Zur Struktur dessen, was wirklich berechenbar ist

H.-D. Ebbinghaus und M. Grohe*

Erschienen in *Philosophia Naturalis*, 36:91–116, 1999.

Der erste Gödelsche Unvollständigkeitssatz und eine Vielfalt von Unentscheidbarkeitsresultaten[1] zeigen, daß es konkrete Fragestellungen gibt, die sich nicht algorithmisch lösen lassen. Ergebnisse der Komplexitätstheorie[2] legen dar, daß es konkrete Fragestellungen gibt, die sich zwar algorithmisch lösen lassen, für die jedoch jede algorithmische Lösung so aufwendig ist, daß ihrer praktischen Ausführbarkeit enge Grenzen gesetzt sind. Beide Sachverhalte sind vielfältig auch in die erkenntnistheoretische Diskussion eingeflossen.[3] Der Bereich des praktisch Berechenbaren selbst steht jedoch noch außerhalb dieser Diskussion. Um die Struktur dieses Bereichs — zunächst auf mathematische Weise — aufzuhellen, bedarf es eines mathematischen Modells praktischer Berechenbarkeit, das der intuitiven Vorstellung möglichst nahe kommt. Wir wählen dazu, wie es heute üblich ist, das Modell der polynomialen (Schrittzahl-)Komplexität, identifizieren also die praktisch lösbaren Probleme mit den Problemen der Komplexitätsklasse P. Der Ausleuchtung dessen, was praktisch berechenbar ist, entspricht damit die Frage nach der Struktur der Klasse P. Die sog. *Endliche Modelltheorie*[4], ein Gebiet im Grenzbereich von mathematischer Logik, Informatik und Kombinatorik, hat in den letzten Jahren eine Reihe von Resultaten erzielt, die zu einer Erhellung dieser Frage beitragen können. Die Reichhaltigkeit von P wird dabei gemessen durch die "sprachlich-inhaltliche" Komplexität logischer Systeme, welche die Probleme in P beschreiben. Zwar sind zentrale Fragen noch offen; doch die vorliegenden Ergebnisse lassen die folgende Aussage zu:

Für die Klasse der praktisch lösbaren Probleme gibt es nur die zwei einander entgegengesetzten Möglichkeiten:

*Institut für mathematische Logik der Universität Freiburg, Eckerstraße 1, 79104 Freiburg

[1] Vgl. [11, 17].

[2] Vgl. [18].

[3] Literaturhinweise findet man in [9].

[4] Vgl. [10].

(a) Die praktisch berechenbaren Probleme sind im Kern *Varianten eines einzigen Problems*; die Klasse P hat daher eine sehr einfache und übersichtliche Struktur.

(b) Die praktisch berechenbaren Probleme sind von einer *Vielgestaltigkeit, die sich jedem systematischen Zugriff entzieht.*

Alle Möglichkeiten, die zwischen diesen beiden Extremen liegen, lassen sich bereits jetzt ausschließen.

Ziel unserer Ausführungen ist es, diese für die Struktur des uns rechnerisch Zugänglichen sowohl mathematisch als auch erkenntnistheoretisch bedeutsame Dichotomie vorzustellen. Es wird sich zeigen, daß eine Entscheidung darüber, welche der beiden Möglichkeiten (a), (b) wirklich vorliegt, äußerst schwierig sein dürfte.

Neben P sind auch viele andere Komplexitätsklassen untersucht worden.[5] Dabei handelt es sich sowohl um Klassen, die in P enthalten sind, also aus praktisch entscheidbaren Problemen bestehen, als auch um Klassen, die oberhalb von P liegen, bei denen also der erlaubte Aufwand den praktisch möglichen übertreffen kann. Es hat sich herausgestellt, daß für die Klassen oberhalb von P das Analogon von (a) gilt, während für die Teilklassen von P, ähnlich wie für P selbst, die Frage nach der Gültigkeit von (a) oder (b) offen ist. P nimmt daher im Hinblick auf die Frage nach der inneren Struktur, der ja unsere Aufmerksamkeit gilt, eine Sonderstellung ein: Die Klasse des praktischen Berechenbaren könnte nach heutigem Kenntnisstand eine Grenze bilden, an der die innere Struktur von großer Einfachheit zu großer Kompliziertheit umschlägt.

Wir werden versuchen, die Darstellung möglichst wenig mit mathematischen Einzelheiten zu belasten; weitergehende technische Informationen und Literaturhinweise finden sich in den Fußnoten.

Für wertvolle Ratschläge möchten wir Gerhard Vollmer herzlich danken.

1 Berechenbarkeit

Zu Beginn der dreißiger Jahre gelang es, den Begriff des Algorithmus auf überzeugende Weise zu präzisieren. Die Präzisierung fällt nach heutiger Auffassung mit dem naiven Begriff des Algorithmus in der Weise zusammen, daß jeder Algorithmus im präzisen Sinn ein Algorithmus im naiven Sinn ist und

[5] Beispiele folgen im Text.

daß umgekehrt auch zu jedem Algorithmus im naiven Sinn ein Algorithmus im präzisen Sinn existiert, der dasselbe leistet[6] (sog. *Church-Turing-These*).

Algorithmen arbeiten mit "handhabbaren" Objekten, die man in aller Regel auf natürliche Weise durch 0-1-Wörter kodieren kann. Sie dienen verschiedenen Zwecken: Es gibt Algorithmen zur Berechnung von Funktionen (etwa Algorithmen zur schriftlichen Multiplikation zweier in Dualdarstellung gegebener natürlicher Zahlen), Algorithmen zur Entscheidung von Fragestellungen (sog. *Entscheidungsalgorithmen*, etwa Algorithmen, die entscheiden, ob eine vorgegebene Zahl eine Primzahl ist) oder Algorithmen zur Erstellung von Listen (etwa Algorithmen, die alle lösbaren diophantischen Gleichungen in zwei Unbekannten auflisten). Man kann sich leicht klarmachen, daß die Präzisierung des Algorithmenbegriffs geleistet ist, wenn es gelungen ist, den Begriff des Entscheidungsalgorithmus zu präzisieren. Dabei kann man, wie bereits angedeutet, voraussetzen, daß Entscheidungsalgorithmen 0-1-Wörter verarbeiten. Oft identifiziert man die Fragestellung, die ein Entscheidungsalgorithmus entscheidet, mit der Menge L aller Eingabewörter, die zur Antwort "Ja" führen. Eine Menge von Wörtern wie L nennt man eine *formale Sprache*. Entscheidungsalgorithmen entscheiden also formale Sprachen. Die Church-Turing-These ist demnach gleichwertig zu der Aussage, daß die im intuitiven Sinn entscheidbaren formalen Sprachen mit den im präzisen Sinn entscheidbaren formalen Sprachen zusammenfallen.

Für die These sprechen eine über Jahrzehnte gewachsene Erfahrung, aber auch erfolgreiche Versuche einer inhaltlichen Rechtfertigung, die von intuitiven Analysen des Algorithmenbegriffs ausgehen.[7] Ein weiteres Argument beruht darauf, daß die verschiedenen Präzisierungen des Algorithmenbegriffs äquivalent sind, also zur gleichen Klasse entscheidbarer formaler Sprachen führen.

In den sechziger Jahren begannen Bemühungen, auch den Begriff der *praktischen* Berechenbarkeit zu präzisieren. Auf Überlegungen von Cobham [4] und Edmonds [12] geht der Vorschlag zurück, praktische Berechenbarkeit mit polynomialer Berechenbarkeit gleichzusetzen (sog. *Cobham-Edmonds-These*). Dabei heißt eine formale Sprache L *polynomial entscheidbar*, wenn es für sie einen Entscheidungsalgorithmus gibt, bei dem die Zahl der Schritte (die "Zeit"), die bei vorgegebener Eingabe w erforderlich ist, um eine Antwort auf die Frage "Gehört w zu L?" zu erhalten, polynomial von der

[6] Beispiele vereinzelter Kritik findet man in [19] und [22]. Zur Geschichte der Präzisierungsbemühungen vgl. man [5].

[7] Vgl. etwa [23].

Länge von w abhängt.[8]

Die Klasse aller formalen Sprachen, die polynomial entscheidbar sind, wollen wir P ("polynomiale Zeit") nennen. P ist also in mathematisch ausreichender Genauigkeit definiert, sobald wir eine Präzisierung des Algorithmenbegriffs fest vorgeben. Es stellt sich heraus, daß die heute üblichen Präzisierungen des Algorithmenbegriffs nicht nur in bezug auf die prinzipielle Berechenbarkeit gleichwertig sind, sondern alle zur gleichen Klasse P führen und daher auch in bezug auf die praktische Berechenbarkeit äquivalent sind.

Anders als die Church-Turing-These bringt die Cobham-Edmonds-These eine Reihe von Schwierigkeiten mit sich.

Während der Begriff der prinzipiellen Berechenbarkeit — vom Standpunkt der klassischen Mathematik aus — unveränderlich vorgegeben ist, unterliegt die Vorstellung darüber, was praktisch berechenbar ist, der jeweiligen Situation: Praktische Berechenbarkeit beim Landeanflug eines Flugzeugs meint etwas anderes als praktische Berechenbarkeit einer theoretischen mathematischen Konstruktion! Insbesondere läßt sich nicht ausschließen, daß der Begriff der praktischen Berechenbarkeit einem Wandel unterworfen ist, der seine Ursache in technischen Weiterentwicklungen hat. Sollte dies in weitem Maß zutreffen, müßte eine "dauerhafte" Fixierung von vornherein inadäquat sein.

Ein weiterer Einwand bezieht sich darauf, daß die Komplexität eines Algorithmus an seiner ungünstigsten Laufzeit gemessen wird. Bei einigen Algorithmen tritt aber der ungünstige Fall nur selten ein und ist damit für die Praxis nicht unbedingt bedeutsam. In der Tat gibt es für die Praxis wichtige Verfahren (z.B. das Simplexverfahren der linearen Optimierung), die eine exponentielle Zeitkomplexität haben.

Schließlich enthält die Klasse P Probleme, die man gemeinhin wohl nicht mehr als praktisch berechenbar ansehen möchte. Bei der Definition von P unterliegen nämlich die Polynome, die zur Abschätzung der Schrittzahl dienen, keiner Beschränkung hinsichtlich Grad und Größe der Koeffizienten. P enthält damit nachweisbar[9] für beliebig vorgegebene natürliche Zahlen k formale Sprachen, bei denen (für unendlich viele Argumente) die erforderlichen Schrittzahlen in der k-ten Potenz der Inputlänge wachsen.

Trotz dieser Einwände gilt zur Zeit die Komplexitätsklasse P als das beste Modell für die praktische Berechenbarkeit. Vermutlich würde jeder

[8] D.h. $\leq p(\text{Länge von } w)$ für ein von der Eingabe w unabhängiges Polynom p mit reellen Koeffizienten.

[9] Man vergleiche die Zeithierarchietheoreme in [18].

andere Versuch, die Klasse der praktisch berechenbaren Probleme zu beschreiben,[10] auf ähnliche oder sogar noch schwerer wiegende Einwände stoßen. Zum Beispiel würde eine — letztlich willkürliche — Beschränkung der Polynome nach Grad oder Koeffizientengröße die Unabhängigkeit von der Wahl der Präzisierung des Algorithmenbegriffs zerstören und somit unter Umständen bereits durch leichte technische Fortschritte in Frage gestellt.

Damit haben wir die Ausgangsbasis für unsere Fragestellung erreicht: Wir haben den Bereich des praktisch Berechenbaren, den wir erhellen wollen, mit einer gewissen intuitiven Berechtigung als die Komplexitätsklasse P präzisiert.

2 Strukturen

Bislang haben wir Entscheidungsprobleme mit formalen Sprachen aus 0-1-Wörtern identifiziert. Diese Gleichsetzung haben wir damit gerechtfertigt, daß die Einzelfälle eines Entscheidungsproblems "handhabbar" sein müssen, damit man sie algorithmisch verarbeiten kann, und daß aller Erfahrung nach handhabbare Objekte durch 0-1-Wörter kodiert werden können. Doch diese Argumentation, so überzeugend sie auch erscheinen mag, verdeckt einen wesentlichen Punkt: Die Kodierung eines Entscheidungsproblems durch eine formale Sprache erfordert selbst schon einen gewissen Aufwand. Die Komplexität von Problemen, wie wir sie im letzten Abschnitt definiert haben, berücksichtigt hingegen nur den Aufwand, der zur Lösung der bereits kodierten Version erforderlich ist. Nun ist es unter Umständen möglich, durch eine geschickte, aber aufwendige Kodierung die Lösung eines Problems erheblich zu vereinfachen, also einen erheblichen Teil des Lösungsaufwandes durch einen höheren Kodierungsaufwand wettzumachen. Intuitiv steht bei Fragen der praktischen Berechenbarkeit natürlich der Gesamtaufwand zur Diskussion. Wir können deshalb den Kodierungsaufwand nicht unberücksichtigt lassen. Aus diesem Grund werden wir Entscheidungsprobleme nicht mehr durch formale Sprachen, sondern weit unmittelbarer durch Klassen endlicher Strukturen modellieren. Wir werden dann sehen, daß man Klassen von Strukturen computergerecht kodieren kann, ohne ihre Komplexität zu verfälschen. Zugleich gewinnen wir einen methodischen Vorteil: Indem wir Entscheidungsprobleme durch Strukturklassen modellieren, schaffen wir

[10] Etwa die Präzisierung durch die Komplexitätsklasse BPP (=bounded error probabilistic polynomial time) (vgl. [2]).

eine Brücke zur Logik und damit zu semantischen Betrachtungsweisen, die die weitere Diskussion maßgeblich voranbringen werden.

Wir erläutern unser Vorgehen an zwei Beispielen.

Das Verbindungsproblem. Es fragt danach, ob die Städte, die zu einem Verkehrsverbund gehören, in dem Sinn miteinander verbunden sind, daß sich von jeder Stadt des Verbundes jede andere Stadt des Verbundes — nötigenfalls mit Umsteigen — erreichen läßt. Wir modellieren das Problem dadurch, daß wir die Städte des Verbundes zu einer Menge G zusammenfassen, über der wir eine zweistellige Relation E^G definieren, indem wir festsetzen:

> E^G trifft auf a und b aus G (in dieser Reihenfolge) genau dann zu, wenn es eine direkte Verbindung von a nach b gibt.

Das Paar (G, E^G) ist ein *Graph*[11]; er besteht aus einer *Trägermenge* G und der *Interpretation* eines *zweistelligen Relationssymbols* E durch eine zweistellige Relation E^G, die *Kantenrelation* des Graphen.

Wir schreiben fortan $E^G ab$, um anzudeuten, daß E^G auf a und b zutrifft.

Ein Graph $\mathcal{G} = (G, E^G)$ heißt *zusammenhängend*, wenn je zwei verschiedene Punkte a, b von G in der Relation E stehen oder über eine Kantenfolge

$$E^G a a_1, E^G a_1 a_2, \ldots, E^G a_{n-1} a_n, E^G a_n b$$

verbindbar sind. Der Verbundenheit der Städte eines Verkehrsverbundes entspricht somit der Zusammenhang des zugehörigen Graphen, und dem Verbindungsproblem entspricht damit die Klasse $\mathbb{G}_Z$ der zusammenhängenden Graphen.

Das Turnierproblem. Es fragt danach, ob es bei einem Turnier, in dem jeder Teilnehmer mit jedem anderen um Sieg oder Niederlage kämpft, einen nach der Zahl der gewonnenen Kämpfe eindeutigen Sieger gibt. Wir modellieren dieses Problem durch besondere Graphen, sog. *Turniergraphen.* Bei ihnen gilt für je zwei verschiedene Punkte a und b der Trägermenge G entweder $E^G ab$ ("a schlägt b") oder $E^G ba$ ("b schlägt a"). Turniergraphen mit einem eindeutig bestimmten Sieger mögen *Sieggraphen* heißen. Dem Turnierproblem entspricht damit die Klasse $\mathbb{G}_S$ der Sieggraphen.

Ein Graph $\mathcal{G} = (G, E^G)$ ist eine *Struktur.* Zu jeder Struktur gehört eine *Trägermenge* — hier G — und eine Menge V von Relationssymbolen, ihr *Vokabular* — hier $\{E\}$[12]; jedes Relationssymbol aus V besitzt eine Stellenzahl

[11] Genauer: ein *gerichteter* Graph.

[12] $\{E\}$ bezeichnet die Menge, die genau aus dem Element E besteht.

und wird durch eine entsprechend stellige Relation über der Trägermenge *interpretiert* — hier E durch E^G.[13] *Geordnete* Graphen haben die Gestalt $(G, E^G, <^G)$. Dabei ist $<$ ein zweistelliges Relationssymbol und $<^G$ eine Ordnungsrelation über G, d.h. eine zweistellige Relation, die die Elemente von G anordnet. Geordnete Graphen haben das Vokabular $\{E, <\}$. Sie sind Beispiele von Strukturen, die eine Ordnung tragen, von *geordneten* Strukturen.

Unter einer Strukturklasse verstehen wir im folgenden eine Klasse endlicher Strukturen des gleichen Vokabulars.[14] Wir haben also das Verbindungsproblem und das Turnierproblem jeweils als eine Strukturklasse modelliert. Statt z.B. zu fragen, ob ein gewisses Turnier einen eindeutig bestimmten Sieger hat, fragen wir jetzt gleichwertig, ob der zugehörige Turniergraph zur Strukturklasse $\mathbb{G}_S$ der Sieggraphen gehört.

Wenn wir Graphen $\mathcal{G} = (G, E^G)$ mit Hilfe eines Algorithmus daraufhin prüfen möchten, ob sie zur Klasse $\mathbb{G}_S$ gehören, müssen wir sie "algorithmengerecht" eingeben. Auf naheliegende Weise kann dies in Form einer Tabelle T geschehen. Dabei entsprechen die Zeilen und Spalten von T den Elementen des Graphen; das Feld, das zur Zeile des Elements a und zur Spalte des Elements b gehört, trägt eine Eins, wenn $E^G ab$ gilt, und eine Null, wenn $E^G ab$ nicht gilt. Ist z.B. $G = \{1, 2, 3\}$ und gilt $E^G ab$ genau für $(a, b) = (1, 2), (2, 1), (1, 3)$, so sieht T, falls wir die natürliche Reihenfolge der Elemente unterstellen, folgendermaßen aus:

$$\begin{array}{c} 0\ 1\ 1 \\ 1\ 0\ 0 \\ 0\ 0\ 0 \end{array}$$

T birgt alle strukturellen Eigenschaften von $\mathcal{G}$ [15] und ist in diesem Sinne eine adäquate Kodierung von $\mathcal{G}$.[16]

[13] Strukturen, wie wir sie gerade definiert haben, heißen genauer *relationale* Strukturen. Strukturen im üblichen Sinn können auch noch *Verknüpfungen* und *ausgezeichnete Elemente* enthalten.

[14] Wir verlangen zusätzlich, daß eine Klasse mit einer Struktur auch jede dazu isomorphe Struktur enthält.

[15] Graphen mit gleichen Tabellen sind isomorph.

[16] Wenn man nun noch die Zeilen von T nebeneinandersetzt, gewinnt man schließlich ein 0-1-*Wort*, das den Graphen $\mathcal{G}$ kodiert. Der Klasse $\mathbb{G}_S$ der Sieggraphen entspricht dann eine Menge von Kodierungen, also eine Menge von 0-1-Wörtern, d.h. eine formale Sprache. Die Betrachtungen, die wir hier für Graphen, also für Strukturen des Vokabulars $\{E\}$, durchführen, lassen sich leicht auf Strukturen eines beliebigen endlichen Vokabulars ausdehnen.

Wir kommen nun zu der eingangs angedeuteten Kodierungsproblematik. Ein Graph $\mathcal{G}$ besitzt i.a. viele Tafeln $T_1, T_2, \ldots$, die von verschiedenen Anordnungen seiner Elemente herrühren. Solange wir nur an irgend einem Entscheidungsalgorithmus für $\mathbb{G}_S$ interessiert sind, spielt es keine Rolle, mit welcher der jeweils möglichen Tafeln dieser Algorithmus arbeitet. Wenn wir jedoch an ein praktisches (also polynomial zeitbeschränktes) Entscheidungsverfahren denken, ändert sich dies grundlegend. Der Zeitaufwand möglicher Algorithmen kann nämlich stark von der jeweils gewählten Tafel, d.h. von der jeweils gewählten Anordnung der Elemente, abhängen. Wählen wir z.B. die Anordnung so, daß (im Sinne der Turniergraphen) Spieler mit mehr Siegen vor Spielern mit weniger Siegen stehen, können wir ein schnelles Entscheidungsverfahren finden, wir brauchen ja nur anhand der ersten beiden Zeilen der jeweiligen Tabelle festzustellen, ob der erste Teilnehmer mehr Siege erfochten hat als der zweite. Legt man für die Tabellen andere Anordnungen zugrunde, etwa die Startnummern der Spieler, ihr Lebensalter oder ihre Körpergröße, Anordnungen also, die mit dem Ausgang eines Turniers nur wenig zu tun haben müssen, kann der Entscheidungsaufwand höher ausfallen. Vielleicht könnte sogar — wenn wir die Anordnungen sehr "ungeschickt" wählen — die Existenz eines polynomial zeitbeschränkten Algorithmus ausgeschlossen sein.[17]

Wir berühren damit den entscheidenden Punkt: Bei der Frage der *prinzipiellen* Entscheidbarkeit spielt es keine Rolle, welche Tafeln wir zugrunde legen; wir können ja einen hohen Aufwand für die Herstellung geeigneter Tafeln in das Verfahren integrieren. Wie man zeigen kann, gilt dies gleichermaßen für alle hinreichend hohen Komplexitätsschranken und insbesondere für alle in der Literatur behandelten wichtigen Komplexitätsklassen *oberhalb* von P.[18] Polynomial entscheidbare Problemklassen dagegen könnten ihre polynomiale Entscheidbarkeit dem Umstand verdanken, daß die Tafeln geschickt gewählt sind, daß also Entscheidungsaufwand gespart wird auf Kosten des Aufwandes für die Herstellung geeigneter Tafeln. Diese Problematik und die daraus entspringenden Schwierigkeiten, die den Inhalt unserer

[17] Dieser Fall kann bei $\mathbb{G}_S$ nicht eintreten und auch nicht bei der Klasse $\mathbb{G}_Z$ der zusammenhängenden Graphen. Wählt man statt $\mathbb{G}_S$ oder $\mathbb{G}_Z$ die Klasse $\mathbb{G}_H$ der *hamiltonschen* Graphen (also der Graphen, deren Punkte man entlang gerichteter Kanten so durchwandern kann, daß jeder Punkt genau einmal aufgesucht wird), so gibt es zwar Anordnungen, die polynomial zeitbeschränkte Entscheidungsalgorithmen zulassen; doch ist es ein offenes Problem, ob das für alle Anordnungen gilt.

[18] So für die Klasse NP der nicht-deterministisch polynomial zeitbeschränkten Probleme, für die Klasse PSPACE der mit polynomial beschränktem Speicherbedarf entscheidbaren Probleme oder für die Klasse E der mit exponentiellem Zeitaufwand (der Gestalt $2^{p(\text{Länge der Eingabe})}$) entscheidbaren Probleme.

Ausführungen bilden, sind somit eine Besonderheit von P.[19]

Die Beschränkung auf formale Sprachen, so naheliegend sie bei unbefangener Betrachtung auch erscheinen mag, kann also im Bereich des praktisch Berechenbaren zu wesentlichen Verzerrungen führen. Zugleich zeigen unsere Beispiele, daß sich viele Entscheidungsprobleme unmittelbarer und adäquater durch Strukturklassen modellieren lassen. Beides ist für uns Grund genug, *Entscheidungsprobleme fortan nicht mehr durch formale Sprachen, sondern durch Strukturklassen wiederzugeben.*

Die neue Auffassung schließt formale Sprachen nicht aus; denn sie lassen sich auf recht natürliche Weise als Strukturklassen modellieren. Auch hierzu ein Beispiel.

Wir legen das Alphabet $A = \{0, 1\}$ zugrunde. Die formale Sprache L bestehe aus den Palindromen über A, also den symmetrischen 0-1-Wörtern wie $0, 0110, 010010$. Jedem 0-1-Wort w weisen wir nun eine Struktur $\mathcal{A}^w$ eines festen Vokabulars zu, wobei sich Wörter und zugeordnete Strukturen eindeutig entsprechen. Dann können wir L durch die Strukturklasse $\mathbb{K}(L)$ [20] modellieren, die aus den Strukturen $\mathcal{A}^w$ mit $w \in L$ besteht. Als Vokabular wählen wir $\{<, P_0, P_1\}$ mit einstelligen Relationssymbolen P_0 und P_1. Ist z.B. w das Wort 0110, so bestehe die Trägermenge A^w von $\mathcal{A}^w$ aus den Zahlen 1,2,3,4 (entsprechend der Zahl der Buchstaben von w), $<^w$ sei die natürliche Kleiner-Relation auf A^w; P_0^w treffe dann auf die Zahlen aus A^w zu, die solchen Stellen entsprechen, an denen im Wort w der Buchstabe 0 steht, also auf die Zahlen 1 und 4, und P_1^w treffe in ähnlicher Weise auf die Zahlen 2 und 3 zu.[21]

Durch den Übergang von formalen Sprachen zu Strukturklassen gewinnen wir in vielen Fällen nicht nur eine größere Natürlichkeit bei der Modellierung, es erschließt sich, wie wir bereits eingangs angedeutet haben, eine neue Dimension: Für die Beschreibung von Strukturen stehen uns in den Sprachen der mathematischen Logik weittragende Instrumente zur Verfügung. Wir werden damit in den Stand versetzt, semantische Gesichtspunkte in unsere komplexitätstheoretisch orientierte Fragestellung einzubringen. Unter-

[19] Und von Teilklassen wie der Komplexitätsklasse L der mit logarithmisch beschränktem Speicherbedarf entscheidbaren Probleme.

[20] Genauer: durch ihren isomorphen Abschluß.

[21] Kodiert man jetzt die Struktur $\mathcal{A}^w$ wieder durch Tafeln, so bietet sich dazu auf natürliche Weise eine Ordnung an, nämlich die Ordnung $<^w$, die ja bereits zur Struktur gehört. Die Kodierungsproblematik stellt sich hier also gar nicht — sicher in Übereinstimmung mit dem, was wir erwarten sollten.

suchungen dieser Art fallen in das noch recht junge Gebiet der *deskriptiven Komplexitätstheorie*[22].

Zunächst jedoch bleibt das Problem, wie sich Strukturen, letztlich also abstrakte Gebilde, rechnerisch handhaben lassen, und damit die Frage, wie sich Strukturen als Eingaben von Rechnern kodieren lassen, ohne den Berechnungsaufwand zu verfälschen.[23] Wir behalten dazu die bisherige Kodierung durch Tafeln (bzw. durch Wörter, die die Tafeln wiedergeben) bei. Die Abhängigkeit von den jeweils verwendeten Ordnungen beseitigen wir dadurch, daß wir *alle* Ordnungen berücksichtigen.

Genauer sieht unser Vorgehen folgendermaßen aus: Gegeben sei eine Strukturklasse $\mathbb{K}$, z.B. die Klasse $\mathbb{G}_S$ der Sieggraphen. Intuitiv bestimmt $\mathbb{K}$ ein Entscheidungsproblem, das danach fragt, ob eine Struktur des Vokabulars von $\mathbb{K}$ zu $\mathbb{K}$ gehört oder nicht. Es sei $L(\mathbb{K})$ die formale Sprache, die aus *allen* (linear notierten) Tafeln der Strukturen aus $\mathbb{K}$ besteht, d.h. aus den Tafeln für *alle* möglichen Anordnungen. Im Falle $\mathbb{K} = \mathbb{G}_S$ gehört z.B. der Graph $\mathcal{G} = (\{1,2,3\}, \{(1,2),(1,3),(3,2)\})$ [24] zu $\mathbb{G}_S$. $L(\mathbb{G}_S)$ enthält also für $\mathcal{G}$ (in linearer Notation) die folgenden Tafeln:

0 1 1
0 0 0 für die Anordnung 1,2,3;
0 1 0

0 1 1
0 0 1 für die Anordnung 1,3,2;
0 0 0

0 0 0
1 0 1 für die Anordnung 2,1,3;
1 0 0

0 0 0
1 0 0 für die Anordnung 2,3,1;
1 1 0

[22] Vgl. [10].

[23] Denkbar ist natürlich auch die Konzeption von Algorithmen und Rechnern, die direkt mit Strukturen arbeiten, ohne auf eine Anordnung ihrer Elemente zurückzugreifen; vgl. etwa [1].

[24] Es gelte also $E^G 12, E^G 13$ und $E^G 32$.

0 0 1
1 0 1 für die Anordnung 3,1,2;
0 0 0

0 1 0
0 0 0 für die Anordnung 3,2,1.
1 1 0

Dann besage die praktische Entscheidbarkeit von $\mathbb{K}$, daß $L(\mathbb{K})$ zu P gehört. Intuitiv gesehen präzisieren wir dadurch gerade, daß die praktische Entscheidbarkeit nicht von einer mehr oder minder geschickt gewählten Anordung der Elemente abhängen sollte. Wir können zusätzlich ins Feld führen, daß diese Definition der praktischen Entscheidbarkeit die Definition für formale Sprachen umfaßt. Ist nämlich L eine formale Sprache und $\mathbb{K}$ die ihr entsprechende Strukturklasse, so stellt man leicht fest, daß L genau dann zu P gehört (also im alten Sinn praktisch entscheidbar ist), wenn $L(\mathbb{K})$ zu P gehört (wenn also $\mathbb{K}$ im neuen Sinn praktisch entscheidbar ist). Es ist daher nur konsequent und vereinfacht die Sprechweisen, wenn wir P fortan nicht mehr als die Klasse der praktisch entscheidbaren formalen Sprachen definieren, sondern als *die Klasse der praktisch entscheidbaren Strukturklassen.* Unter P verstehen wir also fortan die Menge der praktisch entscheidbaren *Strukturklassen.*

3 Zur inneren Struktur von P

Bevor wir uns der inneren Struktur von P genauer zuwenden, wollen wir einige Möglichkeiten auflisten, die wir erwarten können.

(1) P besteht aus Strukturklassen, die in einem präzisierbaren Sinn Varianten einer einzigen Strukturklasse $\mathbb{K}_0$ sind. $\mathbb{K}_0$ ist dann die Essenz dessen, was wir praktisch berechnen können. Der Bereich des praktisch Berechenbaren hat eine äußerst durchsichtige Struktur.

(2) P enthält mehrere — vielleicht sogar unendlich viele — Strukturklassen, die keine Varianten voneinander sind, aber es ist möglich, sich auf systematische Weise einen Überblick über P zu verschaffen. Mit anderen Worten: Es ist möglich, zu berechnen, was praktisch berechenbar ist.

(3) P hat eine so unübersichtliche Struktur, daß es grundsätzlich nicht möglich ist, sich einen systematischen Überblick zu verschaffen.

Eine Klärung der Frage, welche dieser Möglichkeiten zutrifft, verlangt zunächst nach einer Präzisierung der Vorstellungen, die in ihre Beschreibung eingehen.

Zuerst müssen wir uns fragen, was es bedeutet, daß eine Strukturklasse eine *Variante* einer anderen ist. Eine Präzisierung dieses Begriffes und damit der Aussage (1) gehört zum Kern unserer Betrachtungen. Wir bereiten sie im nächsten Abschnitt vor und diskutieren sie dann im letzten Teil.

Wenden wir uns nun der Alternative (2) zu. Welche Möglichkeiten versuchen wir, hier zu erfassen? Es könnte zum Beispiel sein, daß alle Strukturklassen in P entweder Varianten der Klasse $\mathbb{G}_S$ der Siegraphen oder Varianten der Klasse $\mathbb{G}_Z$ der zusammenhängenden Graphen sind, daß aber $\mathbb{G}_S$ und $\mathbb{G}_Z$ keine Varianten voneinander sind. Damit tritt (1) nicht ein. Dennoch hätten wir in diesem Fall einen systematischen Überblick über P — alle Probleme sind Varianten von $\mathbb{G}_S$ oder von $\mathbb{G}_Z$. Wir wollen aber auch ganz andere Möglichkeiten berücksichtigen, die Klasse P systematisch zu erfassen. Zum Beispiel könnte es sein, daß P gerade aus allen Strukturklassen besteht, die durch ein Programm der Programmiersprache PASCAL erkannt werden können, welches keine Schleifenanweisung wie WHILE oder REPEAT...UNTIL verwendet. Weil wir sicherlich alle Programme der Programmiersprache PASCAL, die dieser Bedingung genügen, systematisch auflisten können, würde uns eine solche Charakterisierung von P ebenfalls die Möglichkeit geben, P systematisch zu erfassen, und damit unter (2) fallen.

Natürlich sind noch viele andere Verfahren denkbar. Eine naheliegende Möglichkeit, sie alle zu erfassen, besteht in der folgenden Präzisierung von (2): Es gibt einen Algorithmus, mit dessen Hilfe die Strukturklassen in P in der Form $\mathbb{K}_0, \mathbb{K}_1, \mathbb{K}_2, \ldots$ aufgelistet werden können. Mit anderen Worten: es gibt eine auf der Menge der natürlichen Zahlen definierte *berechenbare* Funktion σ, deren Werte $\sigma(0), \sigma(1), \ldots$ gerade die Strukturklassen in P sind.

Hier entsteht ein Problem: Berechenbare Funktionen arbeiten mit Wörtern. Während wir die natürlichen Zahlen, z.B. in Dualdarstellung, als Wörter vorgeben können, liegt nicht unmittelbar auf der Hand, wie wir so abstrakte Gebilde wie Strukturklassen in einem vernünftigen Sinn durch Wörter kodieren können.[25] Beschränken wir uns jedoch auf Strukturklassen in P — und das reicht ja! —, dann können wir so vorgehen: Wir beschreiben eine Klasse $\mathbb{K}$ in P durch die folgenden beiden Daten:

[25] Da es mehr Strukturklassen (nämlich überabzählbar viele) als z.B. 0-1-Wörter gibt, kann man ohnehin nicht alle Strukturklassen kodieren.

(i) ihr Vokabular, etwa als Folge der Relationssymbole und deren Stellenzahl;

(ii) einen Algorithmus im Sinne einer Präzisierung des Berechenbarkeitsbegriffs, der sie in polynomialer Zeit entscheidet.[26]

Beide Daten lassen sich leicht durch 0-1-Wörter kodieren und die einzelnen Kodierungen dann zu einem einzigen 0-1-Code $C(\mathbb{K})$ zusammenfassen. Mit $C(\mathbb{K})$ haben wir $\mathbb{K}$ "voll im Griff": Wir können aus dem ersten Teil von $C(\mathbb{K})$ das Vokabular von $\mathbb{K}$ entnehmen[27] und aus dem zweiten Teil einen Algorithmus polynomialer Zeitkomplexität, mit dessen Hilfe wir über die Zugehörigkeit von Strukturen zu $\mathbb{K}$ entscheiden können.

Ähnlich können wir bei anderen bekannten Klassen wie NP oder E vorgehen.

Wir nennen jetzt eine Komplexitätsklasse C wie P, NP oder E *auflistbar*, wenn es eine auf der Menge der natürlichen Zahlen definierte berechenbare Funktion gibt[28], deren Werte $f(0), f(1), f(2), \ldots$ Kodierungen der Gestalt $C(\mathbb{K}_0)$, $C(\mathbb{K}_1)$, $C(\mathbb{K}_2), \ldots$ sind, wobei C gerade aus den Klassen $\mathbb{K}_0, \mathbb{K}_1, \mathbb{K}_2, \ldots$ besteht. Mit anderen Worten: C ist genau dann auflistbar, wenn wir C in dem Sinne systematisch überblicken können, daß sich die (Kodierungen der) Klassen in C mit Hilfe einer berechenbaren Funktion — als Werte von 0, 1, usf. — aufzählen lassen. Wir präzisieren dann (2) als Auflistbarkeit von P.

Schließlich präzisieren wir (3) als Nichtauflistbarkeit von P.

Der Begriff der Auflistbarkeit ist sehr formal und sehr schwach. So ist ja unsere Definition rein algorithmischer Natur, und an die Berechnungskomplexität einer die Klasse P aufzählenden berechenbaren Funktion werden keine Anforderungen gestellt. Insbesondere lassen wir zu, daß die aufzählende Funktion nicht *praktisch* berechenbar ist. P könnte also auflistbar sein, ohne daß wir daraus praktisch bedeutsame Folgerungen ziehen könnten. Außerdem spiegelt sich die Struktur von P lediglich auf abstrakte Weise in dem Algorithmus, der die aufzählende Funktion berechnet.

Man beachte jedoch, daß die Schwäche von (2) zur Stärke von (3) wird: Nichtauflistbarkeit von P bedeutet, daß es keinen Weg gibt, die Probleme

[26] Der Algorithmus kann z.B. als Turingprogramm T gegeben sein, zusammen mit einer natürlichen Zahl k mit der Eigenschaft, daß die Laufzeit von T durch das Polynom n^k beschränkt ist.

[27] Bis auf die genaue Gestalt der Symbole, die ja unwesentlich ist.

[28] Berechenbar etwa mit Hilfe eines Turingprogramms.

aus P systematisch "in den Griff" zu bekommen, unabhängig davon, wie abstrakt und aufwendig ein solcher Weg auch gewählt sein sollte.

4 Logiken und Quantoren

Im vorangehenden Abschnitt haben wir die Möglichkeit der Beschreibung von P auf berechenbarkeits- und komplexitätstheoretisch orientierte Gesichtspunkte abgestellt. Wie schon dort angekündigt, wollen wir uns jetzt stärker inhaltlich orientieren: Wir wollen die Struktur von P an der Struktur einer P beschreibenden Logik messen. Zunächst stellen wir kurz die dazu benötigten logischen Begriffe vor.

Das neben der Aussagenlogik einfachste, zugleich aber auch wichtigste logische System ist das System $\mathcal{L}_1$ der Logik der ersten Stufe. Seinen Sprachen liegt, ähnlich wie den Strukturen, jeweils ein *Vokabular* zugrunde, also eine endliche Menge von Relationssymbolen. Besteht das Vokabular nur aus dem zweistelligen Relationssymbol E, haben die einfachsten *Ausdrücke*, die sog. *atomaren* Ausdrücke, der zugehörigen Sprache der ersten Stufe die Gestalt

$$Exy \text{ und } x = y,$$

wobei $x, y, \ldots$ Variablen für Elemente von Strukturen sind und Exy gelesen wir als "E trifft zu auf x, y" und $x = y$ als "x ist gleich y". Die weiteren Ausdrücke erhält man durch die wiederholte Bildung von *Negationen* (mit Hilfe von $\neg$), *Konjunktionen* (mit $\wedge$), *Disjunktionen* (mit $\vee$), *Implikationen* (mit $\rightarrow$), sowie durch Quantifizierungen der Gestalt "Für alle x" (mit $\forall x$) und "Es gibt ein x" (mit $\exists x$). Dabei werden zusätzlich Klammern benutzt, um die eindeutige Lesbarkeit zu sichern.

Wir verwenden $\varphi, \psi, \ldots$, um Ausdrücke der ersten Stufe zu notieren. Ausdrücke, in denen alle Variablen durch die Quantoren $\forall, \exists$ gebunden sind, heißen *Sätze* der ersten Stufe. So ist

$$\forall x \neg Exx$$

ein Satz der ersten Stufe zum Vokabular $\{E\}$. Er besagt, daß E irreflexiv ist. (Einen Graphen $\mathcal{G} = (G, E^G)$, in dem φ gilt, bezeichnet man als *schleifenfrei*.) Der Satz

$$\varphi = \forall x \forall y (\neg x = y \rightarrow (Exy \vee \exists z (Exz \wedge Ezy)))$$

gilt in $\mathcal{G}$ genau dann, wenn je zwei verschiedene Punkte unmittelbar oder über einen dritten verbunden sind. Ist $\mathcal{G}$ ein Graph, der einen Verkehrsver-

bund beschreibt, so gilt φ in $\mathcal{G}$ genau dann, wenn man von jeder Stadt aus jede andere Stadt mit höchstens einmaligem Umsteigen erreichen kann.

Die Klasse aller *endlichen* Strukturen vom Vokabular $\{E\}$, in denen ein Satz φ vom Vokabular {E} gilt, bezeichnen wir mit

$$\mathrm{Mod}(\varphi).\ ^{29}$$

$\mathrm{Mod}(\forall x \neg Exx)$ ist also die Klasse der schleifenfreien endlichen Graphen.

Wir nennen eine Klasse $\mathbb{K}$ von Strukturen *in der ersten Stufe axiomatisierbar*, wenn es einen Satz φ der ersten Stufe vom gleichen Vokabular wie $\mathbb{K}$ gibt mit

$$\mathbb{K} = \mathrm{Mod}(\varphi).$$

Entsprechend heiße, für ein beliebiges logisches System $\mathcal{L}$ [30], $\mathbb{K}$ *in* $\mathcal{L}$ *axiomatisierbar*, falls es einen $\mathcal{L}$-Satz φ gibt mit

$$\mathbb{K} = \mathrm{Mod}_{\mathcal{L}}(\varphi).$$

Dabei sei $\mathrm{Mod}_{\mathcal{L}}(\varphi)$ die Klasse der endlichen Strukturen, in denen φ im Sinne der Semantik von $\mathcal{L}$ gilt.

Wir können jetzt die in der Einleitung zu diesem Abschnitt angedeutete Beschreibbarkeit einer Komplexitätsklasse C durch eine Logik $\mathcal{L}$ präzisieren: *$\mathcal{L}$ beschreibe* C *genau dann, wenn die in $\mathcal{L}$ axiomatisierbaren Strukturklassen mit den Strukturklassen in* C *zusammenfallen.*

Nehmen wir an, die Logik $\mathcal{L}$ beschreibe P. Dann können wir die Reichhaltigkeit von P dadurch zu ergründen suchen, daß wir die Gesamtheit der in $\mathcal{L}$ axiomatisierbaren Strukturklassen betrachten. Obwohl diese Gesamtheit mit P identisch ist, könnte der Wechsel der Perspektive, nämlich die Hinwendung zur Sichtweise der Logik, neue Erkenntnisse bringen. Nehmen wir z.B. an, schon die Logik $\mathcal{L}_1$ der ersten Stufe beschreibe P. Da wir die Sätze der ersten Stufe (zu beliebigen Vokabularen) systematisch auflisten können und da man zu jedem Satz φ der ersten Stufe effektiv einen polynomial zeitbeschränkten Algorithmus angeben kann, der $\mathrm{Mod}(\varphi)$ entscheidet, wäre dann P auflistbar.

Beschreibt $\mathcal{L}_1$ die Klasse P? Leider nicht. Zwar liegt, wie wir soeben angedeutet haben, jede in der ersten Stufe axiomatisierbare Strukturklasse in P; die Umkehrung gilt jedoch nicht. So liegen die Klasse $\mathbb{G}_Z$ der zusammenhängenden Graphen und die Klasse $\mathbb{G}_S$ der Sieggraphen in P, sie sind

[29] "Mod" erinnert an "Modell". In der Literatur steht $\mathrm{Mod}(\varphi)$ gewöhnlich für die Klasse *aller* (also auch der *unendlichen*) Strukturen, in denen φ gilt.

[30] Eine Definition logischer Systeme findet man in [8].

aber nicht in der ersten Stufe axiomatisierbar. Dasselbe gilt für die Klasse der Mengen (d.h. der Strukturen zum leeren Vokabular) mit einer geraden Zahl von Elementen.[31]

Gibt es überhaupt eine Logik, die P beschreibt? Natürlich hängt eine Beantwortung dieser Frage davon ab, wie der Begriff der Logik definiert wird. Verwenden wir die übliche Definition[32], lautet die Antwort "Ja"; doch die P beschreibenden Logiken, die man kennt, hängen unmittelbar mit der Definition von P zusammen[33] und liefern keine neuen Informationen. Statt uns also in nichtssagender Allgemeinheit zu verlieren, greifen wir im folgenden auf einen besonderen Typ von Logiken zurück, auf sog. *Lindströmlogiken*. Sie entstehen aus der Logik $\mathcal{L}_1$ der ersten Stufe durch die Adjunktion neuer *Quantoren*. Wir stellen uns Quantoren als eine sprachtechnische Möglichkeit vor, Konzepte zu formulieren, wie etwa das Konzept "geradzahlige Menge" oder das Konzept "zusammenhängender Graph".[34] Wir können dann die Struktur von P dadurch zu erfassen suchen, daß wir die Struktur der Gesamtheit der Konzepte untersuchen, die wir in Form von Quantoren an $\mathcal{L}_1$ adjungieren müssen, um eine Logik zu erhalten, die P beschreibt. Zunächst jedoch bedarf es einer Klärung der benötigten Begriffe.

Wir schildern die Adjunktion eines Quantors an $\mathcal{L}_1$ am Beispiel der Klasse $\mathbb{G}_Z$ der zusammenhängenden Graphen. Zuvor eine Vereinbarung: Ist $\varphi(x, y)$ ein Ausdruck, in dem höchstens die Variablen x, y nicht durch Quantoren gebunden sind, also als Parameter auftreten, so definiert $\varphi(x, y)$ über jeder Struktur $\mathcal{A}$, deren Vokabular das Vokabular von $\varphi(x, y)$ umfaßt, eine zweistellige Relation $\varphi(x, y)^{\mathcal{A}}$; sie trifft auf zwei Elemente a, b der Struktur $\mathcal{A}$ genau dann zu, wenn $\varphi(a, b)$ in $\mathcal{A}$ gilt. Entsprechend definieren Ausdrücke mit nur einem oder mit mehr als zwei Parametern Teilmengen oder höherstellige Relationen.[35]

Zur Klasse $\mathbb{G}_Z$ der zusammenhängenden Graphen zurückkehrend, wählen wir ein Quantorensymbol $Q_{\mathbb{G}_Z}$ und erweitern die Möglichkeiten der Bil-

[31] Ein Satz der ersten Stufe, der kein Relationssymbol außer dem Gleichheitszeichen enthält und n Quantoren hat, kann nicht zwischen Mengen mit n und Mengen mit $n + 1$ Elementen unterscheiden.

[32] Vgl. [8].

[33] Vgl. [10].

[34] Einer in der Mathematik üblichen Vorgehensweise folgend, identifizieren wir Konzepte mit Strukturklassen, also das Konzept "zusammenhängender Graph" mit der Klasse $\mathbb{G}_Z$ der (endlichen) zusammenhängenden Graphen.

[35] Kommen in den definierenden Ausdrücken weitere Parameter vor, so wird in einer Struktur erst dann eine Relation definiert, wenn diese Parameter durch Elemente der Struktur belegt werden. Wir schließen solche Möglichkeiten im folgenden stillschweigend ein.

dung von Ausdrücken in $\mathcal{L}_1$ um die folgende Regel:

> Sind $\varphi(x)$ und $\psi(y,z)$ Ausdrücke, so ist auch $Q_{\mathbb{G}_Z}xyz[\varphi(x),\psi(y,z)]$ ein Ausdruck.

Um die Bedeutung von $Q_{\mathbb{G}_Z}xyz[\varphi(x),\psi(y,z)]$ in einer (vom Vokabular her passenden) Struktur $\mathcal{A}$ mit der Trägermenge A festzulegen, beachten wir zunächst, daß $\varphi(x)$ eine Teilmenge G von A bestimmt; G besteht aus den Elementen von A, für die $\varphi(x)^{\mathcal{A}}$ gilt. Ferner definiert $\psi(y,z)$ über A die zweistellige Relation $\psi(y,z)^{\mathcal{A}}$. Deren Einschränkung auf Elemente von G bezeichnen wir mit E^G. Damit definieren die Ausdrücke $\varphi(x)$ und $\psi(y,z)$ über $\mathcal{A}$ die Struktur (G,E^G) vom Vokakular $\{E\}$. Wir vereinbaren jetzt:

> $Q_{\mathbb{G}_Z}xyz[\varphi(x),\psi(y,z)]$ gelte in $\mathcal{A}$ genau dann, wenn (G,E^G) zu $\mathbb{G}_Z$ gehört, also ein zusammenhängender Graph ist.

Ein Beispiel: Es sei φ der Satz

$$Q_{\mathbb{G}_Z}xyz[x=x,Eyz].$$

Wir betrachten eine Struktur $\mathcal{A}=(A,E^A)$ vom Vokabular $\{E\}$. Die durch $x=x^{\mathcal{A}}$ bestimmte Teilmenge G von A enthält alle Elemente von A, d.h. es ist $G=A$, und die durch $Eyz^{\mathcal{A}}$ bestimmte Relation über $G=A$ stimmt mit E^A überein, also ist $E^G=E^A$. Damit definieren $x=x$ und Eyz über $\mathcal{A}$ die Struktur $\mathcal{A}$ selbst, und wir erhalten:

$$\varphi \text{ gilt in } \mathcal{A} \text{ genau dann, wenn } \mathcal{A}\in\mathbb{G}_Z,$$

d.h. φ axiomatisiert die Klasse $\mathbb{G}_Z$.

Wir bezeichnen die Logik, die wir durch die gerade geschilderte Erweiterung von $\mathcal{L}_1$ erhalten haben, mit $\mathcal{L}_1(Q_{\mathbb{G}_Z})$. Unser Beispiel besagt dann: $\mathbb{G}_Z$ ist in $\mathcal{L}_1(Q_{\mathbb{G}_Z})$ axiomatisierbar, genauer:

$$\mathbb{G}_Z = \mathrm{Mod}_{\mathcal{L}_1(Q_{\mathbb{G}_Z})}\big(Q_{\mathbb{G}_Z}xyz[x=x,Eyz]\big).$$

Wie man zeigen kann, ist $\mathcal{L}_1(Q_{\mathbb{G}_Z})$ die kleinste "vernünftige" Logik[36], in der $\mathbb{G}_Z$ axiomatisierbar ist. Unser Ziel, die Logik der ersten Stufe auf natürliche Weise um Ausdrucksmöglichkeiten zu erweitern, die es gestatten, die Klasse $\mathbb{G}_Z$ der zusammenhängenden Graphen zu axiomatisieren, ist damit erreicht.

[36] Nämlich *reguläre* Logik.

Ein weiteres Beispiel: Es sei $\mathbb{M}_G$ die Klasse der Mengen mit einer geraden Zahl von Elementen. Wir adjungieren entsprechend einen Quantor $Q_{\mathbb{M}_G}$ an $\mathcal{L}_1$, der die Bildung von Ausdrücken der Gestalt

$$Q_{\mathbb{M}_G}\, x[\varphi(x)]$$

erlaubt; sie gelten in einer Struktur $\mathcal{A}$ genau dann, wenn die durch $\varphi(x)^{\mathcal{A}}$ bestimmte Menge eine gerade Zahl von Elementen hat. So gilt

$$Q_{\mathbb{M}_G}\, x[\forall y(x = y \vee \neg(Exy \vee Eyx))]$$

in einem Graphen genau dann, wenn die Zahl seiner isolierten Punkte (also der Punkte, aus denen keine Kante hinaus- und in die keine Kante hineingeht) gerade ist.

Natürlich können wir $Q_{\mathbb{G}_Z}$ und $Q_{\mathbb{M}_G}$ auch gleichzeitig an $\mathcal{L}_1$ adjungieren. Wir gelangen damit zu der Logik $\mathcal{L}_1(Q_{\mathbb{G}_Z}, Q_{\mathbb{M}_G})$, in der sowohl $\mathbb{G}_Z$ als auch $\mathbb{M}_G$ axiomatisierbar sind. Ist allgemein $\mathcal{Q}$ eine Menge von Quantoren der gerade beschriebenen Art, so bezeichnen wir mit $\mathcal{L}_1(\mathcal{Q})$ die Logik, die aus $\mathcal{L}_1$ durch die simultane Adjunktion aller Quantoren aus $\mathcal{Q}$ entsteht. Man nennt die Logiken der Gestalt $\mathcal{L}_1(\mathcal{Q})$ häufig *Lindströmlogiken* und die Quantoren des oben genannten Typs *Lindströmquantoren*.[37]

Damit haben wir das logische Rüstzeug bereitgestellt, das wir im folgenden benötigen.

5 Zur Reichhaltigkeit von P

Den Schlüssel für unser weiteres Vorgehen liefert die folgende Beobachtung: Es sei $\mathcal{Q}_{\mathsf{P}}$ die Menge der Quantoren, die den Klassen in P entsprechen. Dann gilt:

(∗) $\mathcal{L}_1(\mathcal{Q}_{\mathsf{P}})$ beschreibt P.[38]

So unauffällig diese Feststellung auf den ersten Blick erscheinen mag, sie gibt uns den entscheidenden Hinweis, die Reichhaltigkeit von P zu messen, nämlich an der Reichhaltigkeit, die eine Quantorenmenge $\mathcal{Q}$ haben muß, damit gilt:

(∗∗) $\mathcal{L}_1(\mathcal{Q})$ beschreibt P.

[37] Lindströmquantoren und Lindströmlogiken gehen auf [20] zurück; vgl. auch [21].

[38] Der Beweis dafür, daß die in $\mathcal{L}_1$ axiomatisierbaren Strukturklassen zu P gehören, läßt sich leicht auf $\mathcal{L}_1(\mathcal{Q}_{\mathsf{P}})$ ausdehnen. Umgekehrt ist jede Strukturklasse $\mathbb{K}$ aus P bereits in $\mathcal{L}_1(Q_{\mathbb{K}})$ axiomatisierbar und daher erst recht in $\mathcal{L}_1(\mathcal{Q}_{\mathsf{P}})$.

Wir nennen eine Menge $\mathcal{Q}$ von Lindströmquantoren *auflistbar*, wenn die zugehörigen Strukturklassen auflistbar sind und die Zuordnung der Strukturklassen zu den Quantorensymbolen effektiv ist, etwa in dem Sinn, daß die Kodierungen der Strukturklassen als Quantorensymbole dienen. Dann läßt sich aus den Fakten, die wir erwähnt haben, leicht der folgende Sachverhalt beweisen:

> P ist genau dann auflistbar, wenn es eine auflistbare Menge $\mathcal{Q}$ von Lindströmquantoren gibt, so daß $\mathcal{L}_1(\mathcal{Q})$ die Klasse P beschreibt.

Damit haben wir eine logische Form der Alternative (2) aus Abschnitt 3 gewonnen.

Im Hinblick auf eine Präzisierung der Alternative (1) fragen wir jetzt: Wie einfach kann man $\mathcal{Q}$ wählen, ohne die Gültigkeit von $(**)$ zu verlieren? Oder schärfer: Gibt es gar eine einzige Strukturklasse $\mathbb{K}_0$ mit der Eigenschaft, daß $\mathcal{L}_1(\mathcal{Q}_{\mathbb{K}_0})$ P beschreibt? In diesem Fall wäre (1) realisiert und zugleich präzisiert; die Varianten von $\mathbb{K}_0$, die P ausmachen, würden aus den Strukturklassen bestehen, die man, ausgehend von $\mathbb{K}_0$, mit Mitteln der ersten Stufe axiomatisieren kann; sie würden sich also in übersichtlicher Weise gewinnen lassen. Doch die Antwort ist negativ: Nach [16] kann $(**)$ für kein endliches $\mathcal{Q}$ gelten.

Es gibt sehr einfache unendliche Quantorenmengen. Man erhält sie aus einem einzigen Lindströmquantor Q dadurch, daß man, intuitiv gesprochen, seine höherdimensionalen Varianten einbezieht, die sog. *Vektorisierungen* von Q. Wir erläutern diesen Begriff für die Dimension 2 am Beispiel des Quantors $Q_{\mathbb{G}_Z}$, der zur Klasse $\mathbb{G}_Z$ der zusammenhängenden Graphen gehört.

Sei $Q^2_{\mathbb{G}_Z}$ ein (gegenüber $Q_{\mathbb{G}_Z}$ neues) Quantorensymbol. Es erlaube die Bildung von Ausdrücken der Gestalt

$$Q^2_{\mathbb{G}_Z}\, x_1 x_2 y_1 y_2 z_1 z_2 [\varphi(x_1, x_2), \psi(y_1, y_2, z_1, z_2)].$$

Sie gelten in einer Struktur $\mathcal{A}$ passenden Vokabulars[39] genau dann, wenn mit

$$G := \varphi(x_1, x_2)^{\mathcal{A}} \text{ und } E^G := \text{ die Beschränkung von } \psi(y_1, y_2, z_1, z_2)^{\mathcal{A}} \text{ auf } G$$

die Struktur (G, E^G) zu $\mathbb{G}_Z$ gehört, also ein zusammenhängender Graph

[39] Bei Belegung etwaiger Parameter mit Elementen aus A.

ist.[40] Wir nennen $Q^2_{\mathbb{G}_Z}$ die 2-*Vektorisierung* von $Q_{\mathbb{G}_Z}$. Entsprechend definiert man allgemein für $n \geq 1$ die n-*Vektorisierung* $Q^n_{\mathbb{G}_Z}$ von $Q_{\mathbb{G}_Z}$. Insbesondere ist $Q^1_{\mathbb{G}_Z} = Q_{\mathbb{G}_Z}$.

Für einen Quantor $Q_{\mathbb{K}}$ sei $\mathcal{Q}_{\mathbb{K}}$ die Menge der Quantoren $Q^1_{\mathbb{K}}, Q^2_{\mathbb{K}}, Q^3_{\mathbb{K}}, \ldots$ Sie ist unendlich, und dennoch birgt sie im wesentlichen nur ein Konzept, nämlich $\mathbb{K}$.

Ein für uns entscheidendes Resultat von Dawar [6] besagt in dieser Terminologie:

> Wenn P auflistbar ist, so gibt es eine Strukturklasse $\mathbb{K}_0$ mit der Eigenschaft, daß $\mathcal{L}_1(\mathcal{Q}_{\mathbb{K}_0})$ die Klasse P beschreibt. $\mathbb{K}_0$ kann als Klasse von Graphen gewählt werden und liegt (natürlich) in P.

Sollte also P auflistbar sein, d.h. systematisch erfaßbar in dem von uns als sehr schwach kritisierten Sinn, so gibt es ein graphentheoretisches Konzept $\mathbb{K}_0$, das praktisch entscheidbar ist und in dem Sinn den Kern des praktisch Berechenbaren bildet, daß alle praktisch entscheidbaren Probleme Varianten von $\mathbb{K}_0$ sind, die aus $\mathbb{K}_0$[41] mit Prozessen erster Stufe[42] sowie durch Vektorisierungen hervorgehen. Es gilt dann also eine präzise Form von (1).[43] Damit können wir zusammenfassen:

Dichotomiesatz. Es gilt entweder (a) oder (b), wobei

(a) Es gibt eine Strukturklasse $\mathbb{K}_0$ mit der Eigenschaft, daß $\mathcal{L}_1(\mathcal{Q}_{\mathbb{K}_0})$ die Klasse P beschreibt.

(b) P ist nicht auflistbar. Insbesondere gibt es keine auflistbare Quantorenmenge $\mathcal{Q}$, für die $\mathcal{L}_1(\mathcal{Q})$ die Klasse P beschreibt.

Wir haben also die in der Einleitung angekündigte Alternative erreicht. Es ist heute völlig offen, ob (a) gilt oder ob (b) gilt. Wahrscheinlich ist

[40] Man beachte, daß G keine Teilmenge von A ist, sondern eine zweistellige Relation über A, d.h. eine Menge von geordneten Paaren von Elementen von A; die vierstellige Relation $\psi(y_1, y_2, z_1, z_2)^{\mathcal{A}}$ über A wird aufgefaßt als eine zweistellige Relation zwischen geordneten Paaren von Elementen von A; E^G ist also eine zweistellige Relation über G.

[41] Und den durch atomare Sätze der Gestalt $Rc_1 \ldots c_n$ bzw. $c_1 = c_2$ mit Konstanten $c_1, c_2, \ldots$ definierten Klassen.

[42] Komplementbildungen entsprechend der Negation, Vereinigungen entsprechend der Disjunktion und Projektionen entsprechend der Quantifierung mit dem Existenzquantor. (In der ersten Stufe sind $\wedge, \rightarrow, \forall$ mit $\neg, \vee, \exists$ definierbar!)

[43] Die Existenz eines solchen "universellen" Konzepts erinnert an die vollständigen Probleme der Komplexitätstheorie; in der Tat besteht hier eine weitgehende methodische Parallelität; vgl. [10].

es auch nicht einfach, eine Entscheidung zu fällen. So würde nämlich ein Beweis von (b) zugleich zeigen, daß NP, die nichtdeterministische Version von P, von P verschieden ist, und damit das immer noch als unangreifbar eingeschätzte "P = NP?"-Problem der Komplexitätstheorie lösen.[44] (Ein Beweis von (a) dagegen bräuchte dieses Problem gar nicht zu berühren.)

Beschränkt man sich auf Klassen *geordneter* Strukturen, so gilt (a).[45] Sicherlich ist dies ein wichtiger Sonderfall, erfaßt er doch alle formalen Sprachen. Wir hoffen aber, klargemacht zu haben, daß die Verallgemeinerung auf beliebige Strukturklassen, also auf Klassen von nicht notwendig geordneten Strukturen, intuitiv geboten ist und daß die von uns befolgte Präzisierung der polynomialen Entscheidbarkeit ein Weg ist, den Entscheidungsaufwand sauber von einem möglichen Kodierungsaufwand zu trennen. Für die bekannten Komplexitätsklassen, denen höhere Komplexitätsschranken zugrunde liegen, besteht, wie sich herausstellt, die Kodierungsproblematik nicht mehr. Wie wir bereits in der Einleitung festgestellt haben, ist auch die Alternative des Dichotomiesatzes entschieden: Es gilt (a). Insofern erfaßt die noch ungelöste Problematik des Dichotomiesatzes eine Besonderheit von P.

Hat sich die Mühe, die wir haben aufwenden müssen, um den Dichotomiesatz zu formulieren und die in ihn eingehenden Begriffsbildungen vorzustellen und zu motivieren, gelohnt? Die Antwort hängt von der Information ab, die wir über den Bereich des praktisch Berechenbaren erhalten haben. Diese Information ist, so meinen wir, beträchtlich. Zunächst scheint ja der Unterschied zwischen polynomialer und hyperpolynomialer Komplexität lediglich eine Frage des Aufwandes zu sein. So beruhen die Schwierigkeiten der Kodierung, auf die wir für das praktisch Berechenbare gestoßen sind, letztlich auf der Tatsache, daß die Anzahl der möglichen Ordnungen einer Struktur exponentiell mit deren Mächtigkeit wächst.[46] Wie eine Analyse der Beweise zeigt, die in den Dichotomiesatz eingehen, entspricht die Alternative (a) der Möglichkeit, entsprechende Teile des Hyperpolynomialen noch praktisch zu beherrschen. Dann gewinnt also der Bereich des praktisch Berechenbaren die Übersichtlichkeit, die wir von höheren Komplexitätsklassen kennen. Andernfalls, und das ist die wesentliche Aussage des Dichotomie-

[44] NP ist nämlich auflistbar, weil es nach [13] durch die sog. existentielle Logik der zweiten Stufe beschrieben wird; vgl. [10].

[45] Als Klasse $\mathbb{K}_0$ kann man nach [14] eine geeignete Version des Konzepts "Iteration" wählen.

[46] Genauer: die Anzahl der modulo Isomorphie verschiedenen geordneten Versionen einer Struktur.

satzes, bleibt uns der Bereich des praktisch Berechenbaren seiner Struktur nach in seiner Ganzheit grundsätzlich verschlossen; insbesondere existiert dann kein Weg, aus bekannten praktisch berechenbaren Problemen durch systematische Variation alle praktisch berechenbaren Probleme zu erhalten.[47] Es gilt dann also so etwas wie ein Unvollständigkeitssatz der praktischen Berechenbarkeit. Der Schritt vom Polynomialen zum Exponentiellen[48] ist dann nicht mehr ein nur quantitativer, sondern ein tiefgreifend struktureller, vergleichbar dem Schritt vom Endlichen (mit den es begleitenden Schwierigkeiten der endlichen Kombinatorik) zum Unendlichen (mit der Eleganz transfiniter Methoden).

Welche der Alternativen erwarten wir? Die Meinungen sind geteilt, die Frage selbst war in jüngerer Vergangenheit Gegenstand intensiver Diskussionen. Vielleicht zeichnet sich zur Zeit eine Mehrheit für die Alternative (b), also für die Unübersichtlichkeit von P, ab. Jedoch sind, wie uns scheint, zur Zeit weder die Anhänger von (b) noch die Anhänger von (a) in der Lage, überzeugende Argumente für ihre Ansicht vorzubringen.

Auf die weitere Entwicklung der Datenverarbeitung hat eine Klärung der Dichotomiefrage wohl keine unmittelbaren konkreten Auswirkungen. Es würde also in diesem Fall so ähnlich sein wie bei den Gödelschen Unvollständigkeitssätzen. Auch sie haben — abgesehen von technischen Anwendungen bei konkreten Fragestellungen — keine greifbaren Auswirkungen auf die Mathematik gehabt. Doch sie haben das mathematische Bewußtsein verändert und insbesondere die Vorstellungen von der Tragweite mathematischer Methoden korrigiert. Eine ähnliche Wirkung könnte auch die Klärung der Dichotomiefrage entfalten, jetzt in bezug auf das, was wir rechnerich zu beherrschen vermögen.[49] Das Dichotomieproblem ist daher mehr als nur eine reizvolles mathematisches Problem; es hinterfragt die Natur des uns rechnerisch Zugänglichen unter einem wesentlichen Aspekt, würde uns doch eine Lösung eine grundsätzliche Erkenntnis über die Welt vermitteln, in die wir durch die rasch wachsenden Möglichkeiten der Datenverarbeitung gestellt werden.

[47] Denn sonst wäre P auflistbar!

[48] Genauer: Von P zu den hyperpolynomialen Komplexitätsklassen wie NP, PSPACE, E.

[49] Vielleicht vermag das gar schon die Dichotomie selbst mit ihrer weit auseinanderklaffenden Alternative, sollte sie nur genügend weit bekannt werden.

Literatur

[1] S. Abiteboul and V. Vianu. Generic computation and its complexity. In *Proceedings of the 23rd ACM Symposium on Theory of Computing*, pages 209–219, 1991.

[2] J.L. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* Springer Verlag, 2nd edition, 1995.

[3] A. Chandra and D. Harel. Structure and complexity of relational queries. *Journal of Computer and System Sciences*, 25:99–128, 1982.

[4] A. Cobham. The intrinsic computational difficulty of functions. In *Proceedings of the 2nd International Congress for Logic, Methodology, and Philosophy of Science*, pages 24–30. North-Holland Publishing Company, 1965.

[5] J.N. Crossley. Reminiscences of logicians. In J.N. Crossley, editor, *Algebra and Logic*, Lecture Notes in Mathematics, pages 1–62. Springer-Verlag, 1975.

[6] A. Dawar. Generalized quantifiers and logical reducibilities. *Journal of Logic and Computation*, 5:213–226, 1995.

[7] A. Dawar and L. Hella. The expressive power of finitely many generalized quantifiers. In *Proceedings of the 9th IEEE Symposium on Logic in Computer Science*, 1994.

[8] H.-D. Ebbinghaus. Extended logics: The general framework. In J. Barwise and S. Feferman, editors, *Model–Theoretic Logics*, pages 25–76. Springer-Verlag, 1985.

[9] H.-D. Ebbinghaus. Maschinen und Kreativität: Metamathematische Argumente für das menschliche Denken. *Philosophia naturalis*, 29:1–30, 1992.

[10] H.-D. Ebbinghaus and J. Flum. *Finite Model Theory.* Springer-Verlag, 1995.

[11] H.-D. Ebbinghaus, J. Flum, and W. Thomas. *Einführung in die Mathematische Logik.* Spektrum Akademischer Verlag, 2nd edition, 1996.

[12] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[13] R. Fagin. Generalized first–order spectra and polynomial–time recognizable sets. In R. M. Karp, editor, *Complexity of Computation, SIAM-AMS Proceedings, Vol. 7*, pages 43–73, 1974.

[14] M. Grohe. Complete problems for fixed–point logics. *Journal of Symbolic Logic*, 60:517–527, 1995.

[15] Y. Gurevich. Logic and the challenge of computer science. In E. Börger, editor, *Current trends in theoretical computer science*, pages 1–57. Computer Science Press, 1988.

[16] L. Hella. Logical hierarchies in PTIME. In *Proceedings of the 7th IEEE Symposium on Logic in Computer Science*, pages 360–368, 1992.

[17] H. Hermes. *Aufzählbarkeit, Entscheidbarkeit, Berechenbarkeit.* Springer--Verlag, 1961.

[18] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley Publishing Company, 1979.

[19] L. Kalmar. An argument against the plausibility of church's thesis. In A. Heyting, editor, *Constructivity in Mathematics*, pages 72–80. North-Holland Publishing Company, 1959.

[20] P. Lindström. First-order predicate logic with generalized quantifiers. *Theoria*, 32:186–195, 1966.

[21] P. Lindström. Prologue. In *Quantifiers: Logics, Models and Computation*, pages 12–36. Kluwer Academic Publishers, 1995.

[22] R. Péter. Rekursivität und Konstruktivität. In A. Heyting, editor, *Constructivity in Mathematics*, pages 226–233. North-Holland Publishing Company, 1959.

[23] A. Turing. On computable numbers, with an application to the "Entscheidungsproblem". *Proceedings of the London Mathematical Society, Series 2*, 42:230–265, 1936.