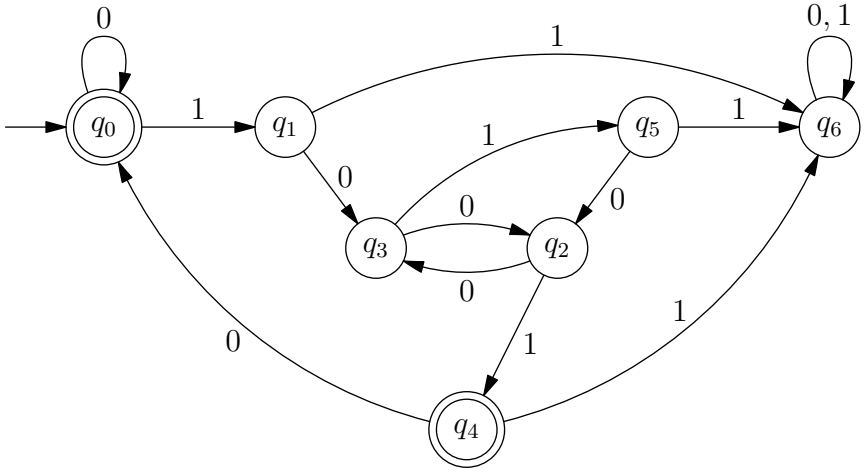
Übung zur Vorlesung Formale Sprachen, Automaten und Prozesse

Aufgabe T4

Werten Sie formal den Lauf des folgenden DFA auf dem Wort $w = 1001$ aus, d.h. bestimmen Sie $\hat{\delta}(q_0, w)$ indem Sie rekursiv die Definition von $\hat{\delta}$ verwenden.



Überlegen Sie zuerst, wie ein Wort aussehen muß, damit der Automat es akzeptiert. Beschreiben Sie dann umgangssprachlich die Sprache, die der Automat erkennt.

Lösungsvorschlag:

Es ist

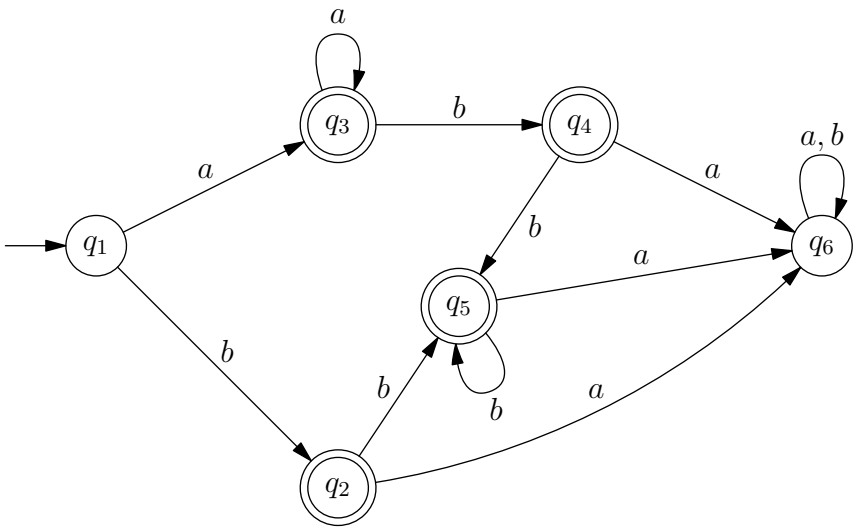
$$\begin{aligned}
\hat{\delta}(q_0, w) &= \hat{\delta}(q_0, 1001) \\
&= \delta(\hat{\delta}(q_0, 100), 1) \\
&= \delta(\delta(\hat{\delta}(q_0, 10), 0), 1) \\
&= \delta(\delta(\delta(\hat{\delta}(q_0, 1), 0), 0), 1) \\
&= \delta(\delta(\delta(\delta(q_0, 1), 0), 0), 1) \\
&= \delta(\delta(\delta(q_1, 0), 0), 1) \\
&= \delta(\delta(q_3, 0), 1) \\
&= \delta(q_2, 1) \\
&= q_4 \in F
\end{aligned}$$

Das Wort wird akzeptiert.

Der Automat erkennt genau die Wörter die 11 nicht als Infix enthalten und durch drei teilbar sind.

Aufgabe T5

Gegeben sei folgender DFA. Welche Sprache erkennt er?



Lösungsvorschlag:

Er erkennt die Sprache $a^*(a+b)b^*$.

Aufgabe T6

Wir nehmen an, daß die Sprache $L = \{\, a^n b^m \mid m \geq n \geq 0 \,\}$ nicht regulär ist. Zeigen Sie, daß unter dieser Annahme auch die folgenden Sprachen nicht regulär sind, indem Sie Abschlußeigenschaften regulärer Sprachen verwenden.

1. $L_1 = \{\, b^n a^m \mid m \geq n \geq 0 \,\}$
2. $L_2 = \{\, a^n b^n \mid n \geq 0 \,\}$
3. $L_3 = \{\, a^n b^n c^n \mid n \geq 0 \,\}$
4. $L_4 = \{\, a^m b^n \mid m \geq n \geq 0 \,\}$
5. $L_5 = \{\, a^{2n} b^n \mid n \geq 0 \,\}$

Lösungsvorschlag:

Im folgenden sei $h\colon \Sigma \to \Sigma^*$ eine Umbenennung von Buchstaben in Wörter. Die Erweiterung von h über Wörtern $h : \Sigma^* \to \Sigma^*$, $a \in \Sigma, w \in \Sigma^*$ mit $h(\epsilon) := \epsilon$ und $h(aw) := h(a)h(w)$ ist dann ein Homomorphismus. Über einer Sprache L sei h definiert als $h(L) := \{\, h(w) \mid w \in L \,\}$.

1. Angenommen L_1 sei regulär. Sei h definiert durch $h(a) := b$ und $h(b) := a$. Dann ist $h(L_1) = \{\, a^n b^m \mid m \geq n \geq 0 \,\} = L$. Widerspruch, da reguläre Sprachen unter Homomorphismen abgeschlossen sind.
2. Angenommen L_2 sei regulär. Dann ist auch $L_2 b^* = L$ regulär, da reguläre Sprachen unter Konkatenation abgeschlossen sind. Widerspruch.

3. Sei $h(a) := a$, $h(b) := b$ und $h(c) := \epsilon$. Dann gilt $h(L_3) = \{ a^n b^n \mid n \geq 0 \} = L_2$. Widerspruch.

4. Durch Induktion über den Aufbau regulärer Ausdrücke läßt sich sofort zeigen, daß reguläre Sprachen unter $\cdot^R$ abgeschlossen sind. Es gilt aber $L_4^R = \{ b^n a^m \mid m \geq n \geq 0 \} = L_1$. Widerspruch.

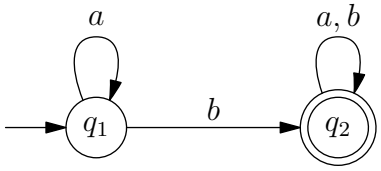
5. Sei $h(a) = a$ und $h(b) = bb$. Dann gilt:

   - $h(L_5) = \{ a^{2n} b^{2n} \mid n \geq 0 \}$
   - $ah(L_5)b = \{ a^{2n+1} b^{2n+1} \mid n \geq 0 \}$

   Somit ist $ah(L_5)b \cup h(L_5) = L_2$. Widerspruch.

Aufgabe T7

Gegeben sei folgender sehr einfacher DFA M. Beginnen Sie, einen regulären Ausdruck r mit der L_{ij}^k-Konstruktion zu konstruieren, der $L(M) = L(r)$ erfüllt. Machen Sie damit weiter, bis es entweder zu langweilig wird oder Sie tatsächlich fertig werden.



Lösungsvorschlag:

Das passiert, wenn man dem Algorithmus stur folgt. Die Ausdrücke werden (unnötigerweise) sehr lang.

- $R_{1,1}^0 = \epsilon + a$
- $R_{1,2}^0 = b$
- $R_{2,2}^0 = \epsilon + a + b$
- $R_{2,1}^0 = \emptyset$
- $R_{1,1}^1 = R_{1,1}^0 + R_{1,1}^0 (R_{1,1}^0)^* R_{1,1}^0 = (\epsilon + a) + (\epsilon + a)((\epsilon + a))^* (\epsilon + a)$
- $R_{1,2}^1 = R_{1,2}^0 + R_{1,1}^0 (R_{1,1}^0)^* R_{1,2}^0 = b + (\epsilon + a)((\epsilon + a))^* b$
- $R_{2,2}^1 = R_{2,2}^0 + R_{2,1}^0 (R_{1,1}^0)^* R_{1,2}^0 = (\epsilon + a + b) + \emptyset((\epsilon + a))^* b$
- $R_{2,1}^1 = R_{2,1}^0 + R_{2,1}^0 (R_{1,1}^0)^* R_{1,1}^0 = \emptyset + \emptyset((\epsilon + a))^* (\epsilon + a)$
- $R_{1,1}^2 = R_{1,1}^1 + R_{1,2}^1 (R_{2,2}^1)^* R_{2,1}^1 = (\epsilon + a) + (\epsilon + a)((\epsilon + a))^* (\epsilon + a) + b + (\epsilon + a)((\epsilon + a))^* b((\epsilon + a + b) + \emptyset((\epsilon + a))^* b)^* \emptyset + \emptyset((\epsilon + a))^* (\epsilon + a)$
- $R_{1,2}^2 = R_{1,2}^1 + R_{1,2}^1 (R_{2,2}^1)^* R_{2,2}^1 = b + (\epsilon + a)((\epsilon + a))^* b + b + (\epsilon + a)((\epsilon + a))^* b((\epsilon + a + b) + \emptyset((\epsilon + a))^* b)^* (\epsilon + a + b) + \emptyset((\epsilon + a))^* b$
- $R_{2,2}^2 = R_{2,2}^1 + R_{2,2}^1 (R_{2,2}^1)^* R_{2,2}^1 = (\epsilon + a + b) + \emptyset((\epsilon + a))^* b + (\epsilon + a + b) + \emptyset((\epsilon + a))^* b((\epsilon + a + b) + \emptyset((\epsilon + a))^* b)^* (\epsilon + a + b) + \emptyset((\epsilon + a))^* b$

- $R^2_{2,1} = R^1_{2,1} + R^1_{2,2}(R^1_{2,2})^* R^1_{2,1} = (\epsilon + a + b) + \emptyset((\epsilon + a))^* b + (\epsilon + a + b) + \emptyset((\epsilon + a))^* b((\epsilon + a + b) + \emptyset((\epsilon + a))^* b)^* \emptyset + \emptyset((\epsilon + a))^*(\epsilon + a)$

$$R = \sum_{q_j \in F} R^n_{1,j} = R^2_{1,2} = b + (\epsilon + a)((\epsilon + a))^* b$$
$$+ b + (\epsilon + a)((\epsilon + a))^* b((\epsilon + a + b) + \emptyset((\epsilon + a))^* b)^*(\epsilon + a + b) + \emptyset((\epsilon + a))^* b$$

Aufgabe H3 (3+5+5 Punkte)
Wir betrachten einen Aufzug, der über folgende Abläufe verfügt: R (Notruf auslösen), A (Türe öffnet sich), Z (Türe schließt sich), U (Aufzug fährt einen Stock nach oben) und D (Aufzug fährt einen Stock nach unten). Es gibt vier Etagen und zu Beginn steht der Aufzug mit geschlossener Tür im Erdgeschoss.

a) Geben sie eine informelle Beschreibung der (sinnvollen) möglichen Abläufe, wie Sie sie z.B. als Spezifikation an einen Hersteller geben würden.

b) Geben Sie die Spezifikation aus a) als DFA an.

c) Auf der Webseite der Vorlesung[1] finden Sie neben diesem Übungsblatt eine Datei, die pro Zeile einen Ablauf enthält. Welche Abläufe widersprechen Ihren Spezifikationen und warum? Hinweis: Die Abläufe sind wohl zu lang, um sie von Hand zu überprüfen.

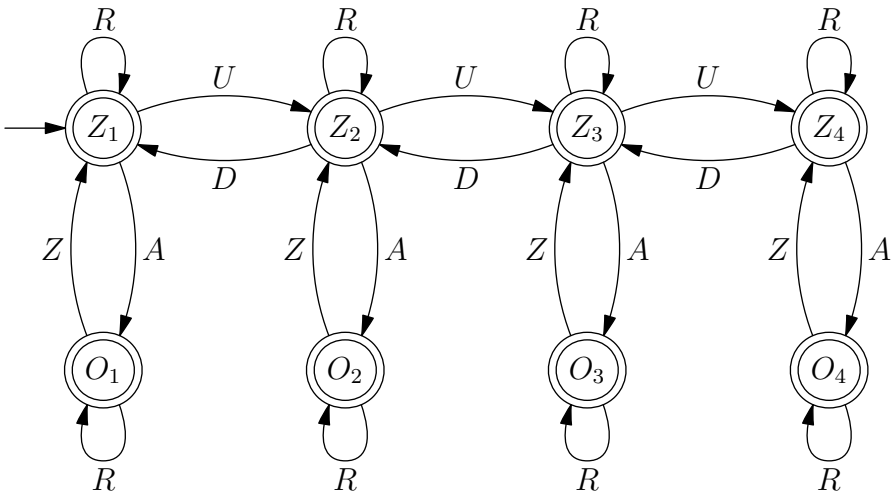
   Freiwillig: Wir haben auch eine zweite, komprimierte Datei hinterlegt, welche sehr lang ist und ein weiteres Beispiel enthält. Wenn Sie wollen, können Sie auch diese bearbeiten und aufschreiben, wie lange Ihr Programm dafür benötigte.

Lösungsvorschlag:

a) Der Aufzug sollte nur mit geschlossener Türe fahren können. Im untersten Stockwerk, darf man nicht nach unten fahren und im obersten nicht nach oben. Ein Notruf sollte jederzeit möglich sein.

b) In folgender Zeichnung fehlt der *Fangzustand*, zu welchen alle Transitionen führen, die nicht eingezeichnet sind. Zum Beispiel fehlt von Z_4 ein abgehender Pfeil, der mit U beschriftet ist. Dieser muß natürlich vorhanden sein und führt hier zum Fangzustand. Der Zustand O_i modelliert die Situation, daß der Aufzug im $i$ten Stockwerk mit offener Türe steht und Z_i mit geschlossener.

[1] `http://tcs.rwth-aachen.de/lehre/FSAP/SS2017/`



c) In der Sprache Python, die man mögen kann oder auch nicht, haben wir eine Lösung in Abbildung 1 erstellt.

Lassen wir dieses Programm laufen, erhalten wir diese Ausgabe. Die mittleren zwei Spezifikationen sind also in Ordnung.

```
Die Tuer ist schon zu und wir sollen sie schliessen.
okay
okay
Oje, wir fahren durch das Dach.
```

Alternativ koennen wir auch den Automaten explizit konstruieren und simulieren anstatt wie oben ihn nur implizit zu simulieren. In Abbildung 2 findet sich ein Beispielprogramm in C.

Aufgabe H4 (10 Punkte)

Sei $L \subseteq \Sigma^*$ eine Sprache und $a \in \Sigma$. Der deterministische endliche Automat A erkenne die Sprache La. Beweisen Sie, daß dann ein DFA B existiert, der die Sprache L erkennt.

Lösungsvorschlag:

Sei $M := (Q, \Sigma, \delta, q_0, F)$ ein deterministischer endlicher Automat, der die Sprache La erkennt. Sei $F' := \{\, q \in Q \mid \delta(q, a) \in F \,\}$. Dann erkennt der DFA $M' := (Q, \Sigma, \delta, q_0, F')$ die Sprache L.

Beweis: Sei $wa \in L(M) = La$. Dann gilt $\hat{\delta}(q_0, wa) \in F$. Somit gilt $\delta(\hat{\delta}(q_0, w), a) \in F$ und $\delta(q_0, w) \in F'$. Somit akzeptiert M' das Wort w.

Andererseits, sei $wa \notin L(M) = La$. Dann gilt $\hat{\delta}(q_0, wa) \notin F$ und analog zu oben $\hat{\delta}(q_0, w) \notin F'$.

```
import sys

def check(line):
  dooropen = False
  floor = 1
  for s in line:
    if s == 'R':
      pass
    if s == 'U':
      floor = floor + 1
      if dooropen:
        return"Mist, wir fahren nach oben, aber die Tuer ist ja offen."
      if floor > 4:
        return"Oje, wir fahren durch das Dach."
    if s == 'D':
      floor = floor - 1
      if dooropen:
        return"Caramba, wir fahren nach unten, aber die Tuer ist ja offen."
      if floor < 1:
        return"Semprini, wir fahren in den verbotenen Keller."
    if s == 'A':
      if dooropen:
        return"Die Tuer ist schon offen und wir sollen sie oeffnen."
      dooropen = True
    if s == 'Z':
      if not dooropen:
        return"Die Tuer ist schon zu und wir sollen sie schliessen."
      dooropen = False
  return"okay"

for line in sys.stdin:
  print check(line)
```

Abbildung 1: Augzuglösung in Python programmiert.

```
#include ⟨stdio.h⟩
#include ⟨unistd.h⟩

int delta[9][128];
int z[] = { 0, 1, 2, 3};
int o[] = { 4, 5, 6, 7};
int fang = 8;

void main() {
  // create dfa table
  int i, q, c;
  for(q = 0; q < 9; q++) {
    for(c = 0; c < 128; c++) {
      delta[q][c] = fang;
    }
  }
  for(i = 0; i < 4; i++) {
    delta[z[i]]['A'] = o[i];
    delta[o[i]]['Z'] = z[i];
    delta[z[i]]['R'] = z[i];
    delta[o[i]]['R'] = o[i];
    if(i < 3) delta[z[i]]['U'] = z[i + 1];
    if(i > 0) delta[z[i]]['D'] = z[i - 1];
  }
  // simulate dfa on each word of stdin
  q = z[0];
  char buffer[4096];
  int endofbuf = 0;
  int n = 0;
  for(; ; ) {
    if(n == endofbuf) {
      endofbuf = read(0, buffer, 4096);
      if(endofbuf == 0) break;
      n = 0;
    }
    c = buffer[n++];
    if(c == '\n') {
      printf(q == fang ? "nicht okay\n" : "okay\n");
      q = z[0];
      continue;
    }
    q = delta[q][c];
  }
}
```

Abbildung 2: Aufzugaufgabe in C gelöst