

Theoretische Informatik

Skript zur Vorlesung Lehrerweiterbildung
WS 2006/07

Literatur:

- Schöning, Uwe: Theoretische Informatik kurzgefasst; BI Wissenschaftsverlag
- Hopcroft, J.; Ullmann, J.: Einführung in die Automaten- und Komplexitätstheorie; Addison-Wesley 1990
- Wegener, I.: Theoretische Informatik; Teubner 1993
- Wagner, K.: Theoretische Informatik, Springer

Gegenstand

Gegenstand der theoretischen Informatik ist es, über konkrete Einzelfragen hinaus, grundsätzliche Fragestellungen zu bearbeiten, wie z.B.

- Was ist das Gemeinsame, das der großen Vielfalt von Computern zu Grunde liegt? Von den konkreten Details muss dabei abstrahiert werden - es sind Modelle zu entwerfen, die dann untersucht werden können (in ihrer Leistungskraft und in ihren Beziehungen zueinander).
- Wie müssen (oder können) Modelle aussehen, die ohne Rücksicht auf technische Grenzen alles ausführen können, was überhaupt von einem Rechner ausführbar ist?
- Was ist überhaupt mit einem Rechner ausführbar? Alles? D.h., sind alle Probleme, die sich mit Mitteln der Informatik oder Mathematik formulieren lassen, lösbar?
- Wie kann man Algorithmen hinsichtlich ihrer Qualität (z.B. Laufzeit, Speicherplatz) vergleichen, ohne sich dabei auf eine bestimmte Maschine zu beziehen?
- Gibt es möglicherweise Probleme, die zwar im Prinzip lösbar sind, aber die - ganz gleich für welche Maschine - so viele Ressourcen erfordern, dass eine praktische Lösung unmöglich ist?
- Kann man mathematische Beweise von einem Rechner führen lassen?
- Die Theoretische Informatik vermittelt hauptsächlich Einsichten. Trotzdem sind viele Gebiete auch eng verbunden mit praktischen Anwendungen (etwa die Theorie der formalen Sprachen mit dem Compilerbau (lexikalische Analyse)).
- Die Theoretische Informatik benutzt eine mathematisch exakte Darstellung und mathematische Beweismethoden.

I. Automaten und formale Sprachen

I.1. Endliche Automaten mit Ausgabe

I.1.1. Grundlagen

- Automat - System, das in der Lage ist, sein Verhalten ohne unmittelbares Eingreifen des Menschen selbst zu steuern
- Verhalten des Automaten - Menge der aufeinanderfolgenden Zustände des Systems, deren Anzahl bestimmt ist durch die innere Struktur des Systems.

Die Kommunikation zwischen der Umwelt und dem Automaten erfolgt über die Ein- und Ausgabe von Symbolen.

Die Beschreibung eines Automaten erfordert die Beantwortung von:

- Welche Eingabe kann er aufnehmen?
- Welche Ausgabe kann er ausgeben?
- Welche inneren Zustände kann er annehmen?
- Welchen Zustand nimmt er an, wenn in einem bestimmten Zustand eine bestimmte Eingabe erfolgt ?
- Welche Ausgabe macht er, wenn in einem bestimmten Zustand eine bestimmte Eingabe erfolgt ?

Beispiel: Mausefalle M

Zustandsmenge $Z = \{ z_1, z_2 \}$ mit z_1 : Falle gespannt
 z_2 : Falle nicht gespannt

Menge der Eingabezeichen $X = \{ m, \bar{m} \}$ m : Maus kommt
 $\bar{m}$: Maus kommt nicht

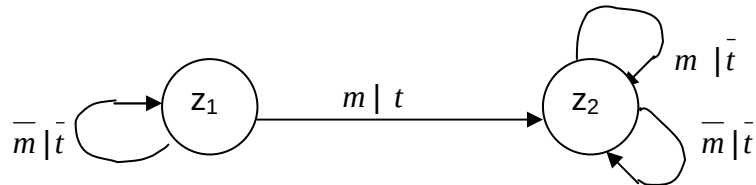
Menge der Ausgabezeichen $Y = \{ t, \bar{t} \}$ t : Maus tot
 $\bar{t}$: Maus nicht tot

Verhalten des Automaten (Tabelle):

Z \ X		X	
		m	$\bar{m}$
Z	z_1	z_2, t	$z_1, \bar{t}$
	z_2	$z_2, \bar{t}$	z_2, t

1. Zeichen : Folgezustand
2. Zeichen : Ausgabezeichen

Zustandsgraph :



Erweiterung des Modells durch Berücksichtigung von Speck

⇒ Erweiterung der Zustandsmenge (Automat M')

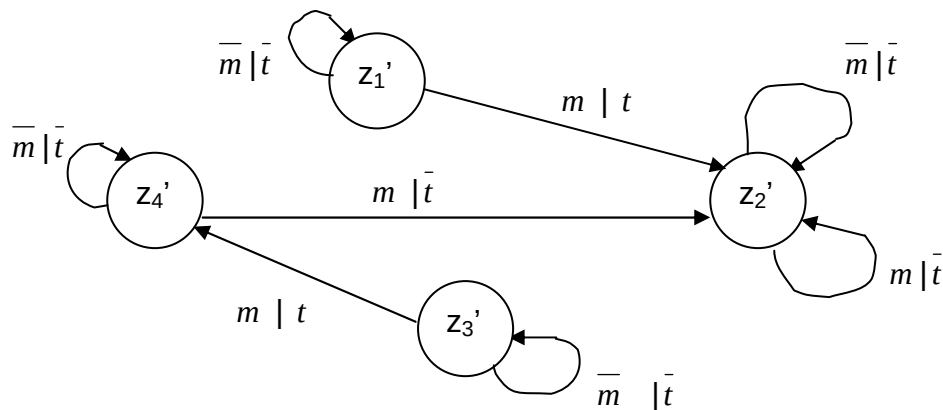
$Z = \{ z_1', z_2', z_3', z_4' \}$ mit

- z_1' : Speck nicht vorhanden, Falle gespannt
- z_2' : Speck nicht vorhanden, Falle nicht gespannt
- z_3' : Speck vorhanden, Falle gespannt
- z_4' : Speck vorhanden, Falle nicht gespannt

Tabelle:

Z \ X		X	
		m	$\bar{m}$
Z	z_1'	z_2', t	$z_1', \bar{t}$
	z_2'	$z_2', \bar{t}$	z_2', t
	z_3'	z_4', t	$z_3', \bar{t}$
	z_4'	$z_2', \bar{t}$	z_4', t

Graph:

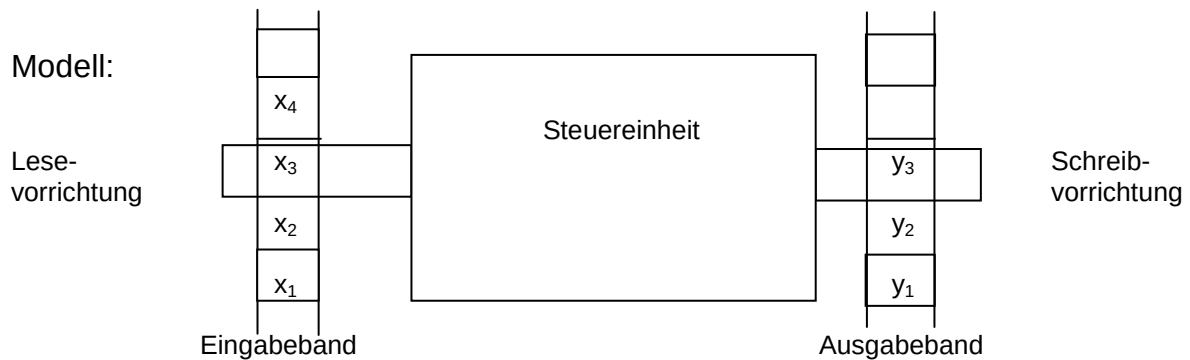


Definition: Ein **Mealy - Automat** ist ein Fünftupel $A = (X, Y, Z, f, g)$ mit nichtleeren Mengen X, Y, Z und Abbildungen

$f: Z \times X \rightarrow Z$

$g: Z \times X \rightarrow Y$

Bezeichnungen: X Eingabealphabet (Menge von Eingabezeichen)
 Y Ausgabealphabet (Menge von Ausgabezeichen)
 Z Zustandsmenge (Menge von Zustandszeichen)
 f Überföhrungsfunktion, Zustandsfunktion
 g Ergebnisfunktion, Ausgabefunktion



Arbeitsweise:

Anfangszustand z setzen	
Lesekopf auf 1. Zeichen der Eingabe setzen	
	<u>while</u> Feld unter Lesekopf nicht leer <u>do</u>
	Zeichen x unter Lesekopf lesen auf Ausgabeband Zeichen $y = g (z, x)$ schreiben Ausgabeband um 1 Feld weiterrücken Zustand $z := f (z, x)$ Eingabezustand um 1 Feld weiterrücken

Darstellung in Tabellenform: $X = \{ x_1, \dots, x_n \}$
 $Y = \{ y_1, \dots, y_m \}$
 $Z = \{ z_1, \dots, z_k \}$

Überföhrungsfunktion

$Z \backslash X$	x_1	x_2	...	x_n
z_1	$f(z_1, x_1)$	$f(z_1, x_2)$	...	$f(z_1, x_n)$
z_2	$f(z_2, x_1)$	$f(z_2, x_2)$	...	$f(z_2, x_n)$
...	...	...	...	...
z_k	$f(z_k, x_1)$	$f(z_k, x_2)$	...	$f(z_k, x_n)$

Ausgabefunktion

$Z \backslash X$	x_1	x_2	...	x_n
z_1	$g(z_1, x_1)$	$g(z_1, x_2)$	...	$g(z_1, x_n)$
z_2	$g(z_2, x_1)$	$g(z_2, x_2)$	...	$g(z_2, x_n)$
...	...	...	...	...
z_k	$g(z_k, x_1)$	$g(z_k, x_2)$	...	$g(z_k, x_n)$

Bemerkung: Oft werden die Tabellen für f und g zu einer einzigen zusammengefasst mit den Eintragungen

$$f(z_i, x_j) ; g(z_i, x_j) \quad (i = 1(1)k ; j = 1(1)n)$$

Darstellung mittels gerichteter Graphen:

Definition: Das geordnete Paar $G = (K, R)$ heißt **gerichteter Graph**, wenn K eine nichtleere Menge und $R \subseteq K \times K$, d.h. R ist eine zweistellige Relation.
 K : Knotenmenge, R : Kantenmenge,
für $(k, k') \in R$ heißt k Anfangsknoten und k' Endknoten.

Dem Mealy - Automaten $A = (X, Y, Z, f, g)$ wird der gerichtete Graph $G_A = (Z, R)$ mit $R = \{ (z, z') : z, z' \in Z \wedge \text{es ex. } x \in X : f(z, x) = z' \}$ zugeordnet.

Dabei wird eine Kante (z, z') bei $f(z, x) = z'$ mit x und $y = g(z, x)$ beschriftet. Dieser Graph heißt **Zustandsgraph von A**.

Beispiel: Es soll ein Mealy - Automat entworfen werden, der ein Eingabewort um ein Zeichen versetzt reproduziert. Das Ausgabewort beginnt immer mit # , das letzte Zeichen des Eingabewortes geht verloren.

Dabei sei $X = \{0, 1\}$, also $Y = \{\#, 0, 1\}$

z.B. Eingabe: 011011
 Ausgabe: #01101

Konstruktionsprinzip: Zustand z2:
Zustand z3:

letztes eingeg. Zeichen 0
letztes eingeg. Zeichen 1

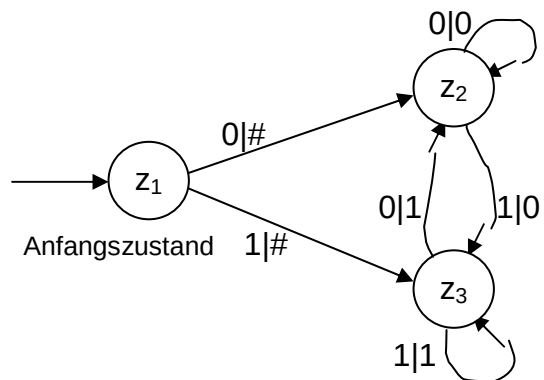


Tabelle:		
	0	1
z_1	$z_2, \#$	$z_3, \#$
z_2	$z_2, 0$	$z_3, 0$
z_3	$z_2, 1$	$z_3, 1$

