Übungsblatt 3 - mit Lösungen

Reguläre Sprachen und endliche Automaten

{Theoretische Informatik}@AIN3
Prof. Dr. Barbara Staehle
Wintersemester 2021/2022
HTWG Konstanz

Aufgabe 3.1 Reguläre Ausdrücke, Grammatiken und endliche Automaten

In dieser Aufgabe geht es um reguläre Ausdrücke, Grammatiken und endliche Automaten r_x, G_x, A_x welche jeweils die formale Sprache L_x erzeugen bzw. akzeptieren, wobei $x \in \{1, 2, \ldots, 12\}$.

Teilaufgabe 3.1.1 3 Punkte

Geben Sie alle Worte an, welche durch folgende reguläre Ausdrücke erzeugt werden (jeweils über einem geeigneten Terminalalphabet, das Sie nicht angeben müssen):

a) r_1 = (a|b)(a|b)

b) r_2 =a(a|b)|b(a|b)

c) r_3 =(ab*)*

d) r_4 =(aa|b)*

e) r_5 =(D|d)(er|ie|as)

f) r_6 =(+|-)?[0-9]+

g) r_7 =[0-9A-F]+

Lösung

a) L_1 = {aa, ab, ba, bb}

b) L_2 = {aa, ab, ba, bb}

c) $L_3 = \{\varepsilon \cup \{aw \mid w \in \{a, b\}^*\}$ (leeres Wort, oder beliebige a,b-Kombination mit führendem a)

d) $L_4 = \{w \in a, b^* \mid a$ kommt in w nur in Blöcken gerader Länge vor$\}$ (z.B. aa, baab, bbaaaa, ...)

e) L_5 enthält alle groß oder klein geschriebener Artikel (der, die, das, Der, Die, Das)

f) L_6 enthält alle ganzen Zahlen (eventuell mit führenden Nullen) mit oder ohne Vorzeichen

g) L_7 enthält alle Hexadezimalzahlen (eventuell mit führenden Nullen)

Teilaufgabe 3.1.2 3 Punkte

Geben Sie die regulären Ausdrücke an, welche die folgenden formalen Sprachen erzeugen (jeweils über einem geeigneten Terminalalphabet, das Sie nicht angeben müssen):

a) L_8 = {Meier, Meir, Meyer, Meyr, Maier, Mair, Mayer, Mayr}

b) L_9 = {1€, 10€, 100€, 1000€, ...}

c) $L_{10} = \{a^n b^m \mid n, m \in \mathbb{N}_0\}$

d) $L_{11} = \{a^n b^m \mid n, m \in \mathbb{N}\}$

e) $L_{12} = \{(ab)^n \mid n \in \mathbb{N}\}$

f) $L_{13} = \{a, b\}^*$

g) $L_{14} = \{a^n \mid n \in \mathbb{N}_0\} \cup \{b^n \mid n \in \mathbb{N}_0\}$

h) $L_{15} = \{a^n \mid n \in \mathbb{N}\} \cup \{b^n \mid n \in \mathbb{N}\}$

LÖSUNG

a) $r_8 =$ M(e|a)(i|y)(e|ε)r

b) $r_9 =$ 10*€

c) $r_{10} =$ a*b*

d) $r_{11} =$ aa*bb*

e) $r_{12} =$ ab(ab)*

f) $r_{13} =$ (a|b)*

g) $r_{14} =$ a*|b*

h) $r_{15} =$ (aa*)|(bb*)

TEILAUFGABE 3.1.3 3 PUNKTE

Geben Sie die **regulären** Grammatiken an, welche die folgenden Sprachen erzeugen:

a) G_1 mit $\mathscr{L}(G_1) = L_1$

b) G_3 mit $\mathscr{L}(G_3) = L_3$

c) G_4 mit $\mathscr{L}(G_4) = L_4$

d) G_{12} mit $\mathscr{L}(G_{12}) = L_{12}$

LÖSUNG

a) $G_1 = (N, \Sigma, P, S) = (\{S, A\}, \{a, b\}, P, S)$ und P :

- $S \rightarrow aA \mid bA$
- $A \rightarrow a \mid b$

b) $G_3 = (N, \Sigma, P, S) = (\{S, B\}, \{a, b\}, P, S)$ und P :

- $S \rightarrow \varepsilon \mid aB$
- $B \rightarrow \varepsilon \mid aB \mid bB$

c) $G_4 = (N, \Sigma, P, S) = (\{S, A\}, \{a, b\}, P, S)$ und P :

- $S \rightarrow \varepsilon \mid aA \mid bS$
- $A \rightarrow a \mid aS$

d) $G_{12} = (N, \Sigma, P, S) = (\{S, B\}, \{a, b\}, P, S)$ und P :

- $S \rightarrow aB$
- $B \rightarrow bS \mid b$

Teilaufgabe 3.1.4 4 Punkte

Geben Sie die **regulären** Grammatiken an, welche die folgenden Sprachen erzeugen:

a) G_{11} mit $\mathscr{L}(G_{11}) = L_{11}$

b) G_{10} mit $\mathscr{L}(G_{10}) = L_{10}$

c) G_7 mit $\mathscr{L}(G_7) = L_7$

d) G_6 mit $\mathscr{L}(G_6) = L_6$

Lösung

a) $G_{11} = (N, \Sigma, P, S) = (\{S, B\}, \{a, b\}, P, S)$ und P :
   - $S \to aS \mid aB$
   - $B \to b \mid bB$

b) $G_{10} = (N, \Sigma, P, S) = (\{S, B\}, \{a, b\}, P, S)$ und P :
   - $S \to \varepsilon \mid aS \mid bB$
   - $B \to \varepsilon \mid bB$

c) $G_7 = (N, \Sigma, P, S) = (\{S, Z\}, \{0, 1, \ldots, 9, A, B \ldots F\}, P, S)$ und P :
   - $S \to 0Z \mid 1Z \mid \ldots \mid FZ$
   - $Z \to \varepsilon \mid 0Z \mid 1Z \mid \ldots \mid FZ$

d) $G_6 = (N, \Sigma, P, S) = (\{S, A, B\}, \{0, 1, \ldots, 9\}, P, S)$ und P :
   - $S \to +A \mid -A \mid 0B \mid 1B \mid \ldots \mid 9B$
   - $A \to 0B \mid 1B \mid \ldots \mid 9B$
   - $B \to \varepsilon \mid 0B \mid 1B \mid \ldots \mid 9B$

Aufgabe 3.2 Reguläre Ausdrücke für Datentypen

Folgende Liste regulärer Ausdrücke beschreibt die für eine Programmiersprache verwendeten Datentypen.

a) `[0]|[-+]?[1-9][0-9]*`

b) `\+?[1-9][0-9]*`

c) `[A-Za-z0-9\-\.]{1,64}`

d) `[^\s]+([\s]?[^\s]+)*`

e) `-?[0-9]{4}(-(0[1-9]|1[0-2])(-(0[0-9]|[1-2][0-9]|3[0-1]))?)?`

f) `([01][0-9]|2[0-3]):[0-5][0-9]:[0-5][0-9](\.[0-9]+)?`

Teilaufgabe 3.2.1 2 Punkte

Geben Sie für **jeden** der regulären Ausdrücke a)-c) jeweils **3** Beispiele für Worte an, welche durch den regulären Ausdruck beschrieben bzw. nicht beschrieben werden.

LÖSUNG

Quelle: FHIR Datatypes, http://hl7.org/fhir/datatypes.html

a) **integer** `[0]|[-+]?[1-9][0-9]*`

   geht 0, 897, 980654, +55, -8

   geht nicht 007, 879.65, -0

b) **positiveInt** `\+?[1-9][0-9]*`

   geht 0, 897, 980654, +55

   geht nicht 007, 879.65, -0, -8

c) **id** `[A-Za-z0-9\-\.]{1,64}`

   geht aBc-12-h-3o-99.45, AHA, 007.0815

   geht nicht abc&123, !!#!!, %78476%

TEILAUFGABE 3.2.2 3 PUNKTE

Geben Sie für **jeden** der regulären Ausdrücke d)-f) jeweils **3** Beispiele für Worte an, welche durch den regulären Ausdruck beschrieben bzw. nicht beschrieben werden.

LÖSUNG

Quelle: FHIR Datatypes, http://hl7.org/fhir/datatypes.html

d) **code** `[^\s]+([\s]?[^\s]+)*`

   geht abv er4, XZy 8 765c

   geht nicht 7890 8, , !!111! (führendes&alleiniges Leerzeichen/Tabulator/Zeilenumbruch) geht nicht, sonst ist alles erlaubt

e) **date** `-?[0-9]{4}(-(0[1-9]|1[0-2])(-(0[0-9]|[1-2][0-9]|3[0-1]))?)?`

   geht -1045, -1045-12-13, 2017-05-31, 2018-02-31

   geht nicht -1045-13-12, 201789, 100-12-12

f) **time** `([01][0-9]|2[0-3]):[0-5][0-9]:[0-5][0-9](\.[0-9]+)?`

   geht 12:12:12, 23:59:59, 23:59:59.99999

   geht nicht 23, 42:00:00, 23:59, 23:59:67.99999

TEILAUFGABE 3.2.3 2 PUNKTE

Beschreiben Sie die für jeden Datentyp zulässigen Eingaben mit Ihren eigenen Worten und geben Sie an, was dieser darstellen könnte.

LÖSUNG

- ganze Zahl: Vorzeichen kann, muss nicht sein, darf nicht mit null starten
- natürliche Zahl: Vorzeichen kann, muss nicht sein, darf nicht mit null starten, nur + als Vorzeichen erlaubt
- ID: Buchstaben, Ziffern, - oder ., maximal 64 Stellen lang
- Code: (führendes&alleiniges Leerzeichen/Tabulator/Zeilenumbruch) geht nicht, sonst ist alles erlaubt

- Datum: US-Format: vierstelliges Jahr muss sein, Monat, Tag getrennt mit Bindestrichen können sein
- Uhrzeit: HH:MM:SS Kommastellen der Sekunden können beliebig genau sein oder weggelassen werden.

Aufgabe 3.3 The Hound of the Baskervilles

Um diese Aufgabe lösen zu können, verwenden Sie die Datei `ACDoyle_Hound-of-the-Baskervilles.txt` welche in Moodle zur Verfügung steht.

Außerdem benötigen Sie eine Linux/Unix-Shell, Cygwin unter Windows oder eine andere Windows-Grep-Lösung, einen Texteditor oder ein Online-Tool wie z.B. RegExr

Hinweise:

- Zählen Sie die Anzahl Ihrer Resultate (z.B. via Option -c) und sehen Sie sich diese an.
- Die Musterlösung geben eine mögliche Lösung an, andere Möglichkeiten gibt es sicher auch!

Teilaufgabe 3.3.1 3 Punkte

Wie oft werden die Helden (Sherlock Holmes und Dr. John H. Watson) im Dokument jeweils erwähnt? Konkret:

a) Wie oft korrekt angesprochen (Titel bzw. Mr.)?

b) Wie oft nur mit dem Nachnamen?

c) Wie oft nur nur mit dem Vornamen?

Lösung

Musterlösung ist work in progress! Ergebnisse in Notepad++ und egrep unterschiedlich!

a)
- `Mr. (Sherlock )?Holmes` ⇒ 37 Mal
- `(Dr.|Mr.) (John )?(H. )?Watson` ⇒ 29 Mal (immer nur Dr. Watson, Ehre wem Ehre gebührt)

b) mit diesen Ausdrücken bin ich nicht glücklich, es fehlen jeweils ein paar Erwähnungen …

- `[^oM][^cr][^k\.] Holmes` ⇒ 93 Mal
- `[^D][^r][^\.] Watson` ⇒ 73 Mal

c)
- `Sherlock\W[^H]` ⇒ 0: Holmes wird nie nur mit dem Vornamen angesprochen
- `John\W[^H]` ⇒ 6: das ist aber nicht John Watson, sondern (IMHO eine Nebenfigur), also wird Watson auch nie nur mit dem Vornamen angesprochen.

Teilaufgabe 3.3.2 3 Punkte

Dann noch ein paar Statistiken. Finden Sie jeweils die Anzahl der

a) direkten Reden

b) Buchstaben

c) Zahlen (die aus einer oder mehreren Ziffern bestehen)

im Dokument.

Lösung

Musterlösung ist work in progress! Ergebnisse in Notepad++ und egrep unterschiedlich!

a) `"[^"]*"` ⇒ 1321

b) `[a-zA-Z]` ⇒ 239842

c) `[0-9]*` ⇒ 38

Teilaufgabe 3.3.3 2 Punkte

Finden Sie mit Hilfe eines **hübschen, bisher noch nicht gefragten** regulären Ausdrucks Ihrer Wahl noch etwas spannendes heraus!

Lösung

Keine Musterlösung verfügbar.

Aufgabe 3.4 DEAs und NEAs

Wie betrachten im folgenden die Automaten A_2,A_3,A_4. In Abbildung 1 ist jeweils ihr Zustandsübergangsdiagramm dargestellt.

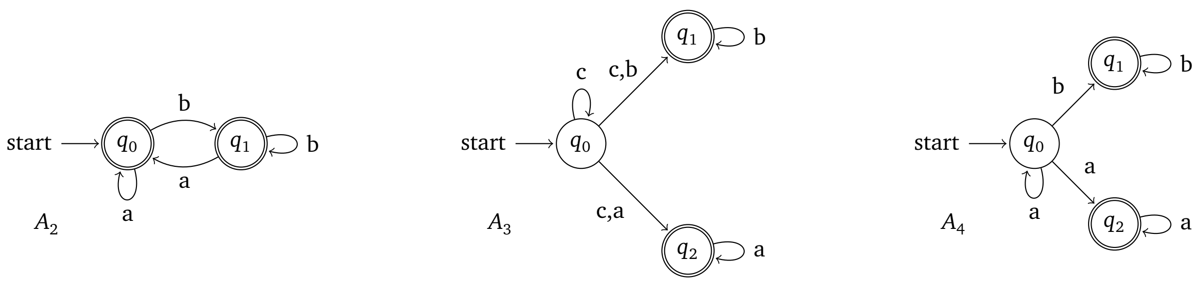


Abbildung 1: Zustandsübergangsdiagramme von A_2,A_3,A_4

Teilaufgabe 3.4.1 1 Punkt

Geben Sie für jeden der Automaten an, ob er ein deterministischer endliche Akzeptor, oder ein nichtdeterministischer endlicher Akzeptor ist. Begründen Sie Ihre Meinung.

Lösung

- A_2 ist ein DEA, da für jeden Zustand und jedes gelesene Zeichen maximal ein möglicher nächster Zustand definiert ist.
- A_3 und A_4 sind NEAs, da jeweils für den Zustand q_0 und das gelesene Zeichen a bzw. c zwei bzw. drei mögliche Nachfolgezustände definiert sind.

Teilaufgabe 3.4.2 2 Punkte

Geben Sie für jeden der Automaten

a) das Eingabealphabet

b) die Zustandsmenge,

c) die Finalmenge,

d) die Zustandsübergangsfunktion in tabellarischer Form an.

LÖSUNG

a) das Eingabealphabet, die Zustandsmenge und Finalmenge

- A_2: $\Sigma = \{a, b\}, Q = \{q_0, q_1\}, F = \{q_0, q_1\}$
- A_3: $\Sigma = \{a, b, c\}, Q = \{q_0, q_1, q_2\}, F = \{q_1, q_2\}$
- A_3: $\Sigma = \{a, b\}, Q = \{q_0, q_1, q_2\}, F = \{q_1, q_2\}$

b) die Zustandsübergangsfunktion in tabellarischer Form.

- A_2:

S/Σ	a	b
q_0	q_0	q_1
q_1	q_0	q_1

- A_3:

S/Σ	a	b	c
q_0	$\{q_2\}$	$\{q_1\}$	$\{q_0, q_1, q_2\}$
q_1	$\{\}$	$\{q_1\}$	$\{\}$
q_2	$\{q_2\}$	$\{\}$	$\{\}$

- A_4:

S/Σ	a	b
q_0	$\{q_0, q_2\}$	$\{q_1\}$
q_1	$\{\}$	$\{q_1\}$
q_2	$\{q_2\}$	$\{\}$

TEILAUFGABE 3.4.3 2 PUNKTE

Geben Sie für jeden Automaten alle Zustände an, welche bei der Verarbeitung der Worte bbb und aab durchlaufen werden. Entscheiden Sie, ob die Worte akzeptiert werden oder nicht.

Welche Methode Sie zur Lösung dieser Aufgabe verwenden, ist egal. Achten Sie jedoch darauf, dass Sie es geeignet darstellen, falls sich der Automat in mehreren Zuständen gleichzeitig befindet.

LÖSUNG

a)
- Notation mit $\hat{\delta}$:
  - A_2: $\hat{\delta}(q_0, bbb) = \hat{\delta}(\delta(q_0, b), bb)) = \hat{\delta}(q_1, bb) = \hat{\delta}(q_1, b) = \delta(q_1, b) = q_1$.
  - A_3: $\hat{\delta}(q_0, bbb) = \hat{\delta}(\delta(q_0, b), bb)) = \hat{\delta}(q_1, bb) = \hat{\delta}(q_1, b) = \delta(q_1, b) = q_1$.
  - A_4: $\hat{\delta}(q_0, bbb) = \hat{\delta}(\delta(q_0, b), bb)) = \hat{\delta}(q_1, bb) = \hat{\delta}(q_1, b) = \delta(q_1, b) = q_1$.
- Notation mit Pfeilen:
  - A_2: $q_0 \xrightarrow{b} q_1 \xrightarrow{b} q_1 \xrightarrow{b} q_1$.
  - A_3: $q_0 \xrightarrow{b} q_1 \xrightarrow{b} q_1 \xrightarrow{b} q_1$.
  - A_4: $q_0 \xrightarrow{b} q_0 \xrightarrow{b} q_0 \xrightarrow{b} q_0$.

b) Notation mit Pfeilen:

- A_2: $q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_0 \xrightarrow{b} q_1$.
- A_3: $q_0 \xrightarrow{a} q_0 \xrightarrow{a} q_1 \xrightarrow{b} ↯$.
- A_4

$$
\begin{tikzcd}
 & & q_2 \arrow[r, dash, "b"] & \text{stop} \\
 & q_0 \arrow[ur, dash, "a"] \arrow[dr, dash, "a"'] & & \\
 & & q_0 \arrow[r, dash, "b"] & q_1 \\
q_0 \arrow[uur, dash, "a"] \arrow[dr, dash, "a"'] & & & \\
 & q_2 \arrow[r, dash, "a"] & q_2 \arrow[r, dash, "b"] & \text{stop}
\end{tikzcd}
$$

Teilaufgabe 3.4.4 2 Punkte

Geben Sie für jeden der Automaten (wenn möglich) 3, zu den Wörtern aus Aufgabe 3.4.3 verschiedene Worte über dem Alphabet $\Sigma = \{a, b, c\}$ an

a) die akzeptiert werden,

b) die nicht akzeptiert werden.

Benutzen Sie Ihre Ergebnisse, um für jeden Automaten dessen akzeptierte Sprache anzugeben.

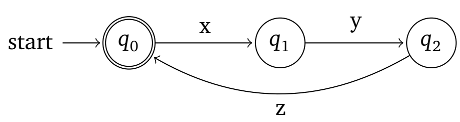
Lösung

- A_2: alle Worte werden akzeptiert
  nicht akzeptiert: a, b, ε
- A_3: akzeptiert: $b, a, bb, cbb, caa, cccca, aacbbcaa$
  nicht akzeptiert: ε, ba, ab, ccc
- A_4: akzeptiert: b, a, ab
  nicht akzeptiert: ε, ba, bab
- $\mathscr{L}(A_2) = \Sigma^*$
- $\mathscr{L}(A_3) = \{c^m \sigma^n | m \in \mathbb{N}_0, n \in \mathbb{N}, \sigma \in \{a, b\}\} \cup \{c\}$
- $\mathscr{L}(A_4) = \{a^n b^m | n, m \in \mathbb{N}_0\} \setminus \varepsilon$

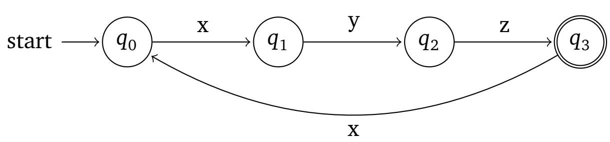
Aufgabe 3.5 Sprache von endlichen Automaten, 3 Punkte

Gegeben Sei das Alphabet $\Sigma = \{x, y, z\}$. Gegeben seinen außerdem die folgenden Automaten:

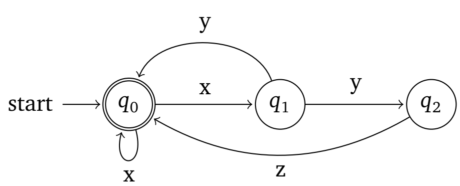
- $A_1 = (Q, \Sigma, \delta_1, F, q_0)$ mit $Q = \{q_0, q_1, q_2\}, F = \{q_0\}$ und δ_1 gegeben durch:



- $A_2 = (Q, \Sigma, \delta_2, F, q_0)$ mit $Q = \{q_0, q_1, q_2, q_3\}, F = \{q_3\}$ und δ_2 gegeben durch:



- $N_3 = (Q, \Sigma, \delta_3, F, q_0)$ mit $Q = \{q_0, q_1, q_2\}, F = \{q_0\}$ und δ_3 gegeben durch:



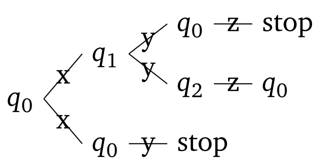
TEILAUFGABE 3.5.1 2 PUNKTE

Betrachten Sie die Worte $\omega_1 = xyz$ und $\omega_2 = xyxyz$. Geben Sie für alle Automaten alle Zustände an, die bei der Verarbeitung der Worte jeweils durchlaufen werden. Entscheiden Sie anschließend, ob die Wort akzeptiert werden oder nicht.

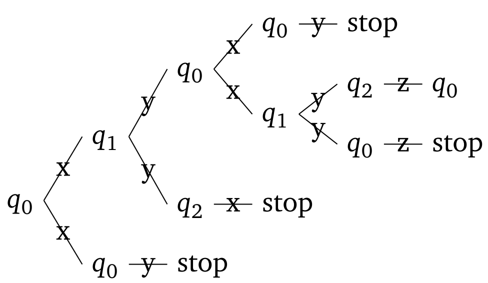
Welche Methode Sie zur Lösung dieser Aufgabe verwenden, ist egal. Achten Sie jedoch darauf, dass Sie es geeignet darstellen, falls sich der Automat in mehreren Zuständen gleichzeitig befindet.

LÖSUNG

- A_1
  - ω_1: $q_0 \xrightarrow{x} q_1 \xrightarrow{y} q_2 \xrightarrow{z} q_0$ ✓ (akzeptiert).
  - ω_2: $q_0 \xrightarrow{x} q_1 \xrightarrow{y} q_2 \xrightarrow{x}$ stop $\lightning$ (nicht akzeptiert).
- A_2
  - ω_1: $q_0 \xrightarrow{x} q_1 \xrightarrow{y} q_2 \xrightarrow{z} q_3$ ✓ (akzeptiert).
  - ω_2: $q_0 \xrightarrow{x} q_1 \xrightarrow{y} q_2 \xrightarrow{x}$ stop $\lightning$ (nicht akzeptiert).
- N_3



  - ω_1:
    ✓ (akzeptiert, weil auf einem Verarbeitungspfad das Wort komplett eingelesen und akzeptiert wurde)



  - ω_2:
    ✓ (akzeptiert, weil auf einem Verarbeitungspfad das Wort komplett eingelesen und akzeptiert wurde)

TEILAUFGABE 3.5.2 2 PUNKTE

Geben Sie für jeden der Automaten die Sprache an, welche er akzeptiert.

LÖSUNG

- $\mathscr{L}(A_1) = \{(xyz)^n \mid n \in \mathbb{N}_0\} = \mathscr{L}((xyz)^*)$
- $\mathscr{L}(A_2) = \{xyz(xxyz)^n \mid n \in \mathbb{N}_0\} = \mathscr{L}(xyz(xxyz)^*)$
- $\mathscr{L}(N_3) = \mathscr{L}((x^* \mid xy \mid xyz)^*)$

Aufgabe 3.6 Reguläre Ausdrücke und DEAs

Betrachten Sie das Alphabet $\Sigma = \{a, b\}$ sowie die Sprachen, die aus den folgenden regulären Ausdrücken (bekannt aus Aufgabe 2.9) erzeugt werden:

a) $L_1 = \mathscr{L}(r_1)$ mit $r_1 = (a|b)(a|b)$

b) $L_2 = \mathscr{L}(r_2)$ mit $r_2 = (a|b)^*$

c) $L_3 = \mathscr{L}(r_3)$ mit $r_3 = a^*|b^*$

d) $L_4 = \mathscr{L}(r_4)$ mit $r_4 = a^+|b^+$

e) $L_5 = \mathscr{L}(r_5)$ mit $r_5 = a^*b^*$

f) $L_6 = \mathscr{L}(r_6)$ mit $r_6 = a^+b^+$

g) $L_7 = \mathscr{L}(r_7)$ mit $r_7 = (ab^*)^*$

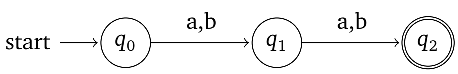
h) $L_8 = \mathscr{L}(r_8)$ mit $r_8 = (aa|b)^*$

Teilaufgabe 3.6.1 4 Punkte

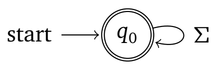
Geben Sie die DEAs A_1, A_2, A_3, A_4 an, welche die Sprachen L_1, L_2, L_3, L_4 akzeptieren.

Lösung

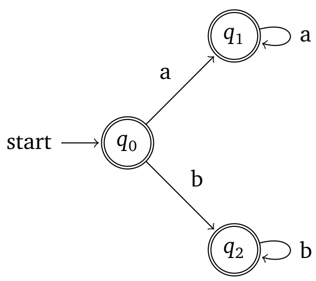
a) $A_1 = (Q, \Sigma, \delta_1, F, q_0)$ entsprechend $r_1 = (a|b)(a|b)$ mit $Q = \{q_0, q_1, q_2\}, F = \{q_2\}$ und δ_1 gegeben durch:



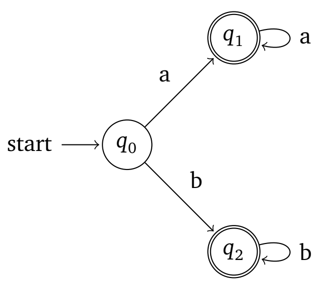
b) $A_2 = (Q, \Sigma, \delta_2, F, q_0)$ entsprechend $r_2 = (a|b)^*$ mit $Q = \{q_0\}, F = \{q_0\}$ und δ_2 gegeben durch:



c) $A_3 = (Q, \Sigma, \delta_3, F, q_0)$ entsprechend $r_3 = a^*|b^*$ mit $Q = \{q_0, q_1, q_2\}, F = \{q_0, q_1, q_2\}$ und δ_3 gegeben durch:



d) $A_4 = (Q, \Sigma, \delta_4, F, q_0)$ entsprechend $r_4 = a^+|b^+$ mit $Q = \{q_0, q_1, q_2\}, F = \{q_1, q_2\}$ und δ_4 gegeben durch:

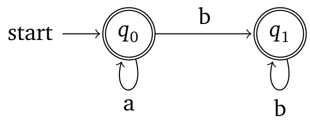


Teilaufgabe 3.6.2 4 Punkte

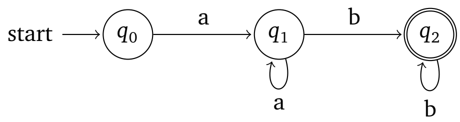
Geben Sie die DEAs A_5,A_6,A_7,A_8 an, welche die Sprachen L_5, L_6, L_7, L_8 akzeptieren.

Lösung

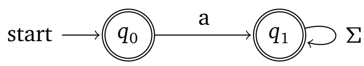
e) $A_5 = (Q, \Sigma, \delta_5, F, q_0)$ entsprechend $r_5 = a^*b^*$ mit $Q = \{q_0, q_1\}, F = \{q_0, q_1\}$ und δ_5 gegeben durch:



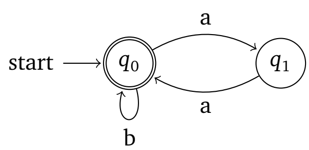
f) $A_6 = (Q, \Sigma, \delta_6, F, q_0)$ entsprechend $r_6 = a^+b^+$ mit $Q = \{q_0, q_1, q_2\}, F = \{q_2\}$ und δ_6 gegeben durch:



g) $A_7 = (Q, \Sigma, \delta_7, F, q_0)$ entsprechend $r_7 = (ab^*)^*$ mit $Q = \{q_0, q_1\}, F = \{q_1\}$ und δ_3 gegeben durch:



h) $A_8 = (Q, \Sigma, \delta_8, F, q_0)$ entsprechend $r_8 = (aa|b)^*$ mit $Q = \{q_0, q_1\}, F = \{q_0\}$ und δ_8 gegeben durch:



Aufgabe 3.7 Paritätscode

Zur Fehlererkennenung bei einer Datenübertragung wird oft der *Paritätscode* genutzt.

Die Grundidee des Paritätscodes ist, ausschließlich Datenpakete zu versenden, die eine gerade Anzahl Einsen aufweisen. Hierzu werden die Datenpakete vor dem Versenden um ein Paritätsbit ergänzt, das die Gesamtzahl der Einsen bei Bedarf gerade werden lässt.

Ein Übertragungsfehler wird dadurch erkannt, dass die Anzahl der Einsen ungerade ist. Ist die Anzahl der Einsen gerade, wurde das Paket korrekt übertragen.

Das vor der Übertragung hinzuzufügende Paritätsbit berechnet sich wie folgt:

- 1, falls die Anzahl der Einsen im Datenpaket ungerade ist
- 0, falls die Anzahl der Einsen im Datenpaket gerade ist

Ein Paritätscode der Länge 4 versieht also die ersten 3 Datenbits mit einem 4. Paritätsbit, so dass die Pakete insgesamt wie folgt aussehen: $000|0, 001|1, 010|1, \ldots$

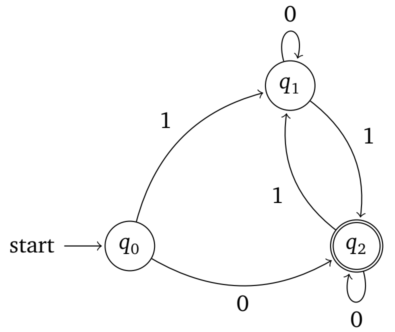
Teilaufgabe 3.7.1 2 Punkte

Konstruieren Sie einen DEA A_P, der die Integrität eines empfangenen Datenpakets überprüft und alle korrekt übertragenen Wörter **beliebiger Länge** akzeptiert. Wurde ein einzelnes Bit des Datenpakets während der Übertragung verfälscht, so soll der Automat das Eingabewort ablehnen.

Das leere Wort soll ebenfalls abgelehnt werden.

Lösung

$A_P = (Q, \Sigma, \delta, F, q_0)$ mit $Q = \{q_0, q_1, q_2\}, \Sigma = \{0, 1\}, F = \{q_2\}$ und δ gegeben durch:



Teilaufgabe 3.7.2 1 Punkt

Wie verhält sich der von Ihnen konstruierte Automat, falls zwei Bits während der Datenübertragung verfälscht wurden? Weist er dieses falsche Wort auch zurück? Begründen Sie Ihre Aussage.

Lösung

Werden zwei Bits verfälscht, so kann der Übertragungsfehler nicht mehr erkannt werden. Genau hierin liegt die größte Schwäche des Parity-Codes.

Teilaufgabe 3.7.3 2 Punkte

Konstruieren Sie eine **reguläre** Grammatik G_P, welche alle korrekt übertragenen Wörter **beliebiger Länge** (also mit einer geraden Anzahl von 1en) erzeugt. Das leere Wort soll nicht in der Sprache enthalten sein.

Lösung

$G_P = (N, \Sigma, P, S)$ mit $N = \{S, U\}, \Sigma = \{0, 1\}$ und P gegeben durch: P :

- $S \to 0S \mid 1U \mid 0$
- $U \to 1S \mid 0U \mid 1$

Aufgabe 3.8 Die Sprache L_x

Betrachten Sie die Sprache L_x die mit Hilfe eines regulären Ausdrucks definiert ist:

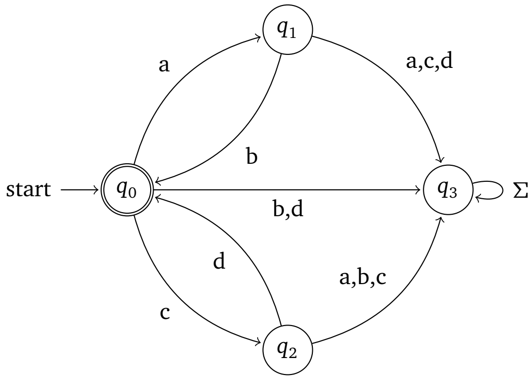
$$L_x = \mathscr{L}(r_x) \qquad \text{und } r_x = ((ab)|(cd))^*$$

Teilaufgabe 3.8.1 3 Punkte

Geben Sie den DEA A_x an, der L_x akzeptiert. Zusatzanforderung an A_x: alle (auch nicht akzeptierte) Worte sollen komplett eingelesen werden.

LÖSUNG

$A_x = (Q, \Sigma, \delta, F, q_0)$ mit $Q = \{q_0, q_1, q_2, q_3\}, \Sigma = \{a, b, c\}, F = \{q_0\}$ und δ gegeben durch:



TEILAUFGABE 3.8.2 2 PUNKTE

Geben Sie die **reguläre** Grammatik G_x an, die L_x erzeugt.

LÖSUNG

$G_x = (\Sigma, N, P, S) = (\{a, b, c, d\}, \{S, B, D\}, P, S)$ mit P gegeben durch $P: \begin{array}{lcl} S & \rightarrow & aB \mid cD \mid \varepsilon \\ B & \rightarrow & bS \mid b \\ D & \rightarrow & dS \mid d \end{array}$

AUFGABE 3.9 DAS ENDE IST ABC

Gegeben sei die Sprache L_1 aller Wörter über $\Sigma = \{a, b, c\}$, die auf abc enden:
$L_1 = \{\omega \in \Sigma^* \mid \omega = xabc, x \in \Sigma^*\}$

TEILAUFGABE 3.9.1 1 PUNKT

Geben Sie einen regulären Ausdruck r_1 an, der L_1 erzeugt.

LÖSUNG

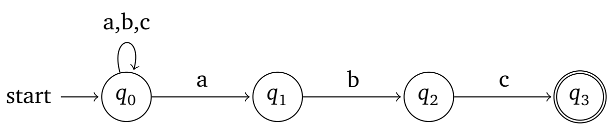
$r_1 = [abc]^* abc$

TEILAUFGABE 3.9.2 2 PUNKTE

Geben Sie einen nichtdeterministischen endlichen Automaten N_1 an, für den $\mathscr{L}(N_1) = L_1$ gilt. Achten Sie darauf, dass Ihr Automat nichtdeterministische Elemente enthält.

LÖSUNG

$N_1 = (\Sigma, Q, \delta, q_0, F) = (\{a, b, c\}, \{q_0, q_1, q_2, q_3\}, q_0, \{q_3\})$ mit δ :



und $\mathscr{L}(A_y) = L_y$.

TEILAUFGABE 3.9.3 2 PUNKTE

Geben Sie einen deterministischen endlichen Automaten $A_1 = (\Sigma, Q, \delta, q_0, F)$ an, für den $\mathscr{L}(A_1) = L_1$ gilt.

LÖSUNG

Siehe Abbildung 2.

Der deterministische endliche Automat A_1 ergibt sich zu:

$$A_1 = (\{a, b, c\}, \{s_0, s_1, s_2, s_3\}, \delta_1, s_0, \{s_3\})$$

δ_1:

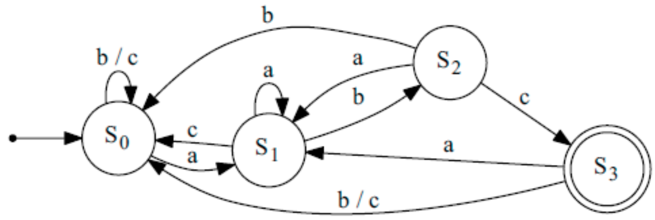


Abbildung 2: aus König et al.: „100 Übungsaufgaben zu Grundlagen der Informatik Band I: Theoretische Informatik“, Oldenbourg

AUFGABE 3.10 01 AM BEGINN UND ENDE (KLAUSUR WS 15/16)

Wir betrachten das Alphabet $\Sigma = \{0, 1\}$ und Sprache $L_x \subseteq \Sigma^*$, welche alle Wörter aus Σ^* enthält, die sowohl mit 01 beginnen, als auch mit 01 enden.

Beispielhafte Wörter aus L_x sind 0101, 01001, 0101011100101.

Die Wörter 01, 01111, 001101 gehören **nicht** zu L_x.

TEILAUFGABE 3.10.1 1 PUNKT

Geben Sie die Sprache L_x formal, als Menge in der deskriptiven Form an.

LÖSUNG

Mögliche Lösungen:

- $L_x = \{\omega \in \Sigma^* \mid \omega = 01\nu01, \nu \in \Sigma^*\}$
- $L_x = \{\omega \in \Sigma^* \mid \exists_{\nu\in\Sigma^*} \omega = 01\nu01\}$
- $L_x = \{01\nu01, \nu \in \Sigma^*\}$
- $L_x = \{\omega \in \Sigma^* \mid \omega$ endet und beginnt mit $01\}$

TEILAUFGABE 3.10.2 1 PUNKT

Geben Sie den regulären Ausdruck r_x an, welcher die Sprache L_x erzeugt, für welchen also $\mathscr{L}(r_x) = L_x$ gilt. Sie können hierfür eine formale, oder eine Unix-ähnliche Syntax wählen.

LÖSUNG

Mögliche Lösungen:

- $r_x = 01(0|1)^*01$
- $r_x = 01[01]^*01$

TEILAUFGABE 3.10.3 3 PUNKTE

Geben Sie den NEA N_x an, welcher L_x akzeptiert, für welchen also $\mathscr{L}(N_x) = L_x$ gilt. Achten Sie darauf, dass Ihr Automat mindestens ein nichtdeterministisches Element enthält!

LÖSUNG

Konstruiere N_x mit $\mathscr{L}(N_X) = L_x$ als $N_x = (Q, \Sigma, \delta, F, s_0)$ mit

- $Q = \{s_0, s_1, s_2, s_3, s_4\}$
- $\Sigma = \{0, 1\}$
- $F = \{s_4\}$
- δ gegeben durch Abbildung 3. Alternative Lösungen möglich, allerdings muss der Automat nichtdeterministische Elemente beinhalten.

δ	0	1
s_0	$\{s_1\}$	$\emptyset$
s_1	$\emptyset$	$\{s_2\}$
s_2	$\{s_2, s_3\}$	$\{s_2\}$
s_3	$\emptyset$	$\{s_4\}$
s_4	$\emptyset$	$\emptyset$

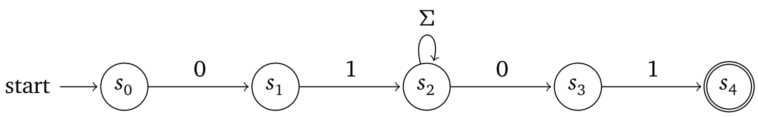


Abbildung 3: Zustandsübergangstabelle und -diagramm für N_x

TEILAUFGABE 3.10.4 3 PUNKTE

Geben Sie den DEA A_x an, welcher L_x akzeptiert, für welchen also $\mathscr{L}(A_x) = L_x$ gilt.
Zusatzanforderung an A_x: alle (auch nicht akzeptierte) Eingabewörter sollen komplett eingelesen werden. Bei Nicht-Akzeptanz soll A_x in einem beliebigen Nicht-Final-Zustand enden.

LÖSUNG

Konstruiere A_x mit $\mathscr{L}(A_x) = L_x$ als $A_x = (Q, \Sigma, \delta, F, s_0)$ mit

- $Q = \{s_0, s_1, s_2, s_3, s_4, s_5\}$
- $\Sigma = \{0, 1\}$
- $F = \{s_4\}$
- δ gegeben durch Abbildung 4. Alternative Lösungen möglich.

δ	0	1
s_0	s_1	s_5
s_1	s_5	s_2
s_2	s_3	s_2
s_3	s_3	s_4
s_4	s_3	s_2
s_5	s_5	s_5

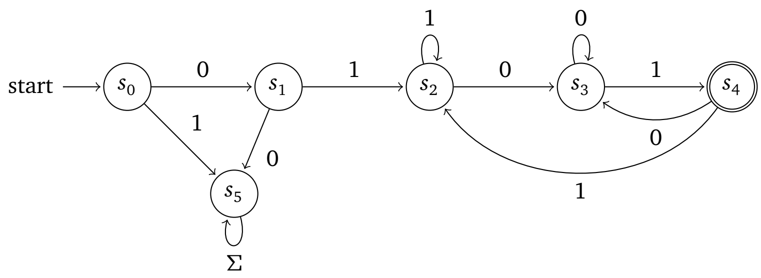


Abbildung 4: Zustandsübergangstabelle und -diagramm für A_x

Teilaufgabe 3.10.5 2 Punkte

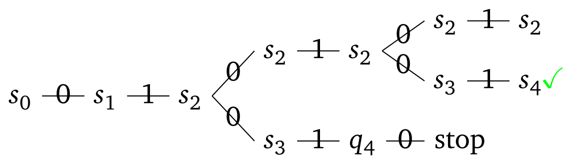
Gegeben sei $\omega_1 \in L_x$ mit $\omega_1 = 010101$.

a) Geben Sie **alle** Zustände an, welche A_x bei der Verarbeitung von ω_1 durchläuft.

b) Geben Sie **alle** Zustände an, welche N_x bei der Verarbeitung von ω_1 durchläuft.

Lösung

a) $A_x : s_0 \xrightarrow{0} s_1 \xrightarrow{1} s_2 \xrightarrow{0} s_3 \xrightarrow{1} s_4 \xrightarrow{0} s_3 \xrightarrow{1} s_4$ ✓

b) 

Teilaufgabe 3.10.6 3 Punkte

a) Geben Sie die reguläre Grammatik G_x an, welche L_x erzeugt.

b) Leiten Sie mit Hilfe der Regeln von G_x das Wort $\omega = 0100101$ aus dem Startsymbol ab.

Hinweis: Die Menge Ihrer Regeln hat keinen Einfluss auf die Punktgebung, alle Punkte erhalten Sie allerdings nur für *reguläre* Regeln.

Lösung

a) Zu beachten ist, dass zuerst eine 0, dann eine 1; dann beliebige Zeichen; am Ende wieder 0 und 1 erzeugt werden. Dies führt zur Grammatik $G_x = (N, \Sigma, P, S)$ mit $\mathscr{L}(G_x) = L_x$:

- $N = \{S, T, U, V\}$
- $\Sigma = \{0, 1\}$
- P gegeben durch: $$\begin{array}{lcl} S & \rightarrow & 0T \\ T & \rightarrow & 1U \\ U & \rightarrow & 0U \mid 1U \mid 0V \\ V & \rightarrow & 1 \end{array}$$

b) Ableitung von 0100101: $S \Rightarrow 0T \Rightarrow 01U \Rightarrow 010U \Rightarrow 0100U \Rightarrow 01001U \Rightarrow 010010V \Rightarrow 0100101$

Aufgabe 3.11 Schlechte Passwörter

Eine der am häufigsten genutzten Passwörter im deutschsprachigen Raum ist „qwertz“. Die IT-Abteilung in der Sie Ihr Praktikum verbringen, möchte deshalb in einem schwach gesicherten System, das als Passwörter nur Kleinbuchstaben erlaubt, alle Passwörter ausmerzen, die gleich, oder ähnlich zu „qwertz“ sind und betreut Sie mit verschiedenen Aufgaben.

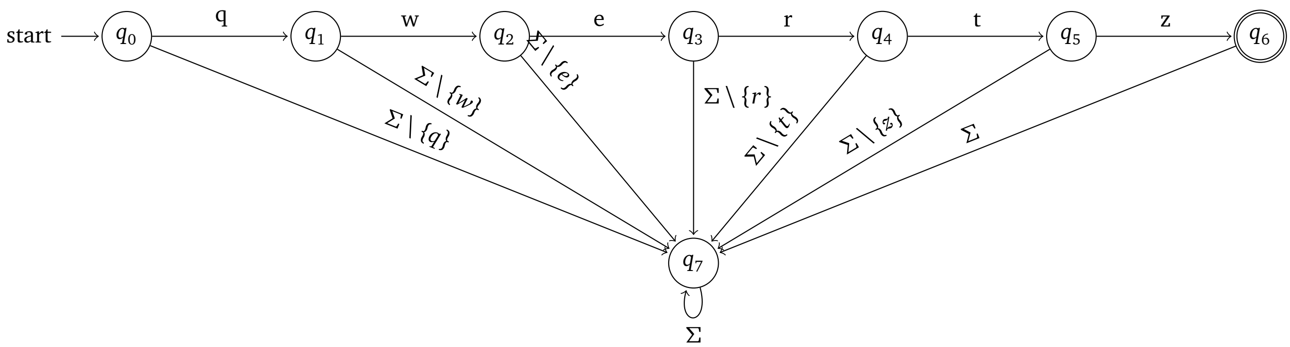
Teilaufgabe 3.11.1 Ein DEA für qwertz, 3 Punkte

Konstruieren Sie einen DEA A_Q, der eine eingegebene beliebig lange Zeichenkette genau dann akzeptiert, falls Sie identisch zu „qwertz“ sind.

Konstruieren Sie Ihren Automaten so, dass er während des Lesens eines Wortes ungleich „qwertz“ nicht abbricht, sondern das Wort komplett einliest und sich nach Abschluss des Einlesens in einem gesonderten Fehlerzustand befindet.

Lösung

$A_Q = (Q, \Sigma, \delta, F, q_0)$ mit δ



- $\Sigma = \{a, b, \ldots, z\}$
- $Q = \{q_0, q_1, \ldots, q_7\}$
- $F = \{q_6\}$

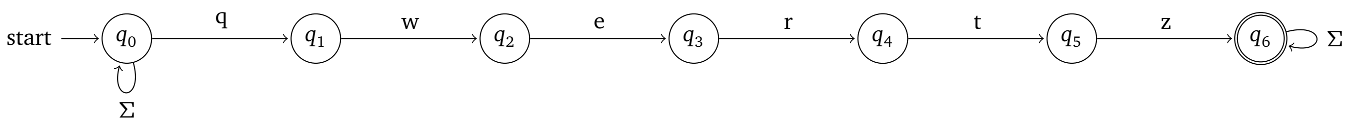
Teilaufgabe 3.11.2 Ein NEA für *qwertz*, 2 Punkte

Konstruieren Sie einen NEA N_Q, der eine eingegebene beliebig lange Zeichenkette genau dann akzeptiert, falls Sie als Teilwort den String „qwertz“ enthält.

Beispiele: qwertz, qwertzz, asdfqwertzuio werden akzeptiert, qwert, quwertz, ertz werden nicht akzeptiert.

Lösung

$N_Q = (Q, \Sigma, \delta, F, q_0)$ mit δ



- $\Sigma = \{a, b, \ldots, z\}$
- $Q = \{q_0, q_1, \ldots, q_6\}$
- $F = \{q_6\}$

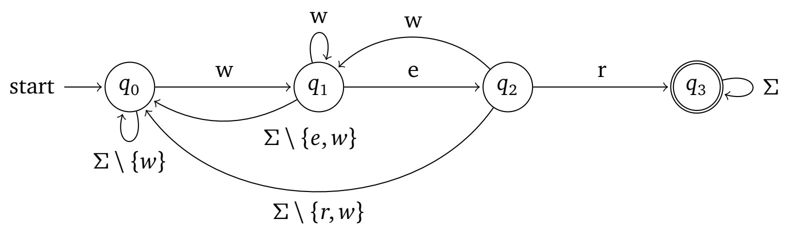
Teilaufgabe 3.11.3 Ein DEA für *wer*, 3 Punkte

Konstruieren Sie einen DEA A_W, der eine eingegebene beliebig lange Zeichenkette genau dann akzeptiert, falls Sie als Teilwort den String „wer“ enthält.

Beispiele: wer, qwertz, werwolf werden akzeptiert, we, wetr, erw werden nicht akzeptiert.

Lösung

$A_W = (Q, \Sigma, \delta, F, q_0)$ mit δ



- $\Sigma = \{a, b, \ldots, z\}$
- $Q = \{q_0, q_1, \ldots, q_3\}$
- $F = \{q_3\}$

Teilaufgabe 3.11.4 Ein DET zur Erkennung von *wer*, 3 Punkte

Konstruieren Sie einen DET T_W, der als Eingabe eine beliebig lange Zeichenkette annimmt und auf dem Ausgabeband für jedes Zeichen ein Kästen schreibt. T_W soll ein markiertes Kästchen schreiben, wenn als Teilwort der String „wer“ gelesen wurde. Insbesondere soll jedes Vorkommen von „wer“ mit einem markierten Kästchen signalisiert werden.

Es bleibt Ihnen überlassen, ob Sie T_W als Mealy- oder als Moore-Automat konstruieren.

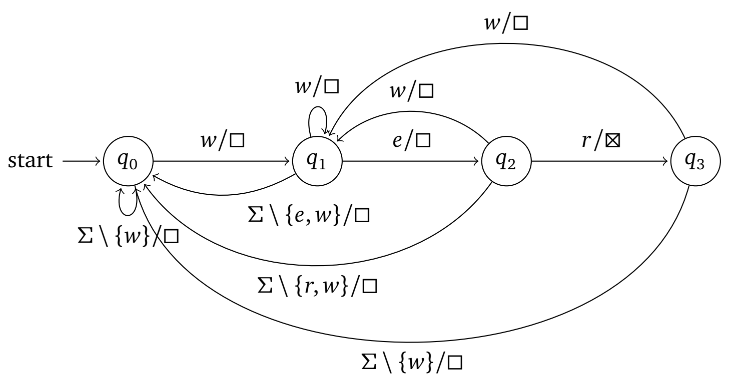
Beispiele:

- wer → □□⊠
- qwertz → □□□ ⊠ □□
- werwolfwer → □□ ⊠ □□□□□□□⊠
- wetr → □□□□

Lösung

$T_W = (Q, \Sigma, \Pi, \delta, \lambda, q_0)$ mit δ, λ siehe erweiterter Zustandsübergangsgraph

- $\Sigma = \{a, b, \ldots, z\}$
- $Q = \{q_0, q_1, \ldots, q_3\}$
- $\Pi = \{\square, \boxtimes\}$



.

Aufgabe 3.12 Informatik-Studiengänge

Die von der Fakultät für Informatik angebotenen Bachelor-Studiengänge sind Allgemeine Informatik (ain), Automobilinformationstechnik (ait), Gesundheitsinformatik (gib) und Wirtschaftsinformatik (win). Wir betrachten das Alphabet aller Kleinbuchstaben $\Sigma = \{a, b, \ldots, z\}$.

Teilaufgabe 3.12.1 Ein NEA für die Informatik, 3 Punkte

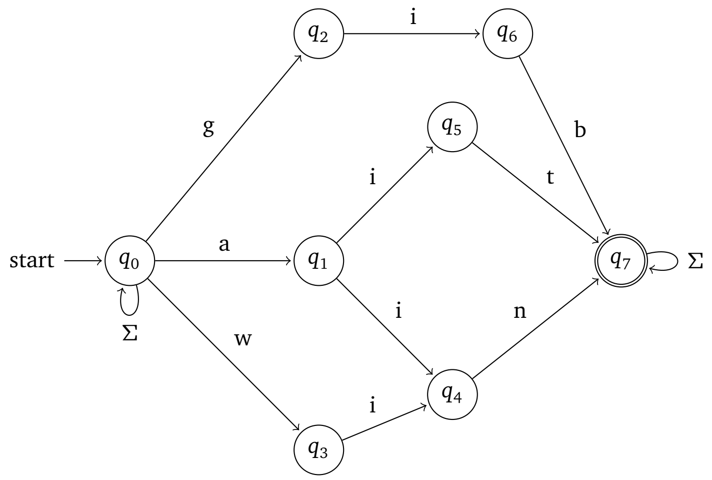
Konstruieren Sie einen NEA N_I, der eine eingegebene beliebig lange Zeichenkette über dem Alphabet Σ genau dann akzeptiert, falls Sie als Teilwort mindestens einen der Strings der genannten Bachelorstudiengänge (ain, ait, gib, win) enthält.

Konstruieren Sie den Automaten so, dass er möglichst wenige Zustände enthält.

Beispiele: ain, aitain, gggibbbb werden akzeptiert, aiai, wien, xyzwi werden nicht akzeptiert.

Lösung

$N_I = (Q, \Sigma, \delta, F, q_0)$ mit δ



- $\Sigma = \{a, b, \ldots, z\}$
- $Q = \{q_0, q_1, \ldots, q_7\}$
- $F = \{q_7\}$

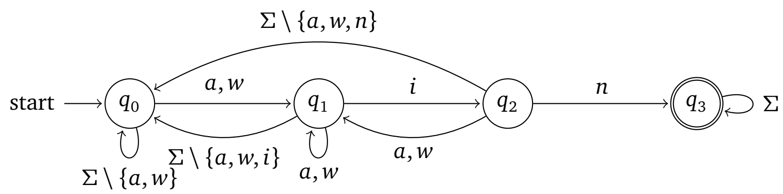
Teilaufgabe 3.12.2 Ein DEA für *(a|w)in*, 3 Punkte

Konstruieren Sie einen DEA A_I, der eine eingegebene beliebig lange Zeichenkette genau dann akzeptiert, falls Sie als Teilwort mindestens einen der Strings ain oder win enthält.

Beispiele: aaain, winner, winainait werden akzeptiert, mai, mawi, xaxixn werden nicht akzeptiert.

Lösung

$A_I = (Q, \Sigma, \delta, F, q_0)$ mit δ



- $\Sigma = \{a, b, \ldots, z\}$

- $Q = \{q_0, q_1, \dots, q_3\}$
- $F = \{q_3\}$

Teilaufgabe 3.12.3 Ein DET zur Erkennung von *(a|w)in*, 3 Punkte

Konstruieren Sie einen DET T_I, der als Eingabe eine beliebig lange Zeichenkette annimmt und auf dem Ausgabeband für jedes Zeichen ein Kästen ausgibt. T_I soll ein markiertes Kästchen schreiben, wenn als Teilwort der String ain oder win vollständig gelesen wurde. Es soll also jedes Vorkommen von ain oder win mit einem markierten Kästchen signalisiert werden.

Es bleibt Ihnen überlassen, ob Sie T_I als Mealy- oder als Moore-Automat konstruieren.

Beispiele:

- ain $\rightarrow \square\square\boxtimes$
- winner $\rightarrow \square\square\boxtimes\square\square\square$
- winainait $\rightarrow \square\square\boxtimes\square\square\boxtimes\square\square\square$
- aixn $\rightarrow \square\square\square\square$

Lösung

Lösung mit Mealy-Automat:
$T_I = (Q, \Sigma, \Pi, \delta, \lambda, q_0)$ mit δ, λ siehe erweiterter Zustandsübergangsgraph

- $\Sigma = \{a, b, \dots, z\}$
- $Q = \{q_0, q_1, \dots, q_3\}$
- $\Pi = \{\square, \boxtimes\}$