Formale Grundlagen

10. Übungsaufgaben

2008-06-17, Lösungen

1. Beschreiben Sie durch einen regulären Ausdruck die Menge aller Wörter über dem Alphabet $\{\,\texttt{0}, \texttt{1}\,\}$,

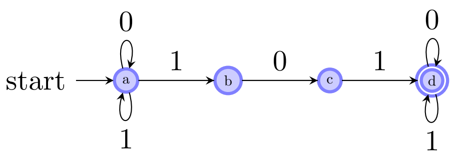
   - die das Wort `101` enthalten.

     Lösung: `.*101.*`

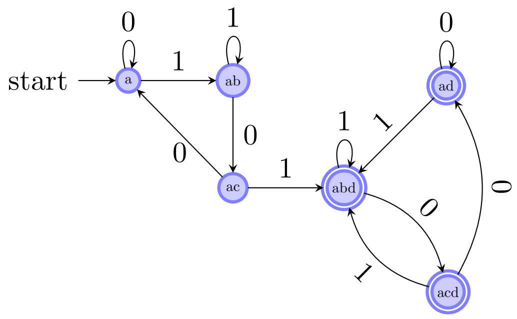
   - die das Wort `101` nicht enthalten.

     Lösung: $0^*(000^*|1)^*0^*$

     Anstatt zu überlegen, kann man auch algorithmisch zu einer Lösung gelangen: Ein Automat, der $L($`.*101.*`$)$ erkennt, ist:

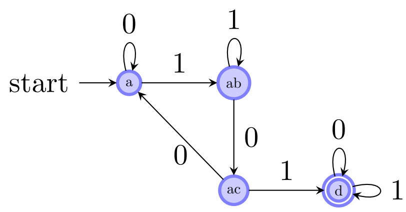


Wir bestimmen einen funktionsgleichen deterministischen Automaten:

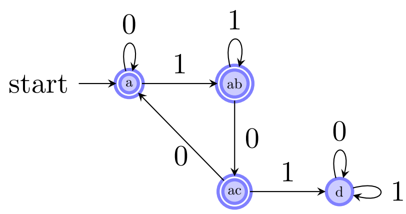


	0	1
a	a	ab
ab	ac	ab
ac	a	abd
abd	acd	abd
acd	ad	abd
ad	ad	abd

Wir bemerken, daß der Automat nach erstmaligem Erreichen eines Finalzustandes (d.h. nach dem ersten Auftreten der Zeichenkette 101) nur noch Finalzustände annimmt. Dieser Automat kann daher vereinfacht werden zu:



Einen Automaten, der die Komplementärsprache erkennt, erhalten wir durch Abänderung der Finalzustände (zum Komplement der ursprünglichen Finalzustände):



Daraus lesen läßt sich leicht ein passender regulärer Ausdruck ablesen.

$$(0^*11^*00)^*(0^*(\epsilon|11^*(\epsilon|0)))$$

- in denen jedes Paar aufeinanderfolgender Nuller vor jedem Paar aufeinanderfolgender Einser kommt.

  Lösung: $(0|10)^*(1|10)^*$

- welche höchstens ein Paar aufeinanderfolgender Nuller und höchstens ein Paar aufeinanderfolgender Einser enthalten.

  Lösung: $(\epsilon|1)(01)^*(10)^*(01)^*(\epsilon|0)|(\epsilon|0)(10)^*(01)^*(10)^*(\epsilon|1)$

2. Beschreiben Sie die gültigen Variablennamen Ihrer Lieblings-Programmiersprache durch einen regulären Ausdruck. Dasselbe für die Kommentare in dieser Sprache.

   Lösung: Z.B.: $[a-zA-Z\_][a-zA-Z\_0-9]^*$

3. Welche Sprache beschreibt der reguläre Ausdruck $(\alpha^*\beta)^*$ über dem Alphabet $\{\,\alpha,\beta\,\}$. Können Sie einen einfacheren regulären Ausdruck für diese Sprache finden?

   Lösung: Die Sprache aller Wörter, deren letztes Symbol ein β ist. Alternativ: $(\alpha|\beta)^*(\epsilon|\beta)$.

4. Beweisen Sie (oder versuchen Sie zumindest möglichst plausibel zu rechtfertigen) die folgenden Rechengesetze für reguläre Ausdrücke:

$$\begin{aligned} r|s &= s|r \\ (r|s)|t &= r|(s|t) \\ (rs)t &= r(st) \\ r(s|t) &= rs|rt \\ (r|s)t &= rt|st \\ \emptyset^* &= \epsilon \\ (r^*)^* &= r^* \\ (\epsilon|r)^* &= r^* \\ (r^*s^*)^* &= (r|s)^* \end{aligned}$$

   Dabei seien zwei reguläre Ausdrücke als *gleich* betrachtet, wenn sie dieselbe Sprache beschreiben.

5. Welche der folgenden Gleichungen für reguläre Ausdrücke gelten allgemein?

$$\begin{aligned} (rs|r)^*r &= r(sr|r)^* \\ (r|s)^* &= r^*|s^* \end{aligned}$$

   Lösung: Wir zeigen zuerst mit Induktion, daß

$$(rs|r)^n r = r(sr|r)^n$$

   für alle $n \in \mathbb{N}$ gilt. Für $n = 0$ ist dies offensichtlich. Für den Induktionsschritt schließen wir: $(rs|r)^{n+1}r = (rs|r)(rs|r)^n r = r(s|\epsilon)r(sr|r)^n = r(sr|r)(sr|r)^n = r(sr|r)^{n+1}$.

   Damit ergibt sich $(rs|r)^*r = \bigcup_{n\in\mathbb{N}}(rs|r)^n r = \bigcup_{n\in\mathbb{N}} r(sr|r)^n = r(sr|r)^*$.

   Die zweite Gleichung ist dagegen falsch; z.B. entspricht rs dem linken, aber nicht dem rechten Ausdruck.

6. Suchen Sie sich ein Programm welches reguläre Ausdrücke verarbeiten kann (z.B. grep, perl, python, java-Bibliothek,. . . , oder einfach einen Texteditor, der nach regulären Ausdrücken suchen kann), und testen Sie damit die diversen regulären Ausdrücke dieses Übungsblattes anhand mehrerer Beispiele.