Prof. T. Nipkow
M. Eberl, J. Rädle, S. Wimmer
Technische Universität München
KEINE ABGABE

Einführung in die theoretische Informatik

Sommersemester 2019 – Übungsblatt Lösungsskizze 2

AUFGABE 2.1. (*Wichtige Begriffe*) Stufe A

Überprüfen Sie, dass Sie die Folgenden Begriffe korrekt definieren können.

- DFA
- NFA
- akzeptierte Sprache
- Zustandsgraph
- Akeptanzbedingung von DFAs/NFAs
- Potenzmengenkonstruktion
- rechtslineare Grammatik

Hinweis: Die Aufgaben 2.2 und 2.3 werden ab dem 06.05. auf AutomataTutor zur Verfügung stehen. Dort können Sie direkt sehen, ob sie die richtige Lösung haben und bekommen individuelle Hilfestellung beim Lösen der Aufgaben. Klicken Sie dafür nach Anmeldung auf "English to NFA" bzw. "English to DFA" und lösen sie die Aufgaben.

AUFGABE 2.2. Stufe B

Geben Sie für jede der folgenden Sprachen über dem Alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ einen NFA an, der genau die jeweilige Sprache erkennt:

(a) Alle Wörter, die `aaab` enthalten.
(b) Alle Wörter, deren drittletzter Buchstabe ein `a` ist, z.B. `babb`. Der Automat sollte nicht mehr als 4 Zustände haben.

Lösungsskizze

Die Lösungsskizzen finden Sie unter `https://www21.in.tum.de/teaching/theo/SS19/ex/theo19-02-lsg.zip`.

AUFGABE 2.3. Stufe B

Geben Sie für jede der folgenden Sprachen über dem Alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ einen DFA (graphisch) an, der genau die jeweilige Sprache erkennt:

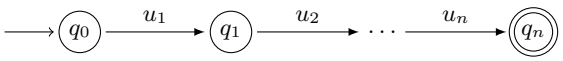
(a) Alle Wörter, die mit einem `b` beginnen.
(b) Alle Wörter gerader Länge.
(c) Alle Wörter ungerader Länge, die auf ein `c` enden.
(d) Die Sprache, die nur das leere Wort enthält.
(e) Alle Wörter, die `aab` enthalten.
(f) Alle Wörter, die eine durch drei teilbare Anzahl von `c` enthalten.
(g) $\mathsf{L} = \{\mathtt{aabcaa}, \mathtt{aacaa}, \mathtt{baa}\}$
(h) Alle Wörter, deren 3-letzter Buchstabe ein `a` ist, z.B. `babb`.

Beschreiben Sie dann in eigenen Worten, wie man im Allgemeinen einen Automaten konstruiert, der eine Sprache erkennt, die ...

(a) am Anfang/Ende jedes Wortes eine bestimmte Sequenz von Buchstaben fordert
(b) nur Worte gerader/ungerader Länge enthält
(c) von jedem Wort verlangt, eine bestimmte Anzahl an Buchstaben zu enthalten
(d) die nur Worte enthält, die eine bestimmte Sequenz von Buchstaben enthält
(e) deren Worte an einer fixierten Position einen eigenen bestimmten Buchstaben haben müssen.

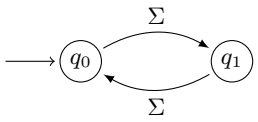
Lösungsskizze
Die Lösungenskizzen finden Sie unter `https://www21.in.tum.de/teaching/theo/SS19/ex/theo19-02-lsg.zip`.
Zu den letzten 5 allgemeineren Punkten:

(a) Sei $u = u_1 \dots u_n$ ein Wort. Wir betrachten dann den folgenden NFA:



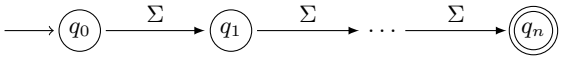
Dieser NFA akzeptiert die Sprache $\{u\}$. Wenn wir bei q_n noch eine Schleife mit Σ hinzufügen, akzeptiert der NFA genau die Sprache aller Wörter, die mit u beginnen. Wenn wir hingegen bei q_0 eine Schleife mit Σ hinzufügen, akzeptiert der NFA genau die Sprache aller Wörter, die mit u enden.
Für ersteren Fall kann man leicht einen DFA daraus bauen, indem man einen Fangzustand hinzufügt. In dem zweiten Fall ist es etwas komplizierter (z.B. NFA determinisieren oder ein ähnlicher Ansatz wie bei Übungsaufgabe 2.8).

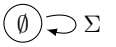
(b)



Wenn q_0 zu einem Endzustand gemacht wird, akzeptiert dieser DFA genau alle Wörter gerader Länge. Wird hingegen q_1 zum Endzustand gemacht, so akzeptiert er genau alle Wörter ungerader Länge.

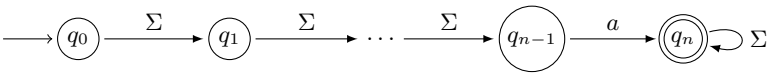
(c) Ähnlich zu Aufgabe a). Der folgende NFA akzeptiert genau alle Wörter der Länge n:



Fügt man noch einen Fangzustand  hinzu und verbindet $q_n \xrightarrow{\Sigma} \emptyset$, so erhalten wir sogar einen DFA.

(d) Dies ist genau die "SuperString"-Sprache aus Übungsaufgabe 2.8.

(e) Es geht um alle Wörter w, die an der Stelle n den Buchstaben a haben. Man kann den NFA aus dem ersten Teil von a) leicht anpassen:



Auch hier erreicht man durch Hinzufügen eines Fangzustands wieder einen DFA.
Wenn man will, dass der n-t-letzte Buchstabe a sein muss (also der an Position $|w| + 1 - n$), so kann man den NFA aus dem zweiten Teil von a) analog anpassen. Der DFA ist hier deutlich komplizierter und hat $\Theta(2^n)$ Zustände (vgl. Beispiel 3.9 aus der Vorlesung oder das Ergebnis aus Übungsaufgabe 2.3 h) von oben).

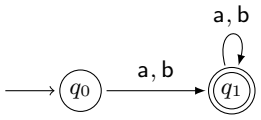
AUFGABE 2.4.

Stufe B

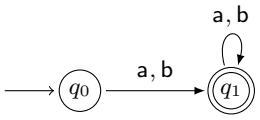
Sei $\Sigma = \{\mathsf{a}, \mathsf{b}\}$ und $\mathsf{B}_n \coloneqq \{w \in \Sigma^* \mid \exists i : w_i = w_{i+n}\}$ die Sprache aller Wörter über Σ, in denen an irgendeiner Stelle der selbe Buchstabe im Abstand n vorkommt. Insbesondere ist B_0 die Menge aller nicht leeren Wörter, und B_1 die Menge aller Wörter in denen ein Buchstabe zweimal hintereinander vorkommt. *Versuchen Sie in beiden Aufgabenteilen NFAs und DFAs mit möglichst wenigen Zuständen anzugeben.*

(a) Geben Sie jeweils einen NFA für B_0, B_1 und B_2 an.
(b) Geben Sie jeweils einen DFA für B_0, B_1 und B_2 an.
(c) Beschreiben Sie kurz, wie der DFA B_n und der NFA B_n für beliebige $n \in \mathbb{N}$ aussehen.
(d) Beurteilen Sie die folgende Aussage: *Es gibt einen NFA für B_n mit $O(n)$-vielen Zuständen, aber jeder DFA zu B_n hat mindestens $\Theta(2^n)$-viele Zustände.*

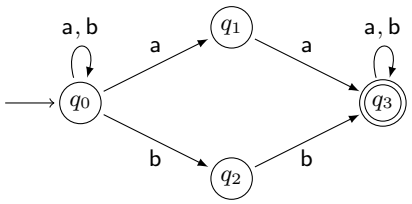
Lösungsskizze



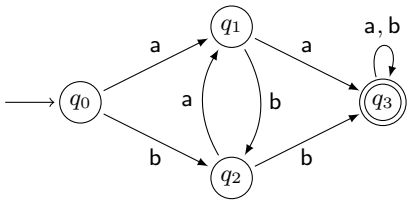
(a) NFA für B_0



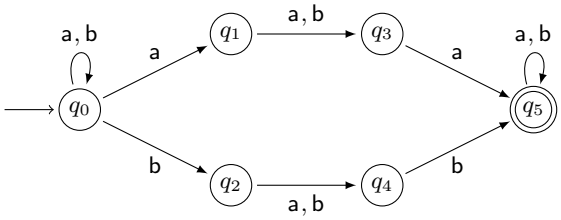
(b) DFA für B_0



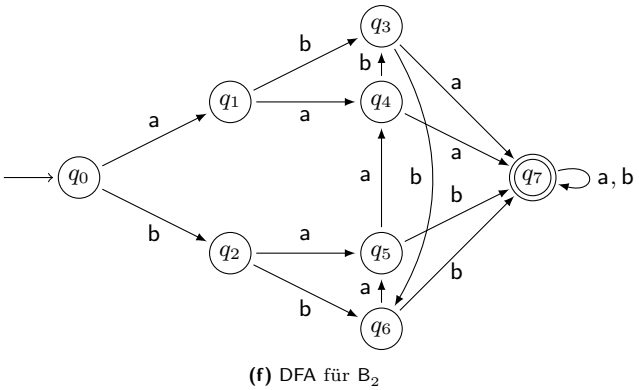
(c) NFA für B_1



(d) DFA für B_1



(e) NFA für B_2



(f) DFA für B_2

(c) Der NFA rät, ob jetzt das i-te Zeichen gelesen wird, welches die Bedingung erfüllt. Der DFA muss sich hingegen immer die letzten n Zeichen die er gelesen hat merken, um zu überprüfen, ob die Bedingung $w_i = w_{i+n}$ erfüllt ist.

(d) Die Aussage ist korrekt. Der Beweis ist ähnlich zu dem Beweis in den Vorlesungsfolien zur Sprache L_k. (Beispiel 3.9)

AUFGABE 2.5. Stufe C

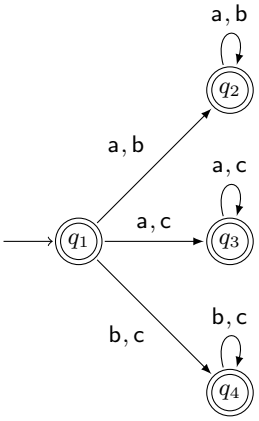
Sei $\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}$ und $\mathsf{L} = \{w \in \Sigma^* \mid \exists x \in \Sigma.\ |w|_x = 0\}$.

(a) Konstruieren Sie einen NFA N mit genau 4 Zuständen und $\mathsf{L}(\mathsf{N}) = \mathsf{L}$.
(b) Determinisieren Sie den NFA N aus (a) mittels der Potenzmengenkonstruktion.
(c) Geben Sie eine rechtslineare Grammatik G (gemäß Satz 3.13) an, so dass $\mathsf{L}(\mathsf{G}) = \mathsf{L}(\mathsf{D})$. D ist der in (b) konstruierte DFA.
(d) Übersetzen Sie die Grammatik G (gemäß Satz 3.14) in einen NFA N', so dass $\mathsf{L}(\mathsf{G}) = \mathsf{L}(\mathsf{N}')$.
(e) Vergleichen Sie die NFAs N' und N bezüglich Zustands- und Transitionszahl. Diskutieren Sie dann, inwiefern Sie Ihre Beobachtung verallgemeinern können.

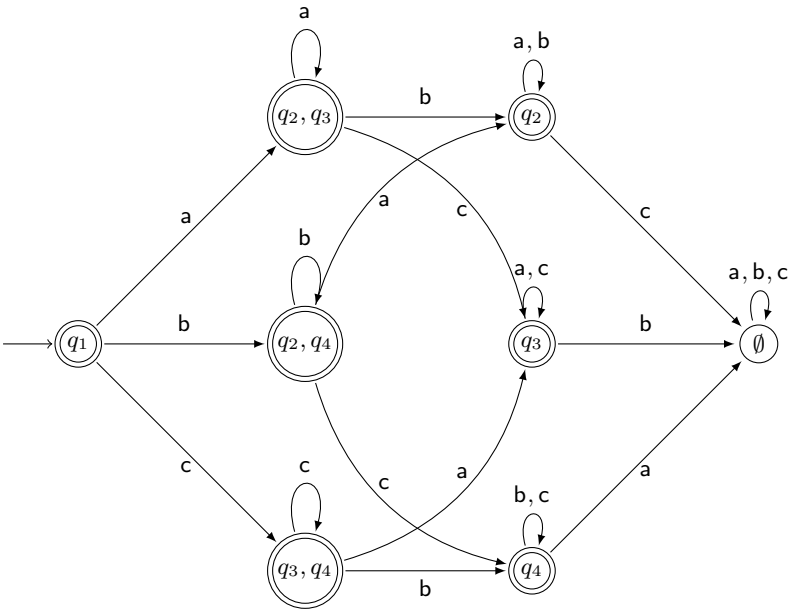
Hinweis: Wir bezeichnen mit $|w|_x$ die Anzahl der Vorkommen des Buchstabens $x \in \Sigma$ in $w \in \Sigma^*$.

Lösungsskizze

(a) Der folgende NFA rät im initialen Zustand welche beiden Buchstaben im Wort auftauchen:



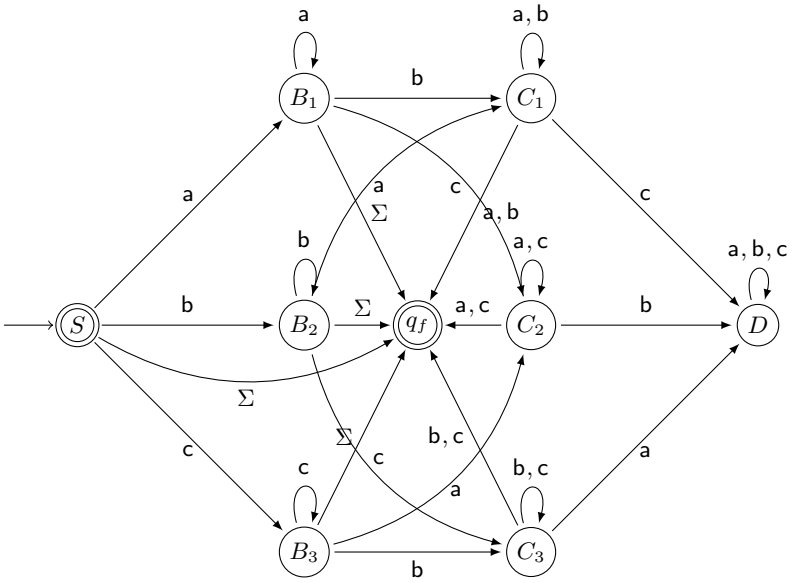
(b) Anwendung der Potenzmengenkonstruktion:



(c) Jeder Zustand entspricht einem Nicht-Terminal, die Terminale sind gerade das Alphabet Σ, der initiale Zustand wird zum Startsymbol. Zur Vereinfachung benennen wir die Zustände von oben nach unten und links nach rechts von A bis C_3, wie oben gezeichnet. Dann ergibt sich folgende Grammatik

- $\mathsf{S} \to \mathsf{aB}_1 \mid \mathsf{bB}_2 \mid \mathsf{cB}_3 \mid \mathsf{a} \mid \mathsf{b} \mid \mathsf{c} \mid \varepsilon$
- $\mathsf{B}_1 \to \mathsf{a} \mid \mathsf{b} \mid \mathsf{c} \mid \mathsf{aB}_1 \mid \mathsf{bC}_1 \mid \mathsf{cC}_2$
- $\mathsf{B}_2 \to \mathsf{a} \mid \mathsf{b} \mid \mathsf{c} \mid \mathsf{bB}_2 \mid \mathsf{aC}_1 \mid \mathsf{cC}_3$
- $\mathsf{B}_3 \to \mathsf{a} \mid \mathsf{b} \mid \mathsf{c} \mid \mathsf{cB}_3 \mid \mathsf{aC}_2 \mid \mathsf{bC}_3$
- $\mathsf{C}_1 \to \mathsf{a} \mid \mathsf{b} \mid \mathsf{aC}_1 \mid \mathsf{bC}_1 \mid \mathsf{cD}$
- $\mathsf{C}_2 \to \mathsf{a} \mid \mathsf{c} \mid \mathsf{aC}_2 \mid \mathsf{cC}_2 \mid \mathsf{bD}$
- $\mathsf{C}_3 \to \mathsf{b} \mid \mathsf{c} \mid \mathsf{bC}_3 \mid \mathsf{cC}_3 \mid \mathsf{aD}$
- $\mathsf{D} \to \mathsf{aD} \mid \mathsf{bD} \mid \mathsf{cD}$

(d) Wir erhalten folgenden Automaten:



(e) Diese Übersetzung (NFA → DFA → Grammatik → NFA) induziert eine Normalform, in der es maximal 2 Endzustände gibt. Die Anzahl der Zustände entspricht der Anzahl der Variablen in der Grammatik plus den dedizierten q_f Zustand. Durch die Potenzmengenkonstruktion von N zu D wird die Zustandzahl vergrößert (möglicherweise exponentiell), und nachdem die Grammatik für jeden Zustand des DFA ein Nichtterminal hat, ist die Zustandszahl von N′ möglicherweise exponentiell größer als die von N.

AUFGABE 2.6. Stufe D

Vervollständigen Sie den Beweis von Satz 3.14 aus der Vorlesung:

Für jede rechtslineare Grammatik G gibt es einen NFA M mit $L(G) = L(M)$.

Lösungsskizze

Sei $G = (\mathsf{V}, \Sigma, \mathsf{P}, \mathsf{S})$ eine rechtslineare Grammatik und $\mathsf{N} = (\mathsf{Q}, \Sigma, \delta, \mathsf{S}, \mathsf{F})$ der dazu im Beweis von Satz 3.14 konstruierte ε-NFA. N enthält genau dann eine ε-Kante (S, ε, q_f) wenn $(\mathsf{S} \to \varepsilon) \in \mathsf{P}$. (Die Definition rechtslinearer Grammatiken erlaubt keine weiteren ε-Produktionen)

Damit kann N in einen NFA M überführt werden indem man gegebenenfalls die ε-Kante entfernt und S zum Endzustand macht: $\mathsf{M} := (\mathsf{Q}, \Sigma, \delta', \mathsf{S}, \mathsf{F}')$ mit $\delta' = \delta \setminus \{(\mathsf{S}, \varepsilon, q_f)\}$ und $\mathsf{F}' = \mathsf{F} \cup \{\mathsf{S} \mid (\mathsf{S} \to \varepsilon) \in \mathsf{P}\}$.

Wir beweisen zunächst die folgende verallgemeinerte Ausssage für alle $n > 0$:

$$\forall V' \subseteq V.\ \forall \omega.\ |\omega| = n \implies q_f \in \hat{\delta}(V', \omega) \iff (\exists X \in V'.\ X \to_G^* \omega)$$

Beweis per Induktion über n. Induktionsbasis: $n = 1$. Sei ω mit $|\omega| = 1$ beliebig, aber fix. D.h. $\omega = a$ für ein $a \in \Sigma$.

$$\begin{aligned}
& q_f \in \hat{\bar{\delta}}(V', \omega) \\
\iff & \exists X \in V'.\ q_f \in \hat{\delta}(X, \omega) \\
\iff & \exists X \in V'.\ q_f \in \delta'(X, a) \\
\iff & \exists X \in V'.\ (X \to a) \in P \vee ((X \to aS) \in P \wedge (S \to \varepsilon) \in P) && \text{(Def. } \delta, \delta') \ (*) \\
\iff & \exists X \in V'.\ X \to_G^* \omega
\end{aligned}$$

Induktionsschritt: Wir fixieren ein beliebiges $n > 0$ und nehmen an die Aussage gilt. Wir müssen die Aussage für $n + 1$ zeigen.

Seien V' mit $V' \subseteq V$, ω mit $|\omega| = n$ und $a \in \Sigma$ beliebig, aber fix. Dann gilt:

$$\begin{aligned}
& q_f \in \hat{\bar{\delta}}(V', a\omega) \\
\iff & q_f \in \hat{\bar{\delta}}(\bar{\delta}'(V', a), \omega) \\
\iff & q_f \in \hat{\bar{\delta}}(\bar{\delta}'(V', a) \setminus \{q_f\}, \omega) && (\omega \neq \varepsilon) \\
\iff & \exists Y \in \delta'(V', a).\ Y \to_G^* \omega && \text{(IH)} \\
\iff & \exists Y \in \delta(V', a).\ Y \to_G^* \omega && (**) \\
\iff & \exists Y \in V.\ \exists X \in V'.\ (X \to aY) \in P \wedge Y \to_G^* \omega && \text{(Def. } \delta) \\
\iff & \exists X \in V'.\ X \to_G^* a\omega
\end{aligned}$$

Dabei ist es gerechtfertigt die IH anzuwenden, da $\delta'(V', a) \setminus \{q_f\} \subseteq V$ wegen der Def. von δ. In den mit $(*)$ und $(**)$ markierten Schritten benutzen wir, dass die einzige mögliche ε-Transition von S nach q_f geht. Zusätzlich wissen wir, dass q_f keine ausgehenden Transitionen hat.

Damit ergibt sich:

$$F \cap \hat{\delta}(S, \omega) \neq \emptyset \iff S \to_G^* \omega \text{ falls } |\omega| > 0$$

Das leere Wort ist in der Sprache gdw. $(S \to \varepsilon) \in P$ gdw. S Endzustand von M ist gdw. M das leere Wort akzeptiert.

AUFGABE 2.7. Stufe D

Entscheiden Sie, ob die folgenden Aussagen korrekt sind oder nicht und begründen Sie Ihre Behauptung, indem Sie entweder ein Gegenbeispiel oder eine passende Konstruktion angeben.

Für jeden NFA $\mathsf{N} = (\mathsf{Q}, \Sigma, \delta, \mathsf{q}_0, \mathsf{F})$ gibt es einen NFA $\mathsf{N}' = (\mathsf{Q}', \Sigma', \delta', \mathsf{q}_0', \mathsf{F}')$ mit $\mathsf{L}(\mathsf{N}) = \mathsf{L}(\mathsf{N}')$ und ...

(a) der Startzustand hat keine eingehenden Kanten.
(b) kein Endzustand hat eine ausgehende Kante.
(c) für jeden Zustand gilt: alle eingehenden Kanten sind mit demselben Zeichen beschriftet.
(d) für jeden Zustand gilt: alle ausgehenden Kanten sind mit demselben Zeichen beschriftet.

Lösungsskizze

(a) Ja: Man erstellt vom Startzustand q_0 eine Kopie q_0', die nur die ausgehenden Kanten von q_0 erbt. Danach wird q_0 zu einem normalen Zustand, während q_0' zu einem Startzustand wird. Formal: $\mathsf{N}' = (\mathsf{Q}', \Sigma', \delta', \mathsf{q}_0', \mathsf{F}')$, wobei
   - $\mathsf{Q}' = \mathsf{Q} \uplus \{\mathsf{q}_0'\}$,
   - $\delta' = \delta \uplus \{(\mathsf{q}_0', x, q) \mid (\mathsf{q}_0, x, q) \in \delta\}$
   - $\mathsf{F}' = \mathsf{F} \cup \{\mathsf{q}_0' \mid \mathsf{q}_0 \in \mathsf{F}\}$
   - $\Sigma' = \Sigma$

(b) Nein: Betrachte einen NFA N für $\mathsf{L} = \{\varepsilon, \mathsf{a}\}$.

(c) Ja: man spaltet jeden Zustand $q \in \mathsf{Q}$ nach dem Buchstaben der eingehenden Kanten auf, d.h. aus q macht man die Zustände (q, x) für jedes $x \in \Sigma$.
Danach setzt man $\delta'((q, x), y, (q', y)) := \delta(q, y, q')$.
Wird dabei der Startzustand aufgespalten wählt man einen beliebigen daraus hervorgehenden Zustand als neuen Startzustand.

(d) Nein: Betrachte einen NFA N mit $\mathsf{L}(\mathsf{N}) = \{\mathsf{a}, \mathsf{b}\} = \Sigma$. Dann gilt $\delta(\mathsf{q}_0, \mathsf{a}) \cap \mathsf{F} \neq \emptyset$ und $\delta(\mathsf{q}_0, \mathsf{b}) \cap \mathsf{F} \neq \emptyset$.

Definition (Substring, Superstring und Superstringsprache)

Sei Σ ein Alphabet und seien v, w Wörter über Σ^*.

(a) v ist ein *Substring* von w genau dann, wenn es Wörter $v_1, v_2 \in \Sigma^*$ gibt mit $w = v_1 v v_2$. Wir schreiben $v \trianglelefteq w$ für v ist ein Substring von w.
(b) Wir nennen w *Superstring* von v, wenn $v \trianglelefteq w$
(c) Die *Superstring-Sprache* für ein Wort $v \in \Sigma^*$ ist definiert als $\mathsf{L}_{v\trianglelefteq} := \{u \in \Sigma^* \mid v \trianglelefteq u\}$

AUFGABE 2.8. Stufe E

Sei $\Sigma = \{\mathsf{m}, \mathsf{i}, \mathsf{a}\}$ das Alphabet.

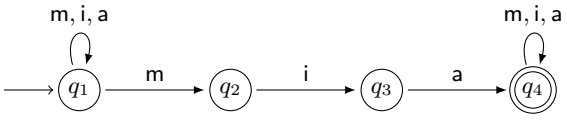
(a) Sei $w = \mathsf{mia}$ und $v = \mathsf{mammamia}$.
   (i) Geben Sie einen NFA an, der die Sprache $\mathsf{L}_{w\trianglelefteq}$ akzeptiert.
   (ii) Geben Sie einen DFA an, der die Sprache $\mathsf{L}_{w\trianglelefteq}$ akzeptiert und nicht mehr Zustände als Ihr NFA aus der vorherigen Teilaufgabe hat.
   (iii) Geben Sie einen NFA an, der die Sprache $\mathsf{L}_{v\trianglelefteq}$ akzeptiert.
   (iv) Geben Sie einen DFA an, der die die Sprache $\mathsf{L}_{v\trianglelefteq}$ akzeptiert und dabei nicht mehr Zustände als Ihr NFA aus der vorherigen Teilaufgabe hat.

(b) Beschreiben Sie die Unterschiede und Gemeinsamkeiten der beiden NFAs und DFAs aus der vorherigen Teilaufgabe. Erklären Sie in möglichst einfacher Sprache, wie Sie es erreicht haben, dass die NFAs und DFAs jeweils die gleiche Anzahl an Zuständen hatten.

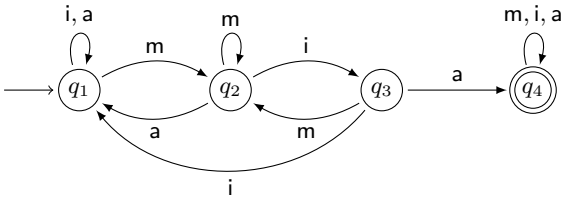
(c) Entwickeln Sie eine allgemeine Konstruktionsvorschrift für NFAs und DFAs, die die Sprache $\mathsf{L}_{w\trianglelefteq}$ für ein beliebiges Wort $w \in \Sigma^*$. Nutzen Sie dafür ihre Beobachtungen aus den vorherigen Aufgabenteilen.

Lösungsskizze

(a) (i) NFA, der $\mathsf{L}_{w \trianglelefteq}$ akzeptiert:



(ii) DFA, der $\mathsf{L}_{w \trianglelefteq}$ akzeptiert:



(iii) Der NFA zur Superstring-Sprache von v hat 9 Zustände und ist ähnlich aufgebaut wie der NFA in Aufgabenteil (i). Es gibt eine Sequenz zum Lesen von v und am Anfangs- und am Endzustand gibt es eine Schleife zum Lesen von m, i und a.

(iv) Ähnlich wie Aufgabenteil (ii) muss man sich überlegen, zu welchem Zustand man zurückgehen muss, wenn man einen Buchstaben liest, der nicht zu mammamia gehört. Hat man zum Beispiel ein m gelesen, so kann entweder mammamia mit einem a fortgesetzt werden, es könnte ein weiteres m gelesen werden, was wiederum dann der Anfang des Worts v sein könnte oder es wird ein i gelesen, welches zum Startzustand zurückführt, da gerade nicht ein Teil von v gelesen worden sein kann.

(b) Der NFA rät, wann das Wort gelesen wird. Wir müssen daher anders als beim DFA nicht sicherstellen, dass wir bei Buchstaben, die nicht unmittelbar zum Wort gehören können, ob sie Teil einer anderen Teilsequenz sind.

(c) Die Konstruktionsvorschrift für NFAs $\mathsf{N} = (\mathsf{Q}, \Sigma, \delta, \mathsf{q}_0, \mathsf{F})$ zu einem Wort w über dem Alphabet Σ lautet:

(i) $\mathsf{Q} = \{u \mid \exists v \in \Sigma^*.\ uv = w\}$

(ii) $\delta = \{(\varepsilon, x, \varepsilon), (w, x, w) \mid x \in \Sigma\} \cup \{(u, x, ux) \mid x \in \Sigma \wedge \exists v \in \Sigma^*.\ uxv = w\}$

(iii) $\mathsf{q}_0 = \varepsilon$

(iv) $\mathsf{F} = \{w\}$

Um einen passenden DFA zu erhalten, kann man die Potenzmengenkonstruktion anwenden.
Man kann aber auch den (minimalen) DFA $\mathsf{D} = (\mathsf{Q}, \Sigma, \delta, \mathsf{q}_0, \mathsf{F})$ direkt konstruieren: Die Zustände sind alle Präfixe von w und die Konstruktion versucht das nächste Zeichen zu lesen um aus dem bis jetzt gelesen Präfix ein neues Präfix von w zu konstruieren. Falls dies nicht möglich ist, springt der DFA zu dem Präfix, das das längste Suffix von aktuell gelesenem Wort ist, zurück. Das Verfahren wird in `https://dl.acm.org/citation.cfm?doid=360825.360855` beschrieben.