5. Expertensysteme

5.1. Allgemeine Eigenschaften von Expertensystemen

5.1.1. Struktur von Expertensystemen

Probleme, zu deren Lösung Expertensysteme besonders geeignet sind, lassen sich durch zwei Eigenarten charakterisieren:

- Sie haben einen hohen Grad an Indeterminiertheit. Daraus ergibt sich die Gefahr einer kombinatorischen Explosion, die eine vollständige und exakte Lösung aus Zeit- und Platzgründen verbietet.
- Es liegen keine mathematisch klar umrissene Strukturen vor. Die Inhalte stammen oft aus dem alltäglichen Leben und sind entsprechend unzulänglich, unvollständig, schlecht strukturiert, unsicher, manchmal sogar inkonsistent und falsch. Trotzdem können Menschen auch für solche Probleme oft brauchbare Lösungen liefern. Ähnliches erwartet man auch von Expertensystemen.

Expertensysteme behandeln solche Probleme dadurch, dass sie das über eine Situation vorhandene Wissen explizit *deklarativ* repräsentieren. Mögliche Arten des Wissens sind:

- Wissen über elementare Tatsachen und die Art und Weise, wie Schlussfolgerungen gezogen werden,
- Wissen über spezielle Situationen,
- Wissen über Vorgehensweisen, Algorithmen und Kontrollstrukturen,
- Wissen über allgemeine Gesetzmäßigkeiten und Alltagswissen.

Wegen der zweiten Eigenart der zu lösenden Probleme ist es schwierig oder fast unmöglich, sie mit einem herkömmlichen Programmieransatz zu lösen, d.h. es ist schwer klare Algorithmen zu formulieren. Es ist aber bei diesen Problemen häufig möglich, das verfügbare Wissen aufzuschreiben und daraus neues Wissen schrittweise herzuleiten. Das geschieht im Wesentlichen in Expertensystemen, d.h. sie leisten eine Transformation der Eingabe in eine Ausgabe, die wie in Tabelle 5.1 dargestellt zu beschreiben ist.

Eingabe	$\Rightarrow$	Ausgabe
komplex		einfach
unstrukturiert		wohlstrukturiert
vage		exakt
unvollständig		vollständig
fehlerbehaftet		richtig
inkonsistent		konsistent

Tabelle 5.1

Die Lösung des Transformationsproblems ist meist durch die Beschreibung der Architektur eines Expertensystems gegeben. Auf das Wesentliche reduziert bestehen Expertensysteme aus zwei Komponenten:

- Wissensbasis

- Inferenzsystem

Das Wissen ist in der Wissensbasis in Form geeigneter Datenstrukturen abgelegt, In Frage kommen dafür Fakten (z.B. atomare Sätze), Regeln (implikative Sätze), Frames (objektartige Strukturen) oder Constraints. Das Inferenzsystem verarbeitet diese Strukturen mit geeigneten Methoden, z.B. logischen Inferenzen, Vererbungsmechanismen oder Constraintpropagierung. Das Wissen wird in einem Expertensystem auf verschiedene Arten repräsentiert, entsprechend der folgenden Einteilung:

- *Explizites* Wissen: Einträge in die Datenstrukturen.
- *Implizites* Wissen: Vom System erzeugtes und in die Datenstrukturen eingetragenes Wissen.
- *Direktes* Wissen: Explizites und implizites Wissen zusammengefasst.
- *Indirektes* Wissen: Wissen über die Datenstrukturen und die Funktionsweise des gesamten Systems, das nicht direkt ist. Dazu gehören die Auswahl der Darstellungsweise, Abbruchkriterien von Verfahren oder Auswahl von Algorithmen.

Inhaltlich lässt sich Wissen in Objektwissen, d.h. Wissen über die Gegenstände eines Problembereichs, und Kontrollwissen, d.h. Wissen über die Art der Lösung und Abarbeitung, unterteilen. Das Kontrollwissen ist seiner Natur nach Metawissen im Verhältnis zum Objektwissen, es entspricht dem indirekten Wissen. Die Lösung eines Problems, sofern es überhaupt lösbar ist, ist implizit im System bereits vorhanden. Die Aufgabe besteht dann nur noch darin, aus dem vorgegebenen expliziten und impliziten Wissen unter Benutzung des indirekten Wissens die Lösung in expliziter Form zu erzeugen.

5.1.2. Entwurf von Expertensystemen

Beim Entwurf eines Expertensystems besteht die Hauptaufgabe des Entwicklers darin, das gesamte Wissen in direktes und indirektes Wissen und das direkte Wissen in explizites und implizites Wissen aufzuteilen. Wie das geschieht, hängt von der Ausdruckskraft der verwendeten Sprache und vom allgemeinen Verständnis des Problems ab. Für gut verstandene Probleme mit vollständiger Information verwendet man am besten optimale Verfahren, die hauptsächlich mit indirektem Wissen arbeiten, ansonsten ist explizites Wissen erforderlich. Dann besteht die Aufgabe darin, möglichst viel Wissen in das System hineinzubringen.

Der Entwurf von Expertensystemen lässt sich, ähnlich wie der anderer Softwaresysteme, in drei Schritte gliedern:

1. Informelle Beschreibung der Anforderungen. Dazu gehört wesentlich die Wissensakquisition. Das Wissen, das man dabei erhält ist gewöhnlich unvollständig, weshalb dieser Schritt im Prinzip nie ganz abgeschlossen ist. Es muss die Möglichkeit zu späteren Ergänzungen des Wissens gegeben sein.
2. Überführung der Anforderungen in eine formale Spezifikation. Diese legt fest, was das System tun soll, sie ist also die funktionale Beschreibung des Systems.
3. Realisierung der Spezifikation durch eine Architektur.

Für die Implementierung von Expertensystemen kann man verschiedene Ebenen unterscheiden, wie in Abbildung 5.1 dargestellt.

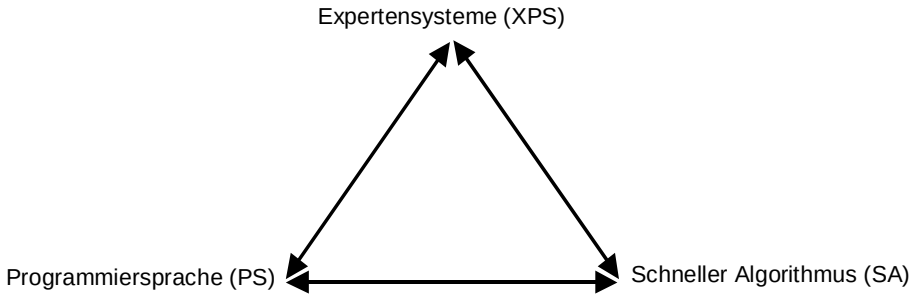

Abbildung 5.1

Die beiden Ebenen in Expertensystemen, die XPS-Ebene und die PS-SA-Ebene, lassen sich dadurch charakterisieren, dass die Aufgaben in XPS in einer intelligenten, flexiblen aber relativ langsamen Sprache formuliert sind, auf der PS-SA-Ebene aber schnelle Algorithmen vorliegen, die viele uniforme Einzelschritte in kurzer Zeit ausführen können. Eine Möglichkeit der Arbeitsteilung zwischen den Ebenen, bei der die Effizienz von SA erhalten bleibt, ist die, SA nicht die ursprüngliche Aufgabe zu übergeben, sondern eine modifizierte, aber uniformere. Die Verwendung von Entwurfswerkzeugen, wie sie auch bei der Entwicklung anderer Software-Systeme üblich ist, unterstützt diese Vorgehensweise.

Es gibt zwei Hauptklassen von Entwurfs-Werkzeugen:

(a) Werkzeuge für den Top-Down-Entwurf,
(b) elementare Low-Level Werkzeuge, die bestimmte Aufgaben vollständig übernehmen.

Zur Unterstützung des Top-Down-Entwurfs sucht man bei einer Klasse von Expertensystemen nach allgemeinen Entwurfsschritten, die für alle Systeme der Klasse annähernd gleich sind, und bildet aus ihnen eine formale Hülle, die dann nur noch mit Inhalt aufgefüllt werden muss. Diese Hüllen gibt es, sie sind kommerziell verfügbar und werden *Shells* genannt. Elementare Werkzeuge arbeiten auf der unteren Ebene. Sie sind nur durch ihr äußeres Verhalten gekennzeichnet und können dort, wo ihre Verhaltensweise benötigt wird, eingesetzt werden. Beispiele für solche Werkzeuge sind Verfahren zur heuristischen Suche oder Interpreter für das Matchen zeitlicher Verläufe in den Bedingungen von Regeln. Die Werkzeuge werden meist in einem *Werkzeugkasten* gesammelt bereitgestellt.

Stehen eine Shell und ein Werkzeugkasten zur Verfügung, dann besteht die Aufgabe der Entwicklung eines Expertensystems darin, die Werkzeuge im durch die Shell vorgegebenen Rahmen so einzusetzen, dass die gewünschte Lösung erreicht wird. Dieser Prozess wird XPS-Programmierung genannt. Vgl. dazu Abbildung 5.2.

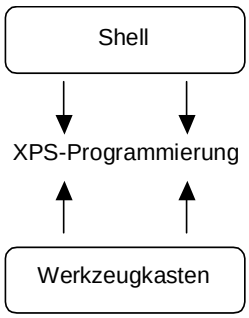

Abbildung 5.2

5.1.3. Bewertungskriterien

Mögliche Kriterien zur Bewertung von Expertensystemen lassen sich in zwei Gruppen unterteilen, die strukturellen Kriterien und die Erfolgskriterien. Für die *strukturellen Kriterien* kann man sich an den drei Schritten beim Systementwurf orientieren: Vorgegebene Anforderungen, formale Spezifikation und Realisierung durch die Architektur, vgl. Abschnitt 5.1.2. Daraus ergeben sich als Bewertungskriterien:

1) Inwieweit entspricht die Spezifikation der Aufgabenbeschreibung?
2) Welche Relationen bestehen zwischen der Spezifikation und den Architekturmerkmalen?
3) Sind die Realisierungsmittel dem Problem angemessen?
4) Welche Schwierigkeiten bei der Realisierung haben zu welchen Kompromissen bei der Erfüllung der ursprünglichen Anforderung geführt?

Die Erfolgskriterien hängen von der Art des erstellten Systems ab. Es gibt allgemeine Anforderungen, die auch an andere Softwareprodukte gestellt werden, und Anforderungen an die Ausgabe des Systems:

1) Expertensysteme sollen praktisch eingesetzt werden, deshalb erwartet man wie bei anderen Softwareprodukten Schnelligkeit, Sicherheit und Benutzerfreundlichkeit. Die Sicherheit kann man nur bedingt garantieren, denn die Verifikation derart komplexer Systeme ist nicht befriedigend gelöst. Die Benutzerfreundlichkeit spielt bei Expertensystemen eine besonders große Rolle wegen der prinzipiellen Unvollständigkeit des Wissens, deshalb sollte man hier auf eine entsprechende Funktionalität achten.

2) Die Beurteilung der Qualität der Ausgabe hängt von ihrer Art ab. Bei Systemen, die bei der Erstellung von Produkten oder beim Fällen von Entscheidungen eingesetzt werden, kann man Begriffe aus dem Bereich der Nutzenfunktionen und der Entscheidungstheorie verwenden. Wird eine Prognose oder Diagnose erstellt, dann hängt die Qualität der Ausgabe von der Trefferquote für die richtige Aussage ab; diese kann man mit Hilfe statistischer Stichprobenverfahren schätzen.

5.2. Behandlung von Unsicherheit und Vagheit

Aussagen im täglichen Leben und so auch in Expertensystemen, die Ausschnitte daraus abbilden, sind nur ganz selten entweder ganz wahr oder ganz falsch. Solche Aussagen gibt es typischerweise in der Mathematik und fast nur dort, aber selbst in der Mathematik sind viele zur Lösungsfindung angestellte Überlegungen unscharf.

Es gibt verschiedene Arten von Unschärfe, deshalb wird sich kein schlauer Kalkül finden lassen, der alle anfallenden Probleme behandeln kann. Statt dessen werden hier die betreffenden Phänomene in gewisse Klassen eingeteilt, die sich nach ihrem Charakter unterscheiden lassen, und es werden einige spezielle Methoden behandelt, die zur Lösung eingesetzt werden können. Zunächst kann man unscharfe Begriffe und Aussagen in zwei Gruppen unterteilen:

(a) *Subjektive* Unschärfe oder *Vagheit*,
(b) *Objektive* Unschärfe oder *Unsicherheit*.

Die subjektive Unschärfe ist in den einzelnen Personen begründet, sie kann aber unterschiedliche Ausprägungen haben, z.B.:

1) Sensorische Unsicherheit: Die Sinnesorgane der Menschen sind nicht normiert und reagieren individuell verschieden auf Reize.
2) Begriffliche und sprachliche Unsicherheit: Die Verwendung von Begriffen und sprachlichen Ausdrucksweisen ist häufig personengebunden und kontextabhängig, z.B. „sehr eindrucksvoll“ oder „ein typischer Italiener“.
3) Subjektive Wahrscheinlichkeit: Eine subjektive Einschätzung der Wahrscheinlichkeit eines Ereignisses, d.h. eine, die nicht auf eine statistische Häufigkeitsverteilung Bezug nimmt. Ein Beispiel ist: „Nach dem Examen mache ich wahrscheinlich eine Weltreise“.

Die objektive Unschärfe betrifft Aussagen, die eigentlich eine scharfe, präzise Interpretation haben, die aber aus verschiedenen Gründen nicht genau bekannt ist. Der Grund ist also ein Informationsmangel. Beispiele sind:

1) Messfehler und numerische Ungenauigkeiten,
2) statistische Aussagen,
3) Unkenntnis von Parametern und allgemeinen Zusammenhängen.

Unscharfe Äußerungen haben weitere Konsequenzen, und zwar indem sie die Grundlage für Handlungen oder Entscheidungen bilden oder in indem sie in Argumentationsketten verwendet werden. In beiden Fällen ist es wichtig zu wissen, bis zu welchem Grad eine unscharfe Aussage richtig ist, d.h. wie weit man sich auf sie verlassen kann. Man muss also einer solchen Aussage eine irgendwie geartete *Bedeutung* zuordnen, wie gewöhnlichen Aussagen, und dazu eine Art *Fehlerrechnung* betreiben. Das folgende Beispiel illustriert, dass dabei leicht Fehler gemacht werden können.

Beispiel

Es soll die Wirksamkeit zweier Medikamente *A* und *B* untersucht werden. Es werden drei Prädikate eingeführt: $Besser(x, y)$, $Besser_F(x, y)$ und $Besser_M(x, y)$. Dabei ist $x, y \in \{A, B\}$ und $Besser(x, y)$ bedeutet, dass man bei einer bestimmten Menge erwachsener Patienten mit x prozentual mehr Heilungserfolge als mit y gehabt hat, bei dem Prädikat $Besser_F(x, y)$ trifft dies nur auf die weiblichen Patienten und bei $Besser_M(x, y)$ nur auf die männlichen Patienten zu. Die folgende Regel

$$Besser_F(x, y) \wedge Besser_M(x, y) \rightarrow Besser(x, y)$$

sieht plausibel aus, ist aber falsch, weil sie gegen elementare statistische Gesetze verstößt, wie das folgende Zahlenbeispiel zeigt. + bedeutet dabei Erfolg, – bedeutet Misserfolg.

Männer:

	+	–	
A	20	180	10% Erfolge
B	4	96	4% Erfolge

$\Rightarrow Besser_M(A, B)$

Frauen:

	+	–	
A	20	20	50% Erfolge
B	37	63	37% Erfolge

$\Rightarrow Besser_F(A, B)$

Gesamt:

	+	–	
A	40	200	16,6% Erfolge
B	41	159	20,5% Erfolge

$\Rightarrow Besser(B, A)$

Wie man sieht, ist also eine genaue Festlegung der Bedeutung unscharfer Aussagen als auch die Überprüfung ihrer Verwendung in Schlussketten erforderlich. Denkbar hierfür sind Hilfsmittel aus der Intervallrechnung und der Statistik. Die wichtigsten zu klärenden Fragen sind:

1) Was ist die Bedeutung der unscharfen Aussage *H*?
2) Welcher Aspekt oder welcher Teil von *H* ist wahr?
3) Welche Handlungsanweisung ist mit *H* verbunden?
4) Welchen Fehler enthält *H* und welche Methode zur Fehlerverfolgung und Fehlererkennung ist bei der Weiterverwendung von *H* zu beachten?

Ein einfaches Fehlermodell für die Fehlerverfolgung ist das folgende: Man geht von zwei Mengen *A* und *B* und einer Abbildung $f: A \rightarrow B$ aus. *A* enthalte möglicherweise fehlerbehaftete Informationen, die mittels *f* nach *B* übertragen werden. Ist $a \in A$, dann bezeichne $A(a)$ die Menge der $x \in A$, die wegen der Fehlerhaftigkeit anstelle von *a* erscheinen können. Die Elemente $x \in A(a)$ können also mit *a* verwechselt werden, sie sollen deshalb Kandidaten für *a* genannt werden. Nun wird definiert:

$$b \in B \text{ ist } \textit{Kandidat für } f(a), \text{ falls } b \in \{y \mid y \in B \wedge \exists\, x \in A(a) \text{ mit } f(x) = y\}$$

Auf diese Weise wird der Fehler in *A* in einer „*f* gemäßen Weise“ von *A* nach *B* fortgepflanzt.

Im Folgenden werden verschiedene Methoden zur Behandlung von Unschärfe genauer betrachtet.

5.2.1. Die Methode der groben Mengen

Diese Methode kann auf subjektive wie auf objektive Unschärfen angewendet werden. Ausgangspunkt ist eine binäre Relation ≈ auf dem betrachteten Universum *U*, die *Ununterscheidbarkeitsrelation*. Sie soll reflexiv und symmetrisch sein. Die Idee der Definition von ≈ ist, dass die vorhandenen objektiven Informationen und subjektiven Kriterien nicht ausreichen um zwei Objekte *a* und *b* mit $a \approx b$ zu unterscheiden. Die Relation ≈ wird nun dazu verwendet, eine beliebige Teilmenge von *U* durch eine größere und eine kleinere Menge anzunähern.

Definition 5.1 (Grobe Menge)

Sei $P(x)$ ein Prädikat über *U*, d.h. $P(x)$ repräsentiert eine Teilmenge von *U*. Dann wird definiert:

i. Die untere Approximation von *P* ist die Menge $P_u = \{x \in U \mid \text{für alle } y \text{ mit } x \approx y \text{ gilt } y \in P\}$
ii. Die obere Approximation von *P* ist die Menge $P_o = \{x \in U \mid \text{es gibt ein } y \in P \text{ mit } x \approx y\}$

Abbildung 5.3 veranschaulicht die Beziehung zwischen den Mengen *P*, P_u und P_o.

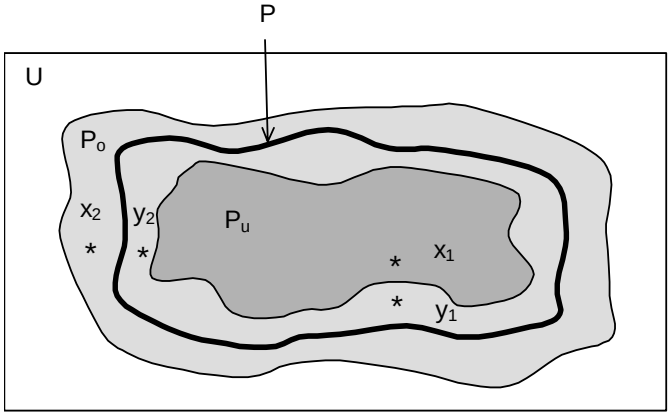

Abbildung 5.3

Die Elemente von P_u erfüllen also alle die Eigenschaft $P(x)$, die von P_o nur teilweise. Die Differenzmenge $\Delta P = P_o - P_u$ heißt *Rand* oder *Unsicherheit* von P. Auf Grund der Definition von P_u und P_o ergeben sich folgende Beziehungen:

1) $(\neg P)_u = \neg P_o$, $(\neg P)_o = \neg P_u$ daraus ergibt sich
   $P_u = \neg(\neg P)_o$, $P_o = \neg(\neg P)_u$
2) $(P \wedge Q)_u = P_u \wedge Q_u$
3) $(P \vee Q)_o = P_o \vee Q_o$
4) $(P \wedge Q)_o \subseteq P_o \wedge Q_o$
5) $P_u \vee Q_u \subseteq (P \vee Q)_u$

Fall (4) z.B. überlegt man sich folgendermaßen: $x \in (P \wedge Q)_o \Leftrightarrow$ es gibt ein y mit $x \approx y$ und $y \in P$ und $x \in P_o$ und $x \in Q_o$. Wenn umgekehrt $x \in P_o$ und $x \in Q_o$, dann weiß man nur, dass es y_1 und y_2 mit $y_1 \approx x \approx y_2$, was aber nicht für $x \in (P \wedge Q)_o$ ausreicht.

In (4) und (5) gelten im Allgemeinen sogar echte Inklusionen. (2) und (3) gelten auch für unendliche Operationen, sie können daher auch auf die entsprechenden Quantifizierungen übertragen werden. Für die Implikation ergibt sich

$$P_o \rightarrow Q_u \subseteq (P \rightarrow Q)_u \subseteq (P \rightarrow Q)_o = P_u \rightarrow Q_o$$

Damit erhält man eine Regel, die es erlaubt, die Unsicherheit fortzupflanzen. Ist die Relation $\approx$ auch noch transitiv, dann gilt

$$P_u = (P_u)_u \text{ und } P_o = (P_o)_o$$

Durch die Transitivität wird die Relation $\approx$ zu einer Äquivalenzrelation. Sie definiert auf U eine Partition in Klassen ununterscheidbarer Objekte. Die Abbildung, die jedem Objekt seine Klasse zuordnet, ist eine Abstraktionsabbildung. Die Abstraktion „vergisst“ diejenigen Beschreibungsmerkmale, über die sowieso nichts bekannt ist.

In vielen Fällen ist die Relation $\approx$ nicht transitiv. Im Allgemeinen gilt hier $P_u \neq (P_u)_u$ und $P_o \neq (P_o)_o$. Das ist typischerweise der Fall, wenn Objekte durch kleine Abweichungen ununterscheidbar werden, z.B. durch kleine numerische Unterschiede. Solche kleinen Unterschiede können sich bei der Iteration zu merkbaren Unterschieden aufsummieren. Ein Beispiel dafür ist die Arithmetik in einem Computer, die eine vorgegebene Genauigkeitsschranke für die Zahlen hat. Bei einer komplexen Rechenoperation aus mehreren einzelnen Schritten hängt der Fehler, der sich ergibt, wesentlich von der Zahl der Schritte ab. Bei beliebiger Unschärfe lässt sich keine Fehlerschranke angeben, man ist dann gezwungen, sich auf kurze Schlussketten zu beschränken.

5.2.2. Wahrscheinlichkeiten

Wahrscheinlichkeiten dienen zur Repräsentation objektiver Unschärfen, zumindest soweit die Wahrscheinlichkeitsaussagen statistische begründet sind. In der Statistik hat der Begriff der *Häufigkeit* zentrale Bedeutung. Unter diesem Aspekt werden Aussagen in vielen Modellen betrachtet und Überlegungen über die asymptotische Trefferquote angestellt.

In einer realen Situation liegen Ereignisse (Hypothesen) $H_1, ..., H_n$ vor, von denen nicht bekannt ist, welche eingetreten sind. Die Wahrscheinlichkeit einer solchen Hypothese, so lange man nichts weiter über sie weiß, ist $P(H_i)$ und heißt *unbedingte* oder *a priori*-Wahrscheinlichkeit. Eine Beobach-

tung E (*Evidenz*) kann ein Hinweis auf die H_1, ..., H_n sein. Mit ihr wird aus der unbedingten eine bedingte oder a posteriori-Wahrscheinlichkeit, geschrieben $P(H_i \mid E)$. Der Zusammenhang zwischen unbedingten und bedingten Wahrscheinlichkeiten wird durch folgende Beziehung ausgedrückt:

$$P(A|B) = \frac{P(A \wedge B)}{P(B)}$$

Aus dieser Definition lässt sich sehr einfach die so genannte Bayessche Regel ableiten, die für die Wahrscheinlichkeitstheorie von grundlegender Bedeutung ist.

Satz 5.1 (Satz von Bayes)

Sei H_1, ..., H_n eine Partition des gesamten Ereignisraums und E ein Ereignis mit $P(E) > 0$ und $P(H_n) > 0$ für $i = 1, ..., n$. Dann gilt für alle i, $i = 1, ..., n$:

$$P(H_i|E) = \frac{P(H_i)P(E|H_i)}{\sum_{j=1}^{n} P(H_j)P(E|H_i)}$$

Der Nutzen der Bayesschen Regl ist, dass man die bedingte Wahrscheinlichkeit der H_i bei gegebenem E berechnen kann. Dazu werden die bedingten Wahrscheinlichkeiten $P(E|H_i)$ und die unbedingten Wahrscheinlichkeiten der H_i benötigt, aber nicht mehr die unbedingte Wahrscheinlichkeit von E. Das folgende Beispiel illustriert den Gebrauch der Bayesschen Regel.

Ein Reisender steht auf dem Bahnsteig und wartet auf den Zug. Bezüglich der Pünktlichkeit des Zugs gibt es zwei Alternativen:

H_1: Der Zug kommt zu spät.
H_2: Der Zug ist pünktlich.

Die a-priori-Wahrscheinlichkeiten seien $P(H_1) = p$ und $P(H_2) = 1 - p$. Weiterhin sei die Zugansage Z gegeben. Ihre Wahrscheinlichkeiten sind $P(Z = \textit{zuverlässig}) = 0.9$ und $P(Z = \textit{unzuverlässig}) = 0.1$. Ist die Ansage zuverlässig, dann ist die Information I richtig, ist sie unzuverlässig, dann kann sie richtig oder falsch sein. Es soll gelten

$$P(I = \textit{richtig} \mid Z = \textit{unzuverlässig}) = q$$

Das Ereignis E sei die Ansage „Der Zug hat Verspätung." Die zu beantwortende Frage ist, ob unter der Bedingung E das Ereignis H_1 eintritt. Die bedingten Wahrscheinlichkeiten $P(E \mid H_i)$ sind

$$P(E \mid H_1) = P(Z = \textit{zuverlässig}) + P(Z = \textit{unzuverlässig}) \cdot q = 0.9 + 0.1 \cdot q$$
$$P(E \mid H_2) = P(Z = \textit{unzuverlässig}) \cdot (1 - q) = 0.1 \cdot (1 - q)$$

Die Bayessche Regel ergibt

$$P(H_1|E) = \frac{0.9p + 0.1pq}{0.9p + 0.1pq + 0.1(1-p)(1-q)}$$

5.2.3. Die Evidenztheorie von Dempster und Shafer

Gegeben sei eine Menge $\mathbf{A} = \{A_1, ..., A_n\}$ von Alternativen. Für **A** soll gelten:

1) Die Menge der Alternativen ist vollständig.
2) Die Alternativen schließen sich gegenseitig aus.

Für die Alternativen gebe es gewisse Hinweise (genannt *Evidenzen*). Ganze Gruppen von Alternativen können eine Evidenz haben, ohne dass die einzelnen Alternativen der Gruppe besonders ausgezeichnet sind. Zum Beispiel kann über eine Gruppe von Personen aus einer größeren Zahl von Personen bekannt sein, dass sie die Blutgruppe B haben, wodurch sie sich von den anderen unterscheiden, aber untereinander nicht. Zur Präzisierung des Verständnisses von Evidenz wird die folgende Definition verwendet. $2^{\mathbf{A}}$ ist dabei die Potenzmenge von **A**.

Definition 5.2 (Basismaß)

Ein *Basismaß* für eine Menge $\mathbf{A} = \{A_1, ..., A_n\}$ von Alternativen ist eine Abbildung

$$m: 2^{\mathbf{A}} \rightarrow [0, 1]$$
$$\text{mit } m(\varnothing) = 0 \text{ und } \sum_{X \subseteq \mathbf{A}} m(X) = 1$$

Das nichtssagende Basismaß m_0 ist durch $m_0(\mathbf{A}) = 1$ und $m_0(X) = 0$ für $X \neq \mathbf{A}$. m_0 gibt die leere Information wieder. Es kann bekannt sein, dass die richtige Alternative in der Menge X liegt und eventuell sogar genauer lokalisiert werden kann. Dann spricht man von *akkumulierter Evidenz*. Sie wird mit Hilfe einer Funktion B (belief) modelliert:

$$B: 2^{\mathbf{A}} \rightarrow [0, 1]$$
$$\text{mit } B(X) = \sum_{Y \subseteq X} m(Y)$$

Ist m das nichtssagende Basismaß, dann ist $B = m$. Dieses B heißt nichtssagende B-Funktion. Man kann Funktionen der Art von B auch ohne Bezug auf m definieren. Zu jeder Funktion B mit den Eigenschaften

1) $B(\varnothing) = 0$
2) $B(\mathbf{A}) = 1$
3) $B(X_i \cup X_j) \geq B(X_i) + B(X_j) - B(X_i \cap X_j)$ für alle $X_i, X_j \subseteq \mathbf{A}$

kann man das zugehörigen Basismaß ausrechnen, z.B. ist $m(\{A_1, A_2\}) = B(\{A_1, A_2\}) - B(\{A_1\}) - B(\{A_2\})$.

Definition 5.3 (Zweifel, obere Wahrscheinlichkeit, Fokalität)

i. Der *Zweifel* an X ist $Zw(X) = B(\mathbf{A}-X)$.
ii. Die *obere Wahrscheinlichkeit* von X ist $B^*(X) = 1 - Zw(X)$.
iii. X ist *fokal*, falls $m(X) > 0$; die Vereinigung aller fokalen Elemente heißt der *Kern* von B.

Die Differenz zwischen $B(X)$ und $B^*(X)$ kennzeichnet die Unwissenheit über X. Die fokalen Mengen sind die einzigen, die zu $B(X)$ einen positiven Wert beitragen, $B(X) = 1$ genau dann, wenn X den Kern enthält. Gilt für eine B-Funktion, dass $B(X)$ und $B^*(X)$ identisch sind, dann folgt aus Definition 5.3: $B(X) + B(\mathbf{A}-X) = 1$ für alle $X \subseteq \mathbf{A}$. In diesem Fall gilt in Gleichung (3) das Gleichheitszeichen statt ≥. Man nennt solche Funktionen *Bayes-B-Funktionen*.

Inferenzen in einem Schlussfolgerungssystem mit Aussagen, die mit Evidenzen versehen sind, können von zweierlei Art sein:

- Akkumulierung mit anderen Evidenzen,
- Verwendung in Regeln, die entweder ganz sicher oder auch mit einer Evidenz versehen sind.

Seien zwei Evidenzen über der Menge **A** durch ihre Basismaße m_1 und m_2 gegeben. Da **A** nach Voraussetzung nur sich gegenseitig ausschließende Alternativen enthält ist eine erster Ansatz zur Definition der Akkumulation der beiden Evidenzen:

$$m_1"+"m_2(H) = \sum_{H=X\cap Y} m_1(X)\cdot m_2(Y)$$

Allerdings kann hierbei der Fall eintreten, dass $m_1(X) > 0$ und $m_2(Y) > 0$ aber gleichzeitig $X \cap Y = \varnothing$, d.h. die akkumulierte Evidenz auf der leeren Menge wäre positiv. Um diesen Effekt zu vermeiden, wird die Summe modifiziert. Es wird die direkte Summe $\oplus$ verwendet und nach *Dempsters Regel* berechnet:

$$m_1 \oplus m_2(\varnothing) = 0$$
$$m_1 \oplus m_2(H) = m_1"+"m_2 \cdot \frac{1}{1-K} \quad \text{für } H \neq 0$$
$$\text{wobei} \quad K = \sum_{X\cap Y=\varnothing} m_1(X)\cdot m_2(Y)$$

Definition 5.4 (Unvereinbarkeit von Evidenzen)

m_1 und m_2 heißen *unvereinbar*, wenn $\sum_{X\cap Y=\varnothing} m_1(X)\cdot m_2(Y) = K = 1$.

Für unvereinbare Evidenzen wird festgelegt, dass ihre Akkumulation nicht definiert ist. Der Wert $Kon(m_1, m_2) = -\log(1-K)$ heißt das Gewicht des Konflikts zwischen m_1 und m_2. Falls sich m_1 und m_2 nicht im Konflikt befinden, ist $Kon(m_1, m_2) = 0$, falls sie unvereinbar sind, ist $Kon(m_1, m_2) = \infty$.

Für das Schlussfolgern unter Verwendung der Evidenzen, d.h. für ihr Propagieren über Regeln hinweg, braucht man eine Darstellungsform für die Regeln, so dass sich die Evidenzen von Regeln und Aussagen kombinieren lassen. Zu diesem Zweck betrachtet man Regeln als Elemente des kartesischen Produkts zwischen Prämissen und Konklusionen, die beide Aussagen darstellen. Sind $\mathbf{A} = \{A_1, ..., A_n\}$ und $\mathbf{B} = \{B_1, ..., B_k\}$ zwei Alternativenmengen. Ein Basismaß m auf **A** wird auf $\mathbf{A} \times \mathbf{B}$ erweitert, indem für $X \subseteq \mathbf{A} \times \mathbf{B}$ festgesetzt wird:

$$m_0(X) = m(\pi_1(X))$$

$\pi_1(X)$ ist die Projektion von X auf **A**, d.h. der **A**-Anteil von X liefert die Evidenz der Prämisse. Die Idee der Propagierung besteht nun darin, die Evidenz von **A** mittels Dempsters Regel mit einer Evidenz für die Regel zu akkumulieren und die akkumulierte Evidenz auf die Alternativenmenge **B** zu projizieren. Die Evidenzpropagierung hat also die Form

WENN	Vorbedingung V mit Evidenz E_1
UND	Regel R mit Evidenz E_2
DANN	Konklusion mit Evidenz E_3

Fügt man diese Evidenzpropagierung zu einer Regelmenge hinzu, dann erhält man ein Schlussfolgerungssystem, das Evidenzen verarbeiten kann. Eine Einschränkung der Evidenztheorie von Dempster und Shafer ist, dass vollständige Mengen sich gegenseitig ausschließender Alternativen gegeben sein müssen. Um die Theorie anwenden zu können, muss diese Bedingung wenigstens annähernd erfüllt sein.

5.2.4. Fuzzy-Werte und Fuzzy-Wahrscheinlichkeiten

Die grundlegende Idee des Fuzzy-Ansatzes ist die Bewertung subjektiver Vagheit, die durch eine Abbildung in das Intervall [0, 1] modelliert wird. Die subjektiven Einschätzungen müssen aber sauber von den objektiven Gegebenheiten unterschieden werden. Auch bei den subjektiven Vagheiten gibt es verschiedene Typen, die zu unterscheiden sind.

Sei $P(x)$ ein einstelliges Prädikat, definiert über dem Universum U. Sei $A = \{a \in U \mid a \text{ erfüllt } P(x)\}$. Die Menge A lässt sich auch durch ihre *charakteristische Funktion* repräsentieren, die folgendermaßen definiert ist:

$$I_A: \quad U \to \{0,1\}$$
$$I_A(x) = \begin{cases} 1 & \text{falls } x \in A \\ 0 & \text{sonst} \end{cases}$$

Will man das Prädikat nicht ganz scharf definieren, d.h. wenn nicht für alle Elemente aus U ganz klar ist, ob sie P erfüllen oder nicht oder wenn man ausdrücken will, dass sie es nur bis zu einem gewissen Grad tun, aber trotzdem mit P gewisse rationale Aspekte verbinden, dann kann man diese Aspekte durch eine verallgemeinerte charakteristische Funktion zu beschreiben versuchen.

Definition 5.5 (Fuzzy-Menge, Fuzzy-Interpretation)

(i) Eine *Fuzzy-Menge* über U ist eine Funktion

$$\mu: U \to [0, 1]$$

μ heißt auch Zugehörigkeits-Funktion.

(ii) Die *Fuzzy-Interpretation* eines Prädikats P über U ordnet P eine Fuzzy-Menge über U zu. Eine Belegung der Variablen x mit $a \in U$ erfüllt $P(x)$ mit dem Grad d, falls $\mu(a) = d$. Man sagt auch: Die Gültigkeit von $P(a)$ wird mit $\mu(a)$ akzeptiert.

Man schreibt manchmal auch μ_P für μ, um den Zusammenhang zwischen P und μ deutlich zu machen. Ein Beispiel für eine Fuzzy-Modellierung ist die Wiedergabe der Begriffe „hohes Fieber“ und „leichtes Fieber“. Diese Begriffe möchte man nicht scharf abgrenzen. Eine Fuzzy-Interpretation durch eine Zugehörigkeits-Funktion könnte z.B. wie in Abbildung 5.4 dargestellt aussehen.

Typisch für eine Modellierung dieser Art ist, dass es einen Bereich der Ambiguität gibt (zwischen 38 und 39). Beide Aussagen haben hier einen gewissen positiven Wert. Wenn aber nur eine dieser Aussagen sicher gemacht werden kann, dann soll die Auswahl dieser Aussage mit den Fuzzy-Mengen in Zusammenhang stehen. In einem solchen Fall wird die Rationalitätsannahme gemacht:

Rationalitätsannahme: Im Zweifelsfall wird die Äußerung mit dem höchsten Akzeptanzgrad gemacht.

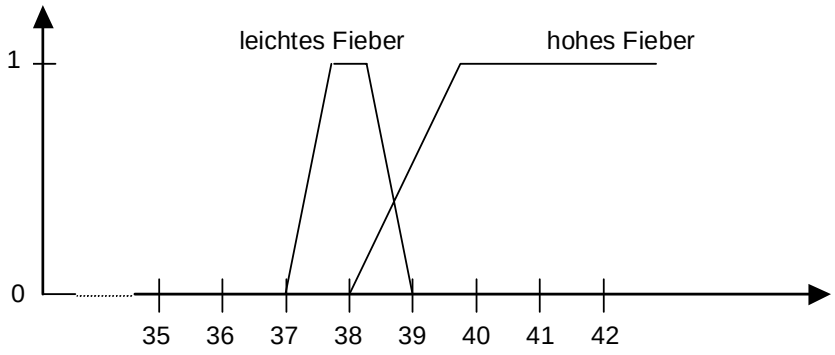

Abbildung 5.4

Es ist nicht sinnvoll völlig beliebige Funktionen zur Modellierung unscharfer Aussagen zu verwenden, vielmehr sollte man einige einschränkende Bedingungen für die Definition solcher Funktionen machen.

Definition 5.6 (Reguläre Zugehörigkeits-Funktion)

Für $T \subseteq \Re$ heißt μ: $T \rightarrow [0, 1]$ *regulär*, falls gilt:

i. μ ist stückweise stetig.
ii. Wenn $x, y \in T$ und $\lambda \in [0, 1]$, dann ist $\mu(\lambda x + (1 - \lambda)y) \geq \min(\mu(x), \mu(y))$.
iii. Es gibt ein $x \in T$ mit $\mu(x) = 1$.

Fasst man die Fuzzy-Werte als verallgemeinerte Wahrheitswerte auf, dann kann man fragen, wie sich solche Wahrheitsfunktionen auf zusammengesetzte Aussagen fortsetzen lassen, d.h. wie berechnet sich der Wahrheitswert einer Konjunktion, Disjunktion, Negation usw. aus den Werten der einzelnen Komponenten? Eine rein mathematische Definition erscheint hier ungeeignet, vielmehr sollte sie sich nach der intendierten Bedeutung der Aussagen richten. Bei einer Konjunktion oder Disjunktion z.B. sollten beide Teile der Aussage je nach Fragestellung mit ihren individuellen Gewichten in die Entscheidung eingehen. Dazu wird zunächst eine relativ einfache Methode der Berechnung eingeführt und anschließend ihre Eigenschaften untersucht. Für die Definition werden drei Funktionen k, d, n: $[0, 1] \rightarrow [0, 1]$ (für Konjunktion, Disjunktion und Negation) benötigt.

$$k(x, y) = \min(x, y)$$
$$d(x, y) = \max(x, y)$$
$$n(x) = 1 - x$$

In der zweiwertigen Logik sind dies genau die üblichen Wahrheitswertfunktionen. k und d müssen folgende Eigenschaften haben:

1) k und d sind kommutativ.
2) k und d sind assoziativ.
3) Zwischen k und d gelten die Distributivgesetze, z.B. $k(x, d(y, z)) = d(k(x, y), k(x, z))$.
4) Die Funktionen $k_y(x) = k(x, y)$ und $d_y(x) = d(x, y)$ für $y \in [0, 1]$ sind stetig und monoton.
5) $k(x, x)$ und $d(x, x)$ sind streng monotone Funktionen.
6) $k(x, y) \leq \min(x, y)$ und $d(x, y) \geq \max(x, y)$.
7) $k(1, 1) = 1$ und $d(0, 0) = 0$.

Die Funktion $k = \min$ und $d = \max$ erfüllen trivialerweise die Bedingungen. Es gilt aber auch die Umkehrung:

Satz 5.2 (Eigenschaften der Funktionen *k* und *d*)

Wenn *k* und *d* die Bedingungen 1) – 7) erfüllen, dann ist $k(x, y) = \min(x, y)$ und $d(x, y) = \max(x, y)$.

Die Eigenschaften der Bedingungen 1) – 7) werden durch die folgenden Betrachtungen charakterisiert:

Bei Bedingung (1) besteht das Problem, dass *x* und *y* nicht vertauscht werden können, wenn sie sehr unterschiedliche Einflussgrößen darstellen, denn dann kann der eine Wert eine größere Rolle spielen als der andere. Häufig drückt das Wort „und“ eine Reihenfolge aus im Sinne von „erst das eine, dann das andere“, auch dann können die Werte nicht vertauscht werden. Ähnliche Überlegungen gelten für die Bedingungen (2) und (3). Bedingung (4) als Forderung nach Stetigkeit schließt Schwellenwertbetrachtungen aus, an denen Sprünge stattfinden. Die Bedingungen (5) und (7) erscheinen sinnvoll, dagegen bleibt Bedingung (6) zunächst unmotiviert.

Ein Problem stellt die Definition der Negation dar. Nach ihr ist die Summe des Wahrheitswerts einer Aussage und ihrer Negation genau 1. Bei der Evidenztheorie wollte man eigentlich genau diesen Effekt vermeiden. Die Definition hat nämlich eine unangenehme Konsequenz. Ist z.B. $x = 0.5$, dann ist auch $n(x) = 0.5$. Daraus folgt $k(x, n(x)) = 0.5$, d.h. ein offensichtlicher Widerspruch wird mit dem relativ hohen Wahrheitswert 0.5 belegt. Außerdem wäre es, selbst wenn man zwischen einer Aussage und ihrer Negation völlig unentschieden ist, nicht rational, beide gleichzeitig zu akzeptieren. Es ist deshalb schwer, eine allgemeine Vorschrift für die Berechnung des Wahrheitswerts der Negation anzugeben. Statt dessen sollte diese je nach Intention festgelegt werden.

In realen Situationen sind oft subjektive und objektive Unschärfen vermischt. Hier wird besonders der Fall betrachtet, dass sich subjektive Vagheit und statistische Unsicherheit, die aus strukturellen Eigenschaften des behandelten Bereichs herstammen, überlagern. Dabei ist die Frage von Bedeutung, welchen Anteil die beiden Unsicherheiten an einem Endergebnis haben, d.h. wie subjektiv oder objektiv dieses ist. Um diesen Zusammenhang zu modellieren wird das Konzept der *unscharfen Zufallsvariablen* verwendet. Man geht dazu von einem Ereignisraum Ω aus. Eine gewöhnliche Zufallsvariable ist eine Abbildung $\zeta: \Omega \rightarrow \Re$, bei einer unscharfen Zufallsvariablen wird statt dessen eine Abbildung benötigt, die keinen genauen Zahlenwert, sondern eine Fuzzy-Menge über $\Re$ liefert. Dazu nimmt man an, dass $\Re$ in endliche viele Teilmengen unterteilt ist, z.B. in ein nach links halboffenes und ein nach rechts halboffenes und dazwischen eine endliche Anzahl von Intervallen. Zwei Ereignisse, die in dasselbe Intervall fallen, gelten als nicht unterscheidbar. Der Wert einer Zufallsvariablen *Y* wird dann als eine der partitionierenden Mengen *A* definiert:

$$Y(\omega) = A \text{ genau dann, wenn } \zeta(\omega) \in A$$

Definition 5.7 (Unscharfe diskrete Zufallsvariable)

Eine *unscharfe diskrete Zufallsvariable* X ist eine Funktion, die jedem $\omega \in \Omega$ eine reguläre Funktion (vgl. Def. 5.6) zuordnet. Als Definitionsbereich der regulären Funktionen wird eine einheitliche Menge $T \subseteq \Re$ zugrunde gelegt.

Statt $X(\omega)$ wird auch X_ω geschrieben. Für $x \in T$ ist $X_\omega(x)$ eine reelle Zahl. Als charakteristische Funktion von X_ω wird ein Intervall gewählt. Betrachtet man das Beispiel der Fiebermessung, dann ist dort das zufällige Ereignis ω der Patient und $\zeta(\omega)$ seine Temperatur. Legt man eine Unterteilung der Temperaturskala in die drei Intervalle „kein Fieber“, „leichtes Fieber“ und „hohes Fieber“ zugrunde, dann gibt das Anlass für eine mengenwertige Zufallsgröße $Y(\omega)$, wobei $\zeta(\omega)$ das *Original* dieser Zufallsgröße genannt wird. Sind die Begriffe unscharf, wie im betrachteten Beispiel, dann

bekommt man die Zufallsvariable $X(\omega)$, deren Wert eines der drei Intervalle ist. In diesem Fall ist das Original $\zeta(\omega)$ bekannt, also wird sich die Wertzuordnung für $X(\omega)$ entsprechend obiger Definition daran orientieren. Liegt ein exakter Wert nicht vor, sondern eine unscharfe Eigenschaft wie z.B. „Blässe“, dann gibt es kein scharfes Original. Man kann eine Fuzzy-Funktion $f(x)$ für „Blässe“ definieren, die folgende Bedeutung haben soll:

„Der Ausgeprägtheitsgrad X für die Blässe wird mit dem Grad $f(x)$ akzeptiert.“

Allgemein gesprochen wird mit X eine Akzeptanz $\alpha_{\zeta(\omega)}$ durch folgende Festlegung assoziiert:

$$\alpha_{\zeta(\omega)}: A_{\zeta(\omega)} \rightarrow [0, 1]$$
$$\alpha_{\zeta(\omega)}(a_{\zeta(\omega)}(x)) = X_\omega(x)$$

Dabei ist $a_{\zeta(\omega)}(x)$ die Aussage „x ist Ausgeprägtheitsgrad von ζ_ω“ (oder: „x ist Kandidat für ζ_ω“) und $A_{\zeta(\omega)}$ die Menge aller solcher Aussagen. Das Original $\zeta(\omega)$ muss dabei nicht vorliegen.

Ist ein unscharfes Symptom wie „Blässe“ ein Indiz für eine Krankheit K, die selbst wieder unscharf ist, d.h. ist „Blässe“ durch K bedingt, dann benötigt man zu seiner Darstellung ein unscharfes Analogon zur bedingten Wahrscheinlichkeit. Dazu wird für zwei Funktionen ξ und ζ die Akzeptanz der Konjunktion zweier Aussagen $a_{\xi(\omega)}(x) \wedge a_{\zeta(\omega)}(x)$ definiert:

$$\alpha_{\xi(\omega),\zeta(\omega)}(a_{\xi(\omega)}(x) \wedge a_{\zeta(\omega)}(x)) = \min\{\alpha_{\xi(\omega)}(a_{\xi(\omega)}(x)), \alpha_{\zeta(\omega)}(a_{\zeta(\omega)}(x))\}$$

Seien U und V zwei weitere Funktionen des Typs $\Omega \rightarrow \Re$. Dann bedeutet die Konjunktion der Aussagen

$$a_{\xi(\omega)}(U(\omega)) \wedge a_{\zeta(\omega)}(V(\omega))$$

für alle $\omega \in \Omega$ „U ist Kandidat für ξ und V ist Kandidat für ζ“, abgekürzt $a(U, V)$. Die Akzeptanz dieser Aussage ist

$$\alpha_{\xi(\omega),\zeta(\omega)}(a(U, V)) = \inf_{\omega\in\Omega} \min\{X_\omega(U(\omega)), X'_\omega(V(\omega))\}$$

Dabei ist ξ das Original von X und ζ das Original von X'. Sind $A, B \subseteq \Re$ mit $U \in A$ und $V \in B$, dann wird in Analogie zur üblichen bedingten Wahrscheinlichkeit definiert:

$$P(U \in A \mid V \in B) = \frac{P(U \in A \wedge V \in B)}{P(V \in B)}$$

wobei $P(V \in B) > 0$ vorausgesetzt ist. Die Aussage $a_{A|B}(x)$ wird erklärt als „x ist Kandidat für $P(\xi \in A \mid \zeta \in B)$“, die zugehörige Akzeptanzfunktion $\alpha_{A|B}$ wird definiert durch

$$\alpha_{A|B}(a_{A|B}(x)) = \sup\, \alpha_{\xi(\omega),\zeta(\omega)}(a(U, V) \mid P(U \in A \mid V \in B) = x)$$

wenn x im Bild der $P(U \in A \mid V \in B)$ liegt. Damit lässt sich nun die bedingte Fuzzy-Wahrscheinlichkeit wie folgt definieren:

Definition 5.8 (Bedingte Fuzzy-Wahrscheinlichkeit)

Die Abbildung $P_{A|B}$: $[0, 1] \rightarrow [0, 1]$ ist definiert durch

$$P_{A|B}(x) = \begin{cases} \alpha_{A|B}(\mathrm{a}(x)) & \text{falls } x \text{ im Bild der } P(U \in A \mid V \in B) \\ 0 & \text{sonst} \end{cases}$$

In dieser Definition von $P_{A|B}$ werden zwei Begriffe zur Behandlung von Unschärfe verwendet:

(a) Eine objektive Wahrscheinlichkeit, die man z.B. aus einer Häufigkeitstabelle bekommen kann: Wie oft wurde *A* behauptet unter der Voraussetzung, dass *B* bereits behauptet wurde?
(b) Eine subjektive Vagheit über individuelle Akzeptanzen.

5.2.5. Sicherheitsfaktoren

Die Sicherheitsfaktoren sind eine Zuordnung numerischer Werte zu Aussagen zusammen mit einem einfachen Mechanismus für ihre Propagierung in einem Regelsystem. Sie wird verwendet zur Darstellung objektiver Unsicherheit, wenn Anhaltspunkte für die Wahrscheinlichkeitsverteilung fehlen. Gegeben sei eine Evidenz *E*, z.B. eine Beobachtung oder Messung, gesucht ist eine Hypothese H_i. Angelehnt an die Terminologie der bedingten Wahrscheinlichkeiten werden folgende Begriffe eingeführt:

$x = \mu_B(H_i \mid E)$ – "Maß der Glaubwürdigkeit von H_i nach Vorliegen von *E*" („Measure of Belief")
$x = \mu_D(H_i \mid E)$ – "Maß des Zweifels an H_i nach Vorliegen von *E*" („Measure of Disbelief")

Dabei sind x, y $\in$ [0, 1]. Der *Sicherheitsfaktor* (Certainty Factor, *CF*) von H_i bei gegebenem *E* ist

$$CF(H_i \mid E) = \mu_B(H_i \mid E) - \mu_D(H_i \mid E)$$

Es ist also $-1 \le CF \le 1$, wobei die einzelnen Werte folgende Bedeutung haben sollen:

CF = 0: Keine Information
CF > 0: Mehr positive als negative Argumente
CF < 0: Mehr negative als positive Argumente

$CF(H \mid E)$ drückt den Einfluss der Evidenz *E* auf den Glauben an *H* aus.

Beim Gebrauch von Sicherheitsfaktoren müssen Schätzungen gemacht werden. Geschätzte Werte sind natürlich unzuverlässig, aber man kann sie für ein Sachgebiet verbessern, indem man sie auf richtige Konsequenzen hin testet und gegebenenfalls verbessert. An *CF* werden bestimmte Anforderungen gestellt (die für *CF*' erfüllt sind):

Bedingungen für CF

1) $CF(H \mid E_1 \wedge E_2) = CF(H \mid E_2 \wedge E_1)$
   $CF(H \mid E_1 \wedge (E_2 \wedge E_3)) = CF(H \mid (E_1 \wedge E_2) \wedge E_3)$
   Die Gleichungen sollen ausdrücken, dass der Sicherheitsfaktor von der Reihenfolge und Zusammenfassung der Beobachtungen unabhängig ist. Es handelt sich hierbei nicht um eine logische Konjunktion, sondern um ein zeitliches Nacheinander.
2) Wenn $CF(H \mid E) = 0$, dann $CF(H \mid E \wedge E_1) = CF(H \mid E_1)$
3) Wenn $1 \neq CF(H \mid E_1) = -CF(H \mid E_2) \neq -1$, dann ist $CF(H \mid E_1 \wedge E_2)$ nicht definiert.

Bei der Kombination von Evidenzen mit Hilfe von Sicherheitsfaktoren soll folgender Schluss gemacht werden:

Von $$E_1 \xrightarrow{CF(E|E_1)} E \xrightarrow{CF(H|E),CF(H|\neg E)} H$$

soll auf $$E_1 \xrightarrow{CF(H|E_1)} H$$

geschlossen werden. Zu diesem Zweck wird eine vierte Bedingung an *CF* gestellt:

4) Wenn $CF(E \mid E_1) = 1$, dann $CF(H \mid E_1) = CF(H \mid E)$
   Wenn $CF(E \mid E_1) = -1$, dann $CF(H \mid E_1) = CF(H \mid \neg E)$
   Wenn $CF(E \mid E_1) = 0$, dann $CF(H \mid E_1) = 0$

Im allgemeinen Fall wird folgende Bedingung gefordert:

$$CF(H \mid E_1) = F(CF(E \mid E_1), CF(H \mid E), CF(H \mid \neg E))$$

Die Funktion *F* soll monoton und stetig in $CF(E \mid E_1)$ sein. Die genaue Form der Funktion ist damit noch nicht gegeben. Eine bestimmte Form ist gegeben, wenn *CF*' als Sicherheitsfaktor gewählt wird, vorausgesetzt die entsprechenden Wahrscheinlichkeiten sind bekannt. Für die Kombination der Evidenzen muss noch eine entsprechende Funktion *G* festgelegt werden durch

$$E_1 \wedge E_2 \xrightarrow{G(CF(H|E_1),CF(H|E_2))} H$$

Der folgende Vorschlag für *F* und *G* ist numerisch einfach auszuwerten und sehr anschaulich:

$$CF(H|E_1) = \begin{cases} CF(E|E_1) \cdot CF(H|E) & \text{für } CF(E|E_1) \geq 0 \\ -CF(E|E_1) \cdot CF(H|\neg E) & \text{für } CF(E|E_1) < 0 \end{cases}$$

$$CF(H|E_1 \wedge E_2) = \begin{cases} x + y - xy & \text{für } x \geq 0, y \geq 0 \\ \dfrac{x+y}{1 - \min(|x|,|y|)} & \text{für } xy < 0 \\ x + y + xy & \text{für } x < 0, y < 0 \end{cases}$$

mit $x = CF(H \mid E_1)$ und $y = CF(H \mid E_2)$.

Um die Zuverlässigkeit der Evidenzpropagierung zu testen, kann man für bekannte Fälle die Abweichungen zu *CF*' graphisch aufzeichnen. Ist das Endergebnis der Schlussfolgerung bekannt, dann kann auch dies zur Beurteilung herangezogen werden. Im Allgemeinen sind die Fehler umso größer, je länger die Folgerungsketten sind und je mehr die Einzelevidenzen voneinander abhängen. Die Methode der Sicherheitsfaktoren ist also dann günstig, wenn zahlreiche Beobachtungen vorliegen, die einzeln keine überragende Bedeutung haben und nicht in langwierige Verarbeitungen eingehen. Sie hat eigentlich nur heuristischen Wert, denn sie beruht nicht auf exakten Modellvorstellungen wie etwa die Bayessche Regel.

Werden Formalisierungen von Unsicherheiten, in Form von Sicherheitsfaktoren, Evidenzen, Wahrscheinlichkeiten oder Fuzzy-Werten, in einem Diagnosesystem eingesetzt, dann steuern sie zum einen den gesamten Ablauf, weil die Hypothesen mit hohen Faktoren eine gewisse Priorität haben, zum andern liefern sie ein Abbruchkriterium. Dieses kann auf unterschiedliche Weise definiert werden, z.B. dadurch, dass Hypothesen, die einen bestimmten Schwellenwert überschreiten, als Kandidaten für die Diagnose genommen werden, oder durch die Auswahl der am höchsten qualifizierten Hypothese.

Bei der Akkumulation von Unsicherheiten entsteht ein generelles Problem aus der Abhängigkeit von Ereignissen, die Anlass zur Einführung von entsprechenden Werten geben. Die Regeln für die Berechnung von Unsicherheitsfaktoren können in solchen Fällen qualitativ beliebig falsch werden. Um Probleme, die sich daraus ergeben, nimmt man meist die Unabhängigkeit der Ereignisse an. Aber das ist in der Praxis mit Vorsicht zu genießen. Angenommen, bei einer Diagnose weisen gewisse Symptomwerte auf Krankheiten hin. In diesem Fall sind entweder die Symptome aus einer gemeinsamen Krankheitsursache zu erklären, dann sind sie nicht unabhängig, oder die Symptomwerte sind unabhängig, dann können sie aber nicht zu einer gemeinsamen Krankheitsdefinition herangezogen werden. In jedem Fall ist also eine genaue Analyse der Abhängigkeiten erforderlich.

5.3. Zusammenfassung

Es wurden allgemeine Beschreibungen der Struktur, des Entwurfs und der Bewertungskriterien für Expertensysteme gegeben. Im Detail wurde das Problem der Darstellung und Behandlung von Unsicherheiten behandelt, das beim praktischen Einsatz der Systeme eine besondere Rolle spielt. Dabei wurde hauptsächlich zwischen objektiver und subjektiver Unsicherheit unterschieden. Im Einzelnen sind folgende Punkte wesentlich:

- Um das Problem der Indeterminiertheit und des Mangels an scharf umrissenen Strukturen zu lösen, wird deklaratives Wissen verwendet, das in der Wissensbasis, der ersten Komponente eines Expertensystems, abgelegt wird. Außerdem wird auch implizites Wissen, direktes und indirektes Wissen verwendet. Das Wissen wird in der zweiten Komponente der Expertensysteme, dem Inferenzsystem, verarbeitet.
- Der Entwurf eines Expertensystems läuft in drei Schritten ab: Informelle Beschreibung der Anforderungen, Überführung in eine formale Spezifikation und Realisierung durch eine Architektur. Es gibt Entwurfswerkzeuge für den Top-Down-Entwurf (Shells) und Low-level-Werkzeuge (Werkzeugkasten).
- Es gibt verschiedene Bewertungskriterien, die die Spezifikation, die Architektur und die Realisierungsmittel betreffen. Bewertet werden das System als Softwareprodukt und die Qualität der Ergebnisse, die es liefert.
- Gründe für subjektive Unschärfe sind sensorische Unsicherheit, sprachliche und begriffliche Unsicherheit und subjektive Wahrscheinlichkeit. Gründe für objektive Unschärfe sind Messfehler und numerische Ungenauigkeiten, statistische Aussagen und Unkenntnis von Parametern und allgemeinen Zusammenhängen.
- Die Methode der groben Mengen basiert auf einer Ununterscheidbarkeitsrelation auf einem Universum U. Eine Menge M werden nicht scharf angegeben, sondern durch eine innere Teilmenge und eine äußere umfassende Menge approximiert. Für die Elemente zwischen diesen beiden Mengen kann nicht angegeben werden, ob sie zu M gehören oder nicht.
- Man unterscheidet zwischen bedingten und unbedingten Wahrscheinlichkeiten. Erstere liegen dann vor, wenn ein Ereignis im Zusammenhang mit anderen betrachtet wird. Der Satz von Bayes stellt einen Zusammenhang zwischen beiden her.
- Mit der Evidenzentheorie von Dempster und Shafer wird versucht, *Unwissenheit* anstelle von *Unsicherheit* wie bei der Wahrscheinlichkeitstheorie zu modellieren. Vorausgesetzt wird eine vollständige Menge sich gegenseitig ausschließender Alternativen. Für diese wird ein Basismaß, d.h. eine Abbildung in das Intervall [0, 1] definiert. Die Werte des Basismaßes werden als Evidenzen aufgefasst und es wird eine Regel für das Propagieren von Evidenzen angegeben.

- Fuzzy-Mengen dienen zur Repräsentation subjektiver Vagheit. Sie sind ebenfalls Abbildung in das Intervall [0, 1]. Aussagen über die Zugehörigkeit von Elementen zu einer Fuzzy-Menge sind Fuzzy-Aussagen. Diese können nach bestimmten Regeln logisch verknüpft werden, so dass sich insgesamt ein Kalkül zum Schlussfolgern über Fuzzy-Aussagen ergibt. Fuzzy-Werte und Wahrscheinlichkeitswerte lassen sich zu Fuzzy-Wahrscheinlichkeiten kombinieren.
- Sicherheitsfaktoren dienen zur Darstellung objektiver Unsicherheiten und stellen vor allem ein einfaches Verfahren zur Propagierung der Werte in einem Regelsystem bereit. Sie sind den Wahrscheinlichkeitswerten ähnlich, schränken aber deren Verwendung durch zusätzliche Bedingungen ein.

6. Planen

6.1. Ein einfacher Planungsalgorithmus

Wenn der Weltzustand zugänglich ist kann ein Planer Wahrnehmungen nutzen, die von der Umgebung geliefert werden, um ein vollständiges und korrektes Modell des aktuellen Weltzustands aufzubauen. Ist ein Ziel gegeben, dann kann er einen geeigneten Planungsalgorithmus aufrufen um einen Aktionsplan zu erzeugen. Danach kann der Planer den Plan Aktion für Aktion ausführen. Der folgende Algorithmus implementiert einen einfachen Planer.

function EINFACHER-PLANER(*Wahrnehmung*) **returns** eine *Aktion*
  static: *KB*, eine Wissensbasis mit Aktionsbeschreibungen
    p, ein Plan, zu Beginn *NoPlan*
    t, ein Zähler für die Zeit, zu Beginn 0
  local variables: *G*, ein Ziel
    current, eine Beschreibung des aktuellen Zustands

  TELL(*KB*, MAKE-PERCEPT-SENTENCE(*Wahrnehmung*, *t*))
  current ← STATE-DESCRIPTION(*KB*, *t*)
  if *p* = *NoPlan* **then**
    G ← ASK(*KB*, MAKE-GOAL-QUERY(*t*))
    p ← IDEAL-PLANNER(*current*, *G*, *KB*)
  if *p* = *NoPlan* **or** *p* ist leer **then** *Aktion* ← *NoOp*
  else
    Aktion ← FIRST(*p*)
    p ← REST(*p*)
  TELL(*KB*, MAKE-ACTION-SENTENCE(*Aktion*, *t*))
  t ← *t* + 1
  return *Aktion*

6.2. Vom Problemlösen zum Planen

Die grundlegenden Repräsentationen eines suchorientierten Problemlösers sind:

- **Repräsentation von Aktionen** Aktionen werden durch Programme dargestellt, die Beschreibungen von Folgezuständen erzeugen.
- **Repräsentation von Zuständen** Eine vollständige Beschreibung des Anfangszustands ist gegeben und Aktionen erzeugen vollständige Zustandsbeschreibungen. Deshalb sind alle Zustandsbeschreibungen vollständig. Bei den meisten Problemen ist ein Zustand eine einfache Datenstruktur. Zustandsrepräsentationen werden nur für die Erzeugung von Nachfolgern, für die Auswertung heuristischer Funktionen und das Zielprädikat benötigt.
- **Repräsentation von Zielen** Die einzigen Informationen, die ein Problemlöser über sein Ziel hat, sind das Zielprädikat und die heuristische Funktion. Sie werden auf Zustände angewendet um deren Wünschbarkeit zu bestimmen, aber sie werden als *Black boxes* verwendet, d.h. der Problemlöser kennt nicht die Kriterien, nach denen sie entscheiden, und kann nicht danach seine Aktionen auswählen.

- **Repräsentation von Plänen** Eine Lösung ist eine Folge von Aktionen. Der Suchalgorithmus betrachtet beim Aufbau von Lösungen nur ununterbrochene Folgen von Aktionen, die beim Anfangszustand beginnen.

Drei *Schlüsselideen*, die dem Planen zugrundeliegen:

1. „Öffne“ die Repräsentation der Zustände, Ziele und Aktionen. Dies wird erreicht durch Verwendung einer geeigneten Repräsentationsform, z.B. der Logik erster Ordnung oder einer Teilmenge von ihr. Zustände und Ziele werden darin als Sätze repräsentiert und Aktionen als logische Beschreibungen von Vorbedingungen und Wirkungen. Dadurch ist es möglich, direkte Verbindungen zwischen Zuständen und Aktionen herzustellen.
2. Der Planer kann neue Aktionen zu seinem Plan hinzufügen wann immer dies erforderlich ist, nicht nur zu Beginn der Planung vom Anfangszustand aus. Planung und Ausführung müssen nicht streng getrennt sein. Indem offensichtliche oder wichtige Entscheidungen zuerst getroffen werden, werden der Verzweigungsfaktor und die Notwendigkeit zum Rücksetzen stark reduziert.
3. Die meisten Teile der Welt sind unabhängig von den meisten anderen Teilen. Deshalb kann man oft ein Ziel als Konjunktion mehrerer voneinander unabhängiger Teilziele formulieren und mit einer Divide-and-Conquer-Strategie lösen.

6.3. Grundlegende Repräsentationen für das Planen

6.3.1. Repräsentation von Zuständen und Zielen

In der STRIPS-Sprache (STanford Research Institute Problem Solver) werden Zustände durch Konjunktionen funktionsfreier Grundliterale repräsentiert. Diese Literale bestehen also nur aus Prädikaten, angewandt auf Konstanten.

Zustandsbeschreibungen müssen nicht vollständig sein. In einer unzugänglichen Welt hat ein Planer oft unvollständige Zustandsbeschreibungen. Eine solche entspricht einer Menge möglicher Zustände, für die der Planer erfolgreiche Pläne zu erstellen versucht. Viele Planungssysteme folgen der „Closed World Assumption“, nach der ein nicht erwähntes positives Literal als falsch angenommen wird.

Ziele werden ebenfalls durch Konjunktionen von Literalen repräsentiert. Diese Literale können auch Variable enthalten. Diese werden, wie die Ziele beim Theorembeweisen, als existentiell quantifiziert angenommen. Es besteht aber ein grundsätzlicher Unterschied zwischen der Frage bei einem Theorembeweiser und dem Ziel bei einem Plan. Letzteres fragt nach einer Folge von Aktionen, die ein Ziel erfüllen, wenn sie ausgeführt werden. Ersteres fragt, ob die Frage wahr ist wenn die Sätze der Wissensbasis wahr sind.

6.3.2. Repräsentation von Aktionen

Die STRIPS-Repräsentation für Aktionen besteht aus drei Komponenten:

- **Aktionsbeschreibung**: Eine Beschreibung dessen, was der Agent an die Umgebung ausgibt um etwas zu tun. Innerhalb des Planers dient sie nur als Name für eine mögliche Aktion.
- **Vorbedingung**: Eine Konjunktion von Atomen (positive Literale), die angibt, was gelten muss bevor der Operator angewandt werden kann.

- **Wirkung**: Eine Konjunktion von Literalen (positiv oder negativ), die beschreibt, wie sich die Situation verändert, wenn der Operator angewandt wird.

Ein Beispiel für die Syntax zur Formulierung eines STRIPS-Operators ist der folgende Operator für die Bewegung von einem Platz zu einem anderen:

$Op(\text{ACTION}: Gehezu(dort), \text{PRECOND}: An(hier) \wedge Pfad(hier, dort), \text{EFFECT}: An(dort) \wedge \neg An(hier))$

Abbildung 6.1 zeigt eine graphische Notation des Operators. Man beachte, dass der Operator keine expliziten Situationsvariablen enthält.

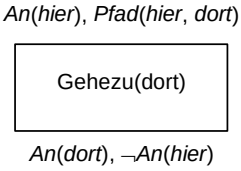

Abbildung 6.1

Ein Operator mit Variablen wird als **Operatorschema** bezeichnet, denn er beschreibt nicht nur eine einzige Aktion, sondern eine Menge von Aktionen, die sich durch Einsetzen verschiedener Konstanten für die Variablen als **Instanzen** des Schemas ergeben. Üblicherweise können nur Instanzen von Operatorschemata ausgeführt werden. Die Planungsalgorithmen sorgen dafür, dass alle Variablen in einem Operatorschema vor der Anwendung gebunden sind.

Ein Operator *o* ist **anwendbar** in einem Zustand *s*, wenn die Variablen in *o* so instanziiert werden können, dass alle Vorbedingungen von *o* erfüllt sind in *s*, d.h. wenn $Precond(o) \subseteq s$. Im resultierenden Zustand gelten alle positiven Literale in *Effect*(*o*), ebenso alle Literale, die schon in *s* gegolten haben, außer denen, die negativ in *Effect*(*o*) sind. Ein Beispiel: In der Anfangssituation gelten die Literale

$$An(ZuHause), Pfad(ZuHause, Supermarkt), ...$$

Dann ist die Aktion *Gehezu*(*Supermarkt*) anwendbar und die Ergebnissituation enthält die Literale

$$\neg An(ZuHause), An(Supermarkt), Pfad(ZuHause, Supermarkt), ...$$

6.3.3. Situationsraum und Planraum

Ein **Situationsraum-Planer** ist ein Algorithmus, der im Raum der Situationen oder Zustände wie ein klassischer Problemlöser einen Pfad vom Anfangszustand zu einem Zielzustand durch Anwendung von Operationen bestimmt. Es gibt zwei Varianten des Situationsraum-Planers, den **Vorwärtsplaner** und den **Rückwärtsplaner**. Der Erste plant vom Anfangszustand zum Zielzustand, der Zweite in umgekehrter Richtung.

Bei der Rückwärtsplanung sind die Zustände im Allgemeinen nur partiell bestimmt. Die Operatoren enthalten aber genügend Informationen um von partiell bestimmten Zuständen auf partiell bestimmte Vorgängerzustände zu schließen. Der Zielzustand besteht gewöhnlich aus wenigen Literalen und auf jedes ist nur eine kleine Zahl von Operatoren anwendbar, während auf den Anfangszustand viele Operatoren anwendbar sind. Der Nachteil der Rückwärtsplanung ist aber die Konjunktion der Literale, von denen jedes ein zu erfüllendes Teilziel darstellt.

Ein **Planraum** ist eine Menge **partieller Pläne** zusammen mit Operatoren, die partielle Pläne in ebensolche überführen. Bei der Erstellung eines Plans für ein Problem beginnt man mit einem einfachen unvollständigen Plan und modifiziert diesen durch Anwendung der Operatoren so lange, bis ein vollständiger Plan für das Problem entsteht. Operatoren sind das Hinzufügen eines Schrittes, das Ordnen von Schritten, das Instanziieren von Variablen u.a. Die Operatoren lassen sich in zwei Gruppen unterteilen. **Verfeinerungsoperatoren** fügen zu einem partiellen Plan Bedingungen (Constraints) hinzu. Dadurch wird die Menge der vollständigen Pläne, die der partielle Plan repräsentiert, eingeschränkt. Alle übrigen Operatoren sind **Modifikationsoperatoren**.

6.3.4. Repräsentation von Plänen

Nach dem **Least Commitment**-Prinzip beim Planen werden Entscheidungen, die im Planungsprozess zu treffen sind, so lange wie möglich zurückgestellt und erst dann getroffen, wenn dies unumgänglich ist. Ein Plan, in dem die Menge der Schritte nur teilweise geordnet ist, heißt **partiell geordnet**. Ist die Menge der Schritte vollständig geordnet, dann heißt der Plan **vollständig geordnet**. Entsteht ein vollständig geordneter Plan aus einem Plan *P* durch Hinzufügen von Constraints, dann heißt er eine **Linearisierung** von *P*. Ein Plan, in dem jede Variable an eine Konstante gebunden ist, heißt **voll instanziierter Plan**.

Ein Plan ist eine Datenstruktur bestehend aus vier Komponenten:

- Eine Menge von Planschritten. Jeder Schritt ist einer der Operatoren für das Problem.
- Eine Menge von Constraints, die die Schritte ordnen. Die Constraints haben die Form $S_i \prec S_j$, gelesen „S_i vor S_j“, was bedeutet, dass Schritt S_i irgendwann vor Schritt S_j kommen muss, aber nicht notwendigerweise unmittelbar vor S_j.
- Eine Menge von Constraints für Variablenbindungen. Die Constraints haben die Form $v = x$, wobei v eine Variable ist, die in einem Schritt vorkommt, und x eine Konstante oder eine andere Variable.
- Eine Menge **kausaler Kanten**. Eine kausale Kante wird geschrieben $S_i \xrightarrow{c} S_j$ und gelesen als „S_i erreicht c für S_j“. Kausale Kanten beschreiben den Zweck von Schritten in einem Plan. Bei der Kante $S_i \xrightarrow{c} S_j$ ist der Zweck von S_i die Vorbedingung c von S_j zu erfüllen.

Der Anfangsplan, der vor Anwendung einer Operation gegeben ist, beschreibt das ungelöste Problem. Er besteht nur aus zwei Schritten, genannt *Start* und *Ende*, und dem Ordnungsconstraint *Start* $\prec$ *Ende*. Beide Schritte enthalten keine Aktion. Der *Start*-Schritt hat keine Vorbedingungen, aber eine Wirkung, nämlich die alle Aussagen hinzuzufügen, die im Anfangszustand wahr sind. Der *Ende*-Schritt hat den Zielzustand als Vorbedingung, aber keine Wirkung. Der Planungsprozess beginnt mit dem Anfangsplan und führt so lange Verfeinerungsschritte durch, bis ein vollständiger Plan entsteht.

Als Beispiel wird ein Anfangsplan für das Schuhe anziehen betrachtet. Er hat folgende Gestalt:

Plan(STEPS: { S_1: *Op*(ACTION: *Start*),
S_2: *Op*(ACTION: *Ende*, PRECOND: *RechterSchuhAn* $\wedge$ *LinkerSchuhAn*)},
ORDERINGS: {$S_1 \prec S_2$},
BINDINGS: {},
LINKS: {})

Abbildung 6.2(a) zeigt eine graphische Notation für Pläne, Abbildung 6.2(b) zeigt den Anfangsplan für das Schuhe-Anziehen-Problem.

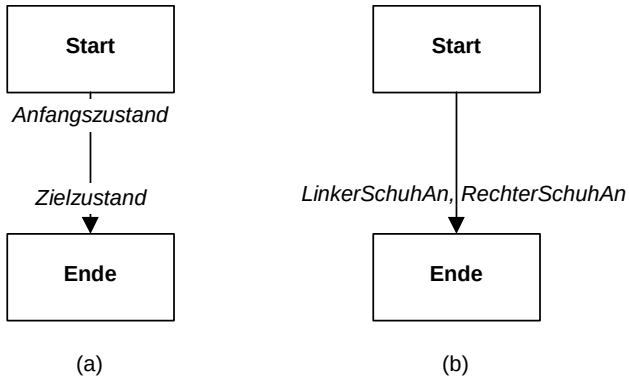

Abbildung 6.2

6.3.5. Lösungen

Eine Lösung ist ein Plan, den ein Planer ausführen kann und der das Erreichen des Ziels garantiert. Vollständig instanziierte, vollständig geordnete Pläne erfüllen diese Bedingung. Trotzdem ist es nicht sinnvoll nur solche Pläne zuzulassen aus drei Gründen:

1. Für viele Probleme, z.B. das Schuhe-Anziehen-Problem, ist es natürlicher, den partiell geordneten Plan als Lösung zu akzeptieren als eine beliebige Linearisierung auszuwählen.
2. Manche Planer können Aktionen parallel zueinander ausführen, für sie sind Pläne, die dies erlauben, vorteilhafter.
3. Partiell geordnete Pläne können leichter mit anderen Plänen zu größeren Plänen zusammengebaut werden.

Deshalb wird eine Lösung so definiert: Eine **Lösung** ist ein **vollständiger**, **konsistenter** Plan. Diese beiden Begriffe werden wie folgt definiert.

Ein Plan ist **vollständig**, wenn jede Vorbedingung jedes Schritts durch einen anderen Schritt erreicht wird. Ein Schritt **erreicht** eine Bedingung, wenn diese eine der Wirkungen des Schritts ist und kein anderer Schritt die Bedingung eventuell ungültig machen kann. Formaler gefasst:

Ein Schritt S_i erreicht eine Vorbedingung c eines Schritts S_j, wenn gilt:

(1) $S_i \prec S_j$ und $c \in \text{EFFECTS}(S_i)$
(2) Es gibt keinen Schritt S_k so dass $\neg c \in \text{EFFECTS}(S_k)$, wobei $S_i \prec S_k \prec S_j$ in irgendeiner Linearisierung des Plans gilt.

Ein Plan ist **konsistent**, wenn er keine Widersprüche in der Ordnung der Schritte und der Variablenbindung enthält. Ein Widerspruch liegt vor, wenn für zwei Schritte S_i und S_j sowohl $S_i \prec S_j$ als auch $S_j \prec S_i$ gilt oder wenn für eine Variable v und zwei verschiedene Konstanten A und B sowohl $v = A$ als auch $v = B$ gilt. Da sowohl $\prec$ als auch = transitive Relationen sind, ist z.B. auch ein Plan mit $S_1 \prec S_2$, $S_2 \prec S_3$ und $S_3 \prec S_1$ inkonsistent.

6.4. Ein Beispiel für partiell geordnetes Planen

Das betrachtete Beispiel ist das Milch-Bananen-Mix-Problem. Es werden zwei vereinfachende Annahmen gemacht: Die *Gehezu*-Aktion kann für Bewegungen zwischen zwei beliebigen Orten verwendet werden und bei der Definition der *Kaufe*-Aktion wird das Geld nicht betrachtet. Als Abkürzungen werden verwendet: *HWG* für *Haushaltswarengeschäft* und *SM* für *Supermarkt*. Der Anfangszustand und der Zielzustand sind wie folgt definiert:

$$Op(\text{ACTION}: Start, \text{EFFECT}: An(ZuHause) \wedge Verkauft(HWG, Rührgerät) \wedge Verkauft(SM, Milch) \wedge Verkauft(SM, Bananen))$$

$$Op(\text{ACTION}: Ende, \text{PRECOND}: An(ZuHause) \wedge Hat(Rührgerät) \wedge Hat(Milch) \wedge Hat(Bananen))$$

Die Aktionen *Gehezu* und *Kaufe* sind wie folgt definiert:

$Op(\text{ACTION}: Gehezu(dort), \text{PRECOND}: An(hier), \text{EFFECT}: An(dort) \wedge \neg An(hier))$

$Op(\text{ACTION}: Kaufe(x), \text{PRECOND}: An(geschäft) \wedge Verkauft(geschäft, x), \text{EFFECT}: Hat(x))$

Abbildung 6.3 zeigt den Anfangsplan.

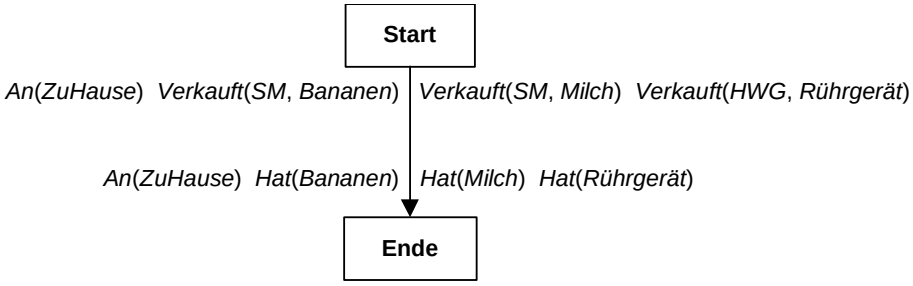

Abbildung 6.3

Ein Planungsalgorithmus **schützt** kausale Kanten, indem er dafür sorgt, dass zwischen zwei Aktionen, die durch eine kausale Kante verbunden sind, keine Aktion eingeschoben wird, die die erreichte Vorbedingung der zweiten Aktion löscht. Muss eine Aktion eingeplant werden, die eine **Bedrohung** für eine kausale Kante in diesem Sinne darstellt, dann kann der Planungsalgorithmus versuchen, diese Aktion entweder vor der geschützten Kante (**Promotion**) oder nach ihr (**Demotion**) einzufügen. Ist dies möglich, dann kann der Planungsprozess fortgesetzt werden, andernfalls muss zu einem früheren Punkt des Planungsprozesses zurückgesetzt werden.

Dies ist in Abbildung 6.4 illustriert. In (a) verletzt die Wirkung des Schritts S_3 die Vorbedingung der Aktion S_2 und bedroht damit die kausale Kante von S_1 nach S_2. Durch Promotion entsteht die Lösung von (b) und durch Demotion die Lösung von (c).

Abbildung 6.5 zeigt den vollständigen Plan mit einer Anordnung der Schritte, die den Ordnungsconstraints entspricht. Der Plan ist fast vollständig geordnet, nur die Reihenfolge der Aktionen *Kaufe*(*Milch*) und *Kaufe*(*Bananen*) ist nicht festgelegt, da es dafür keine Vorgaben in der Definition der Aktionen gibt.

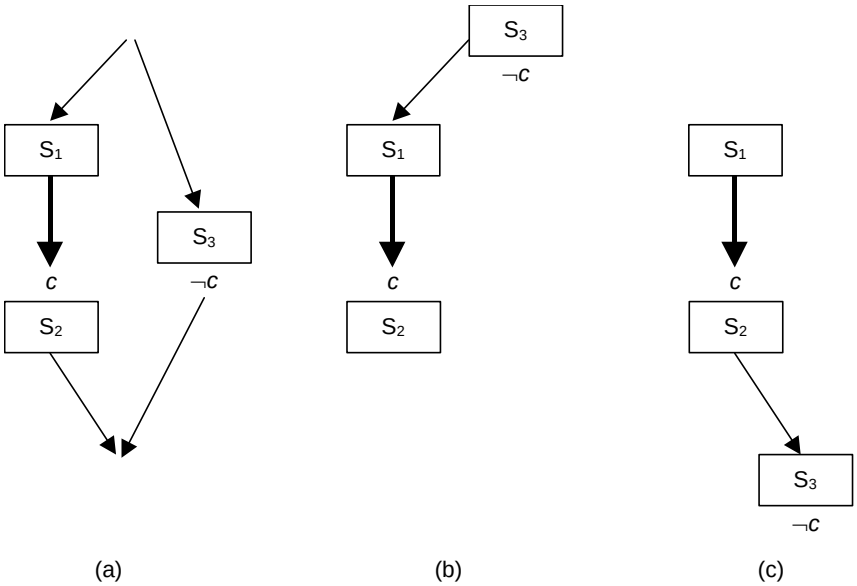

Abbildung 6.4

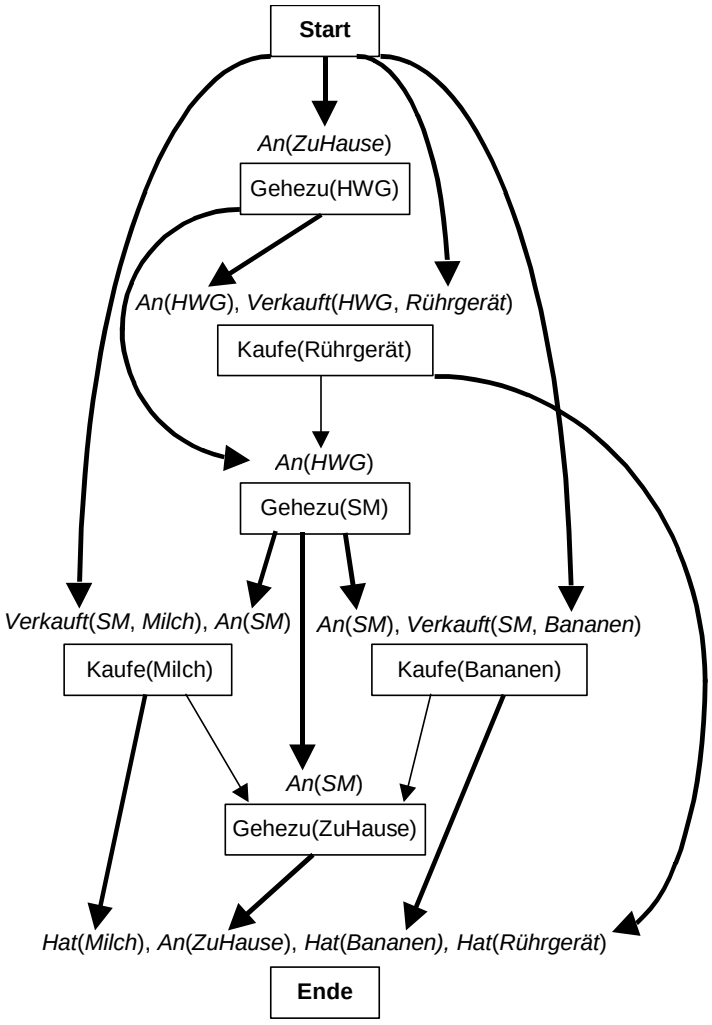

Abbildung 6.5

6.5. Ein Algorithmus für partiell geordnetes Planen

Der folgende Algorithmus ist ein Planer für partiell geordnetes Planen, genannt POP (Partial-Order Planner). Er ist nichtdeterministisch, wie man an den Operationen *Choose* und *Select* erkennen kann.

function POP(*initial*, *goal*, *operators*) **returns** *plan*

plan ← MAKE-MINIMAL-PLAN(*initial*, *goal*)
loop do
  if SOLUTION?(*plan*) **then return** *plan*
  S_{need}, c ← SELECT-SUBGOAL(*plan*)
  CHOOSE-OPERATOR(*plan*, *operators*, S_{need}, c)
  RESOLVE-THREATS(*plan*)
end

function SELECT-SUBGOAL(*plan*) **returns** S_{need}, c

wähle einen Planungsschritt S_{need} aus STEPS(*plan*) mit einer Vorbedingung c,
  die noch nicht erreicht worden ist
return S_{need}, c

procedure CHOOSE-OPERATOR(*plan*, *operators*, S_{need}, c)

choose einen Schritt S_{add} aus *operators* oder STEPS(*plan*), der c als Wirkung hat
if es gibt keinen solchen Schritt **then fail**
füge die kausale Kante $S_{add} \xrightarrow{c} S_{need}$ zu LINKS(*plan*) hinzu
füge das Ordnungsconstraint $S_{add} \prec S_{need}$ zu ORDERINGS(*plan*) hinzu
if S_{add} ist ein neu hinzugefügter Schritt aus *operators* **then**
  füge S_{add} zu STEPS(*plan*) hinzu
  füge $Start \prec S_{add} \prec Ende$ zu ORDERINGS(*plan*) hinzu
end

procedure RESOLVE-THREATS(*plan*)

for each Schritt S_{threat} der eine Kante $S_i \xrightarrow{c} S_j$ in LINKS(*plan*) bedroht **do**
  choose eine der folgenden Aktionen
    Promotion: Füge $S_{threat} \prec S_i$ zu ORDERINGS(*plan*) hinzu
    Demotion: Füge $S_j \prec S_{threat}$ zu ORDERINGS(*plan*) hinzu
  if not CONSISTENT(*plan*) **then fail**
end

6.6. Knowledge Engineering für das Planen

Kurz zusammengefasst sind beim Knowledge Engineering für das Planen folgende Schritte durchzuführen:

- Lege den Gegenstandsbereich fest.
- Lege das Vokabular für Bedingungen (Literale), Operatoren und Objekte fest.
- Definiere Operatoren für den Anwendungsbereich.
- Definiere eine Beschreibung des speziellen Problems.
- Übergib die Problemformulierungen dem Planer und erhalte Pläne zurück.

6.6.1. Die Blockwelt

Gegenstandsbereich: Der Bereich besteht aus einer Menge von Spielzeugblöcken, die auf einem Tisch liegen und einem Roboterarm, der die Blöcke ergreifen und bewegen kann. Die Blöcke können gestapelt sein, aber auf jeden Block passt höchstens ein anderer. Der Arm kann immer nur einen Block ergreifen, er kann ihn auf den Tisch oder auf einen anderen Block setzen. Das Ziel ist, Stapel bestimmter Blöcke zu bilden, z.B. Block A auf B oder C auf D und D auf E.

Vokabular: Die Objekte sind die Blöcke und der Tisch. Sie werden durch Konstanten repräsentiert. *On*(*b*, *x*) bezeichnet, dass Block *b* auf *x* liegt, das ein anderer Block oder der Tisch sein kann. Der Operator für die Bewegung eines Blocks ist *Move*(*b*, *x*, *y*). Dies bedeutet, dass Block *b* von der Position auf *x* zu der Position auf *y* bewegt wird. Eine der Vorbedingungen für die Bewegung eines Blocks ist, dass nichts auf ihm liegt. Dies wird durch die Formulierung *Clear*(*x*) ausgedrückt, die besagt, dass nichts auf x liegt.

Operatoren: Die formale Definition der Bewegungsoperation ist folgendermaßen:

Op(ACTION: *Move*(*b*, *x*, *y*),
PRECOND: $On(b, x) \wedge Clear(b) \wedge Clear(y)$,
EFFECT: $On(b, y) \wedge Clear(x) \wedge \neg On(b, x) \wedge \neg Clear(y)$)

Der Operator für die Bewegung eines Blocks auf den Tisch ist folgendermaßen definiert:

Op(ACTION: *MoveToTable*(*b*, *x*),
PRECOND: $On(b, x) \wedge Clear(b)$,
EFFECT: $On(b, Table) \wedge Clear(x) \wedge \neg On(b, x)$)

Clear(*x*) wird interpretiert als „es gibt einen freien Platz auf x für einen Block“. Unter dieser Interpretation ist *Clear*(*Table*) immer Bestandteil der Anfangssituation und es ist klar, dass *Move*(*b*, *Table*, *y*) die Wirkung *Clear*(*Table*) hat.

6.6.2. Shakeys Welt

Das ursprüngliche STRIPS-Programm wurde zur Steuerung des Roboters Shakey, entwickelt in den 70-er Jahren am SRI, entworfen. Shakeys Welt besteht aus vier Räumen, die hintereinander angeordnet und durch einen Flur verbunden sind. Jeder Raum hat eine Tür und einen Lichtschalter, siehe Abbildung 6.6.

Shakey kann sich von einem Ort zu einem anderen bewegen, er kann bewegliche Objekte schieben (z.B. Kisten), kann auf feste Objekte steigen (z.B. Kisten) und kann Lichtschalter betätigen. Das benötigte Vokabular wird zusammen mit den Operatoren dargestellt.

1. *Go*(*y*): Gehe vom aktuellen Ort zum Ort *y*.
2. *Push*(*b*, *x*, *y*): Schiebe einen Gegenstand *b* vom Ort *x* zum Ort *y*.
3. *Climb*(*b*): Klettere auf eine Kiste *b*.
4. *Down*(*b*): Steige von einer Kiste *b* herunter.
5. *TurnOn*(*ls*): Schalte einen Lichtschalter *ls* an.
6. *TurnOff*(*ls*): Schalte einen Lichtschalter *ls* aus.

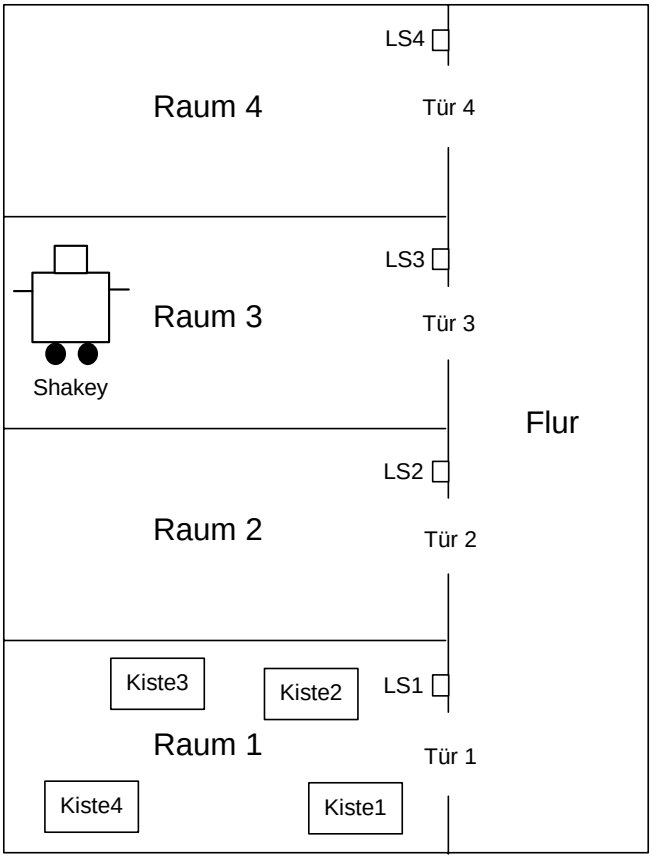

Abbildung 6.6

6.7. Zusammenfassung

Es wurde gezeigt, dass der Situationskalkül ausdrucksstark genug ist um darin Planungsprobleme zu formulieren. Für den Planungsprozess empfiehlt sich nicht ein allgemeiner Theorembeweiser, denn er arbeitet sehr ineffizient. Mit einer eingeschränkten Sprache und speziellen Algorithmen können Planungssysteme komplexe Planungsprobleme lösen. Die Aufgabe beim Planen ist also, eine Sprache festzulegen, die ausdrucksstark genug ist aber auch eingeschränkt genug, so dass effizientes Planen möglich ist. Im Einzelnen sind folgende Punkte wesentlich:

- Planer blicken voraus um Aktionen zu bestimmen, die zum Erreichen des Ziel beitragen. Gegenüber den Problemlösern benutzen sie flexiblere Repräsentationen von Zuständen, Aktionen, Zielen und Plänen.
- Die STRIPS-Sprache beschreibt Aktionen durch Angabe von Vorbedingungen und Wirkungen. Sie umfasst einen großen Teil der Ausdrucksfähigkeit der Logik, aber nicht alle Anwendungsbereiche und Probleme können in ihr dargestellt werden.
- In komplexen Anwendungsbereichen ist es nicht möglich im Situationsraum zu suchen. Statt dessen sucht man im Planraum, indem man mit einem minimalen Plan startet und diesen erweitert, bis man eine Lösung findet. Diese Vorgehensweise ist effizient bei Problemen, bei denen sich die Teilpläne nicht gegenseitig beeinflussen.
- Das Least Commitment-Prinzip besagt, dass ein Planer Entscheidungen aufschieben sollte bis ein triftger Grund dafür vorliegt. Partielle Ordnungen auf Plänen und nicht instanziierte Variablen ermöglichen es, das Least Commitment-Prinzip zu befolgen.

- Die kausale Kante ist eine nützliche Datenstruktur um den Zweck einzelner Schritte festzuhalten. Jede kausale Kante etabliert ein Schutzintervall, in dem eine Bedingung nicht durch einen anderen Schritt zerstört werden darf. Kausale Kanten ermöglichen es unlösbare Konflikte in einem partiellen Plan früh zu entdecken und dadurch nutzlose Suche zu vermeiden.
- Der POP-Algorithmus ist ein korrekter und vollständiger Planungsalgorithmus, der auf der Basis von STRIPS-Repräsentationen arbeitet.