TECHNISCHE UNIVERSITÄT DRESDEN

FORMALE SYSTEME

ÜBUNG 6

Eric Kunze
eric.kunze@tu-dresden.de

TU Dresden, 26. November 2021

letzte Änderung:
24.11.2021, 22:59

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 Zustandsübergangen 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.

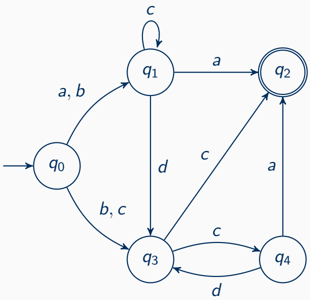
Satz (Pumping-Lemma): Für jede reguläre Sprache $\mathbf{L}$
gibt es eine Zahl $n \geq 0$, so dass gilt:
für jedes Wort $z \in \mathbf{L}$ mit $|z| \geq n$
gibt es eine Zerlegung $z = uvw$ mit $|v| \geq 1$ und $|uv| \leq n$, so dass:
für jede Zahl $k \geq 0$ gilt: $uv^k w \in \mathbf{L}$

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:

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

Die gesuchte Zerlegung ist $u = z_1 \cdots z_i$, $v = z_{i+1} \cdots z_j$, $w = z_{j+1} \cdots z_\ell$.
Der Lauf $(q_0 \ldots q_{i-1} p)(q_{i+1} \ldots q_{j-1} p)^k (q_{j+1} \ldots q_\ell)$ akzeptiert $uv^k w$. $\square$

AUFGABE 1



a) Geben Sie für jedes $z \in \{bc, adc, cda, bcdc, acdc\}$ 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 $\mathbf{L}$ ist genau dann regulär, wenn $\simeq_{\mathbf{L}}$ endlich viele Äquivalenzklassen hat.

Satz: Wenn $\mathbf{L}_1$ und $\mathbf{L}_2$ regulär sind, dann auch $\mathbf{L}_1 \cap \mathbf{L}_2$, $\mathbf{L}_1 \cup \mathbf{L}_2$, $\mathbf{L}_1^*$ und $\overline{\mathbf{L}}_1$.

Satz (Pumping-Lemma): Für jede reguläre Sprache $\mathbf{L}$
gibt es eine Zahl $n \geq 0$, so dass gilt:
für jedes Wort $x \in \mathbf{L}$ 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: $uv^k w \in \mathbf{L}$

AUFGABE 2

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

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

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

Lösung

a) Die Sprache L_1 ist endlich, da i eine obere Grenze hat. Jede endliche Sprache ist regulär.

b) L_2 ist nicht regulär — Pumping-Lemma: Angenommen L_2 sei regulär. Dann existiert nach dem Pumping-Lemma ein $n \geq 0$, sodass jedes Wort $x \in L_2$ mit $|x| \geq n$ gepumpt werden kann. Insbesondere muss dies auch für das Wort $x = a^n b^1 a^{n \cdot 1}$ gelten. Dementsprechend muss es eine Zerlegung

$$x = uvw \quad \text{mit } |uv| \leq n \text{ und } |v| \geq 1$$

geben. Wegen $|uv| \leq n$ muss $uv = a^\ell$ mit $1 \leq \ell \leq n$ gelten (d.h. uv liegt in den ersten a's). Ein Teil der a's muss dem v zugeschrieben werden, d.h. es gilt $v = a^h$ mit $1 \leq h \leq \ell$. Damit können wir nun pumpen, insbesondere mit $k = 2$:

$$uv^2w = a^{\ell-h}a^{2h}a^{n-\ell}ba^n = a^{\ell-h+2h+n-\ell}ba^n = a^{n+h}ba^n \notin L_2$$

im Widerspruch zur Aussage des Pumping-Lemmas, was $uv^2w \in L_2$ sichern würde. Damit kann die Annahme der Regularität nicht richtig gewesen sein und L_2 muss nicht regulär sein.

Aufgabe 3:
Wiederholung

AUFGABE 3

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

a) Für die Grammatik $G = (\{S, X, Y, Z\}, \{a, b\}, \{S \to Y,\ X \to b,\ Y \to aYYb,\ aY \to aZ,\ ZY \to ZX,\ Z \to a\}, S)$ gilt: $abab \in L(G)$.

b) Kann eine Sprache L von einem DFA erkannt werden, so gibt es auch einen ε-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 3

Lösung

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

a) ✗ Für die Grammatik $G = (\{S, X, Y, Z\}, \{a, b\}, \{S \to Y,\ X \to b,\ Y \to aYYb,\ aY \to aZ,\ ZY \to ZX,\ Z \to a\}, S)$ gilt: $abab \in L(G)$.

b) Kann eine Sprache L von einem DFA erkannt werden, so gibt es auch einen ε-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 3

Lösung

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

a) ✗ Für die Grammatik $G = (\{S, X, Y, Z\}, \{a, b\}, \{S \to Y,\ X \to b,\ Y \to aYYb,\ aY \to aZ,\ ZY \to ZX,\ Z \to a\}, S)$ gilt: $abab \in L(G)$.

b) ✓ Kann eine Sprache L von einem DFA erkannt werden, so gibt es auch einen ε-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 3

Lösung

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

a) ✗ Für die Grammatik $G = (\{S, X, Y, Z\}, \{a, b\}, \{S \to Y,\ X \to b,\ Y \to aYYb,\ aY \to aZ,\ ZY \to ZX,\ Z \to a\}, S)$ gilt: $abab \in L(G)$.

b) ✓ Kann eine Sprache L von einem DFA erkannt werden, so gibt es auch einen ε-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 3

Lösung

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

a) ✗ Für die Grammatik $G = (\{S, X, Y, Z\}, \{a, b\}, \{S \to Y,\ X \to b,\ Y \to aYYb,\ aY \to aZ,\ ZY \to ZX,\ Z \to a\}, S)$ gilt: $abab \in L(G)$.

b) ✓ Kann eine Sprache L von einem DFA erkannt werden, so gibt es auch einen ε-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 3

Lösung

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

a) ✗ Für die Grammatik $G = (\{S, X, Y, Z\}, \{a, b\}, \{S \to Y,\ X \to b,\ Y \to aYYb,\ aY \to aZ,\ ZY \to ZX,\ Z \to a\}, S)$ gilt: $abab \in L(G)$.

b) ✓ Kann eine Sprache L von einem DFA erkannt werden, so gibt es auch einen ε-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 3

Lösung

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

a) ✗ Für die Grammatik $G = (\{S, X, Y, Z\}, \{a, b\}, \{S \to Y,\ X \to b,\ Y \to aYYb,\ aY \to aZ,\ ZY \to ZX,\ Z \to a\}, S)$ gilt: $abab \in L(G)$.

b) ✓ Kann eine Sprache L von einem DFA erkannt werden, so gibt es auch einen ε-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 ε-REGELN

Eingabe: CFG $G = \langle V, \Sigma, P, S \rangle$
Ausgabe: ε-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 ε-Regeln aus P'
- Solange es in P' eine Regel $B \to xAy$ gibt, mit

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

  wähle eine solche Regel und setze $P' := P' \cup \{B \to xy\}$
- Falls $S \in V_\varepsilon$ dann definiere ein neues Startsymbol $S' \notin V$, setze $V' := V' \cup \{S'\}$ und $P' := P' \cup \{S' \to S, S' \to \varepsilon\}$. Falls $S \notin V_\varepsilon$, dann verwenden wir einfach $S' := S$ als Startsymbol.

ELIMINIERUNG VON KETTENREGELN

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

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

Eliminieren von Kettenregeln:
$E(A) \ldots$ 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 $B \to B' \in P$ mit $B' \in V$ dann $B' \in E(A)$. Wiederhole.

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

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:

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

Umwandlung in CNF:

(1) Eliminierung von ε-Regeln

(2) Eliminierung von Kettenregeln

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

- ▷ für jedes Symbol $\mathtt{a} \in \Sigma$:
  neue Variable V_a mit Regel $\mathsf{V}_\mathsf{a} \to \mathtt{a}$
- ▷ für Regeln $\mathsf{A} \to w$ mit $|w| > 1$:
  ersetze jedes Vorkommen von $\mathtt{a} \in \Sigma$ in w durch V_a

(4) Reduziere Regeln der Form $\mathsf{A} \to \mathsf{B}_1 \cdots \mathsf{B}_\mathsf{n}$ auf $n = 2$

Für jede Produktionsregel $\mathsf{A} \to \mathsf{B}_1 \cdots \mathsf{B}_\mathsf{n}$ mit $n > 2$:

- ▷ Führe $n - 2$ neue Variablen $\mathsf{C}_1, \ldots, \mathsf{C}_{\mathsf{n-2}}$ ein
- ▷ Ersetze die Regel durch neue Regeln:

$$\begin{aligned}
\mathsf{A} &\to \mathsf{B}_1\mathsf{C}_1 \\
\mathsf{C}_1 &\to \mathsf{B}_2\mathsf{C}_2 \\
&\vdots \\
\mathsf{C}_{\mathsf{n-3}} &\to \mathsf{B}_{\mathsf{n-2}}\mathsf{C}_{\mathsf{n-2}} \\
\mathsf{C}_{\mathsf{n-2}} &\to \mathsf{B}_{\mathsf{n-1}}\mathsf{B}_\mathsf{n}
\end{aligned}$$

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

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

a) Konstruieren Sie eine Grammatik G_1, die keine Regeln der Form $A \to \varepsilon$ für $A \in V$ enthält. Erweitern Sie dazu, wenn nötig, die Grammatik G_0 um ein neues Startsymbol S' und entsprechende Regeln.

b) Geben Sie zu G_1 eine äquivalente Grammatik G_2 an, die keine Kettenregeln, also Produktionen der Form $A \to B$ mit Nichtterminalsymbolen A, B, enthält.

c) Geben Sie eine Grammatik G_3 in Chomsky-Normalform an mit $L(G_3) = L(G_2) \setminus \{\varepsilon\}$.

Aufgabe 5

CYK-Algorithmus

CYK: GRUNDIDEE

gegeben: kontextfreie Grammatik G in CNF
Frage: $w = \mathtt{a}_1 \cdots \mathtt{a}_n \in \mathbf{L}(G)$?

- Falls $|w| = 1$, dann ist $w \in \Sigma$ und es gilt:
  $w \in \mathbf{L}(G)$ genau dann wenn es eine Regel $S \to w$ in G gibt
- Falls $|w| > 1$, dann ist:
  $w \in \mathbf{L}(G)$ genau dann wenn es eine Regel $S \to AB$ und eine Zahl i gibt, so dass gilt

$$A \Rightarrow^* \mathtt{a}_1 \cdots \mathtt{a}_i \qquad \text{und} \qquad B \Rightarrow^* \mathtt{a}_{i+1} \cdots \mathtt{a}_n$$

Idee: Fall 2 reduziert das Problem $S \overset{?}{\Rightarrow}{}^* \, w$ auf zwei einfachere Probleme $A \overset{?}{\Rightarrow}{}^* \, \mathtt{a}_1 \cdots \mathtt{a}_i$ und $B \overset{?}{\Rightarrow}{}^* \, \mathtt{a}_{i+1} \cdots \mathtt{a}_n$, die man allerdings für alle Regeln $S \to AB$ und Indizes i lösen muss

CYK: PRAKTISCHE UMSETZUNG

Vorgehen: $V[i,j]$ = Menge aller A mit $A \Rightarrow^* w_{i,j}$
$\rightsquigarrow$ $V[i,j]$ können in einer Dreiecksmatrix notiert werden

- Diagonale = Fall 1: existiert Terminalsymbolregel
- Fixiere Element $\blacksquare$: sei $\blacktriangleleft$ in der gleichen Zeile ganz links und $\blacktriangledown$ direkt unten drunter
  - wenn eine Regel $\blacklozenge \to \blacktriangleleft\ \blacktriangledown$, dann füge $\blacklozenge$ zu $\blacksquare$ hinzu
  - schiebe $\blacktriangleleft$ nach rechts und $\blacktriangledown$ nach unten und wiederhole

Ist am Ende das Startsymbol $S \in V[1,|w|]$, dann liegt w in der Sprache

Beispiel: Wir betrachten das Wort $w = \texttt{a} + \texttt{b} \cdot \texttt{c}$ der Länge $|w| = 5$.

a	$V[1,1]$	$V[1,2]$	$V[1,3]$	$V[1,4]$	$V[1,5]$
+		$\blacktriangleleft$	$V[2,3]$	$\blacksquare \cup \blacklozenge$	$V[2,5]$
b			$V[3,3]$	$\blacktriangledown$	$V[3,5]$
·				$V[4,4]$	$V[4,5]$
c					$V[5,5]$
	a	+	b	·	c

AUFGABE 5

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

$$
\begin{aligned}
P = \{ \quad & S \to \varepsilon, && S \to AX, && S \to AB, \\
& X \to MB, && && \\
& M \to AB, && M \to AX, && \\
& A \to a, && && \\
& B \to a, && B \to b && \}
\end{aligned}
$$

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) $w_2 = aabbaa$