Grundlagen der theoretischen Informatik

Kurt Sieber

Fakultät IV, Department ETI
Universität Siegen

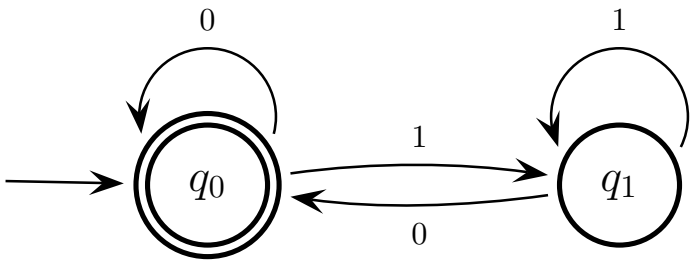
SS 2013

Vorlesung vom 11.04.2013

Definition 2.2 *Ein deterministischer endlicher Automat (kurz: DEA) ist ein 5-Tupel $A = (\Sigma, Q, s, F, \delta)$ mit:*

- Σ *ist ein Alphabet*
- Q *ist eine endliche Menge, deren Elemente wir Zustände nennen*
- $s \in Q$ *ist der sogenannte Startzustand*
- $F \subseteq Q$ *ist die Menge der sogenannten Endzustände oder akzeptierende Zustände*
- $\delta : Q \times \Sigma \to Q$ *ist die sogenannte Übergangsfunktion (totale Funktion!)*

Beispiel (wie oben):



$A = (\Sigma, Q, s, F, \delta)$ mit

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

- $\delta : Q \times \Sigma \to Q$
  $(q_0, 0) \mapsto q_0$
  $(q_0, 1) \mapsto q_1$
  $(q_1, 0) \mapsto q_0$
  $(q_1, 1) \mapsto q_1$

Oder: δ als Tabelle

	0	1
q_0	q_0	q_1
q_1	q_0	q_1

Endliche Automaten

Die bisherige Definition liefert eine *statische Beschreibung* für DEAs. Es ist nur festgelegt, aus welchen "Teilen" ein DEA besteht (Alphabet, Zustände, Startzustand, Endzustände und Übergangsfunktion).

Jetzt muss noch das *dynamische Verhalten* eines DEA beschrieben werden , d.h. die einzelnen *Rechenschritte*, die er beim Lesen eines Wortes durchführt.

Definition 2.3 *Sei $A = (\Sigma, Q, s, F, \delta)$ ein DEA.*

- *Eine Konfiguration von A ist ein Element $(q, w) \in Q \times \Sigma^*$.*
- *Auf der Menge der Konfigurationen von A definieren wir die Übergangsschrittrelation $\vdash_A$ durch*

$$(q, w) \vdash_A (q', w') \quad \Leftrightarrow \quad \text{es existiert ein } a \in \Sigma \text{ mit } w = aw' \text{ und } \delta(q, a) = q'$$

Endliche Automaten

Intuition

- Eine Konfiguration (q, w) beschreibt die Situation des Automaten zu einem bestimmten Zeitpunkt: Er befindet sich aktuell im Zustand q und hat noch das (Rest-)Wort w zu lesen.
- $\vdash_A$ beschreibt den *Übergangsschritt* von einem Zeitpunkt zum nächsten: Das erste Zeichen a des Wortes w wird gelesen (falls $w \neq \varepsilon$) und der Automat wechselt vom aktuellen Zustand q in den Folgezustand $q' = \delta(q, a)$. Vom Wort $w = aw'$ bleibt also das Wort w' übrig.

Warnung

- Die Übergangsfunktion $\delta : Q \times \Sigma \to Q$ gehört zur *statischen* Beschreibung des Automaten A.
- Die Übergangsschrittrelation $\vdash_A \subseteq (Q \times \Sigma^*)^2$ ist zwar mit Hilfe von δ definiert, beschreibt aber das *dynamische* Verhalten des Automaten.

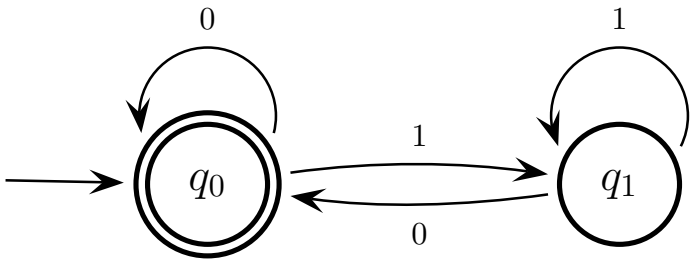
Definition 2.4 *Sei $A = (\Sigma, Q, s, F, \delta)$ ein DEA. Aufbauend auf der Übergangsschrittrelation $\vdash_A$ definieren wir Relationen $\vdash_A^n$ ($n \geq 0$), $\vdash_A^+$ und $\vdash_A^*$ auf der Menge $Q \times \Sigma^*$ aller Konfigurationen durch*

- $(q, w) \vdash_A^n (q', w')$ $\Leftrightarrow$ *es existieren* $(q_0, w_0), \ldots, (q_n, w_n) \in Q \times \Sigma^*$ *mit* $(q, w) = (q_0, w_0), (q', w') = (q_n, w_n)$ *und* $(q_0, w_0) \vdash_A \ldots \vdash_A (q_n, w_n)$.
- $(q, w) \vdash_A^+ (q', w')$ $\Leftrightarrow$ *es existiert ein* $n > 0$ *mit* $(q, w) \vdash_A^n (q', w')$
- $(q, w) \vdash_A^* (q', w')$ $\Leftrightarrow$ *es existiert ein* $n \geq 0$ *mit* $(q, w) \vdash_A^n (q', w')$

$\vdash_A^n$ steht also für eine Folge von n Übergangsschritten, $\vdash_A^+$ für eine nichtleere Folge und $\vdash_A^*$ für eine beliebige (möglicherweise leere) Folge von Übergangsschritten.

Endliche Automaten

Beispiel (wie oben):



Eine mögliche Folge von Übergangsschritten:

$$
\begin{aligned}
(q_0, 1001) &\vdash_A (q_1, 001) && \text{wegen } \delta(q_0, 1) = q_1 \\
&\vdash_A (q_0, 01) && \text{wegen } \delta(q_1, 0) = q_0 \\
&\vdash_A (q_0, 1) && \text{wegen } \delta(q_0, 0) = q_0 \\
&\vdash_A (q_1, \varepsilon) && \text{wegen } \delta(q_0, 1) = q_1
\end{aligned}
$$

Also: $(q_0, 1001) \vdash_A^* (q_1, \varepsilon)$

Oder: $(q_0, 1001) \vdash_A^* (q_0, 1)$

Definition 2.5 *Sei $A = (\Sigma, Q, s, F, \delta)$ ein DEA.*

- *Ein Wort $w \in \Sigma^*$ wird von A akzeptiert genau dann wenn es ein $q \in F$ gibt mit $(s, w) \vdash_A^* (q, \varepsilon)$.*
- *Die von A akzeptierte (oder: erkannte) Sprache $L(A)$ ist definiert durch*

$$\begin{aligned} L(A) &= \{w \in \Sigma^* \mid w \text{ wird von } A \text{ akzeptiert}\} \\ &= \{w \in \Sigma^* \mid \text{es existiert ein } q \in F \text{ mit } (s, w) \vdash_A^* (q, \varepsilon)\} \end{aligned}$$

Beispiel

Für den oben definierten Automaten A gilt:

$$\begin{aligned} L(A) &= \{w \in \{0,1\}^* \mid (q_0, w) \vdash^* (q_0, \varepsilon)\} \\ &= \{w \in \{0,1\}^* \mid w \text{ endet auf } 0\} \cup \{\varepsilon\} \end{aligned}$$

Warum schon wieder ^{n},$^+$,*?
Wo ist das Monoid?

Definition 2.6 *Sei M eine Menge (z.B. $M = Q \times \Sigma^*$). Die* Komposition *$R_1 \circ R_2$ zweier Relationen $R_1, R_2 \subseteq M \times M$ ist definiert durch*

$$R_1 \circ R_2 = \{(x,y) \in M \times M \mid \text{es existiert ein } z \in M \text{ mit } (x,z) \in R_1 \text{ und } (z,y) \in R_2\}$$

- Dann ist $(\wp(M \times M), \circ)$ ein Monoid.
- Neutrales Element ist die *Identität* $id_M = \{(x,x) \mid x \in M\}$.
- Also können wir Potenzen R^n von Relationen $R \subseteq M \times M$ wie üblich definieren (insbesondere $\vdash_A^n$).
- Und weil auch Relationen Mengen sind, definieren wir analog zu Sprachen: $R^+ = \bigcup_{n>0} R^n$ und $R^* = \bigcup_{n\geq 0} R^n$ (insbesondere $\vdash_A^+$ und $\vdash_A^*$)

Lemma 2.7 *Sei M eine Menge und $R \subseteq M \times M$ eine zweistellige Relation. Dann gilt:*

- $R^+ = \bigcup_{n>0} R^n$ *ist die kleinste transitive Relation, die R enthält (d.h. für jede andere transitive Relation R' mit $R \subseteq R'$ gilt auch $R^+ \subseteq R'$.)*
- $R^* = \bigcup_{n\geq 0} R^n$ *die kleinste reflexive transitive Relation, die R enthält (d.h. für jede andere reflexive transitive Relation R' mit $R \subseteq R'$ gilt auch $R^* \subseteq R'$.)*

Definition 2.8

- R^+ *heißt* transitiver Abschluss *der Relation R.*
- R^* *heißt* reflexiver transitiver Abschluss *der Relation R.*

Endliche Automaten

Nächstes Ziel:

"Inhaltliches" Verständnis von endlichen Automaten

Fragestellungen

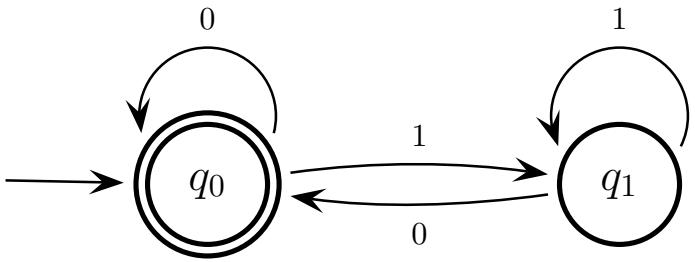
- Wie bestimmt man die Sprache $L(A)$ zu einem vorgegebenen Automaten A?
- Wie beweist man, dass ein Automat tatsächlich die gewünschte Sprache akzeptiert?
- Wie findet man einen Automaten, der eine vorgegebene Sprache akzeptiert?
- Wie findet man heraus, ob es überhaupt einen solchen Automaten für eine vorgegebene Sprache gibt?

Dazu: Grundlegende Überlegungen

- Jeder endliche Automat hat nur ein endliches “Gedächtnis”, nämlich seine endlich vielen Zustände.
- Die einzige Information über das bereits gelesene Anfangsstück eines Wortes steckt im aktuellen Zustand.
- Diese begrenzte Information muss ausreichen, um den Rest des Wortes ‘richtig’ abzuarbeiten.

Endliche Automaten

Beispiel (wie oben):



Welche Information über das bereits gelesene Wort w steckt in den Zuständen q_0 bzw. q_1?

- q_0 bedeutet: Entweder $w = \varepsilon$ oder w endet auf 0.
- q_1 bedeutet: w endet auf 1.

Setzt man diese Bedeutung der Zustände voraus, so ist klar, wie der Automat auf das nächste Zeichen reagieren muss:

- Mit 0 muss er stets in q_0 übergehen, da $w0$ auf 0 endet.
- Mit 1 muss er stets in q_1 übergehen, da $w1$ auf 1 endet.

Entwurf eines Automaten für eine vorgegebene Sprache

- Man macht sich klar, welche Information über das bereits gelesene Anfangswort nötig ist, um mit dem Restwort 'richtig' weiterzuarbeiten?
- Reicht eine endliche Anzahl n von unterschiedlichen Informationen aus, so entwirft man einen Automaten mit n Zuständen.
- Die Zustandsübergänge legt man so fest, dass nach dem Lesen eines Zeichens stets wieder die richtige Information vorliegt.
- Reicht eine endliche Anzahl unterschiedlicher Informationen *nicht* aus, so gibt es keinen endlichen Automaten für die Sprache.

Endliche Automaten

Beispiel: Automat A_{int} für die Sprache

$$L_{int} = \{\varepsilon, -\}\{0, \ldots, 9\}^+ = \{\varepsilon, -\}\{0, \ldots, 9\}\{0, \ldots, 9\}^*$$

Sei $\Sigma = \{-, 0, \ldots, 9\}$. Man muss folgende Fälle für das bereits gelesene Wort w unterscheiden:

- $w = \varepsilon$: Noch nichts gelesen, die gesamte Zahl wird noch erwartet.
- $w = -$: Nur das Vorzeichen gelesen, es wird noch eine *nichtleere* Ziffernfolge erwartet.
- $w \in L_{int} = \{\varepsilon, -\}\{0, \ldots, 9\}^+$: Schon mindestens eine Ziffer gelesen, jetzt darf noch eine *beliebige* Ziffernfolge kommen.
- w enthält '$-$' an der falschen Stelle, d.h. nicht an erster Position: Schon alles verdorben!

Also 4 Zustände für diese 4 Situationen: $q_\varepsilon, q_-, q_{int}, q'$

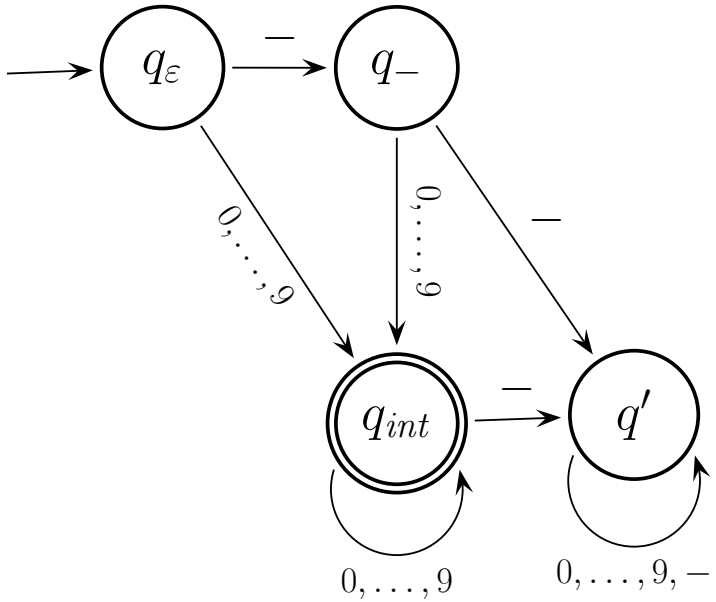
Endliche Automaten

Deshalb definiert man $A_{int} = (\Sigma, Q, s, F, \delta)$, wobei

- $Q = \{q_\varepsilon, q_-, q_{int}, q'\}$
- $s = q_\varepsilon$
- $F = \{q_{int}\}$
- $\delta : Q \times \Sigma \to Q$

$$
\begin{aligned}
(q_\varepsilon, -) &\mapsto q_- \\
(q_\varepsilon, a) &\mapsto q_{int} \quad \text{für alle } a \in \{0, \ldots, 9\} \\
(q_-, -) &\mapsto q' \\
(q_-, a) &\mapsto q_{int} \quad \text{für alle } a \in \{0, \ldots, 9\} \\
(q_{int}, -) &\mapsto q' \\
(q_{int}, a) &\mapsto q_{int} \quad \text{für alle } a \in \{0, \ldots, 9\} \\
(q', a) &\mapsto q' \quad \text{für alle } a \in \Sigma \text{ (Sackgasse!)}
\end{aligned}
$$

Graphische Darstellung



Vereinbarung: Ein Pfeil mit mehreren Zeichen $a_1, \ldots, a_n$ ist Abkürzung für n Pfeile mit den einzelnen Zeichen.

Endliche Automaten

Wie beweist man, dass $L(A_{int}) = L_{int}$? Man zeigt, dass jedes $w \in \Sigma^*$ zum gewünschten Zustand führt, d.h.:

1. Wenn $w = \varepsilon$, dann $(q_\varepsilon, w) \vdash^* (q_\varepsilon, \varepsilon)$.
2. Wenn $w = -$, dann $(q_\varepsilon, w) \vdash^* (q_-, \varepsilon)$.
3. Wenn $w \in L_{int}$, dann $(q_\varepsilon, w) \vdash^* (q_{int}, \varepsilon)$.
4. Wenn $w \notin L_{int} \cup \{\varepsilon, -\}$, dann $(q_\varepsilon, w) \vdash^* (q', \varepsilon)$.

Dies geschieht durch Induktion über $|w|$:
$|w| = 0$, d.h. $w = \varepsilon$: Es gilt $(q_\varepsilon, w) \vdash^0 (q_\varepsilon, \varepsilon)$.
$|w| > 0$, d.h. $w = va$: Fallunterscheidung für v. Wenn $v \in L_{int}$, dann gilt nach Induktionsannahme $(q_\varepsilon, v) \vdash^* (q_{int}, \varepsilon)$. Ist $a \in \{0, \ldots, 9\}$, dann folgt $(q_\varepsilon, w) = (q_\varepsilon, va) \vdash^* (q_{int}, a) \vdash (q_{int}, \varepsilon)$ und das ist der gewünschte Zustand, weil $w = va \in L_{int}$. Ist $a = -$, dann folgt $(q_\varepsilon, w) = (q_\varepsilon, va) \vdash^* (q_{int}, a) \vdash (q', \varepsilon)$ und das ist wieder der richtige Zustand, weil $w \notin L_{int} \cup \{\varepsilon, -\}$. Ähnlich für die übrigen Fälle! □

Endliche Automaten

Beispiel: Ein Automat A für die Sprache $L = \{a^n b^n \mid n \geq 0\}$?

Intuition:

- Um $w \in L$ zu testen, müsste ein endlicher Automat (insbesondere) die Anzahl der a's und b's in w vergleichen.
- Also muss er stets die Anzahl der bisher gelesenen a's kennen.
- Da diese Anzahl beliebig groß werden kann, reichen die endlich vielen Zustände zum Speichern dieser Information nicht aus.
- Also gibt es keinen Automaten A mit $L = L(A)$.

Schlagwort: "Ein endlicher Automat kann nicht zählen."
(Soll heißen: Er kann nicht beliebig hoch zählen).

Exakter Beweis:

Angenommen, es existiert ein DEA $A = (\Sigma, Q, s, F, \delta)$ mit $L(A) = L$.

Wir betrachten alle Wörter der Form a^n mit $n \geq 0$.

Für jedes $n \geq 0$ existiert ein Zustand $q \in Q$ mit $(s, a^n) \vdash_A^* (q, \varepsilon)$ (der Zustand, der durch Lesen von a^n erreicht wird). Da es unendlich viele Wörter a^n gibt, aber nur endlich viele $q \in Q$, muss es (mindestens) zwei Wörter a^i, a^j $(i \neq j)$ geben, mit denen man den *gleichen* Zustand q erreicht, d.h. $(s, a^i) \vdash_A^* (q, \varepsilon)$ und $(s, a^j) \vdash_A^* (q, \varepsilon)$.

Sei $q' \in Q$ mit $(q, b^i) \vdash_A^* (q', \varepsilon)$. Dann gilt $(s, a^i b^i) \vdash_A^* (q, b^i) \vdash_A^* (q', \varepsilon)$ und $(s, a^j b^i) \vdash_A^* (q, b^i) \vdash_A^* (q', \varepsilon)$. Aber $a^i b^i \in L$ und $a^j b^i \notin L$, d.h. sie dürfen nicht beide zum gleichen Zustand q' führen.

Damit ist die Annahme zum Widerspruch geführt, d.h. es existiert kein DEA A mit $L(A) = L$. □

Im Beweis wurde benutzt:

- Das *Schubfachprinzip* (engl. *pigeonhole principle*):
  Wenn man eine Menge von Dingen auf eine gewisse Anzahl von Schubfächern verteilt, und die Anzahl der Dinge ist größer als die Anzahl der Schubfächer, dann landen mindestens zwei Dinge im gleichen Schubfach.
  Hier: Unendlich viele Wörter a^n wurden auf endlich viele Zustände verteilt.
- Folgende Eigenschaft von $\vdash_A^*$:
  Wenn $(q, u) \vdash_A^* (q', \varepsilon)$, dann gilt auch $(q, uv) \vdash_A^* (q', v)$.

Lemma 2.9 *Sei* $A = (\Sigma, Q, s, F, \delta)$ *ein DEA, seien* $u, v \in \Sigma^*$ *und* $q, q' \in Q$. *Dann gilt:*

1. $(q, u) \vdash_A^* (q', \varepsilon) \Leftrightarrow (q, uv) \vdash_A^* (q', v)$
2. $(q, uv) \vdash_A^* (q', \varepsilon) \Leftrightarrow$ *es existiert ein* $q'' \in Q$ *mit* $(q, uv) \vdash_A^* (q'', v) \vdash_A^* (q', \varepsilon)$

Begründung:

1. *Die Übergangsschritte, die* A *bei Abarbeitung eines Wortes* u *macht, werden von einem hinter* u *stehenden Wort* v *nicht beeinflusst. Denn* A *arbeitet ja von links nach rechts, und kann deshalb das Wort* v *während der Abarbeitung von* u *gar nicht sehen.*
2. *Jede Folge von Übergangsschritten für ein Wort* uv *lässt sich aufteilen in die Schritte für* u *und die Schritte für* v. *Das liegt daran, dass der Automat zeichenweise von links nach rechts liest.*

Frage:

Für welche Sprachen L existiert ein DEA A mit $L(A) = L$?

Wir werden beweisen:

Ein DEA A mit $L = L(A)$ existiert genau dann, wenn L eine reguläre Sprache ist.

Die Funktionsschreibweise δ^*

Sei $A = (\Sigma, Q, s, F, \delta)$ ein DEA.

Dann existiert für jede Konfiguration $(q, w) \in Q \times \Sigma^*$ *genau ein* Zustand $q' \in Q$ mit $(q, w) \vdash_A^* (q', \varepsilon)$ (weil für jedes einzelne Zeichen von w stets ein eindeutiger Übergangsschritt existiert).

Diesen eindeutigen Zustand q' bezeichnen wir mit $\delta^*(q, w)$.

Die so definierte Funktion $\delta^* : Q \times \Sigma^* \to Q$ lässt sich auch ohne den Umweg über $\vdash_A^*$ definieren, nämlich durch Induktion über die Länge von $|w|$:

- $\delta^*(q, \varepsilon) = q$
- $\delta^*(q, wa) = \delta(\delta^*(q, w), a)$

oder:

- $\delta^*(q, aw) = \delta^*(\delta(q, a), w)$

Vor- und Nachteile beider Schreibweisen:

In der Funktionsschreibweise haben wir eine eigenständige Bezeichnung für den Ergebniszustand q' (in der die Existenz und Eindeutigkeit implizit enthalten ist). In der Relationsschreibweise müssen wir immer erst die Existenz (und evtl. Eindeutigkeit) von q' erwähnen (und damit einen neuen Namen für einen Zustand einführen).

In der Relationsschreibweise kann man Übergangsschritte besser aneinanderhängen und man hat die Schreibweisen $\vdash^n, \vdash^+, \vdash^*$ mit schönen Gesetzmäßigkeiten (Monoid!). Mit der Funktionsschreibweise geht das nicht so einfach (denn $\delta : Q \times \Sigma \to Q$ kann man nicht mit sich selbst verknüpfen).