
Grundlagen der theoretischen Informatik

Kurt Sieber

Fachbereich Mathematik/Theoretische Informatik
Universität Siegen

Vorlesung vom 25.10.2004 (Stand: 25.10.2004)

Endliche Automaten

Konventionen

Als typische Bezeichnungen verwenden wir (bisher):

- a, b, c für Zeichen
- u, v, w, x, y, z für Wörter
- Σ für ein Alphabet
- L für eine Sprache
- A, B für Automaten
- p, q, r, s für Zustände, speziell s für den Startzustand
- P, Q, F für Zustandsmengen, speziell Q für die Menge aller Zustände, F für die Menge der Endzustände

Endliche Automaten

Bisher: Deterministische endliche Automaten

- Von jedem Zustand $q \in Q$ geht **genau ein** Pfeil mit einem Zeichen $a \in \Sigma$ aus.
- Formal: $\delta : Q \times \Sigma \rightarrow Q$ ist eine totale Funktion.

Jetzt: Nichtdeterminismus erlaubt

- Von jedem Zustand $q \in Q$ dürfen **beliebig viele** Pfeile mit einem Zeichen $a \in \Sigma$ ausgehen.
- Formal: An die Stelle der Funktion $\delta : Q \times \Sigma \rightarrow Q$ tritt eine **Relation** $\Delta \subseteq Q \times \Sigma \times Q$.

Endliche Automaten

Definition 1.7 Ein *nichtdeterministischer endlicher Automat* (kurz: *NDEA*) ist ein 5-Tupel $A = (\Sigma, Q, s, F, \Delta)$ mit:

- Σ, Q, s, F wie beim DEA
- $\Delta \subseteq Q \times \Sigma \times Q$ ist die sogenannte *Übergangsrelation*

Noch zu klären:

- Wie arbeitet ein NDEA?
- Welche Sprache erkennt er?

Endliche Automaten

Definition 1.8

- Die Relation \vdash_A auf der Menge $Q \times \Sigma^*$ aller *Konfigurationen* von A ist definiert durch

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

- Die Relationen \vdash_A^n ($n \geq 0$), \vdash_A^+ , \vdash_A^* und die von A *akzeptierte* oder *erkannte* Sprache $L(A)$ sind wie beim DEA definiert.

Man beachte:

- Selbst \vdash_A kann man für DEAs und NDEAs einheitlich definieren, denn:
- Auch eine Funktion $\delta : Q \times \Sigma \rightarrow Q$ kann man als Relation $\delta \subseteq Q \times \Sigma \times Q$ auffassen, dann bedeutet $\delta(q, a) = q'$ nichts anderes als $(q, a, q') \in \delta$.

Endliche Automaten

Was ist also jetzt neu?

Zur Erinnerung:

- $w \in \Sigma^*$ wird von A **akzeptiert**, genau dann wenn es ein $q \in F$ gibt mit $(s, w) \vdash_A^* (q, \varepsilon)$.
- $L(A) = \{w \in \Sigma^* \mid w \text{ wird von } A \text{ akzeptiert}\}$

Unterschied zum DEA:

- Ein Zustand q mit $(s, w) \vdash_A^* (q, \varepsilon)$ muss nicht existieren (denn der passende Übergang für ein Zeichen aus w kann fehlen.)
- Es kann mehr als ein solcher Zustand q existieren (denn für ein Zeichen aus w kann es mehr als einen Übergang geben).

Endliche Automaten

Wann wird ein Wort w also akzeptiert?

Wenn **mindestens einer** der Zustände, die man mit w erreicht, ein Endzustand ist.

Intuition:

- In einem NDEA können verschiedene Wege für ein Wort w existieren.
- Wir dürfen davon ausgehen, dass der NDEA immer den **richtigen** Weg “errät”.
- D.h. wenn **ein** Weg zu einem Endzustand führt, dann brauchen uns die anderen Wege nicht zu interessieren.

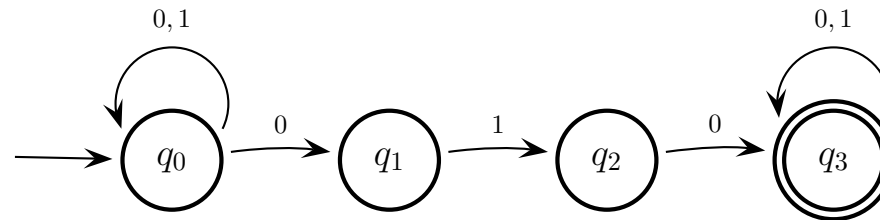
Endliche Automaten

Beispiel:

$$\Sigma = \{0, 1\}$$

$$L = \{w \in \{0, 1\}^* \mid 010 \text{ ist Teilwort von } w\}.$$

Ein NDEA A mit $L(A) = L$ ist:



Formal:

$A = (\Sigma, Q, s, F, \Delta)$ mit $\Sigma = \{0, 1\}$, $Q = \{q_0, q_1, q_2, q_3\}$, $s = q_0$ und

$$\Delta = \{ (q_0, 0, q_0), (q_0, 1, q_0), (q_0, 0, q_1), (q_1, 1, q_2), (q_2, 0, q_3), \\ (q_3, 0, q_3), (q_3, 1, q_3) \}$$

Endliche Automaten

Wieso gilt $L = L(A)$?

Intuition:

Weil der NDEA A die Stelle, an der das Teilwort 010 beginnt, **erraten** und dann mit 010 in den Endzustand q_3 übergehen kann.

Exakter Beweis:

' \subseteq ': Sei $w \in L$.

Dann existieren $u, v \in \{0, 1\}^*$ mit $w = u010v$.

Also gilt $(s, w) = (q_0, u010v) \vdash_A^* (q_0, 010v) \vdash_A^* (q_3, v) \vdash_A^* (q_3, \varepsilon)$.

Damit ist $w \in L(A)$ bewiesen, weil $q_3 \in F$.

' \supseteq ': Sei $w \in L(A)$, d.h. $(q_0, w) \vdash_A^* (q_3, \varepsilon)$, weil F nur q_3 enthält.

Dann muss w das Wort 010 als Teilwort enthalten, weil man nur mit diesem Wort von q_0 nach q_3 gelangen kann.

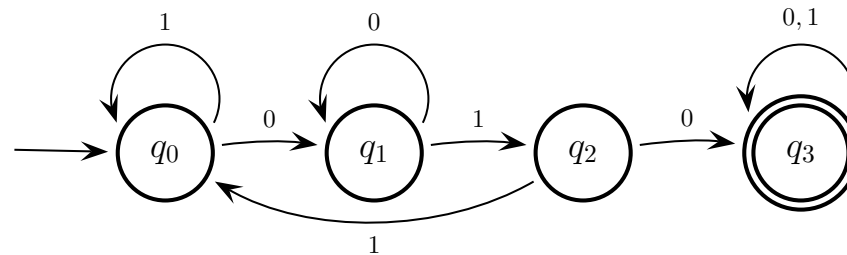
Also ist $w \in L$. □

Endliche Automaten

Wieso ist der Beweis so einfach?

Weil dieser NDEA auf einer sehr einfachen Idee beruht.

Zum Vergleich: Ein DEA A' für die gleiche Sprache



Beweisidee:

Sei w das bisher gelesene Wort.

q_0 bedeutet: $w = \varepsilon$, $w = 1$ oder w endet auf 11, aber enthält nicht 010.

q_1 bedeutet: w endet auf 0, aber enthält nicht 010.

q_2 bedeutet: w endet auf 01, aber enthält nicht 010.

q_3 bedeutet: w enthält 010, d.h. $w \in L$.

Endliche Automaten

Fazit:

- Oft findet man leichter einen NDEA als einen DEA für eine vorgegebene Sprache (weil man mehr Freiheiten hat)
- und auch der Korrektheitsbeweis ist oft einfacher (wenn der NDEA auf einer einfacheren Idee beruht).

Endliche Automaten

Wie kann man einen NDEA implementieren?

- Als nichtdeterministisches Programm mit Zufallsgenerator?

Nein, denn man kann nicht erwarten, dass das Programm den richtigen Weg errät.

- Durch einen backtracking-Algorithmus, der alle möglichen Wege nacheinander durchspielt?

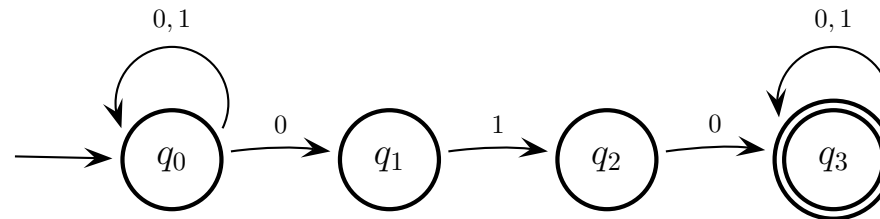
Ja, aber man muss sich überlegen, dass es nur endlich viele mögliche Wege gibt, und wie man sie systematisch ausprobiert.

- Durch einen “parallelen” Algorithmus, der alle möglichen Wege gleichzeitig durchspielt?

Das ist die einfachste Möglichkeit: Man führt einfach Buch über die Menge der Zustände, die vom Startzustand aus mit dem bereits gelesenen Wort erreichbar sind. Diese Menge kann auch leer sein!

Endliche Automaten

Beispiel: $w = 0011010$ auf dem NDEA A



Menge der möglichen Zustände nach Lesen von

- $\varepsilon : \{q_0\}$
- $0 : \{q_0, q_1\}$
- $00 : \{q_0, q_1\}$
- $001 : \{q_0, q_2\}$
- $0011 : \{q_0\}$
- $00110 : \{q_0, q_1\}$
- $001101 : \{q_0, q_2\}$
- $0011010 : \{q_0, q_1, q_3\}$, also $w \in L(A)$, da q_3 mit w erreichbar

Endliche Automaten

Fazit:

- Wir haben einen **deterministischen** Algorithmus für einen **NDEA** A ,
 - bei dem man das Eingabewort w zeichenweise von links nach rechts liest,
 - bei dem man sich aber **Zustandsmengen** anstelle von einzelnen Zuständen merken muss.
- Dieser Algorithmus lässt sich sogar **auf einem DEA** A' durchführen.
- Man muss A' nur mit einem großen Gedächtnis ausstatten, damit er sich **Teilmengen** der Zustandsmenge Q von A merken kann.
- Als Zustände von A' wählen wir deshalb alle Teilmengen von Q ,
- und legen dann die Übergänge zwischen diesen Zustandsmengen passend fest (wie im Beispiel).
- So erhalten wir einen zu A **äquivalenten** DEA, d.h. einen DEA A' mit $L(A) = L(A')$.

Endliche Automaten

Definition 1.9 Sei $A = (\Sigma, Q, s, F, \Delta)$ ein NDEA. Der *Potenzautomat* zu A ist definiert als der DEA $A' = (\Sigma, Q', s', F', \delta)$ mit

- $Q' = \wp(Q)$, die Potenzmenge von Q
- $s' = \{s\}$
- $F' = \{P \in \wp(Q) \mid P \cap F \neq \emptyset\}$
 $= \{P \subseteq Q \mid P \text{ enthält mindestens einen Zustand aus } F\}$
- $\delta : \wp(Q) \times \Sigma \rightarrow \wp(Q)$

$$\delta(P, a) = \{q \in Q \mid \text{es existiert ein } p \in P \text{ mit } (p, a, q) \in \Delta\}$$

(d.h. $\delta(P, a)$ enthält genau die Zustände, die von einem Zustand $p \in P$ durch einen mit a markierten Pfeil erreichbar sind)

Endliche Automaten

Satz 1.10 Sei A ein NDEA und A' der Potenzautomat zu A . Dann gilt $L(A) = L(A')$.

Beweis

Sei $A = (\Sigma, Q, s, F, \Delta)$ und sei $A' = (\Sigma, Q', s', F', \delta)$ der Potenzautomat zu A .

- Wir zeigen zunächst:

$$\delta^*({s}, w) = \{q \in Q \mid (s, w) \vdash_A^* (q, \varepsilon)\}$$

In Worten: $\delta^*({s}, w)$ ist die Menge aller Zustände, die der NDEA A vom Zustand s aus durch Abarbeitung von w erreichen kann. Das bedeutet, dass der Potenzautomat A' die gewünschte 'Buchführung' für den NDEA A richtig durchführt.

Endliche Automaten

Also zu zeigen: $\delta^*({s}, w) = \{q \in Q \mid (s, w) \vdash_A^* (q, \varepsilon)\}$

Beweis durch Induktion über $|w|$:

$|w| = 0$, d.h. $w = \varepsilon$:

$\delta^*({s}, \varepsilon) = \{s\}$ per Definition von δ^*

$\{q \in Q \mid (s, \varepsilon) \vdash_A^* (q, \varepsilon)\} = \{s\}$ per Definition von \vdash_A^*

$|w| > 0$, d.h. $w = va$:

$$\delta^*({s}, va) = \delta(\delta^*({s}, v), a)$$

per Definition der Schreibweise δ^*

$$= \delta(\{q \in Q \mid (s, v) \vdash_A^* (q, \varepsilon)\}, a)$$

nach Induktionsannahme für v

$$= \{q' \in Q \mid \text{es existiert ein } q \in Q \text{ mit } (s, v) \vdash_A^* (q, \varepsilon) \\ \text{und } (q, a) \vdash_A (q', \varepsilon)\}$$

per Definition von δ im Potenzautomaten A

$$= \{q' \in Q \mid (s, va) \vdash_A^* (q', \varepsilon)\}$$

Endliche Automaten

Die Behauptung ist bewiesen:

$$\delta^*({s}, w) = \{q \in Q \mid (s, w) \vdash_A^* (q, \varepsilon)\}$$

und es folgt:

$$\begin{aligned} w \in L(A) &\Leftrightarrow \text{es existiert ein } q \in F \text{ mit } (s, w) \vdash_A^* (q, \varepsilon) \\ &\quad \text{per Definition der akzeptierten Sprache } L(A) \\ &\Leftrightarrow \text{es existiert ein } q \in F \text{ mit } q \in \delta^*({s}, w) \\ &\quad \text{nach der bewiesenen Behauptung} \\ &\Leftrightarrow \delta^*({s}, w) \cap F \neq \emptyset \\ &\Leftrightarrow \delta^*({s}, w) \in F' \\ &\quad \text{per Definition von } F' \text{ im Potenzautomaten } A' \\ &\Leftrightarrow w \in L(A') \\ &\quad \text{weil } \{s\} \text{ der Startzustand von } A' \text{ ist} \end{aligned}$$

Also ist $L(A) = L(A')$, d.h. A und A' sind äquivalent. □

Endliche Automaten

Man beachte:

- Im Beweis wurde benutzt, dass Lemma 1.6 auch für NDEAs gilt.
- Die Definition des Potenzautomaten ist **konstruktiv**, d.h. man hat einen **Algorithmus**, um den Potenzautomaten A' aus dem NDEA A zu erhalten.
- Der Potenzautomat ist natürlich sehr groß, denn $|\wp(Q)| = 2^{|Q|}$. Oft sind aber viele Zustände überflüssig, so dass man sie nachträglich entfernen oder sogar von Anfang an weglassen kann.

Endliche Automaten

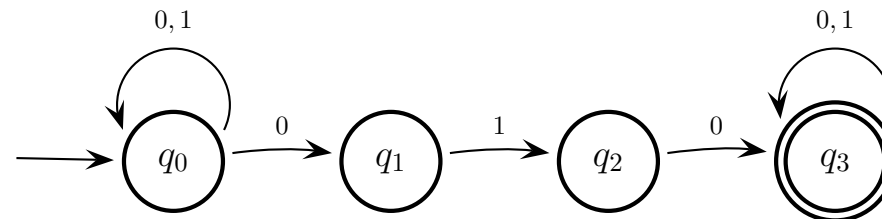
Definition 1.11 Sei $A = (\Sigma, Q, s, F, \delta)$ ein DEA. Ein Zustand $q \in Q$ heißt *erreichbar*, wenn es ein Wort w gibt mit $(s, w) \vdash_A^* (q, \varepsilon)$, ansonsten heißt er *unerreichbar*.

- Unerreichbare Zustände kann man offensichtlich aus einem DEA entfernen, ohne dass sich die akzeptierte Sprache verändert (und ohne dass man die Definition eines DEA verletzt, denn s ist stets erreichbar, und δ lässt sich auf die Menge der erreichbaren Zustände einschränken.)
- Einige “Entwurfsmethoden” für Automaten, wie z.B. die Konstruktion des Potenzautomaten, kann man so abändern, dass der entworfene Automat von vornherein nur erreichbare Zustände enthält.

Endliche Automaten

Beispiel

Sei A wieder der NDEA



Der Potenzautomat von A hat 16 Zustände. Wir bestimmen die erreichbaren unter ihnen.

Der Startzustand $\{q_0\}$ ist erreichbar.

$\delta(\{q_0\}, 0) = \{q_0, q_1\}$, also ist $\{q_0, q_1\}$ erreichbar.

$\delta(\{q_0\}, 1) = \{q_0\}$

$\delta(\{q_0, q_1\}, 0) = \{q_0, q_1\}$

$\delta(\{q_0, q_1\}, 1) = \{q_0, q_2\}$, also ist $\{q_0, q_2\}$ erreichbar.

Endliche Automaten

$\delta(\{q_0, q_2\}, 0) = \{q_0, q_1, q_3\}$, also ist $\{q_0, q_1, q_3\}$ erreichbar.

$\delta(\{q_0, q_2\}, 1) = \{q_0\}$

$\delta(\{q_0, q_1, q_3\}, 0) = \{q_0, q_1, q_3\}$

$\delta(\{q_0, q_1, q_3\}, 1) = \{q_0, q_2, q_3\}$, also ist $\{q_0, q_2, q_3\}$ erreichbar.

$\delta(\{q_0, q_2, q_3\}, 0) = \{q_0, q_1, q_3\}$

$\delta(\{q_0, q_2, q_3\}, 1) = \{q_0, q_3\}$, also ist $\{q_0, q_3\}$ erreichbar.

$\delta(\{q_0, q_3\}, 0) = \{q_0, q_1, q_3\}$

$\delta(\{q_0, q_3\}, 1) = \{q_0, q_3\}$

Wenn wir uns also auf die erreichbaren Zustände beschränken, so erhalten wir einen DEA, der nur 6 statt 16 Zustände hat. Drei davon sind Endzustände, nämlich diejenigen, die den Endzustand q_3 von A enthalten.

Endliche Automaten

Graphisch: Der erreichbare Teil des Potenzautomaten

