Formale Methoden 1

Gerhard Jäger
Gerhard.Jaeger@uni-bielefeld.de

Uni Bielefeld, WS 2007/2008

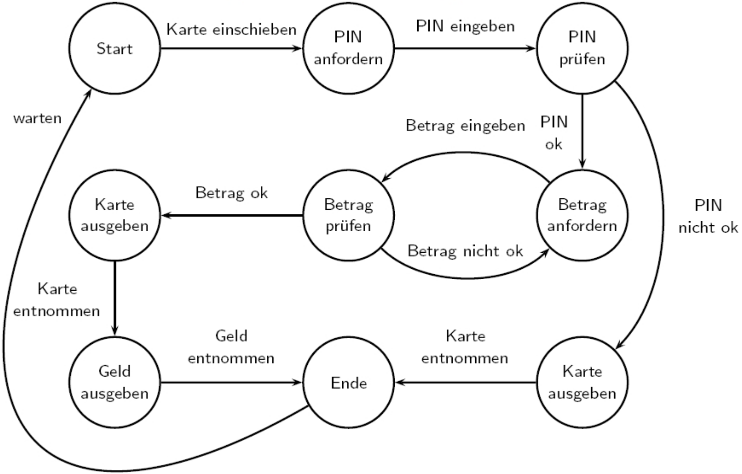
9. Januar 2008

Automaten (informell)

- gedachte Maschine/abstraktes Modell einer Maschine
- verhält sich gemäß bestimmter Regeln
- Verhalten des Automaten hängt von Informationen ab, die der Automat aus der Umgebung bekommt
- Automaten „treffen Entscheidungen“

Ein Beispiel

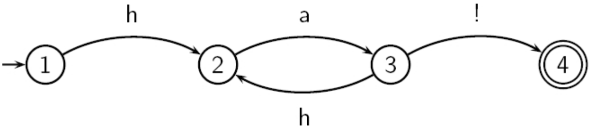
Der „endliche Geldautomat“ (nach Reinhard Völler)



Sprach-Automaten

- Automat bekommt *Input* von seiner Umgebung (z.B. Tastendruck durch den Benutzer)
- Input kann immer als Kette von Symbolen aus einem Alphabet repräsentiert werden (im einfachsten Fall sind das „0“ und „1“)
- Autmat produziert *Output*
- kann ebenfalls als Kette von Symbolen repräsentiert werden

Der Lachautomat (nach Stefan Müller)



Endliche Automaten

- Ein **endlicher Automat**
  - hat endlich viele Zustände,
  - erhält als Input Ketten über ein Alphabet Σ
  - gibt als Output entweder „ja“ oder „nein“ zurück
- Ein endlicher Automat definiert damit eine formale Sprache — die Menge der Inputs, für die er das Symbol „ja“ als Output zurückgibt.

Definition (Endlicher Automat, formale Definition)

Ein *(deterministischer) endlicher Automat* (DFA) M wird spezifiziert durch ein 5-Tupel

$$M = \langle Z, \Sigma, \delta, z_0, E \rangle$$

Hierbei bezeichnet Z die Menge der $Zustände$ und Σ ist das *Eingabealphabet*, $Z \cap \Sigma = \emptyset$. Z und Σ sind endliche Mengen. $z_0 \in Z$ ist der *Startzustand*, $E \subseteq Z$ ist die Menge der *Endzustände* udn $\delta : Z \times \Sigma \mapsto Z$ heißt die *Überführungsfunktion*.

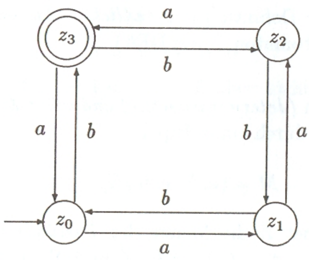
Endliche Automaten: Beispiel

Sei $M = \langle Z, \Sigma, \delta, z_0, E \rangle$, wobei

$$
\begin{aligned}
Z &= \{z_0, z_1, z_2, z_3\} \\
\Sigma &= \{a, b\} \\
E &= \{z_3\} \\
\delta(z_0, a) &= z_1 \\
\delta(z_0, b) &= z_3 \\
\delta(z_1, a) &= z_2 \\
\delta(z_1, b) &= z_0 \\
\delta(z_2, a) &= z_3 \\
\delta(z_2, b) &= z_1 \\
\delta(z_3, a) &= z_0 \\
\delta(z_4, b) &= z_2
\end{aligned}
$$

Endliche Automaten: Beispiel

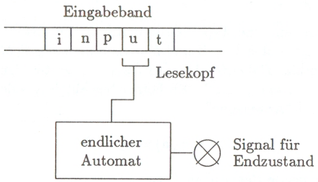
Endliche Automaten können als Graphen repräsentiert werden:



- Startzustand wird durch einen Pfeil gekennzeichnet
- Endzustände werden durch doppelte Umrandung gekennzeichnet
- Überführungsfunktion wird durch etikettierte gerichtete Kanten dargestellt

Endliche Automaten

- Intuition:
  - Automat beginnt im Startzustand
  - Input ist auf einem Eingabeband (z.B. Lochkartenstreifen) geschrieben
  - Pro Zeiteinheit liest der Automat ein Symbol α auf dem Eingabeband und bewegt sich entlang eines mit α etikettierten Pfeils in einen neuen Zustand
  - Wenn der Automat am Ende des Eingabebandes in einem Endzustand ist, wurde die Kette *akzeptiert* (Output: „ja“)
  - Andernfalls wurde die Kette nicht akzeptiert (Output „nein“)



Frage: Welche Sprache akzeptiert der Automat aus dem Beispiel?

Endliche Automaten und formale Sprachen

Definition

Zu einem gegebenen DFA $M = \langle Z, \Sigma, \delta, z_0, E\rangle$ definieren wir eine Funktion $\hat{\delta} : Z \times \Sigma^* \mapsto Z$ durch eine rekursive Definition wie folgt:

$$\begin{aligned} \hat{\delta}(z, \epsilon) &= z \\ \hat{\delta}(z, ax) &= \hat{\delta}(\delta(z, a), x) \end{aligned}$$

Hierbei ist $z \in Z, x \in \Sigma^*$ und $a \in \Sigma$.
Die von M *akzeptierte Sprache* ist

$$L(M) = \{x \in \Sigma^* | \hat{\delta}(z_0, x) \in E\}$$

Endliche Automaten und formale Sprachen

- Definition von $\hat{\delta}$ erweitert Definition von δ von Einzelnen Symbolen zu Symbolketten
- für einzelne Zeichen gilt: $\hat{\delta}(z, a) = \delta(z, a)$
- auch gilt

$$\hat{\delta}(z, a_1 a_2 \dots a_n) = \delta(\dots \delta(\delta(z, a_1), a_2) \dots, a_n)$$

Theorem

Jede durch einen deterministischen endlichen Automaten erkannte Sprache ist regulär (also Typ 3 in der Chomsky-Hierarchie).

Beweisidee

Sei

$$M = \langle Z, \Sigma, \delta, z_0, E \rangle$$

ein DFA. Wir konstruieren eine reguläre Grammatik

$$G = \langle V_T, V_N, S, R \rangle$$

wie folgt:

- $V_T = \Sigma$
- $V_N = Z$
- $S = z_0$

Beweisidee

- Für jeden Übergang

$$\delta(z_1, a) = z_2$$

  gibt es eine Regel

$$z_1 \longrightarrow az_2$$

- Wenn $z_2 \in E$, gibt es außerdem die Regel

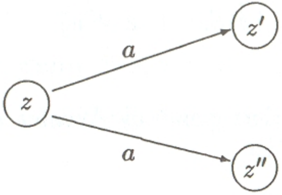
$$z_1 \longrightarrow a$$

- Wenn $z_0 \in E$, gibt es die Regel

$$z_0 \longrightarrow \epsilon$$

Nicht-deterministische Automaten

- Bei einem *deterministischen* Automaten ist in jedem Zustand für jedes Eingabesymbol eindeutig festgelegt, in welchen Zustand der Automat wechselt
- Ein *nicht-deterministischer* Automat darf u.U. zufällig entscheiden, in welchen Zustand er wechselt.
- δ ist bei nicht-deterministischen Automaten nicht unbedingt eine Funktion, sondern eine Relation.



Nicht-deterministische Automaten

Definition

Ein *nicht-deterministischer endlicher Automat* (NFA) M wird spezifiziert durch ein 5-Tupel

$$M = \langle Z, \Sigma, \delta, z_0, E \rangle$$

Hierbei sind

- Z eine endliche Menge, die *Zustände*,
- Σ eine endliche Menge, das *Eingabealphabet*, $Z \cap \Sigma = \emptyset$,
- δ eine Relation $\subseteq Z \times \Sigma \times Z$, *Überführungsrelation*,
- z_0 ist der Anfangszustand, und
- $E \subseteq Z$ ist die Menge der Endzustände.

Nicht-deterministische Automaten

Auch die nicht-deterministische Überführungsrelation kann zu einer Relation $\hat{\delta}$ für Ketten erweitert werden:

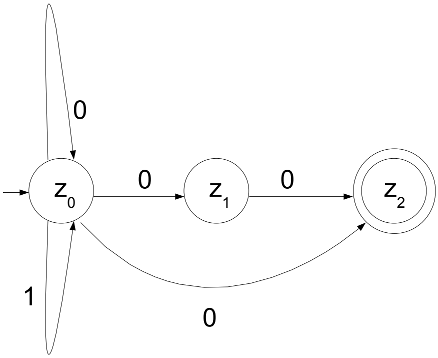
$$\begin{array}{rll} \hat{\delta}(z,\epsilon,z) & & \text{für alle } z \in Z \\ \hat{\delta}(z_1,ax,z_2) & \text{gdw.} & \delta(z_1,a,z_3),\hat{\delta}(z_3,x,z_2) \text{ für ein } z_3 \in Z \end{array}$$

Die von einem NFA M akzeptierte Sprache $L(M)$ ist definiert als

$$L(M) = \{x \in \Sigma^* | \text{es gibt ein } z \in E \text{ so dass } \hat{\delta}(z_0,x,z)\}$$

Nicht-deterministische Automaten

- Beispiel:
  - Der folgende NFA akzeptiert genau die Wörter x über $\{0, 1\}$, die auf 0 enden.



Nicht-deterministische Automaten

Theorem

Jede von einem NFA akzeptierte Sprache ist auch durch einen DFA akzeptierbar.

Beweisidee

Sei

$$M_1 = \langle Z, \Sigma, \delta, z_0, E \rangle$$

ein nicht-deterministischer endlicher Automat. Wir konstruieren daraus einen deterministischen endlichen Automaten

$$M' = \langle Z', \Sigma', \delta', z_0', E' \rangle$$

wie folgt:

- $Z' = \wp(Z)$
- $\Sigma' = \Sigma$
- $\delta'(z_1', a) = \{z \in Z | \text{es gibt ein } z_1 \in z_1' \text{ so dass } \delta(z_1, a, z)\}$
- $z_0' = \{z_0\}$
- $E' = \{z' \in \wp(Z) | z' \cap E \neq \emptyset\}$

Z' akzeptiert die selbe Sprache wie Z.

Endliche Automaten und reguläre Grammatiken

Theorem

Für jede reguläre Grammatik

$$G = \langle V_T, V_N, S, R \rangle$$

gibt es einen NFA

$$M = \langle Z, \Sigma, \delta, z_0, E \rangle$$

mit

$$L(G) = L(M)$$

Beweisidee

Wir nehmen an, dass jede Regel in R die Form $A \to aB$, $A \to a$ oder $S \to \epsilon$ hat. Jede reguläre Grammatik kann in diese Form überführt werden. Wir konstruieren M wie folgt:

- $Z = V_N \cup \{z_\omega\}$
- $\Sigma = V_T$
- $\delta(z_1, a, z_2)$ wenn $z_1 \to a z_2 \in R$
- $\delta(z_1, a, z_\omega)$ wenn $z_1 \to a \in R$
- $z_0 = S$
- Falls $S \to \epsilon \in R$, $E = \{z_0, z_\omega\}$; ansonsten $E = \{z_\omega\}$

M akzeptiert genau die Sprache, die von G generiert wird.

Endliche Automaten und reguläre Sprachen

Theorem

Sowohl die deterministischen als auch die nicht-deterministischen endlichen Automaten akzeptieren genau die regulären Sprachen.