

Gerade wegen der vielen Assoziationen aus dem Alltag ist die Idee der *Assistenz* sehr schwer zu fassen, einzuschränken und präzise zu definieren. In den folgenden Abschnitten soll daher versucht werden, den Begriff der Assistenz anhand einer Analyse der beim menschlichen Problemlösen entscheidenden Handlungsphasen zu erfassen. Grundsätzlich kann in jeder dieser Phasen Assistenz gewährt werden. Aus der Wirkung der Assistenz für den Problemlösevorgang leitet sich dann ihr Typ ab. Die verschiedenen Typen von technischen Assistenzsystemen sollen in dieser Einleitung an Beispielen vorgestellt werden. Aus dem Vergleich der verschiedenen Arten geleisteter Assistenz wird dann für die softwaretechnische Implementierung von Assistenz im Sinn der Taxonomie von WANDKE (siehe [1]) ein Forderungskatalog an ein für bestimmte Aufgaben konfigurierbares multimodales Assistenzsystem abgeleitet. Dessen Konzeption wird später beschrieben.

2.1 Typen von Assistenzsystemen

Den Begriff „Assistenz“ zur Charakterisierung einer speziellen Klasse von technischen Systemen aus dem umgangssprachlichen Verständnis des Wortes heraus zu definieren, führt nicht zum Erfolg. Bedeutet nämlich Assistenz die Zunahme von externen Fähigkeiten für die Lösung einer Aufgabe, so ist grundsätzlich jedes technische Gerät, insbesondere aber auch jedes Softwaresystem, ein Assistenzsystem. Schon ein Schraubenzieher eröffnet dem Nutzer Möglichkeiten, die er mit bloßen Händen vermutlich nicht hat. Ähnliches gilt für ein Telefaxgerät, mit dem der Nutzer eine Fotokopie an einen beliebigen Ort versenden kann, ohne sich eines Postboten bedienen zu müssen. Mit einer Rakete können Menschen sogar andere Planeten erreichen – ein technisches Werkzeug, das den menschlichen Handlungsspielraum deutlich erweitert. Die Beispiele machen deutlich, dass dieser Versuch, Assistenz zu definieren, keine Charakterisierung einer Systemklasse ermöglicht, weil zwischen Schraubenzieher und Raumfähre jedes technische Artefakt als Assistenzsystem charakterisiert werden kann.

Wandke [1] eröffnet in seiner Arbeit einen besseren Zugang zum Begriff „Assistenz“. Über eine Taxonomie von Assistenzsystemen zeigt er, wie ein Versuch zur Klassifikation von Assistenzsystemen aussehen könnte, der sich am Grad der Autonomie des analysierten Systems orientiert:

- Assistenz kann dadurch erreicht werden, dass einfache Funktionen automatisch, das heißt ohne Auslösung durch den Benutzer, ausgeführt werden. In diese Klasse fallen Systeme wie Autopiloten in Flugzeugen, Antiblockiersysteme in Kraftfahrzeugen oder automatische Abschaltungssysteme für technische Geräte in Haushalt oder Produktionsstätten.
- Die nächstkomplexere Klasse von Assistenz kombiniert verschiedene elementare, nicht notwendigerweise autonome Funktionen eines Geräts. Assistenz besteht in diesem Fall also darin, den Nutzer bei der Durchführung eines vorab definierten, nicht modifizierbaren Anwendungsfalls durch geeignete Bündelung von Funktionen für eine komplexe Aufgabe zu unterstützen. Typische Beispiele hierfür sind Installationsassistenten von Softwareprogrammen, die Kombination von verschiedenen Funktionen wie Telefon, Camera Radio, Navigationssystem und Mailclient in einem Mobiltelefon, automatische Heizungssteuerung in Wohnräumen, oder unterstützende Funktionen für behinderte Personen wie selbstfahrende Rollstühle, die durch einfache Fingerbewegungen gesteuert werden können, oder die Kommunikation von Information über sprachliche oder taktile Modalitäten.
- Die komplexeste Form von Assistenz versucht, die Intention des Nutzers einzelner durchgeführter Schritte zu erkennen, daraus die aktuelle Aufgabe, die gelöst werden soll, abzuleiten und geeignete Schritte vorzuschlagen, mit denen eine effiziente Lösung möglich ist. Typische Assistenten dieser Klasse sind Onlinehilfen für Softwaresysteme, die teilweise mithilfe animierter Charaktere ein anthropomorphes Aussehen annehmen.

In den folgenden Abschnitten soll jede dieser drei Klassen anhand eines Beispiels näher illustriert werden.

2.1.1 „Autonome“ Assistenz

Die historisch frühesten, am weitesten verbreiteten und in ihrer Wirkungsweise offensichtlichsten Assistenzsysteme stammen aus dem Bereich der Steuerung von technischen Geräten und Anlagen. Die fortschreitende Digitalisierung und Verbreitung von platzsparender Elektronik hat dazu geführt, dass viele Assistenzsysteme auch für private Nutzer finanzierbar und einsetzbar werden. Vor allem im Automobilbau ist die Idee des Assistenten sehr weit verbreitet. Benmimoun et al. [2] führen wichtige Fahrerassistenzsysteme auf. Fast alle dieser Assistenten zeichnen sich durch autonome Funktionalität aus. Die bekanntesten Beispiele unter ihnen sind: adaptives Kurvenlicht, Reifendruckkontrollsys-

teme, Antiblockiersysteme, elektronische Stabilitätsprogramme, Kollisionswarnsysteme und seit kürzester Zeit auch Nachtsichtsysteme.

Assistenten dieses Typs führen immer genau eine Funktion aus und unterstützen damit den Fahrer bei der Kontrolle seines Fahrzeugs während der Fahrt. Die Assistenten verbessern die Möglichkeiten des Fahrers zur Aufnahme von Informationen und damit auch zur Erkennung der aktuellen Situation und tragen damit zur Erhöhung der Datensicherheit bei, indem sie die sensorischen Fähigkeiten des Fahrers ergänzen. Nach der Klassifikation von WANDKE leisten Fahrerassistenzsysteme also genau in den Handlungsweisen der Aufnahme und der Integration von Information Assistenz. Sie tragen jedoch nicht zur Entscheidung, zur Aktionsausführung und zur Kontrolle der Handlungseffekte bei. Manche der Systeme sind aktive Assistenzsysteme, beispielsweise das Antiblockiersystem, andere wiederum sind passiv wie etwa ein Nachtsichtgerät, das nur Informationen zur Verfügung stellt, also eine Anzeigefunktion wahrnimmt, aber keine weitere Assistenz leistet. Der Benutzer hat in der Regel keine Möglichkeit, das Assistenzsystem in seiner Funktionsweise wesentlich zu beeinflussen.

2.1.2 Assistenz mit einem fixierten Ziel

Komplexere Assistenzsysteme stellen mehrere Assistenzfunktionen gleichzeitig zur Verfügung. Typische Beispiele hierfür sind Funktionen in komplexen Softwaresystemen. Dazu zählen Funktionen zum Druck von Dateien, die automatisch die Kommunikation mit dem auf dem System installierten Druckertreiber übernehmen genauso, wie Systeme zur Unterstützung der Produktionsplanung und -steuerung. Auch Software, die komplette Abläufe in der Betriebswirtschaft organisiert, wie etwa die Softwarepakete von SAP, leisten Assistenzfunktion, und zwar bei der Ausführung von Aktionen und bei der Kontrolle von Effekten. Jedes Expertensystem, wie es aus der Künstlichen Intelligenz bekannt ist (siehe [3–5]), ist ein Assistenzsystem, da es bei Handlungen Entscheidungsunterstützung anbietet. Offensichtlich ist die Liste an Beispielen beliebig verlängerbar, ebenso wie die Liste an Handlungen, die Menschen mit der Hilfe von Werkzeugen durchführen.

Typisch für alle Assistenzsysteme mit fixiertem Ziel ist jedoch, dass sie nicht für andere als die vorgesehenen Aufgaben eingesetzt werden können. Der Grad an Konfigurierbarkeit des Systems ist sehr unterschiedlich, je nachdem, wie komplex das Assistenzsystem ist. Ein bekanntes Beispiel für einen Assistenten, der verschiedene Funktionen kombiniert, ist das Microsoft-Programm *Outlook* (siehe Abb. 2.1). Es integriert Verwaltung von E-Mails, Kontakte, zu erledigende Aufgaben und einen Terminkalender. Anders als autonome Assistenzsysteme eignen sich als Softwaresysteme realisierte Assistenten für beliebig komplexe und parametrisierbare Assistenzfunktionen. Ein weiteres Charakteristikum: viele Assistenzsysteme, die bei komplexen Abläufen Unterstützung leisten, kommunizieren mit dem Benutzer, um Informationen über die aktuelle Situation zu erhalten – ein unverzichtbarer Vorgang, wie folgendes Beispiel illustriert: ein Buchhaltungsprogramm kann nur in Aktion treten, wenn Buchungsvorgänge, die außerhalb der Aktivität des Pro-

Lumière während des Betriebs einer Office-Applikation versucht, aus den Aktionen des Benutzers seine aktuellen Ziele zu erschließen. Damit dies möglich wird, müssen die über die Benutzerschnittstelle beobachtbaren Aktionen wie das Bewegen der Maus, das Aufrufen eines Menüpunkts, das Aktivieren eines Fensters und auch Untätigkeit des Nutzers über einen gewissen Zeitraum in Bezug auf Handlungen, die mit Office-Applikation durchgeführt werden können, interpretiert werden. Die pragmatische Interpretation von Aktionen auf der Ebene der graphischen Benutzeroberfläche umfasst Themen wie Suche, Verschieben des Aufmerksamkeitsfokus, Nachdenken über die nächsten Schritte und das Rückgängigmachen erwünschter Effekte. Alle genannten Themen beschreiben Problemlösestrategien der Anwender für komplexe BediENAufgaben. Die Strategien wurden experimentell aus der Analyse von vielen aufgezeichneten Sitzungen von Probanden mit *Lumière* gewonnen. Dabei mussten Probanden die Tabellenkalkulation Excel benutzen und konnten auf einem zweiten Bildschirm die Assistenzvorschläge eines neuartigen Systems betrachten und in ihre Vorgehensweise bei der Lösung der Aufgabe einbeziehen. Tatsächlich jedoch wurden die Vorschläge nicht von einem softwarebasierten Hilfesystem ermittelt, sondern von einem menschlichen Experten, einem so genannten *Wizard-of-Oz* (siehe [7]), der Probanden auf einem getrennten eigenen Monitor beobachten konnte. Aus diesen Studien konnten die Entwickler von *Lumière* Problemlösestrategien ungeübter Nutzer identifizieren, die von einem Experten viel einfacher und schneller gelöst werden können.

Auffällig an den Benutzerstudien war, dass Experten zwar die Fähigkeit haben, Ziele und Bedürfnisse von Nutzern bei der Lösung einer wichtigen Aufgabe anhand der von den Nutzern durchgeführten Aktionen zu identifizieren. Aber es zeigte sich auch, dass die Experten typischerweise bei Beginn einer Aktionssequenz sehr unsicher waren, welches Ziel der Nutzer gerade verfolgte. In dieser Phase konnten selbst Experten nur schlecht Assistenz leisten. Erst nach einigen Bedienschritten verbesserte sich die Situation, und die Vorschläge wurden präziser und zielgerichteter. Die Studien zeigten darüber hinaus, dass schlechte Assistenz den Benutzern erhebliche Schwierigkeiten bereitete. Obwohl sie davon unterrichtet waren, dass sich das Hilfesystem in einem experimentellen Zustand befand, nahmen sie alle Vorschläge sehr ernst und bezogen sie in ihre Problemlösung mit ein. Falsche Vorschläge sind aber offensichtlich sehr abträglich für die effiziente Lösung einer Aufgabe. Diese Ergebnisse machen sehr deutlich, dass ein wesentliches Kriterium für ein leistungsfähiges Assistenzsystem in seiner Fähigkeit besteht, Vorschläge für Handlungen auf ein Ziel hin zu orientieren.

Aus technischer Sicht besteht die Schwierigkeit, präzise Vorhersagen über das vom Benutzer verfolgte komplexe Ziel zu treffen, wenn erst sehr wenige Schritte bekannt sind, darin, dass einzelne Aktionen mehreren typischen Bedienzeilen zugeordnet werden können. Dieser Nichtdeterminismus besteht schon bei der Zuordnung beobachtbarer Interaktionen zu einem komplexen Ziel. Wenn beispielsweise ein Nutzer mehrere Menüs kurz hintereinander aufruft, kann er sich eventuell unsicher sein, wie er eine bestimmte Programmfunktion aktivieren kann. Vielleicht will er aber auch bisher noch nicht

genutzte Funktionen entdecken – etwa deswegen, weil ihm nicht klar ist, mit welchen Schritten eine komplexe Aufgabe gelöst werden kann. Bei der Entwicklung von *Lumière* wurde daher versucht, diese Zusammenhänge mit Wahrscheinlichkeiten zu beschreiben.

Die Absichten des Nutzers rechtzeitig zu erkennen, ist, wie Studien mit Benutzern bei Microsoft ja gezeigt haben, ein entscheidender Faktor dabei, ob ein Assistenzsystem in der Lage ist, Nutzer dabei zu unterstützen, mithilfe eines Softwarewerkzeugs ihnen gestellte Aufgaben effizient zu lösen. Absichten des Nutzers zu erkennen, aber nicht überprüfen zu können, ob die von Nutzern gewählten Schritte zielführend sind, wäre absurd. Somit ist die Fähigkeit eines Assistenzsystems, Wege zu finden, wie in einer gegebenen Situation eine Aufgabe gelöst werden kann, eine ebenso wichtige Eigenschaft eines leistungsfähigen Assistenzsystems. Natürlich kann ein Assistenzsystem nicht jede denkbare Aufgabe erkennen, sondern muss sich darauf beschränken, bestimmte Aufgabenklassen, für die das Softwarewerkzeug ein Hilfsmittel sein kann, zu unterstützen. Bei der Konzipierung von Assistenz ist also sowohl der Abstraktion bei der Handlungsphase der Zielbildung wie auch der Ausführung von Aktionen eine Grenze gesetzt. Sie hängt davon ab, welche Art von Assistenz zur Verfügung gestellt, und wie detailliert Umsetzung der Assistenz in konkreten Situationen realisiert werden soll.

2.2 Assistenz aus der Sicht des Nutzers

Die Beispiele zeigen deutlich, dass eine Definition von Assistenz über das Spektrum der Funktionen, deren Durchführung ein Gerät oder System für den Nutzer erledigen soll, um damit eine Aufgabe zu lösen, wegen der endlosen Zahl an Möglichkeiten unmöglich ist.

Eine brauchbare Taxonomie darf aber nur eine endliche, ja sogar möglichst kleine Zahl von Unterscheidungen benötigen, um ein überschaubares Begriffssystem zu konstruieren. Aus diesem Grund schlägt Wandke [1] vor, eine Taxonomie von Assistenzsystemen auf einem anderen Ordnungsprinzip als der Funktionsvielfalt aufzubauen. Der entscheidende Punkt liegt laut WANDKE darin, ob ein Benutzer die von einem System angebotenen Funktionen überhaupt einsetzen kann beziehungsweise sie einzusetzen im Stande ist. Dies kann aus verschiedenen Gründen schwierig sein:

- Die vom Nutzer gewünschte Funktion steht überhaupt nicht zur Verfügung.
- Dem Nutzer sind nicht alle zur Verfügung stehenden Funktionen bekannt.
- Der Nutzer ist nicht in der Lage, eine Funktion durchzuführen, weil ein zu hoher sensorischer, kognitiver oder motorischer Aufwand erforderlich ist.

Die Zugänglichkeit von Funktionen ist aber von praktischer Relevanz, weil sie ermöglicht, ein technisches System vollständig, effektiv und effizient einzusetzen.

Assistenz hat also vorrangig die Aufgabe, die Kluft zwischen Systemfunktionalität und Fähigkeiten des menschlichen Nutzers zu überbrücken, d. h. also zu erreichen, dass

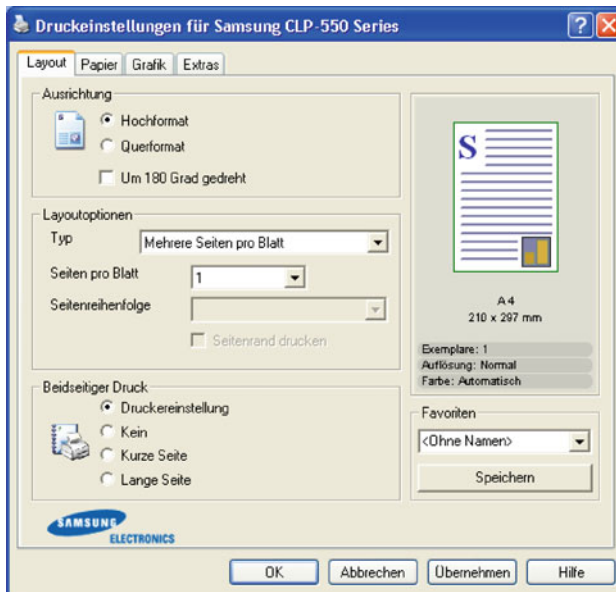


Abb. 2.2 Dialogfenster eines Druckertreibers: Mit Hilfe graphischer Mittel kann bei der Gestaltung von Mensch-Maschine-Schnittstellen dem Nutzer Assistenz bei der Interpretation interner Parameter in Bezug auf eine zu lösende Aufgabe geboten werden

der Nutzer die im Rahmen seines zielgerichteten Handelns notwendigen Funktionen des Systems mit möglichst geringem Aufwand einsetzen kann². Ein Beispiel dafür ist die in Abb. 2.2 zu sehende Schnittstelle, die Assistenz bei der Ansteuerung eines Druckers leistet.

Dass der Nutzer tatsächlich zielgerichtet handelt, ist eine Grundannahme für die Entwicklung interaktiver Assistenzsysteme. Sie lässt sich aus der einem Werkzeug – und genau das ist ja ein technisches Gerät oder Softwaresystem – inhärenten Existenzberechtigung ableiten:

- Ein Schraubenzieher dient dazu, eine Schraube in ein vorgebohrtes Loch hineinzudrehen, um zwei Werkstücke miteinander zu verbinden.
- Ein Flugzeug dient dazu, lange Distanzen in kurzer Zeit zu überwinden.
- Ein Textverarbeitungssystem dient dazu, Text einzugeben, zu formatieren, graphisch zu gestalten und abzuspeichern bzw. zu vervielfältigen.

Aus diesen Beispielen lässt sich verallgemeinern, was in dieser Arbeit unter einer *Aufgabe* und dem *Lösen einer Aufgabe* verstanden wird.

² Siehe dazu [8], Kap. 11 und [9].

Aufgabe (für die ein Assistenzsystem Unterstützung anbieten kann)

Seit Einführung des GOMS-Modells (*goals, operators, methods, and selection rules*) durch CARD, MORAN und NEWELL in [10] wird die kognitive Fragestellung, wie ein gegebener Zustand eines Systems durch Ausführung von Aktionen in einen gewünschten Zustand überführt werden kann, **Aufgabe** genannt.

Lösen einer Aufgabe (Problemlösung)

Das **Lösen einer Aufgabe** besteht in der Ermittlung einer Folge von Aktionen, deren Ausführung die beabsichtigte Zustandsüberführung herstellen kann, und in der Ausführung der ermittelten Folge von Aktionen unter Einbeziehung technischer Geräte bzw. Systeme als Werkzeuge.

Der zentrale Punkt in dieser Definition ist, dass der menschliche Nutzer für das Stellen der Aufgabe verantwortlich ist, und jedes eingesetzte technische Gerät daher nur eine untergeordnete Rolle als Hilfsmittel spielt. Die menschliche Problemlösekompetenz gibt also den Ausschlag dabei, wie die Lösung einer Aufgabe versucht werden soll. Für die Effizienz und Qualität der Lösung ist entscheidend, wie gut der Nutzer die verfügbaren Funktionen der einsetzbaren Hilfsmittel in den Lösungsablauf integrieren kann. Die Definition beschreibt also keine Systeme, die vollautomatisch funktionieren. Wesentlich ist aber, dass Assistenz in der Lage ist, sich an menschlichem Problemlösen und den dazugehörigen Problemlösestrategien zu orientieren, weil nur auf diese Weise sichergestellt werden kann, dass das technische System und ein dazugehöriges Assistenzsystem harmonisch in den Ablauf des Lösungsvorgangs integriert werden können³.

Bei der zweckrationalen Lösung von Aufgaben nehmen nach übereinstimmenden Analysen in der Literatur zur kognitiven Ergonomie (siehe [1]) sechs sich ergänzende Aspekte eine zentrale Stellung ein:

- **Motivation, Aktivierung und Zielsetzung**

Ohne Zielsetzung ist keine Aufgabe definiert; dass dieser Aspekt wichtig ist, leuchtet also unmittelbar ein. Motivation und Aktivierung sind jedoch zwei weitere wichtige Voraussetzungen für die Durchführung von – oft komplizierten – Aufgaben, die mit Hilfe von technischen Systemen gelöst werden sollen. Aktivierung spielt beispielsweise eine wesentliche Rolle bei der Überwachung von prozesstechnischen Anlagen. Im Regelfall läuft alles nach Plan, aber bei Störungen ist es für eine Minimierung von Folgeschäden ausschlaggebend, wie rasch das Personal auf die Störung aufmerksam wird, wie präzise der Gefährlichkeitsgrad eingeschätzt werden kann, und wie effizient

³ Siehe [8], Kap. 11, [11], Kap. 10, 14, und 15 sowie [10].

Gegenmaßnahmen ergriffen werden können. Immer wieder wird bei größeren technischen Störungen deutlich, dass sowohl zu niedrige als auch zu starke Aktivierung zu schwerwiegenden Konsequenzen führen, weil entweder gar keine oder – oft aus Panik – übertriebene oder sogar falsche Gegenmaßnahmen ergriffen werden. Die Kollision eines DHL-Transportflugzeugs mit einer russischen Tupolew Tu-154M der Bashkirian Airlines am 1. Juli 2002 ist ein Beispiel für die Schwierigkeit und Bedeutung der Aktivierung: weil der Fluglotse mit einem Landeanflug auf den Flughafen Friedrichshafen beschäftigt war, übersah er, dass beide Flugzeuge dieselbe Höhe hatten, und eine Kollision bevorstand.

- **Wahrnehmung**

Damit der Nutzer während der Bearbeitung einer Aufgabe zu jeder Zeit richtig reagieren kann, muss er Zugang zur relevanten Information haben. Die Wahrnehmung aller wichtigen Fakten zur richtigen Zeit kann oft schwierig sein, weil entweder der Nutzer mit der Verarbeitung anderer Signale überlastet ist (wie eben auch im Beispiel oben), oder Signale gar nicht oder nur schlecht wahrgenommen werden können (z. B. interner Zustand technischer Geräte, Hindernisse auf der Straße im Dunklen).

- **Integration von Information unter Berücksichtigung der aktuellen Situation**

Die bloße Wahrnehmung elementarer Signale reicht nicht aus, um eine Situation *interpretieren* zu können. Die Signale müssen mit Information aus dem (Langzeit)-Gedächtnis des Nutzers über geltende Tatsachen in Beziehung gebracht werden. Diese Interpretation muss auch erlauben, eine Bewertung der Zielorientierung vorzunehmen. Ein einfaches Beispiel dafür ist die Signalisierung einer Geschwindigkeitsüberschreitung durch ein Navigationssystem. Interpretation von Signalen erfordert also immer Bezugnahme auf Welt- bzw. Domänenwissen (der Messwert des Tachometers hat die Einheit km/h; damit wird eine Geschwindigkeit ausgedrückt; eine Geschwindigkeit von 70 km/h ist innerorts unzulässig, eine Geschwindigkeit von 200 km/h in einer engen Rechtskurve wird so große Fliehkräfte erzeugen, dass das Auto ins Schleudern geraten wird).

- **Fällen von Entscheidungen, Auswahl von Aktionen**

Aus der Interpretation von Information muss zur Lösung einer Aufgabe in zielorientierter Weise eine Reihe von Handlungen abgeleitet werden, mit deren Hilfe die Aufgabe gelöst werden kann – das heißt also, der Nutzer muss Entscheidungen über sein weiteres Vorgehen treffen und diese Entscheidungen auch umsetzen. Bei der Ermittlung erfolgversprechender Handlungen ist Assistenz besonders hilfreich. Assistenz kann darin bestehen, alle sinnvollen Entscheidungsoptionen anzubieten, aus allen Optionen ein(ig)e geeignete herauszufinden oder eine (optimale) Aktion autonom auszuführen. Diese Form von Assistenz sorgt für den denkbar höchsten Automatisierungsgrad.

- **Ausführung von Aktionen**

Werden Aktionen nicht autonom vom Assistenzsystem ausgeführt, so kann auch bei der Ausführungen von Aktionen Assistenz geleistet werden. Beispiele dafür sind Bremsassistenten oder elektronische Vorrichtungen zur Motordrosselung bei Erreichen der erwünschten Höchstgeschwindigkeit in besonders stark motorisierten Autos. Allgemein

geht es bei diesem Assistenztyp meistens um die „Feindosierung“ bei der Steuerung von Aktoren oder um die Vereinfachung der Mensch-Maschine-Interaktion (beispielsweise Spracheingabe oder multimodale Eingabe für Behinderte).

- **Verarbeitung von Effekten der Aktionen und *feedback* darauf**

Die Effekte von Handlungen sind oft durch Wahrnehmung beobachtbar. Zur Kontrolle der Wirkung einer Aktion ist es entscheidend, überprüfen zu können, ob diese Wahrnehmungen mit der erwarteten Wirkung der Handlung übereinstimmen. Nicht immer ist eine direkte Wahrnehmung möglich, über die auf Erfolg oder Misserfolg einer Handlung geschlossen werden kann. Assistenz in dieser Handlungsphase bezieht sich also oft auf die Erweiterung der Möglichkeiten zur Wahrnehmung (Ist ein Datum in einem Gerät abgespeichert worden, Ist beim Einparken der Abstand zur Hausmauer noch ausreichend?). Eine erweiterte Assistenz zur Effektkontrolle bewertet die Auswirkungen von Handlung bezüglich einer zu lösenden Aufgabe.

In Tab. 2.1 wird ein Überblick über die verschiedenen Handlungsphasen, die damit verbundenen Aspekte eines Problemlöseprozesses und dafür vorstellbare Typen von Assistenz gegeben. Die Tabelle fasst die von Wandke [1] erstellte Taxonomie von Assistenzfunktionen zusammen. WANDKE legt den Fokus seiner Untersuchungen auf die Frage, welche Typen von Assistenz es grundsätzlich gibt, ohne Aspekte der Implementierung von Assistenzsystemen auf der Basis von Software zu erörtern. Dies ist angebracht, da Assistenzsysteme, wie die bereits besprochenen Beispiele deutlich machen, nicht nur im Bereich von Softwarelösungen vorstellbar sind. Historisch gesehen ist es sogar genau andersherum: Assistenzsysteme gab es zunächst bei technischen, insbesondere großen technischen Geräten und Apparaturen. Für die Künstliche Intelligenz steht hingegen die Frage im Vordergrund, wie man die vorgeschlagenen Assistenztypen algorithmisch realisiert könnte. Diese Frage stellt das Kernproblem dar, das in diesem Buch detailliert zu besprechen sein wird. In Abschn. 2.4 werden zunächst noch sehr allgemein, ohne auf algorithmische Details einzugehen, Anforderungen an die Algorithmik eines Assistenzsystems aufgeführt, die sich aus WANDKES Typen und einigen weiteren Anforderungen, die im Folgenden entwickelt werden, ableiten lassen.

Bevor dies jedoch geschieht, ist ein weiterer wichtiger Aspekt zu betrachten. Es handelt sich darum, *wann* ein Assistenzsystem Assistenz leisten soll: Nach Nitschke [12] hat jeder der Aspekte aus Tab. 2.1, sobald er in einem Assistenzsystem verfügbar ist, noch zwei Dimensionen, wie er ausgestaltet sein kann:

- **Initiative:** Die Assistenz kann *passiv* (also nur auf explizite Anforderung des Nutzers) oder *aktiv* (also auch autonom) ausgeführt werden.
- **Anpassbarkeit:** Die Assistenzfunktionen in einem System können *statisch* (d. h. nicht veränderbar), *adaptierbar* (d. h. durch den Nutzer parametrisierbar) oder *adaptiv* (d. h. vom System selbst anhand von Informationen aus dem Kontext adaptierbar) sein.

Tab. 2.1 Taxonomie für Assistenzfunktionen. Jede Phase einer zielorientierten Handlung benötigt eine spezielle Form der Unterstützung, die von einem Assistenzsystem angeboten werden kann

Motiv- und Zielbildung	
Schaffung eines optimalen Aktivierungsniveaus	Aktivierungsassistentz
Verstärkung eines Motivs	Coach-Assistentz
Hemmung eines Motivs	Warn-Assistentz
Anregung eines Zielwechsels	Orientierungsassistentz
Informationsaufnahme	
Bereitstellung von Signalen	Anzeigefunktion
Signalverstärkung	Verstärkungsassistentz
Erzeugung von Redundanz	Wiederholungsassistentz
Transformation von Signalen in andere Modalitäten	Präsentationsassistentz
Integration von Information, Berücksichtigung der aktuellen Situation	
Bereitstellung von Erklärungen	Beschriftungen, Anleitungen, Hilfetexte
Bereitstellung externer Bezugssysteme	Übersetzungsassistentz
Erklärung von Systemausgaben	Erklärungsassistentz
Entscheidung über Auswahl einer Aktion	
Information über alle Optionen	Angebotsassistentz
Information über ausgewählte Optionen	Filterassistentz
Vorschlag einer Option	Beraterassistentz
Vorschlag und Ausführung, wenn der Nutzer zustimmt	Delegationsassistentz
Vorschlag und Ausführung, wenn der Nutzer nicht widerspricht	Übernahmeassistentz
Ausführung mit Information an den Nutzer	Informierende Ausführungsassistentz
Ausführung ohne Information an den Nutzer	Stille Ausführungsassistentz
Aktionsausführung: Wie soll die Aktion durchgeführt werden?	
Verstärken von Aktionen	<i>Power</i> -Assistentz
Verkürzen einer Aktionsfolge	<i>Short cut</i> -Assistentz
Alternative Modalitäten bereitstellen	Eingabeassistentz
Effektkontrolle	
Auswirkungen wahrnehmbar machen	Rückmeldungsassistentz
Grad der Zielerreichung bewerten	Kritikassistentz

2.3 Anforderungen an Assistenzsysteme

Die Frage, wann Assistenz zu leisten ist, lässt sich ähnlich schwierig beantworten, wie die Fragen *wozu* und *wie*, die am Beispiel des *Lumière*-Projekts bereits erörtert wurden: Denn auch die Auswahl des richtigen Zeitpunkts ist eine Entscheidung, die von unsicherer Information über den Nutzer bestimmt wird. Ein Assistenzsystem muss dabei zwei Aspekte unter einen Hut bekommen: einerseits die Integration von Wahrnehmungen in den Entscheidungsprozess, obwohl sie oft zu mehrdeutigen Interpretationen Anlass

geben. Andererseits ist das Assistenzsystem gezwungen, eine Assistenzstrategie umzusetzen, die zu einer eindeutig festgelegten nächsten Handlung führt, aber auch Kriterien der Adäquatheit von Assistenz und Kooperation berücksichtigt (siehe [8], Kap. 11). Dieser immer präsente Umstand beeinflusst die algorithmische Umsetzung der in Tabelle 2.1 aufgeführten Assistenzfunktionen. Um in dieser Situation dennoch einer algorithmischen Realisierung von Assistenz näher zu kommen, soll im Folgenden diskutiert werden, welche Inferenz-Fähigkeiten ein Assistenzsystem besitzen muss, um Assistenzfunktionen samt der dafür notwendigen Entscheidungen über Zeitpunkt, Zweck sowie Art und Weise ihrer Durchführung ausführen zu können.

2.3.1 Interaktivität

Auch wenn die Interpretation von Beobachtungen nicht eindeutig ist, muss sich ein Assistenzsystem für eine Hypothese entscheiden, kann sich aber nicht sicher sein, welche die richtige ist. Wie kann es diesem Dilemma entgehen? Eine Möglichkeit ist, die aktuelle Situation wiederzuerkennen und sich zu merken, wie oft in dieser Situation bei Ausführen einer bestimmten Handlung welcher Effekt eingetreten ist. Wenn das Assistenzsystem nun die Wahrnehmung so interpretiert, dass es denjenigen Effekt als gegeben ansieht, der bisher unter identischen Umständen am häufigsten aufgetreten ist, trifft es eine rationale Entscheidung, indem es die Erwartung maximiert, die richtige Hypothese ausgewählt zu haben.

Die beschriebene Strategie soll am Beispiel des Fußgängernavigationssystems ROSE⁴ diskutiert werden. ROSE aktualisiert während der Navigation zu einem Ziel ständig ein Benutzermodell, mit dessen Hilfe das aktuelle Interesse des Nutzers abgeschätzt werden soll, dass er weiterhin Assistenz zum Erreichen seines Ziels benötigt. Für den Interessen-Status des Nutzers werden dabei folgende vier möglichen Werte angenommen:

- *c*: Das Ziel soll noch erreicht werden.
- *i*: Der Nutzer möchte zwischenzeitlich ein anderes Ziel erreichen.
- *a*: Der Nutzer hat das Ziel aufgegeben.
- *e*: Der Nutzer wird von einem äußeren Umstand aufgehalten.

Hypothesen über den Status gewinnt ROSE, indem es aus Wahrnehmungen inferiert. Wie beim Office-Assistenten besteht auch in dieser Anwendung die Schwierigkeit, dass die Wahrnehmungen keine eindeutigen Schlüsse auf den Status zulassen. ROSE hat nämlich im Wesentlichen nur die Möglichkeit, die GPS-Koordinaten für den aktuellen Standort des Nutzers festzustellen. Aus dieser Information lassen sich unter Berücksichtigung einer Zeitreihe von GPS-Koordinaten und Karteninformation über den aktuellen Standpunkt folgende Daten ermitteln:

⁴ Das am Lehrstuhl für Künstliche Intelligenz entwickelte System ist u. a. in [13] beschrieben.

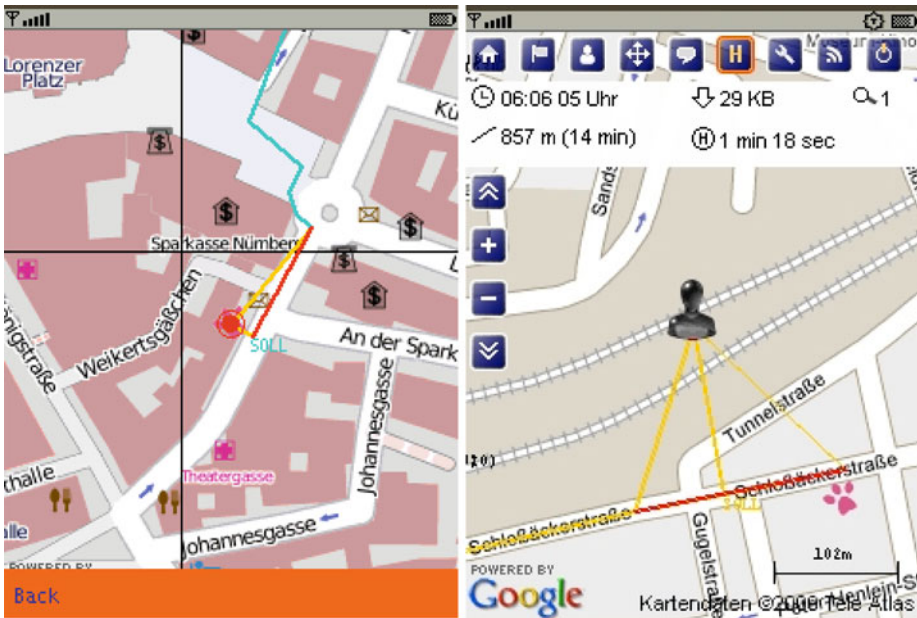


Abb. 2.3 Präsentationsassistenz des mobilen Auskunftssystems ROSE. Der linke *screen shot* zeigt, wie ROSE GPS-Daten in eine für den Nutzer leichter erfassbare Modalität übersetzt und mit Zusatzinformation über nahe liegende *points of interest* anreichert. Der rechte *screen shot* illustriert eine Warnassistenzfunktion von ROSE: auf dem Bildschirm wird graphisch verdeutlicht, dass der Nutzer sich nicht mehr auf dem direkten Weg zum Ziel befindet

- Ist der Nutzer gerade in Bewegung oder steht er an einem Ort?
- Befindet sich in der Nähe des Nutzers ein für ihn interessanter Ort?

ROSE zeigt diese Information für den Nutzer auf dem Display seines Mobilgeräts an, wie im linken *screenshot* in Abb. 2.3 zu sehen. Über diese reine Präsentationsassistenz hinaus kann ROSE auch noch ermitteln, ob sich der Nutzer auf dem Weg zum Ziel befindet oder nicht (siehe rechten *screenshot* in Abb. 2.3).

Desweiteren kann ROSE den Nutzer durch geeignete Hinweise auf seinem Weg zum Ziel unterstützen. Die Effekte dieser Form von Assistenz kann ROSE auch nur mittelbar erschließen: wenn der Nutzer sich ungefähr auf dem Weg zum Ziel befindet, hat er die Hinweise wohl verstanden; ist die Abweichung groß, dann gibt es ein Missverständnis zwischen ROSE und dem Nutzer. Es sei denn, es ist anzunehmen, dass der Nutzer gar kein Interesse mehr am Ziel hat.

Der eben beschriebene Zusammenhang zeigt, wieso das oben eingeführte NutzermodeLL für ROSE so bedeutsam ist: wenn ROSE nämlich ein Missverständnis oder eine Änderung in den Interessen des Nutzers vermutet, ist es nicht sinnvoll, die Effektkontrolle weiter durchzuführen. Vielmehr muss ROSE andere Assistenzfunktionen aktivieren: die

Ziel- und Motivbildung spielt in diesem Zustand wieder eine Rolle. Wenn ROSE nämlich vermutet, dass der Nutzer das Interesse am bisherigen Ziel aufgegeben hat, ist Orientierungsassistentz vorrangig: soll das Ziel gewechselt werden? Wird der Nutzer vermutlich aufgehalten, ist Warnassistentz angebracht: der Nutzer muss daran erinnert werden, dass er zu einem bestimmten Zeitpunkt (bei ROSE meist vor Abfahrt des bei der Zielbildung ausgewählten Nahverkehrsmittels) am Ziel ankommen muss.

Die Diskussion verdeutlicht, dass eine falsche Hypothese über den Interessen-Status des Nutzers dazu führen kann, dass ROSE sämtliche folgenden Handlungen des Nutzers falsch interpretiert, und alle seine Versuche zu assistieren, nur zu Verwirrung führen. Andererseits aber kann ROSE anhand seiner autonom erfassbaren Wahrnehmungen nie so viel Information ansammeln und damit die Unsicherheit über den Interessen-Status so weit reduzieren, dass eine Fehlinterpretation fast sicher ausgeschlossen ist.

Der einzige Ausweg aus diesem Dilemma besteht im Nachfragen beim Nutzer. ROSE muss also in der Lage sein, mit dem Nutzer zu kommunizieren, um sinnvoll Assistenz leisten zu können.

Wie schwierig die Analyse des Interessen-Status nur aufgrund autonom erfasster Wahrnehmungen ist, veranschaulicht das folgende Beispiel für die Implementierung des ROSE-Benutzermodells.

In Abb. 2.4 ist ein Ausschnitt aus dem Benutzermodell von ROSE zu sehen⁵: Um Hypothesen für den Interessen-Status des Nutzers zu gewinnen, bedient sich ROSE eines BAYES-Netzes⁶. In das Netz geht einerseits über Sensoren erfassbare Information ein:

- **Lokalisierung:** Damit wird eine Information bezeichnet, die aus den GPS-Daten ermittelt werden kann: ist der Nutzer in Bewegung oder nicht?
- **POI nahe:** Auch dies ist eine binäre Variable; sie ist wahr, wenn sich in der Nähe der aktuellen Position des Nutzers ein für den Nutzer relevanter *point of interest* befindet, und andernfalls falsch.

Im BAYES-Netz sind andererseits aber auch weitere Größen enthalten, die nicht direkt beobachtbar sind, aber einen Einfluss auf den Interessen-Status des Nutzers haben können:

- **unerwartetes Ereignis:** Während der Nutzer unterwegs zum Ziel ist, ereignet sich ein Vorfall, der den Nutzer darin hindert, weiterzugehen: Z. B. ein unübliches Ereignis auf der Straße; der Nutzer kann stürzen; der Nutzer kann gezwungen sein, einen Umweg zu nehmen. Die Liste möglicher Vorfälle kann beliebig verlängert werden. Allen unerwarteten Ereignissen ist jedoch gemeinsam, dass sie den Nutzer auf dem Weg zum Ziel aufhalten.

⁵ Alle BAYES-Netze in diesem Kapitel wurden mit Hilfe des unter <http://aispace.org/bayes/help/> verfügbaren Tools erstellt, das in [14] beschrieben ist (Letzter Aufruf der genannten Webseite am 03.01.2015).

⁶ BAYES-Netze werden ausführlich in [15], [16] und [17] dargestellt.

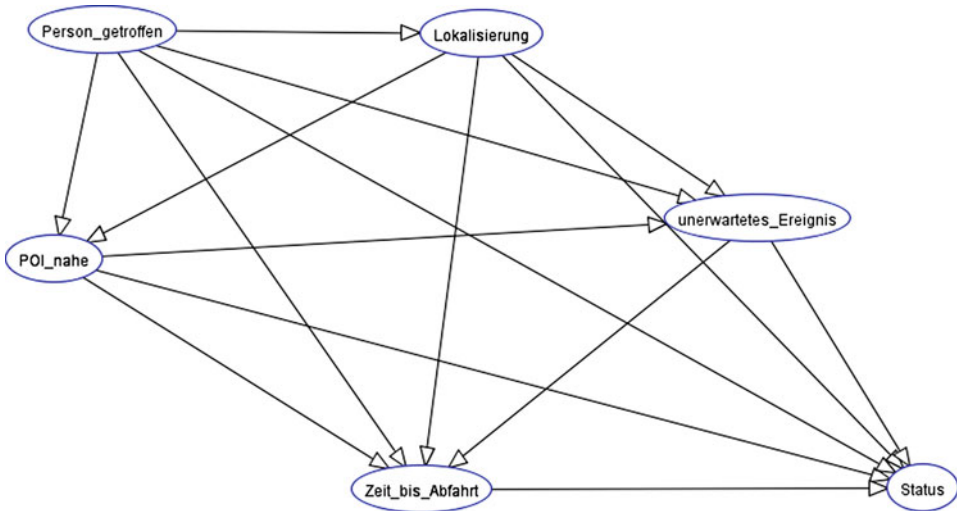


Abb. 2.4 Das Benutzermodell von ROSE: Ein BAYES-Netz repräsentiert die kausalen Abhängigkeiten von beobachtbaren Wahrnehmungen und daraus ableitbaren Interpretationen über den Zustand der Umgebung und den Interessen-Status des Nutzers

- *Person getroffen*: Eine Art unerwarteter Ereignisse verdient eine separate Erwähnung, da sie besonders häufig auftreten kann, wenn der Nutzer ROSE an seinem Wohnort einsetzt. Wer eine bekannte Person – vielleicht sogar nach längerer Zeit wieder – mehr oder weniger zufällig unterwegs trifft, hält sich gerne einige Zeit für eine Unterhaltung auf. ROSE sollte dies bei der Effektkontrolle berücksichtigen und den Nutzer in einer derartigen Situation nicht zum Weitergehen auffordern, wenn die Zeit noch nicht wirklich knapp ist.

Das BAYES-Netz in Abb. 2.4 zeigt eine denkbare Analyse, wie sich die eben beschriebenen Faktoren gegenseitig beeinflussen. Das BAYES-Netz enthält dazu gerichtete Kanten, die wie eine Implikation gelesen werden. Ein Beispiel: Wenn der Nutzer eine *Person getroffen* hat, dann ist eher zu erwarten, dass die *Lokalisierung* keine Bewegung meldet. Wie stark diese Annahme ist, wird in der Theorie der BAYES-Netze durch die bedingte Wahrscheinlichkeit

$$P(\text{Lokalisierung} = \text{Nutzer steht} | \text{Person getroffen} = \text{T})$$

ausgedrückt. Hier wird die Wahrscheinlichkeit, dass eine bestimmte Information zur Lokalisierung vorliegt, in Abhängigkeit vom Wissen, dass der Nutzer ein bestimmtes Interesse verfolgt, ermittelt. Dies widerspricht der tatsächlichen Situation: Dass die ROSE kann nur aus der *beobachtbaren Wahrnehmung* (Lokalisierung) die Wahrscheinlichkeit der *nicht beobachtbaren Information* (Nutzerinteresse) erschließen. Dies geschieht mit

Hilfe des Theorems von BAYES, das bekanntermaßen folgende Umformung ermöglicht:

$$\begin{aligned}
 & P(\text{Person getroffen} = \top | \text{Lokalisierung} = \text{Nutzer steht}) \\
 &= \frac{P(\text{Lokalisierung} = \text{Nutzer steht} \wedge \text{Person getroffen} = \top)}{P(\text{Lokalisierung} = \text{Nutzer steht})} \\
 &= \frac{P(\text{Lokalisierung} = \text{Nutzer steht} \wedge \text{Person getroffen} = \top)}{\sum_{b \in \{\top, \perp\}} P(\text{Person getroffen} = b \wedge \text{Lokalisierung} = \text{Nutzer steht})} \\
 &= \frac{P(\text{Lokalisierung} = \text{Nutzer steht} | \text{Person getroffen} = \top)}{\sum_{b \in \{\top, \perp\}} P(\text{Lokalisierung} = \text{Nutzer steht} | \text{Person getroffen} = b)}
 \end{aligned}$$

Nun kann aus den GPS-Daten darauf geschlossen werden, wie wahrscheinlich die Vermutung ist, dass der Nutzer eine Person getroffen hat. Voraussetzung dafür ist jedoch, dass die Wahrscheinlichkeitsverteilung

$$P(\text{Person getroffen} = x_1 \wedge \text{Lokalisierung} = x_2)$$

für alle möglichen Werte x_1 , die für die Größe *Person getroffen* erschlossen werden können, und für alle möglichen Werte x_2 , die für die Größe *Lokalisierung* wahrgenommen werden können, bekannt ist. In der Praxis ermittelt man diese Wahrscheinlichkeitsverteilung dadurch, dass man in Experimenten zu bestimmten Zeitpunkten und in regelmäßigen Abständen in einem Simulationsexperiment oder während des *live*-Betriebs von ROSE die beiden aktuellen Werte bestimmt. Aus einer großen Zahl solcher Messungen ergibt sich eine Datensammlung, aus der nach der Methode der Maximum-Likelihood-Schätzung⁷ die gesuchte Wahrscheinlichkeitsverteilung approximiert wird.

Auf die beschriebene Art und Weise kann – mit entsprechend mehr Aufwand bei der Datenerhebung – auch die Wahrscheinlichkeitsverteilung für die Größe *Status* bestimmt werden. Analoges gilt natürlich für andere Größen, die ebenfalls nicht unmittelbar mit Wahrnehmungen assoziiert sind (im Beispiel ist unerwartetes Ereignis eine derartige Größe).

In Abb. 2.5 ist zu sehen, wie ein BAYES-Netz, wenn es wie oben beschrieben konstruiert worden ist, dazu eingesetzt werden kann, eine Hypothese über den Interessen-Status des Nutzers abzuleiten. Im Beispiel in der Abbildung wird angenommen, dass sich der Nutzer bewegt (*Lokalisierung* = \top), kein *point of interest* in der Nähe ist (*POI nahe* = \perp) und nur noch wenig Zeit bis zur Abfahrt des bei der Zielfindung avisierten Verkehrsmittels zur Verfügung steht (*Zeit bis Abfahrt*: w). Aus diesen Informationen und den entsprechend der Graphstruktur ermittelten Wahrscheinlichkeitsverteilungen können nun Annahmen über alle nicht beobachteten Größen erschlossen werden.

⁷ Eine Einführung in die Maximum-Likelihood-Schätzung geben [18] und [19].

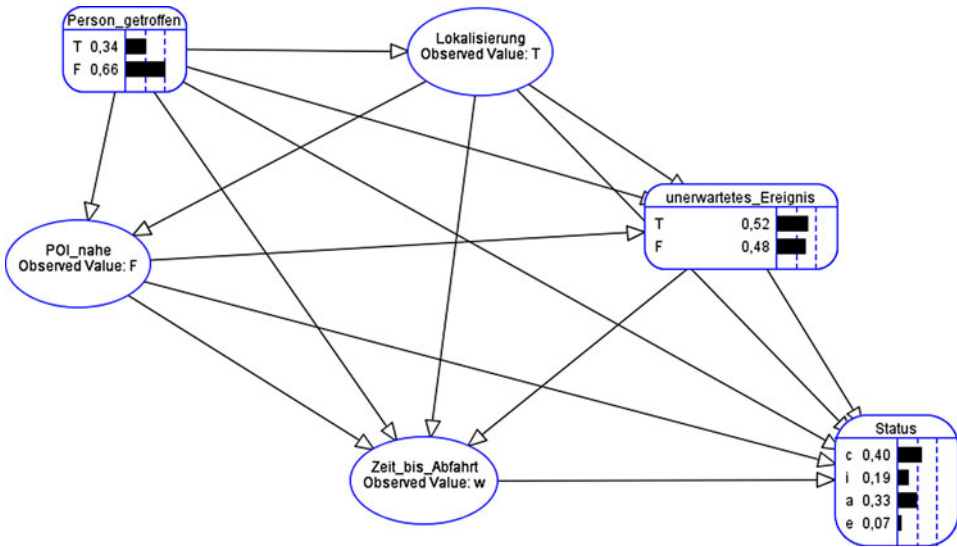


Abb. 2.5 Erschließen von Information über den Nutzer anhand beobachteter Wahrnehmungen und aufgrund der gelernten Wahrscheinlichkeitsverteilungen erschlossener Aussagen

Die Wahrscheinlichkeit, eine Person getroffen zu haben, liegt bei 34 %; die Wahrscheinlichkeit eines unerwarteten Ereignisses bei 52 %. Für den eigentlich gesuchten Interessen-Status des Nutzers gilt:

$$P(c) = 0,4, \quad P(i) = 0,19, \quad P(a) = 0,33, \quad P(e) = 0,07$$

Die wahrscheinlichste Hypothese ist also, dass der Nutzer das Ziel noch erreichen will. Nicht viel weniger wahrscheinlich ist allerdings die Hypothese, dass der Nutzer das Ziel aufgegeben hat. Die Wahrscheinlichkeit für ein temporäres Zwischenziel beträgt 19 %, diejenige dafür, dass der Nutzer durch äußere Umstände aufgehalten wird, nur 7 %.

Nach einer *maximum a-posteriori*-Entscheidung⁸ nimmt das Assistenzsystem also an, dass der Nutzer weiterhin auf direktem Weg zum Ziel kommen möchte.

In Abb. 2.6 ist eine andere Situation skizziert, in der Hypothesen für den Interessen-Status des Nutzers gesucht werden.

In dieser Situation ist nicht nur der Nutzer in Bewegung, sondern auch ein *point of interest* in der Nähe. Dieser Unterschied in den Beobachtungen ruft eine deutliche Änderung in den Wahrscheinlichkeiten für die verschiedenen Hypothesen für den Interessen-Status des Nutzers hervor. Die Wahrscheinlichkeit dafür, dass der Nutzer das Ziel aufgegeben hat, steigt auf 83 %. Die drei anderen Hypothesen sind wesentlich unwahrscheinlicher. Eine Entscheidung nach der *maximum a-posteriori*-Strategie ist also erheblich zuverlässiger als in der Situation aus Abb. 2.5.

⁸ Siehe [20] für eine detaillierte Einführung in und Diskussion von MAP.

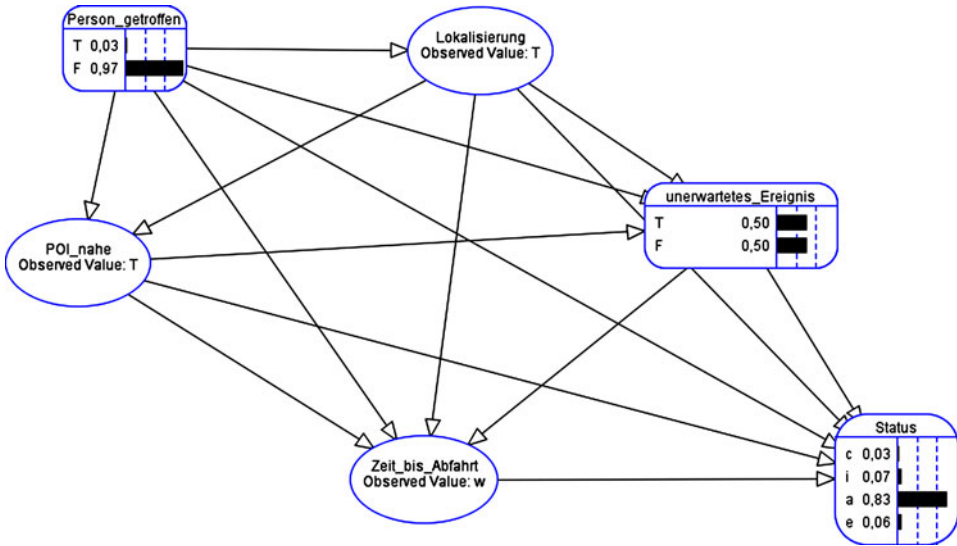


Abb. 2.6 Dieses Beispiel zeigt die a-posteriori-Wahrscheinlichkeiten der nicht beobachtbaren Größen im ROSE-Benutzermodell, wenn sich der Nutzer bewegt, und ein *point of interest* in der Nähe ist

Noch risikoreicher, weil fehleranfälliger, ist die *maximum a-posteriori*-Strategie in Abb. 2.7. In dieser Situation ist kaum mehr Zeit bis zur Abfahrt, der Nutzer bewegt sich nicht auf das Ziel zu, und es ist auch kein *point of interest* in der Nähe. Die Wahrscheinlichkeiten für die Hypothesen zum Interessen-Status des Nutzers liegen bei 27 %, 28 %, 26 % bzw. 19 %, unterscheiden sich also teilweise maximal im Bereich von Zehntel Prozentpunkten. Dass die Werte so eng beieinander liegen, ist anhand der beobachteten Information nachvollziehbar: der Nutzer bewegt sich nicht, obwohl die Abfahrt des von ihm gewählten Verkehrsmittels kurz bevorsteht – warum bleibt er stehen? Es ist sicher kein *point of interest* in der Nähe, die Wahrscheinlichkeit für ein unerwartetes Ereignis beträgt $\frac{1}{2}$, der Nutzer könnte also höchstens einen Bekannten getroffen haben: dafür beträgt die Wahrscheinlichkeit immerhin 71 %. Daher nimmt die Wahrscheinlichkeit dafür, dass der Nutzer das Ziel zwischenzeitlich nicht verfolgt, mit 28 % den höchsten Wert an; der Nutzer ist aber so oft schon weitergelaufen, als er unter Zeitdruck einen Bekannten traf, dass die Wahrscheinlichkeit dafür bei 27 % liegt. Die Differenz von einem Prozentpunkt ist so gering, dass das Risiko der *maximum a-posteriori*-Strategie, einem Messfehler des GPS-Systems aufzusitzen und in Konsequenz dieser Tatsache in ungeeigneter Weise Assistenz bereitzustellen, extrem hoch ist. Schließlich liegt der Unterschied in den Wahrscheinlichkeiten der drei am besten bewerteten Hypothesen innerhalb der Standardabweichung und daher im Rahmen des durch Rauschen im Datencorpus verursachten Fehlers.

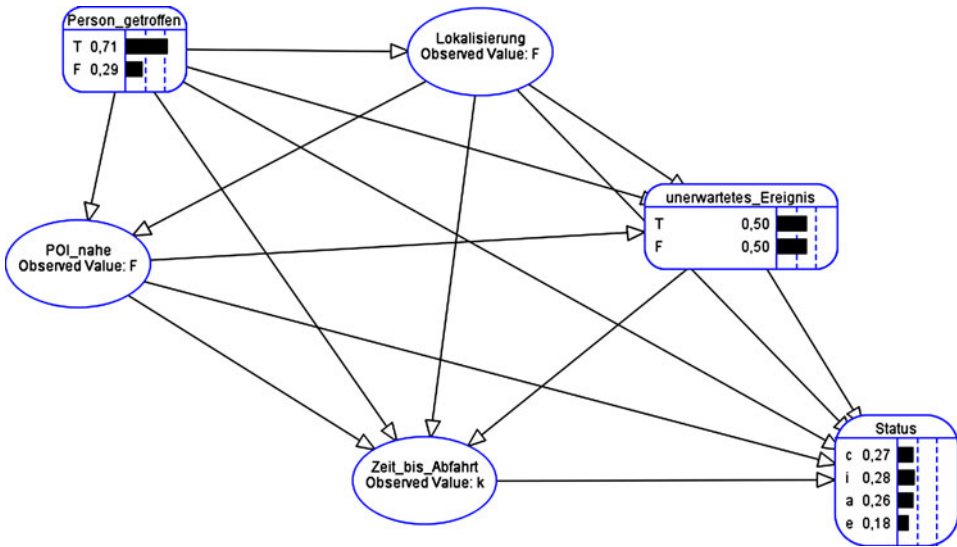


Abb. 2.7 Situation, in der für ein Assistenzsystem eine autonome Bestimmung des Interessen-Status des Nutzers unmöglich ist

Welche Mittel stehen einem Assistenzsystem in einer derartigen Situation noch zur Verfügung, um seinem Verwendungszweck gerecht zu werden, den Nutzer bei der Durchführung seiner Aufgabe zu assistieren?

Offensichtlich benötigt das Assistenzsystem mehr Information über die aktuelle Situation. Da ihm aber keine weiteren Sensoren mehr zur Verfügung stehen, ist es unmöglich, dass sich das Assistenzsystem sozusagen selbst hilft, indem es die Situation durch zusätzliche Wahrnehmungen präziser beobachtet als bisher.

So bleibt nur noch die Möglichkeit, sich vom Nutzer „assistieren“ zu lassen und Information, über die der Nutzer eventuell verfügt, von ihm zu erfragen. Statt eines Sensors bedient sich das Assistenzsystem also der Interaktion, um mehr Wissen über die aktuelle Situation zu erhalten.

Die diskutierten Beispiele illustrieren die Tatsache, dass Assistenzsysteme die Fähigkeit zur Interaktion als konstituierenden Bestandteil besitzen müssen. Wenn nämlich Entscheidungen über Form und Inhalt, wie Assistenz geleistet werden soll, von Tatsachen abhängen, die nicht beobachtet werden können, muss der Nutzer sie durch eigenes Nachdenken, also durch Introspektion, ermitteln und dem Assistenzsystem kommunizieren. Abbildung 2.8 zeigt, welche Auswirkung es hat, wenn der Nutzer nicht beobachtbare Information kommuniziert: Die Variable *Person getroffen* hat nun einen bekannten Wert mit $P(\text{Person getroffen} = \perp) = 1$. Diese neue Information beeinflusst die Bewertung des Nutzerstatus. Dass das Ziel noch erreicht werden soll, wird mit einer Wahrscheinlichkeit von 38 % angenommen, mit 11 % wird eine Unterbrechung vermutet.

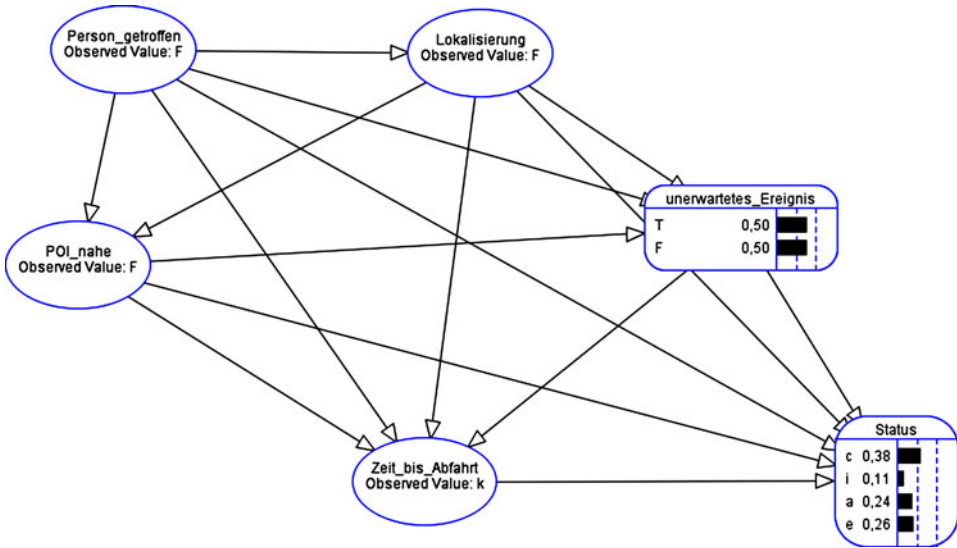


Abb. 2.8 Die Ermittlung des Interessen-Status des Nutzers ist zuverlässiger durchführbar, wenn das System durch Interaktion weitere Informationen über die beobachteten Wahrnehmungen hinaus einholt

Einem gewünschten Abbruch wird eine Wahrscheinlichkeit von 24 % zugeordnet, einem externen Ereignis 26 %. Die neue Information reduziert also die Unsicherheit über den Nutzerstatus im Vergleich zur Situation in Abb. 2.7 erheblich.

Der Unterschied zwischen dem ROSE-Navigationssystem und einem Anti-Blockiersystem (ABS) sollte augenfällig sein: Damit das ABS Assistenz leisten kann, muss es nur messen, ob sich die gebremsten Räder noch drehen. Dazu ist eine Kommunikation mit dem Nutzer nicht nur unnötig, sondern sogar unmöglich: wie soll der Nutzer während der Fahrt, in einer Situation, die seine ganze Aufmerksamkeit erfordert, feststellen, ob die Räder seines Fahrzeugs blockieren? Andererseits gibt es eben, wie an den Beispielen oben verdeutlicht, Situationen, in denen für das Leisten von Assistenz Information über Tatsachen entscheidend ist, die nur dem Nutzer bekannt sind.

Die Fähigkeit, Interaktion mit dem Nutzer betreiben zu können, ist also eine unverzichtbare Eigenschaft für Assistenzsysteme, deren Assistenzfunktionen auf – bei autonomer Arbeitsweise – unsichere Information über den Nutzer und die vom Nutzer wahrnehmbare Umgebung angewiesen sind.

2.3.2 Fähigkeit zur Diagnose

Interaktion ist in vielen Fällen auch für eine weitere zentrale Eigenschaft von Assistenzsystemen unverzichtbar: Wenn ein Assistenzsystem Effektkontrolle durchführt und dabei

feststellt, dass die eingetretenen Effekte nicht den erwarteten entsprechen, entsteht Unsicherheit über die aktuelle Information, die oft nur durch Interaktion aufgelöst werden kann. Ein Beispiel aus dem ROSE-Navigationssystem mag dies verdeutlichen:

Nehmen wir an, das Navigationssystem führe den Nutzer gerade zu einem von ihm ausgewählten Ziel und gebe dazu die geeigneten Instruktionen. Nun sei aber – was das Navigationssystem nicht feststellen kann – eine Straße versperrt. Dadurch ist der Nutzer zu einem Umweg gezwungen. Die Effektkontrolle wird also in Kürze feststellen, dass die vorgeschlagene Route nicht eingehalten wird.

Bei einem handelsüblichen Navigationssystem ist für diesen Fall keine Möglichkeit vorgesehen, mit dem Nutzer über die Ursache der Abweichung in Interaktion zu treten. Das Navigationssystem kann also zwischen falscher Positionierung und der Möglichkeit eines erzwungenen Umwegs nicht unterscheiden. Insbesondere zieht es überhaupt nicht in Betracht, dass sich der Interessenstatus des Nutzers geändert haben kann. Wegen fehlender Interaktionsmöglichkeiten ist also weder die Unsicherheit einfach zu beseitigen, ob der Nutzer seine Interessen geändert hat, oder ob tatsächlich ein externes Ereignis vorliegt, das vom System nicht wahrgenommen werden kann. So berechnet ein handelsübliches Navigationssystem so lange eine neue Route zum Ziel, bis es im äußersten Fall vom entnervten Nutzer abgeschaltet wird.

Ohne Informationen vom Nutzer über die Sperrung versucht es, den Umweg für den Nutzer zu minimieren, und leitet ihn damit oft wieder auf den gesperrten Wegabschnitt. Ein diagnosefähiges Assistenzsystem würde im ersten Fall in anderer Art und Weise als bisher assistieren und im zweiten Fall seine Wegplanung anhand der vom Nutzer erhaltenen Informationen – eventuell weiß er ja, wie weit die Sperre reicht – aktualisieren.

Die Fähigkeiten zur Diagnose und zur Interaktion ergänzen sich also bei der Reduktion von Unsicherheit über die aktuelle Situation.

Effektkontrolle und damit auch Diagnose sind eine große Herausforderung bei der Entwicklung von Assistenzsystemen, da sie mit einem hohen Aufwand verbunden sind, Wissen über die Anwendungsdomäne explizit zu modellieren. Diagnose beruht nämlich auf zumindest folgenden Voraussetzungen:

- Das Assistenzsystem kennt eine Handlungsfolge, mit der der Nutzer die aktuelle Aufgabe lösen kann.
- Das Assistenzsystem kennt die (in Bezug auf die Handlungsfolge) *beabsichtigten* Effekte jeder Handlung und ist somit in der Lage, Effektkontrolle überhaupt erst durchzuführen.
- Das Assistenzsystem vermag die *tatsächlich eingetretenen* Effekte ausgeführter Handlungen festzustellen.
- Das Assistenzsystem ist in der Lage, *beabsichtigte* und *tatsächlich eingetretene* Effekte zu vergleichen und Abweichungen festzustellen.

Alle hier beschriebenen Voraussetzungen für die Implementierung einer Effektkontrolle in einem Assistenzsystem lassen sich an folgendem aus [21] entnommenen und um die

Klauseln für Diagnose erweiterten Beispiel veranschaulichen. Es handelt sich dabei um eine GOLOG-Implementierung einer Simulation für eine vereinfachte Aufzugssteuerung.

```
% A Reactive Elevator Controller with Interactively
%Generated Exogenous Actions

% Primitive action declarations.

primitive_action(goUp). primitive_action(goDown).
primitive_action(resetButton(N)). primitive_action(toggleFan).
primitive_action(ringAlarm). primitive_action(wait).
primitive_action(startFire). primitive_action(endFire).
primitive_action(callElevator(N)). primitive_action(resetAlarm).
primitive_action(changeTemp).

% GOLOG Procedures

proc(control, wait : while(some(n,buttonOn(n)),
    pi(n,?(buttonOn(n)):serveFloor(n)))).

proc(serveFloor(N), while(-atFloor(N), if(aboveFloor(N), goDown, goUp)) :
    resetButton(N)).

proc(rules,?(fire & -alarmOn):ringAlarm:while(alarmOn,wait) #
    ?(tooHot & -fan):toggleFan #
    ?(tooCold & fan):toggleFan #
    ?(-(fire & -alarmOn v tooHot & -fan v tooCold & fan))).

% exoTransition under the assumption that at most one exogenous action
% can occur after each control action.

exoTransition(S1,S2):- requestExogenousAction(E,S1),
    (E = nil, S2 = S1; \+ E = nil, S2 = do(E,S1)).

requestExogenousAction(E,S):-
    write('Enter an exogenous action, or nil.'), read(E1),
    % IF exogenous action is nil, or is possible THEN no problem
    ((E1 = nil ; poss(E1,S)) -> E = E1 ;
    % ELSE print error message, and try again.
    diagnosis(E1,S),
    write('>> Action not possible. Try again.'), nl,
    requestExogenousAction(E,S)).

% Preconditions for Primitive Actions

poss(goUp,S):- atFloor(N,S), topFloor(T), N < T.
poss(goDown,S):- atFloor(N,S), firstFloor(F), F < N.
poss(resetButton(N),S):- atFloor(N,S), buttonOn(N,S).
```

```

poss(toggleFan,S).
poss(ringAlarm,S) :- fire(S).
poss(startFire,S) :- \+ fire(S).
poss(endFire,S) :- fire(S).
poss(callElevator(N),S) :- \+ buttonOn(N,S).
poss(changeTemp,S).
poss(resetAlarm,S) :- alarmOn(S).
poss(wait,S).

% Successor State Axioms for Primitive Fluents.

buttonOn(N,do(A,S)) :- A = callElevator(N) ;
    buttonOn(N,S), \+ A = resetButton(N).
fire(do(A,S)) :- A = startFire ; \+ A = endFire, fire(S).
fan(do(A,S)) :- A = toggleFan, \+ fan(S) ; \+ A = toggleFan, fan(S).
atFloor(N,do(A,S)) :- A = goDown, atFloor(M,S), N is M - 1 ;
    A = goUp, atFloor(M,S), N is M + 1 ;
    \+ A = goDown, \+ A = goUp, atFloor(N,S).
alarmOn(do(A,S)) :- A = ringAlarm ; \+ A = resetAlarm, alarmOn(S).
temp(T,do(A,S)) :- A = changeTemp, temp(T1,S), (\+ fan(S), T is T1 + 1 ;
    fan(S), T is T1 - 1) ;
    \+ A = changeTemp, temp(T,S).
tooHot(S) :- temp(T,S), T > 3.
tooCold(S) :- temp(T,S), T < -3.
aboveFloor(N,S) :- atFloor(M,S), N < M.

% Initial Situation.

atFloor(1,s0). temp(0,s0). topFloor(6). firstFloor(1).

% Restore suppressed situation arguments.

restoreSitArg(tooCold,S,toCold(S)). restoreSitArg(fire,S,fire(S)).
restoreSitArg(buttonOn(N),S,buttonOn(N,S)). restoreSitArg(fan,S,fan(S)).
restoreSitArg(tooHot,S,tooHot(S)).
restoreSitArg(atFloor(N),S,atFloor(N,S)).
restoreSitArg(alarmOn,S,alarmOn(S)). restoreSitArg(temp(T),S,temp(T,S)).
restoreSitArg(aboveFloor(N),S,aboveFloor(N,S)).
restoreSitArg(requestExogenousAction(E),S,requestExogenousAction(E,S)).

% Main Clause

elevator :- doR(control,rules,s0,S).

% Diagnosis rules

diagnosis(ringAlarm,S) :- \+ fire(S), write('keep calm. there's no fire'),
    nl.

```

```

diagnosis(resetAlarm,S) :- \+ alarmOn(S), write('no alarm at all'), nl.
diagnosis(startFire,S) :- fire(S), write('elevator is already burning'),
    nl.
diagnosis(endFire,S) :- \+ fire(S), write('ok. no more fire'),
    nl.
diagnosis(callElevator(N),S) :-
    buttonOn(N,S), write('be patient. elevator will arrive soon'), nl.
diagnosis(resetButton(N),S) :-
    \+ atFloor(N,S), write('elevator not yet arrived at this floor'), nl;
    \+ buttonOn(N,S), write('nobody waiting at this floor'), nl.
diagnosis(A,S) :- write('no diagnosis for '),
    write(A), write(' '), nl.

```

Im Abschnitt `MainClause` wird die Zielklausel `elevator` definiert, deren Beweis durch einen Prolog-Interpreter die GOLOG-Prozedur `control` ausführt. Sie erfüllt die erste Voraussetzung, da sie die Handlungsfolge festlegt, mit der die Aufgabe der Aufzugsteuerung erledigt werden kann. Im Abschnitt `Successor State Axioms for Primitive Fluents` sind die Effekte jeder definierten primitiven Aktion festgelegt. Die Aufzugsteuerung kennt also alle *beabsichtigten* Effekte einer durchgeführten Aktion. Wenn das GOLOG-Programm ausgeführt wird, konstruiert der Prolog-Interpreter nach jedem Schritt einen Situationsterm, der beschreibt, welche Aktionen bisher ausgehend von der initialen Situation ausgeführt worden sind. Mit Hilfe der primitiven Fluents und der Vorbedingungen für die primitiven Aktionen stellt der GOLOG-Interpreter fest, ob die *beabsichtigten* Effekte auch *tatsächlich eingetreten* sind. Damit ist die dritte Voraussetzung gegeben. Schließlich ist der GOLOG-Interpreter in der Lage festzustellen, ob die *tatsächlich eingetretenen* Effekte auch die *beabsichtigten* sind: dies ist dann nicht der Fall, wenn die Vorbedingungen für die nächste auszuführende Aktion nicht erfüllt sind. Mit Hilfe der Klauseln im Abschnitt `Diagnosis Rules` stellt der GOLOG-Interpreter genau fest, welche *beabsichtigten* Effekte nicht eingetreten sind. Auf diese Weise führt er eine Diagnose durch, die eine explizite Aussage darüber erlaubt, warum zum aktuellen Zeitpunkt die Aufzugsteuerung nicht funktioniert.

Dieses Beispiel illustriert, wie ein Assistenzprogramm mit Diagnosefähigkeit versehen werden kann, wenn genügend Wissen über die Aufgaben des Nutzers bekannt und geeignet formalisiert ist. An dem Beispiel wird aber noch nicht deutlich, wie ein Assistenzsystem weiter Unterstützung leisten kann, selbst wenn und gerade dann, wenn beabsichtigte Effekte nicht tatsächlich eingetreten sind. Offensichtlich ist die Fähigkeit zur Diagnose nur dann sinnvoll einsetzbar, wenn das Assistenzsystem seine Unterstützung an die aktuelle Situation anpassen kann.

Im weiter oben diskutierten Szenario des ROSE-Fußgängernavigationssystems ist die Diagnose aus einem Abgleich zwischen erwarteten und tatsächlich gemessenen Positionen des Nutzers berechenbar. Klarerweise wird von einem Assistenzsystem erwartet, nicht nur Diskrepanzen festzustellen, sondern auch darauf reagieren zu können. Handelsübliche Navigationssysteme berechnen einfach eine neue Route zum bisherigen Ziel; dies ist

sicherlich eine Möglichkeit, auf eine kritische Diagnose zu reagieren; wie weiter oben bereits diskutiert, ist sie aber wenig situationsadaptiv, weil nicht versucht wird, die Unsicherheit über die aktuelle Situation zu beseitigen. Eine viel unflexiblere Art und Weise, auf Abweichungen des Nutzers zur geplanten Route zu reagieren, wäre die starre Meldung „*Bitte wenden Sie!*“, wie sie von vielen KFZ-Navigationssystemen bekannt ist. Mit Hilfe einer derartigen Meldung alleine ist es aber nicht möglich, für die Lösung der Aufgabe hilfreiche Assistenz zu leisten, die den Problemlöseprozess des Nutzers zielorientiert zu unterstützen vermag. Die Diskussion verdeutlicht, dass die Frage der adäquaten, d. h. in Bezug auf die Aufgabe optimale Assistenz bietenden, Reaktion auf eine kritische Diagnose sehr eng mit der Frage, wie die Zielorientierung eines Assistenzsystems hergestellt werden kann, verknüpft ist. Dieser Hypothese soll im folgenden Abschnitt anhand eines Vergleichs zwischen den bisher diskutierten interaktiven Systemen mit einem autonomen System zur Kontrolle eines Roboters weiter nachgegangen werden.

2.3.3 Fähigkeit zur Korrektur

Das Beispiel des ROSE-Navigationssystems gab bereits Hinweise darauf, dass die Fähigkeit eines Assistenzsystems, auch bei nicht beabsichtigten Situationen Unterstützung zu leisten, davon abhängt, wie gut es Diagnosen über die Diskrepanz zwischen beabsichtigten und tatsächlich eingetretenen Effekten in Bezug auf die zu lösende Aufgabe interpretieren kann.

Deutlicher als bei interaktiven Assistenzsystemen wird dies bei autonomen Systemen. In den beiden Skizzen in Abb. 2.9 symbolisiert der große Kreis rechts einen autonomen Roboter, der als Ziel das Feld links oben erreichen will. Der Roboter kann den durch die Umrisse gegebenen Bereich nicht verlassen. Die kleinen Kreise stehen für bewegliche Hindernisse, die sich auf und ab bewegen, und zwar in die von den Pfeilen andeutete Richtung. Über Sensoren kann der Roboter sowohl die statischen als auch die beweglichen Hindernisse wahrnehmen. Genauer gesagt: er nimmt wahr, dass im gemessenen Abstand der Weg nicht frei ist; eine Unterscheidung zwischen verschiedenen Typen von Hindernissen ist nicht möglich. Auf die Diagnose, dass Hindernisse in der Nähe sind, muss der Roboter reagieren, um Kollisionen zu vermeiden. Üblicherweise sind in Robotern Hindernisvermeidungsalgorithmen implementiert, die ein physikalisches Modell realisieren (siehe [22], Kap. 4): der Roboter bewegt sich mit einer bestimmten Geschwindigkeit auf das Ziel zu – daraus lässt sich ein zweidimensionaler Vektor für die Kraft, die den Roboter antreibt, ermitteln. Hindernisse werden auf folgende Weise in das Modell integriert: sie üben eine abstoßende Kraft auf den Roboter aus, die wie der Differenzvektor zwischen Hindernis und Roboter gerichtet sind, und deren Größe vom Abstand zwischen Hindernis und Roboter abhängt. Der Roboter bestimmt seine Bewegungsrichtung dann anhand der Ersatzkraft, die aus der eigenen Vorwärtsbewegung und den abstoßenden Kräften der vorhandenen Hindernisse resultiert.

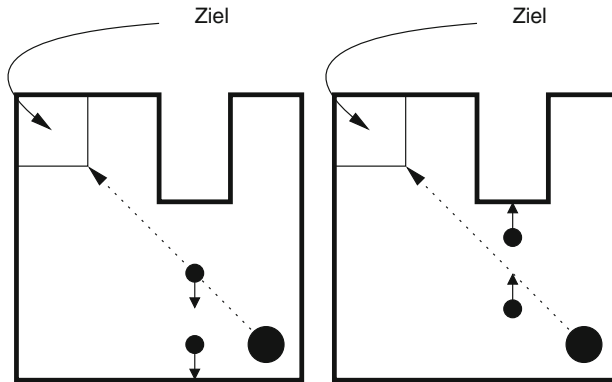


Abb. 2.9 Ein Roboter (symbolisiert durch den *großen Kreis*) bewegt sich auf das Ziel zu. Bewegliche Hindernisse (*kleine Kreise*) versuchen, ihn davon abzuhalten

Im Szenario, das in Abb. 2.9 zu sehen ist, wirken zunächst die abstoßenden Kräfte der statischen Hindernisse auf den Roboter. Da sie umso größer werden, umso näher der Roboter auf ein Hindernis zufährt, wird der Roboter von den statischen Hindernissen ferngehalten, wenn er nicht zu schnell beschleunigt. Zusätzlich wirken auch die abstoßenden Kräfte der beweglichen Hindernisse auf den Roboter. Da er, um das Ziel zu erreichen, die beweglichen Hindernisse passieren muss, wird der Roboter von den beweglichen Hindernissen abgestoßen, je mehr er sich ihnen nähert. Da sich die Hindernisse auf und ab bewegen, wirkt diese abstoßende Kraft permanent gegen die Fahrtrichtung des Roboters in Richtung auf das Ziel. Die Ersatzkraft zeigt tendenziell nach oben und wird den Roboter hinter den Mauervorsprung drängen. Dort ist die Luftlinie zum Ziel aber immer direkt durch die Mauer gerichtet, an der sich der Roboter im schlimmsten Fall für immer „festbeißen“ wird. Die statische Mauer löst eine Kraft auf den Roboter aus, die der Vorwärtskraft des Roboters gerade entgegen gerichtet ist. Der Roboter wird sich also der Mauer erst nähern, dann wieder von ihr abgestoßen. Daher wird er schließlich vor der Mauer in eine Oszillationsbewegung übergehen oder, falls sich die Kräfte das Gleichgewicht halten, einfach stehen bleiben.

In der beschriebenen Situation steht dem Roboter nur eine einzige Möglichkeit zur Diagnose zur Verfügung. Er kann die Entfernung zum Ziel berechnen und daraus seine Vorwärtsbewegung neu anpassen, also auf die Diagnose reagieren. Endet diese Reparaturstrategie, wie oben beschrieben, im Stillstand oder in einer anderen Bewegung, die den Roboter auf Dauer nicht dem Ziel näher bringt, fehlt dem Roboter jedoch jede Möglichkeit, zu analysieren, warum seine Fahrbewegungen nicht zum Ziel führen.

Die Reparaturstrategie ist also nicht hilfreich für die Lösung der Aufgabe, weil die Wahrnehmungen aus der Umgebung des Roboters nicht in Hinblick auf die für die Lösung auszuführenden Schritte interpretiert werden. Der Roboter kann die gestellte Diagnose nicht ausnutzen, um seine Reparaturstrategie geeignet zu modifizieren. Dem Roboter fehlt

nämlich die Kenntnis über eine Handlungsfolge, die zum Ziel führt. Die Bewegung in direkter Richtung auf das Ziel ist nämlich nur eine Schätzung für einen ersten Schritt zur Lösung der Aufgabe. Der Roboter kann daher keine späteren Situationen vorhersehen, in denen andere Schritte als die Bewegung direkt auf das Ziel die gestellte Aufgabe lösen.

Die Fähigkeit zur Reparatur ist also mit der Fähigkeit verbunden, Zielorientierung so umzusetzen, dass mehrere Schritte zum Ziel eingeplant werden. Das bedeutet aber unmittelbar, dass ein Assistenzsystem mit Zielorientierung Hypothesen darüber generieren können muss, welche Situationen sich aus der Ausführung einer Reihe von Aktionen ergeben können. Das Assistenzsystem simuliert also zukünftige Zustände durch *Probehandeln*, indem es überprüft, ob die eventuelle Ausführung einer Aktion die *beabsichtigten* Effekte erreichen kann. Das Assistenzsystem wird Hypothesen, in denen die *eventuell eintretenden* Effekte nicht mit den *beabsichtigten* übereinstimmen, nicht weiterverfolgen. Diese Auswahl von Hypothesen, die eine hohe Chance haben, zur Lösung der gestellten Aufgabe beizutragen, ist bereits eine erste Form von Diagnose und Reparatur.

Der Roboter kann die eben beschriebenen Eigenschaften umsetzen, indem er die Grundannahme aufgibt, mit einer einzigen Fahrbewegung das Ziel erreichen zu können. Stattdessen kann er eine plausible – beispielsweise an seiner mittleren Geschwindigkeit orientierten – Zahl von einzelnen Schritten voraussehen. Die Schritte selbst lassen sich durch einen Routensuch-Algorithmus wie etwa den A*-Algorithmus⁹ oder Varianten davon¹⁰ finden. Sofern der Roboter in der Lage ist, seine aktuelle Position – zumindest ungefähr – zu bestimmen, kann er eine Effektkontrolle nach jedem Schritt durchführen. Stimmt die tatsächlich erreichte Position nicht mit der beabsichtigten überein, kann der Roboter eine neue Folge von Schritten suchen, durch deren Ausführung er das Ziel erreichen kann. In der oben besprochenen Szene in Abb. 2.9 wird der Roboter also nicht einfach versuchen auf der Luftlinie das Ziel anzufahren, sondern er wird einen Pfad durch die Umgebung suchen, der Hindernisse vermeidet und dennoch zum Ziel führt.

Offensichtlich ist diese Strategie umso erfolgreicher, umso präziser zukünftige Situationen vorhergesehen werden können. Könnte der Roboter also die Bewegung der Hindernisse aufzeichnen und daraus ihre mittleren Geschwindigkeiten schätzen, hätte er weitere Möglichkeiten zur Reparatur:

- Anhand der Schätzung lassen sich so viele Warteschritte einplanen, bis in einer Situation sich die Hindernisse auf einer Position befinden, die dem Roboter die Passage durch die Hindernisse hindurch ermöglicht.
- Eine Ersatzroute kann berechnet werden, indem berücksichtigt wird, wie sich die Hindernisse bewegen. Auch in diesem Fall werden zur Reparatur Hypothesen über zukünftige Situationen benötigt.

⁹ Der A*-Algorithmus wird in [23] vorgestellt.

¹⁰ Eine bei der Routenplanung oft verwendete Variante ist D*; bei diesem Verfahren (siehe [22], Abschn. H.3) wird der Neuplanungsaufwand minimiert, wenn der ursprünglich vorgesehene Pfad nicht eingehalten werden konnte.

In beiden Fällen folgt jedoch aus der Simulation zukünftiger Zustände *nicht*, dass die Diagnose ausgeschaltet werden kann, wenn die für die Lösung der Aufgabe gefundenen Schritte ausgeführt werden. Es gibt nämlich keine Garantie, dass die Schätzung für das Bewegungsverhalten richtig war, und nicht während der Abarbeitung der Schritte unerwartete Ereignisse auftreten, die verhindern, dass die *beabsichtigten* Effekte erreicht werden. Die Effektkontrolle nach jedem einzelnen Schritt ist also eine wichtige Eigenschaft eines Assistenzsystems.

Das Beispiel des autonomen Roboters verdeutlicht aber auch – wie beim ROSE-Navigationssystem schon beschrieben –, dass Diagnose nur dann zielorientiert sein kann, wenn sie beachtet, dass Abweichungen von *erwarteten* Effekten in *allen* Handlungsphasen im Sinn von [1] auftreten können. Zielorientierte Reparatur ist nur möglich, wenn Diskrepanzen der richtigen Handlungsphase zugeordnet werden. Ein zielorientiertes Assistenzsystem integriert dazu die Simulation von Aktionen das Schätzen von erwarteten Werten für unsichere Information, das Ausführen von Aktionen und die Kontrolle von deren Effekten. Durchführung von Assistenz ist damit eine hierarchisch angelegte Aufgabe, die Effektkontrolle permanent in allen Handlungsphasen auszuführen.

2.3.4 Fähigkeit zur Erklärung

Die Bezugnahme auf die richtige Handlungsphase ist auch von großer Bedeutung, wenn das Assistenzsystem Ergebnisse von Diagnosen, Planungen und Aktionen dem Benutzer kommuniziert, wenn es also sein Vorgehen *erklärt*. Was bedeutet *Erklären* als Aktivität eines zielorientierten Assistenzsystems genau? Verschiedene Antworten sind auf diese Frage vorstellbar:

- Das Assistenzsystem sucht zur Lösung einer Aufgabe nach einer geeigneten Folge von Aktionen, indem es zukünftige Zustände simuliert. Heißt *Erklären*, Gründe für die gewählte Schrittfolge angeben zu können?
- Wenn Aktionen ausgeführt werden, vergleicht das Assistenzsystem erwartete mit beobachteten Effekten. Heißt *Erklären* also, die nicht erfüllte Erwartung benennen zu können?
- Heißt *Erklären* darüber hinaus sogar, weitere Konsequenzen eines eingetretenen, aber nicht erwarteten Effekts finden und benennen zu können?
Insbesondere geht es dabei um Konsequenzen dafür, ob das bisher angestrebte Ziel noch erreicht werden kann oder nicht.

Eine Antwort darauf, welche dieser Fragen für Assistenzsysteme relevant sind, kann aus dem Zweck von Assistenz abgeleitet werden, der von [1] auf S. 144 folgendermaßen bestimmt wird:

Users or operators have to understand the outcomes of their action and evaluate whether or not the outcome is congruent with their expectations. In man-machine-systems, the outco-

me of an action often cannot be perceived directly. As a result, users need assistance in (1) realizing the effects of their actions and (2) in interpreting these effects as a success or failure.

In einer Umgebung, die ein Softwaresystem verändern kann, ist der Zweck von Assistenz um noch einen Aspekt weiter zu fassen: Nutzer benötigen nicht nur Assistenz bei der Wahrnehmung von Effekten eigener Handlungen, sondern auch von Handlungen des Softwaresystems. Wenn eine Handlung scheitert, ist aber die Motivation des Nutzers, eine für ihn relevante Aufgabe zu lösen, nicht zwangsläufig auch erloschen, sondern besteht in der Regel weiter. Daher gibt es zwei weitere denkbare Leistungen einer *Erklärung*:

- Soll ein Assistenzsystem erklären können, wie eine Aufgabe auf eine andere als bisher geplante Art und Weise gelöst werden kann?
- Bedeutet *Erklären*, wenn aus den Konsequenzen nicht erwarteter Effekte eine bisher vorgesehene Lösung einer Aufgabe unmöglich geworden ist, neue Ziele bzw. neue Motivation oder Aktivierung zu gewinnen?

Sogar neue Ziele konstruieren zu können, wenn die Lösung der aktuellen Aufgabe schwierig oder unmöglich geworden ist, führt über das *Erklären* im engeren Sinn, also das Erklären von erledigten oder noch durchzuführenden Lösungsschritten, hinaus und geht in die Assistenzform der Beratung über. Wie in späteren Kapiteln zu besprechen sein wird, sind auch die technischen Anforderungen an Beratung bei der Zielbildung weitaus höher als beim Erklären – Abschn. 2.3.5 wird einen vertieften Einblick in diesen Aspekt geben.

Anhand eines Beispiels aus der ROSE-Domäne zur Motivbildung (Aktivierungsassistenz) und zur Effektkontrolle (Rückmeldungsassistenz) soll in diesem Abschnitt zunächst argumentiert werden, dass ein Assistenzsystem für Erklärungen möglichst viel explizites Wissen über die aktuelle Situation benötigt, um die Ergebnisse seiner Effektkontrolle in Hinblick auf die zu lösende Aufgabe als Erklärungen zu formulieren. Mit ihrer Hilfe kann das Assistenzsystem dem Nutzer dann leichter die nötige Unterstützung bieten, die gegebene Information wie eine zusätzliche Wahrnehmung in der aktuellen Situation zu verstehen und damit seine Kenntnis über den Sachstand bei der Lösung seiner Aufgabe zu vertiefen.

Diese Anforderungen stellen hohe Ansprüche an die Implementierung eines Assistenzsystems, da normalerweise die systeminterne Repräsentation des vorhandenen Wissens für den Nutzer nicht verständlich ist. Oft wird auch auf ausführliche Repräsentation von aktueller Information zugunsten von schnellen algorithmischen Lösungen bei der Realisierung von *user interfaces* verzichtet. Beispielhaft dafür sind die BAYES-Netze in den Abb. 2.4, 2.5, 2.6 und 2.7. Die Knoten in diesen Netzen sind starke und aus Sicht der vom Nutzer durchgeführten Handlung willkürliche Vergröberungen der Sachverhalte, die in der aktuellen Situation gelten können: Jeder Knoten steht für eine ganze Äquivalenzklasse von Zuständen. Die Knoten sind eben so konzipiert, dass sie die für die Analyse des Interessenstatus des Nutzers notwendige Information für den gewählten Inferenzalgorithmus geeignet repräsentieren, darüber hinaus aber keine Information über die aktuelle Situation speichern.

Genau diese wäre aber für ausführliches, situationsadaptives *feedback* entscheidend, um eine der Aufgabe angemessene Rückmeldeassistentz zu gewährleisten. Daraus die Konsequenz zu ziehen, in den benutzten Modellen mehr Information zu speichern, um den aktuellen Sachverhalt präziser explizit repräsentieren zu können, stößt aber rasch an technische Grenzen: der Suchaufwand zur Berechnung von Entscheidungen und Ergebnissen steigt enorm an; eine Modellierung für ein Anwendungsproblem wird rasch unhandlich, insbesondere dann, wenn Echtzeitverhalten gefordert wird.

Andererseits sind aber gerade diese Algorithmen, die mit „minimalistischen“ Zustandsüberführungsmodellen im Suchraum die für eine Anwendung interessanten Bereiche identifizieren, bei der Kontrolle autonomer Systeme sehr erfolgreich: Fast alle modernen Robotersteuerungen modellieren das Kontrollproblem, den Roboter zu navigieren und zielführende Aktionen durchführen zu lassen, als sogenannten *Partially Observable Markov Decision Process* (POMDP)¹¹. Das mathematische Modell des *Partially Observable Markov Decision Process* wird auch oft benutzt (siehe etwa [26]), um Dialogsteuerungen für meist natürlich-sprachliche Mensch-Maschine-Dialoge zu implementieren. Dabei wird gerade die Vergrößerung des Raums der durch Sprechhandlungen erreichbaren Zustände verwendet, um ein einfaches Zustandsüberführungsmodell der Aufgabe, über die Dialog geführt wird, zu konstruieren. Allerdings liegt, wie oben besprochen, gerade in dieser Vergrößerung die große Gefahr, entscheidende Informationen zu verlieren, die unverzichtbar sind, wenn der Mensch-Maschine-Dialog auch die Assistenz thematisieren soll, die das System dem Nutzer leistet.

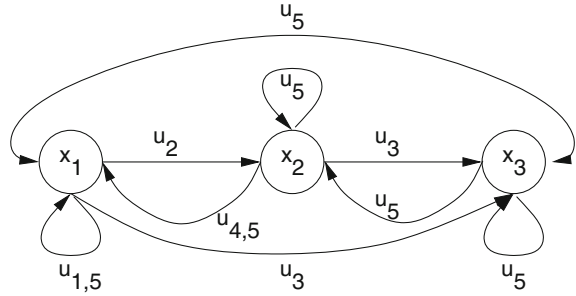
Das Thema dieses Abschnitts ist ja, dass Erklärungen, die dem Nutzer situationsgerechte Interpretation von Effekten und durch das Assistenzsystem kommunizierte Wahrnehmungen erlauben, viel explizites Wissen über die aktuelle Situation beim Assistenzsystem voraussetzen. Die flache Modellierung von Domäne und Situation, wie sie bei einem *Partially Observable Markov Decision Process* durchgeführt wird, ist jedoch für derartige Erklärungen hinderlich: Nicht nur die Vergrößerung des Suchraums jedoch führt zu Entscheidungen, die vom Nutzer nicht erwartete und daher schwer verständliche Effekte produzieren. Vielmehr trägt auch die Art und Weise, wie Entscheidungen für zukünftige Handlungen getroffen werden, dazu bei. Als exemplarischer Beleg für diese Aussage soll Entscheidungsfindung in einer sehr einfachen, aber doch beispielhaften autonomen Kontrolle untersucht werden: Abbildung 2.10 zeigt das Zustandsüberführungsmodell für die Kontrolle der Assistenzfunktion im ROSE-Navigationssystem.

Für den Zustand der Assistenz, der vom Kontrollalgorithmus gesteuert wird, gibt es drei verschiedene Möglichkeiten:

- x_1 : Assistenz läuft.
- x_2 : Assistenz ist unterbrochen.
- x_3 : Assistenz ist beendet.

¹¹ Eine detaillierte Darstellung darüber ist in [24, 25], Kap. 14 und 15 gegeben; später in Kap. 6 wird das POMDP-Modell als Paradigma für interaktives Planen erörtert werden.

Abb. 2.10 Graph der Zustandsübergangsfunktion für die Kontrolle des Assistenzverhaltens im ROSE-Navigationssystem



Der Übergang zwischen den Zuständen wird durch fünf verschiedene Aktionen, die die Kontrolle durchführen kann, ausgelöst:

- u_1 : Assistenz weiterführen
- u_2 : Assistenz unterbrechen
- u_3 : Assistenz beenden
- u_4 : Assistenz wieder aufnehmen
- u_5 : Interessenstatus des Nutzers abfragen

Die Zustandsüberföhrungsfunktion ist durch den Graphen in Abb. 2.10 gegeben. Andere Zustände als die im Graphen eingezeichneten sind nicht definiert und treten daher mit Sicherheit nicht ein. Im allgemeinsten Fall eines stochastischen Entscheidungsprozesses ist der Übergang von einem Zustand zu einem Folgezustand nichtdeterministisch. Aus sehr vielen früheren ähnlichen Fällen lässt sich abschätzen, wie wahrscheinlich welcher Übergang ist¹².

Aus dieser Festlegung der Zustandsüberföhrungsfunktion lässt sich die folgende Tabelle für die Übergangswahrscheinlichkeiten zwischen zwei Zuständen bei einer gegebenen Aktion definieren:

- Für Übergänge, die in den Zustand x_1 führen:

$$\begin{array}{lll}
 p(x_1|x_1, u_1) = 1 & p(x_1|x_2, u_1) = 0 & p(x_1|x_3, u_1) = 0 \\
 p(x_1|x_1, u_2) = 0 & p(x_1|x_2, u_2) = 0 & p(x_1|x_3, u_2) = 0 \\
 p(x_1|x_1, u_3) = 0 & p(x_1|x_2, u_3) = 0 & p(x_1|x_3, u_3) = 0 \\
 p(x_1|x_1, u_4) = 0 & p(x_1|x_2, u_4) = 1 & p(x_1|x_3, u_4) = 0 \\
 p(x_1|x_1, u_5) = \frac{1}{3} & p(x_1|x_2, u_5) = \frac{1}{3} & p(x_1|x_3, u_5) = \frac{1}{3}
 \end{array}$$

¹² Dieses Lernverfahren wird in [27] eingeföhrt.

- Für Übergänge, die in den Zustand x_2 führen:

$$\begin{array}{lll}
 p(x_2|x_1, u_1) = 0 & p(x_2|x_2, u_1) = 0 & p(x_2|x_3, u_1) = 0 \\
 p(x_2|x_1, u_2) = 1 & p(x_2|x_2, u_2) = 0 & p(x_2|x_3, u_2) = 0 \\
 p(x_2|x_1, u_3) = 0 & p(x_2|x_2, u_3) = 0 & p(x_2|x_3, u_3) = 0 \\
 p(x_2|x_1, u_4) = 0 & p(x_2|x_2, u_4) = 0 & p(x_2|x_3, u_4) = 0 \\
 p(x_2|x_1, u_5) = \frac{1}{3} & p(x_2|x_2, u_5) = \frac{1}{3} & p(x_2|x_3, u_5) = \frac{1}{3}
 \end{array}$$

- Für Übergänge, die in den Zustand x_3 führen:

$$\begin{array}{lll}
 p(x_3|x_1, u_1) = 0 & p(x_3|x_2, u_1) = 0 & p(x_3|x_3, u_1) = 0 \\
 p(x_3|x_1, u_2) = 0 & p(x_3|x_2, u_2) = 0 & p(x_3|x_3, u_2) = 0 \\
 p(x_3|x_1, u_3) = 1 & p(x_3|x_2, u_3) = 0 & p(x_3|x_3, u_3) = 0 \\
 p(x_3|x_1, u_4) = 0 & p(x_3|x_2, u_4) = 0 & p(x_3|x_3, u_4) = 0 \\
 p(x_3|x_1, u_5) = \frac{1}{3} & p(x_3|x_2, u_5) = \frac{1}{3} & p(x_3|x_3, u_5) = \frac{1}{3}
 \end{array}$$

Der Übergang zwischen verschiedenen Zuständen erfolgt deterministisch. Existiert in Graphen eine Kante, so wird der Übergang zwischen den beiden von der Kante verbunden Zuständen mit Sicherheit ausgeführt.

Neben der Zustandsübergangsfunktion verfügt ein *Partially Observable Markov Decision Process* noch über eine weitere Wissensquelle über die modellierte Domäne: In jedem Zustand gibt es zu jeder ausführbaren Aktion eine heuristisch festgelegte Bewertung – den sogenannten *reward* oder *immediate payoff*. Diese Größe sagt aus, wie gut durch Ausführen der betrachteten Aktion im betrachteten Zustand das Kontrollziel erreicht werden kann.

Welche Auswirkungen hat dies Methodik, Wissen über eine Domäne zu repräsentieren auf die Fähigkeit eines Assistenzsystems, präzise Erklärungen abzugeben? Um diese Frage zu diskutieren, sei folgende Definition des *immediate payoff* angenommen:

$$\begin{array}{lll}
 r(x_1, u_1) = 20 & r(x_2, u_1) = -1000 & r(x_3, u_1) = -1000 \\
 r(x_1, u_2) = -20 & r(x_2, u_2) = -1000 & r(x_3, u_2) = -1000 \\
 r(x_1, u_3) = 20 & r(x_2, u_3) = -1000 & r(x_3, u_3) = -1000 \\
 r(x_1, u_4) = -1000 & r(x_2, u_4) = 50 & r(x_3, u_4) = -1000 \\
 r(x_1, u_5) = -1 & r(x_2, u_5) = -1 & r(x_3, u_5) = -1
 \end{array}$$

Um eine Entscheidung zu treffen, welche Aktion in einem Zustand auszuführen ist, werden alle aktuell möglichen Optionen unabhängig voneinander bewertet. Schließlich wird diejenige Aktion ausgewählt, für die die beste Bewertung ermittelt werden konnte.

Die Realisierung der Kontrolle eines autonomen Systems auf die eben beschriebene Art und Weise ist dazu geeignet, Unsicherheiten in der Auswirkung von Aktionen zu berücksichtigen. Aus diesem Grund wurden oben bei der Spezifikation der Zustandsübergangsfunktionen Wahrscheinlichkeiten verwendet. So kann man auf unkomplizierte Weise

der Tatsache Rechnung tragen, dass im allgemeinen Fall die Ausführung einer Aktion in einem bestimmten Zustand nicht immer in denselben Zustand führt, also nichtdeterministisch ist¹³.

Aus diesem Grund ist die Entscheidungsfindung etwas komplizierter als oben suggeriert: Die Kontrolle ist sich nicht sicher, in welchem Zustand sich das System zum aktuellen Zeitpunkt befindet, sie hat nur Annahmen über die Wahrscheinlichkeit, mit der jeder der definierten Zustände eingenommen sein kann. Die Bewertung der besten Aktion muss dies berücksichtigen, indem sie nicht nur berechnet, welche Aktion die beste Bewertung in einem einzigen Zustand erhält, sondern sie muss für jeden Zustand die bestbewertete Aktion finden und diese Bewertung mit der Wahrscheinlichkeit, dass sich das System gerade im betrachteten Zustand befindet, gewichten. Es wird also der Erwartungswert des *immediate payoff* über alle Aktionen und Zustände errechnet:

$$V = \max_u \sum_{i=1}^3 p_i \cdot r(x_i, u)$$

Die Kontrolle entscheidet sich dann für dasjenige u , das den höchsten Wert für V liefert. Die Formel zeigt auch, dass die Annahme, zu jedem Zeitpunkt wäre der Zustand des Systems eindeutig festgelegt, einen Spezialfall darstellt, nämlich gerade den, in dem alle Wahrscheinlichkeiten p_i dafür, dass sich das System im Zustand i befindet, den Wert 0 haben, bis auf eine einzige, die den Wert 1 hat.

Anhand der oben definierten Übergangswahrscheinlichkeiten und *immediate payoffs* sollen die Konsequenzen dieser Vorgehensweise, Entscheidungen zu treffen, weiter untersucht werden: In unserem konkreten Fall wird V auf folgende Weise berechnet:

$$V = \max_u (p_1 \cdot r(x_1, u) + p_2 \cdot r(x_2, u) + p_3 \cdot r(x_3, u))$$

Da p_1 , p_2 und p_3 zusammen eine Wahrscheinlichkeitsverteilung bilden, gilt:

$$p_3 = 1 - p_1 - p_2$$

Daraus ergeben sich folgende Bewertungen für die einzelnen Aktionen:

$$\begin{aligned} u_1 : & 20 \cdot p_1 - 1000 \cdot p_2 - 1000 \cdot (1 - p_1 - p_2) = 1020 \cdot p_1 - 1000 \\ u_2 : & -20 \cdot p_1 - 1000 \cdot p_2 - 1000 \cdot (1 - p_1 - p_2) = 980 \cdot p_1 - 1000 \\ u_3 : & 20 \cdot p_1 - 1000 \cdot p_2 - 1000 \cdot (1 - p_1 - p_2) = 1020 \cdot p_1 - 1000 \\ u_4 : & -1000 \cdot p_1 + 50 \cdot p_2 - 1000 \cdot (1 - p_1 - p_2) = 1050 \cdot p_2 - 1000 \\ u_5 : & -p_1 - p_2 - (1 - p_1 - p_2) = -1 \end{aligned}$$

¹³ Dies geschieht beispielsweise, weil GPS-Empfänger aus technischen Gründen eine hohe Messgenauigkeit aufweisen. Je nach (nicht vorhersagbarem, da zufälligem) Messfehler, wird bei einer genau bestimmten Bewegung von einer festgelegten Position aus eine vom Messfehler verzerrte Annahme über die erreichte Position ermittelt. Zwar wird vom Messfehler erwartet, dass er nicht vorhanden ist, aber in der Praxis werden bei wiederholter Durchführung der eben beschriebenen Bewegung verschiedene Zielpositionen erreicht, jede mit einer bestimmten Wahrscheinlichkeit.

Beim Wettbewerb um die beste Bewertung konkurrieren gemäß der Formel für V also:

$$\begin{array}{ll} 1020 \cdot p_1 - 1000 & \text{für die Aktionen } u_1 \text{ und } u_3 \\ 1050 \cdot p_2 - 1000 & \text{für die Aktion } u_4 \\ -1 & \text{für die Aktion } u_5 \end{array}$$

Die entscheidende Beobachtung bei der Analyse dieses Beispiels ist, dass unabhängig davon, ob Unsicherheit über den aktuellen Zustand besteht oder nicht, der Folgezustand nichtdeterministisch ist: Mit $p_1 = 1$ (d.h.: $p_2 = p_3 = 0$) fällt die Entscheidung für u_1 oder u_3 zufällig (siehe oben): das einzige pragmatische Entscheidungskriterium, nämlich der *immediate payoff*, ist für beide Aktionen identisch.

Nicht einmal, wenn der *immediate payoff* an allen definierten Stellen unterschiedlich ist, kann eine nichtdeterministische Reaktion des Systems ausgeschlossen werden. Um dies zu veranschaulichen, setzen wir

$$r(x_1, u_3) = 10.$$

Damit wird die beste Aktion anhand folgender Bewertungskriterien ausgewählt:

$$\begin{array}{ll} 1020 \cdot p_1 - 1000 & \text{für die Aktion } u_1 \\ 1050 \cdot p_2 - 1000 & \text{für die Aktion } u_4 \\ -1 & \text{für die Aktion } u_5 \end{array}$$

Bei dieser Definition der *payoff*-Funktion r ist die Entscheidung deterministisch: einem gefundenen Maximum von V ist genau eine Aktion zugeordnet. In dieser Situation kann die Wahrscheinlichkeitsverteilung für p_1 , p_2 und p_3 dafür verantwortlich sein, dass keine eindeutige Entscheidung getroffen werden kann: u_1 wird genau dann ausgewählt, wenn gilt:

$$\begin{array}{ll} 1020 \cdot p_1 - 1000 > 1050 \cdot p_2 - 1000 & \text{und} & 1020 \cdot p_1 - 1000 > -1 \\ p_1 > \frac{105}{102} p_2 & & p_1 > \frac{999}{1020} \end{array}$$

Abbildung 2.11 veranschaulicht dieses Entscheidungskriterium graphisch. Für die Analyse des Systemverhaltens bedeutsam ist, dass auf den Grenzgeraden immer eine nichtdeterministische Entscheidung zwischen zwei gleichwertigen Optionen stattfindet, ohne dass es dafür ein sachlogisches Argument gäbe. Der Nichtdeterminismus bei den Entscheidungen des Kontrollalgorithmus macht es dem Assistenzsystem zusätzlich zur fehlenden expliziten Information über die aktuelle Situation also geradezu unmöglich, die zu Beginn dieses Abschnitts gestellten Anforderungen zur Erklärung verschiedener Handlungsphasen zu erfüllen: Dem Assistenzsystem steht ja als einzige Information zur Verfügung, dass aufgrund der gegebenen Wahrscheinlichkeitsverteilung für die drei möglichen Zustände der berechnete *immediate payoff* zur Auswahl einer bestimmten aus einer

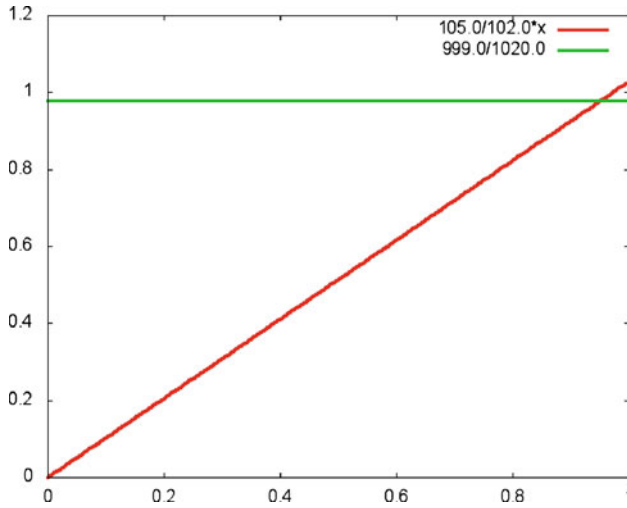


Abb. 2.11 Darstellung der Grenzgeraden, auf denen sich die Entscheidung für u_1 , u_4 oder u_5 ändert. Unter der gestrichelten Geraden fällt die Entscheidung für u_5 , über der gestrichelten und links von der durchgezogenen Geraden für u_1 , über der gestrichelten und rechts der durchgezogenen Geraden für u_4 .

Menge erlaubter Aktionen geführt hat. Diese Information ist aber für ein Assistenzsystem wenig zweckdienlich, weil sie die Auswirkungen von Aktionen für den Nutzer nicht oder höchstens auf sehr schwer verständliche Weise wahrnehmbar macht. Betrachten wir dazu die drei Möglichkeiten, die das Modell in Abb. 2.10 zulässt:

- $p_1 = 1, p_2 = p_3 = 0$: In diesem Fall wird folgende Bewertung berechnet:

$$V = 1020 - 1000 = 20$$

Es wird u_1 ausgewählt.

- $p_1 = p_3 = 0, p_2 = 1$: Ist die Assistenz also sicher unterbrochen, gilt:

$$V = 1050 - 1000 = 50$$

Es wird u_4 ausgewählt.

- $p_1 = p_2 = 0, p_3 = 1$: Ist die Assistenz beendet, gilt:

$$V = -1$$

Das System wählt u_5 aus, bestimmt also den Interessenstatus des Nutzers neu.



Abb. 2.12 Beispiele für aufeinander folgende Routenanweisungen des Auskunftssystems ROSE

Die drei Fälle zeigen Folgendes:

- Es wird jeweils diejenige Aktion gewählt, deren *immediate payoff* im aktuellen Zustand am höchsten ist.
- Wenn die Assistenz läuft, wird der Zustand nicht geändert. Das System überprüft den Interessenstatus nicht.
- Ist die Assistenz unterbrochen, wird sie sofort ohne Überprüfung des Interessenstatus wieder aufgenommen.

Diese Entscheidungen sind rational auf Grundlage des definierten *immediate payoff*. Obwohl die Festlegung seiner Werte zunächst plausibel schien, stellen sie sich jetzt dennoch als wenig problemadäquat heraus. Vor allem das Überprüfen des Interessenstatus kommt viel zu kurz. Der Nutzer kennt jedoch seine Interessen und wird Aktionen des Assistenzsystems immer in diesem Kontext interpretieren. Aus dieser Perspektive betrachtet, ist das Vorgehen des Assistenzsystems in vielen Situationen nicht nachvollziehbar.

Wenn das Assistenzsystem nun versucht, seine Aktionen plausibel zu machen (Erklärungsassistenz), muss es seine internen Fakten über die verschiedenen Phasen einer durchgeführten Aktion mitteilen. Dazu gehören insbesondere die Phase der Motiv- und Zielbildung sowie der Entscheidung, welche Aktion (aus einer Menge von Optionen) ausgewählt wurde. Die Motivbildung ist im Kooperationsbestreben des Assistenzsystems begründet, mit dem Nutzer ein gemeinsames Ziel, nämlich dessen aktuell zu lösende Aufgabe, zu erreichen. Die Erklärung der Zielbildung dient also vor allem der Vermeidung von Missverständnissen über die Aufgabe und der Planung und Durchführung einer Lösung für sie.

Die Erklärung, die gegeben werden kann, liegt also alleine im Modell für die Kontrolle begründet. Das Modell in Abb. 2.10 definiert drei Zustände, in denen sich das Assistenz-

system befinden kann; sie bilden zusammen mit der *payoff*-Funktion die Wissensbasis für Erklärungsassistenz. Bei einer Entscheidung für u_1 , also für die Fortführung der Assistenz steht dem ROSE-Navigationssystem auch noch die Information über das nächste Teilstück der Route zur Verfügung (siehe Abb. 2.12). Aus ihr wird die neue Erklärung dann formuliert. Bei Auswahl der Aktion u_5 wird eine Frage an den Nutzer gerichtet. Ob der Nutzer den Sinn der Frage zum aktuellen Zeitpunkt erkennt, und die Frage in die aktuelle Assistenz- und Interaktionssituation einordnen kann, darüber liegt im Assistenzmodell keine explizite Information vor. Dasselbe gilt für die Aktionen u_2 (Assistenz unterbrechen), u_3 (Assistenz beenden) und u_4 (Assistenz wiederaufnehmen). Was das Assistenzsystem also dem Nutzer über seine Entscheidung mitteilen kann, ist auf den berechneten Wert von V und die Strategie der Entscheidung für die Aktion mit dem höchsten Wert von V beschränkt. Eine Systemmeldung wie etwa

Ich gebe jetzt keine Weginformationen mehr; das ist das Beste, was ich gerade tun kann.

ist aber wohl für den Nutzer nicht hilfreich, um die Aktion des Systems in den Handlungsablauf zu integrieren. Der Meldung fehlt jeglicher Bezug zum aktuellen Geschehen und jede Begründung für die getroffene Entscheidung.

Dieser Kritik kann sicherlich durch eine andere *payoff*-Funktion begegnet werden; allerdings lässt sich auf Grund der schwer vorhersagbaren Einflussfaktoren auf V kaum verhindern, dass unspezifizierte Effekte, auftreten, deren Begründung keinen sinnvollen Bezug zur zu lösenden Aufgabe hat. Dieser Effekt wird sogar noch verstärkt, wenn nicht nur ein Schritt, sondern eine endliche Zahl T von Schritten in die Zukunft der *payoff* für T Schritte lange Folgen berechnet wird. In einer solchen Situation wird der Wert von V auch noch von den Zustandsübergangswahrscheinlichkeiten beeinflusst. Ist die zugehörige Verteilung nicht deterministisch, kumuliert sich die Unsicherheit weiter, ohne dass dazu eine sachlogische Erklärung möglich wäre.

Also wäre ein Ausweg, das Modell für die Assistenz zu verbessern und mit Hilfe von mehr Zuständen nicht nur den internen Zustand des Assistenzsystems, sondern auch Daten über die Benutzer, in unserem Beispiel insbesondere seinen Interessenstatus, zu modellieren. Derartige Zustände könnten beispielsweise den Status des Nutzers und den Status des Systems miteinander verbinden. Im bisher diskutierten Beispiel des ROSE-Navigationssystems gäbe es dann folgende Zustände:

$$\begin{aligned} &x_1/s_1, x_1/s_2, x_1/s_3, x_1/s_4 \\ &x_2/s_1, x_2/s_2, x_2/s_3, x_2/s_4 \\ &x_3/s_1, x_3/s_2, x_3/s_3, x_3/s_4 \\ &x_4/s_1, x_4/s_2, x_4/s_3, x_4/s_4 \end{aligned}$$

Danach lässt sich dann das Zustandsüberführungsmodell und die *payoff*-Funktion anpassen. In diesem ausführlicheren Modell sind mehr Informationen über den aktuellen Zustand enthalten: der Interessenstatus liefert eine Begründung für die Entscheidung, die dem Nutzer auch plausibel kommuniziert werden kann:

Anscheinend benötigen Sie zur Zeit gar keine Routenanweisungen. Deshalb werde ich keine Weginformationen mehr geben.

In einem interaktiven System, in dem der Nutzer auf Systemmeldungen reagieren kann, bestünde dann sogar die Möglichkeit zu widersprechen:

Doch, ich brauche immer noch Routenanweisungen!

Problematisch bei dieser Vorgehensweise, die aktuelle Situation durch Vergrößerung der Zustandsmenge präziser zu erfassen, ist jedoch, dass immer komplexere Verteilungen für die Wahrscheinlichkeiten, mit denen einzelne Zustände eingenommen werden bzw. mit denen der Übergang von einem Zustand in einen anderen modelliert wird, geschätzt werden müssen. In der Literatur hat sich die Erkenntnis aufgrund vieler praktischer Experimente (siehe z. B. [25, 28]) durchgesetzt, dass ein großer Zustandsraum nicht handhabbar ist, weil nicht genügend Daten für das Schätzproblem zur Verfügung stehen können. Dies bedeutet aus der Sicht der Entwicklung eines Assistenzsystems, dass stets ein Kompromiss zwischen Detailliertheit des Modells und seiner Lernbarkeit eingegangen werden muss. Dieser Kompromiss stellt einen Verlust an Information über die aktuelle Situation dar. Er zieht eine Verschlechterung der Fähigkeit des Assistenzsystems nach sich, Erklärungen zu generieren, so dass die aktuelle Situation vom Nutzer mental rekonstruiert werden kann. Dies ist aber eine wesentliche Voraussetzung für die erfolgreiche Integration von neuer Information zum Zweck der Situationserkennung.

2.3.5 Fähigkeit zur Relaxation

Eine weitere konsequente Forderung aus der Fähigkeit eines Assistenzsystems, eine Diagnose tatsächlich eingetretener, aber nicht erwarteter Effekte zu stellen, ist die Anforderung, dass das Assistenzsystem den Nutzer dabei unterstützen kann, die Phase der Motiv- und Zielbildung wieder aufzunehmen. Zum Zweck einer effizienten Orientierungsassistenz sollte ein Assistenzsystem den Wechsel zu einem anderen Ziel unterstützen können, falls für die ursprüngliche Aufgabe keine Lösung mehr möglich ist.

Ohne diese Fähigkeit würde ein Assistenzsystem die Unterstützung des Nutzers beenden, sobald der aktuelle Lösungsweg für eine Aufgabe sich als undurchführbar herausgestellt hat. Genau dieses Verhalten jedoch würde dem Wesen der Assistenz widersprechen, die, solange sie nicht explizit gestoppt wird, immer einen neuen Weg finden muss, die aktuelle Aufgabe zu lösen, solange ein solcher Weg noch existiert. Das neue Ziel sollte verständlicherweise den bisherigen so ähnlich wie möglich sein. Um ein neues Ziel zu finden, muss der Zustandsraum nach einem geeigneten Punkt, dem neuen Ziel, durchsucht werden.

Schwierig ist die Frage, von welchen Prinzipien die Suche gesteuert werden kann. Es gibt sehr viele Zustände, die vom aktuellen aus erreichbar sind, aber mit dem ursprünglichen Ziel gar nichts oder höchstens sehr wenig gemein haben. Alleine schon die Frage

der Ähnlichkeit zwischen Zuständen ist problematisch: Beschreibt man die Ähnlichkeit von Zuständen durch die Zahl identischer Eigenschaften zwischen den Zuständen, hat man noch lange kein geeignetes Kriterium für den Vergleich von Zuständen: War beispielsweise im Szenario des ROSE-Navigationssystems das bisherige Ziel durch eine Busverbindung am 15. September um 18:03 Uhr definiert, ist der Zustand, in dem die Busverbindung zum selben Ort, zur selben Zeit, aber am 17. Oktober gewählt wird, anhand der Zahl der identischen Eigenschaften dem ursprünglichen Ziel ähnlich, aber in pragmatischen Sinne alles andere als brauchbar. Algorithmische Verfahren, die es ermöglichen, einen Vorschlag für ein modifiziertes Ziel zu ermitteln, sind unter der Bezeichnung *Relaxationsverfahren* bekannt. Der Begriff Relaxation bedeutet, dass die Anforderungen an das bisherige Ziel so lange in möglichst kleinen Schritten abgemildert werden, bis ein neues, erfüllbares Ziel gefunden ist. Sie basiert auf heuristischen Funktionen, mit deren Hilfe der Nutzen und Schaden der Modifikation eines Zustands im Vergleich zum ursprünglichen Ziel bestimmt werden kann. Zlotkin und Rosenschein [29] definieren den Abstand $d(s, f)$ zwischen zwei Zuständen s und f , wenn es eine Folge $o = [o_1, \dots, o_k]$ von Aktionen gibt, die s in f überführt, unter Bezug auf die Kosten der einzelnen Aktionen:

$$\text{Cost}(o) = \sum_{1 \leq i \leq k} o_i \quad (2.1)$$

$$d(s, f) = \min_o \text{Cost}(o) \quad (2.2)$$

Zur Bestimmung von $d(s, f)$ müssen also alle Pläne durchsucht werden. $d(s, f)$ ist dann reflexiv und symmetrisch und erfüllt die Dreiecksungleichung; damit kann die Funktion als Metrik auf dem Zustandsraum definiert werden, mit deren Hilfe Abstände zwischen Zuständen ermittelt werden können¹⁴. Die Relaxation von Zielen ist aber auch unter Verwendung solcher Bewertungsfunktionen keine einfache Aufgabe. Insbesondere ist es schwierig, systematische Ansätze dafür zu entwickeln,

- welche Eigenschaften eines Ziels überhaupt relaxiert werden dürfen,
- welche Eigenschaften keinesfalls relaxiert werden dürfen,
- welche Eigenschaften eher als andere relaxiert werden dürfen,
- und welche Werte einer Eigenschaft für die Relaxation überhaupt interessant sind.

Ein Beispiel: wie kann ein Navigationssystem in einem Suchraum ein anderes dem Anwendungszweck angemessenes Ziel finden? Kann es eine Heuristik dafür geben? Was soll die Heuristik wie bewerten? Sprechen logische Argumente für eine Kombination von relaxierten Werten, die ein neues Ziel konstituieren, sprechen persönliche Argumente dafür oder sprechen Argumente über erwartetes Verhalten (z.B. des Nutzers bisher oder des Nutzertyps bisher) dafür?

¹⁴ Ähnliche Metriken werden im Fallbasierten Schließen (*case bases reasoning*) untersucht und angewandt. Einen Überblick über diese Thema gibt [30].

Aus technischer Sicht ist allen Varianten gemeinsam, dass sie einen Ansatz dafür zulassen sollen, mit dessen Hilfe die Eigenschaften des Ziels (oder eines als Ziel in Frage kommenden Zustands) zumindest partiell geordnet werden können. Zweck eines derartigen Ansatzes ist, die Suche nach einem alternativen Ziel heuristisch zu steuern. Zur Definition solcher heuristischer Ordnungen des Suchraums gibt es in der Literatur vier Ansätze:

1. Als Gewichtsvektor. Mittels Funktionen auf linear gewichteten Merkmalen (vergleiche etwa [31]) wird pro Merkmal ein Score berechnet, der mit anderen Merkmalen verrechnet wird. In der Regel ergibt sich dabei eine reelle Zahl als Bewertung für jeden Zustand, so dass Zustände sogar total geordnet sind.
2. Als probabilistisches Modell, meist unter Einsatz der BAYES-Regel ([32]) oder informationstheoretischen Größen wie Mutual Information (siehe [33]).
3. Als explizites relationales, symbolisch beschriebenes Modell der partiellen Ordnungen über mehrere Attribute hinweg (etwa in [34]).
4. Als logische Formeln ([35–38]).

Die Anforderung an ein Assistenzsystem, die Fähigkeit zu besitzen, Ziele relaxieren zu können, stellt also aus technischer Sicht eine weitere Erhöhung der Komplexität eines Assistenzsystems dar: es muss in der Lage sein, im Suchraum anhand partieller Ordnungen navigieren zu können, um unter Zuhilfenahme eines Ähnlichkeitsbegriffs im Suchraum schnell Alternativen für ein gegebenes Ziel finden zu können.

Für die Umsetzung aller in diesem Abschnitt aufgestellten Anforderungen bietet sich aufgrund ihrer Heterogenität eine hybride Systemarchitektur für die Realisierung eines Assistenzsystems an. Sie erlaubt es, für einzelne Assistenzfunktionen spezifische, möglichst gut geeignete Problemlöseverfahren zu integrieren. Auf diese Weise soll, wie in späteren Kapiteln weiter ausgearbeitet wird, die Realisierung konfigurierbarer, an verschiedene Anwendungen adaptierbarer Assistenzsysteme unterstützt und vereinfacht werden.

2.4 Zusammenfassender Überblick

Bevor jedoch das Konzept in seinen einzelnen Aspekten vertieft diskutiert wird, sollen zunächst noch einmal Anforderungen an Assistenzsysteme, die sich aus Sicht der Künstlichen Intelligenz an die Algorithmen zur Realisierung von Assistenz stellen, zusammengestellt werden:

- Assistenz ist **zielorientiert**. Ohne Zielorientierung kann das Assistenzsystem nicht prüfen, ob eine einzelne Handlung Konsequenzen hat, die später verhindern, dass der Benutzer sein Ziel erreichen kann.

- Assistenz ist **interaktiv**. Autonome Assistenz erfordert sichere Entscheidungen bezüglich des Ziels, welche in Szenarien mit subjektiven Parametern nicht getroffen werden können.
- Assistenz erfolgt **über mehrere Schritte hinweg**. Um das Ziel zu erreichen, müssen mehrere Teilziele, die sich gegenseitig beeinflussen, koordiniert werden.
- Assistenz kann auch bei partieller Beobachtbarkeit von Situation und Nutzer geleistet werden. Ohne diese Forderung könnte Assistenz nur in vollständig kontrollierten Umgebungen geleistet werden.
- Assistenz muss nichtdeterministische Aktionen behandeln können. Die tatsächlichen Effekte von Handlungen und das Verhalten der Umwelt sind nicht kontrollierbar.
- Assistenz *soll* in der Lage sein, existierende Problemlösekomponenten zu integrieren. Dafür sprechen methodologische Gründe wie die einfachere Erweiterbarkeit und Anpassbarkeit des Assistenzsystems. Mitunter sind die einzelnen Schritte der Assistenz so komplex (z. B. Navigation), dass ohnehin spezielle Repräsentationen und Algorithmen verwendet werden müssen.

Literatur

1. H. Wandke, Theor. Issues in Ergonomics Sci. **6**(2), 129 (2005)
2. M. Benmimoun, A. Pütz, A. Zlocki, L. Eckstein, in *Proceedings of the FISITA 2012 World Automotive Congress, Lecture Notes in Electrical Engineering*, vol. 197 (Springer Berlin Heidelberg, 2013), S. 537–547. 10.1007/978-3-642-33805-2_43. http://dx.doi.org/10.1007/978-3-642-33805-2_43
3. J.C. Giarratano, G.D. Riley, *Expert Systems: Principles and Programming*, 4th edn. (Course Technology, 2004)
4. K. Darlington, *The Essence of Expert Systems* (Prentice Hall, 1999)
5. J. Liebowitz (ed.), *The Handbook of Applied Expert Systems* (CRC Press, 1998)
6. E. Horvitz, J. Breese, D. Heckerman, D. Hovel, K. Rommelse, in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann, Madison, 1998), S. 256–265
7. J.F. Kelley, in *CHI '83: Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (ACM, New York, 1983), S. 193–196. <http://doi.acm.org/10.1145/800045.801609>
8. A. Dix, J. Finlay, G.D. Abowd, R. Beale, *Human Computer Interaction*, 3rd edn. (Pearson Education, 2004)
9. R.C. Houghton, Jr., Commun. ACM **27**(2), 126 (1984). <http://doi.acm.org/10.1145/69610.357985>
10. S. Card, T.P. Moran, A. Newell, *The Psychology of Human-Computer Interaction* (Lawrence Erlbaum, 1983)
11. C.D. Wickens, J. Lee, Y.D. Liu, S. Gordon-Becker, *Introduction to Human Factors Engineering*, 2nd edn. (Prentice-Hall, Upper Saddle River, 2003)
12. J. Nitschke, Assistenz bei auswahlprozessen. Ph.D. thesis, Humboldt-Universität zu Berlin (2004)

13. B. Ludwig, B. Zenker, in *IMC 2009, CCIS*, vol. 53, ed. by D. Tavangarian, T. Kirste, D. Timmermann (Springer, Rostock, 2009), *CCIS*, vol. 53, S. 97–107
14. D. Poole, A. Mackworth, R. Goebel, *Computational Intelligence* (Oxford University Press, 1998)
15. J. Pearl, *Causality* (Cambridge University Press, 2009)
16. A. Darwiche, *Modeling and Reasoning with Bayesian Networks* (Cambridge, 2009)
17. D. Koller, N. Friedman, *Probabilistic Graphical Models – Principles and Techniques* (MIT Press, 2009)
18. I.J. Myung, J. Math. Psychol. **47**(1), 90 (2003). [http://dx.doi.org/10.1016/S0022-2496\(02\)00028-7](http://dx.doi.org/10.1016/S0022-2496(02)00028-7)
19. L. Fahrmeir, R. Künstler, I. Pigeot, G. Tutz, *Statistik – Der Weg zur Datenanalyse*, 7th edn. (Springer, 2010)
20. H.W. Sorenson, *Parameter Estimation: Principles and Problems* (Marcel Dekker, 1980)
21. R. Reiter, *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems* (MIT Press, 2001)
22. H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun, *Principles of Robot Motion – Theory, Algorithms, and Implementations* (MIT Press, 2005)
23. N. Nilsson, *Principles of Artificial Intelligence* (Morgan Kaufmann, San Francisco, 1980)
24. M.L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley-Interscience, 1994)
25. S. Thrun, W. Burgard, D. Fox, *Probabilistic Robotics* (MIT Press, 2005)
26. J. Williams, S. Young, IEEE Audio, Speech and Lang. Process. **15**(7), 2116 (2007)
27. R.S. Sutton, A.G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, 1998)
28. B. Thomson, J. Schatzmann, K. Weilhammer, H. Ye, S. Young, in *NAACL-HLT '07: Proceedings of the Workshop on Bridging the Gap* (Association for Computational Linguistics, Morristown, 2007), S. 9–16
29. G. Zlotkin, J.S. Rosenschein, *Negotiation and Goal Relaxation* (Elsevier, 1991)
30. M.M. Richter, *Handbuch der Künstlichen Intelligenz* (Oldenbourg, 2000), chap. Fallbasiertes Schließen
31. M.T. Gervasio, M.D. Moffitt, M.E. Pollack, J.M. Taylor, T.E. Uribe, in *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces* (ACM, New York, 2005), S. 90–97. <http://doi.acm.org/10.1145/1040830.1040857>
32. I. Schwab, W. Pohl. Learning user profiles from positive examples (1999). citeseer.ist.psu.edu/schwab99learning.html
33. S.Y. Jung, J.H. Hong, T.S. Kim, IEEE Trans. on Knowl. and Data Eng. **17**(6), 834 (2005). 10.1109/TKDE.2005.86. <http://dx.doi.org/10.1109/TKDE.2005.86>
34. W. Kießling, in *VLDB '02: Proceedings of the 28th international conference on Very Large Data Bases* (VLDB Endowment, 2002), S. 311–322
35. G. Brewka, T. Eiter, Artif. Intell. **109**, 297 (1999)
36. G. Brewka, in *Fuzzy Sets, Logics and Reasoning About Knowledge*, ed. by H.P.D. Dubois, E.P. Klement (Kluwer, 1999), S. 381–394
37. G. Brewka, T. Eiter, in *Proc. 6th Intl. Conference on Knowledge Representation and Reasoning, KR-98* (Trento, Italy, 1998)
38. G. Brewka, J. Artif. Intell. Res. **4** (1996)



<http://www.springer.com/978-3-662-44818-2>

Planbasierte Mensch-Maschine-Interaktion in
multimodalen Assistenzsystemen

Ludwig, B.

2015, XI, 348 S. 82 Abb., Hardcover

ISBN: 978-3-662-44818-2