hat einen Teil, den man beliebig oft wiederholen – „aufpumpen“ – kann.*

# DAS PUMPING-LEMMA

**Satz (Pumping-Lemma):** Für jede reguläre Sprache  $\mathbf{L}$  gibt es eine Zahl  $n \geq 0$ , so dass gilt:  
für jedes Wort  $z \in \mathbf{L}$  mit  $|z| \geq n$   
gibt es eine Zerlegung  $z = uvw$  mit  $|v| \geq 1$  und  $|uv| \leq n$ , so dass:  
für jede Zahl  $k \geq 0$  gilt:  $uv^k w \in \mathbf{L}$

*Beweis:* Sei  $\mathcal{M}$  ein DFA für  $\mathbf{L}$  mit  $|Q|$  Zuständen. Wir wählen  $n = |Q| + 1$ .

Ein akzeptierender Lauf für ein beliebiges Wort  $z$  mit  $|z| = \ell \geq n$  muss in den ersten  $n$  Schritten einen Zustand  $p$  zweimal besuchen (sagen wir: nach  $i$  und  $j$  Schritten), hat also die Form:

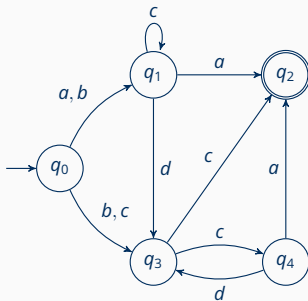
$$q_0 \xrightarrow{z_1} q_1 \xrightarrow{z_2} \dots \xrightarrow{z_{i-1}} q_{i-1} \xrightarrow{z_i} p \xrightarrow{z_{i+1}} q_{i+1} \xrightarrow{z_{i+2}} \dots \xrightarrow{z_{j-1}} q_{j-1} \xrightarrow{z_j} p \xrightarrow{z_{j+1}} q_{j+1} \xrightarrow{z_{j+2}} \dots \xrightarrow{z_\ell} q_\ell$$

Die gesuchte Zerlegung ist  $u = z_1 \dots z_i$ ,  $v = z_{i+1} \dots z_j$ ,  $w = z_{j+1} \dots z_\ell$ .

Der Lauf  $(q_0 \dots q_{i-1} p)(q_{i+1} \dots q_{j-1} p)^k (q_{j+1} \dots q_\ell)$  akzeptiert  $uv^k w$ .



# AUFGABE 1



- a) Geben Sie für jedes  $z \in \{bc, adc, cda, bc dc, ac dc\}$  alle Zerlegungen  $z = uvw$  mit  $u, w \in \Sigma^*$ ,  $v \in \Sigma^+$  an, sodass für alle  $k \geq 0$  gilt:  $uv^k w \in L(\mathcal{M})$ . Begründen Sie Ihre Antworten.
- b) Ermitteln Sie eine Zahl  $n \in \mathbb{N}$ , sodass für alle  $z \in L(\mathcal{M})$  mit  $|z| \geq n$  gilt, dass eine Zerlegung  $z = uvw$  mit  $u, w \in \Sigma^*$ ,  $v \in \Sigma^+$  und  $|uv| \leq n$  existiert, sodass für alle  $k \geq 0$  gilt:  $uv^k w \in L(\mathcal{M})$ .

## **Aufgabe 2:**

### ***Regularität von Sprachen***

---

# BEWEIS VON NICHTREGULARITÄT

**Satz (Myhill & Nerode):** Eine Sprache  $L$  ist genau dann regulär, wenn  $\simeq_L$  endlich viele Äquivalenzklassen hat.

**Satz:** Wenn  $L_1$  und  $L_2$  regulär sind, dann auch  $L_1 \cap L_2$ ,  $L_1 \cup L_2$ ,  $L_1^*$  und  $\bar{L}_1$ .

**Satz (Pumping-Lemma):** Für jede reguläre Sprache  $L$  gibt es eine Zahl  $n \geq 0$ , so dass gilt:  
für jedes Wort  $x \in L$  mit  $|x| \geq n$   
gibt es eine Zerlegung  $x = uvw$  mit  $|v| \geq 1$  und  $|uv| \leq n$ , so dass:  
für jede Zahl  $k \geq 0$  gilt:  $uv^k w \in L$

## AUFGABE 2

Gegeben ist das Alphabet  $\Sigma = \{a, b\}$ . Welche der folgenden Sprachen  $L_j$  über  $\Sigma$  mit  $1 \leq j \leq 2$  ist regulär? Beweisen Sie Ihre jeweilige Antwort.

a)  $L_1 = \{a^i b^j : 1 \leq i \leq 15\}$

b)  $L_2 = \{a^n b^m a^{n \cdot m} : n, m \geq 0\}$



## **Aufgabe 3:**

### ***Wiederholung***

---

## AUFGABE 3

Beweisen oder widerlegen Sie unter Verwendung von Resultaten aus der Vorlesung folgende Aussagen.

- a) Für die Grammatik  $G = (\{S, X, Y, Z\}, \{a, b\}, \{S \rightarrow Y, X \rightarrow b, Y \rightarrow aYYb, aY \rightarrow aZ, ZY \rightarrow ZX, Z \rightarrow a\}, S)$  gilt:  $abab \in L(G)$ .
- b) Kann eine Sprache  $L$  von einem DFA erkannt werden, so gibt es auch einen  $\varepsilon$ -NFA  $\mathcal{M}$  mit  $L(\mathcal{M}) = L$ .
- c) Für jeden NFA  $\mathcal{M}$  mit Wortübergängen gibt es einen äquivalenten NFA.
- d) Es gibt eine reguläre Sprache, für welche die Anzahl der Äquivalenzklassen der *Nerode*-Rechtskongruenz endlich ist.
- e) Wenn es für eine Sprache  $L$  ein  $n \in \mathbb{N}$  gibt, so dass die *Nerode*-Rechtskongruenz  $\simeq_L$  höchstens  $n$  Äquivalenzklassen hat, so kann  $L$  von einem DFA erkannt werden.
- f) Für jede Sprache  $L$  gilt:  $L = \bigcup_{u \in L} [u]_{\simeq_L}$ .

# Aufgabe 4

## *Chomsky-Normalform*

---

# ELIMINIEREN VON $\varepsilon$ -REGELN

**Eingabe:** CFG  $G = \langle V, \Sigma, P, S \rangle$

**Ausgabe:**  $\varepsilon$ -freie CFG  $G' = \langle V', \Sigma, P', S' \rangle$  mit  $\mathbf{L}(G') = \mathbf{L}(G)$

- ▶ Initialisiere  $P' := P$  und  $V' := V$
- ▶ Berechne  $V_\varepsilon = \{A \in V \mid A \Rightarrow^* \varepsilon\}$
- ▶ Entferne alle  $\varepsilon$ -Regeln aus  $P'$
- ▶ Solange es in  $P'$  eine Regel  $B \rightarrow xAy$  gibt, mit

$$A \in V_\varepsilon \quad |x| + |y| \geq 1 \quad B \rightarrow xy \notin P'$$

wähle eine solche Regel und setze  $P' := P' \cup \{B \rightarrow xy\}$

- ▶ Falls  $S \in V_\varepsilon$  dann definiere ein neues Startsymbol  $S' \notin V$ , setze  $V' := V' \cup \{S'\}$  und  $P' := P' \cup \{S' \rightarrow S, S' \rightarrow \varepsilon\}$ . Falls  $S \notin V_\varepsilon$ , dann verwenden wir einfach  $S' := S$  als Startsymbol.

# ELIMINIERUNG VON KETTENREGELN

Eine **Kettenregel** ist eine Regel der Form  $A \rightarrow B$ .

Sei  $G = \langle V, \Sigma, P, S \rangle$   $\epsilon$ -frei. Eine äquivalente Grammatik ohne Kettenregeln ist gegeben durch  $G' = \langle V, \Sigma, P', S \rangle$ :

## Eliminieren von Kettenregeln:

$E(A)$  ... Menge aller  $B \in V$ , die man von  $A \in V$  aus über Kettenregeln erreichen kann:

- (1)  $A \in E(A)$
- (2) Falls  $B \in E(A)$  und  $B \rightarrow B' \in P$  mit  $B' \in V$  dann  $B' \in E(A)$ .  
Wiederhole.

$$\implies P' = \bigcup_{A \in V} \left\{ A \rightarrow w \mid \text{es gibt } B \rightarrow w \in P \text{ mit } w \notin V \text{ und } B \in E(A) \right\}$$

# DIE CHOMSKY-NORMALFORM

Eine kontextfreie Grammatik  $G = \langle V, \Sigma, P, S \rangle$  ist in **Chomsky-Normalform (CNF)**, wenn alle ihre Produktionsregeln eine der beiden folgenden Formen haben:

$$A \rightarrow BC \quad (\text{mit } B, C \in V) \quad \text{oder} \quad A \rightarrow c \quad (\text{mit } c \in \Sigma)$$

## Umwandlung in CNF:

- (1) Eliminierung von  $\varepsilon$ -Regeln
- (2) Eliminierung von Kettenregeln

(3) Extrahiere Regeln der Form  $A \rightarrow c$ , so dass alle anderen Regeln  $B \rightarrow w$  keine Terminale mehr in  $w$  enthalten.

- ▷ für jedes Symbol  $a \in \Sigma$ :  
neue Variable  $V_a$  mit Regel  $V_a \rightarrow a$
- ▷ für Regeln  $A \rightarrow w$  mit  $|w| > 1$ :  
ersetze jedes Vorkommen von  $a \in \Sigma$  in  $w$  durch  $V_a$

(4) Reduziere Regeln der Form  $A \rightarrow B_1 \cdots B_n$  auf  $n = 2$

Für jede Produktionsregel  $A \rightarrow B_1 \cdots B_n$  mit  $n > 2$ :

- ▷ Führe  $n - 2$  neue Variablen  $C_1, \dots, C_{n-2}$  ein
- ▷ Ersetze die Regel durch neue Regeln:

$$\begin{aligned} A &\rightarrow B_1 C_1 \\ C_1 &\rightarrow B_2 C_2 \\ &\vdots \\ C_{n-3} &\rightarrow B_{n-2} C_{n-2} \\ C_{n-2} &\rightarrow B_{n-1} B_n \end{aligned}$$

## AUFGABE 4

Betrachten Sie die Grammatik  $G_0 = \langle V, \Sigma, P, S \rangle$  mit  $V = \{S, T, U, V, R\}$ ,  $\Sigma = \{a, b\}$  und

$$P = \{ \begin{array}{llll} S \rightarrow \varepsilon, & S \rightarrow aSb, & S \rightarrow T, & S \rightarrow R, \\ T \rightarrow bbT, & T \rightarrow U, & U \rightarrow aaU, & U \rightarrow bbT, \\ V \rightarrow bSa, & R \rightarrow \varepsilon, & R \rightarrow bSa & \end{array} \}$$

- Konstruieren Sie eine Grammatik  $G_1$ , die keine Regeln der Form  $A \rightarrow \varepsilon$  für  $A \in V$  enthält. Erweitern Sie dazu, wenn nötig, die Grammatik  $G_0$  um ein neues Startsymbol  $S'$  und entsprechende Regeln.
- Geben Sie zu  $G_1$  eine äquivalente Grammatik  $G_2$  an, die keine Kettenregeln, also Produktionen der Form  $A \rightarrow B$  mit Nichtterminalsymbolen  $A, B$ , enthält.
- Geben Sie eine Grammatik  $G_3$  in Chomsky-Normalform an mit  $L(G_3) = L(G_2) \setminus \{\varepsilon\}$ .



## Aufgabe 5

### *CYK-Algorithmus*

---

## AUFGABE 5

Gegeben ist folgende Grammatik  $G = (V, \Sigma, P, S)$  mit  $V = \{S, X, M, A, B\}$ ,  $\Sigma = \{a, b\}$  und

$$P = \left\{ \begin{array}{lll} S \rightarrow \varepsilon, & S \rightarrow AX, & S \rightarrow AB, \\ X \rightarrow MB, & & \\ M \rightarrow AB, & M \rightarrow AX, & \\ A \rightarrow a, & & \\ B \rightarrow a, & B \rightarrow b & \end{array} \right\}$$

Verwenden Sie den CYK-Algorithmus (mit der Matrix-Notation aus der Vorlesung), um für die folgenden Wörter  $w_i$  zu entscheiden, ob  $w_i \in L(G)$  ist.

a)  $w_1 = aaabba$

b)  $w_2 = aabbaa$