

Max Springenberg, 177792

1.1

1.1.1 Welche Sprache wird durch den erweiterten regulären Ausdruck $(ab^+)^+$ über dem Alphabet $\Sigma = \{a, b\}$ beschrieben? Wie lautet ein äquivalenter regulärer Ausdruck ohne die in der Vorlesung eingeführte erweiterte Syntax?

Der Ausdruck $(ab^+)^+$ beschreibt die Sprache $L((ab^+)^+)$

Die durch den regulären Ausdruck beschriebenen Regeln für L sind:

1. jedes Wort fängt mit einem a an
2. auf jedes a folgt mindestens ein b
3. es muss mindestens ein a und ein b aufeinander folgend vorkommen

Ein äquivalenter Ausdruck für $(ab^+)^+$ ohne die in der Vorlesung vorgestellte erweiterte Syntax wäre $abb^*(abb)^*$, da:

$$(abb^* \equiv ab^+) \wedge (v^+ \equiv vv^*, \text{ mit } v = abb^*)$$

1.1.2

(a) Konstruieren Sie einen regulären Ausdruck für die Menge L aller Wörter aus $\{0, 1\}$, die ihr erstes Zeichen genau zwei weitere Male, also insgesamt dreimal, enthalten.

Beispiel: $000 \in L, 0010 \in L, 0111010 \in L, 0010011 \in L, \epsilon \in L$.

Offensichtlich sind mögliche erste Zeichen 0 und 1 oder gar kein Zeichen.

Damit entspricht der reguläre Ausdruck a , der die Sprache entscheidet, der Form $a = (1v + 0u)^*$, mit:

$v \in \{0, 1\}$ als beliebiges Wort, das mindestens zwei Einsen und $u \in \{0, 1\}$ als beliebiges Wort, das mindestens zwei Nullen enthält.

Nun bleibt die Form der regulären Ausdrücke v und u zu klären.

Dem Beschriebenen kann entnommen werden, dass eine Analogie im Aufbau der regulären Ausdrücke, die u und v beschreiben existiert.

v kann mit 0 und 1 anfangen und vor der ersten 1 darf beliebig oft 0 vorkommen. Daraus folgt $v = 0^*1v'$, wobei v' wie auch v mit 0 und 1 anfangen kann und vor der ersten 1 darf beliebig oft 0 vorkommen. Ferner kann v mit einer beliebigen Folge aus $\{0, 1\}$ enden. Daraus folgt $v' = 0^*1(0^*1)^*$ und $v = 0^*1v' = 0^*10^*1(0^*1)^*$.

Da v und u analog aufgebaut sind kann u mit $u = 1^*0u' = 1^*01^*0(1^*0)^*$ gebildet werden.

Der reguläre Ausdruck a hat damit also die Form:

$$a = (1v + 0u)^* = (10^*10^*1(0^*1)^* + 01^*01^*0(1^*0)^*)^*$$

und es gilt $L = L(a)$

1.1.3

- (b) Passwort p ueber $\Sigma = \{a, \dots, z, A, \dots, Z, 1, \dots, 9\}$ mit Bedingungen:**
- (i) Auf keinen Kleinbuchstaben folgt direkt ein Grobuchstabe.**
 - (ii) Auf keinen Grobuchstaben folgt direkt ein Kleinbuchstabe.**
 - (iii) Es ist mindestens eine Ziffer enthalten.**
 - (iv) Das erste Zeichen ist keine Ziffer.**

Konvention: Zeichen aus Σ seien in drei Gruppen eingeteilt, sodass:

$$\Sigma \equiv KL \cup GR \cup NUM$$

, mit $KL = \{a, \dots, z\}, GR = \{A, \dots, Z\}, NUM = \{1, \dots, 9\}$

1. Sprache aller richtigen Passwoerter:

Aus (i) und (ii) folgt der Ausdruck $a = (k^+n^+g^+ + g^+k^+n^+ + n^+)^*$, mit $k \in KL, g \in GR, n \in NUM$

Aus (iii) und (iv) folgt der Ausdruck $b = (g^+ + k^+)nw$, mit $k \in KL, g \in GR, n \in NUM$ und w als regularen Ausdruck, der (i) und (ii) erfuehlt

Somit kann ein legitimes Passwort p durch den regulaeren Ausdruck:

$$p = ba = (g^+ + k^+)n(k^+n^+g^+ + g^+k^+n^+ + n^+)^*$$

formuliert werden.

2. Sprache aller falschen Passwoerter:

Fuer die Sprache aller falschen Passwoerter muessen wir alle Aussagen negieren und verodern, da bereits eine Regelverletzung zu einem unsicherem Passwort fuehrt.

- (i) es existiert ein Kleinbuchstaben, auf den direkt ein Grobuchstabe folgt.
- (ii) es existiert ein Grobuchstabe, auf den direkt ein Kleinbuchstabe folgt.
- (iii) Es ist keine Ziffer enthalten.
- (iv) Das erste Zeichen ist eine Ziffer.

Aus (i) folgt der Ausdruck $a_f = \Sigma^*g^+k^+\Sigma^*$, mit $k \in KL, g \in GR, n \in NUM$

Aus (ii) folgt der Ausdruck $b_f = \Sigma^*k^+g^+\Sigma^*$, mit $k \in KL, g \in GR, n \in NUM$

Aus (iii) folgt der Ausdruck $c_f = (g^*k^*)^*$, mit $k \in KL, g \in GR$

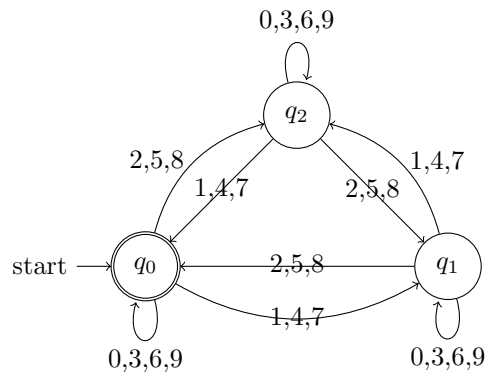
Aus (iv) folgt der Ausdruck $d_f = n\Sigma^*$, mit $n \in NUM$

daraus folgt der Ausdruck:

$$p_f = a_f + b_f + c_f + d_f = \Sigma^*g^+k^+\Sigma^* + \Sigma^*k^+g^+\Sigma^* + (g^*k^*)^* + n\Sigma^*$$

1.2

1.2.1 Gegeben sei der folgende DFA. Erläutern si die Funktion.



Der Automat berechnet mod_3 .

Summiert man die modulo-Werte der Ziffern einer Dezimalzahl auf und rechnet modulo auf dieser Summe erhaelt man das selbe ergebnis, wie wenn man modulo auf der gesamten Zahl rechnet.

Der Automat wechselt fuer jede Ziffer gemaess dem Modulo-Rest dieser Ziffer in den naechsten Zustand. Dabei wird fuer jede Ziffer x wie im Zustand q_i wie folgt vorgegangen:

- (i) gilt $x \equiv_3 0$ wird im Zustand verweilt
- (ii) gilt $x \equiv_3 1$ wird in den Zustand $q_{\text{mod}_3(i+1)}$ gewechselt
- (iii) gilt $x \equiv_3 2$ wird in den Zustand $q_{\text{mod}_3(i+2)}$ gewechselt

Folglich gilt fuer die TransitionsFunktion:

$$\delta(q_i, \sigma) = q_{\text{mod}_3(i + \text{mod}_3(\sigma))}$$

Es wird nur im Startzustand q_0 akzeptiert. Damit berechnet der Automat mod_3 korrekt.

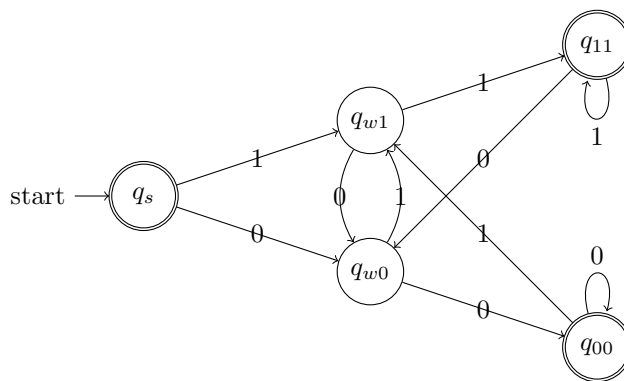
1.2.2

(a) beschreiben Sie die von dem Automaten beschriebene Sprache umgangssprachlich in 1-2 Sätzen.

Der Automat entscheidet Die Sprache L ueber das Alphabet $\Sigma = \{0, 1\}$, fuer die alle Woerter entweder mit einer 0 beginnen oder eine 0 an einer Stelle $i \in \{2 * k | k \in \mathbb{N}\}$ enthalten.

Konstruieren sie einen DFA fuer die Sprache

$$L = \{w \in \{0, 1\}^* \mid w \text{ endet auf zwei gleiche Zeichen}\}$$



(c) Konstruieren sie einen DFA fuer die Sprache

$$L = \{w \in \{0, 1\}^* \mid \text{Fuer jedes Praefix } p \text{ von } w \text{ gilt: } |\#_0(p) - \#_1(p)| \leq 2\}$$

