

Automatentheorie

(Quelle: <http://www.ifi.unizh.ch/cl/Glossar/Automatentheorie.html>)

Ursprung

griech. automatos - sich von selbst bewegend

Definition

Die Automatentheorie befasst sich mit der formalen mathematischen Beschreibung und Untersuchung von [Automaten](#), d.h. von Modellen diskreter sequentieller informationsverarbeitender Systeme. Die Automatentheorie befasst sich weniger mit der internen Struktur als mit dem Verhalten solcher Systeme; dies wird wiederum darauf reduziert, welche Folgen von Eingaben in das System welche Ausgaben erzeugen können. Bei unendlichen (wachsenden) Automaten spielt jedoch auch die Art und Weise des Wachstums eine wichtige Rolle. Bei Automaten mit potentiell unendlichem Speicher ist die Art des Speicherzugriffs und die Folge der Speicherinhalte wichtig.

Oft werden akzeptierende Automaten ([Akzeptor](#)) betrachtet, bei denen nur ihre akzeptierte Wortmenge interessiert, d.h. die Menge endlicher Eingabefolgen (Wörter), die den Automat von einem (oder mehreren) Anfangszuständen in einen von (im allgemeinen) mehreren Endzuständen überführen; die Ausgabe besteht also jeweils nur aus einem "Ja" oder "Nein" nach Ende der Eingabe. Symmetrisch dazu werden auch erzeugende Automaten ([Generator](#)) betrachtet.

Ferner behandelt man nicht nur [deterministisch](#) arbeitende Automaten (bei denen die Änderung des jeweiligen Zustands und die jeweilige Ausgabe funktional nur von der momentanen Eingabe und dem aktuellen Zustand abhängt), sondern auch [nicht-deterministische Automaten](#), bei denen Relationen anstelle von Funktionen die mögliche Arbeitsweise beschreiben. Bei der Beschreibung nicht-deterministischer Automaten ist es erlaubt, dass zu einem Paar aus momentaner Eingabe und aktuellem Zustand mehrere mögliche Nachfolgezustände und Angaben angegeben werden, wobei im konkreten Ablauf jeweils nicht-deterministisch eine Wahl getroffen werden muss. So wird also von einem nicht-deterministischen Akzeptor eine Eingabefolge genau dann akzeptiert, wenn sie bei geschickter Auswahl aus den jeweils erlaubten Folgezuständen von einem der Anfangs- zu einem der Endzustände führen kann.

Klassen von Akzeptoren entsprechen Klassen von Wortmengen (d.h. [formale Sprachen](#)). Die bekannteste Klasseneinteilung bei Akzeptoren erfolgt über die Art der Speicher:

- Endliche Akzeptoren([endliche Automaten](#)) haben endliche Speicherkapazität, die von ihnen akzeptierbaren Wortmengen sind die [regulären Sprachen](#);
- Akzeptoren mit Kellerspeicher ([Kellerautomaten](#)) entsprechen [kontextfreien Sprachen](#);
- Akzeptoren mit linear beschränktem Speicher ([linear beschränkte Automaten](#)) entsprechen [kontextsensitiven Sprachen](#);
- [Turing-Maschinen](#) charakterisieren die allgemeinsten mit endlichen Mitteln erzeugbaren formalen Sprachen, die aufzählbaren (oder Typ-0-)Sprachen.

Diese Aussagen beziehen sich jeweils auf die nicht-deterministische Version - bei Kellerautomaten sind die deterministischen weniger leistungsfähig und bei linear beschränkten Automaten ist noch immer unbekannt, ob Determinismus zu einer echten Einschränkung führt.

Automaten

Ursprung

griech. automatos - sich von selbst bewegend

Definition

Automaten sind Maschinen, die abstrakte Prozeduren vollziehen. Es werden vier Typen von abstrakten Automaten unterschieden: [endliche Automaten](#) (finite state automata), [Kellerautomaten](#) (push down automata), [linear beschränkte Automaten](#) (linear bounded automata) und [Turing-Maschinen](#).

Akzeptor

Definition

Ein Akzeptor ist ein [Automat](#) (oder [Algorithmus](#)), welcher die syntaktische Korrektheit einer Eingabekette ("string") testet (also ermittelt, ob die Eingabekette ein Satz der Sprache ist). Als Output wird lediglich "yes" ausgegeben, wenn der Input syntaktisch korrekt ist, oder "no" in allen anderen Fällen. Syntaktische Korrektheit wird relativ zur durch den Automaten definierten Sprache bestimmt.

Generator

Ursprung

lat. generare - erzeugen

Definition

Ein Generator ist ein [Programm](#), welches anhand der [Grammatik](#) des Programms einen Satz erzeugt. Als möglicher Output gelten alle Sätze, die gemäss der Grammatik zulässig sind.

Endlicher Automat

(engl.) Finite State Automaton

Ursprung

griech. automatos - sich von selbst bewegend

Definition

Ein endlicher Automat ist definiert durch eine Anzahl von Zuständen und eine Anzahl von Relationen, die jedem Zustand und jeder möglichen Eingabe aus einem endlichen Repertoire von Eingabewerten einen oder mehrere Folgezustände zuordnen.

Es gibt zwei Varianten: Den [deterministischen](#) und den [nicht-deterministischen](#) endlichen Automaten. Beim deterministischen endlichen Automaten gibt es für einen Zustand und einen Eingabewert nur einen einzigen Folgezustand.

In der Sprachverarbeitung werden Endliche Automaten (u.a.!) für die Morphologieanalyse verwendet, für das Identifizieren von Benannten Einheiten ("named entities") oder auch für die Syntaxanalyse, wenn eine Annäherung einer korrekten grammatischen Beschreibung ausreicht, um die geforderten Ergebnisse zu erzielen. Siehe auch [Finite State Grammar](#)

Kellerautomat

(engl.) Push Down Automaton

Ursprung

griech. automatos - sich von selbst bewegend

Definition

Die Leistungsfähigkeit [Endlicher Automaten](#) hat Grenzen: Es fehlt ein Gedächtnis, um sich beliebig tief geschachtelte Konstruktionen zu merken. Sprachen mit derartigen geschachtelten Strukturen sind [kontextfrei](#) und können demnach mit [Akzeptoren](#) nicht erkannt werden. Aufgabe des Kellerautomaten ist es, den Anfang und das Ende von geschachtelten Konstruktionen zu erkennen. Dazu braucht er ein Gedächtnis. Als Gedächtnis dient dem Automat ein [Kellerspeicher](#) oder Stapel (stack). Der Kellerautomat besitzt wie die Akzeptoren endlich viele Zustände. Ausgehend vom Startzustand endet der Automat nach endlich vielen Schritten entweder im Endzustand oder im Falschzustand. In Abhängigkeit vom Eingabezeichen, vom aktuellen Zustand und vom obersten Kellersymbol wird ein Zustandswechsel ausgelöst und die Kellerspitze verändert. Im Keller können beliebig viele Symbole abgelegt werden. Im Kellerspeicher kann aber nur das oberste Zeichen verändert werden. Es stehen dafür drei Speicheroperationen zur Verfügung: Push (legt ein Zeichen zuoberst in den Speicher), Pop (entfernt das oberste Kellerzeichen) und # (ändert den Keller nicht). Es gilt somit das Prinzip: [Last In First Out](#) (LIFO). Das neue Zeichen wird zuoberst in den Speicher gelegt. Es können in einem Schritt auch mehrere Zeichen hinterlegt werden. Das zuletzt hinterlegte Zeichen wird wieder zuerst entfernt.

Linear Beschränkte Automaten

(engl.) Linear Bounded Automata

Ursprung

griech. automatos - sich von selbst bewegend

Definition

Ein [Automat](#) heisst linear beschränkt, wenn er mit einem endlichen Band auskommt. Solche Automaten sind in der Praxis von Bedeutung, da Computer im allgemeinen nur über endlich viel Speicher verfügen. Das Band dieser Automaten ist links und rechts jeweils durch zwei unterschiedliche Steuerzeichen begrenzt, die nicht überfahren und auch nicht überschrieben werden können. Linear beschränkte Automaten akzeptieren Sprachen, die mindestens vom Typ 1 ([kontextsensitive Sprachen](#)) der [Chomsky-Hierarchie](#) sind.

Turing-Maschine

Definition

Eine Turing-Maschine (Universeller Automat) ist ein von Alan M. Turing (1912-1954) entworfenes (und nach ihm benanntes) Gedankenmodell einer universellen Rechenmaschine mit einem unendlich grossen Speicher. Die Steuereinheit der Turing-Maschine besteht wie bei den [Kellerautomaten](#) aus endlich vielen Zuständen. Der Kellerspeicher wird ersetzt durch ein beidseitig unbegrenztes Arbeitsband, welches als Eingabe-, Ausgabe- und Speichermedium benutzt wird. Das Arbeitsband ist unterteilt in Felder, die jeweils ein Zeichen eines Bandalphabets aufnehmen können. Der Lese-Schreib-Kopf (LS-Kopf) kann sich beliebig über das Band bewegen und ein Feld lesen oder neu beschreiben. Arbeitsweise der Turing-Maschine:

- ein Symbol auf dem Arbeitsfeld wird gelesen,
- abhängig vom gelesenen Symbol und dem aktuellen Zustand schreibt der LS-Kopf ein Symbol auf das Arbeitsfeld, läuft der LS-Kopf entweder ein Feld nach links oder rechts oder verharrt auf dem Feld, und die Maschine bleibt im selben Zustand oder geht in einen anderen über.

Hinsichtlich der Äquivalenz zwischen [Automaten](#) und [Grammatiken](#) von [natürlichen Sprachen](#) entspricht die Turing-Maschine den [allgemeinen Sprachen](#) (Typ 0 der [Chomsky-Hierarchie](#)) und ist somit die mächtigste Maschine aller Automaten, da sie eine rekursiv aufzählbare Menge von Ketten (Sätzen) zu erzeugen vermag.

Grammatik

Ursprung

griech. grammatike (techne) zu grammtikos - die Buchstaben betreffend

Definition

Der Begriff Grammatik wird verwendet, um mehrere Wissensbereiche zu bezeichnen: Traditionsgemäss sind dies die morphologischen und syntaktischen Eigenschaften einer [menschlichen Sprache](#). Folgende Bereiche können mit dem Begriff Grammatik abgedeckt werden:

- Ein System struktureller Regeln, welche die Grundlage der linguistischen [Generierung](#) und des Sprachverständnisses sind.
- Eine Sprachtheorie oder ein Modell linguistischer [Kompetenz](#).
- Systematische Beschreibung der formalen Regularitäten einer natürlichen Sprache in Form eines Nachschlagewerkes oder Lehrbuchs.

Die Charakteristika solcher Grammatiken variieren je nach ihrem beabsichtigten Anwendungsbereich. In der Sprachtechnologie beschreibt eine Grammatik (die elektronisch aufgenommen wurde) normalerweise die Struktur einer Sprache auf verschiedenen Ebenen: Wort (morphologische Grammatik), Phrase, Satz usw. Eine Grammatik kann die Struktur sowohl hinsichtlich der Oberfläche ([Syntax](#)) als auch der Bedeutung ([Semantik](#) und [Diskurs](#)) beleuchten.

Reguläre Grammatik

Definition

Die Reguläre Grammatik (Finite State Grammar) ist ein [Grammatikmodell](#), welches auf der linearen Struktur der Sprache beruht. Es dient der Erzeugung einer unendlichen Menge von Sätzen mit Hilfe einer endlichen Menge von [rekursiven](#) Regeln über einem endlichen Wortschatz. Die Produktion von Sätzen verläuft von links nach rechts, d.h. die Wahl des am weitesten links stehenden Elements (= erstes Wort im Satz) determiniert die Wahlmöglichkeit für das unmittelbar folgende Element usw. Die in einer Sprache gültigen Wahlmöglichkeiten können in einem Zustandsdiagramm dargestellt werden, das als Anweisung an einen [Automaten](#) zu interpretieren ist, die in der jeweiligen Sprache grammatischen Sätze zu erzeugen. Der Vorteil des Grammatikmodells liegt in seiner formalen Einfachheit. Die Finite State Grammar ist zur Beschreibung [natürlicher Sprachen](#) jedoch nicht ausreichend, weil auch zwischen nicht unmittelbar aufeinanderfolgenden Elementen syntaktische Abhängigkeiten bestehen können (z.B. wenn ein Relativsatz eingeschoben wird).

Natürliche Sprache

Definition

Im Gegensatz zu den [Programmiersprachen](#) werden die menschlichen Sprachen als natürliche Sprachen bezeichnet. Die automatische Verarbeitung der natürlichen Sprachen ist der Forschungsbereich der [Computerlinguistik](#).

Formale Sprachen

Ursprung

lat. formalis - die Form betreffend

Definition

Im Unterschied zu [natürlichen Sprachen](#) basieren formale (oder auch künstliche, logische) Sprachen auf Sprachsystemen, die von der [Logik](#) und/oder der Mathematik konstruiert wurden. Formale Sprachen zeichnen sich durch Eindeutigkeit, Explizitheit und leichte Überprüfbarkeit aus. Zur Definition formaler Sprachen werden ein Alphabet sowie Regeln für die Sprache benötigt.

Beispiel

Alphabet: a, b; Regeln: Das kürzeste Wort der Sprache ist ab, Ein Wort der Sprache muss genausoviele a wie b enthalten, Alle a müssen vor den b stehen; Beispielwörter der Sprache: ab, aabb, aaabbb, aaaabbbb usw.

Chomsky-Hierarchie

Ursprung

Bezeichnung nach dem amerikanischen Linguisten N. Chomsky

Definition

Die Chomsky-Hierarchie ist ein Rangieren der Sprachen nach ihrer syntaktischen Komplexität und Ausdruckskraft, das von [Noam Chomsky](#) definiert wurde. Die Sprachen des Typs 3 sind dabei diejenigen, welche am wenigsten komplex und ausdrucksstark sind, Sprachen des Typs 0 sind am komplexesten und ausdrucksstärksten.

- reguläre Sprachen (Typ 3);
- kontextfreie Sprachen (Typ 2);
- kontextsensitive Sprachen (Typ 1);
- allgemeine (unbeschränkte) Sprachen (Typ 0).

Reguläre Sprachen

Definition

Die regulären Sprachen entsprechen dem Typ 3 in der [Chomsky-Hierarchie](#). Aus einer regulären Sprache kann ein [endlicher Automat](#) erstellt werden, umgekehrt kann zu jedem endlichen Automaten eine reguläre Sprache gefunden werden, die der Automat erkennt. [Formale Sprachen](#) L heissen genau dann regulär, wenn es eine lineare [Grammatik](#) G mit $L = L(G)$ gibt.

In einer Phrasenstruktur-Regel der regulären Sprachen steht auf der linken Regelseite genau ein [Nonterminal](#). Auf der rechten Seite steht genau ein [Terminal](#), gefolgt von höchstens einer Variable. (Dies ist die Definition von [rechtslinearen](#) Phrasenstruktur-Grammatiken. PS-Grammatiken, bei denen höchstens eine Variable vorausgehen darf, heissen [linkslinear](#).)

Beispiel

$A \rightarrow b, A \rightarrow bC$

Linkslineare Grammatik

Definition

Eine [Grammatik](#) G heisst linkslinear, wenn alle Produktionen die Form $A \rightarrow Bx$ haben, wobei B auch leer sein kann, nicht jedoch x ! Dann ist die Klasse der Sprachen, die durch linkslineare Grammatiken erzeugt werden die selbe, die auch durch rechtslineare Grammatiken erzeugt werden, nämlich die Klasse der [regulären Sprachen](#).

rechtslineare Grammatik

Definition

Eine [Grammatik](#) G heisst rechtslinear, wenn alle Produktionen die Form $A \rightarrow xB$ haben, wobei B auch leer sein kann, nicht jedoch x ! Dann ist die Klasse der Sprachen, die durch rechtslineare Grammatiken erzeugt werden die selbe, die auch durch [linkslineare Grammatiken](#) erzeugt werden, nämlich die Klasse der [regulären Sprachen](#).

Kontextfreie Sprachen

Ursprung

lat. con-textus - Zusammenhang

Definition

Die Klasse der durch [Kellerautomaten](#) darstellbaren Sprachen ist gleich der Klasse der kontextfreien Sprachen. Dies entspricht [Grammatiken](#) vom Typ 2 der [Chomsky-Hierarchie](#). Sprachen niedrigeren Typs (0 oder 1) können nicht von Kellerautomaten erkannt werden. Daraus ergibt sich, dass jede kontextfreie Sprache von einem Kellerautomaten erkannt wird. Umgekehrt ist jede Sprache kontextfrei, die von einem Kellerautomaten erkannt werden kann. Bei einer kontextfreien Phrasenstruktur-Regel steht auf der linken Regelseite genau eine Variable. Auf der rechten Regelseite steht eine Zeichenkette aus V^+ ([Kleene-plus](#))

Beispiel

$A \rightarrow BC, A \rightarrow bBCc$. Die meisten Programmiersprachen lassen sich kontextfrei beschreiben.

Kontextsensitive Sprachen

Ursprung

lat. con-textus - Zusammenhang

Definition

Die kontextsensitiven Sprachen gehören zum Typ 1 der [Chomsky-Hierarchie](#). In Sätzen kontextsensitiver Sprachen kommen überlappende Abhängigkeiten zwischen den einzelnen Satzteilen vor. Ein [Kellerautomat](#) reicht daher nicht aus, um diese Strukturen zu erkennen, da die Abarbeitungsfolge nicht symmetrisch angeordnet ist (d.h. es besteht keine zentrale Einbettung). Sie ist beliebig und verlangt daher nach einem [Automaten](#) mit beliebigem Speicherzugriff. Kontextsensitive Sprachen werden anhand [Linear Beschränkter Automaten](#) analysiert.

Bei einer kontextsensitiven Phrasenstrukturregel wie im untenstehenden Beispiel stehen auf der linken und rechten Regelseite beliebige Folgen von [Terminalen](#) und Variablen, wobei die rechte Regelseite mindestens so lang sein muss wie die linke.

Beispiel

$A B C \rightarrow A D E C$. Die einzigen beweisbar kontextsensitiven Konstruktionen natürlicher Sprache wurden im Zürichdeutsch gefunden. *Ich glaub das mer am Hans s'Huus hälfed aastriche.*

Allgemeine Sprachen

Definition

Die allgemeinen Sprachen (oder auch unbeschränkte Sprachen) entsprechen dem Typ 0 in der [Chomsky-Hierarchie](#). Sie sind so definiert, dass die Ersetzungsregeln gar keinen Beschränkungen mehr unterworfen sind. Eine Sequenz von [nicht-terminalen Symbolen](#) darf also in eine völlig andere Sequenz von Nichtterminalen umgeschrieben werden oder auch durch leere Ketten ersetzt werden. Die [Automaten](#), welche allgemeine Sprachen erkennen, heißen [Turing-Maschinen](#).

Beispiel

Zu den allgemeinen [Sprachen](#) gibt es keine natürlichsprachlichen Beispiele.

Deterministisch

Ursprung

lat. determinare - bestimmen

Definition

Eine Prozedur ist deterministisch, wenn zu jedem Zeitpunkt während des Prozesses bestimmt ist, wie [weitergefahren](#) werden soll. Im Bereich der Programmierung wird manchmal ein Unterschied zwischen [determiniert](#) und deterministisch gemacht: Lässt ein terminierendes Programm bei ein und derselben Eingabe verschiedene Abläufe zu, die aber alle stets das gleiche Resultat liefern, so nennt man es determiniert (wegen des Ergebnisses), aber nicht deterministisch (wegen der Abläufe).

Nicht-Determinismus

Ursprung

lat. determinare - bestimmen

Definition

In der Programmierung meint Nicht-Determinismus das Offenlassen mehrerer Möglichkeiten. Man unterscheidet drei Arten von Nicht-Determinismus im Hinblick auf das Vorgehen bei der Programmausführung:

- bei böswilligem Nicht-Determinismus wird stets die schlechte Möglichkeit (z.B. ein Ablauf, der nicht zu Terminierung führt) gewählt;
- bei gutwilligem Nicht-Determinismus wird stets eine gute Möglichkeit (z.B. eine, die zu einem Endzustand führt) gewählt;
- bei willkürlichem Nicht-Determinismus tritt beides (gut oder böse) auf.

In der Automatentheorie verwendet man den gutwilligen Nicht-Determinismus: Ein Automat akzeptiert ein Wort nicht-deterministisch, wenn es unter allen Verarbeitungsmöglichkeiten mindestens eine gibt, die das Akzeptierungskriterium erfüllt.