Aufgabe 3.7 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 $r_6 = a^+b^+$

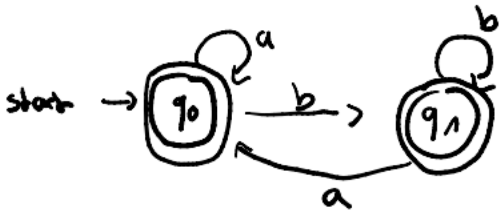
g) $L_7 = \mathscr{L}(r_7)$ mit $r_7 = (ab^*)^*$

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

Teilaufgabe 3.7.1 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.

a) L1 = L (r1) mit r1 = (a|b)(a|b)
b) L2 = L (r2) mit r2 = (a|b) ∗
c) L3 = L (r3) mit r3 = a ∗ |b ∗
d) L4 = L (r4) mit r4 = a +|b +



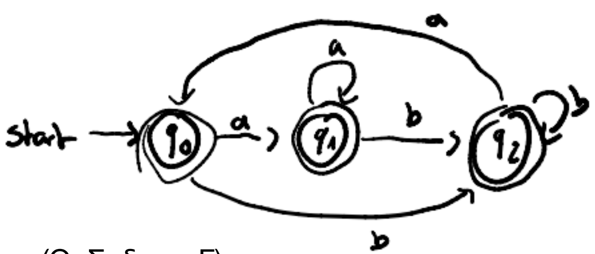
$N_x = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1\}$
$\Sigma = \{a, b\}$
$F = \{q_0, q_1\}$

Teilaufgabe 3.7.2 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.

e) L5 = L (r5) mit r5 = a ∗ b ∗ f)
f) L6 = L (r6) mit r6 = a + b + g)
g) L7 = L (r7) mit r7 = (ab∗) ∗ h)
h) L8 = L (r8) mit r8 = (aa|b) ∗



$N_x = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{a, b\}$
$F = \{q_0, q_1, q_2\}$

Aufgabe 3.8 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:

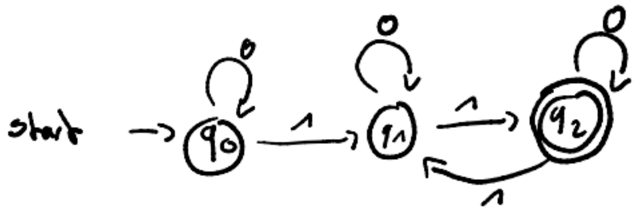
- 1, falls die Anzahl der Einsen im Datenpaket ungerade ist
- 0, falls die Anzahl der Einsen im Datenpaket gerade ist

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

Teilaufgabe 3.8.1 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.



$N_x = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1, q_2\}$
$\Sigma = \{0, 1\}$
$F = \{q_2\}$

Wie wird das leere Wort in einem DEA abgelehnt?

Teilaufgabe 3.8.2 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.

2 gleiche verfälschte bits (1,1 || 0,0) **=>** immer noch gerade Anzahl von 1ern
2 ungleiche verfälschte bits (1,0 || 0,1) **=>** wird nicht akzeptiert

Teilaufgabe 3.8.3 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.

Gp = (N,Σ,P,S) = ({S, A, B}, {0, 1}, P, S) s=> 0A | 1C
A=> 0A | 0B | 0
B => 1C
C => 0C | 1A | 1C | 1

Aufgabe 3.9 Automaten für Zahlen-Sprachen

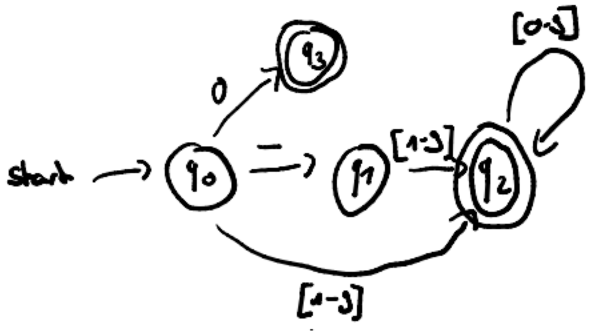
Für die Sprachen L_Z und L_G hatten Sie auf Übungsblatt 2 schon Grammatiken und reguläre Ausdrücke angegeben.

Für die alle Teil-Aufgaben gilt: $\Sigma = \{-, 0, 1, \dots, 9\}$.

Teilaufgabe 3.9.1 Die Sprache der ganzen Zahlen, 3 Punkte

$L_Z \subseteq \Sigma^*$ mit $L_Z = \{\dots, -78562, -11, -10, \dots -1, 0, 1, \dots 9, 10, \dots, 5906, \dots,\}$ sei die Sprache der ganzen Zahlen. Achtung: Zahlen mit führenden Nullen ($-09, -0000340, 0123, 000, \dots$) sind nicht in L_Z enthalten!

Geben Sie einen endlichen Automaten (DEA oder NEA) an, welcher L_Z akzeptiert.



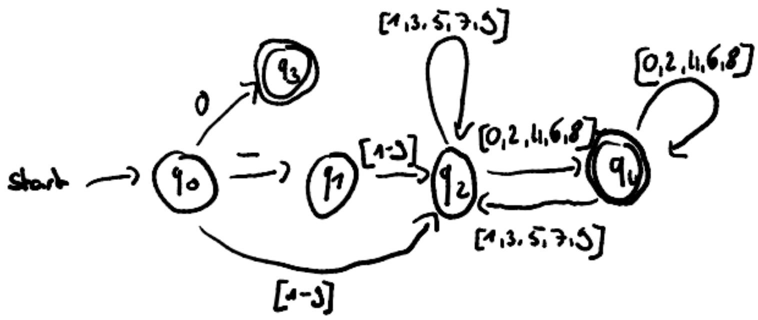
$N_z = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1, q_2, q_3\}$
$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -\}$
$F = \{q_2, q_3\}$

Teilaufgabe 3.9.2 Die Sprache der geraden Zahlen, 4 Punkte

$L_G \subseteq \Sigma^*$ mit $L_G = \{\dots, -78562, -12, -10, \dots -2, 0, 2, \dots 8, 10, \dots, 5906, \dots,\}$ sei die Sprache der geraden Zahlen. Achtung: Zahlen mit führenden Nullen ($-08, -0000340, 0124, 000, \dots$) sind nicht in L_G enthalten!

Geben Sie einen endlichen Automaten (DEA oder NEA) an, welcher L_G akzeptiert.



$N_z = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1, q_2, q_3, q_4\}$
$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -\}$
$F = \{q_4, q_3\}$

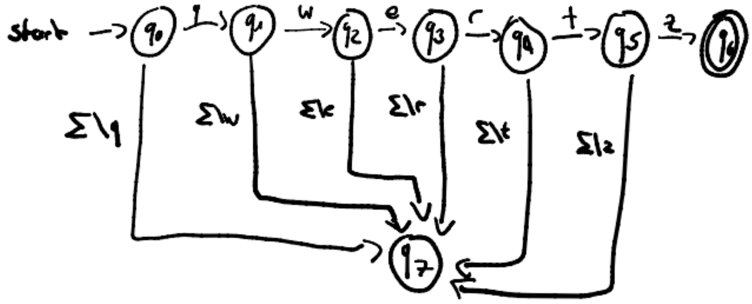
Aufgabe 3.12 Schlechte Passwörter

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 3.12.1 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.



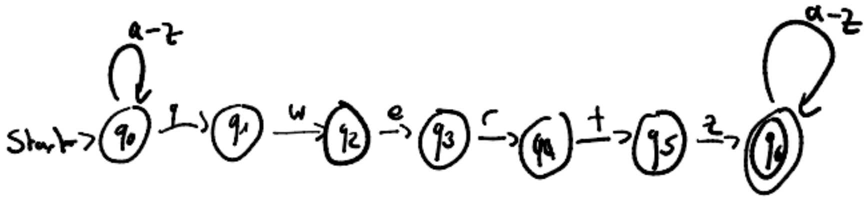
$A_q = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$
$\Sigma = \{a\text{-}z\}$
$F = \{q_0\}$

Teilaufgabe 3.12.2 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.



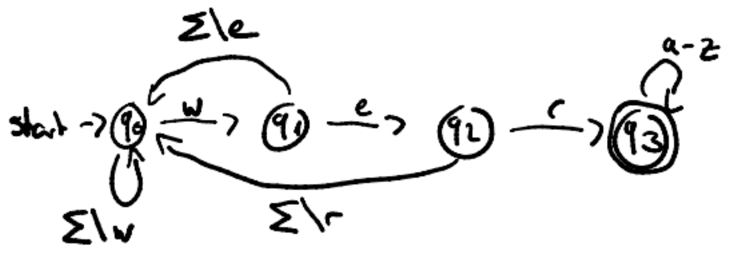
$N_a = (Q, \Sigma, \delta, q_0, F)$

$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$
$\Sigma = \{a\text{-}z\}$
$F = \{q_6\}$

TEILAUFGABE 3.12.3 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.



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

$Q = \{q_0, q_1, q_2, q_3\}$
$\Sigma = \{a\text{-}z\}$
$F = \{q_3\}$

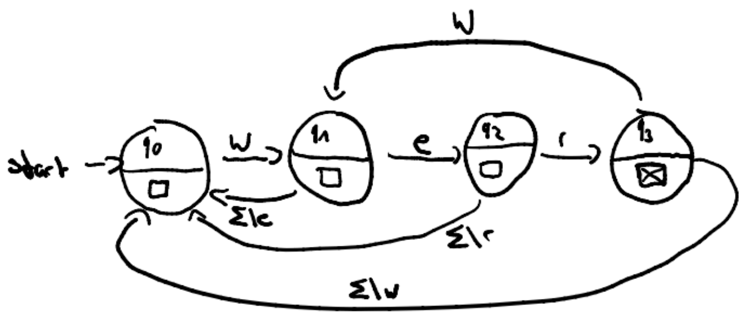
TEILAUFGABE 3.12.4 EIN DET ZUR ERKENNUNG VON *WER*, 3 PUNKTE

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 → □□□□



$T_W = (Q, \Sigma, \delta, \lambda, q_0, F)$

$Q = \{q_0, q_1, q_2, q_3\}$
$\Sigma = \{a\text{-}z\}$
$F = \{$□ ,⊠$\}$