Grundlagen der theoretischen Informatik

Kurt Sieber

Fakultät IV, Department ETI
Universität Siegen

SS 2013

Vorlesung vom 23.04.2013

Als Folgerung erhalten wir

Satz 2.21 $\mathcal{L}_{reg} \subseteq \mathcal{L}_{EA}$.

Beweis:

Eine Sprache ist regulär, wenn sie sich duch wiederholte Anwendung von $\cup$, $\circ$ und * aus endlichen Mengen aufbauen lässt.

Da $\mathcal{L}_{EA}$ abgeschlossen ist unter $\cup$, $\circ$ und *, bleibt nur noch zu zeigen, dass $\mathcal{L}_{EA}$ alle endlichen Teilmengen von Σ^* enthält.

Zunächst gilt dies für die einelementigen Mengen:

- $\{\varepsilon\} \in \mathcal{L}_{EA}$ nach Satz 2.20.
- Ist $w = a_1 \dots a_n \neq \varepsilon$, so ist $\{w\} = \{a_1\} \circ \dots \circ \{a_n\} \in \mathcal{L}_{EA}$ nach Satz 2.20.

Daraus folgt es für alle endlichen Mengen:

- $\emptyset \in \mathcal{L}_{EA}$ nach Satz 2.20.
- Ist $L = \{w_1, \dots, w_n\} \neq \emptyset$, so ist $L = \{w_1\} \cup \dots \cup \{w_n\} \in \mathcal{L}_{EA}$ nach Satz 2.20.

Endliche Automaten

Die Sätze 2.20 und 2.21 (und ihre Beweise) sind wichtig für die Praxis (Compilerbau).

Denn sie liefern ein Verfahren, um aus der 'Mengenbeschreibung' einer regulären Sprache einen endlichen Automaten zu konstruieren.

Ein solches Verfahren wird in Scanner-Generatoren (z.B. Lex) verwendet, wobei man dort natürlich mehr auf die Effizienz der Algorithmen achten muss (vgl. Vorlesung Compilerbau) als wir es hier tun.

Als Eingabe verlangt ein Scanner-Generator sogenannte *reguläre Ausdrücke*, das sind formale Versionen unserer Mengenbeschreibungen, die wir später noch einführen werden.

Weitere Abschlusseigenschaften von $\mathcal{L}_{EA}$

Satz 2.22 *$\mathcal{L}_{EA}$ ist abgeschlossen unter Komplement, Durchschnitt, Mengendifferenz, Potenzierung und Spiegelung, d.h. wenn die Sprachen L_1 und L_2 von endlichen Automaten erkannt werden, dann werden auch $L_1 \cap L_2$, $\Sigma^* \setminus L_1$, $L_1 \setminus L_2$, $L_1^n (n \geq 0)$ und L_1^R von endlichen Automaten erkannt (und es gibt jeweils einen Algorithmus, mit dem man den neuen Automaten konstruieren kann).*

Beweis:

- Komplement:
  s. Übungsblatt 2, Aufgabe 4

Endliche Automaten

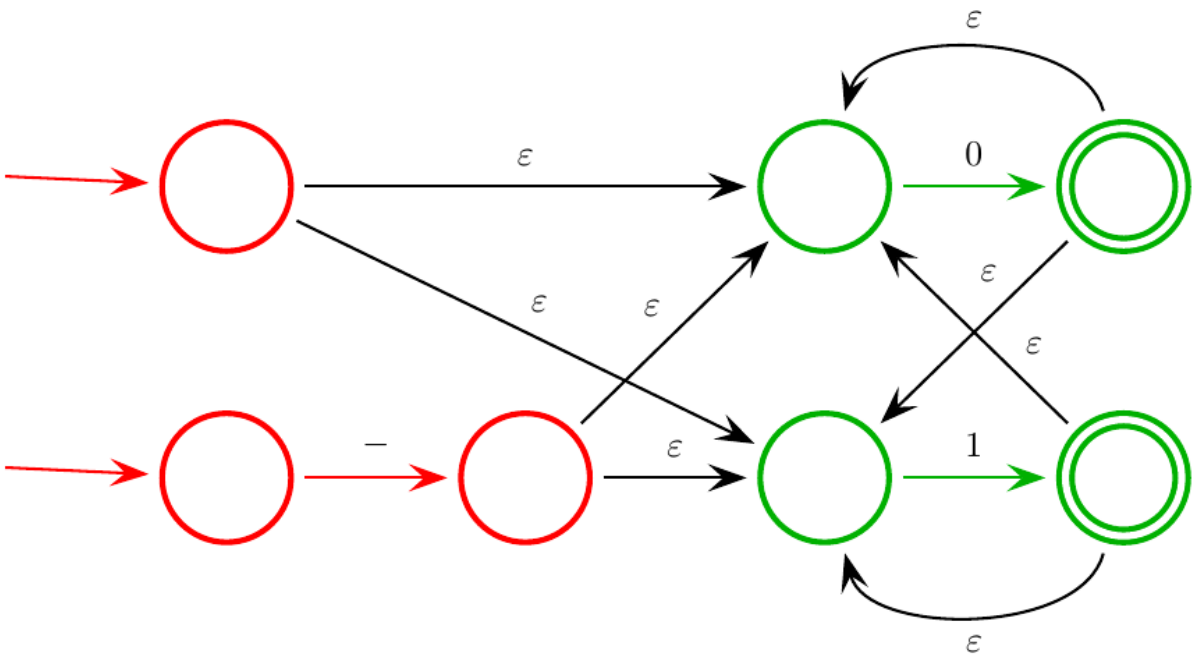
- Durchschnitt:
  Wenn $L_1, L_2 \in \mathcal{L}_{EA}$, dann gilt $L_1 \cap L_2 = \Sigma^* \setminus ((\Sigma^* \setminus L_1) \cup (\Sigma^* \setminus L_2)) \in \mathcal{L}_{EA}$ wegen der Abgeschlossenheit unter Vereinigung und Komplement.
- Mengendifferenz:
  Wenn $L_1, L_2 \in \mathcal{L}_{EA}$, dann gilt $L_1 \setminus L_2 = L_1 \cap (\Sigma^* \setminus L_2) \in \mathcal{L}_{EA}$ wegen der Abgeschlossenheit unter Durchschnitt und Komplement.
- Potenzierung:
  Wenn $L_1 \in \mathcal{L}_{EA}$, dann gilt $L_1^n = L_1 \circ \ldots \circ L_1 \in \mathcal{L}_{EA}$ für alle $n \geq 1$ wegen der Abgeschlossenheit unter $\circ$, und $L_1^0 = \{\varepsilon\} \in \mathcal{L}_{EA}$ gilt sowieso.
- Spiegelung:
  s. Übungsblatt 2, Aufgabe 4

Die Beweise der Sätze 2.20, 2.21 und 2.22 liefern uns Algorithmen zur Konstruktion von ε-NDEAs aus Mengenausdrücken.

Beispiel:
Mit den Konstruktionen für $\{\varepsilon\}$, $\{a\}$ und $\cup$ erhält man ε-NDEAs für die Sprachen $\{\varepsilon, -\}$ und $\{0, 1\}$

Endliche Automaten

Daraus ergibt sich mit den Konstruktionen für $^+$ und $\circ$ ein ε-NDEA für die Sprache $L_{bin} = \{\varepsilon, -\} \circ \{0,1\}^+$ aller Binärdarstellungen ganzer Zahlen.



Für diesen könnte man jetzt wieder einen äquivalenten NDEA bzw. DEA konstruieren.

Zum Beweis von $\mathcal{L}_{reg} = \mathcal{L}_{EA}$ fehlt noch

Satz 2.23 $\mathcal{L}_{EA} \subseteq \mathcal{L}_{reg}$

Beweis:

Sei $A = (\Sigma, Q, s, F, \delta)$ ein DEA mit $Q = \{q_1, \ldots, q_n\}$ und $s = q_1$.

Es ist zu zeigen, dass $L(A)$ regulär ist.

Dazu betrachten wir—für alle $i, j \in \{1, \ldots, n\}$—die Sprachen

$$L_{ij} = \{w \in \Sigma^* \mid (q_i, w) \vdash_A^* (q_j, \varepsilon)\}$$

$L(A)$ lässt sich als (endliche) Vereinigung einiger dieser Sprachen L_{ij} darstellen, nämlich:

$$\begin{aligned} L(A) &= \{w \in \Sigma^* \mid \text{ es existiert ein } q_j \in F \text{ mit } (q_1, w) \vdash_A^* (q_j, \varepsilon)\} \\ &= \bigcup_{q_j \in F} L_{1j} \end{aligned}$$

Deshalb genügt es zu zeigen, dass die Sprachen L_{ij} regulär sind.

Dazu betrachten wir eine weitere 'Verfeinerung' der Sprachen L_{ij}.

Für jedes $k \in \{1, \ldots, n+1\}$ definieren wir

$$L_{ij}^k = \{w \in \Sigma^* \mid (q_i, w) \vdash_A^* (q_j, \varepsilon), \text{ wobei in } \vdash_A^* \text{ nur Zwischenzustände } q_l \text{ mit } l < k \text{ benutzt werden}\}$$

Dann gilt $L_{ij} = L_{ij}^{n+1}$ (weil $l < n+1$ keine Einschränkung ist),

und die Sprachen L_{ij}^k lassen sich durch Induktion über k definieren:

$$L_{ij}^1 = \{w \in \Sigma^* \mid (q_i, w) \vdash_A^* (q_j, \varepsilon) \quad \textit{ohne} \quad \text{Zwischenzustände}\}$$

$$= \begin{cases} \{a \in \Sigma \mid \delta(q_i, a) = q_j\} \text{ falls } i \neq j \\ \{a \in \Sigma \mid \delta(q_i, a) = q_j\} \cup \{\varepsilon\} \text{ falls } i = j \end{cases}$$

$L_{ij}^{k+1} = L_{ij}^{k} \cup L_{ik}^{k} \circ (L_{kk}^{k})^* \circ L_{kj}^{k}$

Diese Gleichung erhält man durch folgende Überlegung:

Wenn $w \in L_{ij}^{k+1}$, dann gibt es zwei Möglichkeiten:

Entweder q_k taucht gar nicht als Zwischenzustand in der Folge $(q_i, w) \vdash_A^* (q_j, \varepsilon)$ auf.

Dann gilt schon $w \in L_{ij}^{k}$.

Oder wir können die Folge $(q_i, w) \vdash_A^* (q_j, \varepsilon)$ überall dort unterteilen, wo q_k auftaucht.

Dann erhalten wir eine Zerlegung $w = w_1 \dots w_m$ $(m \geq 2)$ mit

$(q_i, w_1 \dots w_m) \vdash_A^* (q_k, w_2 \dots w_m) \vdash_A^* \dots \vdash_A^* (q_k, w_m) \vdash_A^* (q_j, \varepsilon)$

in der alle $\vdash_A^*$ nur noch Zwischenzustände q_l mit $l < k$ enthalten.

Mit Lemma 2.9 folgt dann

$$\begin{aligned}(q_i, w_1) &\vdash_A^* (q_k, \varepsilon)\\ (q_k, w_l) &\vdash_A^* (q_k, \varepsilon) \text{ für } l = 2, \ldots, m-1\\ (q_k, w_m) &\vdash_A^* (q_j, \varepsilon)\end{aligned}$$

wobei die $\vdash_A^*$ immer noch die gleichen Zwischenzustände enthalten, also nur Zustände q_l mit $l < k$.

Also gilt $w_1 \in L_{ik}^k$, $w_2, \ldots, w_{m-1} \in L_{kk}^k$ und $w_m \in L_{kj}^k$

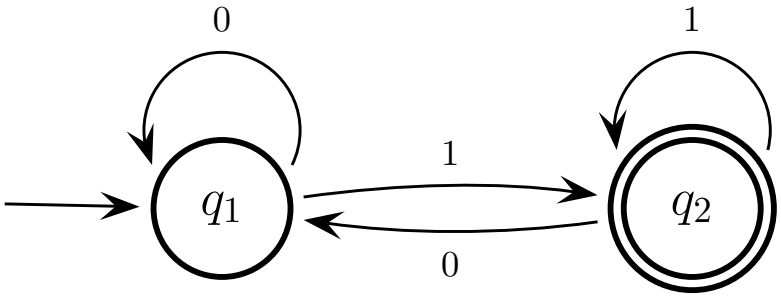
und damit $w = w_1 \ldots w_m \in L_{ik}^k \circ (L_{kk}^k)^* \circ L_{kj}^k$.

Damit ist '$\subseteq$' bewiesen und '$\supseteq$' ergibt sich ähnlich (einfacher!), indem man Folgen von Übergangsschritten *zusammensetzt*.

Da alle L_{ij}^1 endlich sind, und jedes L_{ij}^{k+1} durch Anwendung der Operationen $\cup$, $\circ$ und * aus einigen $L_{i'j'}^k$ entsteht, folgt (durch Induktion über k), dass alle L_{ij}^k regulär sind. $\square$

Auch der Beweis von Satz 2.23 ist konstruktiv. Er zeigt uns, wie wir einen 'Mengenausdruck' für die von einem DEA erkannte Sprache finden können.

Beispiel: Sei A der DEA



Wir bestimmen einen Mengenausdruck für $L(A)$, wobei wir (schon bei den Zwischenrechnungen) Vereinfachungen durchführen, damit der Gesamtausdruck nicht zu groß wird.

$L(A) = L_{12}$ da q_2 einziger Endzustand ist.

$= L_{12}^3$ da A zwei Zustände hat.

$$
\begin{aligned}
L_{12}^3 &= L_{12}^2 \cup L_{12}^2 \circ (L_{22}^2)^* \circ L_{22}^2 && \text{laut Gleichung für } L_{ij}^{k+1} \\
&= L_{12}^2 \cup L_{12}^2 \circ (L_{22}^2)^+ && \text{da } L^* \circ L = L^+ \\
&= L_{12}^2 \circ \{\varepsilon\} \cup L_{12}^2 \circ (L_{22}^2)^+ && \text{da } \{\varepsilon\} \text{ neutrales Element für } \circ \text{ ist} \\
&= L_{12}^2 \circ (\{\varepsilon\} \cup (L_{22}^2)^+) && \text{da } \circ \text{ distributiv über } \cup \text{ ist} \\
&= L_{12}^2 \circ (L_{22}^2)^* && \text{da } \{\varepsilon\} \cup L^+ = L^*
\end{aligned}
$$

$$
\begin{aligned}
L_{12}^2 &= L_{12}^1 \cup L_{11}^1 \circ (L_{11}^1)^* \circ L_{12}^1 && \text{laut Gleichung für } L_{ij}^{k+1} \\
&= \{1\} \cup \{\varepsilon, 0\} \circ \{\varepsilon, 0\}^* \circ \{1\} && \text{laut Gleichung für } L_{ij}^1 \\
&= \{1\} \cup \{\varepsilon, 0\}^+ \circ \{1\} && \text{da } L \circ L^* = L^+ \\
&= \{1\} \cup \{0\}^* \circ \{1\} && \text{da } (L \cup \{\varepsilon\})^+ = L^* \\
&= \{0\}^* \circ \{1\} && \text{da } \{1\} \subseteq \{0\}^* \circ \{1\}
\end{aligned}
$$

Reguläre Sprachen

$$
\begin{aligned}
L_{22}^2 &= L_{22}^1 \cup L_{21}^1 \circ (L_{11}^1)^* \circ L_{12}^1 && \text{laut Gleichung für } L_{ij}^{k+1} \\
&= \{\varepsilon, 1\} \cup \{0\} \circ \{\varepsilon, 0\}^* \circ \{1\} && \text{laut Gleichung für } L_{ij}^1 \\
&= \{\varepsilon, 1\} \cup \{0\} \circ \{0\}^* \circ \{1\} && \text{da } (L \cup \{\varepsilon\})^* = L^* \\
&= \{\varepsilon, 1\} \cup \{0\}^+ \circ \{1\} && \text{da } L \circ L^* = L^+
\end{aligned}
$$

Also gilt $L(A) = \{0\}^* \circ \{1\} \circ (\{\varepsilon, 1\} \cup \{0\}^+ \circ \{1\})^*$

Natürlich ließe sich dieser Ausdruck noch weiter vereinfachen, letztendlich zu $\{0, 1\}^* \circ \{1\}$, weil das die von A erkannte Sprache ist. In seiner jetzigen Form ist er aber aufschlussreicher, weil er noch die Konstruktionsidee aus dem Beweis von Satz 2.23 erkennen lässt:

Die Menge $\{0\}^* \circ \{1\}$ enthält alle Wörter, die von q_1 nach q_2 führen, wobei nur q_1 als Zwischenzustand benutzt wird. Und die Menge $\{\varepsilon, 1\} \cup \{0\}^+ \circ \{1\}$ enthält alle Wörter, die von q_2 nach q_2 führen, wobei auch wieder nur q_1 als Zwischenzustand benutzt wird.

Reguläre Sprachen

Mit den Sätzen 2.21 und 2.23 ist bewiesen, dass

$$\mathcal{L}_{reg} = \mathcal{L}_{EA}$$

und deshalb sprechen wir ab jetzt nur noch von *regulären Sprachen*.

Mit anderen Worten:

Wir haben bisher nur *eine* Sprachklasse kennengelernt, nämlich die Klasse der regulären Sprachen, aber wir haben unterschiedliche Formalismen zur Darstellung regulärer Sprachen:

- DEAs
- NDEAs
- ε-NDEAs
- und 'Mengenausdrücke'

Problem: Für 'Mengenausdrücke' haben wir keine formale (sondern nur die übliche mathematische) Schreibweise.

Reguläre Sprachen

Deshalb: Reguläre Ausdrücke

Definition 2.24 *Sei Σ ein Alphabet. Die Menge aller regulären Ausdrücke über Σ ist induktiv definiert durch:*

1. *$\emptyset$ ist ein regulärer Ausdruck über Σ.*
2. *Jedes $a \in \Sigma$ ist ein regulärer Ausdruck über Σ.*
3. *Wenn α und β reguläre Ausdrücke über Σ sind, dann sind auch $(\alpha\beta)$, $(\alpha \mid \beta)$ und $(\alpha^\star)$ reguläre Ausdrücke über Σ.*

Ein regulärer Ausdruck über Σ ist also ein Wort über dem Alphabet $\Sigma' = \Sigma \cup \{\emptyset, (,), |, \star\}$, das nach den Regeln 1. bis 3. aufgebaut ist.

Damit ist die *Syntax* der regulären Ausdrücke festgelegt.

Es fehlt noch die *Semantik*.

Reguläre Sprachen

Ein regulärer Ausdruck soll natürlich die Sprache beschreiben, die durch den 'entsprechenden Mengenausdruck' definiert ist.

Definition 2.25 *Die von einem regulären Ausdruck* α beschriebene Sprache $L(\alpha)$ *ist durch Induktion über die Größe von* α *wie folgt definiert:*

1. $L(\varnothing) = \emptyset$
2. $L(a) = \{a\}$
3. $L((\alpha\beta)) = L(\alpha) \circ L(\beta)$

   $L((\alpha \mid \beta)) = L(\alpha) \cup L(\beta)$

   $L((\alpha^\star)) = (L(\alpha))^*$

Man beachte:
Links stehen *Zeichen* $\varnothing$, $\mid$ und *.
Rechts stehen mathematische Schreibweisen: $\emptyset$, $\{\ \}$, $\cup$, $\circ$ und *.

Vereinbarung zur Einsparung von Klammern

- Die Konkatenation bindet stärker als '|'.
- '$\star$' bindet am stärksten.
- Konkatenation und '|' sind linksassoziativ, d.h. $\alpha\beta\gamma$ steht für $(\alpha\beta)\gamma$ und $\alpha \mid \beta \mid \gamma$ für $(\alpha \mid \beta) \mid \gamma$.

  (Ebenso gut könnte man Rechtsassoziativität vereinbaren, da die Mengenoperationen $\cup$ und $\circ$ sowieso assoziativ sind.)

Beispiele

- $L_{nat} = \{0, \ldots, 9\}^+ = L((0 \mid \ldots \mid 9)\,(0 \mid \ldots \mid 9)^\star)$
- $L_{int} = \{\varepsilon, -\} \circ L_{nat} = L((\varnothing^\star \mid -)(0 \mid \ldots \mid 9)\,(0 \mid \ldots \mid 9)^\star)$
- $L(a^\star\, b^\star) = \{a\}^* \circ \{b\}^* = \{a^m\, b^n \mid m, n \geq 0\}$
- $L((ab)^\star) = (\{a\} \circ \{b\})^* = \{ab\}^* = \{(ab)^n \mid n \geq 0\}$

Reguläre Sprachen

Per Definition der Semantik regulärer Ausdrücke ist es klar, dass die von einem regulären Ausdruck beschriebene Sprache stets regulär ist. Umgekehrt lässt sich jede reguläre Sprache durch einen regulären Ausdruck beschreiben, denn:

- Alle endlichen Sprachen erhält man mit Konkatenation und Vereinigung aus $\emptyset$, $\{\varepsilon\}$ und den Mengen $\{a\}$ mit $a \in \Sigma$ (vgl. Beweis zu Satz 2.23).
- Diese Mengen kann man durch reguläre Ausdrücke beschreiben, nämlich $\emptyset = L(\varnothing)$, $\{\varepsilon\} = \emptyset^* = L(\varnothing^\star)$ und $\{a\} = L(a)$.

Also gilt:

Satz 2.26 *Eine Sprache ist genau dann regulär, wenn sie sich durch einen regulären Ausdruck beschreiben lässt.*

Reguläre Sprachen

Mit den regulären Ausdrücken haben wir eine *formale Syntax* zur Beschreibung regulärer Sprachen (anstelle der "üblichen mathematischen Schreibweise" bei Mengenausdrücken).

Natürlich übertragen sich die bisherigen Beobachtungen über Mengenausdrücke unmittelbar auf reguläre Ausdrücke, insbesondere:

Die Konstruktion von ε-NDEAs aus dem Beweis zu Satz 2.20 (und Satz 2.21) liefert zu jedem regulären Ausdruck einen äquivalenten ε-NDEA.

Die Konstruktion der Sprachen L_{ij} aus dem Beweis von Satz 2.23 lässt sich so abwandeln, dass sie zu jedem DEA einen äquivalenten regulären Ausdruck liefert.

Damit haben wir *Übersetzungsalgorithmen*, um all die unterschiedlichen Darstellungen regulärer Sprachen (DEAs, NDEAs, ε-NDEAs und reguläre Ausdrücke) ineinander überzuführen.

Bezug zur Praxis

Reguläre Ausdrücke werden benutzt

- als Eingabe für Scanner-Generatoren (Compilerbau)
- als Suchmuster in Editoren und Suchmaschinen
- zur Textverarbeitung in Programmiersprachen

Meistens hat man eine erweiterte Syntax für reguläre Ausdrücke:

- zusätzliche Zeichen wie ϵ, $^+$ oder ein Zeichen für das Komplement
- Schreibweisen für endliche Zeichenmengen, z.B. [a-$z$$A$-$Z$]
- Einführung von Namen zur Wiederverwendung eines Ausdrucks, z.B $nat = [0\text{-}9]^+$, $int = (\epsilon \mid -)\, nat$

Intern werden reguläre Ausdrücke in EAs umgewandelt,
im Prinzip nach unserer Theorie, aber mit effizienteren Algorithmen
(z.B. direkt vom regulären Ausdruck zum DEA).

Reguläre Sprachen

Nachdem wir wissen, dass $\mathcal{L}_{reg} = \mathcal{L}_{EA}$ gilt, können wir jetzt all unsere Formalismen und auch die bewiesenen Abschlusseigenschaften kombinieren, um zu zeigen, dass eine Sprache regulär ist.

Beispiel:
Sei $L = \{w \in \{0,1\}^* \mid w \text{ enthält } 010, \text{ aber nicht } 110\}$.
Dann gilt $L = L((0 \mid 1)^\star 010\,(0 \mid 1)^\star) \setminus L((0 \mid 1)^\star 110\,(0 \mid 1)^\star)$,
also ist L regulär.

Aber auch, um zu zeigen, dass eine Sprache *nicht* regulär ist.

Beispiel:
Sei $L = \{w \in \{a,b\}^* \mid \#_a(w) = \#_b(w)\}$
(wobei $\#_a(w)$ für die Anzahl der as im Wort w steht).
Dann ist $L \cap L((a^\star b^\star)) = \{a^n b^n \mid n \geq 0\}$,
d.h. wenn L regulär wäre, dann wäre auch $\{a^n b^n \mid n \geq 0\}$ als Durchschnitt zweier regulärer Sprachen wieder regulär.
Aber von dieser Sprache wissen wir schon, dass sie nicht regulär ist.

Entscheidbarkeitsfragen

Neben den Übersetzungsalgorithmen interessieren uns auch *Entscheidungsalgorithmen*. Das sind Algorithmen, die gewisse Fragestellungen stets korrekt mit 'ja' oder 'nein' beantworten, z.B.

- die Frage, ob ein Automat die leere Sprache erkennt
- die Frage, ob zwei Automaten äquivalent sind.

In jedem Fall muss geklärt werden,

- welche Eingabedaten der Algorithmus erhält,
- welche Frage über die Eingabedaten er beantworten soll

  (also wie seine Ausgabe von der Eingabe abhängen soll)

Satz 2.27 *Für jede der folgenden Fragestellungen gibt es einen Entscheidungsalgorithmus.*

1. *Eingabe: Ein DEA A und ein Wort w. Frage: Ist $w \in L(A)$?*

   (das sogenannte Wortproblem für DEAs)

2. *Eingabe: Ein DEA A. Frage: Ist $L(A) = \emptyset$?.*

   (das Leerheitsproblem für DEAs)

3. *Eingabe: Ein DEA A. Frage: Ist $L(A) = \Sigma^*$?.*

4. *Eingabe: Zwei DEAs A_1 und A_2. Frage: Ist $L(A_1) \subseteq L(A_2)$?*

5. *Eingabe: Zwei DEAs A_1 und A_2. Frage: Ist $L(A_1) = L(A_2)$?*

   (das Äquivalenzproblem für DEAs)

Beweis:

1. Der Algorithmus muss nur den Ablauf des DEA A bei Eingabe w simulieren und dann überprüfen, ob der erreichte Zustand ein Endzustand ist.

2. $L(A) = \emptyset$ gilt genau dann, wenn *kein* Endzustand erreichbar ist. Also berechnet man die Menge der erreichbaren Zustände von A und überprüft, ob darunter ein Endzustand ist.

3. Es gilt $L(A) = \Sigma^* \Leftrightarrow \Sigma^* \setminus L(A) = \emptyset$. Also konstruiert man zu A einen DEA $\bar{A}$ mit $L(\bar{A}) = \Sigma^* \setminus L(A)$ (nach dem Verfahren aus der Übung) und überprüft, ob $L(\bar{A}) = \emptyset$.

4. $L(A_1) \subseteq L(A_2)$ gilt genau dann, wenn $L(A_1) \cap (\Sigma^* \setminus L(A_2)) = \emptyset$. Also konstruiert man zu A_1 und A_2 einen DEA A mit $L(A) = L(A_1) \cap (\Sigma^* \setminus L(A_2))$ und überprüft, ob $L(A) = \emptyset$.

5. Um $L(A_1) = L(A_2)$ zu testen, überprüft man, ob $L(A_1) \subseteq L(A_2)$ und $L(A_2) \subseteq L(A_1)$ gilt. □

Natürlich gilt Satz 2.27 auch für NDEAs, ε-NDEAs oder reguläre Ausdrücke anstelle von DEAs.

Denn wir können jeden dieser Formalismen in einen äquivalenten DEA übersetzen und brauchen dann nur noch die entsprechende Frage für den DEA zu beantworten

(weil es ja Fragen über die erkannte Sprache sind).