Logik in Informatik, Mathematik und Philosophie*

Thomas Strahm

Es soll im Folgenden versucht werden, einige wesentliche Aspekte der Logik im interdisziplinären Spannungsfeld zwischen Informatik, Mathematik und Philosophie zu erläutern. Dabei will ich mich weitgehend auf die Entwicklungen der modernen Logik beschränken, welche ihren Ursprung in der Mitte des 19. Jahrhunderts haben. Ich will dabei ein gewisses Schwergewicht legen auf die Bedeutung der Logik für die Grundlagen der Mathematik sowie deren Anwendungen in der Informatik, jedoch stets versuchen, den grossen Einfluss der Arbeiten zur Logik in der Philosophie auf die Mathematik und Informatik zu illustrieren.

So steht es wohl ausser Zweifel, dass Aristoteles (384-322) der Urvater der Logik ist. Sein Ziel war es, objektive Gesetze des menschlichen Denkens zu finden und diese durch Schemata für gültige Argumente oder Schlüsse, sogenannte Syllogismen, zu repräsentieren. Ein Argument ist eine Konfiguration der Form

$$\frac{\textit{Prämissen}}{\textit{Konklusion}}$$

Ausgehend von einer endlichen Menge von *Prämissen* wird die *Konklusion* erschlossen. Betrachten wir etwa den Schluss

$$\frac{\text{Alle } P \text{ sind } Q, \quad a \text{ ist } P}{a \text{ ist } Q},$$

so ist dabei augenfällig, dass die Prämissen des Schlusses die Konklusion erzwingen, unabhängig von der konkreten Interpretation der Prädikate P und Q sowie der Individuenkonstante a. So kann man zum Beispiel für P das Prädikat *Mensch*, für Q das Prädikat *sterblich* und für a das Individuum *Sokrates* wählen, um folgende "berühmte" Instanz des obigen Schlusses zu erhalten:

$$\frac{\text{Alle } \textit{Menschen} \text{ sind } \textit{sterblich}, \quad \textit{Sokrates} \text{ ist ein } \textit{Mensch}}{\textit{Sokrates} \text{ ist } \textit{sterblich}}$$

Genau diese Unabhängigkeit der Interpretation von Prädikaten und Konstanten ist das wesentliche Charakteristikum von gültigen Argumenten oder Schlussfiguren. Wir werden darauf im Folgenden noch einmal zu sprechen kommen.

Die aristotelischen Ideen wurden von den Stoikern aufgenommen und vor allem in der Scholastik wesentlich weiter entwickelt. Der Einfluss dieser Entwicklungen auf die moderne Logik ist beträchtlich.

*Vortrag gehalten am 29. Januar 1999 anlässlich der Veranstaltung zum Theodor-Kocher-Preis 1998 der Universität Bern.

Die Ursprünge der modernen Logik

Die moderne Logik ist eine symbolisch-mathematische Logik, welche auf einer mathematischen Kunstsprache aufbaut. Charakteristisch für die moderne Logik ist ihre *Kalkülisierung*, d.h. neue Theoreme und Schlussschemata werden, ausgehend von vorgegebenen Axiomen und Regeln, auf gleichsam mechanische Art und Weise abgeleitet. Als wichtiger Vorläufer der modernen Logik gilt Gottfried Wilhelm Leibniz (1646-1716). Seine logischen Schriften wurden grösstenteils erst lange nach seinem Tode veröffentlicht.

Die eigentlichen Begründer der modernen Logik sind der englische Mathematiker George Boole (1815-1864) und der deutsche Philosoph und Mathematiker Gottlob Frege (1848-1925). Boole hat 1847 in seinem einflussreichen Aufsatz "The mathematical analysis of logic, being an essay towards a calculus of deductive reasoning" die heutige *Aussagenlogik* begründet.

Die formale Sprache der Aussagenlogik erlaubt es, ausgehend von Platzhaltern $P, Q, R, \ldots$ für atomare Aussagen, mit Hilfe von aussagenlogischen Junktoren komplexere Aussagen zu bilden: zu gegebenen Aussagen A und B hat man neue Aussagen $\neg A$ ("*nicht* A"), $A \wedge B$ ("A *und* B"), $A \vee B$ ("A *oder* B") sowie $A \to B$ ("A *impliziert* B", "*Aus* A *folgt* B"). Ein *Kalkül* der Aussagenlogik basiert nun auf endlich vielen Axiomenschemata und Herleitungsregeln, mit deren Hilfe *gültige* Aussagen, d.h. Aussagen, die wahr sind unabhängig vom Wahrheitswert ihrer atomaren Teilaussagen, abgeleitet werden können. Beispiele für ein Axiomenschema, beziehungsweise für eine Herleitungsregel, sind etwa das Gesetz des ausgeschlossenen Dritten (Tertium non datur) und der Modus Ponens:

$$A \vee \neg A, \qquad \frac{A \quad A \to B}{B}$$

Für beliebige Aussagen A gilt entweder A oder "*nicht* A", und falls A und $A \to B$ abgeleitet worden sind, so darf auch B abgeleitet werden. Etwas formaler ist ein *Beweis* oder eine *Herleitung* einer Aussage A in einem solchen Kalkül dann einfach eine endliche Folge von Aussagen, deren letztes Glied gerade A ist und die korrekt ist bezüglich der Axiome und Herleitungsregeln des Kalküls. Herleitungen in einem Kalkül können also gewissermassen auf "mechanische" Art und Weise konstruiert werden.

Gottlob Frege hat in seiner im Jahre 1879 verfassten "Begriffsschrift" die sogenannte *Prädikatenlogik erster Stufe* begründet, welche bis heute das zentrale System der Logik darstellt und die Grundlage für die meisten Anwendungen der Logik in Mathematik, Philosophie und Informatik bildet. Die Prädikatenlogik ist eine Erweiterung der Aussagenlogik mit der neuen Ausdrucksmöglichkeit der sogenannten *Existenz- und Allquantifikation*. Eine entsprechende Sprache basiert auf Variablen $x, y, z, \ldots$ für Individuen sowie Symbolen $P, Q, \ldots$ für i.a. mehrstellige Prädikate oder Relationen, mit deren Hilfe atomare Aussagen der Form $P(x)$, $Q(x, y, z)$ etc. gebildet werden können. Darauf aufbauend können neue, komplexere Aussagen durch Verwendung der aussagenlogischen Junktoren $\neg$, $\wedge$, $\vee$ und $\to$ wie oben geformt werden, und zusätzlich hat man für jede Aussage $A(x)$ neue Aussagen $\exists x A(x)$ ("$A(x)$ *trifft auf mindestens ein* x *zu*") und $\forall x A(x)$ ("$A(x)$ *trifft auf alle* x *zu*"). Ein Kalkül für die Prädikatenlogik erster Stufe erweitert den aussagenlogischen Kalkül um zusätzliche Axiomenschemata und Herleitungsregeln, z.B.

$$A(y) \to \exists x A(x), \qquad \frac{A(x)}{\forall x A(x)}$$

Eine zentrale Unterscheidung in fast allen Bereichen der Logik ist diejenige zwischen *Syntax* und *Semantik*. Während formale Zeichenreihen für Aussagen sowie Axiome und Regeln zum

Ableiten von neuen Ausdrücken zur Syntax gehören, befasst sich die Semantik mit der Interpretation und Bedeutung von syntaktischen Ausdrücken. Im Falle einer Sprache der Prädikatenlogik erster Stufe beispielweise ist eine Interpretation gegeben durch die Festlegung eines Individuenbereiches, über den die Individuenvariablen laufen, sowie durch eine Interpretation der Prädikatsymbole der Sprache als Prädikate oder Relationen auf diesem Individuenbereich; die aussagenlogischen Junktoren haben dann ihre übliche Bedeutung, und Existenz- sowie Allquantifikationen werden über dem gegebenen Individuenbereich interpretiert. Damit ist eine Aussage in einer Interpretation entweder wahr oder falsch.

Die Vollständigkeit der Prädikatenlogik

Wie wir schon oben erwähnt haben, sind die sogenannten *(logisch) gültigen Aussagen* von zentralem Interesse, d.h. diejenigen Aussagen, welche unter *allen* Interpretationen wahr sind. Solche Aussagen gelten aufgrund ihrer *logischen Struktur* und nicht aufgrund der spezifischen Interpretation der in ihr auftretenden Prädikatsymbole. So ist beispielsweise leicht einzusehen, dass die folgenden beiden Aussagen gültig sind:

$$\forall x(P(x) \to P(x)), \qquad \exists x \forall y Q(x,y) \to \forall y \exists x Q(x,y)$$

Wenn wir hingegen bei der zweiten Aussage das Vorder- und Hinterglied der Implikation vertauschen, also stattdessen

$$\forall x \exists y Q(x,y) \to \exists y \forall x Q(x,y)$$

betrachten, so erhalten wir eine Aussage, die *nicht gültig* ist: Beispielsweise kann man als Individuenbereich die natürlichen Zahlen $\mathbb{N} = \{0, 1, 2, \ldots\}$ wählen und Q als die Kleiner-Relation auf $\mathbb{N}$ interpretieren; unter dieser Interpretation ist $\forall x \exists y Q(x,y)$ wahr, jedoch $\exists y \forall x Q(x,y)$ falsch, insgesamt ist also die obige Aussage falsch in der Struktur $(\mathbb{N}, <)$.

Bis um 1930 wurde das Herleitungssystem der Prädikatenlogik erster Stufe verwendet ohne die Gewissheit, dass sich darin tatsächlich *alle* gültigen Aussagen ableiten lassen. Diese zentrale Eigenschaft der *Vollständigkeit* der Logik erster Stufe wurde im Jahre 1929 vom österreichischen Mathematiker Kurt Gödel (1906-1978) bewiesen.

Vollständigkeit der Prädikatenlogik erster Stufe (Gödel 1929)
Alle logisch gültigen Aussagen sind herleitbar im System der Prädikatenlogik erster Stufe.

Die Vollständigkeit der Prädikatenlogik ist ein bedeutendes Ergebnis. Es ist die erste in einer Reihe von bahnbrechenden Einsichten Kurt Gödels. Sein berühmtestes Theorem wird uns im nächsten Teilkapitel noch beschäftigen.

Bisher haben wir uns nur mit der reinen Prädikatenlogik erster Stufe befasst und mit denjenigen Aussagen, die *logisch* gültig sind. Bei der Axiomatisierung von konkreten Teilbereichen der Mathematik betrachtet man Mengen T von zusätzlichen, sogenannten nicht-logischen Axiomen und verwendet dann den Formalismus der Prädikatenlogik, um aus T neue Aussagen A abzuleiten. Interessiert man sich beispielsweise für die Theorie der natürlichen Zahlen $\mathbb{N}$, so wählt man zuerst eine geeignete Sprache mit Prädikatsymbolen für grundlegende Eigenschaften auf $\mathbb{N}$, etwa die Beziehung, dass eine Zahl die Summe zweier anderer Zahlen ist. In

einem weiteren Schritt formuliert man eine Menge nicht-logischer Axiome in dieser Sprache, zum Beispiel Axiome über die Addition und Multiplikation oder das Schema der vollständigen Induktion.

Auf den ersten Blick scheint es, dass wir mit der axiomatischen Methode ein universelles Instrument an der Hand haben, um Mathematik zu formalisieren und mathematische Wahrheiten zu beweisen. Dass dem jedoch nicht so ist, soll im nächsten Abschnitt erläutert werden.

Das Hilbertsche Programm und die Gödelschen Unvollständigkeitssätze

Am Ende des vorletzten Jahrhunderts schuf Georg Cantor (1845-1918) das "Paradies" der unendlichen Mengen. Die neue mengentheoretische Sichtweise führte zu einer grundlegenden Wandlung der Beweismethoden in der Mathematik, und ein Rechtfertigungsbedarf der verwendeten Prinzipien war angezeigt. Diese Notwendigkeit einer Grundlegung wurde verstärkt durch die Entdeckung von Widersprüchen in den ersten Formulierungen der Mengenlehre, etwa der berühmten *Russelschen Antinomie*.

Zur Sicherung der Grundlagen der Mathematik hat der deutsche Mathematiker David Hilbert (1862-1943) deshalb ein Programm formuliert. Hilbert war neben Henri Poincaré der einflussreichste Mathematiker seiner Zeit und wahrscheinlich das letzte Universalgenie in der Mathematik. Ziel des Hilbertschen Programms war die Repräsentation der Mathematik in einem universellen formalen System, in dem sich alle wahren mathematischen Aussagen beweisen lassen. Die Widerspruchsfreiheit dieses Systems sollte durch rein kombinatorisches Operieren auf endlichen Beweisfiguren in einem finiten Fragment des Systems selbst nachgewiesen werden.

Hilbertsches Programm
Die gesamte Mathematik wird in einem "grossen" Axiomensystem **M** repräsentiert. Die Widerspruchsfreiheit von **M** wird in einem kleinen finiten Teil **F** von **M** nachgewiesen.

Hilbert setzte grosse Hoffnung in sein Programm, bis im Jahre 1931 der damals 25-jährige Kurt Gödel seine bemerkenswerten und wegweisenden Unvollständigkeitssätze bewies. Gödels Theoreme zeigen, dass, sobald ein mögliches Axiomensystem **M** ein Minimum der Theorie der natürlichen Zahlen umfasst, es immer Aussagen über die natürlichen Zahlen gibt, welche *wahr* aber in **M** *nicht beweisbar* sind. Zudem ist Hilberts Ziel, die Konsistenz von **M** in einem "kleinen" Fragment von **M** nachzuweisen, nicht realisierbar: falls **M** widerspruchsfrei ist, so gehört die Aussage, dass **M** konsistent ist gerade zu den wahren, in **M** nicht beweisbaren Aussagen.

Gödelsche Unvollständigkeitssätze (Gödel 1931)
In jedem widerspruchsfreien Axiomensystem, welches die elementare Arithmetik umfasst, gibt es *arithmetische* Aussagen, die *wahr* aber *nicht beweisbar* sind. Die Widerspruchsfreiheit des Axiomensystems selbst ist eine solche Aussage.

Auch wenn Gödels Ergebnisse in ihrer Aussage negativ sind, so bedeuten sie keineswegs das Scheitern der axiomatischen Methode. Die Unvollständigkeitssätze zeigen nur, dass das Hilbertsche Programm in der Hinsicht modifiziert werden muss, dass wir unseren Glauben an ein einziges Axiomensystem, welches alle Fragen beantwortet, aufgeben müssen. Rückblickend haben sich die Gödelschen Sätze sogar sehr befruchtend auf die Entwicklungen der mathematischen Logik ausgewirkt, und die axiomatische Methode hat seither grosse Bedeutung erlangt.

Auf diese Entwicklungen hatte Kurt Gödel auch nach seinen aufsehenerregenden Ergebnissen einen nachhaltigen Einfluss, welcher sich in richtungsweisenden Arbeiten auf fast allen Gebieten der Logik ausgewirkt hat. Es ist deshalb nicht vermessen, ihn als den herausragendsten Logiker des zwanzigsten Jahrhunderts zu bezeichnen.

Die klassischen Gebiete der mathematischen Logik

Die mathematische Logik lässt sich in die folgenden vier klassischen Gebiete aufteilen: *Beweistheorie*, *Rekursionstheorie*, *Modelltheorie* und *Mengenlehre*.

Die Ursprünge der *Beweistheorie* gehen zurück auf David Hilbert und sein Programm zur Grundlegung der Mathematik. In der Beweistheorie wird ein syntaktischer Zugang zur Logik verfolgt, und es wird versucht, konstruktive Aspekte von mathematischem Schliessen auszunutzen. Wir werden im nächsten Abschnitt auf einige Aspekte der Beweistheorie und der konstruktiven Mathematik etwas ausführlicher zu sprechen kommen.

Der zentrale Begriff, welcher der *Rekursionstheorie* oder *Berechenbarkeitstheorie* zugrunde liegt, ist derjenige des Algorithmus. Dabei steht das Studium von formal-mathematischen Berechenbarkeitsmodellen zur Charakterisierung des intuitiven Berechenbarkeitsbegriffs im Zentrum. Die grundlegenden Ansätze der Rekursionstheorie wurden in den dreissiger Jahren entwickelt und hatten einen nachhaltigen Einfluss auf die Informatik; wir werden uns damit in einem eigenen Kapitel noch detaillierter beschäftigen.

Die *Modelltheorie* untersucht die Beziehungen zwischen einer formalen Sprache und ihren verschiedenen Interpretationen oder Modellen. Eines der ältesten Ergebnisse der Modelltheorie ist das *Theorem von Löwenheim* (1915), welches besagt, dass eine Aussage, die in einer Interpretation mit unendlicher Trägermenge wahr ist, bereits in einem Modell mit *abzählbar* unendlichem Träger erfüllt werden kann. In jüngerer Zeit wurden Methoden der Modelltheorie erfolgreich auf Probleme der Algebra und Analysis angewendet.

Das zentrale System der heutigen *Mengenlehre* geht zurück auf Ernst Zermelo (1871-1953) und Abraham Fraenkel (1891-1965). Die Zermelo-Fraenkelsche Mengenlehre mit Auswahlaxiom ZFC wird allgemein als die adäquate Formalisierung von Cantors zentralen Ideen akzeptiert, und sie hat einen ausgezeichneten Stellenwert als Fundament der gesamten Mathematik. Trotzdem ist es wichtig zu betonen, dass auch ZFC den durch die Gödelschen Unvollständigkeitssätze aufgezeigten Grenzen der Axiomatisierbarkeit unterliegt: falls ZFC widerspruchsfrei ist, so lässt sich diese Aussage in ZFC nicht beweisen; natürlich sind die meisten Mathematiker von der Konsistenz von ZFC überzeugt, jedoch lässt sich diese Überzeugung auf der Grundlage von ZFC nicht formal verifizieren. Sogar scheinbar elementare Fragestellungen über unendliche Mengen lassen sich in ZFC nicht beantworten. So ist etwa das berühmte *Cantorsche Kontinuumsproblem*, d.h. die Frage nach der Kardinalität der Menge der reellen Zahlen, von

den Axiomen von ZFC unabhängig. Dieses herausragende und zentrale Ergebnis geht zurück auf Gödel (1938) und Cohen (1963).

Beweistheorie und Konstruktivität

Die Gödelschen Unvollständigkeitssätze haben gezeigt, dass das Hilbertsche Programm zur Grundlegung der Mathematik modifiziert werden muss. Betrachten wir beispielsweise das Standardaxiomensystem der Zahlentheorie erster Stufe mit Addition, Multiplikation und der Kleiner-Relation auf den natürlichen Zahlen sowie dem Schema der vollständigen Induktion, so ist die Konsistenz der Zahlentheorie eine Aussage, welche sich in der Zahlentheorie nicht beweisen lässt. Selbstverständlich können wir etwa in einer mengentheoretischen Metatheorie sehr leicht nachweisen, dass die Zahlentheorie widerspruchsfrei ist, indem wir zeigen, dass in der Standardstruktur der natürlichen Zahlen, welche in unserer Metatheorie existiert, alle Axiome der Zahlentheorie erster Stufe wahr sind. Ein solcher Konsistenzbeweis ist jedoch in keiner Weise informativ.

Im Jahre 1936 lieferte der deutsche Mathematiker Gerhard Gentzen (1909-1945) einen völlig neuartigen Widerspuchsfreiheitsbeweis für die Arithmetik erster Stufe, welcher wesentlich mehr Information liefert als der naive mengentheoretische Beweis. Gentzen hat gezeigt, dass sich mit der Arithmetik eine ganz bestimmte transfinite Grösse, eine sogenannte Ordinalzahl, in Verbindung bringen lässt, welche sehr genau ihre Beweisstärke charakterisiert. Über einer finiten Grundtheorie lässt sich mit Hilfe eines sehr einsichtigen Induktionsprinzips entlang dieser Ordinalzahl die Konsistenz der Arithmetik nachweisen. Gentzens Ergebnis kann als eine Art Realisierung des Hilbertschen Programms unter gelockerten Rahmenbedingungen angesehen werden und steht deshalb am Anfang des sogenannten *erweiterten Hilbertschen Programmes*. Die Bestimmung von beweistheoretischen Ordinalzahlen von Axiomensystemen gehört heute zum Kern der Beweistheorie. Die Ordinalzahl eines formalen Systems gibt Auskunft über seine arithmetische Beweisstärke, d.h. sie quantifiziert die Kluft zwischen den beweisbaren und den wahren arithmetischen Aussagen.

Ein weiteres wichtiges Teilgebiet der Beweistheorie ist der Entwurf und die Analyse von Axiomensystemen für die mathematische Praxis. Eine zentrale Beobachtung in diesem Zusammenhang ist die Tatsache, dass sich grosse Teile der Mathematik bereits in Axiomensystemen entwickeln lassen, die man als beweistheoretisch sehr schwach bezeichnen kann. Solche Systeme sind in der Regel nur sehr "kleine" Teilfragmente der Zermelo-Fraenkelschen Mengenlehre. Dabei interessiert man sich nicht nur für die Frage, welche Axiome man benötigt, um bestimmte mathematische Sätze zu beweisen, sondern auch umgekehrt, ob die Axiome selbst *aus* diesen Sätzen folgen; das entsprechende Programm ist unter dem Namen *Reverse Mathematics* bekannt.

Ein zentraler Untersuchungsgegenstand der Beweistheorie ist das Studium von konstruktiven Aspekten von Beweisen im Zusammenhang mit einer konstruktiven Grundlegung der Mathematik. Die bekannteste Richtung des Konstruktivismus ist der sogenannte *Intuitionismus*, welcher auf den holländischen Mathematiker L.E.J. Brouwer (1881-1966) zurückgeht. Brouwer war ein scharfer Kritiker der mengentheoretischen Sichtweise und der dadurch entstandenen nicht-konstruktiven Art und Weise des Beweisens. Insbesondere sollten Existenzaussagen in der Mathematik nur dann zulässig sein, wenn entsprechende mathematische Objekte angegeben werden können, welche diese erfüllen. Diese Forderung führte unter anderem zu Brouwers

Ablehnung des sogenannten *Tertium non datur* der klassischen Logik, das heisst der Annahme des Axioms "A oder *nicht* A" für beliebige Aussagen A. Im Folgenden wollen wir anhand eines einfachen mathematischen Satzes zeigen, wie die Verwendung des Tertium non datur zu nicht-konstruktiven Existenzbeweisen führen kann.

Satz *Es gibt irrationale Zahlen a und b, so dass a^b eine rationale Zahl ist.*

Beweis Wir unterscheiden die folgenden zwei Fälle:

1. Fall: $\sqrt{2}^{\sqrt{2}}$ *ist rational.* Dann wählen wir für a und b gerade die Zahl $\sqrt{2}$. Da $\sqrt{2}$ bekanntlich eine irrationale Zahl ist, gilt in diesem Fall die Behauptung des Satzes.

2. Fall: $\sqrt{2}^{\sqrt{2}}$ *ist irrational.* Dann wählen wir für a die nach Voraussetzung irrationale Zahl $\sqrt{2}^{\sqrt{2}}$ und für b wieder $\sqrt{2}$ und erhalten

$$a^b = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\sqrt{2}} = \sqrt{2}^2 = 2.$$

Damit ist auch in diesem Fall a^b eine rationale Zahl. □

Im Sinne der klassischen Mathematik ist unser Beweis einwandfrei. Trotzdem stellt man vielleicht unbefriedigt fest, dass wir aufgrund des gegebenen Argumentes die beiden irrationalen Zahlen a und b, so dass a^b rational ist, nicht wirklich angeben können, da wir nicht wissen, ob der erste oder der zweite Fall im Beweis eintritt. Anders formuliert verwendet der Beweis das Tertium non datur in der Instanz

$$\sqrt{2}^{\sqrt{2}} \text{ ist rational} \quad \textit{oder} \quad \sqrt{2}^{\sqrt{2}} \text{ ist } \textit{nicht} \text{ rational},$$

und wir wissen aufgrund des Beweises nicht, welches der beiden Disjunktionsglieder wahr ist (in der Tat ist es das zweite!).

Brouwer hat die Gesetze der intuitionistischen Logik nie explizit formuliert. Die ersten formalen Ansätze des Intuitionismus gehen auf den Holländer Arned Heyting (1898-1980) zurück, der um 1930 Systeme der intuitionistischen Aussagen- und Prädikatenlogik sowie der intuitionistischen Arithmetik präsentierte. Das Studium von formalen Systemen für konstruktive Mathematik bildet seither einen wichtigen Bestandteil der Beweistheorie und der mathematischen Logik. Es wurde vor allem durch die im Jahre 1967 veröffentlichten Arbeiten von Errett Bishop zu den konstruktiven Grundlagen der Analysis neu belebt. In den siebziger Jahren wurden zahlreiche neue Systeme für Konstruktivismus vorgeschlagen, deren Analyse sich bis heute als äusserst fruchtbar erwiesen hat. Besonders hervorzuheben sind Fefermans Systeme expliziter Mathematik und Martin-Löfs Theorien transfiniter Typen. Es hat sich gezeigt, dass die entsprechenden Ideen auch im Zusammenhang mit Fragestellungen in der theoretischen Informatik von grossem Interesse sind. Wir werden auf diesen Aspekt später zu sprechen kommen.

Entscheidende Bedeutung kommt der Beweistheorie im Zusammenhang mit der konstruktiven Rechtfertigung von relativ starken klassischen Systemen zu. Dabei werden mit sehr aufwendigen und weit entwickelten beweistheoretischen Methoden relativ starke Teilsysteme der klassischen Mengenlehre und Analysis auf entsprechende konstruktive Formalismen reduziert. Da die entsprechenden Reduktionen in einem finiten Metarahmen durchgeführt werden können, ergibt sich daraus eine Grundlegung der klassischen Systeme mit Hilfe von konstruktiven Prinzipien.

Berechenbarkeitstheorie

Der zentrale Begriff, welcher der Berechenbarkeits- oder Rekursionstheorie zugrunde liegt, ist derjenige des *Algorithmus* (Al Chwarismi 825). Unter einem Algorithmus versteht man eine endliche, eindeutige Beschreibung eines effektiven Verfahrens zur Lösung eines Problems. "Effektiv" soll dabei bedeuten, dass das Verfahren auf ausschliesslich mechanische Art und Weise, d.h. im Prinzip von einer Maschine durchgeführt werden kann. Beispiele von Algorithmen sind etwa die Regeln zur Addition und Multiplikation von natürlichen Zahlen, das Verfahren zur Bestimmung der n-ten Primzahl, gegeben eine Zahl n, sowie der berühmte Algorithmus von Euklid zur Berechnung des grössten gemeinsamen Teilers zweier Zahlen.

Die obige Charakterisierung von "Algorithmus" bringt einen *intuitiven Berechenbarkeitsbegriff* zum Ausdruck und beschreibt auf informelle Art und Weise, was ein Computer *im Prinzip*, d.h. ohne Rechenzeit- und Rechenplatzbeschränkung, berechnen kann. Dieser nichtformalisierte Algorithmusbegriff mag zwar genügen, um von bestimmten Problemen zu zeigen, dass sie eine algorithmische Lösung besitzen; er ist jedoch unbrauchbar um nachzuweisen, dass ein gegebenes Problem aus prinzipiellen Gründen *nicht* berechenbar ist, d.h. *keine* algorithmische Lösung besitzen *kann*. Wünschenswert ist deshalb eine streng mathematische Fassung des Algorithmusbegriffs, d.h. die Definition von formal-mathematischen Berechenbarkeitsmodellen. Solche Modelle wurden in den dreissiger Jahren von den mathematischen Logikern Church, Gödel, Herbrand, Kleene, Post und Turing gefunden. Das heute bekannteste Modell, welches vom Engländer Alan Turing (1912-1954) um 1936 entwickelt wurde, ist wohl dasjenige der *Turing-Maschine*.

Bei einer Turing-Maschine handelt es sich nicht um eine konkrete Maschine, sondern um eine bloss hypothetische "Rechenmaschine", welche als formal-mathematisches Modell zu verstehen ist. Sie basiert auf einem zweiseitig unendlichen Band, welches in einzelne Felder unterteilt ist. In jedem dieser Felder kann ein Symbol aus einem endlichen Alphabet stehen, und zu jedem Zeitpunkt wird genau ein Feld von einem Lese-/Schreibkopf abgetastet. Die Maschine besitzt zudem eine endliche Menge von internen Zuständen. Das Verhalten der Turing-Maschine zu einem bestimmten Zeitpunkt ist eindeutig bestimmt durch deren momentanen internen Zustand und das Symbol im eben abgetasteten Feld. Das aktuell gelesene Zeichen wird durch ein anderes Zeichen (eventuell dasselbe!) ersetzt, und der Lese-/Schreibkopf bewegt sich ein Feld nach links oder nach rechts. Am Anfang einer Berechnung wird die Eingabe auf das Band geschrieben. Danach beginnt die Maschine zu "rechnen", und bevor die Rechnung beendet wird (falls sie überhaupt abbricht!), geht die Maschine in einen ausgezeichneten Zustand über, den sogenannten Haltezustand. Das Resultat der Berechnung steht dann auf dem Band.

Die Turing-Maschine liefert nur eine mögliche Formalisierung des intuitiven Algorithmusbegriffes. In den dreissiger Jahren wurde eine ganze Reihe weiterer Berechenbarkeitsmodelle präsentiert, die bis heute von zentraler Bedeutung sind. Dazu gehören der ungetypte Lambdakalkül (Church), die partiell-rekursiven Funktionen (Kleene) sowie der Gödel-Herbrand Gleichungskalkül. Obschon diese Modelle den Algorithmusbegriff auf sehr unterschiedliche Art und Weise charakterisieren, lässt sich zeigen, dass jedes Modell dieselbe Klasse von Funktionen definiert, d.h. die Modelle sind *äquivalent*. Diese Tatsache bildet ein sehr starkes Indiz dafür, dass der intuitive Algorithmusbegriff adäquat formalisert wird, d.h. *jeder* (terminierende) Algorithmus lässt sich mit Hilfe eines und damit aller erwähnten Modelle formalisieren. Dies ist genau die Aussage der sogenannten *Churchschen These*, benannt nach dem Logiker Alonzo Church (1903-1995). Die These lässt sich nicht formal-mathematisch beweisen, da sie

sich auf einen intuitiven Berechenbarkeitsbegriff bezieht; trotzdem zweifelt niemand an ihrer Gültigkeit.

> **Churchsche These (Church 1934)**
> Jedes der genannten formalen Berechenbarkeitsmodelle stimmt mit dem intuitiven Algorithmusbegriff überein.

Mit den verschiedenen formalen Berechenbarkeitsmodellen hat man nun ein Instrument zur Verfügung, das gestattet, von bestimmten Problemen nachzuweisen, dass sie *keine* algorithmische Lösung besitzen oder anders ausgedrückt, dass diese Probleme *unentscheidbar* sind. Die zwei wohl berühmtesten Ergebnisse in dieser Hinsicht sind die *Unentscheidbarkeit des Halteproblems* sowie die *Unentscheidbarkeit der Prädikatenlogik erster Stufe*.

> **Unentscheidbarkeit des Halteproblems (Turing 1936)**
> Es gibt keinen Algorithmus H, der für jeden Algorithmus F und jede Eingabe a entscheidet, ob F auf a terminiert.
>
> **Unentscheidbarkeit der Prädikatenlogik (Church, Turing 1936)**
> Es gibt keinen Algorithmus G, der für jede Formel A der Prädikatenlogik erster Stufe entscheidet, ob A logisch gültig ist.

Es steht ausser Zweifel, dass die erwähnten Arbeiten über die Formalisierung des Algorithmusbegriffes sowie die Grenzen der Berechenbarkeit nicht nur für die (konstruktiven) Grundlagen der Mathematik von Interesse sind, sondern vor allem auch von fundamentaler Bedeutung für die Grundlagen der Informatik; so gehört Berechenbarkeitstheorie auch heute zum Grundstock jedes Informatikstudiums. Im weiteren ist zu betonen, dass vor allem Turings Ideen einen nicht zu unterschätzenden Einfluss auf den Bau von digitalen Computern in den vierziger Jahren ausübten.

Die Berechenbarkeits- und Rekursionstheorie hat sich in den letzten dreissig Jahren rasant weiterentwickelt. Im Vordergrund der Arbeiten standen Fragen nach einer Verallgemeinerung des Berechenbarkeitsbegriffs; die entsprechenden Antworten führten zu zentralen Zusammenhängen mit Teilsystemen der Mengenlehre sowie zu einem noch umfassenderen Verständnis der gewöhnlichen Berechenbarkeitstheorie.

Komplexitätstheorie

Im vorhergehenden Teilkapitel haben wir uns mit der Frage beschäftigt, ob ein gegebenes Problem *im Prinzip* eine algorithmische Lösung besitzen kann, und wir haben berühmte Beispiele von algorithmisch unlösbaren Problemen diskutiert. Im Zentrum des Interesses der Komplexitätstheorie steht nicht die Frage nach der Existenz von Algorithmen, sondern die Analyse von Algorithmen hinsichtlich ihres *Rechenzeit- und Speicherplatzbedarfes*. In vielen Fällen liefert auch hier die Turingmaschine das geeignete Modell. Die Rechenzeit entspricht der Anzahl elementarer Schritte in einer Berechnung und der Rechenplatz der Anzahl Felder, die dabei auf dem Band der Turingmaschine benötigt werden.

Im Bezug auf die praktische Durchführbarkeit und Effizienz von Algorithmen ist die Unterscheidung zwischen *polynomialen* und *exponentiellen* Algorithmen fundamental: Die Rechenzeit eines polynomialen Algorithmus ist beschränkt durch ein Polynom in der Länge

der Eingabe, währenddem die Anzahl der Rechenschritte für einen Exponentialzeitalgorithmus exponentiell in der Länge des Inputs wächst. Es hat sich gezeigt, dass die Identifikation von “effizient” mit “polynomial” und “ineffizient” mit ”exponentiell” in vielen Fällen eine adäquate Unterscheidung bezüglich der Effizienz von Algorithmen liefert.

Für viele Probleme ist die Frage nach der Existenz eines polynomialen Algorithmus offen. Ein berühmtes Problem ist beispielsweise das sogenannte *Travelling-Salesman-Problem*. Ist es bei einer Strassenkarte mit n Städten einem Handelsreisenden bei gegebener Kilometerbeschränkung möglich, bei einer Rundreise jede Stadt einmal zu besuchen? Ein anderes Problem ist das *Bin-Packing-Problem*. Ein Speditionsunternehmen hat n Kästen mit unterschiedlichem Gewicht, und es soll ermittelt werden, ob k Lastwagen ausreichen, um die Kästen zu transportieren, wobei jeder Lastwagen einer Gewichtsbeschränkung s unterliegt. Obschon alle bekannten Algorithmen zur Lösung dieser beiden Probleme exponentiell sind, ist für sie charakteristisch, dass von einer *möglichen Lösung* in polynomialer Zeit verifiziert werden kann, ob die Lösung korrekt ist. Beispielsweise ist es sehr leicht, von einer gegebenen Rundreise zu überprüfen, ob jede Stadt darin vorkommt und die vorgegebene Kilometerbeschränkung eingehalten wird.

Die Menge derjenigen Probleme, welche einen polynomialen Verifikationsalgorithmus besitzen, heisst *NP*; das Kürzel *NP* steht für *nichtdeterministisch-polynomial* und bezieht sich auf eine alternative Definition von polynomialer Verifizierbarkeit mittels sogenannter nichtdeterministischer Turingmaschinen. Die Klasse *NP* umfasst unzählige wichtige Probleme, von denen man nicht weiss, ob sie eine polynomiale Lösung besitzen; viele dieser Probleme sind für die Praxis von Interesse. Die Frage, ob die Klasse P der in polynomialer Zeit entscheidbaren Probleme und die Klasse *NP* der in polynomialer Zeit verifizierbaren Probleme gleich sind, gehört zu den grossen ungelösten Fragen der Informatik und der Mathematik. Obschon vieles darauf hindeutet, dass P und *NP* verschieden sind, ist es bis heute nicht gelungen, einen Beweis für diese These zu finden.

In den letzten zwanzig Jahren sind grosse Anstrengungen unternommen worden, verschiedene Fragestellungen aus dem Bereich der Komplexitätstheorie aus der Sicht der Logik zu betrachten und Methoden und Techniken der Logik zu ihrem besseren Verständnis einzusetzen. Man ist sehr daran interessiert, Charakterisierungen von Komplexitätsklassen (z.B. P und *NP*) zu finden, die in ihrer Definition nicht mehr auf die Turingmaschine oder ähnliche Modelle Bezug nehmen, sondern vielmehr zu einem maschinenunabhängigen, intrinsischen Verständnis dieser Klassen führen; solche Charakterisierungen können beispielsweise von modelltheoretischer oder beweistheoretischer Natur sein. Die entsprechenden Untersuchungen eröffnen nicht nur wichtige neue Einsichten in die Struktur einzelner Komplexitätsklassen, sondern auch von Grunde auf neue (äquivalente) Formulierungen der zahlreichen offenen Probleme in der Komplexitätstheorie, zum Beispiel dem *P-NP*-Problem. Die Aktivitäten im Bereich *Logik und Komplexitätstheorie* gehören heute zu einem etablierten und sehr attraktiven Teilgebiet der theoretischen Informatik.

Weitere Anwendungen der Logik in der Informatik

In den beiden vorhergehenden Abschnitten haben wir die Bedeutung der Berechenbarkeitstheorie für die Grundlagen der Informatik sowie den Einsatz von logischen Methoden in der Komplexitätstheorie erläutert. Im Folgenden will ich eine kurze Übersicht weiterer Anwendungen der Logik in der Informatik skizzieren. Auf ausgewählte Aspekte soll dann später

etwas detaillierter eingegangen werden.

Logische Methoden werden seit langem dazu verwendet, die *Semantik von Programmiersprachen* auf ein formales Fundament zu stellen. Dabei unterscheidet man zwischen der *prozeduralen Semantik* und der *denotationellen Semantik* eines Programms. Während die prozedurale Semantik einer Programmiersprache genau darüber Auskunft gibt, *wie* eine bestimmte Anweisung ausgeführt wird, so untersucht die denotationelle Semantik bloss das "statische" Input-Output-Verhalten eines Programms. Verschiedene Programme können im Sinne einer denotationellen Semantik gleich sein, obschon ihr prozedurales Verhalten verschieden ist. In der Terminologie von Gottlob Frege kann man sagen, dass es sich bei der Unterscheidung von prozeduraler und denotationeller Semantik um diejenige zwischen "Sinn" und "Bedeutung" handelt.

Die Logik hatte zudem einen sehr direkten Einfluss auf die Entwicklung und die Grundlagen von zwei berühmten Programmiersprachen, nämlich von LISP und PROLOG. Die funktionale Sprache LISP wurde in den sechziger Jahren vom Amerikaner John McCarthy am MIT entwickelt. LISP basiert direkt auf dem sogenannten ungetypten Lambda-Kalkül von Church, den wir in einem vorhergehenden Abschnitt im Zusammenhang mit einer möglichen Formalisierung des intuitiven Algorithmusbegriffs bereits erwähnt haben. PROLOG steht für "PROgramming in LOGic" und zählt heute zu der Familie der sogenannten deklarativen Programmiersprachen. Die fundamentale Idee, Logik als Programmiersprache zu verwenden, geht auf Kowalski und Colmerauer zurück, und der erste PROLOG Interpreter wurde 1972 von Roussel in Marseille implementiert. PROLOG basiert auf einem Fragment der Prädikatenlogik erster Stufe, welches eine prozedurale Interpretation zulässt und sich deshalb sehr gut als Programmiersprache eignet. Theorie und Praxis der Logikprogrammierung haben sich sehr schnell zu einem wichtigen Anwendungsgebiet der Logik in der Informatik entwickelt. Methoden der Logik sind nicht nur von Bedeutung für funktionale und deklarative Sprachen, sondern stellen heute auch ein wichtiges Instrument zur Untersuchung der Grundlagen von objektorientierten Sprachen dar.

Die *Spezifikation* und *Verifikation* von Software und Hardware gehören ohne Zweifel zu wichtigen Gebieten der Informatik, die in grossen Masse von logischen Methoden Gebrauch machen. Die Spezifikation einer Softwarekomponente beschreibt auf einer abstrakten Ebene das gewünschte Verhalten der Software. Dabei ist es üblich, Spezifikationen in einer formallogischen Sprache auszudrücken. Dies erleichtert dann die Verifikation, das heisst den formalmathematischen Nachweis, dass ein konkretes Computerprogramm eine bestimmte Spezifikation erfüllt. Dabei ist man daran interessiert, den Prozess der Verifikation zumindest teilweise zu automatisieren, das heisst, ein entsprechender Korrektheitsbeweis soll mit Unterstützung des Computers gefunden werden. In gleichem Masse sind die Spezifikation und Verifikation von Hardware von Bedeutung, wo formal-logische Methoden eingesetzt werden, um beispielsweise die korrekte Arbeitsweise eines Chips zu garantieren. Wir werden im nächsten Abschnitt im Zusammenhang mit der Extraktion von Programmen aus Beweisen auf einen interessanten Aspekt des Themas "Spezifikation und Verifikation" noch einmal zu sprechen kommen.

In der Informatik spielen die verschiedensten Formen von *Wissensrepräsentation* eine zentrale Rolle. Als Beispiele seien Expertensysteme in der Künstlichen Intelligenz oder deklaratives Wissen über (verteilte) Softwaresysteme genannt. Um Wissen automatisch verarbeiten zu können, bedarf es einer formal-logischen Sprache zur Wissensrepräsentation. Das Ableiten von neuem Wissen geschieht dann mit Hilfe von logischen Regeln und Axiomen. Es ist na-

heliegend, den Formalismus der Prädikatenlogik erster Stufe für Wissensrepräsentation und -deduktion zu verwenden. Allerdings hat sich gezeigt, dass in vielen Fällen die klassische Logik für die Wissensrepräsentation von nicht-mathematischem Wissen erhebliche Mängel aufweist. Dies führte zu intensiven Bemühungen, die verschiedensten Formen von *nichtklassischen Logiken* für die Zwecke der Informatik fruchtbar zu machen. Dazu gehören beispielsweise modale und temporale Logiksysteme, welche schon lange zuvor im Bereich der philosophischen Logik intensiv studiert wurden. In einigen der folgenden Teilkapiteln werden wir Fragen der Wissensrepräsentation etwas ausführlicher besprechen.

Beweise als Programme

Wir haben in einem vorhergehenden Abschnitt die Bedeutung von konstruktiven Aspekten von Beweisen für die Grundlagen der Mathematik erläutert und dabei die intuitionistische Logik als einen möglichen Zugang zum Konstruktivismus diskutiert. Eine wesentliche Forderung an konstruktive Existenzbeweise ist die Möglichkeit der Extraktion von expliziten Zeugen, welche die entsprechende Existenzaussage erfüllen. Etwas allgemeiner kann man sagen, dass konstruktive Beweise das "Ablesen" von Algorithmen zulassen. Diese Eigenschaft ist von grossem Interesse im Zusammenhang mit der direkten Extraktion eines Algorithmus aus einem konstruktiven Beweis für seine Spezifikation. Falls etwa $Spec(x, y)$ eine Spezifikation mit Eingabe x und Ausgabe y beschreibt, so kann man oft aus einem *konstruktiven* Beweis π der Aussage, dass es für jeden Input x einen Output y gibt, so dass $Spec(x, y)$ gilt, einen Algorithmus f_π gewinnen, welcher die Spezifikation *Spec* erfüllt:

$$\pi \,:\, \forall x\, \exists y\, Spec(x, y) \quad \longmapsto \quad f_\pi \,:\, \forall x\, Spec(x, f_\pi(x)).$$

Eine besonders einfache und einsichtige Möglichkeit, Beweise direkt als Programme zu interpretieren, wird durch den sogenannten Curry-Howard-Isomorphismus oder die Formulas-as-Types-Interpretation beschrieben. Gemäss dieser Interpretation wird jeder Aussage A ein sogenannter Typ $[A]$ zugeordnet. $[A]$ ist zu verstehen als die Menge der Beweise von A und ist induktiv nach dem Aufbau der Aussage A definiert. Beispielsweise ist im Falle der logischen Implikation ein Element von $[A \to B]$ gegeben als ein Algorithmus, d.h. eine effektive Operation g von $[A]$ nach $[B]$, die einen Beweis π von A in einen Beweis $g(\pi)$ von B transformiert. Die intuitionistische Logik ist korrekt in Bezug auf die Curry-Howard-Interpretation, d.h. der Typ $[A]$ einer intuitionistisch gültigen Aussage A ist nicht leer.

Ist A nun gerade die Aussage $\forall x\, \exists y\, Spec(x, y)$, so liefert ein konstruktiver Beweis π, dass der Typ $[A]$ nicht leer ist, tatsächlich einen Algorithmus f_π, welcher die gegebene Spezifikation erfüllt. Da der Algorithmus f_π aus einem Beweis für seine Spezifikation extrahiert wurde, ist f_π *automatisch* verifiziert, d.h. f_π ist korrekt in Bezug auf seine Spezifikation! Von grossem Interesse ist in diesem Zusammenhang natürlich ein zumindest teilweise computerunterstütztes Finden eines Beweises für eine gegebene Spezifikation. Der Themenbereich "Proofs as programs" bildet heute ein aktives Forschungsgebiet der theoretischen Informatik, und es existieren die verschiedensten Systeme, welche das Experimentieren mit den in diesem Abschnitt diskutierten Ideen ermöglichen.

Wissensverarbeitung in der Prädikatenlogik

Grundsätzlich bietet es sich an, die klassische Prädikatenlogik erster Stufe als allgemeinen Rahmen für Wissensrepräsentation zu verwenden. So basiert beispielsweise ein Expertensy-

stem auf einer Menge von nicht-logischen Fakten, der sogenannten Datenbasis; neues Wissen kann daraus mit Hilfe der Axiome und Regeln der Prädikatenlogik erster Stufe abgeleitet werden. Selbstverständlich ist man im Bereich von Informatikanwendungen sehr daran interessiert, den Ableitungsprozess möglichst weitgehend zu automatisieren. Dabei ist man unmittelbar mit der Unentscheidbarkeit der Prädikatenlogik konfrontiert, d.h. mit der Tatsache, dass es kein algorithmisches Verfahren gibt, welches entscheidet, ob eine bestimmte Aussage aus einer Datenbasis logisch folgt. Andererseits hat die Prädikatenlogik die sogenannte Eigenschaft der *Semi-Entscheidbarkeit*, welche besagt, dass ein Beweis einer Aussage aus der Wissensbasis im Prinzip algorithmisch gefunden werden kann, falls die entsprechende Aussage tatsächlich logisch aus der Datenbasis folgt.

Zweifellos sind im Bereich von automatischen Beweissuchverfahren für die Prädikatenlogik sehr zentrale und zum Teil in mancherlei Hinsicht fruchtbare Ansätze entwickelt worden; allerdings sind solche Verfahren im Allgemeinen nicht effizient und nur in Spezialfällen praktikabel. Eine wesentlich positivere Perspektive ergibt sich, wenn man auf vollautomatische Beweissuchverfahren verzichtet und stattdessen sogenannte *interaktive Beweisverfahren* in den Vordergrund stellt. Bei der interaktiven Suche nach einem Beweis werden gewisse "Routineschritte" im Beweis automatisch ausgeführt, währenddem sich zusätzlich der Benutzer des Beweissystems mit Entscheidungshilfen aktiv in die Suche eines Beweises einschalten kann. Auf diese Weise können oft Probleme der Ineffizienz von vollautomatischen Suchverfahren verringert werden.

Nichtmonotones Schliessen

Beim Reflektieren über einen Problembereich unserer Welt spielen allgemeine Regeln, die in bestimmten Ausnahmesituationen nicht angewendet werden dürfen, eine wesentliche Rolle. Zentral ist dabei die Fähigkeit, einerseits aus unvollständigen Informationen Folgerungen zu ziehen und andererseits bei zusätzlicher Information bereits gemachte Schlüsse wieder zurückzunehmen.

Betrachten wir etwa die Aussage "Vögel fliegen". Es können zwar nicht alle Vögel fliegen, aber wir nehmen an, dass ein Vogel typischerweise fliegen kann. Wenn ich zum Beispiel nur die Information habe, dass Tweety ein Vogel ist, so bin ich in der Regel bereit zu schliessen, dass Tweety fliegen kann. Ein solcher Schluss gründet auf unvollständiger Information: Da ich kein explizites Wissen habe, dass Tweety atypisch bezüglich des Prädikats "Fliegen" ist, nehme ich an, dass Tweety ein typischer Vogel ist und deshalb fliegen kann. Wenn ich nun aber zu einem späteren Zeitpunkt erfahre, dass Tweety ein Pinguin, ein Strauss oder ein kranker Vogel ist, so bin ich in Anbetracht dieser zusätzlichen Information bereit, meinen vorher gemachten Schluss zurückzunehmen. Die eben beschriebene Art und Weise des Schliessens ist *nicht monoton* in dem Sinne, dass ich mit mehr Information weniger folgern kann.

Im Gegensatz dazu ist die Prädikatenlogik ein *monotones* Herleitungssystem. Wenn eine Aussage A aus Axiomen T herleitbar ist und S eine Obermenge von T ist, so kann A trivialerweise auch aus S abgeleitet werden. Trotzdem könnte man versuchen, Regeln mit Ausnahmen in der gewöhnlichen Prädikatenlogik direkt zu repräsentieren, indem man alle Ausnahmen explizit auflistet. Die Liste der Ausnahmen, unter denen beispielsweise ein Vogel nicht fliegen kann, ist jedoch sehr lange, und selbst wenn wir in der Lage wären, alle Ausnahmen zu kennen, könnten wir nicht ableiten, dass Tweety fliegen kann, solange wir nur wissen, dass Tweety ein Vogel ist.

In den letzten zwanzig Jahren sind zahlreiche Ansätze mit dem Ziel der Formalisierung des nichtmonotonen Schliessens vorgeschlagen worden. Viele der neuen nichtmonotonen Logiksysteme gehen auf Forscher im Bereich der Künstlichen Intelligenz zurück. Die entsprechenden Ideen wurden sehr bald von traditionellen Logikern aufgenommen und weiterentwickelt. In neuerer Zeit haben nichtmonotone Logiken auch die Aufmerksamkeit von Philosophen auf sich gezogen, die an möglichen Anwendungen, beispielsweise im Bereich der Wissenschaftstheorie, interessiert sind.

Modale Logiken

Zentraler Gegenstand der Modallogik ist die logische Analyse der Modalitäten "notwendig" und "möglich", welche ihren Ursprung einmal mehr bei Aristoteles hat; weitere Studien zur Modallogik wurden von den Stoikern und später im Mittelalter unternommen. Die moderne Modallogik nahm ihren Anfang um 1918 mit den Arbeiten von Clarence Lewis (1883-1964) zu Fragen im Zusammenhang mit der logischen Implikation. Wichtige Beiträge in dieser ersten Phase der modernen Modallogik gehen zudem auf Rudolf Carnap (1891-1970) und Jan Lukasiewicz (1878-1956) zurück. Eine zweite grosse Welle in der Entwicklung der (modernen) Modallogik nahm ihren Anfang mit den aufsehenerregenden Arbeiten des damaligen Teenagers Saul Kripke, der zu Beginn der sechziger Jahre die sogenannte *Semantik möglicher Welten* der modalen Operatoren "notwendig" und "möglich" entwickelte. Zur selben Zeit haben eine ganze Reihe weiterer Philosophen ähnliche Ideen vorgeschlagen, die Arbeiten von Kripke waren jedoch mit Abstand die einflussreichsten. Seither hat sich die Modallogik zu einem der aktivsten und wichtigsten Gebiete der philosphischen Logik entwickelt. In neuerer Zeit haben modale Ansätze eine grosse Bedeutung im Zusammenhang mit Anwendungen in der Informatik erlangt.

Die Sprache der Modallogik erweitert diejenige der Aussagen- oder Prädikatenlogik um die zwei Operatoren $\Box$ ("notwendig") und $\Diamond$ ("möglich"). Mit jeder Aussage A hat man eine neue Aussage $\Box A$ ("A ist notwendig wahr") sowie $\Diamond A$ ("A ist möglicherweise wahr"). Ein häufiges Axiomenschema der Modallogik ist etwa $\Box A \rightarrow \Diamond A$, welches intuitiv besagt, dass, wenn A notwendig wahr ist, A auch möglicherweise wahr ist. Im Folgenden möchte ich kurz die wesentliche Idee der Semantik möglicher Welten nach Kripke ("Kripke-Semantik") erläutern. Eine Kripke-Interpretation geht aus von einer *Menge von möglichen Welten* W sowie einer *Erreichbarkeitsrelation* $R \subseteq W \times W$. Falls für zwei Welten w und w' in der Menge W die Beziehung $w\ R\ w'$ gilt, so bedeutet dies, dass die Welt w' im Sinne von R von der Welt w aus erreichbar ist. Basierend auf W und R werden dann die modalen Operatoren $\Box$ und $\Diamond$ wie folgt interpretiert:

- $\Box A$ ist wahr in der Welt $w \in W$, falls A wahr ist in *allen* Welten $w' \in W$ mit $w\ R\ w'$;
- $\Diamond A$ ist wahr in der Welt $w \in W$, falls A wahr ist in mindestens einer Welt $w' \in W$ mit $w\ R\ w'$.

In einer Welt w *notwendig wahr* zu sein bedeutet also, in allen von w erreichbaren Welten wahr zu sein, währenddem in w *möglicherweise wahr* zu sein nur heisst, in mindestens einer von w aus erreichbaren Welt wahr zu sein. Man beachte, dass unter dieser Interpretation $\Diamond A$ auch als $\neg\Box\neg A$ abgekürzt werden kann.

Ich will an einem kleinen Beispiel aus der Informatik die Semantik möglicher Welten illustrieren. Dabei soll aufgezeigt werden, wie Modallogik verwendet werden kann, um Eigenschaften

von Programmen auszudrücken. Zu diesem Zweck betrachten wir das Programm *Primzahl drucken*, welches sukzessive die unendliche Sequenz der Primzahlen $2, 3, 5, 7, 11, \ldots$ ausdruckt:

l_0: $y_1 := 2$
l_1: print(y_1)
l_2: $y_1 := y_1 + 1$
l_3: $y_2 := 2$
l_4: if $(y_2)^2 > y_1$ then goto l_1
l_5: if $(y_1 \bmod y_2) = 0$ then goto l_2
l_6: $y_2 := y_2 + 1$
l_7: goto l_4

Ein möglicher Zustand in einer Berechnung des Programms *Primzahl drucken* ist nun ein Tripel (l_i, m, n), wobei l_i mit $0 \leq i \leq 7$ die aktuelle Zeilenmarke angibt, und m beziehungsweise n den aktuellen Inhalt der Variablen y_1 beziehungsweise y_2 festlegen. Die Menge der möglichen Welten entspricht nun der Menge der möglichen Zustände, und ein Zustand (l_j, m', n') ist erreichbar von einem Zustand (l_i, m, n), falls es eine Berechnung im Sinne unseres Programmes, ausgehend von (l_i, m, n), gibt, die zu (l_j, m', n') führt. Beispielsweise ist der Zustand $(l_7, 3, 3)$ erreichbar von $(l_5, 3, 2)$. Wir nehmen weiter an, dass wir zusätzlich ein Prädikat $(at\, l_1)$ zur Verfügung haben, welches genau dann wahr ist, wenn sich der Programmablauf auf Zeile l_1 befindet. Die Aussage

$$\Box(at\, l_1 \rightarrow \mathit{Primzahl}(y_1)),$$

betrachtet im Ausgangszustand $(l_0, 0, 0)$, bringt nun zum Ausdruck, dass, wenn immer der Programmablauf sich auf Zeile l_1 befindet, der Inhalt der Variablen y_1 eine Primzahl ist. Mit anderen Worten beschreibt die obige Aussage einen Teil der *Korrektheit* unseres Programms, nämlich die Tatsache, dass das Programm *nur* Primzahlen druckt. Man kann jetzt die Sprache der Modallogik verwenden, um weitere Korrektheitsaussagen unseres Programms zu spezifizieren, zum Beispiel die Tatsache, dass *jede* Primzahl schliesslich gedruckt wird oder dass die Primzahlen in aufsteigender Reihenfolge ausgegeben werden.

Das Studium von Logiken zur Analyse von Programmen, sogenannten *Programmlogiken*, gehört zu einem gut untersuchten Teilgebiet der theoretischen Informatik. Wichtige Erweiterungen der Modallogik bilden diesbezüglich die *dynamischen Logiken*: Ein typisches Konstrukt der dynamischen Logik hat etwa die Gestalt $\Box_\alpha A$ mit der intendierten Bedeutung "A ist wahr nach jeder Ausführung des Programms α".

Weitere wichtige Logiksysteme, welche die gewöhnliche Modallogik durch "indizierte" Modaloperatoren erweitern, sind die *Wissenslogiken*. Hier bedeutet $\Box_a A$ etwa "Der Agent *a weiss*, dass A". Ein charakteristisches Axiom ist dann zum Beispiel gegeben durch $\Box_a A \rightarrow \Box_a \Box_a A$: wenn der Agent a weiss, dass A, dann weiss a auch, dass er weiss, dass A. Logiken des Wissens haben eine lange und fruchtbare Tradition in der Philosophie. In neuster Zeit haben sie zu äusserst interessanten Anwendungen in der Informatik im Zusammenhang mit der formalen Analyse von verteilten Systemen geführt; dabei ist ein Agent etwa ein Knoten in einem Computersystem. Zentrale Fragestellungen sind zum Beispiel die Klärung der Begriffe des *gemeinsamen Wissens* und des *verteilten Wissens* einer Gruppe von Agenten.

Schliesslich haben Modallogiken zu wichtigen Untersuchungen in Bezug auf die Metamathematik der Zahlentheorie Anlass gegeben. Interpretiert man $\Box A$ als "A ist beweisbar in der

Zahlentheorie", so stellt man fest, dass gültige Eigenschaften des Beweisbarkeitsprädikats der Arithmetik mit bekannten Axiomen der Modallogik korrespondieren. Das Gebiet der *Beweisbarkeitslogiken* bildet heute einen eigenständigen Themenbereich innerhalb der mathematischen Logik.

Temporale Logiken

Die ursprüngliche Motivation für das Studium von logischen Aspekten von "Zeit" ist einerseits philosophischer, andererseits linguistischer Natur. Das traditionelle Paradigma, dass logische Aussagen einen "zeitlosen" und "ewigen" Charakter haben, wird in der temporalen Logik aufgegeben. Für die Linguistik ist ein logisches Verständnis der Tempi offensichtlich von fundamentalem Interesse. In neuerer Zeit spielen temporale Ansätze eine wichtige Rolle in der Informatik im Zusammenhang mit der Spezifikation und Verifikation von verteilten Systemen und parallelen Prozessen.

Die üblichen temporalen Modalitäten sind G, H, F und P, wobei F dual zu G sowie P dual zu H ist. Die intendierte Bedeutung von G und H ist wie folgt gegeben:

$$\begin{array}{lcl} \mathsf{G}A & : & A \text{ ist in der Zukunft immer wahr,} \\ \mathsf{H}A & : & A \text{ ist in der Vergangenheit immer wahr.} \end{array}$$

$\mathsf{F}A$ definiert man nun als $\neg\mathsf{G}(\neg A)$ und $\mathsf{P}A$ als $\neg\mathsf{H}(\neg A)$. Damit erhalten $\mathsf{F}A$ und $\mathsf{P}A$ die intendierte Bedeutung "A ist in der Zukunft manchmal wahr", beziehungsweise "A ist in der Vergangenheit manchmal wahr". Zwei typische Postulate der Temporallogik sind etwa gegeben durch die beiden Schemata

$$A \to \mathsf{H}(\mathsf{F}A), \qquad A \to \mathsf{G}(\mathsf{P}A)$$

Die Semantik von temporalen Logiken ist ebenfalls eine Kripke-Semantik, wobei den möglichen Welten verschiedene Zeitpunkte entsprechen, welche durch eine Erreichbarkeitsrelation miteinander verbunden sein können. Die einzelnen Temporallogiken unterscheiden sich durch verschiedene Zeitmodelle. Zum Beispiel kann Zeit linear, dicht oder verzweigt sowie mit oder ohne Endpunkte sein.

Dies bringt mich zum Schluss meiner Ausführungen. Ich hoffe, dass es mir gelungen ist, einige der wesentlichen Aspekte der Entwicklungen der modernen Logik in Informatik, Mathematik und Philosophie zu illustrieren. Dabei habe ich aufzuzeigen versucht, wie zentral die Einsichten in der Philosophie und im Bereich der Grundlagen der Mathematik für die Anwendungen der Logik in der Informatik sind. Andererseits stellt man heute fest, dass verschiedene Fragestellungen zur Logik in der Informatik ihrerseits einen grossen Einfluss auf neue Arbeiten in der mathematischen und philosophischen Logik ausüben. Ich bin davon überzeugt, dass gerade dieser interdisziplinäre Charakter der Logik immer wieder eine neue Faszination und zugleich eine Herausforderung bedeutet, wodurch die Logik zu einem äusserst attraktiven und fruchtbaren Zweig unserer Wissenschaften wird.