

Informatik I: Einführung in die Programmierung

7. Automaten: Akzeptoren & Transduktoren

Albert-Ludwigs-Universität Freiburg



Bernhard Nebel

30. Oktober 2015

1 Endliche deterministische Automaten



- Motivierendes Beispiel
- Formale Grundlagen
- Verhalten eines DEAs
- Teilstring-Erkennung

- Endliche deterministische Automaten
 - Motivierendes Beispiel
 - Formale Grundlagen
 - Verhalten eines DEAs
 - Teilstring-Erkennung
- Transduktoren
- Welt & Modell
- Zusammenfassung & Ausblick

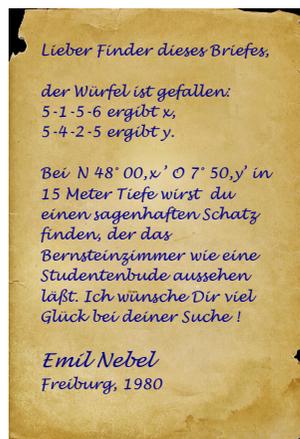
30. Oktober 2015

B. Nebel – Info I

3 / 30

Vorweg ...

Vor kurzem war ich auf unserem Dachboden und fand einen Würfel und einen Brief.



- Endliche deterministische Automaten
 - Motivierendes Beispiel
 - Formale Grundlagen
 - Verhalten eines DEAs
 - Teilstring-Erkennung
- Transduktoren
- Welt & Modell
- Zusammenfassung & Ausblick

30. Oktober 2015

B. Nebel – Info I

4 / 30

Was steckt in dem Würfel?

- In dem Würfel gibt es ein Mechanismus, der die **Abfolge** von nach oben gerichteten Würfelseiten **erkennt**.
- Nachdem die richtige Folge „gewürfelt“ wurde, schlägt dann von innen ein kleines Männchen (oder ein Modellbauservo) mit einem Hämmerchen die Koordinaten.
- Uns interessiert hier, wie man solche Folgen von Ereignissen erkennen kann.
- Dazu kann man **endliche Automaten** als **Akzeptoren** einsetzen.
- Der endliche Automat ist ein Konzept, das überall in der Informatik vorkommt.
- Endliche Automaten sind ein sehr eingeschränktes **Berechnungsmodell**, das aber oft adäquat ist und einfach einzusetzen ist.



- Endliche deterministische Automaten
 - Motivierendes Beispiel
 - Formale Grundlagen
 - Verhalten eines DEAs
 - Teilstring-Erkennung
- Transduktoren
- Welt & Modell
- Zusammenfassung & Ausblick

30. Oktober 2015

B. Nebel – Info I

5 / 30

- Ein **Alphabet** ist eine endliche, nicht-leere Menge (von Symbolen oder Zeichen), meist mit Σ bezeichnet.
- In unserem Fall besteht das Eingabealphabet aus den Würfelseiten, d.h. $\Sigma = \{1, 2, 3, 4, 5, 6\}$.
- Ein **Wort** über einem Alphabet Σ ist eine Folge von Zeichen aus Σ , z.B. wäre 5156 ein Wort.
- Eine (formale) **Sprache** ist eine beliebige (endliche oder unendliche) Menge von Wörtern.
- Endliche Automaten kann man nutzen, um **Sprachen zu akzeptieren**.

Endliche deterministische Automaten
 Motivierendes Beispiel
 Formale Grundlagen
 Verhalten eines DEAs
 Teilstring-Erkennung
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

Ein **deterministischer endlicher Automat** (DEA) ist ein Quintupel $A = \langle Q, \Sigma, \delta, q_0, F \rangle$, wobei

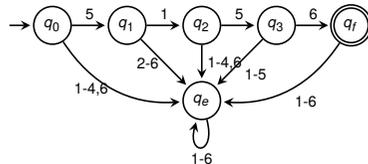
- Q ist die endliche **Zustandsmenge**,
- Σ ist das **Eingabealphabet**,
- $\delta : Q \times \Sigma \rightarrow Q$ ist die **Übergangsfunktion**,
- q_0 ist der **Anfangszustand**,
- $F \subseteq Q$ ist die Menge der (akzeptierenden) **Endzustände**.

Endliche deterministische Automaten
 Motivierendes Beispiel
 Formale Grundlagen
 Verhalten eines DEAs
 Teilstring-Erkennung
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

Die Übergangsfunktion wird entweder durch eine **Übergangstabelle** oder durch ein **Übergangsdiagramm** angegeben.

In unserem Fall (zu erkennendes Wort: 5156) könnte das wie folgt aussehen (q_e bezeichnet einen Fehlerzustand und $F = \{q_f\}$).

	1	2	3	4	5	6
q_0	q_e	q_e	q_e	q_e	q_1	q_e
q_1	q_2	q_e	q_e	q_e	q_3	q_e
q_2	q_e	q_e	q_e	q_e	q_e	q_f
q_3	q_e	q_e	q_e	q_e	q_e	q_f
q_f	q_e	q_e	q_e	q_e	q_e	q_e
q_e						



Endliche deterministische Automaten
 Motivierendes Beispiel
 Formale Grundlagen
 Verhalten eines DEAs
 Teilstring-Erkennung
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

Beachte: In Übergangsdiagrammen wird der **absorbierende Fehlerzustand** q_e und alle Übergänge dorthin in der Regel nicht angegeben.

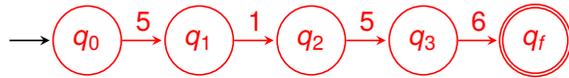
- Anfänglich befindet sich der Automat im **Startzustand** q_0 .
- Der Automat erhält ein Wort $w = a_1 a_2 \dots a_n$ über Σ als **Eingabe** (darf auch leer sein, d.h. $n = 0$).
- Der Automat liest (beginnend bei a_1) jeweils ein **Eingabezeichen** a_i und basierend auf dem **aktuellen Zustand** q wechselt er in den **Nachfolgezustand** $q' = \delta(q, a_i)$.
- Das macht der Automat, so lange Eingabezeichen gelesen werden können.
- Ist am Ende der Automat in einem der **Endzustände** F , dann wird das Eingabewort w als **akzeptiert** angesehen.
- Ansonsten ist das Wort nicht akzeptiert.
- Die Menge aller von A akzeptierten Worte ist die von A akzeptierte (oder erkannte) Sprache oder einfach die **Sprache von A** , symbolisch $\mathcal{L}(A)$.

Endliche deterministische Automaten
 Motivierendes Beispiel
 Formale Grundlagen
 Verhalten eines DEAs
 Teilstring-Erkennung
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

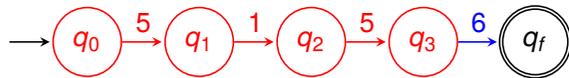
Beispiele



Eingabe: 5156 5156 156 156 56 56 6 6
 Eingabe akzeptiert



Eingabe: 515156 515156 15156 15156 5156 5156 156 156
 156
 Kein Übergang von q3 aus möglich! Eingabe nicht akzeptiert.

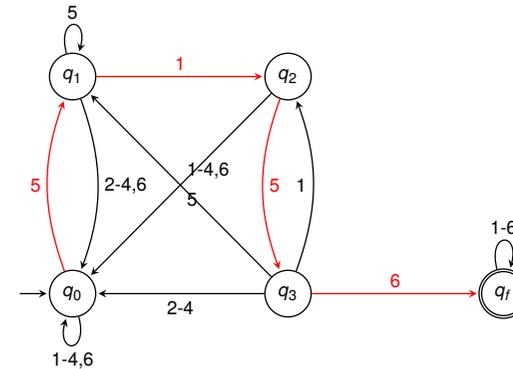


Endliche deterministische Automaten
 Motivierendes Beispiel
 Formale Grundlagen
 Verhalten eines DEAs
 Teilstring-Erkennung
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

Teilstring-Erkennung



Das letzte Beispiel zeigte: Bei unserem Würfel wollen wir eigentlich alle Folgen akzeptieren, die 5156 als Teilstring enthalten, z.B. auch 55156, oder 5155156, oder 515156 oder ... 5156...

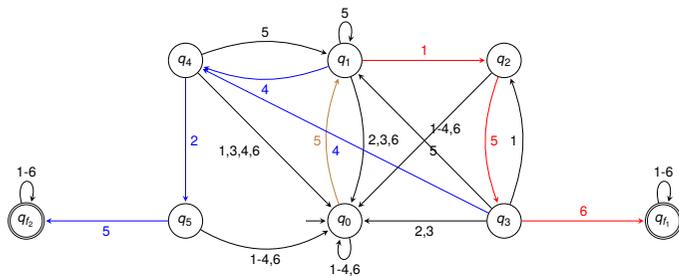


Endliche deterministische Automaten
 Motivierendes Beispiel
 Formale Grundlagen
 Verhalten eines DEAs
 Teilstring-Erkennung
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

Nord- und Ostkode integrieren



Wir haben ja auch noch 5425 als Teilstring zu erkennen! Das können wir in den Automaten integrieren:



Endliche deterministische Automaten
 Motivierendes Beispiel
 Formale Grundlagen
 Verhalten eines DEAs
 Teilstring-Erkennung
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

2 Transduktoren



- Moore-Automat
- Umsetzung
- Python-Skript für Beispiel

Endliche deterministische Automaten
 Transduktoren
 Moore-Automat
 Umsetzung
 Python-Skript für Beispiel
 Welt & Modell
 Zusammenfassung & Ausblick

Nach der Akzeptanz ist vor der Akzeptanz!



- Wir haben jetzt einen Automaten, der alle Wörter akzeptiert, die 5156 oder 5425 als Teilstring enthalten.
- Eigentlich wollen wir ja aber eine Maschine haben, die „ewig“ läuft und die jeweils nach einem akzeptierten Teilwort eine **Ausgabe** macht.
- Wir wollen keinen **Akzeptor**, sondern einen **Transduktor** – einen Automaten, der auch Ausgaben macht und nie stoppt.
- Hier verzichtet man zumeist auf Endzustände.
- Mit solchen Transduktoren kann man gut das Verhalten **eingebetteter Systeme** beschreiben.

Endliche deterministische Automaten
 Transduktoren
 Moore-Automat
 Umsetzung
 Python-Skript für Beispiel
 Welt & Modell
 Zusammenfassung & Ausblick

Moore-Automaten



Ein **Moore-Automat** (nach Edward F. Moore) ist ein endlicher Automat, der in jedem Zustand ein Zeichen ausgeben kann. Es ist ein 6-Tupel $A = \langle Q, \Sigma, \Lambda, \delta, \lambda, q_0 \rangle$, wobei

- Q ist die endliche Zustandsmenge,
- Σ ist das Eingabealphabet,
- Λ ist das **Ausgabealphabet**,
- $\delta : Q \times \Sigma \rightarrow Q$ ist die Übergangsfunktion,
- $\lambda : Q \rightarrow \Lambda$ ist die **Ausgabefunktion**.
- q_0 ist der Startzustand.

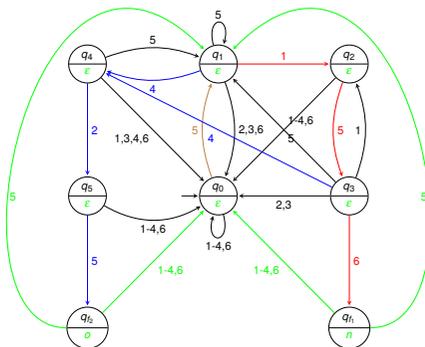
Kommt der Automat in einen Zustand q , dann gibt er das Zeichen $\lambda(q)$ aus. Oft werden diese Ausgabezeichen als Aktionen verstanden (oder sind Eingaben für andere Automaten).

Endliche deterministische Automaten
 Transduktoren
 Moore-Automat
 Umsetzung
 Python-Skript für Beispiel
 Welt & Modell
 Zusammenfassung & Ausblick

Beispiel: Der Würfel-Moore-Automat



Sei $\Lambda = \{n, o, \varepsilon\}$, dann könnte unser Würfelautomat so ausschauen (die grünen Teile sind neu):

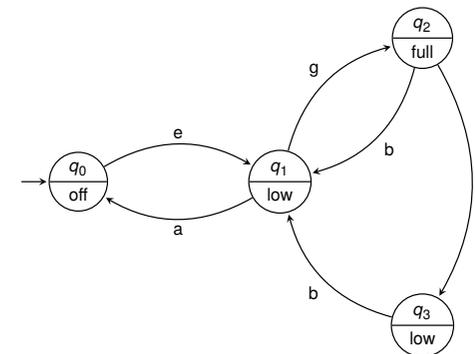


Endliche deterministische Automaten
 Transduktoren
 Moore-Automat
 Umsetzung
 Python-Skript für Beispiel
 Welt & Modell
 Zusammenfassung & Ausblick

Beispiel: Ein hypothetische Motorsteuerung



$\Sigma = \{e, a, g, b, n\}$, wobei e für „ein“, a für „aus“, g für „Gas geben“, b für „bremsen“, n für „nicht drehende Räder“ steht. $\Lambda = \{\text{off, low, full}\}$.



Endliche deterministische Automaten
 Transduktoren
 Moore-Automat
 Umsetzung
 Python-Skript für Beispiel
 Welt & Modell
 Zusammenfassung & Ausblick

Die Innereien des Würfels

- Wie **implementiert** man denn solch einen abstrakten Automaten?
- Schauen wir doch einmal in den Würfel hinein:



- Batterien (4×AA-Akkus, also 4,5-6 Volt),
- Servomotor,
- pyboard (mit einem ARM-5 Prozessor, Beschleunigungsmesser, usw.), auf dem Micropython läuft

Das Würfel-Programm

- `side_up()`: Bestimmt mit Hilfe des Beschleunigungsmessers, welche Seite oben liegt. Bei unklaren Werten wartet die Funktion, bis eine stabile Lage eingetreten ist.
- `new_input()`: Erzeugt ein neues Eingabesymbol für den Automaten (Zahl zwischen 1 und 6), wenn der Würfel 500 Millisekunden stabil lag.
- `next_state(state, input)`: Das ist die Übergangsfunktion, die den nächsten Zustand berechnet.
- `output_symbol(state)`: Berechnet das zum Zustand gehörige Ausgabesymbol.
- `automaton()`: Enthält die Endlosschleife zur Ausführung des Automaten.
- `code_knock(code)`: Klopft entsprechend dem angeforderten Code.

Der Seitenerkennung mittels Beschleunigungssensor

Die Erdbeschleunigung von 1g entspricht einem Messwert von rund 20.

Seitenerkennung

```
thres = 12
def side_up():
    while True:
        x = acc.x(); y = acc.y(); z = acc.z()
        if x > thres: return 5 #x up
        if x < -thres: return 2 #x down
        if y > thres: return 6 #y up
        if y < -thres: return 1 #y down
        if z > thres: return 3 #z up
        if z < -thres: return 4 #z down
        # no stable situation yet
```

Symbolerzeugung

Symbolerzeugung

```
def new_input():
    while True:
        curr = side_up()
        new = curr
        start = pyb.millis()
        while (curr == new and
              pyb.elapsed_millis(start) <= 500):
            new = side_up()
        if curr == new:
            return curr
```

Erzeugt i.W. alle 0,5 Sekunden ein neues Eingabesymbol, also nicht nur, wenn die Seite gewechselt wird. D.h. Automat muss auch etwas anders aussehen!

Die Übergangsfunktion

Übergangsfunktion

```
def next_state(state, input):
    if state == 0: # initial state
        if input == 5: return 1
        return 0
    elif state == 1: # '5' read
        if input == 5: return 1
        if input == 1: return 2
        if input == 4: return 4
        return 0
    elif state == 2: # '51' read
        if input == 1: return 2 # repetition!
        if input == 5: return 3
        return 0
    elif ...
```

Endliche deterministische Automaten
 Transduktoren
 Moore-Automat
 Umsetzung
 Python-Skript für Beispiel
 Welt & Modell
 Zusammenfassung & Ausblick

Beachte: Jeder Zustand hat eine Schleife für das Zeichen, das dafür notwendig war, in den Zustand zu kommen.

Der Automat & die Ausgabefunktion

Der Automat & die Ausgabefunktion

```
def automaton():
    state = 0
    while True:
        if sw(): return # if switch is pressed, exit
        state = next_state(state, new_input())
        code_knock(output_symbol(state))

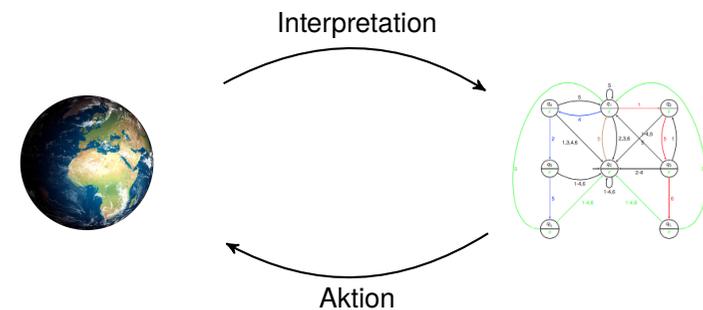
def output_symbol(state):
    if state == 10:
        return "north"
    elif state == 11:
        return "east"
    else:
        return None
```

Endliche deterministische Automaten
 Transduktoren
 Moore-Automat
 Umsetzung
 Python-Skript für Beispiel
 Welt & Modell
 Zusammenfassung & Ausblick

3 Welt & Modell

Endliche deterministische Automaten
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

Die reale Welt & formale Modelle



Endliche deterministische Automaten
 Transduktoren
 Welt & Modell
 Zusammenfassung & Ausblick

Bevor wir **formale Modelle** (wie Moore-Automaten) einsetzen können, müssen zuerst die Messwerte/Eingaben **interpretiert** und in **Symbole** umgesetzt werden. Die **Interpretation** und das **Modell** beeinflussen sich dabei gegenseitig (Beispiel: Würfelseitenerkennung und Automat)
 Werden wir in der Info I aber **nicht vertiefen**.

- Endliche Automaten sind ein einfaches **Berechnungsmodell**.
- **Formale Sprachen** sind eine Menge von Wörtern.
- **Deterministische endliche Automaten (DEAs)** sind **Akzeptoren**, sie können Sprachen akzeptieren.
- **Transduktoren** sind endliche Automaten (ohne Endzustand), mit denen Eingaben in Ausgaben überführt werden können.
- Der **Moore-Automat** macht in jedem Zustand eine Ausgabe.
- Endliche Automaten können das **Verhalten eingebetteter Systeme** gut beschreiben.
- Was wir völlig ignoriert haben: **Energieeffizienz** (das pyboard braucht 80mA im Wachmodus).