Theoretische Informatik

Frank Bergmann

June 13, 2000

Contents

1 Übungsblatt 6

1.1 Aufgabe 20

1.1.1 Aufgabenstellung

Welche der folgenden Sprachen sind regulär? Beweisen Sie Ihre Aussagen:

1. $L_1 \stackrel{def}{=} \{wcw \,|\, w \in \{0,1\}^n\}$ für ein festes $n \in N$

2. $L_2 \stackrel{def}{=} \left\{wcw \,|\, w \in \{0,1\}^*\right\}$

3. $L_3 \stackrel{def}{=} \left\{w \in \{0\}^* \mid |w| \neq 2^n \text{ für alle } n \in N\right\}$

4. $L_4 \stackrel{def}{=} \left\{ww^R \,|\, w \in \{0,1\}^*\right\}$

1.1.2 Lösung

1. Angenommen die Sprache $L_1 \overset{def}{=} \{wcw \mid w \in \{0,1\}^n\}$ wäre für ein festes $n \in N$ regulär. Man betrachtet dann DFAs M_i, die genau einen Wert aus L_1 erkennen. So gilt zum Beispiel für $n = 2$:

$$\begin{array}{lll} M_1 & \text{akzeptiert} & \{00c00\} \\ M_2 & \text{akzeptiert} & \{01c01\} \end{array}$$

   Zu einem gegebenem $n \in N$ gibt es dann 2^n solcher DFAs M_i. Dies sind jedoch endlich viele und somit ist L_1 dann die Vereinigung regulärer Mengen und ist somit regulär.

2. Angenommen $L_2 \overset{def}{=} \left\{wcw \mid w \in \{0,1\}^*\right\}$ wäre regular. Dann gibt es nach dem Pumping-Lemma eine Konstante $n \in N$, desweiteren gibt es für das Wort $\{0^n c 0^n\} \in L_2$ eine Zerlegung:

$$0^n c 0^n = xyz \text{ mit } 0 < |y| < n \text{ und } xy^i z \in L_2 \forall i \geq 0$$

   Daraus ergeben sich die folgenden Fälle:
   1. Fall: y enthält ein c daraus folgt: $xy^2z \in L_2$ würde 2 c's enthalten. In dem Fall wäre das Wort aber kein Element der Sprache mehr.
   2. Fall: y enthält kein c und z enthält ein c. Durch das Pumping-Lemma erhält man so links des c's mehr Nullen, als rechts davon. Diese Worte müssten in L_2 sein, sind es aber nicht. Also auch hier ein Widerspruch.
   3. Fall: Dieser Fall ist analog zu *Fall 2:* y enthält kein c jedoch x enthält eines. Somit erhält man diesmal rechts von c mehr Nullen, als links davon. Wieder stellt dies einen Widerspruch dar.
   Zusammenfassend lässt sich also sagen, dass L_2 nicht regulär ist.

3. Angenommen die Sprach $L_3 \overset{def}{=} \left\{w \in \{0\}^* \mid |w| \neq 2^n \text{ für alle } n \in N\right\}$ wäre regulär. Dann wäre auch das Komplement der Sprache

$$\overline{L}_3 \overset{def}{=} \left\{w \in \{0\}^* \mid |w| = 2^n \text{ für alle } n \in N\right\}$$

   regulär. Sei nun n die Konstante des Pumping-Lemmas und $2^m > n$ gewählt. Dann besitzt das Wort $w = 0^{2^n}$ eine Zerlegung $w = xyz$ mit $0 < |y| < n$ und $xy^iz \in \overline{L_3}$ für alle $i \geq 0$.
   Insbesondere gilt auch $xy^2z \in \overline{L_3}$ dass heist, $\left|xy^2z\right|$ ist eine 2er Potenz. Damit gilt:

$$2^n < 2^m + 1 \leq \left|xy^2z\right| \leq 2^m + n < 2^n + 2^m = 2^{m+1}$$

   Daraus folgt aber, dass $\left|xy^2z\right|$ keine 2er Potenz gewesen sein kann. Dies stellt einen Widerspruch zur Annahme dar und somit kann L_3 nicht regulär gewesen sein.

4. Angenommen die Sprache $L_4 \stackrel{def}{=} \left\{ww^R \,|\, w \in \{0,1\}^*\right\}$ wäre regulär. Mit dem wissen, dass aufgrund der Abschlusseigenschaften dann auch die Sprache:

$$L_4' = L_4 \cap \{0^r 1^s 0^n \,|\, n, r, s \geq 0\} = \left\{0^n 1^{2r} 0^n \,|\, n, r \geq 0\right\}$$

regulär ist. Nun kann man einen Homomorphismus h_1 aufstellen mit:

$$\begin{aligned} h_1(a) &= 0 \\ h_1(b) &= 0 \\ h_1(1) &= 1 \end{aligned}$$

Dann ist die Sprache:

$$L_4'' = h_1^{-1}(L_4') \cap \left\{a^r 1^{2r} b^r \,|\, r, n \geq 0\right\}$$

ebenfalls regulär. Betrachtet man nun den Homomorphismus h_2 mit:

$$\begin{aligned} h_2(a) &= a \\ h_2(1) &= \epsilon \\ h_2(b) &= b \end{aligned}$$

Die Sprache definiert durch:

$$L_4''' = h_2(L_4'') = \{a^n b^n \,|\, n \geq 0\}$$

müsste dann ebenfalls regulär sein, jedoch wissen wir, dass L_4''' nicht regulär ist. Demzufolge kann auch L_4 nicht regulär sein.

1.2 Aufgabe 21

1.2.1 Aufgabenstellung

Beweisen Sie, dass die Sprache

$$L \stackrel{def}{=} \left\{0^i 1^j \,|\, i, j \geq 1 \wedge ggT(i,j) = 1\right\}$$

nicht regulär ist.
Hinweis: Sie können folgenden Satz benutzen.

Satz

Zu jedem $i, j \in Z$ mit $ggT(i,j) = 1$ existieren $u, v \in Z$ mit $ui + vj = 1$.

1.2.2 Lösung

Zuerst eine Überlegungen zu dem Hinweis: Zu jedem $i, j \in Z$ mit $ggT(i, j) = 1$ existieren $u, v \in Z$ mit $ui + vj = 1$.

Folgerung: zu $i, j \in N \mid i, j > 1$ mit $ggT(i, j) = 1$ existieren $u, v \in Z$ mit $ui + vj = 1$.

Beweis: Nach dem Satz existieren $u, v \in Z$ mit $ui + vj = 1$. Da $i, j > 0$ sind, muss entwerde $u < 0$ und $v > 0$ gelten, oder aber $u > 0$ und $v < 0$ gelten. Im ersten Fall ist die Folgerung bereits bewiesen. Für den zweiten Fall gilt:

Sei nun also $u \geq 0$ und $v \leq 0$. Dann gilt:

$$\begin{aligned} ui + vj = 1 \quad &\iff \quad ui - uij + uij + vj = 1 \\ \Rightarrow \quad (u - uj)\, i \quad &+ (v + uj)\, j = 1 \end{aligned}$$

man definiert nun $u' = (u - uj)$ und $v' = (v - uj)$ dann gilt:

$$u'i + v'j = 1$$

Da weiterhin gelten soll, dass $u < 0$ und $v > 0$ (bzw. $u' < 0$ und $v' > 0$) sind noch folgende Fälle zu betrachten:

$u = 0 \Rightarrow vj = 1 \Rightarrow j = 1$ Widerspruch
$v = 0 \Rightarrow ui = 1 \Rightarrow i = -1$ Widerspruch
$u' = 0 \Rightarrow u - uj = 0 \Rightarrow j = 1$ Widerspruch
$v' = u \Rightarrow u + uj = 1 \Rightarrow r(j - 1) = 1 \Rightarrow j + \frac{1}{r}n$ Widerspruch
$\Rightarrow j = 0,$ da aber $r < 0$ auch hier Widerspruch

Nun aber zurück zur Aufgabe. Zuerst wird angenommen, dass die Sprache $L \overset{def}{=} \left\{0^i 1^j \mid i, j \geq 1 \wedge ggT(i, j) = 1\right\}$ regulär sei. Dann gibt es nach dem Pumping-Lemma ein $n \in N$, so dass für $w \in L$ mit $|w| \geq n$ gilt:

$$w = xyz,\ 0 < |y| < n,\ xy^i z \in L \text{ für alle } i \geq 0$$

Sei nun $w = 0^n 1^{n'}$, wobei n' die kleinste Primzahl größer als n ist. Somit ist der $ggT(n, n') = 1$. Damit ist $w \in L$. Da weiterhin gilt, dass die Wortlänge $|w| \geq n$ ist, existiert eine Zerlegung $w = xyz$. Wenn y Nullen und Einsen enthalten würde, so würde es beim "Pumpen" einen Widerspruch geben, da dann das Wort nicht nur aus einem Block Nullen und einem Block Einsen bestehen würde. Also enthält y nur Nullen, oder nur Einsen. OBdA seien dies hier nur Nullen.

Dann ist $w = xyz_1z_2$ mit $z_1 \in \{0\}^*, z_2 \in \{1\}^*$. Weiterhin gilt: $|yz_1| = n$ und $|z_2| = n'$. Daraus folgt aber: $n = |y| + |xz_1| = s + r$ (mit $0 < s < n, r > 0$). Da aber n' Primzahl ist folgt, dass der $ggT(s, n') = 1$ ist. Nach unserer Folgerung gibt es nun $u, v \in Z$ mit $u < 0, v > 0$ und es gilt:

$$\begin{aligned} us + vn' \quad &= \quad 1 \\ u < 0 \Rightarrow u \quad &= \quad -|u| \\ us + rn' = 1 \quad &\iff \quad uvs + vrn' = r \iff |u|\, rs + r \in n' = r \\ \iff \quad v\,|u|\, s + r \quad &= vrn' \text{ mit } r, |u|, s, v > 0 \end{aligned}$$

Nach dem Pumping-Lemma gilt, $xy^{r|u|}z \in L$, dass heist der $ggT\left(\left|xy^{r|u|}z_1\right|, |z_2|\right) = 1$. Jedoch gelten:

$$\begin{aligned}
&\left|xy^{r|u|}z_1\right| = |y|\,r\,|u| + |xz_1| = sr\,|u| + r \text{ und}\\
&ggT\left(\left|xy^{r|u|}z_1\right|, |z_2|\right) = ggT\left(sr\,|u| + r, n'\right) = ggT\left(vrn', n'\right) = n' > 1
\end{aligned}$$

Da dies einen Widerspruch zur Anname, dass der $ggT(\cdot,\cdot) = 1$ ist, darstelt ist L nicht regulär.

1.3 Aufgabe 22

1.3.1 Aufgabenstellung

Sei L die Menge aller korrekt geformten arithmetischen Ausdrücke über dem Alphabet $\{\mathrm{op}, (,), \cdot, +\}$.

1. Geben Sie eine kontextfreie Grammatik für L an.
2. Beweisen Sie, dass Ihre Grammatik L erzeugt.
3. Beweisen Sie: L ist nicht regulär

1.3.2 Lösung

1. Zu generieren ist also die Grammatik $G = (V, \Sigma, P, S)$ dabei sind:

   $$\begin{aligned}
   &\Sigma = \{op, (,), \cdot, +\}\\
   &S = S\\
   &V = \{S\}
   \end{aligned}$$

   Für P gilt:

   $$P = \{ \quad \begin{aligned} &S \to op\\ &S \to S + S\\ &S \to S \cdot S\\ &S \to (S) \quad \} \end{aligned}$$

   Die so definierte Grammatik ist kontextfrei, da gilt:

   $$\forall p \in P \colon p \in (V - \Sigma) \times V^*$$

2. Nun ist zu beweisen, dass diese Grammatik auch L erzeugt. Sei nun $w \in \Sigma^+$. Zu zeigen ist nun, dass es eine Ableitung $S \overset{*}{\Rightarrow} w$ gibt, für die gilt, dass w ein korrekt geformter arithmetischer Ausdruck ist. Dies soll mittels Induktion über die Wortlänge $|w|$ bewiesen werden.
   Verankerung $|w| = 1$*:* "$\Rightarrow$"

   $$S \overset{*}{\Rightarrow} w \text{ mit } |w| = 1 \Rightarrow S \Rightarrow w \Rightarrow op \Rightarrow w \text{ ist korrekt geformt}$$

   "$\Leftarrow$"

   $w = op$ ist der einzige korrekt geformte Ausdruck der Länge $|w| = 1$, $S \to op \in P$

somit folgt: $S \overset{*}{\Rightarrow} w = op$
Annahme: Die Grammatik erzeuge für Wortlängen $1, \ldots, n$ korregt geformte arithmetische Ausdrücke.
Schritt $|w| = n+1$: "$\Rightarrow$" Es gilt:

$$
\begin{aligned}
&S \overset{*}{\Rightarrow} w \ |w| > 1 \Rightarrow w \neq op, \text{ also gibt es drei Möglichkeiten, wie das Wort aussehen kann :}\\
&1)\, S \Rightarrow S + S \overset{+}{\Rightarrow} w_1 + w_2 = w, \text{ mit } S \overset{*}{\Rightarrow} w_1 \wedge S \overset{*}{\Rightarrow} w_2\\
&2)\, S \Rightarrow S \cdot S \overset{+}{\Rightarrow} w_1 \cdot w_2 = w, \text{ mit } S \overset{*}{\Rightarrow} w_1 \wedge S \overset{*}{\Rightarrow} w_2\\
&3)\, S \Rightarrow (S) \overset{+}{\Rightarrow} (w_1) = w, \text{ mit } S \overset{*}{\Rightarrow} w_1
\end{aligned}
$$

In jedem Fall ist $|w_1| \leq |w|, |w_2| \leq n$, also gilt nach Induktionsvoraussetzung, dass w_1 und w_2 korrekt geformte Ausdrücke sind. Und damit sind auch die Ausdrücke $w_1 + w_2, w_1 \cdot w_2$ und (w_1) wohlgeformte arithmetische Ausdrücke.
"$\Leftarrow$" Sei nun w ein korrekt geformter Ausdruck der Länge $|w| = n+1$. Das bedeutet, dass w nicht nur aus op besteht, sondern entweder $+, \cdot$ oder Klammern enthält. damit sind für:

$$
\begin{aligned}
1) \quad & w = w_1 + w_2\\
2) \quad & w = w_1 \cdot w_2\\
3) \quad & w = (w_1)
\end{aligned}
$$

w_1 und w_2 korrekt geformte Ausdrücke der Länge $\leq n$.
Nach Induktionsvoraussetzung gilt also $S \overset{*}{\Rightarrow} w_1$ und $S \overset{*}{\Rightarrow} w_2$. Damit ist:

$$
\begin{aligned}
&1)\, S \Rightarrow S + S \overset{+}{\Rightarrow} w_1 + S \overset{+}{\Rightarrow} w_1 + w_2 = w\\
&2)\, S \Rightarrow S \cdot S \overset{+}{\Rightarrow} w_2 \cdot S \overset{+}{\Rightarrow} w_1 \cdot w_2 = w\\
&3)\, S \Rightarrow (S) \overset{+}{\Rightarrow} (w_1) = w
\end{aligned}
$$

Also gilt in jedem Fall $S \overset{*}{\Rightarrow} w$ für Wörter der Wortlänge $|w| = n+1$. Und dies war zu zeigen.

3. Angenommen L wäre regulär. Dann kann man einen Homomorphismus definieren mit:

$$
\begin{aligned}
h(op) &= \epsilon\\
h() &= \epsilon\\
h(\cdot) &= \epsilon\\
h('(') &= a\\
h(')') &= b
\end{aligned}
$$

dann wäre auch die Sprache

$$L' = h(L) = \{a^r b^s \,|\, r, s \geq 0\} = \{a^n b^n \,|\, n \geq 0\}$$

regulär. Jedoch wissen wir von dieser, dass sie nicht regulär ist. Und somit kann auch L nicht regulär sein.

1.4 Aufgabe 23

1.4.1 Aufgabenstellung

Wandeln Sie folgende kontextfreie Grammatik $G = (\{S, A, B, C, D, E\}, \{a, b\}, P, S)$ mit

$$P \stackrel{def}{=} \{ \quad \begin{array}{rl} S \to AE|BE|C & A \to aAa|B \\ B \to bB|bb & C \to aCaa|D \\ D \to baD|abD|aa & E \to EE|\epsilon \quad \} \end{array}$$

mittels der aus der Vorlesung bekannten Algorithmen in eine reduzierte kontextfreie Grammatuik um. Dokumentieren Sie alle Zwischenergebnisse.

1.4.2 Lösung

Um G zu reduzieren sind drei Schritte zu tun.

1. Zuerst sind alle ϵ-Produktionen zu entfernen. Die Produktion $S \to \epsilon$ ist nur dann erlaubt, wenn $\epsilon \in L(G)$ ist.
2. Dann sind alle zyklischen Produktionen zu entfernen.
3. Zum Schluss sind dann noch alle überflüssigen Symbole zu entfernen. Das heißt also:

$$\begin{array}{lll} a) & \forall A \in V - \Sigma \exists w \in \Sigma^*: & A \stackrel{*}{\Rightarrow} w \\ b) & \forall A \exists \alpha: & S \stackrel{*}{\Rightarrow} \alpha A \beta \end{array}$$

Dann also los. Doch zunächst schreibe ich die gegebenen Produktionen nochmals ausführlich auf, dann sieht man leichter, welche Produktionen zu entfernen sind :)

$$\begin{array}{rrl} S \to AE & S \to BE & S \to C \\ B \to bB & B \to bb & \\ D \to baD & D \to abD & D \to aa \\ A \to aAa & A \to B & \\ C \to aCaa & C \to D & \\ E \to EE & E \to \epsilon & \end{array}$$

Damit lässt sich doch schon mehr anfangen.

1. Zu der gegebenen Grammatik ergibt sich:

$$\begin{array}{rcl} V_1 & = & \{E\} \\ V_2 & = & \{E\} \cup \{E\} = \{E\} = V_n \\ P_1 & = & \{P\} \cup \{S \to A, S \to B, E \to E\} \\ P_2 & = & P_1 - \{E \to \epsilon\} \\ P' & = & P_2 \cup \{S' \to S\} \end{array}$$

Nun somit fällt die Produktion $E \to \epsilon$ herraus. Und man bekommt die Grammatik G' definiert durch:

$$G' = (\{S, S', A, B, C, D, E\}, \{a, b\}, P', S')$$

mit den Produktionen:

$$\begin{aligned}
P' = \{ \quad & S \to AE, S \to BE, S \to C, S \to A, S \to B \\
& A \to aAa, A \to B \\
& B \to bB, B \to bb \\
& C \to aCaa, C \to D \\
& D \to baD, D \to abD, D \to aa \\
& E \to EE, E \to E \\
& S' \to S \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \}
\end{aligned}$$

Das neue Startsymbol ist demnach S'.

2. Als nächstes sind alle Zyklischen Produktionen zu entfernen. Der Algorithmus liefert hier:

$$\begin{aligned}
& V_{S'}^{(1)} = \{S\}, V_{S'}^{(2)} = \{S\} \cup \{A, B, C\} = \{S, A, B, C\} \\
& V_{S'}^{(3)} = \{S, A, B, C\} \cup \{A, B, C, D\} = \{S, A, B, C, D\} \\
& V_{S'}^{(4)} = \{S, A, B, C, D\} \cup \{A, B, C, D\} = \{S, A, B, C, D\} = V_{S'}^{(N)} \\
& V_{S}^{(1)} = \{A, B, C, D\}, V_{S}^{(2)} = \{A, B, C, D\}, V_{S}^{(3)} = \{A, B, C, D\} = V_{S}^{(N)} \\
& V_{A}^{(1)} = \{B\} = V_{A}^{(N)} \\
& V_{B}^{(N)} = \emptyset \\
& V_{C}^{(1)} = \{D\}, V_{C}^{(2)} = \{D\} = V_{C}^{(N)} \\
& V_{D}^{(1)} = \emptyset \\
& V_{E}^{(1)} = \{E\}, V_{E}^{(2)} = \{E\} = V_{E}^{(N)}
\end{aligned}$$

Damit kommt man nun auf die Grammatik G'', welche definiert ist durch:

$$G'' = (\{S', S, A, B, C, D, E\}, \{a, b\}, P'', S')$$

mit:

$$\begin{aligned}
P'' = \{ \quad & P' - \{S \to A, S \to B, S \to C, A \to D, C \to D, E \to E, S' \to S\} \cup \\
& \{ S' \to AE, S' \to BE, S' \to aAa, S' \to bB, S' \to bB, S' \to bb, S' \to aCaa, S' \to baD \\
& S' \to abD, S' \to aa, S \to aAa, S \to bB, S \to bB, S \to bb, S \to aCaa, S \to baD, S \to abD, S \to \\
& A \to bB \\
& C \to baD, C \to abD, C \to aa \\
& E \to EE\}
\end{aligned}$$

man gelangt so schließlich zu den Produktionen:

$$\begin{aligned}
P'' = \{ \quad & S' \to AE, S' \to BE, S' \to aAa, S' \to bB, S' \to bb, S' \to aCaa, S' \to baD, S' \to abD, S' \to aa \\
& S \to AE, S \to BE, S \to aAa, S \to bB, S \to bb, S \to aCaa, S \to baD, S \to abD, S \to aa \\
& A \to aAa, A \to bB, A \to bb \\
& B \to bB, B \to bb \\
& C \to baD, C \to abD, C \to aCaa, C \to aa \\
& D \to baD, D \to abD, D \to aa \\
& E \to EE
\end{aligned}$$

3. Zum Schluss sind alle Symbole zu entfernen, welche nicht benötigt werden. Dies funktioniert analog, wie im Skript auf Seite 55. Der Algorithmus greift in zwei Schritten:

   1. Schritt:

$$
\begin{aligned}
&V_1 = \{S', S, A, B, C, D\}\\
&V_2 = \{S', S, A, B, C, D\} = V_N\\
&\overline{G} = \left(\overline{V}, \{a, b\}, \overline{P}, S'\right) \text{ mit}\\
&\overline{V} = \{S', S, A, B, C, D\} \cup \{a, b\}\\
&\overline{P} = P'' \cap \left(\overline{V} \times \overline{V}^*\right)\\
&\overline{P} = P'' - \{S' \to AE, S' \to BE, S \to AE, S \to BE, E \to EE\}
\end{aligned}
$$

   2. Schritt:

$$
\begin{aligned}
&H_0 = \{S'\}, H_1 = \{S'\} \cup \{A, B, C, D\} = \{S', A, B, C, D\}\\
&H_2 = \{S', A, B, C, D\} = H_N\\
&G''' = (V''', \{a, b\}, P''', S') \text{ mit}\\
&V''' = \{S', A, B, C, D\} \cup \{a, b\}\\
&P''' = \overline{P} \cap \left(V''' \times (V''')^*\right)
\end{aligned}
$$

   so gelangt man zu den endgültigen Produktionen:

$$
\begin{aligned}
P''' = \{ \quad & S' \to aAa, S' \to bB, S' \to bb, S' \to aCaa, S' \to baD, S' \to abD, S \to aa\\
& A \to aAa, A \to bB, A \to bb\\
& B \to bB, B \to bb\\
& C \to aCaa, C \to baD, C \to abD, C \to aa\\
& D \to baD, D \to abD, D \to aa \qquad \}
\end{aligned}
$$

   dies ist nun die endgültige reduzierte Grammatik.