Komplexitätsanalysen in der Künstlichen Intelligenz*

Bernhard Nebel
Fakultät für Informatik
Universität Ulm
D-89069 Ulm
e-mail: nebel@informatik.uni-ulm.de

November 1994

Zusammenfassung

Die Analyse der Berechenbarkeitskomplexität von typischen KI-Problemen sowie der Entwurf effizienter Verfahren zum Lösen dieser Probleme hat in den letzten Jahren verstärkt Interesse gefunden. In diesem Beitrag wollen wir auf den Sinn von Komplexitätsanalysen eingehen, exemplarisch das Vorgehen bei solchen Analysen darstellen und einige Methoden zum Umgang mit dem Komplexitätsproblem skizzieren.

*Erscheint in *Künstliche Intelligenz*.

1 Einleitung

Die Analyse der Berechenbarkeitskomplexität von KI-Problemen sowie der Entwurf effizienter Verfahren zur Lösung von KI-Problemen hat in den letzten Jahren verstärkt Interesse gefunden, dokumentiert durch zahlreiche Beiträge auf den großen KI-Konferenzen. Seit kurzem werden auch spezielle Workshops zu diesem Thema veranstaltet, wie z.B. der Workshop zum Thema "Tractable Reasoning" während der AAAI'92, das AAAI Spring Symposium '93 zum Thema "AI and NP-hard Problems" und der Workshop "Algorithms, Complexity, and Commonsense Reasoning" während der ECAI'94.

Die in der KI bearbeiteten Probleme, wie z.B. Sprach- und Bildverstehen, Planen, Konfiguration und Diagnose, sind bekanntermaßen alle sehr schwierig. Deshalb stellt sich die Frage, welchen Erkenntnisgewinn solche Komplexitätsanalysen bringen können, und ob es überhaupt sinnvoll ist, sich Gedanken darüber zu machen, wie schwierig ein KI-Problem vom komplexitätstheoretischen Standpunkt her ist. Ist nicht jedes KI-Problem trivialerweise zumindest NP-schwierig oder sogar unentscheidbar? Und ist es nicht gerade ein wesentliches Merkmal der KI, diese Probleme trotzdem zu lösen?

Wenn man KI als Wissenschaftsgebiet auffaßt, das die Analyse und Nachbildung menschlicher kognitiver Prozesse als *Berechnungsprozesse* zum Gegenstand hat, dann bietet es sich an, die in der Informatik entwickelte Theoriebildung auch für dieses Gebiet anzuwenden. Wenn man kognitive Prozesse als Berechnungsprozesse interpretiert, dann unterliegen auch kognitive Prozesse den gleichen Beschränkungen, wie sie für Berechnungsprozesse gelten. Insbesondere kann die Komplexitätstheorie uns *Hinweise* geben, welche Art von Problemen nicht zufriedenstellend gelöst werden können. Dies gilt auch, wenn wir einen mehr technischen Standpunkt vertreten und nicht an der Nachbildung von kognitiven Prozessen interessiert sind, sondern an der Nachbildung der kognitiven *Fähigkeiten* mit Hilfe von Berechnungsprozessen.

Abgesehen von diesen mehr grundsätzlichen Argumenten kann die Analyse der Komplexität von KI-Problemen uns helfen, ein besseres Verständnis des Problems zu gewinnen. Neben einer Analyse der logischen Eigenschaften eines Problems, wie sie heute von einem großen Teil der KI als notwendig für das Verständnis eines Problems angesehen wird (Genesereth & Nilsson 1987; Bibel *et al.* 1993), bietet eine Komplexitätsanalyse Einsichten in die „algorithmische" Natur des Problems und unterstützt uns dabei, effiziente Verfahren zur Lösung des Problems (unter Umständen unter Einschränkungen der Allgemeinheit oder der Qualität der Lösung) zu finden.

In einem bekannten KI-Lehrbuch (Rich & Knight 1991, S. 44) wird KI folgendermaßen definiert:

> ...the study of techniques for solving exponentially hard problems in

polynomial time by exploiting knowledge about the problem domain.

Ich halte diese Aussage zum einen für zu pessimistisch, da sie impliziert, daß alle KI-Probleme exponentielle Laufzeit erfordern, zum anderen für zu optimistisch, da sie suggeriert, daß wir Probleme, die exponentielle Laufzeit erfordern, durch welchen „Trick" auch immer, in polynomialer Zeit lösen können – was per Definition unmöglich ist.

Tatsächlich meinen die Autoren des Buches wohl auch nicht, was sie schreiben, da sie diese Aussage später in ihrem Buch relativieren. Der Begriff dessen, was als „Lösung" im Sinne der KI gilt, ist signifikant verschieden vom Lösungsbegriff, wie er im Rest der Informatik verwendet wird. In der KI geht es weniger darum, Lösungsverfahren zu finden, die in allen Fällen die korrekte Antwort finden, sondern man ist auch zufrieden mit Verfahren, die in allen „interessanten Fällen" eine „halbwegs korrekte" Antwort finden. Mit anderen Worten, es wird nicht das ursprüngliche Problem sondern ein abgeschwächtes Problem gelöst (Bylander 1991).

Doch bevor man den Anspruch auf korrekte Lösungen aufgibt, sollte man untersuchen, ob das Problem wirklich schwierig ist und welche speziellen Aspekte des Problems die Ursache für die Schwierigkeit sind. Dabei ist die Tatsache, daß es exponentiell viele potentielle Lösungen gibt (Enzinger *et al.* 1994, S. 78), zwar notwendig aber nicht hinreichend für die Schwierigkeit eines Problems. Es gibt beispielsweise exponentiell viele Pfade in einem gerichteten, azyklischen Graphen. Das Finden eines kürzesten Pfades ist jedoch effizient realisierbar.[1]

Im allgemeinen scheinen Probleme wie Sprach- und Bildverstehen von Menschen mühelos in Echtzeit gelöst zu werden, was darauf hindeutet, daß diese Probleme einen nicht zu hohen Berechnungsaufwand erfordern. Auf der anderen Seite existieren Probleme, die auch Menschen nicht mühelos lösen können, wie z.B. die Diagnose eines technischen Geräts, Brettspiele oder das Lösen von Kreuzworträtseln (Levesque 1988). Wie stellen wir aber nun fest, ob ein Problem effizient lösbar ist oder erheblichen Zeitaufwand für seine Lösung erfordert?

Die Komplexitätstheorie gibt uns hier die geeigneten Werkzeuge an die Hand, die Schwierigkeit eines Problems zu beurteilen. Können wir zeigen, daß ein Problem in polynomialer Zeit lösbar ist, hat es in den meisten Fällen wenig Sinn, sich über approximative Lösungen Gedanken zu machen. Kann man umgekehrt zeigen, daß ein Problem NP-schwierig ist, so wissen wir (mit an Sicherheit grenzender Wahrscheinlichkeit), daß das Problem in seiner Allgemeinheit nicht in angemessener Zeit gelöst werden kann. Der Beweis der NP-Schwierigkeit eines KI-Problems (oder auch eines anderen beliebigen algorithmischen Problems) bedeutet jedoch normalerweise nicht das Ende der Untersuchung. Solch ein Resultat

[1] Das Finden eines *längsten* Pfades scheint jedoch nicht effizient realisierbar zu sein.

initiiert in den meisten Fällen vielmehr Anstrengungen, Wege zu finden, interessante Aspekte des Problems mit welchen Methoden auch immer zumindest annäherungsweise zu lösen.

Im weiteren werden die grundlegenden Annahmen der Komplexitätstheorie skizziert und darauf aufbauend verschiedene Formen von Komplexitätsanalysen exemplarisch dargestellt. Im letzten Abschnitt werden dann einige Möglichkeiten angerissen, mit dem Komplexitätsproblem umzugehen.

2 Grundlegende Annahmen der Komplexitätstheorie

In der Komplexitätstheorie[2] versucht man, Probleme anhand ihres Bedarfs an *Berechnungsressourcen* (Rechenzeit, Speicherbedarf) in Abhängigkeit von der Größe der Eingabe zu klassifizieren. Dabei bedeutet der Begriff *Problem* in diesem Kontext soviel wie „generische Fragestellung", die verschiedene *Instanzen* hat. Ein spezifisches Kreuzworträtsel ist also im Sinne der Komplexitätstheorie kein Problem, sondern eine Instanz des Problems „Kreuzworträtsellösen".

Zumeist beschränkt man sich in der Komplexitätstheorie auf die Untersuchung sogenannter *Entscheidungsprobleme*, Probleme bei denen die Antwort nur „ja" oder „nein" lauten kann. Beim Kreuzworträtsellösen würde man beispielsweise nicht nach den Worten fragen, die eingesetzt werden müssen, sondern nur danach, ob das Rätsel lösbar ist. Diese Einschränkung auf Entscheidungsprobleme ist jedoch weniger einschneidend als es auf den ersten Blick erscheint. Zum einen geben die erforderlichen Berechnungsressourcen für ein Entscheidungsproblem eine untere Grenze für das korrespondierende allgemeine Problem. Zum anderen kann man in den meisten Fällen aus einem effizienten Verfahren für das Entscheidungsproblem ein effizientes Verfahren für das korrespondierende allgemeine Problem ableiten.

Als *effizient lösbar* wird ein (Entscheidungs-)Problem angesehen, wenn ein Polynom existiert, daß die Anzahl der Rechenschritte auf einer *deterministischen, sequentiellen Maschine* (z.B. Ein- oder Mehrband-Turingmaschine) in Abhängigkeit von der Instanzengröße nach oben abschätzt. Die Klasse der Probleme, die in polynomialer Zeit auf den genannten Maschinen gelöst werden können, wird mit P bezeichnet. Als nicht effizient lösbar werden alle Probleme angesehen, deren Anstieg an Rechenzeitbedarf mit der Größe der Instanz nicht polynomial beschränkt werden kann, also z.B. Probleme mit exponentiellem Zeitbedarf.

Der Unterschied zwischen polynomialem und exponentiellem Zeitbedarf wird insbesondere dann deutlich, wenn man sich die Auswirkung möglicher Fortschritte

[2] Gute Lehrbücher zu diesem Thema sind (Balcázar *et al.* 1988; Balcázar *et al.* 1990; Garey & Johnson 1979; Papadimitriou 1994).

der Rechnertechnologie auf die Lösbarkeit von Problemen vergegenwärtigt. Unter der Annahme, daß ein Problem M_1, das mit n^2 Rechenschritten gelöst werden kann, bis zu einer Instanzengröße von m_1 in *annehmbarer* Zeit mit Hilfe von konventioneller Rechnertechnologie gelöst wird, so kann eine um 10^6 schnellere Technologie Instanzen bis zur Größe $1000m_1$ in annehmbarer Zeit lösen. Im Falle des Problems M_2, das 2^n Rechenschritte benötigt und für das Instanzen bis zur Größe m_2 in annehmbarer Zeit gelöst werden können, bringt der Technologiesprung jedoch lediglich einen Zuwachs der zumutbaren Instanzengröße auf $20 + m_2$.

Man sollte sich bei der Unterscheidung zwischen „effizient“ und „nicht effizient“ lösbaren Problemen aber immer vor Augen halten, daß es sich dabei um eine mathematische Abstraktion handelt. Diese ist für die Praxis nur dann relevant, wenn im Falle polynomialen Zeitbedarfs die Exponenten der Polynome eine moderate Größe besitzen und im Falle exponentiellen Zeitbedarfs der „schlechteste“ Fall, in dem der Zeitbedarf sehr hoch ist, signifikant oft auftritt. Allerdings zeigt die Erfahrung, daß die Voraussagegüte dieser Abstraktion sehr gut ist. Aus diesem Grund ist das häufig geäußerte Argument, daß es sich bei einer Komplexitätsanalyse ja „nur um eine Analyse des schlechtesten Falls“ handelt, so lange fragwürdig, wie nicht empirische Untersuchungen zeigen, daß der „schlechteste Fall“ in der Praxis tatsächlich irrelevant ist.[3]

Um nun ein Problem bezüglich der erforderlichen Berechnungsressourcen zu klassifizieren, müssen wir also „lediglich“ feststellen, ob ein polynomialer Algorithmus existiert oder unmöglich ist. Tatsächlich scheint es aber in einer sehr großen Anzahl von Fällen, in denen keine polynomialen Algorithmen bekannt sind, unmöglich zu sein zu beweisen, daß superpolynomiale Zeit erforderlich ist. Die formale Klassifikation dieser Probleme ist eine der Hauptherausforderungen der Komplexitätstheorie.

Als ein formales Werkzeug zur Klassifikation dieser Probleme wird das Maschinenmodell der *nichtdeterministischen Turingmaschine* benutzt. Diese Maschinen haben die Möglichkeit, während der Berechnung nichtdeterministisch einen aus mehreren möglichen Nachfolgezuständen auszuwählen. Eine solche nichtdeterministische Maschine akzeptiert eine Eingabe, wenn es eine Folge von nichtdeterministischen Entscheidungen gibt, die zu einem akzeptierenden Zustand führen. Eine nichtdeterministische Maschine löst ein Entscheidungsproblem in polynomialer Zeit, wenn genau die Instanzen, die mit „ja“ beantwort werden müßten, von der Maschine akzeptiert werden, und dies mit Hilfe polynomial vieler Rechenschritte geschieht. Die Klasse aller Entscheidungsprobleme, die von nichtdeterministischen Maschinen in polynomialer Zeit gelöst werden können, wird mit NP bezeichnet. Offensichtlich gilt $\mathsf{P} \subseteq \mathsf{NP}$. Ob die Inklusion in anderer Richtung gilt, ist jedoch offen – dies ist das berühmte $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ Problem.

[3] Dann ist es jedoch auch meist möglich, analytische, komplexitätstheoretische Argumente zu finden, die diese Tatsache erklären.

Obwohl man nicht weiß, ob $\mathsf{P} \neq \mathsf{NP}$, kann man doch „schwierigste“ Probleme innerhalb der Klasse NP identifizieren. Das technische Werkzeug für die Bestimmung der relativen Schwierigkeit von Problemen ist dabei die ressourcenbeschränkte *Reduktion* zwischen Problemen. Die Klasse der „schwierigsten“ Probleme innerhalb von NP, die auch NP-*vollständig* genannt werden, haben die Eigenschaft, daß alle Probleme in NP in polynomialer Zeit lösbar sind, falls ein NP-vollständiges Problem in polynomialer Zeit gelöst werden kann. Ein Beispiel eines NP-vollständigen Problems ist das *Erfüllbarkeitsproblem für die Aussagenlogik*, kurz *SAT*. Tatsächlich ist schon das etwas einfachere Problem *3SAT* NP-vollständig, bei dem es um die Erfüllbarkeit von Formeln in konjunktiver Normalform mit höchstens drei Literalen pro Klausel geht.

Kann man zeigen, daß ein Problem mindestens so schwierig wie ein NP-vollständiges Problem ist, daß es NP-*schwierig* ist, so impliziert das, daß nach dem heutigen Stand der Kunst dieses Problem nicht in polynomialer Zeit gelöst werden kann. Die Verhältnisse zwischen polynomialen, nicht-deterministisch polynomialen, NP-vollständigen und NP-schwierigen Problemen sind in Abbildung 1 noch einmal zusammenfassend dargestellt.

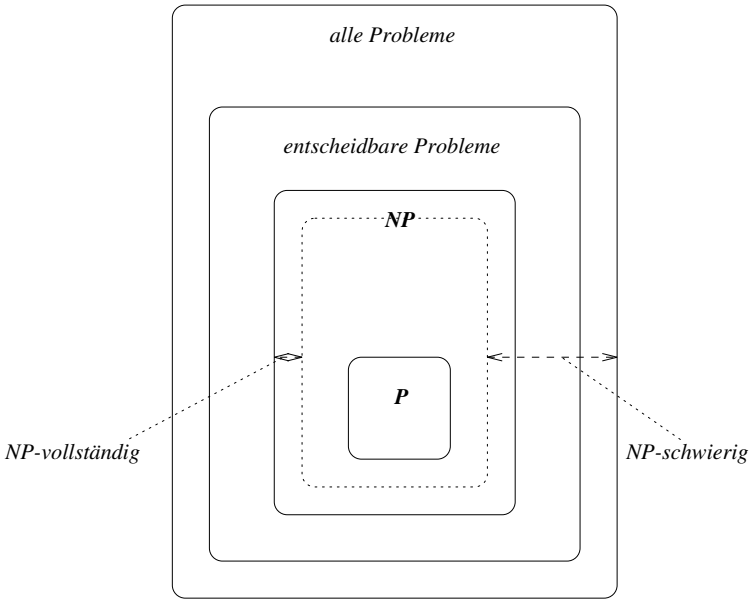

Abbildung 1: P, NP, NP-schwierige und NP-vollständige Probleme

Da es sehr viele voneinander unabhängige Versuche gegeben hat, polynomiale Algorithmen für NP-vollständige Probleme zu entwerfen, ohne daß diese Versuche je zu einem Erfolg geführt hätten, geht man heute davon aus, daß NP-schwierige Probleme tatsächlich nicht effizient lösbar sind.

Neben der Klasse NP existieren eine Menge von anderen sogenannten Komplexitätsklassen.[4] Zum einen gibt es Klassen innerhalb von P, die es erlauben parallelisierbare Probleme zu charakterisieren. Dies ist jedoch eine Thematik, die innerhalb der KI bisher weitgehend ausgeklammert worden ist (siehe aber z.B. (Kasif 1990; Kasif & Delcher 1994)). Zum anderen gibt es Klassen, die es ermöglichen, NP-schwierige Probleme genauer zu charakterisieren. Auf den Sinn solcher Charakterisierungen werden wir weiter unten eingehen.

3 Komplexitätsanalysen

Eine wesentliche Voraussetzung für eine Komplexitätsanalyse ist, daß das Problem überhaupt erst einmal eindeutig und formal spezifiziert wird, wobei unter Umständen von unwichtigen Details abstrahiert werden muß, um die Spezifikation handhabbar zu halten. Solche Problemspezifikationen, die in der KI durchaus nicht die Regel sind, haben – unabhängig von der Komplexitätsanalyse – den großen Vorteil, daß sie die wesentlichen Ideen eines Ansatzes kommunizierbar machen. Beispielsweise enthält das Papier von Brachman und Levesque (1984), das eines der ersten Papiere zum Thema Komplexität in der KI war, als einen wesentlichen Beitrag eine einfache und elegante Spezifikation der Semantik von sogenannten *Beschreibungslogiken* oder *KL-ONE-ähnlichen Sprachen* (Brachman & Schmolze 1985; Nebel & Smolka 1991; Schmolze & Woods 1992).

Die Feststellung, daß ein KI-Problem NP-schwierig sei, ist allerdings alleine meist nicht sehr interessant. Interessant ist solch ein Resultat nur dann, wenn es verschiedene Auffassungen über die Schwierigkeit des Problems gibt, wie z.B. bei der Plangenerierung in der Blockwelt (Gupta & Nau 1992), oder wenn vermutet oder behauptet wird, daß ein bestimmtes Problem effizient lösbar sei (Nebel 1988). Ein NP-Schwierigkeitsresultat alleine gibt uns meist relativ wenig Information darüber, welche Aspekte des Problems für die Schwierigkeit verantwortlich sind, welche Aspekte trotz allem effizient gelöst werden können und in welcher Beziehung das Problem zu anderen Problemen steht.

3.1 Polynomiale Spezialfälle

Eine der interessantesten Fragestellungen bei NP-schwierigen Problemen ist, ob die Formulierung des Problems vielleicht zu allgemein ist, und ob die in der Praxis auftretenden Instanzen „gutartig", d.h. effizient lösbar sind.

Obwohl beispielsweise das *SAT* Problem NP-vollständig ist, gibt es syntaktisch eingeschränkte Varianten des Problems, für die Erfüllbarkeit einer aussagenlogischen Formel in polynomialer Zeit bestimmt werden kann. Zu diesen Varianten

[4] In (Johnson 1990) findet sich eine Übersicht über den Stand der Kunst.

gehören unter anderem *HORNSAT* und *2SAT*. Bei *HORNSAT* ist die aussagenlogische Formel eine Konjunktion von Horn-Klauseln und bei *2SAT* ist sie eine Konjunktion von Klauseln, die jeweils aus maximal zwei Literalen bestehen. Wenn also in einer Anwendung die syntaktische Form der benutzten Formeln eingeschränkt werden kann, ist es möglich, das Erfüllbarkeitsproblem effizient zu lösen.

Ähnliches kann man auch bei der Anwendung von Repräsentationsformalismen in der KI beobachten. Zum Beispiel tritt bei der Anwendung des Allenschen Intervallkalküls (Allen 1983) im Kontext vom Verstehen natürlicher Sprache nur eine eingeschränkte Menge von Intervallrelationen auf (Song & Cohen 1988). Die gleiche Teilmenge von Relationen ergibt sich auch im Bereich der Diagnose von technischen Systemen (Nökel 1991). Interessanterweise ist für diese Teilmenge von Relationen das Folgerungsproblem im Intervallkalkül polynomial (Nökel 1989; Vilain *et al.* 1989), obwohl es für den allgemeinen Fall NP-schwierig ist (Vilain & Kautz 1986).

Ein weiteres Problem, das im allgemeinen NP-schwierig ist, aber in der Praxis effizient gelöst werden kann, ist die Interpretation von „Landschaftsskizzen", die aus Luftaufnahmen gewonnen wurden (Reiter & Mackworth 1989). Während das Problem im allgemeinen NP-schwierig ist, ist der Spezialfall, bei dem keine Flußquellen auf einer Straße liegen, in polynomialer Zeit entscheidbar (Selman 1994).

Selbst wenn es nicht möglich ist, anwendungsrelevante Spezialfälle zu identifizieren, ist die Analyse polynomialer Spezialfälle oft interessant, da sie uns Aufschluß darüber gibt, welche Aspekte des Problems zu der „kombinatorischen Explosion" führen. Kirousis und Papadimitriou (1985) haben beispielsweise die Komplexität des Problems untersucht, eine Strichzeichnung als die Projektion der sichtbaren Teile einer aus Polyedern bestehenden Szene zu interpretieren. Dieses Problem ist im allgemeinen NP-vollständig, wie Kirousis und Papadimitriou (1985) zeigen konnten, scheint aber in der Praxis meist in annehmbarer Zeit gelöst werden zu können (Waltz 1975). Eine mögliche Erklärung für dieses Phänomen könnte die Beobachtung sein, daß das Problem für Szenen, in denen alle Flächen senkrecht aufeinander stehen, in linearer Zeit gelöst werden kann (Kirousis & Papadimitriou 1985).

3.2 Klassifizierung aller Spezialfälle

Die Identifikation von polynomialen Spezialfällen gibt uns Aufschluß über einige der möglichen Quellen der Komplexität. Eine Untersuchung *aller möglichen Spezialfälle entlang verschiedener Dimensionen* und eine Bestimmung der exakten Grenze zwischen polynomialen und NP-schwierigen Spezialfällen ist aber sehr viel

interessanter.[5] Zudem kann eine solche vollständige Klassifizierung benutzt werden, um schnell festzustellen, ob eine Anwendung nur einen effizient zu lösenden Spezialfall des allgemeinen Problems benötigt.

Ein Beispiel einer solchen vollständigen Klassifikation ist das Papier von Donini et al. (1991), in dem die Komplexität der Subsumption für alle Beschreibungslogiken bestimmt wird, die die von Brachman und Levesque (1984) vorgeschlagene Sprache $\mathcal{FL}^-$ sowie weitere in der Literatur vorgeschlagenen Operatoren (Nebel 1990a) enthalten. Donini et al. (1991) konnten zwei bezüglich der Menge der zugelassenen Operatoren maximale Beschreibungslogiken identifizieren, für die das Subsumptionsproblem polynomial ist. Damit wurde das von Brachman und Levesque (1984) gestellte Problem der Bestimmung des „Berechnungskliffs" in Beschreibungslogiken gelöst.

Ähnliche Analysen wurden im Bereich der Plangenerierung durchgeführt. Bylander (1994) klassifizierte propositionale STRIPS-Planung (Fikes & Nilsson 1971) bezüglich der Ausdrucksfähigkeit der Operatoren und Bäckström und Nebel (1993) klassifizierten einen ähnlichen Planungsformalismus bezüglich lokaler und globaler Beschränkungen auf der Menge der Operatoren. In beiden Fällen sind die Beschränkungen, die zu Polynomialität führen, jedoch sehr einschneidend, so daß diese Resultate vermutlich nur für sehr wenige Anwendungen relevant sind.

In den beschriebenen Fällen war es möglich, den Raum aller Spezialfälle manuell zu analysieren. Dieser Raum kann jedoch so groß sein, daß eine manuelle Analyse unmöglich ist. Bei der Klassifizierung aller Spezialfälle des Allenschen Intervallkalküls (Allen 1983) bezüglich der zugelassenen Intervallrelationen gibt es beispielsweise $2^{2^{13}}$ Spezialfälle. Durch Untersuchungen zur Struktur dieses Raums und einer rechnerunterstützten Fallanalyse war es aber auch in diesem Fall möglich, die genaue Grenze zwischen polynomialen und NP-vollständigen Spezialfällen zu bestimmen (Nebel & Bürckert 1994).

3.3 Differenzierende Analysen

In vielen Fällen ist ein Problem nur informell gegeben, und es existieren verschiedene Formalisierungen. Im Falle der nicht-strikten Vererbung von Eigenschaften in semantischen Netzen gibt es zum Beispiel mindestens acht verschiedene formale Ansätze (Touretzky *et al.* 1987), und es ist nicht klar, welcher dieser Ansätze der beste ist. Eine Möglichkeit, zwischen diesen Ansätzen zu differenzieren, ist natürlich ihre Komplexität. Hierbei handelt es sich also nicht um die Identifizierung von effizient lösbaren Spezialfällen, sondern um den Vergleich verschiedener

[5] Man beachte dabei, daß diese Grenzziehung nur für die gewählten Dimensionen gilt. Im allgemeinen existiert kein größtes polynomiales Teilproblem eines NP-vollständigen Problems (Orponen *et al.* 1986).

miteinander konkurrierender (und inkompatibler) Formalisierungen eines informell gegebenen Problems.

Eine Komplexitätsanalyse (Selman & Levesque 1993) zeigt, daß es verschiedene Komplexitätsquellen bei der nicht-strikten Vererbung gibt. Zum einen werden im schlechtesten Fall exponentiell viele „Extensionen“ erzeugt, zum anderen kann jede Extension wiederum exponentiell viele Pfade eines Netzes enthalten. Dabei stellt sich die Frage, ob es tatsächlich notwendig ist, alle Extensionen und alle Pfade zu betrachten, um zu entscheiden, ob ein Objekt eine bestimmte Eigenschaft ererbt.

Die Untersuchung der verschiedenen Formalisierungen des intuitiven Begriffs von nicht-strikter Vererbung zeigt, daß Schlußfolgerungen in Vererbungsnetzen lediglich bezüglich der „skeptischen“ Variante bei „entkoppelter“ Vererbung in polynomialer Zeit berechnet werden können. In allen anderen Fällen ist das Problem NP-schwierig. Auch unter der Einschränkung, daß die Vererbungsnetze nur logarithmische Tiefe haben, also sehr „buschig“ sind, bleiben diese Ergebnisse im wesentlichen gültig (Selman & Levesque 1993).

3.4 Komplexitätsanalysen als Plausibilitätstest

Behauptungen über Zusammenhänge zwischen Problemen oder über die relative Effizienz von verschiedenen Methoden sollten auch komplexitätstheoretisch sinnvoll sein. Wenn beispielsweise behauptet wird, daß ein Problem P *Teilproblem* eines anderen Problems Q ist, so sollte P nicht schwieriger sein als Q. Ähnliches gilt für den Fall, daß behauptet wird, eine Methode M_1 sei effizienter als M_2.

Dean und Boddy (1988) untersuchten beispielsweise das Problem der *temporalen Projektion* über partiell geordneten Mengen von Aktionen. Dabei handelt es sich um die Überprüfung, ob eine Bedingung notwendigerweise – also in allen Linearisierungen der partiellen Ordnung – vor einer Aktion gilt.

Die Motivation bei der Analyse des temporalen Projektionsproblems und der Entwicklung eines Approximationsverfahrens für dieses Problem war die Vermutung, daß temporale Projektion ein essentielles Teilproblem der nicht-linearen Planung sei. Aus dem Papier (Dean & Boddy 1988) geht hervor, daß die Autoren der Meinung sind, daß insbesondere die *Planvalidierung* der Kontext ist, in dem die temporale Projektion von Nutzen sein kann. Dabei wird ein Plan als *valide* angesehen, wenn er erfolgreich ausgeführt werden kann und den Zielzustand herstellt.

Wenn man verschiedene Spezialfälle untersucht, die durch Beschränkungen der Form der Operatoren und der Zustände definiert sind, so stellt man fest, daß temporale Projektion strikt schwieriger sein kann als Plangenerierung (Nebel & Bäckström 1994). Außerdem stellt man fest, daß im allgemeinen Fall Planvalidierung und temporale Projektion die selbe Komplexität besitzen und es keine

offensichtliche Dekomposition des Planvalidierungsproblems in temporale Projektionsprobleme gibt. Für den wichtigen Spezialfall, daß alle Aktionen nicht-konditional sind, existiert allerdings eine solche Dekomposition. In diesem Fall kann man die Validität eines Planes auf eine Konjunktion von temporalen Projektionen reduzieren. Verblüffenderweise ist jedoch in diesem Fall die Komplexität des ursprünglichen Problems niedriger als die Komplexität der Teilprobleme. Planvalidierung kann in polynomialer Zeit entschieden werden, während temporale Projektion NP-schwierig ist (Nebel & Bäckström 1994). Die Hypothese, daß temporale Projektion, so wie von Dean und Boddy (1988) definiert, ein essentielles Teilproblem des nicht-linearen Planens und der Planvalidierung ist, scheint also zumindest von einer algorithmischen Sichtweise her sehr fragwürdig zu sein.

Ein Beispiel, bei dem einfache komplexitätstheoretische Überlegungen benutzt wurden, um die Plausibilität einer Hypothese bezüglich relativer Effizienz verschiedener Methoden zu überprüfen, ist der Vergleich von Plangenerierung und Plananpassung (Nebel & Koehler 1994). Es konnte gezeigt werden, daß die Plananpassung keinen komplexitätstheoretischen Gewinn bringt, und daß in machen Fällen Plananpassung sogar zu zusätzlichen Kosten führen kann.

3.5 Klassifizierung NP-schwieriger Probleme

Während die Klassifizierung „polynomial vs. NP-schwierig“ in den meisten Fällen ausreichend ist, um die Grenze zwischen effizient und nicht effizient lösbaren Fällen zu bestimmen,[6] ist es in manchen Fällen sinnvoll, die genaue Komplexitätsklasse eines NP-schwierigen Problems zu bestimmen.

Die Tatsache, daß ein Problem nicht nur NP-schwierig, sondern auch vollständig für NP ist, besagt beispielsweise, daß dieses Problem unter Umständen mit ähnlichen Methoden angegangen werden kann wie andere NP-vollständige Probleme. Ist das Problem jedoch nicht als NP-vollständig klassifizierbar, stellt sich in vielen Fällen heraus, daß es vollständig für eine höhere Komplexitätsklasse ist. In diesen Fällen muß man meist sehr viel einschneidendere Restriktionen in Kauf nehmen, um polynomiale Spezialfälle zu erhalten.

Wie oben schon erwähnt wurde, sind im Falle propositionaler STRIPS Planung die polynomialen Spezialfälle sehr stark eingeschränkt. Der tiefere Grund dafür scheint zu sein, daß dieses Problem PSPACE-vollständig ist (Bylander 1994). Dabei ist PSPACE die Klasse der Probleme, die mit Hilfe polynomialen Speicherplatzes (ohne Zeitbeschränkung) gelöst werden können. Ebenso wie man annimmt, daß P echt in NP enthalten ist, wird auch vermutet, daß NP echt in PSPACE enthalten ist. Das heißt, man vermutet, daß die PSPACE-vollständigen Probleme „schwerer“ als die NP-vollständigen Probleme sind.

[6] Es gibt jedoch auch Probleme, die nicht in P zu liegen scheinen, von denen aber nicht angenommen wird, daß sie NP-schwierig sind.

Ein Problem, daß von seiner Komplexität her zwischen den NP-vollständigen und den PSPACE-vollständigen Problemen liegt, ist das Problem der propositionalen Wissensrevision. Dabei geht es darum zu bestimmen, ob eine Proposition aus einer Wissensbasis, die aufgrund neuer Information *minimal* geändert wurde, logisch folgt. Dieses Problem ist im Falle *syntaktisch definierter Minimalität* vollständig für die Komplexitätsklasse Π_2^p (Nebel 1991), die zwischen NP und PSPACE angesiedelt ist. Intuitiv besagt dies, daß es zwei Quellen der Komplexität in diesem Problem gibt. In diesem Fall sind es propositionale Folgerbarkeit und die exponentielle Anzahl alternativer minimaler Änderung der Wissensbasis. Insbesondere folgt aus der Π_2^p-Vollständigkeit, daß beide Quellen beseitigt werden müssen, um polynomiale Spezialfälle zu erhalten (Eiter & Gottlob 1992; Nebel 1994).

Interessanterweise besitzt das Wissensrevisionsproblem die gleiche Komplexität wie eine Menge anderer Probleme des Alltagsschließens, wie z.B. das propositionale Abduktionsproblem (Eiter & Gottlob 1993) und viele Formen des propositionalen nichtmonotonen Schließens (Stillman 1992; Gottlob 1992; Papadimitriou & Sideri 1992). Dies impliziert, daß all diese Probleme in polynomialer Zeit ineinander übersetzbar sind. In einem abstrakten algorithmischen Sinne sind also all diese Probleme äquivalent. Obwohl uns dieses Resultat nicht hilft, effiziente Verfahren zur Lösung dieser Probleme zu finden, so gibt es uns doch eine gewisse Einsicht in die algorithmische Natur dieser Probleme und hilft, die verschiedenen Formen des Alltagsschließens miteinander in Beziehung zu setzen.

4 Umgang mit dem Komplexitätsproblem

Ist ein Problem als NP-schwierig klassifiziert worden, stellt sich die Frage nach dem weiteren Vorgehen. Wie schon in der Einleitung bemerkt, ist man an diesem Punkt normalerweise nicht am Ende der Untersuchung angekommen, sondern man versucht, Lösungen von Teilaspekten des Problems zu finden. In manchen Fällen reicht es aus, einen Spezialfall zu lösen. In anderen Fällen muß man jedoch den Zeitbedarf gegen die Qualität der Lösung abwägen, wobei man verschiedene Wege einschlagen kann.

Tatsächlich kann man den Umgang mit dem Komplexitätsproblem als eines der zentralen Themen der KI auffassen. Typischerweise versucht man in der KI, NP-schwierige Probleme durch den Einsatz von Heuristiken und insbesondere durch Ausnutzung domänenspezifischen Wissens zumindest approximativ zu lösen (Enzinger *et al.* 1994; Kaindl 1994). Dabei wird die Entwicklung effizienter Verfahren in den meisten Fällen durch praktisches Experimentieren bestimmt und nicht von abstrakten Überlegungen zur Komplexität eines Problems geleitet (Bylander 1991).

Analytische, komplexitätstheoretische Betrachtungen können dann „nur“ noch die Erklärung dafür liefern, warum eine bestimmte Vorgehensweise erfolgreich ist. Solche Analysen sind aber notwendig, wenn es um die Bestimmung des Anwendungsbereiches, die Übertragbarkeit und das “scaling up” von Verfahren geht.

4.1 Natürliche Einschränkungen des Problems

Wie schon im letzten Abschnitt angemerkt, ist es oft möglich, daß in einer Anwendung gar nicht das allgemeine Problem, sondern lediglich ein Spezialfall gelöst werden muß. Die im letzten Abschnitt vorgestellten Beispiele machten dabei im wesentlichen davon Gebrauch, daß eine Anwendung nur ein eingeschränktes Vokabular der zugrundeliegenden formalen Repräsentationssprache benutzt. Dies ist jedoch nicht die einzige mögliche Art, zu polynomialen Spezialfällen zu gelangen. In manchen Fällen können Problemparameter identifiziert werden, die innerhalb einer Anwendung als konstant oder logarithmisch beschränkt angenommen werden können und damit zu einem polynomialen Zeitverhalten führen.

Im Falle der morphologischen Analyse von natürlichsprachlichen Worten wurde beispielsweise die Zweiebenen-Morphologie vorgeschlagen (Koskenniemi 1983), ein Ansatz, der NP-vollständig in der Anzahl der morphologischen Regeln ist, wie Barton (1986) zeigte. Allerdings ist für die NP-Vollständigkeit eine spezielle Art von Regeln verantwortlich, die „Harmonie-Prozesse“ zwischen Vokalen realisieren. Diese spezielle Art von Regeln ist für die Beschreibung von bekannten natürlichen Sprachen nur in sehr begrenzter Zahl erforderlich. Tatsächlich scheint das Finnische mit zwei solchen Regeln die Sprache zu sein, die die meisten Harmonieregeln benötigt (Koskenniemi & Church 1988). Unter der Voraussetzung, daß die Anzahl der Harmonieregeln auf zwei beschränkt werden kann, ist die morphologische Analyse jedoch polynomial.

Ähnlich ist der Fall des Subsumptionsproblems in Beschreibungslogiken gelagert, wenn man zuläßt, daß Begriffe *definiert* werden können – eine Möglichkeit, die in allen existierenden Repräsentationssystemen für Beschreibungslogiken unterstützt wird. In Komplexitätsanalysen von Beschreibungslogiken wird meist von dieser Möglichkeit abstrahiert (Brachman & Levesque 1984) und angenommen, daß alle Begriffe undefiniert seien. Bei einer Komplexitätsanalyse stellt man aber fest, daß diese Abstraktion nicht zu vernachlässigende Kosten impliziert. Während das Subsumptionsproblem in der von Brachman und Levesque eingeführten Sprache $\mathcal{FL}^-$ polynomial ist, wenn man annimmt, daß alle Begriffe undefiniert sind, wird das Problem NP-schwierig, falls Begriffe definiert werden können (Nebel 1990b). Allerdings scheint das Subsumptionsproblem für $\mathcal{FL}^-$ in den praktisch relevanten Fällen effizient lösbar zu sein, auch wenn Begriffsdefinitionen erlaubt sind. Dies scheint auch für das ähnlich gelagerte Problem der Subsumptionsbestimmung in objekt-orientierten Datenbanken zu gelten (Bergamaschi & Nebel 1994). Der Grund dafür ist, daß Definitionen im Normalfall keine

beliebige Schachtelungstiefe haben. Tatsächlich kann man zeigen, daß bei einer logarithmischen Beschränkung der Definitionstiefe das Subsumptionsproblem polynomial wird.

In diesen beiden Fällen haben wir es also mit der nicht sehr häufigen Situation zu tun, daß ein als nicht effizient lösbar klassifiziertes Problem in der Praxis nicht zu übermäßigem Zeitbedarf führt. Die NP-Schwierigkeitsresultate geben uns aber einen Hinweis darauf, daß der Rechenzeitbedarf sehr empfindlich bezüglich der identifizierten Problemparameter ist – eine Vermutung, die bei einem empirischen Test von verschiedenen auf Beschreibungslogiken basierenden Repräsentationssystemen bestätigt wurde (Heinsohn *et al.* 1994).

Neben der Möglichkeit, daß Werte für einen Problemparameter als mit Sicherheit zur Polynomialität führend identifiziert werden können, gibt es auch die Möglichkeit, daß bestimmte Werte eines Problemparameters mit hoher Wahrscheinlichkeit zu schnellen Antworten führen. Empirische Untersuchungen des Graphfärbbarkeitsproblems (Cheeseman *et al.* 1991) und des Erfüllbarkeitsproblems für Klauseln mit konstanter Länge (Mitchell *et al.* 1992) scheinen beispielsweise den Schluß nahe zu legen, daß es durch gewisse Problemparameter spezifizierte Bereiche gibt, in denen Instanzen in den meisten Fällen relativ schnell gelöst werden können.

4.2 Erzwungene Einschränkungen des Problems

Im Falle von anwendungsunabhängigen Systemen, wie beispielsweise Wissensrepräsentationssystemen, kann man nicht davon ausgehen, daß die Anwendung schon dafür sorgen wird, daß „alles gut geht“. Zumindest sollte man in der Lage sein, die Bedingungen zu spezifizieren, unter denen das System zuverlässig arbeiten kann.

Brachman und Levesque gehen in ihren Papieren (Brachman & Levesque 1984; Levesque & Brachman 1987) sogar noch einen Schritt weiter. Sie wollen einem Repräsentationssubsystem in einer KI-Anwendung einen ähnlichen Status wie einem Datenbanksystem in konventionellen Informatikanwendungen geben. Es soll das relevante Wissen „speichern“ und ein garantiertes Antwortverhalten zeigen (Levesque 1986). Als eine Möglichkeit, solch ein Verhalten zu garantieren, schlagen sie vor, die Ausdruckskraft von Repräsentationssprachen soweit zu begrenzen, daß die relevanten Schlußfolgerungsprobleme polynomial werden.

Diese Forderung, die von einem technischen Standpunkt her sinnvoll scheint, wird jedoch heftig kritisiert (Doyle & Patil 1991). Der Hauptkritikpunkt ist dabei, daß die Einschränkung der Ausdruckskraft einer Repräsentationssprache, die zur Polynomialität führt, die Repräsentationssprache so ausdrucksschwach macht, daß sie kaum noch verwendbar ist. Tatsächlich muß man Brachman und Levesque (Brachman & Levesque 1984) an dieser Stelle aber zu gute halten, daß sie die

Einschränkung der Ausdruckskraft einer Repräsentationssprache als *einen* Weg sehen, Polynomialität zu erreichen, aber nicht als den einzig möglichen propagieren.

Diese Diskussion wirft die Frage auf, ob es prinzipiell möglich ist, deklarative Wissensrepräsentationsformalismen und -systeme unabhängig von ihrer Anwendung zu betrachten. Dies scheint zumindest zweifelhaft zu sein, da für verschiedene Anwendungen unterschiedliche Anforderungen an die Ausdruckskraft der Repräsentationssprache gestellt werden und die Ansprüche an die Qualität und Zuverlässigkeit der Antwort variieren, die Effizienz aber in den meisten Fällen eine wesentliche Rolle spielt.

4.3 Approximationen

Ist es nicht möglich, tolerierbare Restriktionen des Problems zu identifizieren, die zu Polynomialität führen, so kann man unter Umständen durch Reduktion der Ansprüche an die Qualität der Antworten ein besseres Antwortzeitverhalten erreichen. Insbesondere ist es oft möglich, durch die Aufgabe des Optimalitätsanspruchs zu polynomialen Lösungen zu gelangen. Beispielsweise ist die Generierung von nicht-optimalen Plänen in der Blockwelt polynomial, während ein Optimalitätsanspruch zu NP-Schwierigkeit führt (Gupta & Nau 1992).

Allerdings ist bei den meisten KI-Problemen die Optimalität einer Lösung sowieso von untergeordneter Bedeutung oder es existiert nicht einmal ein solcher Begriff. Insbesondere wenn wir die im wesentlichen logikbasierten Probleme betrachten, wie z.B. Subsumptionsbestimmung, Diagnose, temporale Projektion oder nichtmonotones Schließen, wird es schwierig, den Begriff der Approximation einer Lösung überhaupt vernünftig zu definieren.

Deshalb weicht man oft auf korrekte aber unvollständige Verfahren aus. Um aber auch weiterhin ein gewisses Antwortverhalten *garantieren* zu können, versucht man, diese Abschwächung möglichst deklarativ zu beschreiben. Ein Beispiel dafür ist Patel-Schneider's vierwertige Semantik für Beschreibungslogiken (Patel-Schneider 1989), die zu einer bezüglich der Standardsemantik korrekten aber unvollständigen Subsumptionsrelation führt. Andere Beispiele für diesen Weg sind die von Levesque und Lakemeyer vorgeschlagenen Varianten für Modallogiken (Levesque 1984; Lakemeyer 1987).

Oft werden auch korrekte (aber unvollständige) mit vollständigen (aber inkorrekten) Verfahren kombiniert, wie z.B. in den Arbeiten von Cadoli und Schaerf (1991; 1992), die Folgen von Interpretationen zur Approximation benutzen, oder in der Arbeit von Selman und Kautz (1991), die Horn-Theorien zur Approximation einer beliebigen propositionalen Theorie verwenden. Im letzteren Fall wird angenommen, daß die Berechnung der approximierenden Horn-Theorien "off-line"

vorgenommen wird, da diese Berechnung sehr aufwendig werden kann. Erste Erfahrungen mit dieser Methode scheinen zu belegen, daß dieses Vorgehen in einigen Fällen tatsächlich sinnvoll ist (Kautz & Selman 1994).

Eine weitere interessante Variante zur Approximation der Lösung von sogenannten "constraint-satisfaction" Problemen, z.B. dem Problem, eine erfüllende Belegung für eine propositionale Formel zu finden, ist nicht-systematische Suche. In vielen Fällen kann mit Hilfe solcher Verfahren eine Lösung schneller gefunden werden als mit systematischer Suche, wie beispielsweise die Arbeiten zu dem GSAT-Verfahren zeigen (Selman *et al.* 1992). Im Gegensatz zu den oben beschriebenen Verfahren kann man jedoch bei GSAT und ähnlichen Ansätzen keine Garantien für das Finden einer Lösung (auch abgeschwächt) geben.

Schließlich sollte auch noch das Gebiet der Lernbarkeitstheorie erwähnt werden.[7] Dabei geht es um das Lernen von Begriffen aus Beispielen in polynomialer Zeit, wobei allerdings nicht erwartet wird, daß man den exakten Begriff erlernt. Stattdessen gesteht man zu, daß dieser Begriff durch einen Lernalgorithmus approximiert wird, also mit einem Fehler behaftet ist, der allerdings quantifizierbar sein muß.

4.4 Redefinition des Problems

Schließlich kann man die NP-Schwierigkeit eines Problems auch zum Anlaß nehmen, das Problem radikal umzudefinieren. Dies liegt insbesondere dann nahe, wenn durch die ursprüngliche Problemdefinition eine kognitive Fähigkeit beschrieben werden soll, die Menschen ohne großen (Rechenzeit-) Aufwand erbringen.

Enzinger et al. (1994) betrachten beispielsweise verschiedene KI-Probleme, die traditionell als (NP-schwierige) Suchprobleme formuliert werden, und kontrastieren diese Formulierungen mit Problemvereinfachungen und Reformulierungen, deren Plausibilität sie mit Argumenten aus den Kognitionswissenschaften begründen.

Man muß sich an dieser Stelle aber klar machen, daß die ursprüngliche Formulierung mit der Reformulierung unter Umständen überhaupt nichts mehr zu tun hat und auch der Anwendungsbereich völlig verschieden sein kann. Bei einem Vergleich der konsistenzbasierten Diagnosemethode (Reiter 1987) mit der fallbasierten Diagnosemethode (Riesbeck & Schank 1989) kann man beispielsweise feststellen (Enzinger *et al.* 1994), daß es bei der ersten Methode (anscheinend) erforderlich ist, einen exponentiellen Lösungsraum abzusuchen,[8] während die zweite

[7] Eine Einführung in das Gebiet findet sich beispielsweise in dem Artikel von Hoffmann (1991).

[8] Tatsächlich ist das Problem der Bestimmung von notwendigerweise defekten Komponenten in diesem Kontext im propositionalen Fall Π_2^p-vollständig, wie sich beispielsweise direkt aus den Ergebnissen in (Nebel 1991) ergibt.

Methode lineare Komplexität besitzt – ein Grund, die fallbasierte Methode zu favorisieren.

Bei dieser Wertung sind allerdings zwei Punkte zu beachten. Erstens ist für die konsistenzbasierte Diagnose kein Wissen über die Zusammenhänge zwischen Fehlern und ihren Symptomen erforderlich, sondern es wird lediglich das Wissen über die Struktur des Systems und der Funktion der einzelnen Komponenten benötigt. Das heißt, die konsistenzbasierte Diagnose ist auch in solchen Kontexten anwendbar, in denen noch keine Erfahrung mit den möglichen Auswirkungen von Komponentenfehlern vorliegen. Solche Erfahrungen sind jedoch eine unabdingbare Voraussetzung für die fallbasierte Diagnose. Zweitens ist unter der bei der fallbasierten Diagnose gemachten Annahme, daß nur eine einzige Komponente ausfällt, auch die konsistenzbasierte Diagnose polynomial, falls – wie in den meisten Fällen gegeben – die Systembeschreibung in propositionaler Horn-Logik gegeben ist.

4.5 Zeitbeschränkte Exploration des Suchraums

In den weitaus meisten Fällen wird man jedoch nicht umhin kommen, mit der NP-Schwierigkeit eines Problems zu leben und zu akzeptieren, daß man nur Instanzen bis zu einer gewissen Größe lösen kann. Natürlich kann man versuchen, die in diesen Fällen immer notwendige Suche durch gute Heuristiken zu begrenzen oder die elementaren Schritte (die in polynomialer Zeit berechnet werden können) bei der Exploration des Suchraums möglichst groß zu machen (McAllester 1991). Die Größe der Instanzen, die noch in zumutbarer Zeit lösbar ist, wird jedoch wegen des exponentiellen Laufzeitanstiegs meist moderat bleiben.

5 Zusammenfassung

Falls man, wie allgemein üblich, KI als Wissenschaftsgebiet begreift, das die Analyse und Nachbildung kognitiver Fähigkeiten mit Hilfe von Berechnungsprozessen zum Gegenstand hat, bietet es sich an, die in der Informatik entwickelten analytischen Werkzeuge auch in der KI anzuwenden. Insbesondere die Komplexitätstheorie ist geeignet, die Analyse der algorithmischen Struktur von KI-Problemen zu unterstützen. Solche Analysen, die eine logische Analyse ergänzen, können helfen, Quellen der Komplexität zu identifizieren und Aspekte des Problems zu isolieren, die effizient gelöst werden können. Daneben kann die Komplexitätstheorie eingesetzt werden, verschiedene Ansätze zu vergleichen oder in Beziehung zu setzen, sowie Hypothesen über die relative Effizienz verschiedener Methoden zu überprüfen.

In diesem Zusammenhang markiert die Feststellung, daß ein Problem NP-schwierig ist, im Normalfall nicht das Ende der Untersuchung, sondern stellt eine Her-

ausforderung dar, die für die Praxis relevanten Fälle effizient zu lösen. Dies kann durch Einschränkung des Problems, Approximation oder andere Ansätze geschehen. Die Feststellung der NP-Schwierigkeit kann unter Umständen auch der Ausgangspunkt für die Redefinition einer Problemstellung sein, wenn mit Hilfe der ursprünglichen Problemdefinition eine kognitive Fähigkeit beschrieben werden sollte, die Menschen ohne großen Aufwand erbringen. In vielen Fällen muß man sich jedoch damit abfinden, daß ein Problem exponentielle Zeit erfordert und aus diesem Grund nur Instanzen moderater Größe in angemessener Zeit gelöst werden können.

Danksagung

Ich möchte mich bei Gerd Brewka, Andrea Hemprich, Jana Koehler, Werner Nutt, Uwe Schöning und den anonymen Gutachtern für Kommentare zu früheren Versionen dieses Papiers bedanken.

Literatur

J. F. Allen (1983), Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

C. Bäckström und B. Nebel (1993), Complexity results for SAS^+ planning. In: *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, S. 1430–1435, Chambery, Frankreich.

J. L. Balcázar, J. D. und J. Gabarró (1988), *Structural Complexity I.* Springer-Verlag, Berlin, Heidelberg, New York.

J. L. Balcázar, J. Diaz und J. Gabarró (1990), *Structural Complexity II.* Springer-Verlag, Berlin, Heidelberg, New York.

G. E. Barton (1986), Computational complexity in two-level morphology. In: *Proceedings of the 14th Annual Meeting of the ACL*, S. 53–59, New York, NY.

S. Bergamaschi und B. Nebel (1994), Automatic building and validation of complex object database schemata supporting multiple inheritance. *Applied Intelligence*, 4(2): 185–204.

W. Bibel, S. Hölldobler und T. Schaub (1993), *Wissensrepräsentation und Inferenz: Eine grundlegende Einführung.* Vieweg, Braunschweig.

R. J. Brachman und H. J. Levesque (1984), The tractability of subsumption in frame-based description languages. In: *Proceedings of the 4th National Conference of the American Association for Artificial Intelligence*, S. 34–37, Austin, TX.

R. J. Brachman und J. G. Schmolze (1985), An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216.

T. Bylander (1991), Tractability and Artificial Intelligence. *Journal of Experimental and Theoretical Artificial Intelligence*, 3:171–178.

T. Bylander (1994), The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69(1–2):165–204.

M. Cadoli und M. Schaerf (1991), Approximate entailment. In: E. Ardizzone, S. Gaglio, and F. Sorbello (Hrsg.), *Trends in Artificial Intelligence: Proc. of the 2nd Congress of the Italian Association for Artificial Intelligence, AI*IA*, S. 68–77. Springer-Verlag, Berlin, Heidelberg, New York.

M. Cadoli und M. Schaerf (1992), Approximation in concept description languages. In: B. Nebel, W. Swartout, and C. Rich (Hrsg.), *Principles of Knowledge Representation and Reasoning: Proceedings of the 3rd International Conference*, S. 342–353, Cambridge, MA. Morgan Kaufmann.

P. Cheeseman, R. Kanefsky und W. M. Taylor. Where the *really* hard problems are. In: *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, S. 331–337, Sydney, Australien. Morgan Kaufmann.

T. L. Dean und M. Boddy (1988), Reasoning about partially ordered events. *Artificial Intelligence*, 36(3):375–400.

F. M. Donini, M. Lenzerini, D. Nardi und W. Nutt (1991), Tractable concept languages. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, S. 458–465, Sydney, Australien. Morgan Kaufmann.

J. Doyle und R. S. Patil (1991), Two theses of knowledge representation: Language restrictions, taxonomic classification, and the utility of representation services. *Artificial Intelligence*, 48(3):261–298.

T. Eiter und G. Gottlob (1992), On the complexity of propositional knowledge base revision, updates, and counterfactuals. *Artificial Intelligence*, 57:227–270.

T. Eiter und G. Gottlob (1993), The complexity of logic-based abduction. In P. Enjalbert, A. Finkel, and K. W. Wagner (Hrsg.), *Proceedings Tenth Symposium on Theoretical Aspects of Computing STACS-93*, pages 70–79, Würzburg. Springer-Verlag.

A. Enzinger, F. Puppe und G. Strube (1994), Problemlösen ohne Suchen? *KI*, 1/94:73–81.

R. E. Fikes und N. Nilsson (1971), STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208.

M. R. Garey und D. S. Johnson (1979) *Computers and Intractability—A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA.

M. R. Genesereth und N. J. Nilsson (1987), *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann, Los Altos, CA.

G. Gottlob (1992), Complexity results for nonmonotonic logics. *Journal for Logic and Computation*, 2(3).

N. Gupta und D. S. Nau (1992), On the complexity of blocks-world planning. *Artificial Intelligence*, 56(2):223–254.

J. Heinsohn, D. Kudenko, B. Nebel und H.-J. Profitlich (1994), An empirical analysis of terminological representation systems. *Artificial Intelligence*, 68(2):367-397.

A. G. Hoffmann (1991), Die Theorie des Lernbaren – ein Überblick, *KI* 1/91:7–11.

D. S. Johnson (1990), A catalog of complexity classes. In: J. van Leeuwen (Hrsg.), *Handbook of Theoretical Computer Science, Vol. A*, S. 67–161. MIT Press.

H. Kaindl (1994), Problemlösen durch Suche. *KI*, 1/94:81–84, 1994.

S. Kasif und A. L. Delcher (1994), Local Consistency in parallel constraint networks. *Artificial Intelligence*, 69(1–2):307–327.

S. Kasif (1990), On the parallel complexity of discrete relaxation in constraint networks. *Artificial Intelligence*, 45(3):275–286.

H. A. Kautz und B. Selman (1994), An empirical evaluation of knowledge compilation. In: *Proceedings of the 12th National Conference of the American Association for Artificial Intelligence*, Seattle, WA. MIT Press. Erscheint demnächst.

L. M. Kirousis und C. H. Papadimitriou (1985), The complexity of polyhedral scenes. In: *The 26th Annual Symposium on the Foundations of Computer Science*, S. 175–185, Portland, OR.

K. Koskenniemi und K. W. Church (1988), Complexity, two-level morphology and Finish. In: *Proceedings of the 12th International Conference on Computational Linguistics*, S. 335–340, Budapest, Ungarn.

K. Koskenniemi (1983), Two-level morphology: A general computational model for word-form recognition and production. Technischer Bericht No. 11, University of Helsinki, Department of General Linguistics, Helsinki, Finnland.

G. Lakemeyer (1987), Tractable meta-reasoning in propositional logics of belief. In: In: *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, S. 402–408, Mailand. Morgan Kaufmann.

H. J. Levesque und R. J. Brachman (1987), Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3:78–93.

H. J. Levesque (1984), A logic of implicit and explicit belief. In: *Proceedings of the 4th National Conference of the American Association for Artificial Intelligence*, S. 198–202, Seattle, WA.

H. J. Levesque (1986), Making believers out of computers. *Artificial Intelligence*, 30(1):81–108.

H. J. Levesque (1988), Logic and the complexity of reasoning. *Journal of Philosophical Logic*, 17:355–389.

D. McAllester (1991), Natural language based inference procedures applied to Schubert's steamroller. In: *Proceedings of the 9th National Conference of the American Association for Artificial Intelligence*, S. 915–920, Anaheim, CA. MIT Press.

D. Mitchell, B. Selman und H. J. Levesque. Hard and easy distributions of SAT problems. In: *Proceedings of the 10th National Conference of the American Association for Artificial Intelligence*, S. 459–465, San Jose, CA. MIT Press.

B. Nebel and C. Bäckström (1994), On the computational complexity of temporal projection, planning, and plan validation. *Artificial Intelligence*, 66(1):125–160.

B. Nebel und H.-J. Bürckert (1994), Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. In: *Proceedings of the 12th National Conference of the American Association for Artificial Intelligence*, S. 356–361, Seattle, WA. MIT Press.

B. Nebel und J. Koehler (1994), Plan reuse versus plan generation: A theoretical and empirical analysis, *Artificial Intelligence*. Erscheint demnächst. Eine Vorversion ist erhältlich als DFKI Bericht RR-93-33.

B. Nebel und G. Smolka (1991), Attributive description formalisms ... and the rest of the world. In: O. Herzog and C. Rollinger (Hrsg.), *Text Understanding in LILOG*, Vol. 546 *Lecture Notes in Artificial Intelligence*, S. 439–452. Springer-Verlag, Berlin, Heidelberg, New York.

B. Nebel (1988), Computational complexity of terminological reasoning in BACK. *Artificial Intelligence*, 34(3):371–383.

B. Nebel (1990a), *Reasoning and Revision in Hybrid Representation Systems*, Vol. 422 *Lecture Notes in Artificial Intelligence*. Springer-Verlag, Berlin, Heidelberg, New York.

B. Nebel (1990b), Terminological reasoning is inherently intractable. *Artificial Intelligence*, 43:235–249.

B. Nebel (1991), Belief revision and default reasoning: Syntax-based approaches. In: J. A. Allen, R. Fikes, and E. Sandewall (Hrsg.), *Principles of Knowledge Representation and Reasoning: Proceedings of the 2nd International Conference*, S. 417–428, Cambridge, MA. Morgan Kaufmann.

B. Nebel (1994), Base revision operations and schemes: Semantics, representation, and complexity. In: *Proceedings of the 11th European Conference on Artificial Intelligence*, Amsterdam, Niederlande. Wiley. Erscheint demnächst.

K. Nökel (1989), Convex relations between time intervals. In: J. Rettie and K. Leidlmair (Hrsg.), *Proceedings der 5. Österreichischen Artificial Intelligence-Tagung*, S. 298–302. Springer-Verlag, Berlin, Heidelberg, New York.

K. Nökel (1991), *Temporally Distributed Symptoms in Technical Diagnosis*, Vol. 517 *Lecture Notes in Artificial Intelligence*. Springer-Verlag, Berlin, Heidelberg, New York.

P. Orponen, D. A. Russo und U. Schöning (1986), Optimal Approximations and Polynomially Levelable Sets. *SIAM Journal on Computing*, 15(2): 399–408.

C. H. Papadimitriou und M. Sideri (1992), On finding extensions of default theories. In: *Proc. International Conference in Database Theory*, S. 276–281. Springer-Verlag.

C. H. Papadimitriou (1994), *Computational Complexity*. Addison-Wesley, Reading, MA.

P. F. Patel-Schneider (1989), A four-valued semantics for terminological logics. *Artificial Intelligence*, 38(3):319–351.

R. Reiter und A. K. Mackworth (1989), A logical framework for depiction and image interpretation. *Artificial Intelligence*, 41:125–155.

R. Reiter (1987), A theory of diagnosis from first principles. *Artificial Intelligence*, 32(1):57–95.

E. Rich und K. Knight (1991), *Artificial Intelligence*. McGraw-Hill, New York, NY, 2nd edition edition.

C. K. Riesbeck und R. C. Schank (1989), *Inside Case-Based Reasoning*. Erlbaum, Hillsdale, NJ.

J. G. Schmolze und W. A. Woods (1992), The KL-ONE family. In: F. Lehmann (Hrsg.), *Semantic Networks in Artificial Intelligence*. Pergamon Press.

B. Selman und H. A. Kautz (1991), Knowledge compilation using Horn approximations. In: *Proceedings of the 9th National Conference of the American Association for Artificial Intelligence*, S. 904–909, Anaheim, CA. MIT Press.

B. Selman und H. J. Levesque (1993), The complexity of path-based defeasible inheritance. *Artificial Intelligence*, 62:303–339.

B. Selman, H. J. Levesque und D. Mitchell (1992), A new method for solving hard satisfiability problems. In *Proceedings of the 10th National Conference of the American Association for Artificial Intelligence*, S. 440–446, San Jose, CA. MIT Press.

B. Selman (1994), Domain-specific complexity tradeoffs. In: *Proceedings of the 11th European Conference on Artificial Intelligence*, S. 416–420, Amsterdam, Niederlande. Wiley.

F. Song und R. Cohen (1988), The interpretation of temporal relations in narrative. In *Proceedings of the 7th National Conference of the American Association for Artificial Intelligence*, S. 745–750, Saint Paul, MI.

J. Stillman (1992), The complexity of propositional default logics. In: *Proceedings of the 10th National Conference of the American Association for Artificial Intelligence*, S. 794–799, San Jose, CA. MIT Press.

D. S. Touretzky, J. F. Horty und R. H. Thomason (1987), A clash of intuitions: The current state of nonmonotonic multiple inheritance systems. In: *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, pages 476–482, Mailand. Morgan Kaufmann.

M. B. Vilain und H. A. Kautz (1986), Constraint propagation algorithms for temporal reasoning. In *Proceedings of the 5th National Conference of the American Association for Artificial Intelligence*, S. 377–382, Philadelphia, PA.

M. B. Vilain, H. A. Kautz und P. G. van Beek (1989), Constraint propagation algorithms for temporal reasoning: A revised report. In: D. S. Weld and J. de Kleer (Hrsg.), *Readings in Qualitative Reasoning about Physical Systems*, S. 373–381. Morgan Kaufmann, San Mateo, CA.

D. Waltz (1975), Generating semantic descriptions from drawings of scenes with shadows. In: P. Winston (Hrsg.), *The Psychology of Computer Vision*. McGraw-Hill, New York, NY.