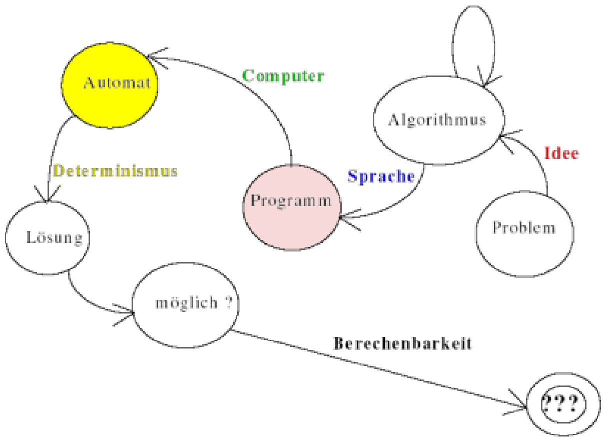
Automaten & Formale Sprachen

(Teilgebiete der Theoretischen Informatik)

Einführung



Die Theoretische Informatik liefert zusammenfassend allgemeine mathematische Grundlagen der Information, ihrer Darstellung sowie effizienten Verarbeitung und besteht aus den folgenden größeren Teilgebieten, die alle eng miteinander verzahnt sind:

- **Algorithmentheorie** – erforscht und analysiert Algorithmen:
    - **Definition**: Ein **Algorithmus** ist ein, als Folge von Anweisungen beschriebenes, schrittweise ablaufendes Verfahren zur Lösung eines Problems, das folgenden Bedingungen genügt:
        - *Finitheit*: Anweisungsfolge ist endlich
        - *Universalität*: Lösung für eine größere Klasse von Aufgaben
        - *Terminiertheit*: nach endlich vielen Schritten beendet
        - *Determiniertheit*: gleiche Eingabe führt zu gleicher Ausgabe
        - *Determinismus*: zu jeder Zeit ist die nächste Anweisung eindeutig festgelegt

        (Algorithmen werden veranschaulicht durch: Ablaufdiagramme, Pseudocode, Programmiersprache, (umgangssprachlich) verbal)
- **Automatentheorie** – beschäftigt sich mit dem Studium von Automaten (Modellrechnern) - z.B. endliche Automaten, Kellerautomaten, Turingmaschinen - und mit den von diesen Automaten lösbaren Problemen
- **Berechenbarkeitstheorie** – Hauptfragen: Welche Art Aufgaben kann welche Klasse von Maschinen (Automaten – s.o.) lösen? Welche Art von Problemen benötigt leistungsfähigere Maschinen? Welche Probleme sind überhaupt lösbar?
- **Komplexitätstheorie** – betrachtet die Komplexität (Rechenzeit, Speicherplatzbedarf) von algorithmisch behandelbaren Problemen auf verschiedenen, mathematisch definierten, formalen Rechnermodellen
- **Formale Sprachen** (Grundlage der Entwicklung der Programmiersprachen) – betrachtet formalisierte Grammatiken und die durch diese Grammatiken erzeugten formalen Sprachen; das Problem, ob ein Wort zu einer formalen Sprache gehört, wird durch Automaten gelöst.

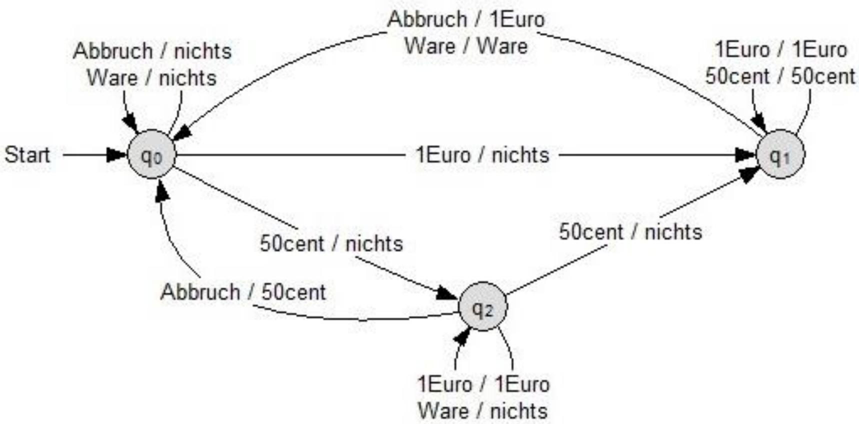
Automatentheorie – Einführung am Beispiel

(Alltagsbeispiele: Getränke-, Süßwaren-, Spiel-, Geld-, Ticketautomat)

<u>**Unser erster (Getränke-)Automat:**</u> 1 Getränk für 1€ erwerbbar, akzeptierte Münzen: 1€ oder 50 cent, 2 Knöpfe: Abbruch bzw. Ware, Form der Übergänge: Eingabe (Aktion) / Ausgabe

(Übergangs-)

Diagramm:



Definition:

Ein **so genannter MEALY-Automat** besteht aus

- einem *Eingabealphabet* X (endliche, nichtleere Menge der zulässigen Zeichen)
- einem *Ausgabealphabet* Y (endliche, nichtleere Menge von Ausgabesymbolen)
- einer *Zustandsmenge* Z (mit *genau einem Anfangszustand* z_0)
- einer *Übergangsfunktion* $f_{ü}: X\ x\ Z \rightarrow Z$, die jeder Zeichen-Zustandskombination genau einen Zustand zuordnet
- einer *Ausgabefunktion* $f_a: X\ x\ Z \rightarrow Y$, die jeder Zeichen-Zustands-Kombination genau einen Ausgabewert zuordnet

Ist eine der letzten beiden Relationen keine Funktion – d.h. es existieren Zuordnungen, die nicht eindeutig sind – so heißt der Automat **nichtdeterministischer endlicher Automat (NEA)**.

Zurück zum obigen Getränkeautomaten:

Übergangsfunktion (hier $\delta = f_{ü}$):

δ	1Euro	50cent	Abbruch	Ware
q_0	q_1	q_2	q_0	q_0
q_1	q_1	q_1	q_0	q_0
q_2	q_2	q_1	q_0	q_2

Ausgabefunktion (hier $\lambda = f_a$):

λ	1Euro	50cent	Abbruch	Ware
q_0	nichts	nichts	nichts	nichts
q_1	1Euro	50cent	1Euro	Ware
q_2	1Euro	nichts	50cent	nichts

Arbeitsweise eines endlichen Automaten

Ein endlicher Automat arbeitet in diskreten Schritten. Zu einem gewissen Zeitpunkt befinde er sich im Zustand z. Falls ein Eingabezeichen e vorhanden ist, liest er dieses beim nächsten Takt und geht in den Zustand $z_{neu} = f_{ü}(z,e)$ über, wobei er das Zeichen $a = f_a(z,e)$ ausgibt. Der Automat beginnt seine Arbeit definitionsgemäß im Startzustand z_0. Dieser Vorgang lässt sich verdeutlichen, wenn wir die für den Automaten vorgesehenen Eingaben auf einem Eingabeband, die von ihm erstellten Ausgaben auf einem Ausgabeband und seinen aktuellen Zustand sowie die Funktionen f und g in einer Steuereinheit zusammenfassen: