Ressourcenadaptierende Raumbeschreibung: Ein beschränkt-optimaler Lokalisationsagent

Dissertation
zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
der Technischen Fakultät
der Universität des Saarlandes

von

Anselm Blocher

Saarbrücken
3. Dezember 1999

4° H
76
6475

40 H 76-6475

Promotionskolloquium:	30. Dezember 1999
Dekan:	Prof. Dr. Wolfgang Paul
Gutachter:	Prof. Dr. Dr. h.c. Wolfgang Wahlster
	Prof. Dr. Jörg Siekmann

Kurzzusammenfassung

Das Ziel der vorliegenden Arbeit ist die Entwicklung eines künstlichen Agenten zur ressourcenadaptierenden, situationsangepaßten Raum- und Wegbeschreibung im Rahmen einer kognitiv adäquaten Mensch-Maschine-Interaktion. Dazu wurden bereits bestehende Verfahren wesentlich erweitert und neue entwickelt. Insbesonders wurden erstmals auf dem Gebiet der räumlichen Lokationsbeschreibungen wegbezogene Relationen und linguistische Heckenausdrücke formalisiert und in ein KI-System integriert. Der ressourcenadaptierende Charakter des Gesamtsystems, das innerhalb dieser Arbeit realisiert wurde, basiert auf der Verwendung unterbrechbarer Berechnungsverfahren, die in eine Management-Shell zur dynamischen Optimierung der Verteilung beschränkter Ressourcen eingebettet sind. Somit kann jederzeit ein mit wachsendem Ressourceneinsatz sich verbesserndes Resultat geliefert werden.

Short Abstract

In the framework of cognitive adequate human-computer-interaction, this work aims at developing an artificial agent for resource-adaptively describing space and path-like constellations. Therefore, computational methods that already existed had to be extended while others had to be developed entirely. In particular, path-like relations as well as linguistic hedges became for the first time formalized and integrated in an AI-system within the field of spatial descriptions. The resource-adapting character of this works´ system is based on interruptable computational methods embedded in a management shell to optimize the distribution of bounded resources. So, at any time a result can be provided that improves with increasing resources available.

Für meine Adid

Wenn er künftig einen weniger frivolen Gegenstand wählte und sich noch ein bißchen mehr zusammennähme, so würde er Dinge machen, die über alle Begriffe wären.

Johann Wolfgang von Goethe über Rodolphe Töpffer

La seule chose que je regrette dans ma vie, c'est de ne pas avoir fait de bandes dessinées.

Pablo Picasso

Au fond, mon seul rival international, c'est Tintin!

Charles de Gaulle

Ein Bild / Wort sagt mehr als tausend Worte / Bilder.

Danksagung

Diese Arbeit hätte natürlich nicht ohne meine Eltern Gisela Butzengeiger-Blocher und Dr. Rudolf Blocher entstehen können. Für ihre Liebe und die mir zuteil gewordene jahrelange, geduldige Unterstützung und ihr Vertrauen in meine Fähigkeiten möchte ich ihnen an dieser Stelle von ganzem Herzen danken.

Besonderer Dank gebührt Prof. Dr. Dr. h.c. Wolfgang Wahlster, der mir die Gelegenheit zur Promotion gab und dessen Förderung ich nicht vergessen werde.

Professor Dr. Jörg Siekmann schulde ich Dank für seine Bereitschaft, diese Arbeit als Zweitgutachter zu begleiten.

Aus dem Kreis meiner Kolleginnen und Kollegen am Lehrstuhl und im Sonderforschungsbereich, die eigentlich alle eine Erwähnung verdient hätten, möchte ich Frau Dr. Eva Bolz hervorheben, bei der ich mich bedanken möchte für die excellente Zusammenarbeit über fast ein Jahrzehnt hinweg und die vielen Brainstorming-Sitzungen, die diese Arbeit vorangebracht haben. Sehr geholfen bei der praktischen aber auch bei der theoretischen Ausarbeitung meines Themas haben alle *meine* Diplomanden, angefangen bei Dipl.-Inform. Christian Kray über Dipl.-Inform. Frank Wittig bis zu Axel Beckert. Ihnen allen bin ich zu Dank verpflichtet.

Meiner geliebten Frau Astrid danke ich aus ganzem Herzen, daß sie so lange zu mir gehalten hat und daß sie bei mir ist.

Die dieser Arbeit zugrundeliegende Forschung wurde von der Deutschen Forschungsgemeinschaft im Rahmen des Sonderforschungsbereichs 378 *„Ressourcenadaptive kognitive Prozesse“*, Teilprojekt A4: REAL (Ressourcenadaptierende Lokalisation) *„Objektlokalisaton und Sprachproduktion“*, gefördert.

Technische Anmerkungen

Angewandte Methoden

Bei der Erstellung des informatischen Systems eines beschränkt-optimalen Lokalisationsagenten, das in dieser Arbeit beschrieben wird, wurden vor allem Methoden der Künstlichen Intelligenz angewendet, wobei die Bereiche räumliches Schließen, Anytime-Algorithmen sowie Sprachverarbeitung im Vordergrund standen. Der interdisziplinäre Charakter des Sonderforschungsbereichs ermöglichte die Integration von Ergebnissen aus den Nachbardisziplinen der Kognitionswissenschaft. Neben einem starken Einfluß der Linguistik konnten insbesondere Resultate einer Kooperation mit der Psychologie Verwendung finden.

Entwicklung und Implementation des Systems erfolgten unter strikter Beachtung der Prinzipien der objektorientierten Programmierung in der KI-Programmiersprache Common Lisp mit dem Common Lisp Object System (CLOS). Das Kernsystem ist sowohl unter Lucid Common Lisp als auch unter Allegro Common Lisp auf unterschiedlichen Plattformen wie UNIX, Linux oder Windows NT lauffähig. Diese Plattformunabhängigkeit konnte durch die Verwendung der Programmiersprache Java auch für die graphische Oberfläche und die Benutzerschnittstelle erreicht werden. Letztere kommuniziert auf Dateiebene mit dem Kernsystem.

Verwendete Notation

Die in dieser Arbeit verwendeten Hervorhebungen haben folgende Semantik:

- neu eingeführte *Begriffe* werden kursiv gesetzt,
- `Funktionen, Relationen u.ä.` in Schreibmaschinenschrift und
- SYSTEM- ODER PROJEKTNAMEN in Kapitälchen;
- *„Zitate“* stehen kursiv in An- und Abführungszeichen während
- alle anderen *Hervorhebungen* geneigt erscheinen.

Die Literaturangaben entsprechen der gängigen APA-Konvention; innerhalb eines laufenden Satzes wird REAL (1996) zitiert, ansonsten erscheint die Referenz vollständig in runden Klammern: (vgl. (SFB 378, 1997)). Falls erforderlich werden die Verweise durch Angabe von Seitenzahlen präzisiert.

Inhaltsverzeichnis

Teil I

Interaktion von Objektlokalisation und Sprachproduktion

Der Mensch ist immer auf der Suche: Nach der Partnerin oder dem Partner fürs Leben, nach dem Sinn desselben oder einfach nach dem Weg zum Strand oder seiner Brille. Um sein Ziel zu erreichen, muß er häufig mit anderen Menschen, von denen er sich nützliche Informationen erhofft, kommunizieren.

Was die beiden ersten der genannten Suchen angeht, hat sich daran nichts geändert und das soll es auch gar nicht. Anders ist die Situation bei den beiden letzten Beispielen: Hier kann moderne Technik wertvolle Unterstützung leisten, wie das Beispiel des Siegeszuges der Fahrernavigationssysteme zeigt.

Doch egal wie hochentwickelt diese Co-Piloten auch sein mögen, können sie einen menschlichen Beifahrer (noch) nicht ersetzen: Seine Fähigkeit, Wegbeschreibungen auf veränderte Umweltbedingungen wie z.B. ein neu gebautes Haus oder unterschiedliche Fahrgeschwindigkeiten anzupassen, erleichtert dem Fahrer das Verständnis, selbst wenn manche Aussage vage oder suboptimal ist. Abbildung 1 illustriert dies am Beispiel von zwei Fahrzeugen, die sich mit stark unterschiedlichen Geschwindigkeiten einer Kreuzung nähern: Der LKW fährt langsam, so daß der Beifahrer eine ausführliche Wegbeschreibung geben kann, die u.a. mit der Nennung der Kirche eine Landmarke als ein Referenzobjekt enthält. Sowohl die Enkodierung als auch die Dekodierung durch den Fahrer, dessen Aufmerksamkeit durch die Teilnahme am Verkehr sowieso eingeschränkt ist, erfordert Zeit. Verringert sich die zur Verfügung stehende Zeit, etwa weil sich der PKW sehr schnell auf die Kreuzung zu bewegt, paßt der menschliche Co-Pilot seine Äußerung diesen Umständen an, und verbalisiert nur das absolut notwendige *„Gleich links!"*. Das Risiko, einer Fehlinterpretation wegen des Weglassens einer Verankerung durch eine Referenz wird mehr als aufgewogen durch die Tatsache, daß eine komplexere Äußerung erst *nach* Passieren der Kreuzung, also zu spät, hätte generiert, verstanden und in fahrerische Handlungen umgesetzt werden können (vgl. (Maaß, 1996)).

Das Verhalten sollte also dem Kooperationsprinzip *„Leiste Deinen Beitrag so, wie er vom gegenwärtigen Stadium des Dialoges und vom Dialogziel gefordert wird"* von Grice (1975) entsprechen. Der Erzeugung einer solchen situationsangepaßten Wegbeschreibung liegt ein ressourcenadaptierender Prozeß (vgl. (Wahlster & Tack, 1997; Jameson & Buchholz, 1998)) zugrunde, der Variationen von Ressourcen (hier der Zeit) berücksichtigt und gegebenenfalls

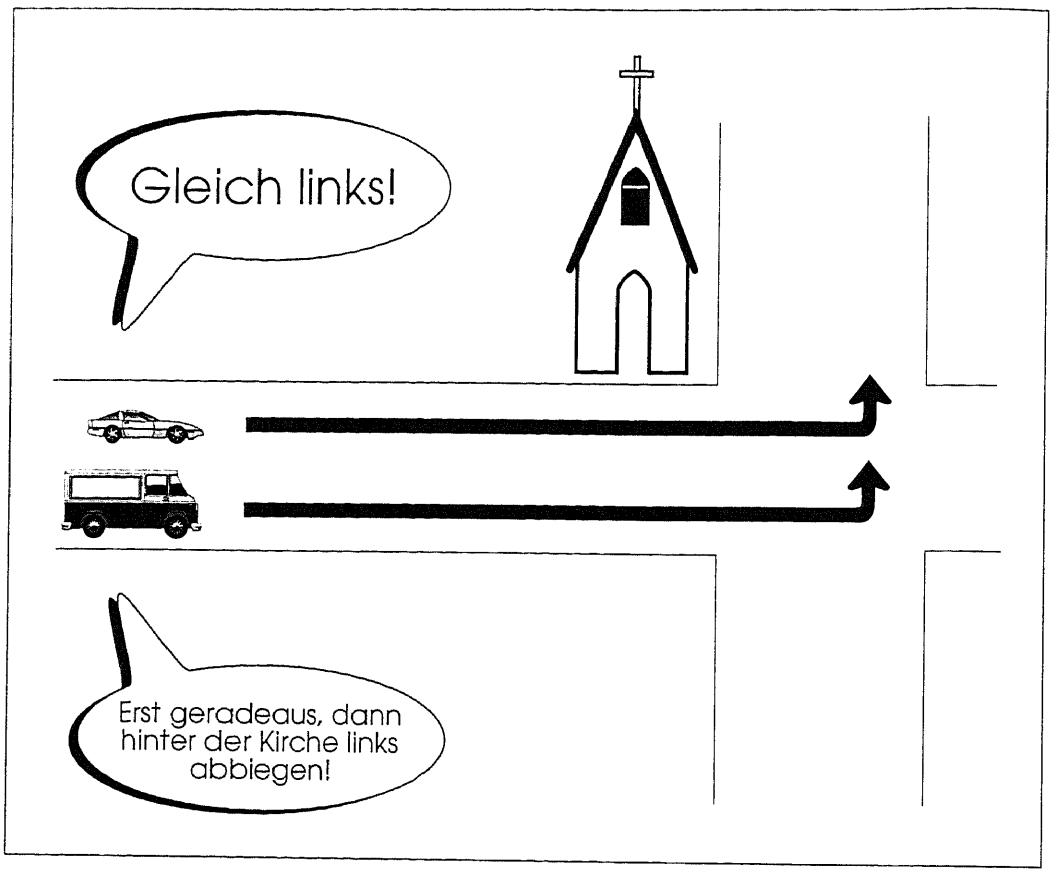

Abbildung 1: Ressourcenadaptierende Wegbeschreibung

unterschiedliche Verarbeitungsstrategien einschlägt.

Ziel der vorliegenden Arbeit als *ein* Schritt zu einer kognitiv adäquaten Mensch-Maschine-Interaktion ist die Entwicklung eines künstlichen Agenten, der ressourcenadaptierende Raumbeschreibungen erzeugen kann. Konkret soll das kognitive System in der Lage sein, unter variierenden Ressourcenbeschränkungen Wo-Fragen über seine aktuelle visuelle Umgebung in unterschiedlichen Raumszenarien durch adäquate Orts- und Wegbeschreibungen beantworten zu können.

Zur Erreichung dieses Ziels mußten in wissenschaftlicher Hinsicht inbesondere folgende Fragestellungen untersucht werden:

- *„Welche dynamischen Raumkonzepte gibt es, wie können sie formalisiert und schließlich in ein KI-System zur Generierung von Wegbeschreibungen integriert werden?“*

  Bislang wurden bei der Umsetzung von Konzepten der räumlichen Referenz hauptsächlich statische Relationen berücksichtigt (vgl. z.B. (Gapp, 1997)). Während somit bereits gute Grundlagen für statische Raumbeschreibungen erzielt werden konnten, werden in dieser Arbeit nun auch dynamische Konzeptualisierungen insbesondere für die Generierung von Wegbeschreibungen erforscht und eingesetzt.

  Dynamische Raumkonzepte, die z.B. durch Präpositionen wie *entlang*

ausgedrückt werden, zeichnen sich dadurch aus, daß sie nicht an einem einzelnen Punkt lokalisiert werden können; vielmehr müssen sie durch mehrere Punkte, deren Lage sich zeitlich und/oder räumlich unterscheidet, repräsentiert werden. In der vorliegenden Arbeit wird daher zuerst untersucht, welche dynamischen Raumkonzepte unterschieden werden können und wie sie mit natürlichsprachlichen räumlichen Ausdrücken korrespondieren. Daraus kann durch eine klare Formalisierung basierend auf Konzepten der statischen Relationen eine Menge von dynamischen Pfadrelationen gewonnen werden, die als Grundlage für die Generierung dynamischer räumlicher Beschreibungen, wie etwa Wegbeschreibungen, dient.

- *„Wie können linguistische Heckenausdrücke, die im Rahmen von natürlichsprachlichen Raum- und Wegbeschreibungen starke Verwendung finden, im Sinne einer besseren Mensch-Maschine-Kommunikation formalisiert und in ein System räumlicher Relationen integriert werden?“*

  Hier liegt ein völlig neuer Ansatz zur Optimierung der Verbalisierung räumlicher Ausdrücke vor: Erstmals werden Konzepte wie Vagheit und Präzision in diesem Kontext unter dem Aspekt der Raumbeschreibung aus informatischer Sicht untersucht und formalisiert, mit dem Ziel, eine resultierende Äußerung so zu modifizieren, daß sie den Grice'schen Maximen besser genügt.

- *„Wie wirken sich Ressourcenbeschränkungen auf Analyse und Generierung räumlicher Beschreibungen aus und wie können diese Effekte, im Sinne von Grice genutzt, in ein KI-System integriert werden?“*

  In einer engen Kooperation mit dem Fachbereich Psychologie konnten in experimentellen Studien abhängig von der jeweiligen Ressourcenlage unterschiedliche Verarbeitungsstrategien bei der Kommunikation über räumliche Lagebeziehungen zwischen menschlichen Versuchspersonen festgestellt werden. Die vorliegende Arbeit zeigt Möglichkeiten auf, durch die Verwendung von jederzeit unterbrechbaren Anytime-Algorithmen dieses Verhalten auf informatischer Ebene umzusetzen.

- *„Über welche Eigenschaften und Fähigkeiten muß ein System zur Generierung von Orts- und Wegbeschreibungen hinsichtlich der dynamischen Optimierung der Auswirkungen von Ressourcenbeschränkungen verfügen, und welche Modelle zur Verteilung dieser beschränkten Ressourcen können zum Einsatz kommen?“*

  Ausgehend von dem in (Zilberstein, 1993) präsentierten Ansatz der Compilierung von Anytime-Algorithmen wird in der vorliegenden Arbeit eine ressourcenadaptierende Management-Shell für ein System zur Generierung von Raumbeschreibungen vorgestellt. Durch ihre prinzipielle Unabhängigkeit von den eigentlich problemlösenden Verfahren konnte diese Shell universell einsetzbar entwickelt werden. Die verwendeten Methoden zur Optimierung der Ressourcenverteilung zeichnen sich

durch unterschiedliche Komplexität und verschiedene Herangehensweisen aus, wobei z.B. auch Erfahrungen aus früheren Durchläufen berücksichtigt werden können.

Durch die Bearbeitung dieser Fragestellungen und den vorgeschlagenen Lösungen werden bisherige Ansätze zur Generierung sprachlicher Raumbeschreibungen, wie z.B. der von Gapp (1997), der als Ausgangspunkt für diese Arbeit gedient hat, wesentlich erweitert.

Der beschränkt-optimale Lokalisationsagent nutzt die Management-Shell zur Kontrolle der die Berechnungen ausführenden jederzeit unterbrechbaren Anytime-Algorithmen. Die eigentliche Raumbeschreibung wird erzeugt durch mehrere interagierende Komponenten wie der Qualitätsbestimmung für sogenannte potentielle Referenzobjekte oder der Berechnung räumlicher Relationen. Die bereits vorhandenen Verfahren mußten hinsichtlich ihrer und der Anytime-Fähigkeit des Gesamtsystems stark angepaßt und erheblich erweitert werden. Zusätzlich werden erstmals Module zur Berechnung pfadbezogener räumlicher Relationen und linguistischer Hecken vorgestellt und integriert. Anwendbarkeits- und der neu eingeführte Präzisiongrad ermöglichen eine Optimierung des Resultats unter beliebigen Ressourcenbeschränkungen. Dabei besitzt das System die Fähigkeit, aus den gespeicherten Performanzdaten früherer Durchläufe, Schlüsse für die aktuelle Berechnung zu ziehen und so die Ressourcenverteilung zu optimieren.

Anwendungen für den Lokalisationsagenten sind neben den genannten Fahrernavigationssystemen auch die Ansteuerung autonomer mobiler Roboter (vgl. (Stopp, 1998)) oder die Aufwertung von PDAs zu virtuellen Reiseführern (vgl. (Deep Map, 1999)).

Die vorliegende Arbeit ist im Bereich der Künstliche Intelligenz (KI) angesiedelt und entstand im Rahmen des von der Deutschen Forschungsgemeinschaft (DFG) geförderten Sonderforschungsbereichs 378 *„Ressourcenadaptive kognitive Prozesse“* (SFB 378, 1997). An diesem interdisziplinären Projekt beteiligen sich Psychologen, Philosophen, Computerlinguisten und Informatiker der Universität des Saarlandes.

Das Teilprojekt REAL (Ressourcenadaptive Lokalisation) erforscht dabei die Interaktion von ressourcenbeschränkter Objektlokalisation und inkrementeller Sprachproduktion.

Abbildung 2 zeigt die Architektur von REAL: Eine konzeptuelle Ebene, der diese Arbeit angehört, befaßt sich auf einem abstrakten und sprachunabhängigen Niveau mit der Etablierung räumlicher Relationen. Hier spielen wie schon weiter oben erwähnt die Suche nach Referenzobjekten und ihre Bewertung sowie die Relationenberechnung selbst die herausragende Rolle. Die so generierte Objektlokalisation wird an die sprachliche Ebene weitergeleitet, die vorsprachliche Raumbeschreibungen in sprachspezifische Raumkonzepte abbildet und mittels einer Antizipationsrückkopplungsschleife das Hörerverständnis zu verbessern sucht. Parallel dazu wird auf dieser Ebene vor der eigentlichen Versprachlichung eine kontextbezogene Inhaltsplanung durchgeführt, um die Kohärenz des Dialogs zu gewährleisten.

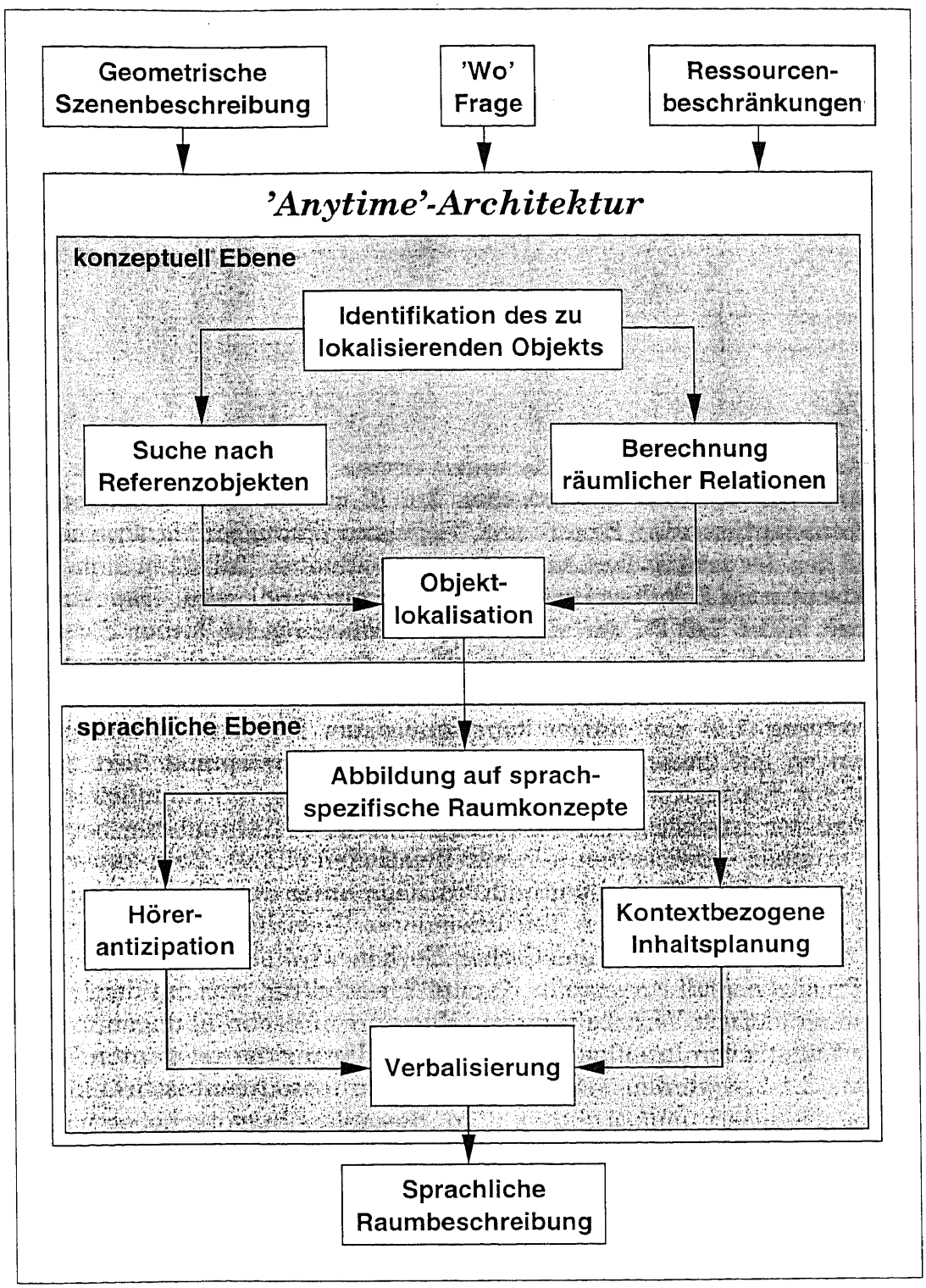

Abbildung 2: Grob-Architektur von REAL

Begleitend zur Entwicklung des beschränkt-optimalen Lokalisationsagenten wurden psychologische Raumkognitionsexperimente entwickelt und durchgeführt, die es ermöglicht haben, in einem ersten Schritt die Generierung von Raumbeschreibungen möglichst gut an das empirisch ermittelte Verhalten menschlicher Kommunikatoren anzupassen.

Die explizite Modellierung von ressourcensensitiven kognitiven Prozessen des Menschen durch einen künstlichen Lokalisationsagenten ist sinnvoll, um einem menschlichen Benutzer das Verständnis zu erleichtern und somit dem Ziel einer Optimierung der Mensch-Maschine-Interaktion näherzukommen.

***

Im Anschluß an diesen einführenden ersten Teil gliedert sich die vorliegende Arbeit in den eher theoretischen Teil II, in dem die Grundlagen einer ressourcenadaptierenden Raum- und Wegbeschreibung im Vordergrund stehen, und den der praktischen Anwendung gewidmeten Teil III, in dem der beschränkt-optimale Lokalisationsagent BOLA vorgestellt wird. Den Abschluß der Arbeit bildet Teil IV, der eine Zusammenfassung des bisher Erreichten und einen Ausblick auf weitere lohnenswerte Forschungsziele beinhaltet.

Teil II beginnt in Kapitel 1 mit einer einleitenden Diskussion zum Stand der Forschung (1.1) und einiger konstituierender Elemente räumlicher Beschreibungen auf diesem Gebiet (1.2-1.3). Der Schwerpunkt liegt zum einem auf der Konzeptualisierung und Formalisierung von räumlichen Relationen und der zugehörigen Berechnungsverfahren (1.4) und hier besonders auf den originär entwickelten n-Punkt-Relationen (1.4.4). Zum anderen hat die Analyse der Begriffe Vagheit und Präzision unter dem Gesichtspunkt der Raumbeschreibung in diesem Teil besonderes Gewicht (1.5): Neben einem Ansatz zur Modellierung linguistischer Hecken wird ein ressourcenadaptierendes Präzisionsmaß vorgestellt. Kapitel 2 beschäftigt sich mit der Notwendigkeit einer aktiven Verarbeitung von Ressourcenbeschränkungen in Systemen der Künstlichen Intelligenz. Nach einer Klärung der relevanten Begrifflichkeiten (2.1-2.4) werden bisherige Ansätze zur ressourcenbeschränkten Berechnung diskutiert und verglichen (2.5). Besonders hervorgehoben wird das Gebiet der Anytime-Berechnung als Basis für die folgenden Kapitel (2.6).

In Teil III werden zunächst die Grundlagen für den zu konstruierenden beschränkt-optimalen Lokalisationsagenten gelegt, indem in Kapitel 3 eine ressourcensensitive Architektur durch eine Anytime-System-Shell entwickelt wird. Nach der Darstellung grundlegender Eigenschaften und der Shell-Architektur (3.1) werden die Mechanismen zur Kontrolle von Modulen und Ressourcen thematisiert (3.2). Daran schließt sich die Beschreibung von Prozeßallokation und Scheduling an (3.3), bevor sich ein weiterer Schwerpunkt mit der dynamische Optimierung der Ressourcenverteilung durch Performanz-Profiling beschäftigt (3.4).

Aufbauend auf den in den vorangegangenen Kapiteln erarbeiteten Bausteinen werden diese in Kapitel 4 zu einem beschränkt-optimalen Lokalisationsagenten zusammengefügt. Zunächst wird ein Teilsystem zur Bewertung von Referenzobjekten vorgestellt (4.1); zentral sind jedoch die Berechnungsverfahren für räumliche Relationen und linguistische Hecken, da an ihnen exemplarisch die Vorgehensweise beschränkt-optimaler Systeme aufgezeigt werden kann (4.2). Einem Beispiellauf des Gesamtsystems (4.3) schließt sich eine erste Validierung des Ansatzes anhand psychologischer Experimente an (4.4).

Den vierten und letzten Teil der Arbeit bilden – wie oben erwähnt – Zusammenfassung und Ausblick.

Teil II

Ressourcenadaptierende Raumbeschreibung

Kapitel 1

Raum- und Wegbeschreibungen mit veränderlicher sprachlicher Präzision

Die kognitionswissenschaftliche Forschung hat in den letzten Jahren durch interdisziplinäre Ansätze zwischen den Bereichen Linguistik, Psychologie, Informatik, Neurowissenschaft und auch Philosophie eine breite Grundlage zum besseren Verständnis des Gebietes der *Raumbeschreibung* erarbeitet.

Dieses Kapitel widmet sich zuerst dem Thema der *Raumreferenz*, indem neben der Einführung einiger Begriffe auch die Grundlagen dieser Arbeit vorgestellt werden. Danach erfolgt eine Analyse der Konstituenten einer Raumreferenz, das *Referenzobjekt* und die *räumliche Relation*. Die Konzeptualisierung und Formalisierung räumlicher Relationen bilden den Kern dieses Kapitels. Insbesondere werden Pfadrelationen behandelt, die in dieser Arbeit erstmals für ein System zur Raum- und Wegbeschreibung entwickelt und integriert wurden. Schließlich wird intensiv auf die Möglichkeit einer ressourcenadaptierenden Präzisierung von Lokalisationsausdrücken durch die Verwendung von *linguistische Hecken* eingegangen, die zu diesem Zweck analysiert und formal beschrieben wurden.

1.1 Raumreferenz

Bevor das Konzept *Raumreferenz* als die Bezugnahme eines Sprechers auf räumliche Gegebenheiten (vgl. (Herrmann & Grabowski, 1994; Schweizer, 1985; Wunderlich, 1982)) in den Kontext von visueller Wahrnehmung und Sprache gestellt wird, soll auf den Begriff der *Referenz* im allgemeinen eingegangen werden.

1.1.1 Referenz

Der Begriff der *Referenz* spielt bei der wissenschaftlichen Beschäftigung mit der Interpretation und Verarbeitung natürlicher Sprache – neben anderen Ausdrucksformen des Menschen wie Syntax, Semantik, Intention und Sprech-

akt – eine zentrale Rolle. Im Bereich der Künstlichen Intelligenz trifft dies besonders auf dialogbasierte Systeme, wie z.B. Navigationsassistenten im Auto zu. Hier muß der Hörer in der Lage sein, Referenzen aufzulösen, um zu verstehen, was der Sprecher sagen will.

Searle (1969) (deutsche Übersetzung: (Searle, 1994)) unterscheidet vier Arten von *Sprechakten*:

1. *Äußerungsakte* sind Äußerungen von Wörtern, Morphemen, Sätzen.
2. *Propositionale Akte* reflektieren den Bedeutungskern einer Äußerung, der den Wahrheitswert eines Satzes bestimmt (vgl. (Bußmann, 1990)). dabei wird unterschieden zwischen
   (a) *Referenz*, dem Bezug auf Objekte und Sachverhalte der realen Welt und
   (b) *Prädikation*, dem Zusprechen von Eigenschaften an die Referenten.
3. *Illokutionäre Akte* repräsentieren vollständige Sprechakte, z.B. Behaupten, Fragen, Befehlen, Versprechen etc.
4. *Perlokutionäre Akte* beziehen sich auf Konsequenzen und Wirkungen für Handlungen, Gedanken u.ä. eines Hörers (vgl. (Austin, 1962)).

Die Generierung *propositionaler Akte* ist zentrales Thema dieser Arbeit. Während *Referenz* nach Searle (1994, S. 44) ein *hinweisenden Ausdruck* ist, *„der dazu dient, ein Ding, einen Prozeß, ein Ereignis, eine Handlung oder sonstige Arten von Individuen oder partikularen Objekten zu identifizieren"*, erlaubt die *Prädikation*, Objekte anderen gegenüber abzuheben, so daß sie z.B. als Referenten verwendet werden können. Beides ist für Lokalisationsbeschreibungen unabdingbar. Hinweisende Ausdrücke lassen sich wie folgt klassifizieren:

- bestimmt / unbestimmt: *das Haus / ein Haus*;
- singulär / pluralisch: *das Haus / die Häuser*;
- individuell / universell: *das Haus / Häuser*;
- paradigmatisch: Eigennamen, Nominalausdrücke mit bestimmtem Artikel, Possessivpronomen und Pronomina.

Die *singuläre bestimmte Referenz* hat bei Ortsangaben eine dominante Stellung:

„Das Fahrrad steht vor dem Haus."

Zur Analyse dieser Gruppe hinweisender Ausdrücke hat Searle (1994, S. 121-126) sogenannte *Axiome der Referenz* aufgestellt:

- *Axiom der Existenz: „Alles, worauf verwiesen wird, muß existieren."*
- *Axiom der Identität: „Wenn ein Prädikat einem Objekt zukommt, so kommt es allem zu, was mit dem Objekt identisch ist."*

- *Axiom der Identifikation: „Wenn ein Sprecher auf einen Gegenstand verweist, dann identifiziert er diesen Gegenstand abgesondert von allen anderen Gegenständen für den Zuhörer oder ist in der Lage, dies auf Verlangen jederzeit zu tun."*

Eine im Sinne des Verstehens beim Hörer erfolgreiche Referenz muß sich demnach auf *genau ein existierendes* Objekt beziehen, das der Partner mit den ihm zur Verfügung stehenden Mitteln identifizieren kann. Die Identifikation kann dabei vom Sprecher durch *Prädikation* erleichtert bzw. überhaupt erst ermöglicht werden.

1.1.2 Sprache und Sehen

Die vorliegende Arbeit steht im Kontext der Entwicklung wissensbasierter Systeme zur Integration von visueller Wahrnehmung und der Verarbeitung natürlicher Sprache. Dabei soll zum einem aus kognitionswissenschaftlicher Sicht eine algorithmische Beschreibung der komplexen Verarbeitungsprozesse entwickelt werden, die dieser Interaktion zugrunde liegen (vgl. (Wahlster, 1989)). Zum anderen wird aus ingenieurwissenschaftlicher Sicht daran gearbeitet, die Resultate eines bildverarbeitenden Systems Benutzern besser zugänglich und leichter verständlich zu machen. Wichtige Grundlagen für die Modellierung der Beziehung zwischen Sprachgenerierung und Bildverstehen wurden insbesondere innerhalb des Projektes VITRA (Visual Translator) im SFB 314 *„Künstliche Intelligenz – Wissensbasierte Systeme"* gelegt (vgl. (Herzog, Blocher, Gapp, Stopp & Wahlster, 1996)). Im Unterschied zu früheren natürlichsprachlichen Zugangssystemen wie HAM-ANS (vgl. (Wahlster, Marburger, Jameson & Busemann, 1983)) und NAOS (vgl. (Neumann & Novak, 1986)), die lediglich *a posteriori* Analysen retrospektiver Szenenbeschreibungen ermöglichten, konnte hier erstmals automatisch sprachliche Beschreibungen für aus realen Bildfolgen gewonnene Trajektorien simultan zum Fortschreiten der Szenenfolge inkrementell erzeugt werden[1].

Nach Herzog, Rist und André (1990) müssen beim natürlichsprachlichen Zugang zu bildverstehenden Systemen drei Verarbeitungsebenen unterschieden werden:

- Die *sensorische Ebene* leistet den Übergang von der Bildfolge zur Szenenfolge im dreidimensionalen Raum indem sie die beobachtbaren Objekte identifiziert und klassifiziert.

- Die *konzeptuelle Ebene* realisiert den Schritt von der rekonstruierten geometrischen Szenenbeschreibung (vgl. (Neumann & Novak, 1986)) zu einer konzeptuellen Beschreibung des Szenengeschehens mit dem Ziel einer *referenzsemantischen Verankerung* der sprachlichen Entitäten (vgl. (Carsten & Janson, 1985; Fürnsinn, Khenkhar & Ruschkowski, 1984; Hußmann & Schefe, 1984)).

[1]Diese Bildfolgen realisierte die Gruppe von Prof. Dr. Nagel (IITB Karlruhe) im Rahmen einer Kooperation im SFB 314 (vgl. (Nagel, 1985, 1988)).

- Die *linguistische Ebene* schließlich überführt die aus der Perzeption gewonnenen konzeptuellen Strukturen in natürlichsprachliche Äußerungen.

Die notwendige perzeptuelle (bzw. korrekt geometrische) Verankerung räumlicher Referenzausdrücke erfolgt demnach auf der konzeptuellen Ebene durch die Definition einer *Referenzsemantik*. Dabei wird die Klasse der Lokalisationsausdrücke nach Herskovits (1986) durch Präpositionen in ihrer räumlichen Bedeutung (vgl. (Retz-Schmidt, 1988)) zusammen mit dem zu *lokalisierenden Objekt (LO)* und einem oder mehreren *Referenzobjekten (RO)* gebildet. Die semantische Analyse dieser Ausdrücke führt dann zum Begriff der *räumlichen Relation (Rel)* als einzelsprachunabhängiges Konzept (vgl. (Herskovits, 1986; Landau & Jackendoff, 1993; Pribbenow, 1991; Talmy, 1983)), das – ganz im Sinne von Searle – die Kernbedeutung einer Referenz in Form einer *Proposition* repräsentiert.

Das folgende Beispiel soll diesen elementaren Sachverhalt illustrieren:

(1) *„Die Lampe hängt über dem Tisch."*

(2a) *„Das Auto steht vor dem Haus."*

(2b) *„Das Auto parkt vor dem Haus."*

Abstrahiert man bei den Lokalisierungsausdrücken (1) bis (2b) von nichträumlicher Information, erhält man eine sprachunabhängige Darstellung der Lagebeziehung zwischen LO und RO. Unter Verwendung einer allgemeinen propositionalen Notation `( Rel LO RO )` entspricht dies:

(1') `( über Lampe Tisch )`

(2') `( vor Auto Garage )`

Da die in den Sätzen (2a) und (2b) enthaltene räumliche Information identisch ist, können beide Äußerungen durch dieselbe Proposition (2') repräsentiert werden. Die konkrete Definition räumlicher Relationen wird in Abschnitt 1.4 behandelt.

Neben der Bestimmung räumlicher Beziehungen und relevanter Bewegungsvorgänge erkennt die Szenenfolgenanalyse von VITRA auch vermutete Handlungsintentionen und Planinteraktionen beobachteter Agenten (vgl. (Retz-Schmidt, 1992)). Ein vorläufiger Textplan wird durch ein als *Antizipationsrückkopplungsschleife* konzipiertes *Hörermodell* auf Verständlichkeit und Korrektheit überprüft und gegebenenfalls korrigiert, um eine optimale Verständlichkeit für den Hörer zu gewährleisten (vgl. (Blocher, 1994; Schirra, 1994; Blocher & Schirra, 1995; Stopp & Blocher, 1996)). Dies entspricht dem Kooperationsprinzip von Grice (1975), demzufolge eine Äußerung alle relevanten Fakten enthalten soll, jedoch keinerlei redundante Information, weder explizit noch anderweitig für den Hörer erschließbar. Auf diesen Anspruch wird in Abschnitt 1.5 im Zusammenhang mit der Präzision von räumlichen Lokalisationsaussagen näher eingegangen.

In VITRA wurde die Interaktion von Objektlokalisation und Sprachproduktion unter unterschiedlichen Anwendungsszenarien untersucht:

- Die Beantwortung von Fragen über Beobachtungen in einer Straßenverkehrsszene (vgl. (André, Bosch, Herzog & Rist, 1986; Schirra, Bosch, Sung & Zimmermann, 1987)).
- Die Generierung von Simultanbeschreibungen
  - anhand kurzer Ausschnitte aus Videoaufnahmen eines Fußballspiels (vgl. (André, Herzog & Rist, 1988; Herzog et al., 1989)) bzw.
  - von verschiedenen Straßenverkehrsszenen mit Fahrzeugen und Fußgängern (vgl. (Herzog & Rohr, 1995)).
- Die Erzeugung inkrementeller, multimodaler Wegbeschreibungen basierend auf einem 3D-Modell des Saarbrücker Campus (vgl. (Herzog, Maaß & Wazinski, 1993; Maaß, 1994)).
- Die natürlichsprachliche Interaktion mit einem autonomen mobilen Robotersystem (Längle, Lüth, Herzog, Stopp & Kamstrup, 1995; Stopp, Gapp, Herzog, Längle & Lüth, 1994; Stopp, 1998).

Diese Arbeit baut auf Vorarbeiten des Projektes VITRA insbesondere hinsichtlich der Behandlung räumlicher Relationen auf (vgl. hierzu 1.4)[2]. Der von Gapp (1997) vorgestellte Ansatz zu deren Verarbeitung wird gleich in mehrere Richtungen erweitert: Neben der Integration pfadbezogener räumlicher Relationen wie `entlang`, werden Verfahren zur Berechnung der Präzision beliebiger gradierter Konzepte vorgestellt. Diese finden Verwendung bei der Verarbeitung von linguistischen Heckenausdrücken, die ihrerseits auf räumliche Relationen angewendet werden können. Eine letzte wesentliche Erweiterung des Ansatzes liegt in der konsequenten Berücksichtigung von Ressourcenbeschränkungen bei der derart verfeinerten Generierung von Lokalisationsausdrücken.

Verwandte Arbeiten zu dem hier vorgestellten Ansatz finden sich zum einen im DFG-Schwerpunktprogramm *„Raumkognition"*, das sich mit dem Erwerb, der Revision, der Organisation und dem Umgang mit Wissen über räumliche Umgebungen befaßt. Dabei werden im Themenbereich *„Raumrepräsentation und höhere kognitive Prozesse"* Basiskonzepte und -prozesse erforscht. Die für die Raumkognition zentralen räumlichen Konzepte sollen mit Hilfe von Axiomen formal charakterisiert werden (vgl. (Eschenbach & Habel, 1995)). Zur Untersuchung der Modellierung räumlich organisierter Wissenstrukturen dienen sogenannte Aspektkarten, die eine zugrundeliegende räumliche Struktur repräsentieren (vgl. (Barkowsky, Berendt, Freksa & Kelter, 1997)). Für die beispielhafte Ansteuerung eines autonomen Rollstuhls werden zur Bearbeitung der Navigationsaufgabe eine hierarchische Taxonomie von Landmarken-, Routen- und Überblickswissen verwendet (vgl. (Werner, Krieg-Brückner, Mallot, Schweizer & Freksa, 1997)).

In einem weiteren interessanten Ansatz untersucht der Bielefelder SFB 360 *„Situierte Künstliche Kommunikatoren"* kognitive Leistungen in einer

[2]Im Rahmen dieser Vorarbeiten ist zu erwähnen, daß bereits erste Schritte in Richtung auf eine unterbrechbare Raumbeschreibung gemacht wurden (vgl. (Blocher & Stopp, 1998)).

natürlichen Kommunikationssituation anhand der Montage eines Baufix--Flugzeugmodells durch einen autonomen Roboter. Die visuelle und verbale Wissensrepräsentation basiert auf einer integrativen Architektur durch das prozedurale semantische Netzwerk System ERNEST (Erlangen semantic network system). Der künstliche Kommunikator nimmt über Bild- und Sprachsensoren seine Umwelt und Handlungen wahr und verarbeitet verbale Objektlokalisationen in einem hybriden Ansatz zu einer konzeptuellen Beschreibungen (vgl. (Socher, Fink, Kummert & Sagerer, 1996)). Im Projekt CODY (Concept Dynamics) werden räumliche Ausdrücke im dreidimensionalen Raum evaluiert (vgl. (Vorwerg, Socher, Fuhr, Sagerer & Rickheit, 1997)).

Im Gegensatz zu dem in der vorliegenden Arbeit vorgestellten System eines beschränkt-optimalen Lokalisationsagenten fehlen bei den oben beschriebenen Absätzen ressourcenadaptierende Aspekte nahezu völlig. Dies gilt in eingeschränktem Maße auch für die Verarbeitung pfadbezogenen Relationen und von Vagheit. Andererseits kann festgestellt werden, daß trotz unterschiedlicher Herangehensweisen auf einigen Teilgebieten vergleichbare Ergebnisse erzielt wurden. Dies bezieht sich insbesondere auf die Modellierung und Verarbeitung statischer räumlicher Relationen sowie experimentelle Resultate.

1.2 Referenzrahmen

Die Analyse einer räumlichen Relation erfolgt immer innerhalb eines Bezugssystems, das als *Referenzrahmen* oder *Referenzsystem (RS)* bezeichnet wird und eine Perspektive induziert (vgl. (Herrmann & Grabowski, 1994; Retz-Schmidt, 1988)). Ein solches Bezugssystem erst erlaubt es Sprecher und Hörer, Perzeption und Sprache auf korrespondierende mentale Repräsentationen räumlicher Relationen abzubilden (ohne daß etwa unterschiedliche Blickrichtungen angenommen werden), um somit eine gemeinsame Kommunikation zu ermöglichen (vgl. (Miller & Johnson-Laird, 1976)). Dabei wird zwischen *egozentrischen* (*deiktischen*) und *allozentrischen* Referenzsystemen unterschieden. Erstere werden durch den Aufenthaltsort des Sprechers etabliert, während sich letztere weiter untergliedern: Der Ursprung *intrinsischer* Referenzrahmen liegt in einer zweiten Person oder einem Objekt, wobei die Richtung durch eine sogenannte *prominente Front* festgelegt wird. Dies kann z.B. durch die Blickrichtung der Person, die Fahrtrichtung eines Wagens oder die Eingangstüre eines Hauses geschehen. Von *extrinsischer* Orientierung letztlich spricht man, wenn die Richtung durch ein explizit gemachtes, weiteres Objekt bestimmt wird. In diesem Zusammenhang spricht man auch von intrinsischer, extrinsischer oder deiktischer *Gebrauchsart* einer räumlichen Relation.

Die oben eingeführte formale Notation für Propositionen wird um die optionale Angabe des Referenzsystems zu `( Rel LO RO RS )` erweitert.

Die unterschiedlichen Arten der Etablierung eines Referenzsystems spielen insbesondere bei Präpositionen wie *links* oder *über* eine Rolle (vgl. Abschnitt 1.4.3). Da bei den in dieser Arbeit vorgestellten Verfahren die Art und

Weise der Etablierung eines Referenzrahmens nur eine untergeordnete Rolle spielt, wird zu diesem Thema auf (Maaß, 1996) und (Gapp, 1997) verwiesen.

1.3 Referenzobjekte

Ein wichtiger Bestandteil von Lokalisationsbeschreibungen sind *Referenzobjekte*. Sie dienen zur Verankerung der räumlichen Referenz, die dadurch aufgelöst werden kann. Neben der von Searle oben erwähnten *Prädikation*, die durch das Zusprechen von Eigenschaften Referenzobjekte (leichter) identifizierbar macht, sind *Idealisierungen* und *Abstraktionen* häufig zu beobachtende Phänomene bei der Wahl geeigneter Objekte.

1.3.1 Eigenschaften von Referenzobjekten

Damit bestimmte Objekte als Ankerpunkte für Referenzen dienen können, müssen sie über Eigenschafen verfügen, die sie hervorheben gegenüber allen anderen Objekten und speziell gegenüber denjenigen, von denen sie als Referenz benutzt werden. In einem ersten Schritt zur Bestimmung eines Referenzobjekts muß eine Vorauswahl erfolgen, welche Objekte überhaupt in Frage kommen. Nach Habel und Pribbenow (1988) liegen sie in einer Nähe-Umgebung um das zu lokalisierende Objekt, wobei die Ausdehnung dieser Region mit der des LO korrespondiert. So ist diese Umgebung bei einem Haus wesentlich größer als bei einem Buch. (Bei der Bestimmung eines solchen Bereiches können auch kontextuelle Abhängigkeiten eine Rolle spielen.)

Damit läßt sich ein erstes Kriterium festhalten, das Einfluß auf die Qualität eines Objektes als Referenz hat:

- *Distanz* spielt nicht nur bei der Ermittlung potentieller Referenzobjekte eine wichtige Rolle: Prinzipiell steigt die Qualität eines Referenzobjektes mit abnehmender Entfernung zu dem zu lokalisierenden Objekt.

Konnte wie eben beschrieben eine Menge potentieller Referenzobjekte gefunden werden, muß eine Bewertung ihrer Qualität als Referenz vorgenommen werden. Ausgehend von den Erkenntnissen einer experimentellen Clusteranalyse von (Sadalla, Burroughs & Staplin, 1980) unterscheidet Gapp (1996, 1997) zwei Klassen relevanter Eigenschaften von Referenzobjekten:

- *Objektspezifische Merkmale* sind direkt mit einem potentiellen Referenzobjekt verbundene Eigenschaften, die zur Abgrenzung untereinander, aber auch gegenüber dem zu lokalisierenden Objekt (vgl. (Talmy, 1983)) verwendet werden:
  - *Visuelle Salienz* setzt sich nach Treisman (1988) aus folgenden Charakteristika zusammen:
    * *Farbe*, insbesondere ein Farbunterschied zur Umgebung, ist einer der wichtigsten Faktoren der visuellen Salienz, da sie besonders einfach vom Menschen wahrgenommen werden kann (vgl. (Carter & Carter, 1981)).

  * *Größe* ist hier nicht absolut gemessen zu verstehen, sondern innerhalb der perzeptiven Unschärfe als relativ. In der Regel sind Referenzobjekte nicht kleiner als die Objekte, auf die sie verweisen.
  * *Form* in Sinne von Auffälligkeit innerhalb einer ansonsten gleichartigen Menge ist ebenfalls eine relatives Konzept.
- *Mobilität* eines Referenzobjekts kann dieses unbrauchbar werden lassen, es sei denn, das zu lokalisierende Objekt bewegt sich auch. Statische Objekte sind in der Regel bessere Referenzen, da keinerlei Unsicherheit über ihren Aufenthaltsort zu einem späteren Zeitpunkt existiert.
- *Referenzsystem*: Hier sind Objekte zu präferieren, die über eine intrinsische Front verfügen (vgl. Abschnitt 1.2), da verschiedene Untersuchungen zeigen, daß eine intrinsische Gebrauchsart von Relationen häufig vorgezogen wird (vgl. z.B. (Ehrich, 1985)). Insgesamt ist die Frage der Wahl eines Referenzsystems allerdings noch nicht geklärt (vgl. (Schober, 1995)).

- *Kontextuelle Merkmale* sind durch die kommunikative Situation gegeben; sie basieren also nicht auf inhärenten Eigenschaften der betrachteten Objekte.

  - *Identifizierbarkeit* ist für Referenzobjekte eine essentielle Voraussetzung. Sie kann beispielsweise durch Verdeckungen eingeschränkt werden.
  - *Abschirmung durch Störobjekte* tritt auf, wenn sich zwischen zu lokalisierendem und Referenzobjekt noch weitere Objekte befinden. Die so mögliche Abdeckung vermindert die Qualität des Referenzobjektes.
  - *Funktionale Abhängigkeiten*, wie z.B. zwischen einem Computer und einem Monitor, beschreiben nicht-räumliche Beziehungen zwischen Objekten, die eine Lokalisation erleichtern können (vgl. (Hirtle & Heidorn, 1993)).
  - *Vorwissen* kann die Wahl eines Referenzobjekts dadurch beeinflussen, daß entweder bestimmte Präferenzen des Sprechers verstärkt werden, oder daß auf den Hörer besondere Rücksicht genommen wird.
  - *Vorerwähntheit* spielt insbesondere in Situationen eine Rolle, in denen mehr als eine Lokalisierung erforderlich ist, wie z.B. bei Wegbeschreibungen. Durch die Referierung auf bereits übereinstimmend bekannte Objekte kann eine Folgeäußerung besser verständlich sein.

Zur Bewertung potentieller Referenzobjekte ist es notwendig, eine geeignete Kombination der vorgestellten Eigenschaften (incl. der *Distanz*) zu bestimmen. Gapp (1997) schlägt dazu einen gewichteten Merkmalsvektor vor.

1.3.2 Abstraktion und Idealisierung

Bei der Verwendung von räumlichen Relationen in Lokalisationsbeschreibungen spielen Details in der Regel keine Rolle (vgl. (Landau & Jackendoff, 1993)): Der Mensch kümmert sich, wenn er etwa ein Haus als Referenz benennt, relativ wenig um solche Einzelheiten wie einen Erker und abstrahiert davon. Dies kann die Generierung räumlicher Ausdrücke erleichtern, da bei der geometrischen Repräsentation von Objekten vorhandene Details eventuell vernachlässigt werden können, ohne die Qualität der Referenz zu gefährden. Im Gegenteil kann diese durch eine aus einer Weglassung resultierenden größeren Klarheit eventuell sogar verbessert werden.

Als Idealisierungen komplexer Objekte werden häufig Vereinfachungen in Form fixer Klassen wie Schwerpunkt oder umschreibender Quader herangezogen (vgl. (Miller & Johnson-Laird, 1976; André, Bosch, Herzog & Rist, 1987)). Abbildung 1.1 zeigt eine ausführliche Klassifikation gängiger Idealisierungen im zwei- und drei-dimensionalen Raum.

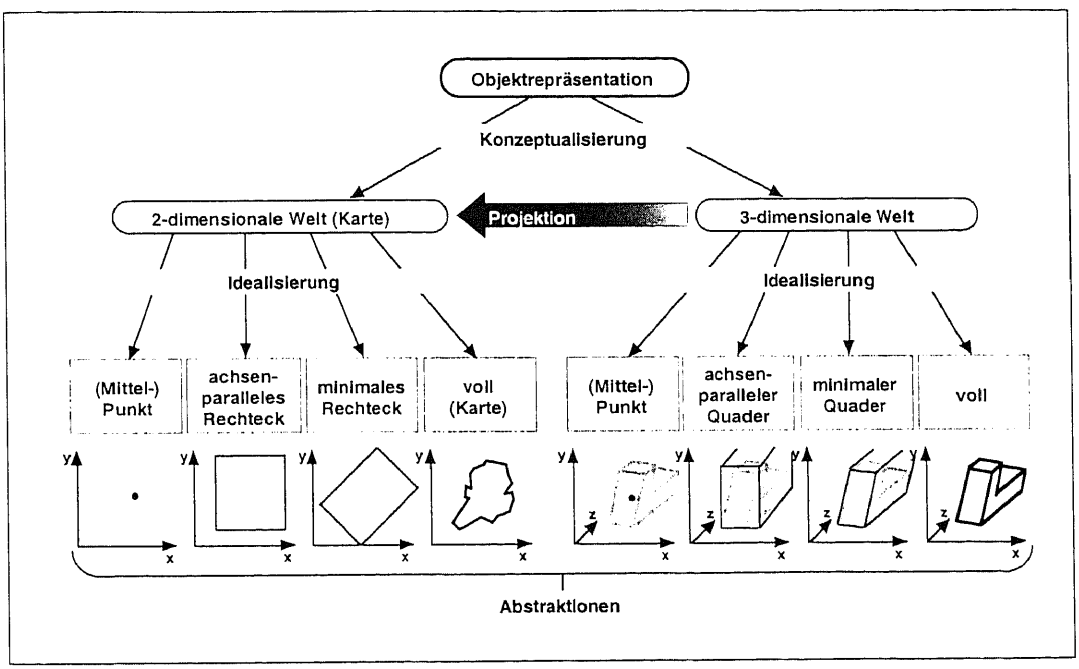

Abbildung 1.1: Klassifikation von zwei- und drei-dimensionalen Idealisierungen

Diese Idealisierungen sind zwar häufig ausreichend, ein flexibles, kontextsensitives Verfahren zur Generierung stufenlos ineinanderübergehender Idealisierungen zur Laufzeit, wie es von Krüger (1999) vorgestellt wird, ist jedoch vorzuziehen. Dabei müssen die relevanten geometrischen Aspekte der zu vereinfachenden Objekte möglichst lange erhalten werden, so daß z.B. der markante Freiraum unter einer Brücke weiter referenzierbar bleibt (vgl. (Butz & Krüger, 1996)).

Abbildung 1.2 illustriert diese Eigenschaft am Beispiel eines Terminalgebäudes des Frankfurter Flughafens. Während hier z.B. jeweils benachbar-

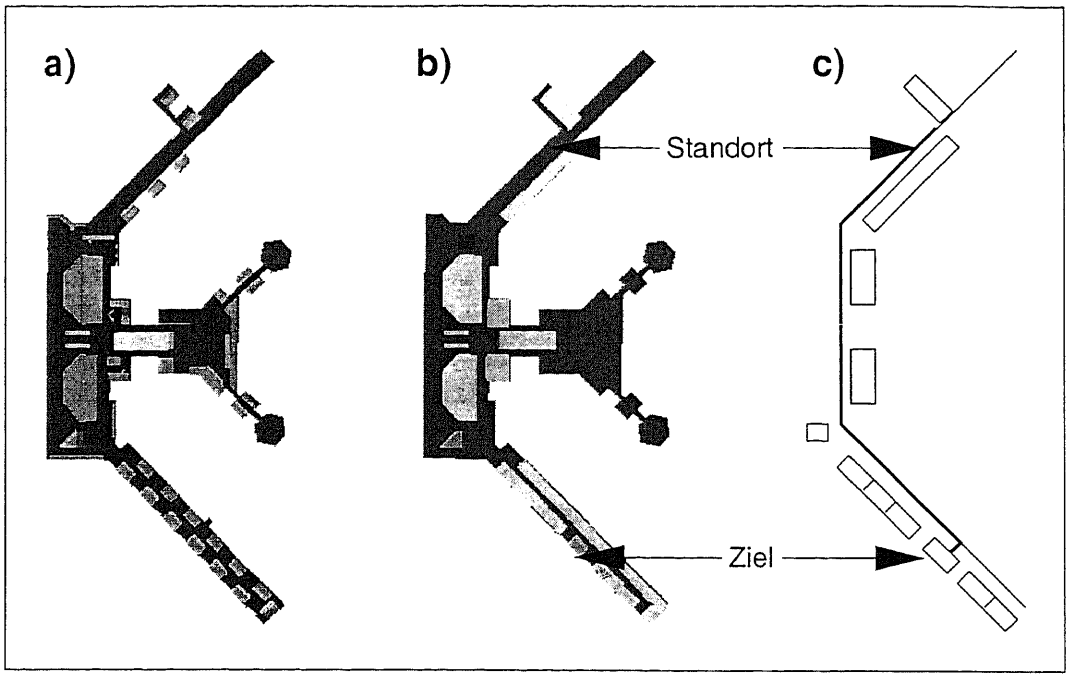

Abbildung 1.2: Drei unterschiedliche graphische Abstraktionen eines Flughafengebäudes

te Gates zu einem Block verschmolzen werden, bleibt dasjenige, an dem ein Fluggast einchecken muß, klar erkennbar.

Dieses Zusammenfassen mehrerer gleichartiger Objekte zu einem einzigen resultiert in einem sogenannten *integrierten Referenzobjekt*, zu dessen *Konstruktion* allerdings konzeptuelles Wissen über die beteiligten Objekte erforderlich ist. Der Vorteil liegt einerseits in der größeren Auffälligkit eines solchen Objektes, andererseits in einer Verminderung der Anzahl der potentiellen Referenzen (dies unter der Voraussetzung, daß das integrierte Objekt nicht zusätzlich sondern alternativ bearbeitet wird). Selbstverständlich können die weiter oben beschriebenen Abstraktionsverfahren auch auf integrierte Referenzobjekte angewendet werden, wie auch in Abb. 1.2 zu sehen ist[3].

Ausgehend von einer vollständigen geometrischen Modellierung (wie in Abb. 1.2a) können durch unterschiedlich komplexe Abstraktionen einerseits graphische Informationen sowohl auf hochauflösenden (Farb-)Bildschirmen (Abb. 1.2a oder b) als auch auf dem monochromen Display eines PDA (Personal Digital Assistent) adäquat visualisiert werden (z.B. Abb. 1.2c), andererseits kann durch Verwendung geeigneter abstrakter Modelle die Rechengeschwindigkeit, etwa bei der Bestimmung räumlicher Relationen, erhöht werden. Dies geht natürlich mit einem Verlust an Genauigkeit auch in der abgeleiteten Information einher.

[3]Entsprechende Verfahren werden auch von Kartographen verwendet (vgl. (Powitz, 1993)).

1.4 Räumliche Relationen: Von 2-Punkt- zu n-Punkt

Dieser Abschnitt beschäftigt sich mit *Berechnungsverfahren* für räumliche Relationen. Dazu werden in einem ersten Punkt einige Grundlagen erläutert, die zum weiteren Verständnis erforderlich sind. Nach der Vorstellung eines semantischen Modells wird die Einführung eines Qualitätsmaßes für räumliche Relationen motiviert, das diese vergleichbar macht. In diesem Zusammenhang kann eine Klassifikation der Relationen vorgenommen werden, an der sich die anschließenden konkreten Berechnungsverfahren orientieren.

Der Ausbau der Fähigkeit von Systemen der Künstlichen Intelligenz zur Raumbeschreibung von rein statischen zu potentiell dynamischen Lagebeziehungen, wie sie beispielsweise in Wegbeschreibungen üblich sind, erfordert die Erweiterung des Modells räumlicher Relationen, das bislang im wesentlichen nur 2-Punkt-Relationen beinhaltete. Dazu ist die hier vorgenommene erstmalige Formalisierung und Modellierung von n-Punkt-Relationen (oder auch Pfadrelationen) ein wichtiger Schritt und zentrales Thema und Ziel dieses Teils der Arbeit.

1.4.1 Grundlegendes

Die Berechnung räumlicher Lagebeziehungen in dieser Arbeit folgt in weiten Teilen einem Ansatz, der von Gapp (1997) im Rahmen des SFB 314 entwickelt wurde. Er stellt ein dreistufiges Modell vor, das – ganz im Sinne von Herskovits (1986) – auf einer klaren Trennung von kontextspezifischem konzeptuellem Wissen einerseits und der expliziten Grundbedeutung räumlicher Relationen andererseits basiert.

Abbildung 1.3 zeigt zentral die Ebene der kernsemantische Definition von räumlichen Relationen, die sich in die Referenzsemantik sowie einen lexikalischen Teil untergliedert, die direkt mit den darunter angeordneten Komponenten zur Verarbeitung geometrischer Information resp. zur Generierung sprachlicher Ausdrücke interagieren. Diese beiden Ebenen werden zur semantische Ebene zusammengefaßt, die ihrerseits mit einer konzeptuellen Stufe kommuniziert, auf der z.B. unterschiedliche Gebrauchsarten räumlicher Relationen und relevante Objekteigenschaften modelliert werden.

Ziel der Generierung von Lokalisationsausdrücken ist eine *möglichst gute* Beschreibung einer bestimmten räumlichen Konstellation, dergestalt, daß ein Hörer sich buchstäblich *ein Bild von den Lagebeziehungen machen* kann. Das setzt voraus, daß bei der Auswahl der für den Lokalisationsausdruck angewandten Relation ein Qualitätsmaß angelegt werden kann, das sie über andere, ebenfalls korrekte Relationen heraushebt.

Abbildung 1.4 zeigt ein zu lokalisierendes Objekt LO, für das zwei Referenzobjekte RO1 und RO2 mit den entsprechenden, durch (hervorgehobene) intrinsische Fronten etablierten Referenzsystemen zur Auswahl stehen. Unter der Annahme, daß RO1 und RO2 abgesehen von ihrer Lokation identisch sind, kann sicherlich dennoch ein Unterschied in der Qualität der Proposi-

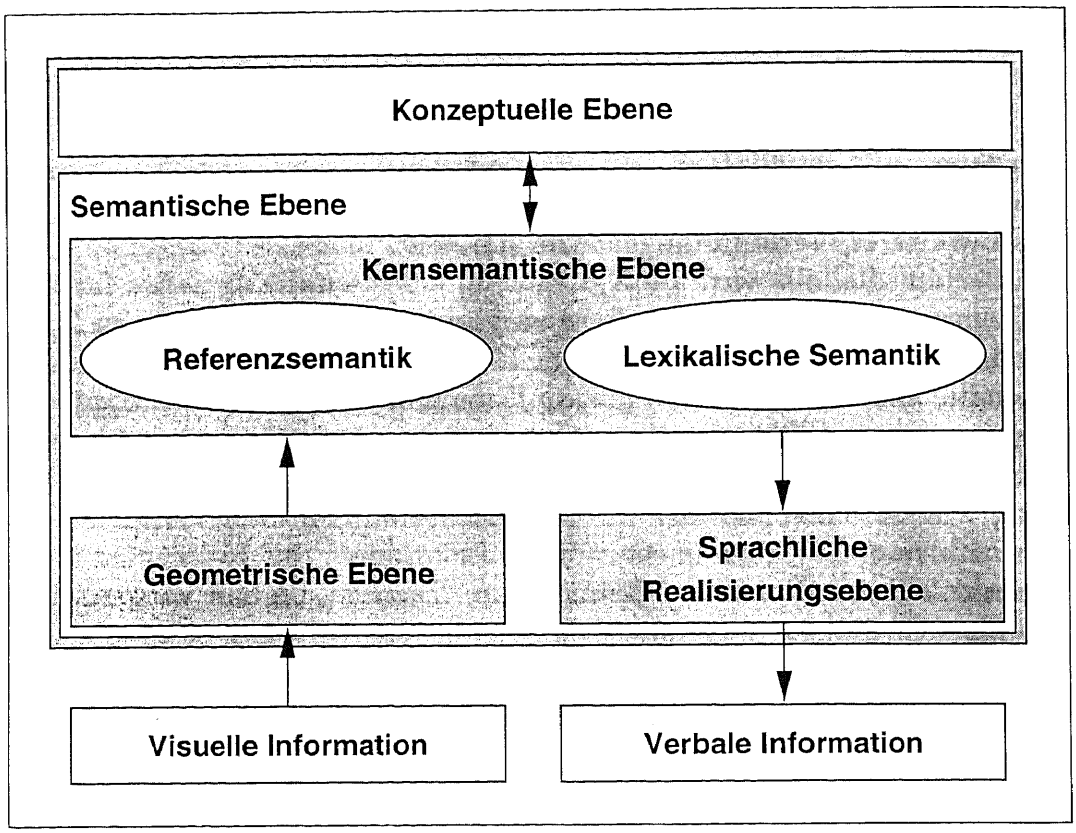

Abbildung 1.3: Modell der Semantik räumlicher Lagebeziehungen nach Gapp

tionen (vor LO RO1 RS1) und (vor LO RO2 RS2) festgestellt werden. Soll nur eine Frage der Art *„Befindet sich LO vor ROx?"* beantwortet werden, so spielt die konkrete Qualität, mit der die Konstellation beschrieben wird, keine große Rolle. Hier ist es offensichtlich ausreichend, zwischen *vor* und dem Gegenteil *hinter* zu diskriminieren. Anders sieht es aus, wenn die Frage *„Wo befindet sich LO?"* lautet: In diesem Fall, ist die Antwort *„Vor RO1"* sicher vorzuziehen. Die bessere *Anwendbarkeit* der Proposition (vor LO RO1 RS1) gegenüber (vor LO RO2 RS2) läßt sich am unterschiedlichen Winkel zwischen der jeweils durch das Referenzsystem induzierten Vorzugsrichtung und der direkten Verbindung von RO1 bzw. RO2 zu LO ablesen. Man spricht in diesem Zusammenhang von der Winkelabweichung als *essentiellem Parameter* der Klasse der *winkelabhängigen Relationen*. Analog dazu sind Relation wie nahe *distanzabhängig*.

Es ist naheliegend, die essentiellen Parameter nicht nur zur Definition räumlicher Relationen sondern auch zu ihrer Bewertung zu verwenden (vgl. (André, Herzog & Rist, 1989)).

Die Unterteilung in winkel- und distanzabhängig ist nicht die einzige Möglichkeit, räumliche Relationen zu klassifizieren. Nimmt man als linguistische Entitäten Ausdrücke wie Präpositionen und Adverbien, die räumliche Beziehungen kodieren, zur Grundlage einer semantischen Analyse, so lassen sich folgende Kriterien extrahieren:

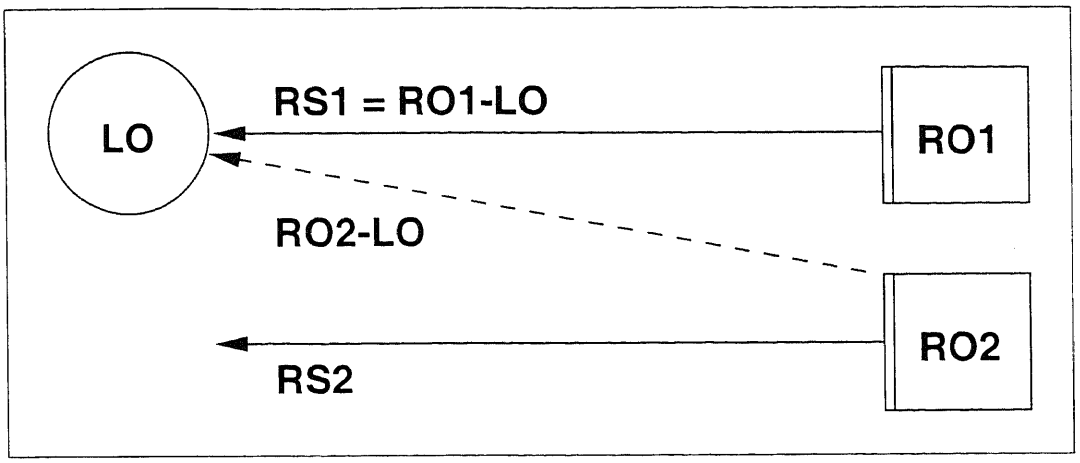

Abbildung 1.4: Bewertung räumlicher Relationen

- *Essentieller Parameter*;
  - *Distanz* z.B. `bei`;
  - *Winkel* z.B. `vor`;
- *Komplexität der Relation*;
  - *Elementar* z.B. `in`;
  - *Zusammengesetzt* aus Relationen mit
    - *identischen essentiellen Parametern* z.B. `links_oben`[4];
    - *unterschiedlichen essentiellen Parametern* z.B. `links_an`; hier sind nach Ansicht des Autors auch komplexere Fälle wie `auf` oder `zwischen` einzuordnen;
- *Komplexität des zu lokalisierenden Objekts*;
  - *2-Punkt-Relationen* kodieren Beziehungen, die mit einer minimalen Repräsentation der beteiligten Objekte als singulärer Punkt auskommen, z.B. `hinter`;
  - *N-Punkt-Relationen* beschreiben Lagen, bei denen das zu lokalisierende Objekt *nicht* allein durch *einen* Punkt beschrieben werden kann, da seine Form von Relevanz ist; dies ist typischerweise bei Wegen, die als Trajektorien repräsentiert werden der Fall; Elemente dieser Klasse werden häufig als Pfadrelation bezeichnet, wie z.B. `entlang`;
- *Lage des Referenzsystems*
  - *Intern* wird die Relation auf einen Innenraum bezogen, z.B. `oben`;
  - *Extern* wird auf einen Raum außerhalb des Bezugsobjektes referiert, z.B. `über`.

[4]Hierbei handelt es um eine Besonderheit der deutschen Sprache: Eine derartige Kombination ist beispielsweise im Englischen nicht gebräuchlich.

1.4.2 Vergleichbarkeit (1): Der Anwendbarkeitsgrad

Nachdem im vorigen Abschnitt die Anwendbarkeit als Qualitätsmaß für räumliche Relationen motiviert wurde, sollen nun notwendige Anforderungen für den sogenannten *Anwendbarkeitsgrad* (AG) definiert werden.

- *Gradierbarkeit*: Abgesehen von einigen wenigen Ausnahmen, wie z.B. `in`, die einen *binärem* Charakter haben, sind räumliche Relationen in der Regel nicht nur entweder *anwendbar* oder *nicht anwendbar*. Vielmehr hat ihre Anwendbarkeit einen stetig wachsenden oder sinkenden Verlauf. Folglich muß auch ein entsprechendes Qualitätsmaß gradierbar sein.

- *Vergleichbarkeit*: Relationen müssen sowohl bei unterschiedlichen Referenzobjekten als auch bei unterschiedlichen essentiellen Parametern auf einfache Weise vergleichbar bleiben. Eine auf bestimmte Teilmengen beschränkte Vergleichbarkeit ist zu vermeiden.

- *Verrechenbarkeit*: Bei zusammengesetzten Relationen wie z.B. `links_vor` sollte es möglich sein, bereits berechnete Ergebnisse (in diesem Fall also für `links` und `vor`) weiterzuverwenden und zu einer Gesamtqualität zu verrechnen.

- *Kognitive Plausibilität*: Die erzielten Ergebnisse bei Qualitätsvergleichen zwischen Relationen müssen experimentellen Befunden standhalten.

Die Verwendung *kubischer Splinefunktionen* zur Kodierung der essentiellen Parameter Distanz und Winkel erfüllt zusammen mit den nachfolgend vorgestellten Berechnungsverfahren diese Kriterien. Dabei wird das Resultat der Verrechnung der Parameter auf das Intervall [0;1] der reellen Zahlen abgebildet. Der Wert Null repräsentiert eine nicht anwendbare, der Wert Eins eine optimal anwendbare räumliche Relation.

1.4.3 2-Punkt-Relationen

Zur Berechnung des Anwendbarkeitsgrades räumlicher 2-Punkt-Relationen wird – wie in Abb. 1.5 gezeigt – ein von der Ausdehnung des Referenzobjekts und der Gebrauchsart der Relation abhängiges *lokales Koordinatensystem* etabliert.

Es skaliert den Raum, in dem Distanz und Winkel zwischen zu lokalisierendem und Referenzobjekt bestimmt werden. Der Anwendbarkeitsgrad wird dann durch eine geeignete Abbildung mittels der erwähnten Splines errechnet. Wie experimentelle Untersuchungen von Gapp hinsichtlich der Bewertung winkelabhängiger Relationen gezeigt haben, gibt es eine lineare Korrelation zwischen Richtungsabweichung und Anwendbarkeitsgrad bei quadratischen Referenzobjekten (vgl. (Gapp, 1995b, 1995a)). Dies spricht für die Sinnhaftigkeit des Verfahrens.

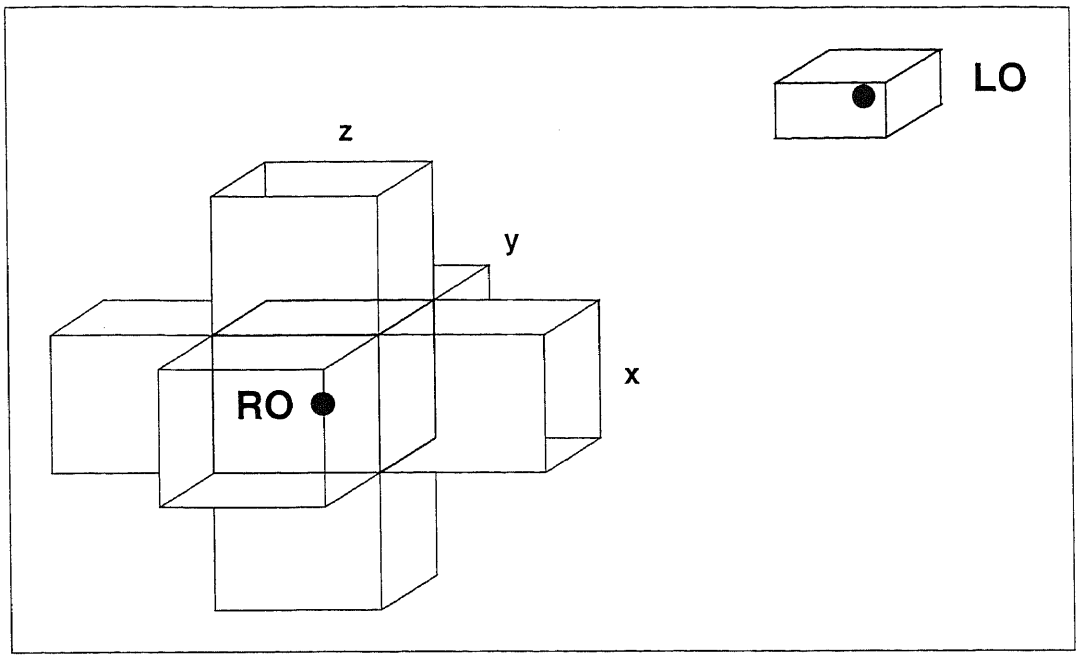

Abbildung 1.5: Lokales Koordinatensystem und Winkelabweichung

Abbildung 1.6 visualisiert die Anwendbarkeitsstrukturen für die räumlichen Relationen `über` bzw. `rechts`. Dabei ist zu beachten, daß in beiden Fällen der Anwendbarkeitsraum durch ein Distanzkonzept zusätzlich eingeschränkt wurde. Zusammengesetzte räumliche Relationen lassen sich durch Verknüpfung der Resultate ihrer Konstituenten berechnen. So führt Gapp z.B. den Anwendbarkeitsgrad für `rechts_hinter` auf die gewichteten Minima der Teil-Relationen zurück.

Näheres zu den Berechnungsverfahren für 2-Punkt-Relationen findet sich bei Gapp (1997).

1.4.4 N-Punkt-Relationen

Eine andere Klasse räumlicher Relationen sind n-Punkt- oder auch Pfadrelationen, die durch sprachliche Ausdrücke wie *entlang*, *vorbei* oder *hin (zu)* ausgedrückt werden und die besonders bei Wegbeschreibungen durch z.B. Navigationsassistenten große Bedeutung haben. Sie wurden bislang weit weniger betrachtet, als die zuvor behandelten 2-Punkt-Relationen (vgl. (Blocher, Essig, Krüger & Maaß, 1999; Kray & Blocher, 1999)). Dies mag an der größeren Komplexität liegen, da n-Punkt-Relationen dann Anwendung finden, wenn ein Pfad mit mindestens zwei Punkten, einem Start- und einem Endpunkt, vorliegt.

Abbildung 1.7 illustriert dies: Trajektorie (a) ist sicher ein besserer Repräsentant einer abstrakten Relation `entlang` (die eine Facette der Semantik der entsprechenden Präposition besitze) als Trajektorie (b). Es gibt aber auf beiden Trajektorien keinen *einzelnen* Punkt, der für die Bestimmung des Anwendbarkeitsgrades dieser Relation herangezogen werden kann.

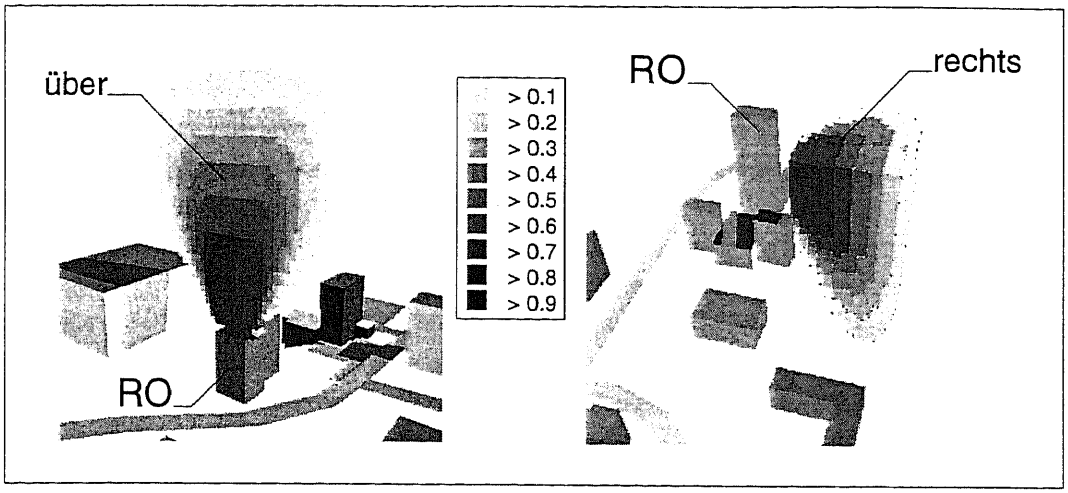

Abbildung 1.6: Ausschnitte der 3D-Anwendbarkeitsstrukturen von `über` und `rechts`

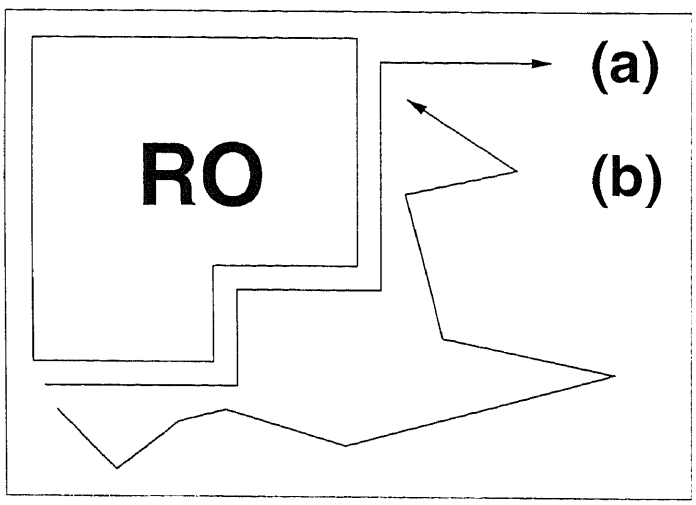

Abbildung 1.7: Zwei Trajektorien

Im folgenden wird ein erster Ansatz für Berechnungsverfahren von Relationen dieser Klasse vorgestellt, der insbesondere den Vorteil besitzt, bestimmte (geometrische) Konzepte mit Gapps Verfahren gemein zu haben. Dadurch ist sowohl die Verwendung von Teilresultaten als auch eine allgemeine Vergleichbarkeit gewährleistet.

1.4.4.1 Pfadbezogene räumliche Ausdrücke im Deutschen

Auch bei Pfadpräpositionen, wie unterschiedliche Typen pfadbezogener räumlicher Ausdrücke von nun an verkürzend genannt werden sollen, liegt ein Hauptproblem in ihrer inhärenten Unbestimmtheit:

(3) *„Der Weg zu dem Park."*

Räumlicher Ausdruck	Bedeutung	Formalisierung
zu	• Annäherung	$D \xrightarrow{D} IMD$
bis	• Annäherung	$D \xrightarrow{*} IMD$
in	• Übergang von außerhalb nach innerhalb	$MD \xrightarrow{M} I$
nach	• Annäherung	$D \xrightarrow{*} IMD$
an	• Annäherung bis Berührung	$D \xrightarrow{D} MD$
gegen	• Annäherung bis Berührung	$D \xrightarrow{M} IM$
von	• Zunahme der Entfernung	$IMD \xrightarrow{*} D$
aus	• Übergang von innerhalb nach ausserhalb	$I \xrightarrow{M} MD$
entlang	• Konstanz der Entfernung, Passieren des Randes	$IMD \xrightarrow{*} IMD$
vorbei	• Annäherung, Zunahme der Entfernung	$D \xrightarrow{D} D$
durch	• Annäherung, Betreten, Verlassen	$IMD \xrightarrow{I} IMD$
um	• Drehbewegung	$MD \xrightarrow{MD} MD$
	• Annäherung, Passieren des Randes, Zunahme der Entfernung	$MD \xrightarrow{MD} MD$

Tabelle 1.1: Einige pfadbezogene räumliche Ausdruck und ihre Kernbedeutung

Satz (3) kann mehrere Bedeutungen haben, da weder bekannt ist, wo der Weg beginnt, noch wo er genau endet: in der Nähe, aber außerhalb des Parks, an seinem Rand oder sogar innerhalb? Mindestens diese Lesarten sind ohne weiteres vertretbar. Auch bei der Verbalisierung räumlicher Lagebeziehungen existieren Mehrdeutigkeiten: so kann über jemanden, der einen Park verläßt, beides gesagt werden:

(4a) *„Er kommt von dem Park."*

(4b) *„Er kommt aus dem Park."*

Auch bei n-Punkt-Relationen muß also die kernsemantische Bedeutung extrahiert werden, bevor Berechnungsverfahren anwendbar sind, deren Ergebnisse dann mit dem zuvor abstrahierten konzeptuellen und kontextuellen Wissen ergänzt werden.

Tabelle 1.1 zeigt einige der gebräuchlichsten deutschen Pfadpräpositionen und ihre Kernbedeutung[5]. Es wurden allerdings nicht sämtliche Bedeutungs-

[5]Kursorische Untersuchungen der entsprechenden räumlichen Ausdrücke im Englischen, Französischen und Japanischen legen die Hypothese nahe, daß eine sprachübergreifende Menge dieser Basiskonzepte existiert. An dieser Stelle wären eingehende experimentelle Studien wünschenswert.

facetten aufgenommen, sondern nur die jeweils wichtigsten. Dabei wurde neben einer rein natürlichsprachlichen auch eine formalere Definition der jeweiligen Kernsemantik erstellt. Diese Formalisierung orientiert sich an einer Teilmenge der von Egenhofer (1991) entwickelten *Semantik topologischer Relationen*: I steht für Egenhofers `inside`, das Enthaltensein, M für `meet`, Berührung, und D bedeutet Kontaktlosigkeit (`disjoint`).

Ein Übergang der Art

$$\mathrm{IM}D \xrightarrow{M} \mathrm{D} \tag{1.1}$$

ist zu lesen als:

> *„Die entsprechende Relation beschreibt einen Pfad, der entweder innerhalb oder am Rand oder außerhalb des Referenzobjektes beginnt. Unterwegs tritt mindestens eine Berührung auf, bevor der Pfad außerhalb des Referenzobjektes endet."*

Je eine Variante dieser Kernbedeutung könnte beispielsweise durch die Sätze (4a) und (4b) wiedergegeben werden.

Fettdruck symbolisiert die hauptsächliche Verwendung, normale Großbuchstaben stehen für eine mögliche, wohingegen Schrägstellung einen ungewöhnlichen, aber nicht falschen Gebrauch signalisiert. Ein Asteriskus (*) über dem Pfeil deutet an, daß während des Übergangs *keine* spezielle Relation erfüllt sein muß.

1.4.4.2 Semantische Grundkonzepte

Die Mehrzahl der in Tab. 1.1 aufgeführten Pfadpräpositionen beschreibt eine Annäherung an ein Objekt (das offensichtlich als erstrebenswertes Ziel angesehen werden kann), während eine Distanzzunahme nur durch *aus* und *von* ausgedrückt wird. Die Existenz eines Konzeptes, das eine Drehung repräsentiert (*um*), zeigt, daß neben der Distanz auch der Winkel eine entscheidende Rolle spielt. Dieselben essentiellen Parameter wie bei 2-Punkt-Relationen beschreiben, diesmal allerdings durch ihre Veränderung, also auch grundlegende Aspekte von Pfadrelationen.

Ferner kann offensichtlich bei einigen Pfadrelationen eine 2-Punkt-Relation *abgespalten* werden: Einer solchen zusammengesetzten Relation entspricht in Satz (5) der pfadbezogene Ausdruck *in*, dessen Kernbedeutung sich hier auf den Endpunkt (und nur auf diesen!) des Weges bezieht.

(5) *„Er geht in dem Park."*

Diese Dekomponierbarkeit kann dazu verwendet werden, aus elementaren Relationen komplexere zu konstruieren, die entsprechend leichter durch experimentelle Untersuchungen zu validieren sind.

Aus den aus Tab. 1.1 abgeleiteten Beobachtungen läßt sich eine Semantik für Pfadrelationen aufbauen, bei der aus fünf elementaren Bausteinen komplexere Relationen zusammengesetzt werden können, indem sie entweder miteinander oder mit 2-Punkt-Relationen kombiniert werden:

- *Distanz-Zunahme*: Der Endpunkt einer Trajektorie befindet sich näher am Referenzobjekt als der Startpunkt.
- *Distanz-Abnahme*: Der Startpunkt einer Trajektorie befindet sich näher am Referenzobjekt als der Endpunkt.
- *Distanz-Konstanz*: Die Distanz aller Punkte einer Trajektorie zum Referenzobjekt ist gleich.
- *Winkel-Änderung*: Die Endpunkte einer Trajektorie bilden mit den Referenzobjekt einen Winkel.
- *Winkel-Konstanz*: Die Endpunkte einer Trajektorie bilden mit den Referenzobjekt keinen Winkel.

(Zumindest im Deutschen (sowie auf englisch und französisch) scheint eine Winkelabweichung – im Gegensatz zu einer Distanzveränderung – eher neutral oder ungerichtet zu sein.)

1.4.4.3 Geometrische Pfadrelationen

Wendet man sich der geometrischen Seite der Berechnung pfadbezogener räumlicher Relationen zu, so ist es wegen der erhöhten Komplexität dieses Relationentyps angebracht, das angestrebte Ziel noch einmal klar herauszustellen:

> Gegeben sei ein pfadförmiges (oder pfadförmig idealisiertes) zu lokalisierendes Objekt. Gesucht wird die n-Punkt-Relation, die die Lagebeziehung dieses Objektes zu einem beliebigen Referenzobjekt am besten beschreibt, sowie der entsprechende Anwendbarkeitsgrad.

Definition 1.1 (Trajektorie) *N Punkte $p_1 ... p_i ... p_N$ definieren eine* Trajektorie. *Die Endpunkte p_1 und p_N bezeichnen Start und Ende der Trajektorie und definieren sich entweder über eine explizit gegebene Richtung oder den Berechnungvorgang selbst.*

Für die Berechnung pfadbezogener räumlicher Relation werden folgende Annahmen getroffen:

1. *Das zu lokalisierende Objekt wird als Trajektorie repräsentiert.* Dies bezieht sich auf die Tatsache, daß eine Pfadrelation nur sehr schwer auf ein nicht pfadförmiges Objekt angewendet werden kann, wie Satz (6) zeigt.

   (6) * *„Der Grenzstein entlang der Mauer."*

   (Die Repräsentation selbst obliegt der geometrischen Ebene unter Berücksichtigung konzeptueller Eigenheiten.)

2. *Die Trajektorie wird als Ganzes beschrieben.* Häufig erscheint es zweckmäßig, eine Trajektorie in einzelne Abschnitt zu zerlegen, die, jeweils mit Pfadrelationen assoziiert, eine aussagekräftigere Beschreibung ermöglichen. Da ein solches Vorgehen aber eine Analyse der gesamten Trajektorie voraussetzt (um relevante Teilstücke zu identifizieren), ist es sinnvoll, bei diesem Arbeitsgang auch gleich eine Relation für die *ganze* Trajektorie zu generieren. (Da sich das im folgenden vorgeschlagene Verfahren rekursiv auf erkannte Teilstücke anwenden läßt, wird auf die Dekomposition nicht weiter eingegangen. Sie ist Teil der konzeptuellen Ebene.)

3. *Es wird der* Verlauf *einer Trajektorie beschrieben.* Dies kann nicht durch 2-Punkt-Relationen erfolgen. Somit müssen n-Punkt-Relationen verwendet werden.

4. *Die Berechnungsverfahren werden in das Modell der Semantik räumlicher Relationen integriert und erweitern dieses.* Folglich müssen die Anwendbarkeitsgrade vergleichbar sein. Dies wird erreicht durch die Verwendung derselben essentiellen Parameter – Distanz und Winkel – und von Teilresultaten aus der Generierung von 2-Punkt-Relationen, sowie der identischen Berechnung von Referenzsystem und Referenzpunkten (den einander nächsten Punkte von LO und RO).

5. *Pfadrelationen orientieren sich an Pfadpräpositionen.* Da das geometrische Modell über die Referenzsemantik mit linguistischen Konzepten verknüpft ist, sollen n-Punkt-Relationen kernsemantische Aspekte der entsprechenden pfadbezogenen räumlichen Ausrücke reflektieren.

Die Umsetzung dieser Punkte wird nun erläutert.

1.4.4.4 2-Punkt-Trajektorien

Die Analyse von Trajektorien erfordert einige Definitionen:

Definition 1.2 (2-Punkt-Trajektorie) *Eine Trajektorie, die aus genau zwei unterschiedlichen Punkten besteht, heißt* 2-Punkt-Trajektorie.

Definition 1.3 (n-Punkt-Trajektorie) *Eine Trajektorie, die aus mehr als zwei unterschiedlichen Punkten besteht, heißt* n-Punkt-Trajektorie. *Sie kann auf einfache Weise aus $n-1$ 2-Punkt-Trajektorien zusammengesetzt werden. Umgekehrt können beliebig komplexe n-Punkt-Trajektorien in eine eineindeutige Folge von 2-Punkt-Trajektorien aufgesplittet werden.*

Der letzte Punkt legt nahe, bei der Analyse von Pfadrelationen zuerst die Lagebeziehungen zwischen 2-Punkt-Trajektorien und einem Referenzobjekt zu untersuchen; in einem zweiten Schritt kann dann versucht werden, die gewonnenen Erkenntnisse auf n-Punkt-Trajektorien zu übertragen.

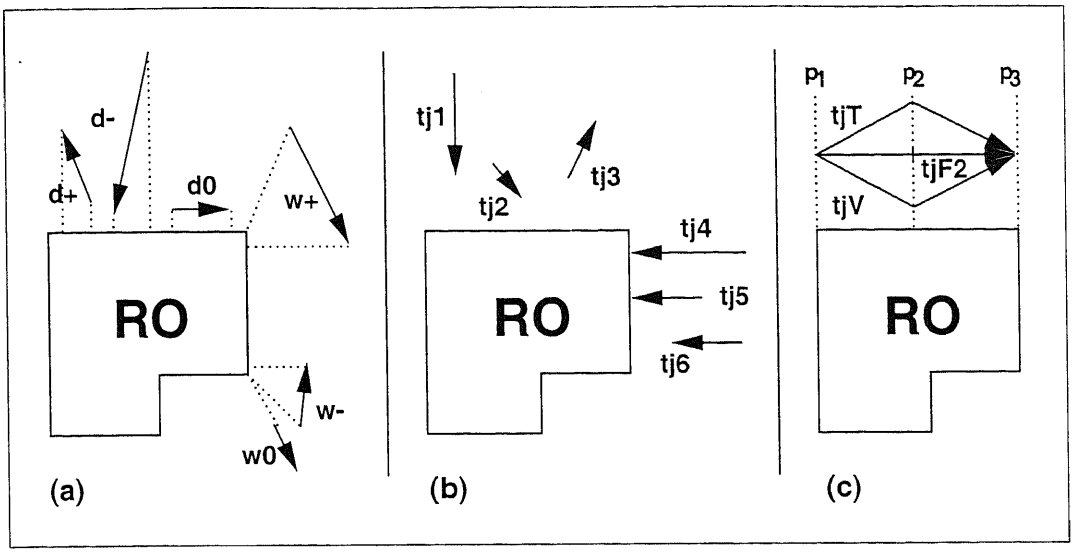

Abbildung 1.8: Trajektorien: (a) Änderungen (b) Qualitäten (c) Krümmungen

Da mathematisch gesehen die Lage jeder Trajektorie über Vektoren eindeutig lokalisiert werden kann, und diese durch Distanz und Winkel ausdrückbar sind, wird Punkt 4 aus dem vorangegangenen und den Beobachtungen aus Abschnitt 1.4.4.2 entsprochen.

Um, wie in Punkt 3 angesprochenen, den Verlauf einer Trajektorie durch Distanz und Winkel zu beschreiben, müssen die jeweiligen Änderungen dieser essentiellen Parameter von Trajektorienpunkt p_1 zu Punkt p_2 betrachtet werden: Distanz und/oder Winkel können entweder zunehmen, gleichbleiben oder abnehmen wie in Abb. 1.8a dargestellt[6].

Diese Unterscheidung ähnelt den weiter oben gemachten Beobachtungen und erlaubt die Definition *elementarer n-Punkt-Relationen* in Tab. 1.2[7].

Veränderung	**Distanz**		**Winkel**	
Zunahme	`weg_von`	$d+$	`rechtsrum_um`	$w+$
Abnahme	`hin_zu`	$d-$	`linksrum_um`	$w-$
Keine	`längs`	$d0$	`nicht_drehend`	$w0$

Tabelle 1.2: Elementare n-Punkt-Relationen

Abbildung 1.8b zeigt, daß Trajektorien danach sortiert werden können, wie gut sie eine elementare Pfadrelation repräsentieren: $tj1$ trifft die Kernbedeutung von `hin_zu` besser als $tj2$. Abgesehen davon können auch *unterschiedliche* Relationen verglichen werden: Die Qualität der Relation `weg_von` wie sie durch $tj3$ repräsentiert wird, liegt zwischen denen von $tj1$ und $tj2$. Die

[6]Die Veränderung beim Winkel bezieht sich dabei immer auf eine Grundrichtung der Referenzsystems.

[7]Bei der Auflistung wurden *„sprechende"* Bezeichner für die abstrakten Relationen gewählt; die Suffixe +, -,0 stehen bei `drehend*` für *im Uhrzeigersinn*, *gegen den Uhrzeigersinn* bzw. *keine Drehung*.

Trajektorien $tj4$ bis $tj6$ sind alle optimale Repräsentanten der elementaren Relation `hin_zu`, die in Tab. 1.2 *nur* über den Verlauf definiert wurde. Zur Differenzierung von $tj4$ bis $tj6$ müssen zusätzlich zum Verlauf noch andere Faktoren herangezogen werden, wie z.B. die allgemeine Distanz. Da diese, wie in Abschnitt 1.4.3 beschrieben, auch mittels 2-Punkt-Relationen ausgedrückt werden kann, bietet sich hier die Kombination der verschiedenen Relationenklassen an.

Aus dem oben Gesagten folgt, daß der Anwendbarkeitsgrad der Relationen `hin_zu` als die durch die Länge der Trajektorie dividierte Differenz der Distanzen (Δd) der Punkte p_1 und p_2 relativ zum Referenzobjekt ausgedrückt werden kann. Analoges gilt für `weg_von` und `längs`. Bei winkelabhängigen Pfadrelationen errechnet sich der Anwendbarkeitsgrad aus dem Quotienten der Winkeldistanzen ($\Delta\angle$) und 2π. Werden die verwendeten Distanzen und Winkel durch dieselben Verfahren wie bei 2-Punkt-Trajektorien berechnet, so ist eine Vergleichbarkeit der resultierenden Anwendbarkeitsgrade gegeben. Tabelle 1.3 illustriert die Berechnung der Maßzahlen für die Anwendbarkeitsgrade für elementare Pfadrelationen.

Relation	**Maßzahl** ($M_{essParam}$)	**Anwendbarkeitsgrad**
`weg_von`	$M_{dist} = \frac{\Delta d_{RO}(p_2,p_1)}{\lvert\overrightarrow{p_1p_2}\rvert}$	M_{dist}
`hin_zu`		$-M_{dist}$
`längs`		$1 - \lvert M_{dist}\rvert$
`rechtsrum_um`	$M_{wink} = \frac{\Delta\angle_{RO}(p_2,p_1)}{2\pi}$	M_{wink}
`linksrum_um`		$-M_{wink}$
`nicht_drehend`		$1 - \lvert M_{wink}\rvert$

Tabelle 1.3: Berechnung der Anwendbarkeitsgrade elementarer n-Punkt-Relationen bei 2-Punkt-Trajektorien

1.4.4.5 N-Punkt-Trajektorien

In der Realität bestehen Trajektorien, die Objekte wie z.B. Wege repräsentieren, aus mehr als zwei Punkten. Deshalb müssen die vorgestellten Berechnungsverfahren auf n-Punkt-Trajektorien ausgedehnt werden. Dies erfolgt am besten durch eine Gewichtung der Maßzahlen mit der jeweiligen Teilstücklänge, so daß auch eine inkrementelle Verarbeitung garantiert werden kann.

Abbildung 1.8c zeigt jedoch, daß bei n-Punkt-Trajektorien Unterschiede im Verlauf auftreten können, die nicht durch elementare n-Punkt-Relationen beschrieben werden können. In diesen Fällen muß die Differenzierung über die Krümmung der Trajektorie erfolgen. Hierbei lassen sich drei Typen abstrakter Relationen unterscheiden:

- tjP nähert sich zuerst dem Referenzobjekt und entfernt sich dann: Eine derartige Krümmung korrespondiert mit der Relation `vorbei`.

- tjT entfernt sich zuerst vom Referenzobjekt und nähert sich dann: Diese Krümmung spiegelt sich in der künstlichen Relation `trip` wider. Für sie scheint es keinen entsprechenden räumlichen Ausdruck im Deutschen zu geben, was der unter Punkt 5 genannten Anforderung zuwider läuft. Da jedoch die beiden anderen, über die Krümmung definierbaren Relationen sinnvoll sind, wurde `trip` der Vollständigkeit halber mit aufgenommen[8].

- tjF behält die Entfernung über den gesamten Verlauf bei, eine Krümmung tritt nicht auf. Dies korrespondiert mit einer Relation `längs2`[9].

Eine Krümmung kann auf zwei Arten verarbeitet werden: Einerseits kann versucht werden, eine geeignete Dekomponierung zu finden, was Punkt 2 widerspricht, dem zufolge die Trajektorie als Ganzes analysiert werden soll. Andererseits können aber die essentiellen Parameter eingehender untersucht werden. Dann zeigt sich, daß eine Krümmung, die im einfachsten Fall bei einer Trajektorien mit drei Punkten auftritt, durch die Differenz der $\Delta d_{RO}(p_2, p_1)$ und $\Delta d_{RO}(p_3, p_2)$ beschreibbar ist. Diese Differenz ist entweder positiv (`trip`), negativ (`vorbei`) oder Null (`längs2`).

Eine entsprechende Maßzahl, anhand derer die einzelnen Anwendbarkeitsgrade dieser Klasse krümmungsabhängiger Pfadrelationen berechnet werden, ist wie folgt definiert:

$$M_{krum} = sign(\Delta d_{RO}(p_2, p_1) - \Delta d_{RO}(p_3, p_2)) \left(1 - \frac{|\overrightarrow{p_1 p_3}|}{|\overrightarrow{p_1 p_2}| + |\overrightarrow{p_2 p_3}|}\right). \tag{1.2}$$

Auf Trajektorien mit mehr als drei Punkten kann diese Maßzahl wie zuvor mittels Gewichtung, hier über die inneren Punkte, ausgedehnt werden.

1.4.4.6 Resultate

Tabelle 1.4 faßt die semantischen Aspekte pfadbezogener räumlicher Ausdrücke, wie sie in Abschnitt 1.4.4.2 (Tab. 1.1) entwickelt wurden, mit den in den letzten Abschnitten definierten abstrakten Relationen zusammen. Diese können miteinander oder mit 2-Punkt-Relationen kombiniert werden, wobei sich letztere nur auf den jeweils explizit angegebenen Punkt einer Trajektorie beziehen.

Auch wenn Tab. 1.4 den Eindruck erwecken mag, so existiert *keine 1:1-Übereinstimmung* zwischen einer gegebenen Pfadpräposition und einer bestimmten Pfadrelation, da eine einzelne Pfadpräposition mehrere unterschiedliche Situationen beschreiben und umgekehrt dieselbe Situation eventuell durch unterschiedliche Pfadpräpositionen beschrieben werden kann. Es besteht in diesem Fall also eine n:m-Beziehung zwischen Sprache und durch

[8] In diesem Zusammenhang ist festzuhalten, daß abstrakte Relationen öfter in bestimmten natürlichen Sprachen kein Äquivalent haben, so sind z.B. *links* und *rechts* als intrinsische Konzepte bei den Aborigenes unbekannt.

[9] Ob und gegebenenfalls inwieweit `längs` und `längs2` unterschiedliche Konzepte repräsentieren, ist noch nicht geklärt.

Räumlicher Ausdruck	Formalisierung	Pfadrelation (elementar oder kombiniert)
zu	$\mathrm{D} \xrightarrow{D} \mathit{IM}\mathrm{D}$	hin_zu
bis	$\mathrm{D} \xrightarrow{*} \mathit{I}\mathrm{MD}$	hin_zu
in	$\mathit{M}\mathrm{D} \xrightarrow{M} \mathrm{I}$	hin_zu $\wedge$ in(p_N)
nach	$\mathrm{D} \xrightarrow{*} \mathrm{IMD}$	hin_zu $\wedge$ in(p_N)
an	$\mathrm{D} \xrightarrow{D} \mathrm{MD}$	hin_zu $\wedge$ kontakt(p_N)
gegen	$\mathrm{D} \xrightarrow{M} \mathit{I}\mathrm{M}$	hin_zu $\wedge$ kontakt(p_N)
von	$\mathrm{IMD} \xrightarrow{*} \mathrm{D}$	weg_von
aus	$\mathrm{I} \xrightarrow{M} \mathit{M}\mathrm{D}$	in(p_1) $\wedge$ weg_von
entlang	$\mathrm{IMD} \xrightarrow{*} \mathrm{IMD}$	längs
vorbei	$\mathrm{D} \xrightarrow{D} \mathrm{D}$	vorbei
durch	$\mathit{I}\mathrm{MD} \xrightarrow{I} \mathit{IM}\mathrm{D}$	vorbei $\wedge$ $\exists i$: in(p_i)
um	$\mathrm{MD} \xrightarrow{MD} \mathrm{MD}$	rechtsrum_um $\vee$ linksrum_um
	$\mathrm{MD} \xrightarrow{MD} \mathit{M}\mathrm{D}$	(rechtsrum_um $\vee$ linksrum_um) $\wedge$ vorbei

Tabelle 1.4: Pfadpräpositionen und die korrespondierenden Pfadrelationen

eine Geometrie repräsentiertem Raum. Die dadurch aufkommende Vagheit der Sprache kann durch die Einbeziehung kontextuellen Wissens vermindert oder beseitigt werden.

Tabelle 1.5 listet zu einigen der in Abbildungen gezeigten Trajektorien exemplarische Ergebnisse für die Berechnung von Anwendbarkeitsgraden nach den vorgestellten Methoden auf. Wie zu erwarten, ist die Anwendbarkeit von Trajektorie (a) in Abb. 1.7 signifikant höher als die von Trajektorie (b). Auch die Beispiele aus Abb. 1.9 erzielen intuitiv korrekte Resultate.

Abb.	Trajektorie	Pfadrelation	AG	Räumliche Ausdruck
1.7	(a)	längs	0.76	entlang
	(b)	längs	0.59	entlang
1.9	tjA	hin_zu	0.68	nach, zu
	tjE	weg_von	0.68	von
	tjV	vorbei	0.57	vorbei
	tjU	rechtsrum_um	0.63	um
	tjD	durch	0.71	durch

Tabelle 1.5: Anwendbarkeitsgrade der Beispieltrajektorien

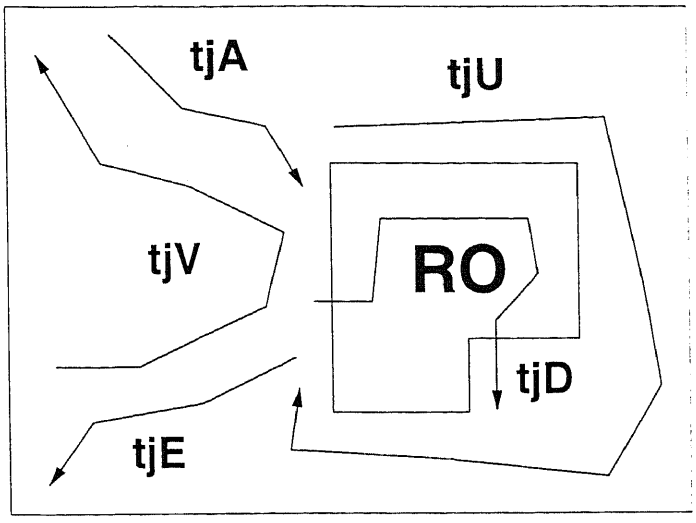

Abbildung 1.9: Beispielhafte Trajektorien

1.5 Ein ressourcensensitives Präzisionskonzept

Bei der Verarbeitung natürlicher Sprache tritt häufig das Problem der Relativierung von Aussagen auf. Dabei kann es zu Präzisierungen (Satz 7) oder Einschränkungen (Satz 8) des Bedeutungsinhalts oder seiner Gültigkeit kommen:

(7) *„Er ist sehr groß."*

(8) *„Er ist ziemlich groß."*

Die Modifikatoren, die dies in der Sprache leisten, in den obigen Beispielen die Partikeln *sehr* und *ziemlich*, werden als *linguistische Hecken* bezeichnet. Sie enthalten sozusagen eine Zusatzinformation und verbessern somit in Sinne von Grice (1975) das kommunikative Verständnis: sei es indem die Mitteilung exakter oder bestimmter wird oder auch wenn eine Vagheit oder eine Unsicherheit ausgedrückt wird, die der Hörer mit in seine Interpretation aufnehmen kann. Demnach spielt die sprachliche Unbestimmtheit – *Vagheit* – eine, wie Wittgenstein (1953) sagt, wichtige Rolle bei Analyse und Generierung von Äußerungen, obwohl dies der Grice'schen Maxime *„Sei klar!"* zuwider läuft.

Unbestimmtheit in natürlicher Sprache kann sich auf unterschiedliche Weise manifestieren, wie die folgende Klassifikation nach Pinkal (1985) zeigt.

1. *Kontextabhängigkeit* ist ein Unbestimmtheitsphänomen, das selten wahrgenommen (und/oder verbalisiert) wird.

   (a) *Semantische Kontextabhängigkeit*: Die Bedeutung von Worten hängt z.T. von der Situation ab, in der sie geäußert werden.

   (b) *Pragmatische Kontextabhängigkeit*: Die Bedeutung von Worten hängt z.T. von der Intention ab, in der sie geäußert werden, z.B. als Drohung.

2. *Wahrnehmbare Unbestimmtheit* drückt sich in Worten aus und kann auf zwei Arten untergliedert werden:

   (a) *Grad der Unbestimmtheit*:

      i. *relativ*: Beliebige Präzisierungen sind möglich (*groß*).

      ii. *randunscharf*: Ausdrücke mit einem ausgedehnten, definiten Anwendbarkeitsbereich und anschließender Übergangszone (Farben).

      iii. *punktuell*: Scheinbar präzise Ausdrücke der Alltagssprache mit kleinem Anwendbarkeitsbereich und kleiner Übergangszone: *„Der Fisch ist zwei Kilo schwer"*. Eine wesentliche Abweichung ist unüblich.

   (b) *Epistemische Unbestimmtheit* oder auch Unsicherheit, Ungewißheit:

      i. *Pragmatische Unbestimmtheit*: Hier mangelt es an Information, z.B. *„München besteht nicht nur aus ein paar Häusern"*.

      ii. *Semantische Unbestimmtheit*: Hier kann kein eindeutiger Wahrheitswert zugewiesen werden (*„Sie ist eine schöne Frau"*).

         A. *Vagheit* läßt unendlich viele Lesarten zu.

            - *Genuine Vagheit* kann in natürlicher Sprache nicht beliebig präzisiert werden (etwa die Farbe *blau*).

         B. *Mehrdeutigkeit* läßt nur endlich viele Lesarten zu.

            - *Ambiguität* bedeutet zwei sich gegenseitig ausschließende alternative Lesarten (*Zug*).
              - *Polysemie*: ein Ausdruck (*Bank*) verbalisiert zwei unterschiedliche Konzepte.
              - *Homonymie*: gleichlautende, aber unterschiedlichen Wortklassen angehörende Ausdrücke verbalisieren unterschiedliche Konzepte (z.B. *sieben* als Zahl bzw. Verb).

      iii. *Verwendungsvielfalt* ist eine Subklasse von Vagheit und Mehrdeutigkeit und schließt die Fälle ein, in denen eine Zuordnung zu Vagheit oder Mehrdeutigkeit nicht gelingt. Diese Gruppe kann aber gegenüber genuiner Vagheit bzw. Ambiguität abgegrenzt werden.

Wie vorstehend beschrieben, ist es in vielen Fällen möglich, sprachliche Unbestimmtheiten zu präzisieren und damit dem in (Grice, 1975) formulierten Kooperationsprinzip für das im allgemeinen von Gesprächsteilnehmern erwartete Dialogverhalten näherzukommen, indem eine Verringerung unterschiedlicher Lesarten herbeigeführt wird[10].

Im Rahmen einer ressourcenadaptierenden Raumbeschreibung sollte ein Dialogsystem der KI in der Lage sein, beide Phänomene zu interpretieren und gegebenenfalls bei der Sprachproduktion und -analyse beachten. In der vorliegenden Arbeit wird ein Ansatz zur Interpretation linguistischer Hecken vorgestellt, der auf einer allgemeinen Formalisierung des Konzeptes *Präzision* für gradierte Aussagen, wie z.B. Anwendbarkeitsgraden, beruht und qualitativ bessere Äußerungen erzeugt. Ferner kann eine Analyse von Aussagen

[10] Eine der Maxime von Grice lautet dementsprechend *„Vermeide Mehrdeutigkeiten!"*

auch hinsichtlich ihrer Präzision zu einer genaueren Repräsentation der intendierten Vorstellung führen. Eine Integration in das Gapp'sche Modell der Semantik räumlicher Ausdrücke dient als exemplarische Anwendung[11].

1.5.1 Linguistische Hecken

Räumliche Relationen sind Konzepte, die nach Kolde (1986) entweder der Gruppe der *inhärent vagen oder unscharfen Konzepte*, wie z.B. `entfernt`, `nahe` oder `vor`, oder aber der Klasse der *einfachen binären Konzepte* wie `in` zugeordnet werden. Letztere können in Gegensatz zu den *präzisen binären Konzepten* – etwa mathematische Aussagen – durchaus modifiziert werden, wie das Beispiel `fast_in` belegt.

Diese Modifikation erfolgt durch die sogenannten linguistischen Hecken, die sich einer simplen Einordnung nach grammatikalischen oder an der Wortart orientierten Kriterien entziehen, da sie sowohl als einzelne Lexeme (*sehr*), Nebensätze (*„wie ich meine"*), Suffixe (*gelb-lich*), Betonung oder sogar non- bzw. paraverbal (Augenzwinkern, Zögern) auftreten ((vgl. (Bolinger, 1972; Kolde, 1986)).

Die folgende Definition erlaubt eine erste Einschränkung dessen, was für die vorliegende Arbeit sinnvoll als linguistische Hecke betrachtet werden soll:

„Als linguistische Hecke (kurz: Hecke, engl.: linguistic hedge) bezeichnen wir sprachliche Einheiten, die Prädikationen nach Grad oder Hinsicht ihres Zutreffens modifizieren und als Operatoren interpretiert werden können, welche die Vagheit des sprachlichen Konzepts, auf das sie angewendet werden, verstärken oder abschwächen."

(Wahlster, 1977)

Hiermit wird die Betonung auf die Gradierbarkeit der durch die Heckenausdrücke zu modifizierbaren sprachlichen Konzepte gelegt.

Linguistische Hecken lassen sich auf unterschiedliche Weise klassifizieren. Die beiden wichtigsten Klassifikation werden im folgenden vorgestellt.

1.5.1.1 Funktional-sprachliche Klassifikation

Die *funktional-sprachliche Klassifikation* gliedert Hecken nach ihren Auswirkungen auf die Zugehörigkeitsfunktion einer *Fuzzy-Menge* in eine Einengung *Konzentration* bzw. Ausweitung der *Zugehörigkeitsbereiches Dilation* und die sogenannte *Kontrastintensifikation* (vgl. Abb. 1.10)[12].

[11] Beide Teilbereiche wurden in Zusammenarbeit mit Christian Kray entwickelt und von ihm im Rahmen einer Diplomarbeit implementiert (vgl. (Kray, 1998)).

[12] Eine Fuzzy-Menge nach Zadeh (1965, 1975, 1993) eignet sich sehr gut zur Beschreibung *unscharfen Wissens*, da die *Zugehörigkeit* zu einer solchen als graduelles Konzept aufgefaßt wird. Die *Zugehörigkeitsfunktion* eines Objektes kodiert den Grad des Zutreffens der die Fuzzy-Menge beschreibenden Eigenschaft.

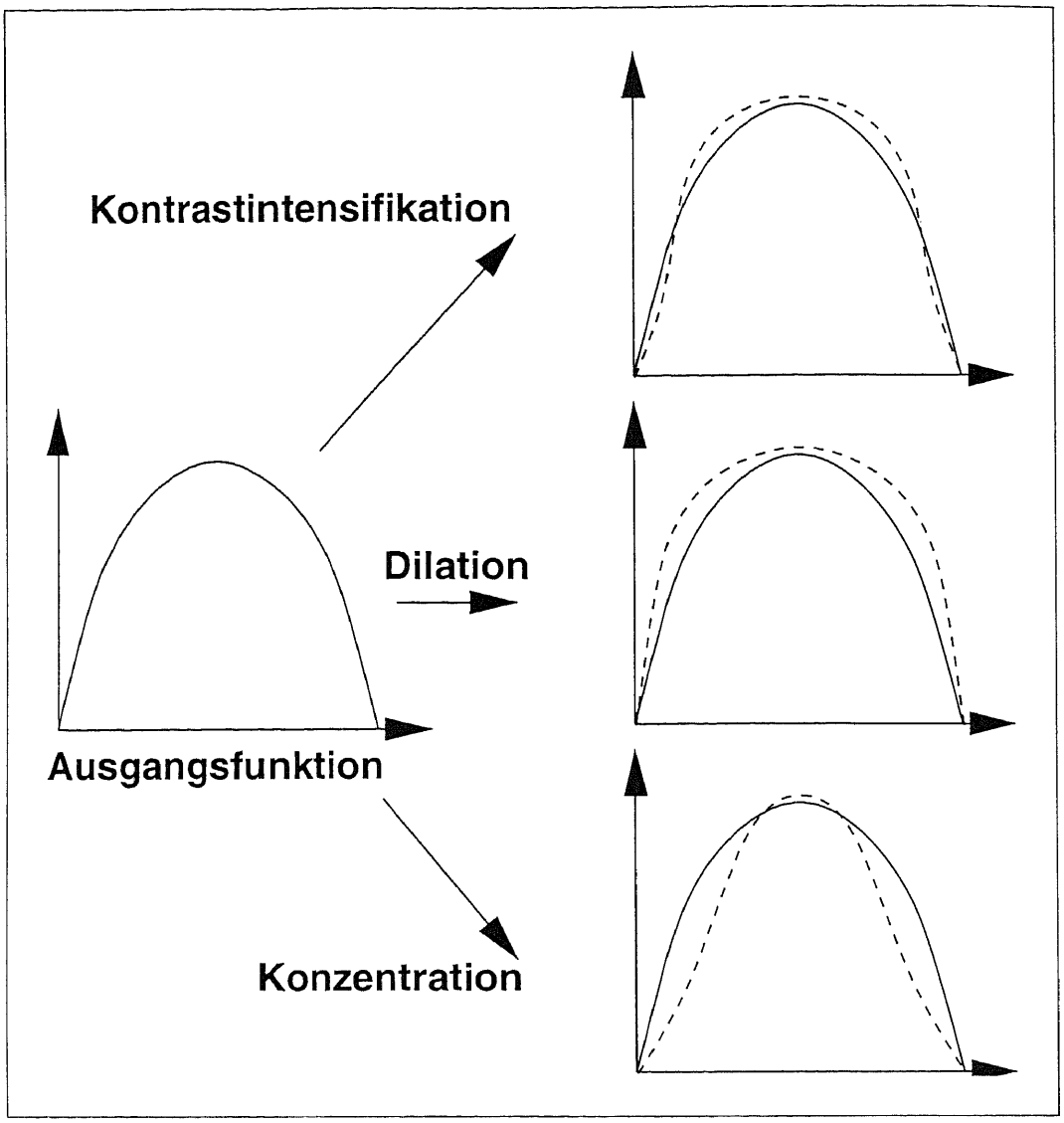

Abbildung 1.10: Modifikationen einer Mengenzugehörigkeitsfunktion

1.5.1.2 Linguistische Klassifikation

Eine vollständige *linguistische Klassifikation* von Heckenausdrücken würde den Rahmen dieser Arbeit sprengen. Deshalb soll an dieser Stelle lediglich eine Eingrenzung auf für das Thema der Raumbeschreibung relevante linguistische Hecken erfolgen. Abbildung 1.11 (nach Kray (1998)) zeigt eine Hierarchie, die trotz ihrer Komplexität keinen Anspruch auf Vollständigkeit erhebt (vgl. (Kolde, 1986; Helbig, 1990; Helbig & Helbig, 1990)).

Bei natürlichsprachlichen Lokalisationsangaben treten insbesondere zwei Gruppen von (verbalen) Modifikatoren auf: *Modalworte* und *Partikeln*. Da erstere allerdings den Grad der *Verläßlichkeit* einer Aussage beschreiben, sind sie weniger für Gradierung räumlicher Ausdrücke im Sinne einer Modifikation deren Anwendbarkeit geeignet als die Partikeln. Auch die Wirkung von Abtönungspartikeln ist eher kommunikativer Natur (*bloß*, *vielleicht*).

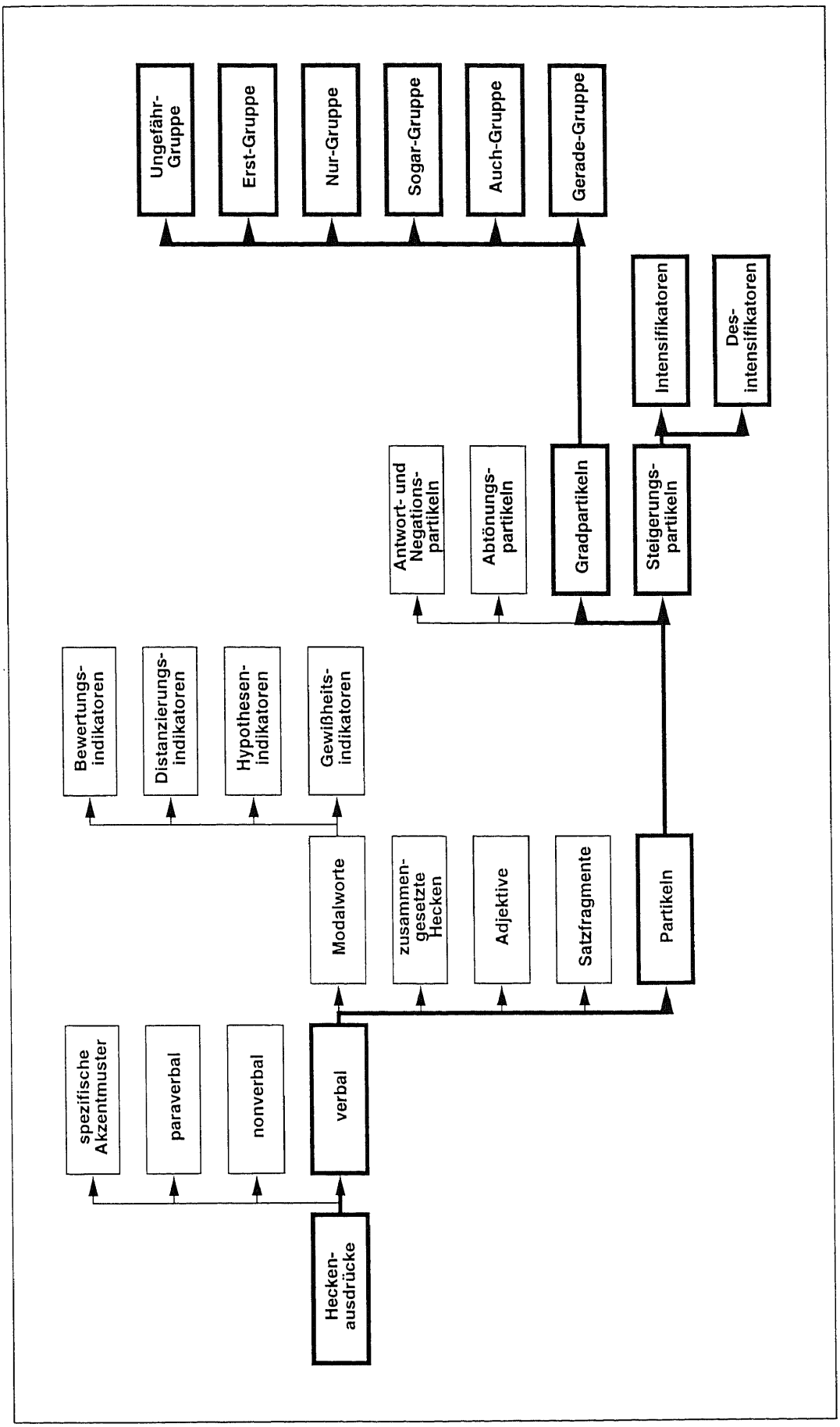

Abbildung 1.11: Linguistische Klassifikation von Heckenausdrücken

Damit können – durchaus bewußt vereinfachend – die in dieser Arbeit betrachteten linguistischen Hecken auf *Gradpartikeln* und *Steigerungspartikeln* engegrenzt werden[13]. Elemente dieser Gruppen haben eine semantische Wirkung und verändern so die Anwendbarkeit räumlicher Relationen ohne ihre Bedeutung an sich oder gar den Wahrheitsgehalt der gesamten Aussage zu tangieren.

Gradpartikeln (oder auch Gradierungspartikeln) lassen sich – wie in der Abbildung zu sehen – nach der Art ihrer Quantifizierung unterscheiden, während die Steigerungspartikeln in *Intensifikatoren* und *Desintensifikatoren* zerfallen.

Auffällig ist, daß beide Klassifikationsansätze in drei Modifikationen münden: Verstärkung, Abschwächung und Akzentuierung. Angesichts der Vielfalt der linguistischen Hecken liegt es deshalb nahe, dieses Faktum als Grundlage einer Definition von *abstrakten* Heckenausdrücken zu nehmen, die von einer konzeptuellen Komponente (wie im Gapp'schen Semantikmodell für räumliche Relationen) in jeweils adäquate sprachliche Entitäten überführt werden. Bevor dies jedoch näher erläutert wird, folgt ein kurzer Überblick über bisherige Ansätze zur Modellierung von linguistischen Hecken.

1.5.1.3 Bisherige Ansätze zur Modellierung

Der Modellierung linguistischer Hecken wurde bisher eher geringe Beachtung geschenkt. Alle Ansätze basieren mehr oder weniger direkt auf Anwendungen der Fuzzy-Mengentheorie und behandeln gradierte Konzepte. Sie selbst gehören ebenfalls in diese Kategorie.

Modifikator	**Berechnung**	**Beispiele**
`Konzentration`	$CON(x) = x^2$	sehr äußerst
`Dilation`	$DIL(x) = x^{1/2}$	ungefähr etwa
`Kontrastintensifikation`	$INT(x) = \begin{cases} 2x^2 & ; x \leq 0{,}5 \\ 1-2(1-x)^2 & ; x > 0{,}5 \end{cases}$	genau exakt

Tabelle 1.6: Berechnung von Modifikatoren linguistischer Hecken

Die ersten, grundlegenden Arbeiten stammen von Zadeh (1972) und Lakoff (1973), die ein Modell vorschlagen, das linguistische Hecken als Modifikatoren über unscharfen Mengen interpretiert. Diese Modifikatoren sind selbst wieder aus elementaren Modifikationsoperatoren wie den zuvor genannten `Konzentration`, `Dilation` und `Kontrastintensifikation` zusammengesetzt (vgl. Tab. 1.6 nach (Zimmermann, 1987)):

[13]Negations- und Antwortpartikeln werden hier nicht näher untersucht.

„The point [...] is that a hedge, h, may be interpreted as an operator, with operand u, which transforms a fuzzy subset M(u) of U into the subset M(hu). To characterize this operator, it is convenient to define several primitive operations on fuzzy sets from which more complicated operators such as hedges may be built up by composition."

(Zadeh, 1972)

Dies kann zur Konstruktion auch von komplexen Hecken mit einfachen Mitteln genutzt werden. Kritisch ist bei diesem und fast allen anderen Ansätzen zu bemerken, daß die konkreten Bewertungen in der Regel introspektiv ermittelt wurden und demzufolge einer empirischen Validierung harren.

Aufbauend auf Zadeh und Lakoff entwickelte Hanßmann (1980) im Rahmen des Projektes SWYSS (Say What You See System) (vgl. (Hußmann & Schefe, 1984)) eine Komponente, die Fragen zu zweidimensionalen Szenen beantwortet. Dieser Ansatz ist der einzige der hier genannten, der sich mit der Anwendung von linguistischen Hecken auf räumliche Relationen befaßt. Hanßmann assoziiert dabei direkt bestimmte Anwendbarkeitsintervalle mit – insgesamt neun verschiedenen – Heckenausdrücken zusammen mit den Wort *„anwendbar"*:

(9) *„„X links von Y" ist nahezu anwendbar."*

Hier ist insbesondere die Annahme zu kritisieren, daß eine feste Zuordnung von bestimmten Anwendbarkeitsgrade zu bestimmten linguistischen Hecken existiert. Ferner sind die Grenzen, wann welchem Anwendbarkeitsgrad welche Hecke zugeordnet wird, introspektiv gewählt. Zudem scheint es wahrscheinlich, daß sie – falls sie existieren – fließend sind (vgl. (Cleeren, Vandenberghe, Gyseghem & Caluwe, 1993)).

Eine der wenigen neueren Arbeiten zur Modellierung linguistischer Hekken ist der Ansatz von Bouchon-Meunier (1992). Auch hier werden Eigenschaften von Fuzzy-Mengen wie zuvor beschrieben modifiziert. Hinzu kommt die Möglichkeit, das semantische Zentrum des mit einer Relation assoziierten Konzeptes zu verschieben.

Ein letzter Ansatz, der hier kurz beleuchtet werden soll, ist die *Prämodifikation* nach Cleeren et al. (1993): Ausgehend von einer experimentellen Studie zu gradierten Adjektiven wurde eine abschnittsweise definierte, lineare Prämodifikationsfunktion bestimmt, mittels derer Fuzzy-Mengenzugehörigkeitsfunktionen so verändert werden können, daß sie die Untersuchungsergebnisse gut approximieren. Dieses Verfahren erlaubt zwar weitgehende Modifikationen – wenn auch nur von Gradpartikeln – und gestattet fließende Übergänge von einer Hecke zur anderen, doch fehlen Operationen zur Verformung (Stichwort: Kontrastintensifikation).

Insgesamt kann festgehalten werden, daß allen Ansätzen das Manko gemein ist, eine direkte Beziehung zwischen einem (oder mehreren) bestimmten Modifikationsoperator(en) und einem Heckenausdruck herzustellen. Somit kann ein Einfluß des Kontextes, in dem eine Raumbeschreibung stattfindet, nur sehr ungenügend berücksichtigt werden – selbst wenn z.B. das semantische Zentrum einbezogen wird.

1.5.1.4 Integration in das Gapp'sche Semantikmodell

Das Phänomen einer Quasi-1:1-Beziehung von linguistischer Hecke und Modifikator der bisherigen Ansätze zur Modellierung von Heckenausdrücken führt – wie oben dargelegt – direkt zu einer fast vollständigen Vernachlässigung des Kontextes. Da hier also ein ähnlich gelagertes Problem wie bei der Modellierung räumlicher Relationen auftritt, bietet sich auch dieselbe Lösung an: Eine klare Trennung zwischen der sprachlichen Ausprägung einer linguistischen Hecke und ihrer durch den Kontext induzierten Auswirkung.

Damit kann auch hier ein dreistufiges Modell der Semantik erstellt werden, daß mit dem von (Gapp, 1997) vorgestellten Ansatz (vgl. Abschnitt 1.4.1) korrespondiert: Auf einer sprachlichen Realisierungsebene, setzt eine kernsemantischen Ebene auf. Im Gegensatz zu ersterer ist letztere einzelsprachunabhängig. Beide zusammen beschreiben vollständig die Semantik eines Heckenausdrucks und interagieren mit der konzeptuellen Ebene, die u.a. das bisherige abstrakte Ergebnis entsprechend dem aktuellen Kontext konkretisiert.

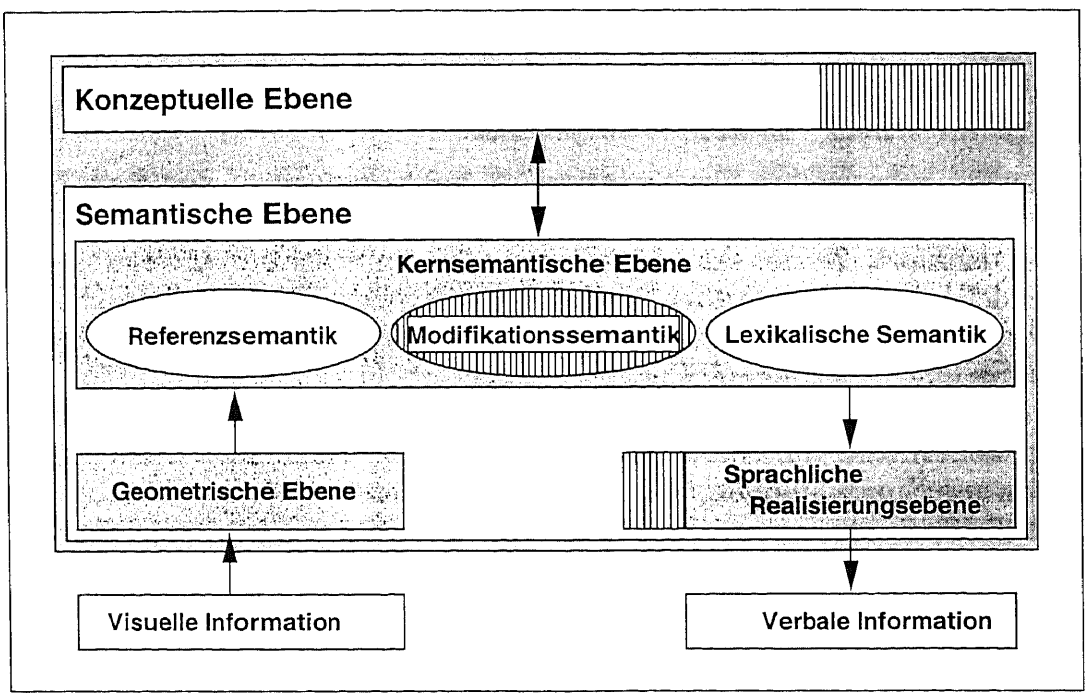

Abbildung 1.12: Erweitertes Modell der Semantik räumlicher Lagebeziehungen

Wie Abb. 1.12 zeigt, läßt sich dieser Ansatz problemlos in das Gapp'sche Semantikmodell einfügen. Dazu muß zum einen die konzeptuelle Ebene um Konzepte erweitert werden, die für die Realisierung von Hecken relevant sind. Insbesondere muß aber die kernsemantische Ebene um eine Komponente zur Modifikation räumlicher Relationen ergänzt werden. Zur Realisierung dieser *Modifikationssemantik* bieten sich die von Zadeh (1972) vorgeschlagenen primitiven Operatoren und ihre Kombination an (vgl. Tab. 1.6). Die konkreten Berechnungsverfahren folgen in Abschnitt 4.2.3.15. Schließlich ist

auch ein adäquater Ausbau der sprachlichen Realisierungsebene erforderlich. (Die genannten Erweiterungen sind in Abb. 1.12 schraffiert dargestellt.)

Die Vorteile dieses Ansatzes sind dieselben wie bei Gapps Basismodell: Modularität, die neue experimentelle Erkenntnisse einfach integrierbar macht, und die prinzipielle Sprachunabhängigkeit außerhalb der jeweiligen sprachlichen Realisierungsebene. Durch die Möglichkeit, in einem einzigen Ansatz räumliche Relationen und linguistische Hecken, zu verarbeiten, kann eine neue Qualität der Modellierung räumlicher Lagebeziehungen erreicht werden.

Die Modifikation der Anwendung räumlicher Relationen durch Heckenausdrücke bedingt aber eine Vergleichbarkeit der jeweiligen Qualitäten: Nicht immer ist ein *verheckter* Ausdruck der bessere. Ferner neigen einige Modifikatoren (z.B. die Quadrierung) dazu, den Anwendbarkeitsgrad einer präzisieren – also im Sinne von Grice besseren – Beschreibung stark zu vermindern. Beide Probleme können durch den im folgenden Abschnitt vorgestellten *Präzisionsgrad* behoben werden.

1.5.2 Vergleichbarkeit (2): Der Präzisionsgrad

Für den Bereich der sprachlichen Kommunikation allgemein und der Raumbeschreibungen insbesondere können durch präzisere Äußerungen Mehrdeutigkeiten und Fehlinterpretationen verringert werden. Umgekehrt steigt mit jeder Verwechslungsvermeidung die Präzision einer Aussage, was insbesondere den ohnehin problematischen Raumbeschreibungen zugute kommt (vgl. (Herrmann & Grabowski, 1994)).

Die Enzyklopädie Brockhaus (1997) definiert *Präzision* als ein Maß der Übereinstimmung von angestrebtem und erzielten Resultat. Damit sind sowohl die Präzision, da sie als Maßzahl unterschiedliche Werte annehmen kann, als auch die (gemessene) Übereinstimmung, auf die sie angewendet wird, graduelle Konzepte. Da dies auch auf die in Abschnitt 1.4 vorgestellten und mittels des Anwendbarkeitsgrades vergleichbaren räumlichen Relationen zutrifft, können diese grundsätzlich auch präzisiert werden, d.h. Relationen mit gleichem Anwendbarkeitsgrad werden anhand ihrer Präzision unterscheidbar. Zusätzlich zur Erfüllung der Anforderungen, denen auch der Anwendbarkeitsgrad genügt, wie Gradierbarkeit, Vergleichbarkeit, Verrechenbarkeit und kognitiver Plausibilität (vgl. Abschnitt 1.4.2), muß ein *Präzisionsgrad* (PG) auf ebendieser Anwendbarkeit einer gradierten Relation basieren.

Die Verfahren, die im folgenden zur Berechnung eines Präzisionsgrades entworfen werden, sind nicht nur auf räumliche Relationen, sondern wie oben beschrieben auf alle graduellen Konzepte, inclusive der Präzision selbst, anwendbar[14].

[14]Diese rekursive Eigenschaft ähnelt der Hintereinandereihung von linguistischen Hecken: *„Er ist sehr, sehr alt"*.

1.5.3 Globale Präzisionsmaße

1.5.3.1 Der globale ungewichtete Flächen-Präzisionsgrad

Ein erster Faktor, der zur Berechnung eines Präzisionsgrades herangezogen werden kann, ist die Fläche unter einer Anwendbarkeitsfunktion. Dies entspricht der Beobachtung, daß eine Relation umso präziser ist, je seltener sie (optimal) anwendbar wird.

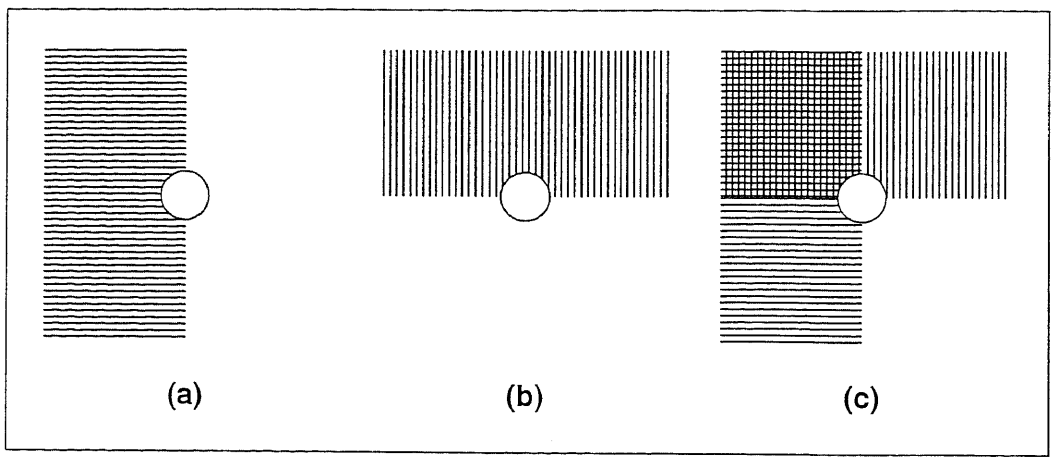

Abbildung 1.13: Präzisierung durch Kombination

In Abbildung 1.13 sind als simples Beispiel binäre Anwendbarkeitsräume für die Relationen `links` (a) bzw. `oben` (b) zu sehen, es wird dabei also nur zwischen *anwendbar* und *nicht anwendbar* differenziert. Die Relation `links_oben` ist als Konjunktion definiert und wird durch die Überlappung beider Räume repräsentiert. Dabei ist die resultierende Fläche in Darstellung (c) (also der eventuelle Suchraum) entsprechend kleiner als bei den einzelnen Komponenten. Falls sich in diesem Bereich ein zu lokalisierendes Objekt befindet, so wäre eine Anwendung der kombinierten Relation in einer Raumbeschreibung präziser.

Da der Bildbereich für Anwendbarkeitsgrade auf das Intervall [0;1] der reellen Zahlen abgebildet wird und unter der Voraussetzung einer adäquaten, relativ gleichen Skalierung des Wertebereichs bezüglich des (der) jeweiligen essentiellen Parameter(s), entspricht dies einem Vergleich der Flächen unter den Kurven. Demnach ist in Abb. 1.14 die Verwendung der Relation `b` an der Stelle x_b gegenüber `a` an der Stelle x_a vorzuziehen, obwohl beide denselben Anwendbarkeitsgrad haben[15].

Um eine Vergleichbarkeit dieses *globalen ungewichteten Flächen-Präzisionsgrades* auch bei Anwendbarkeitsfunktionen mit unterschiedlichen essentiellen Parametern (und damit unterschiedlichen Dimensionen) zu gewährleisten, muß eine Normierung erfolgen. Der Normierungsfaktor nf ist insbesondere von der Menge der betrachteten Anwendbarkeitsfunktionen abhängig, sowie vom gewünschten Bildbereich. Ferner muß berücksichtigt werden, daß

[15] Es kann, muß aber nicht $x_a = x_b$ gelten.

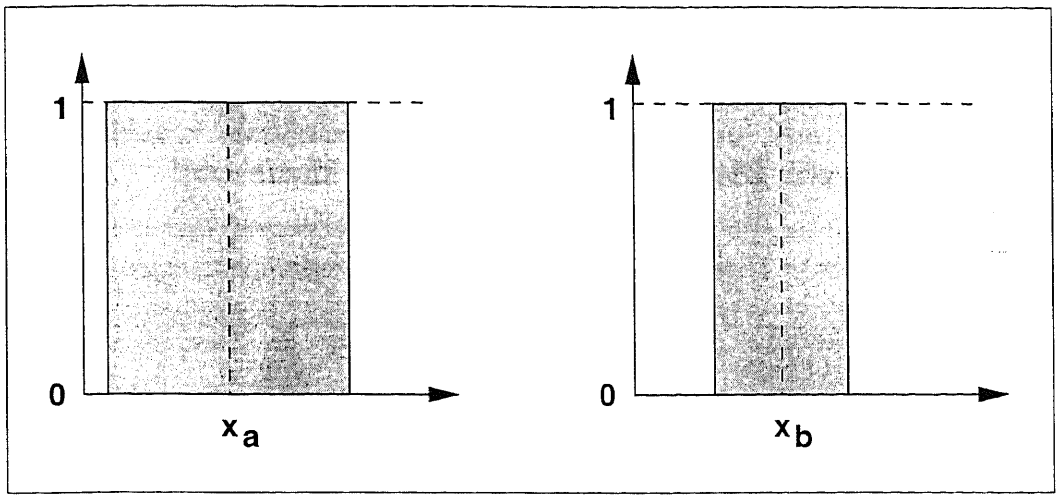

Abbildung 1.14: Präzisionsgrad: Kriterium der globalen ungewichteten Fläche

der Flächeninhalt unter den Anwendbarkeitskurven eventuell unendlich wird (z.B. bei einer Relation `fern`). In der Praxis hat es sich deshalb als vorteilhaft herausgestellt, die jeweiligen Flächen nur innerhalb eines definierten Intervalls $[x_{min}, x_{max}]$, des sogenannten *Relevanzbereiches* (rb) zu berücksichtigen (vgl. Abb. 1.15).

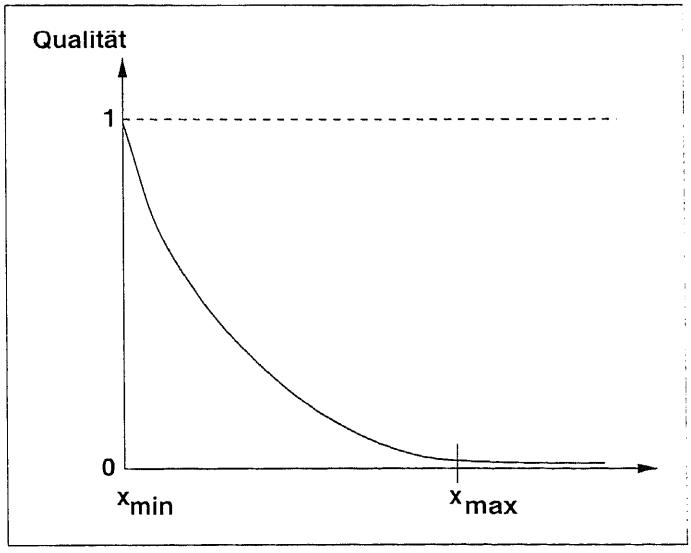

Abbildung 1.15: Präzisionsgrad: Normierung der Gesamtfläche

Damit berechnet sich der globale ungewichtete Flächen-Präzisionsgrad pg_{guF} als:

$$pg_{guF}(x) = nf \cdot \int_{x_{min}}^{x_{max}} f(x)dx. \tag{1.3}$$

1.5.3.2 Der globale gewichtete Flächen-Präzisionsgrad

Eine Bewertung durch den globalen gewichteten Flächen-Präzisionsgrad reicht aber nicht immer aus, um intuitiv eklatante Unterschiede hinsichtlich der Präzision zu erfassen, wie das Beispiel in Abb. 1.16 illustriert:

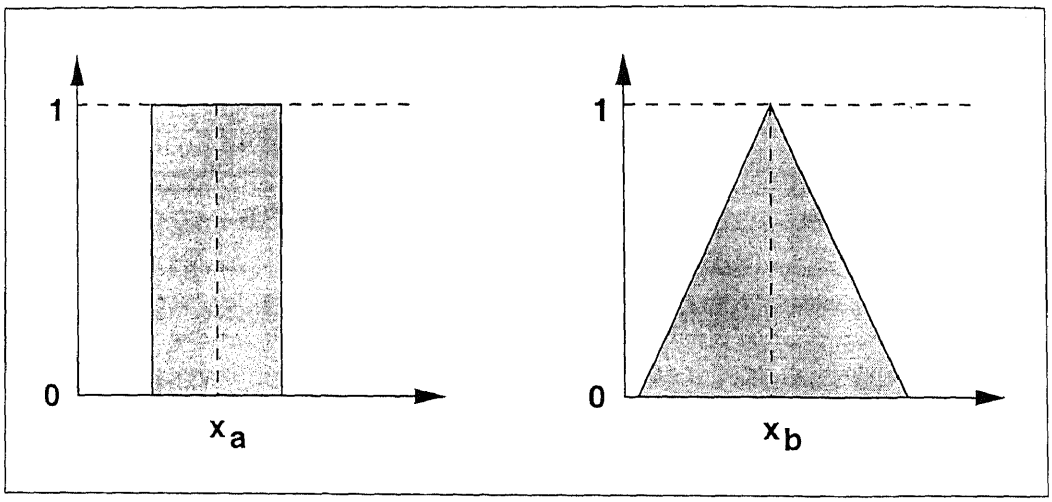

Abbildung 1.16: Präzisionsgrad: Kriterium der globalen gewichteten Fläche

Die Präzision, die man mit der Relation a verbindet, ist wesentlich geringer als die von b, bei der nur eine einzige Position optimal ist und für einen potentiellen Hörer keine Verwechslungsgefahr besteht. Die Fläche unter den Kurven ist zwar gleichgroß, doch ihre *Verteilung* bezüglich der Höhe des Anwendbarkeitsgrades ist unterschiedlich. Verfolgt man diesen Gedanken weiter, so bietet sich zur Bestimmung eines verbesserten Präzisionsgrades eine Gewichtung des jeweiligen Flächenanteils mit dem zugehörigen Anwendbarkeitsgrad und man erhält den *globalen gewichteten Flächen-Präzisionsgrad* pg_{ggF}[16]:

$$pg_{ggF}(x) = nf \cdot \frac{x_{max} - x_{min}}{n} \cdot \int_{x_{min}}^{x_{max}} f(x)^2 dx. \tag{1.4}$$

Auch für diesen Präzisionsgrad muß ein Normierungsfaktor definiert werden.

1.5.4 Lokale Präzisionsmaße

Eine globale Bewertung einer Anwendbarkeitsfunktion reicht jedoch nicht aus, wie Abb. 1.17 zeigt:

Hier sind neben dem Anwendbarkeitsgrad an den Stellen x_a und x_b auch Flächengröße und -lage jeweils identisch. Dadurch kann mit den beiden bisher entwickelten Präzisionsgraden keine Unterscheidung getroffen werden. Dennoch scheint b präziser zu sein, da die Gefahr eines gravierender Irrtums

[16]Man beachte dabei , daß der Wertebereich des Anwendbarkeitsgrades das Intervall [0;1] ist.

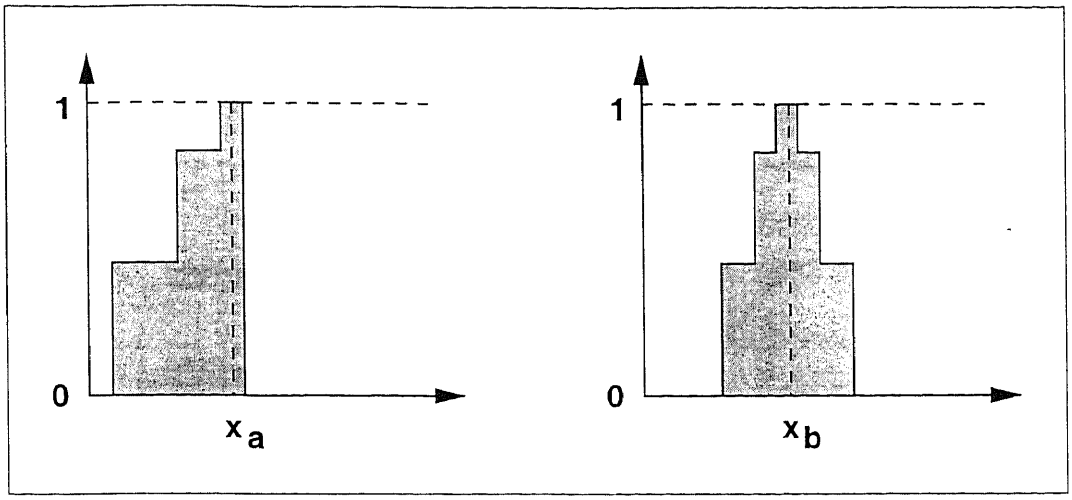

Abbildung 1.17: Funktionen mit gleicher gewichteter Fläche

geringer ist als bei Relation a, die auf einer Seite direkt von *optimal anwendbar* zu *überhaupt nicht anwendbar* übergeht. Eine eventuelle Fehlinterpretation durch den Hörer könnte schwerwiegendere Auswirkungen haben.

Dieses Beispiel legt nahe, lokale Aspekte zu berücksichtigen und deshalb statt der Gesamtfläche nur einen relevanten Flächenabschnitt zu betrachten. Dazu bedarf es allerdings Kriterien für die Grenzen eines solchen Abschnitts.

Die Fuzzy-Mengentheorie unterscheidet anhand eines Funktionswertes von 0,5 darüber, ob ein bestimmtes Objekt als einer Menge *„eher zugehörig"* oder *„eher nicht zugehörig"* angesehen wird. Die entsprechenden Punkte werden im folgenden als *Neutralpunkte* (np) bezeichnet und definieren zusammen mit Maxima und Randpunkten sogenannte *Konzentrationsbereichen* (kb) innerhalb der Anwendbarkeitsfunktionen. Ein Beispiel hierfür zeigt Abb. 1.18.

1.5.4.1 Lokale Flächen-Präzisionsgrade

Wie vorher auf globaler Ebene lassen sich nun zwei Flächen-Präzisionsmaße bestimmen: der *lokale ungewichtete Flächen-Präzisionsgrad* pg_{luF} und der *lokale gewichtete Flächen-Präzisionsgrad* pg_{lgF}:

$$pg_{luF}(x) = AG(x) \cdot \frac{Fläche(rb)}{Fläche(kb(x))} \cdot AG(max(kb)) \tag{1.5}$$

$$pg_{lgF}(x) = AG(x) \cdot \frac{GewFläche(rb)}{GewFläche(kb(x))} \cdot AG(max(kb)). \tag{1.6}$$

Der Relevanzbereich dient hier als Normierungsfaktor, während der maximale Anwendbarkeitsgrad des Konzentrationsbereichs eine mögliche Überbewertung verhindert.

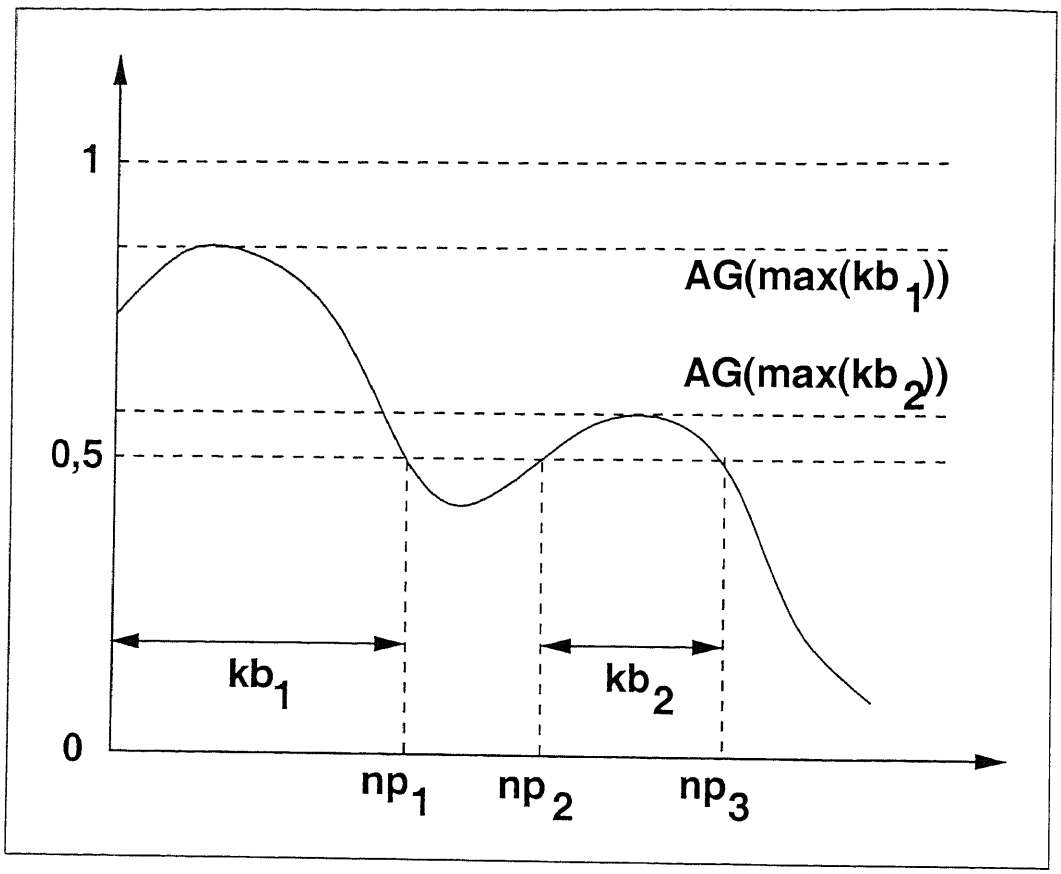

Abbildung 1.18: Neutralpunkte und Konzentrationsbereiche

1.5.4.2 Lokaler Intervall-Präzisionsgrad

Bei den bisherigen lokalen Berechnungsverfahren wird die Position innerhalb eines Konzentrationsbereichs nur zur Bestimmung des Bereichs selber und des Anwendbarkeitsgrades genutzt. D.h. gleiche Anwendbarkeitsgrade führen innerhalb desselben Bereiches zu identischen Präzisionsgraden. Eine Möglichkeit, die *relative Position* (rp) innerhalb eines Konzentrationsbereichs in die Berechnung zu integrieren, besteht in der Bestimmung ihres Abstands zum jeweiligen lokalen Maximum (von dem es *per definitionem* – mindestens – eines gibt). Dazu verwendet man ein *Konzentrationsintervall* (ki), das die relative Position einschließt und vom lokalen Maximum bis zum nächsten Ende des betreffenden Konzentrations*bereichs* reicht. Je näher die Position an einem lokalen Maximum liegt, umso größer ist die Präzision. Als weiterer Faktor geht neben den Anwendbarkeitsgraden der relevanten Position und des lokalen Maximums auch die relative Größe des Konzentrationsintervalls in diesen Präzisionsgraden ein:

$$pg_{II}(x) = AG(x) \cdot \frac{rp}{ki} \cdot \frac{rb}{ki} \cdot AG(max(kb)). \tag{1.7}$$

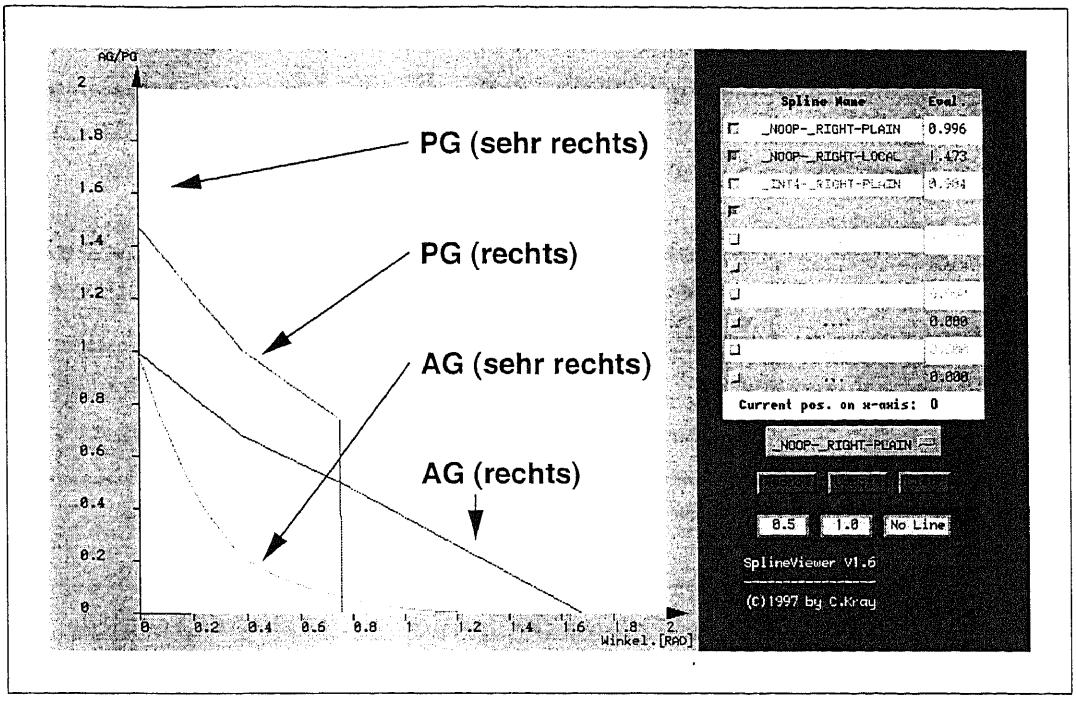

Abbildung 1.19: Anwendbarkeits- und Präzisionsgrad

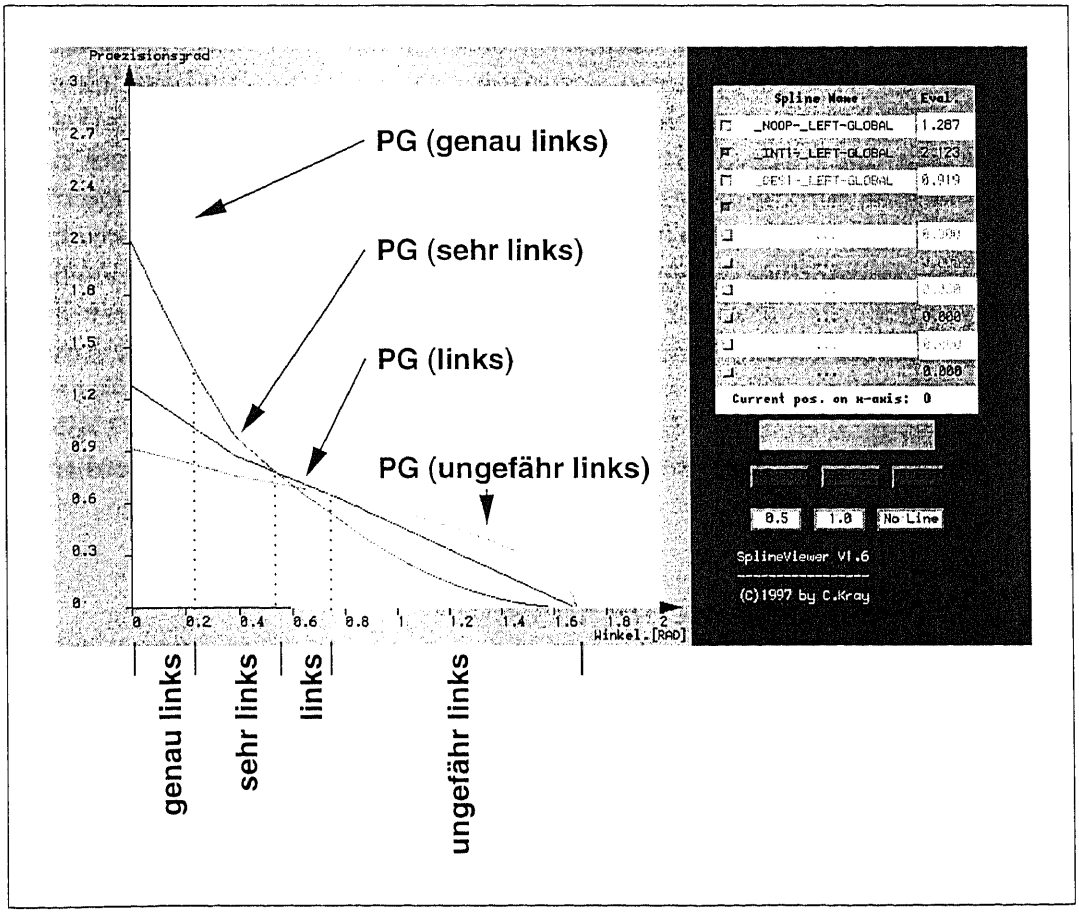

Abbildung 1.20: Globaler ungewichteter Flächen-Präzisionsgrad

1.5.5 Vergleich der Präzisionsmaße

Die vorgestellten Berechnungsverfahren für Präzisionsgrade unterscheiden sich insbesondere hinsichtlich des erforderlichen Rechenaufwandes und bezüglich der Qualität (im Sinne von Verläßlichkeit) ihres Ergebnisses.

Faktoren	**Globale ungew. Fläche**	**Globale gew. Fläche**	**Lokale ungew. Fläche**	**Lokale gew. Fläche**	**Lokales Intervall**
Relevanzbereich	⊗	⊗	⊗	⊗	⊗
Konz.-Bereich			⊗	⊗	⊗
Konz.-Intervall					⊗
Ressourcenbedarf	⊖	⊖	⊖	⊖	⊖
Qualität	⊕	⊕	⊕	⊕	⊕

Tabelle 1.7: Relevante Faktoren der unterschiedlichen Verfahren zur Präzisionsgradberechnung

Tabelle 1.7 zeigt, daß die Qualität mit dem Ressourceneinsatz steigt[17]: Dies ist eine Grundvoraussetzung für die Integration in ein ressourcenadaptierendes System, wie es in Kapitel 4 beschrieben wird. Ein globaler Präzisionsgrad bewertet die gesamte Funktion, während die jeweilige lokale Variante einen kleineren Ausschnitt betrachtet. Einen noch detailgetreueren Faktor bezieht der Intervall-Präzisionsgrad mit der exakten Position relativ zu einem lokalen Maximum ein. Der Verbesserungsschritt vom ungewichteten zum gewichteten Flächen-Präzisionsgrad ist weniger mit dem konkreten Faktor (dem Anwendbarkeitsgrad) verknüpft, da dieser in allen Verfahren eine Rolle spielt, als vielmehr mit der Vermeidung kontra-intuitiver Ergebnisse. Aus diesem Grund wurde auf eine explizite Aufnahme des Anwendbarkeitsgrades in die Tabelle verzichtet. Der Ressourceneinsatz steigt, wenn auch sehr gering, durch die Multiplikation. In den beiden unteren Zeilen ist schematisch der wachsende Ressourcenbedarf und die steigende Qualität der Verfahren dargestellt.

1.5.6 Präzisionsgrade im Beispiel

In diesem Abschnitt soll durch einige ausgewählte Beispiele das Konzept des Präzisionsgrades illustriert werden[18]. Die aus realen Systemläufen gewonnenen Kurven wurden lediglich nachträglich annotiert.

Zuerst verdeutlicht Abb. 1.19 noch einmal den Nutzen des Präzisionsgrades: Während der Anwendbarkeitsgrad einer durch eine linguistischen Hecke (sehr)

[17] In der Tabelle korrespondiert die Größe der Darstellung von ⊕ bzw. ⊖ mit der Höhe der Qualität und des Ressourcenbedarfs.

[18] Das zur Visualisierung verwendete System wurde im Rahmen der Diplomarbeit Kray (1998) erstellt. Aus dieser Arbeit wurden auch die hier präsentierten Beispiele ausgewählt.

präzisierten räumlichen Relation (`rechts`) stets unterhalb der Kurve der *unverheckten* Relation liegt, ermöglicht hier der lokale Präzisionsgrad die Generierung des Ausdrucks *„sehr rechts"* im Intervall von 0 bis knapp 0,2 – also korrekt bei einer nur geringen Winkelabweichung.

Das zweite Beispiel vergleicht den globalen Flächen-Präzisionsgrad in seiner ungewichteten (vgl. Abb. 1.20) und gewichteten (vgl. Abb. 1.21) Ausprägung. Dabei geben die Intervalle unterhalb der x-Achsen die Bereiche an, in denen eine bestimmte Beschreibung generiert würde.

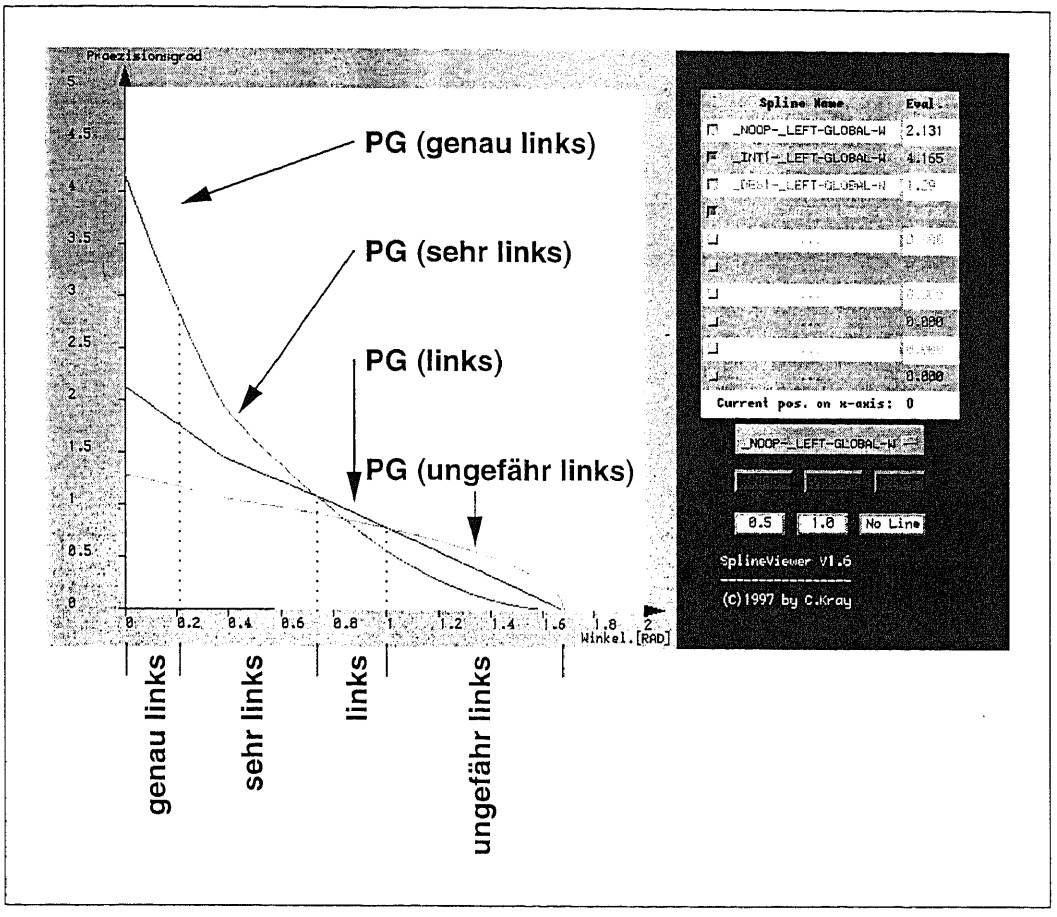

Abbildung 1.21: Globaler gewichteter Flächen-Präzisionsgrad

Deutlich ist beispielsweise zu erkennen, daß im ungewichteten Fall das vierte Intervall, das den Ausdruck *ungefähr links* repräsentiert, besonders großen Raum einnimmt. Diese Beschreibung würde also an mehr Positionen, die ein zu lokalisierendes Objekt einnehmen könnte, generiert als in der ungewichteten Variante des globalen Flächen-Präzisionsgrades. Ein besonders intuitives Resultat liefert der lokale Intervall-Präzisionsgrad: Je restriktiver der Heckenausdruck verstanden wird, umso kleiner ist das entsprechende Intervall in Abb. 1.22.

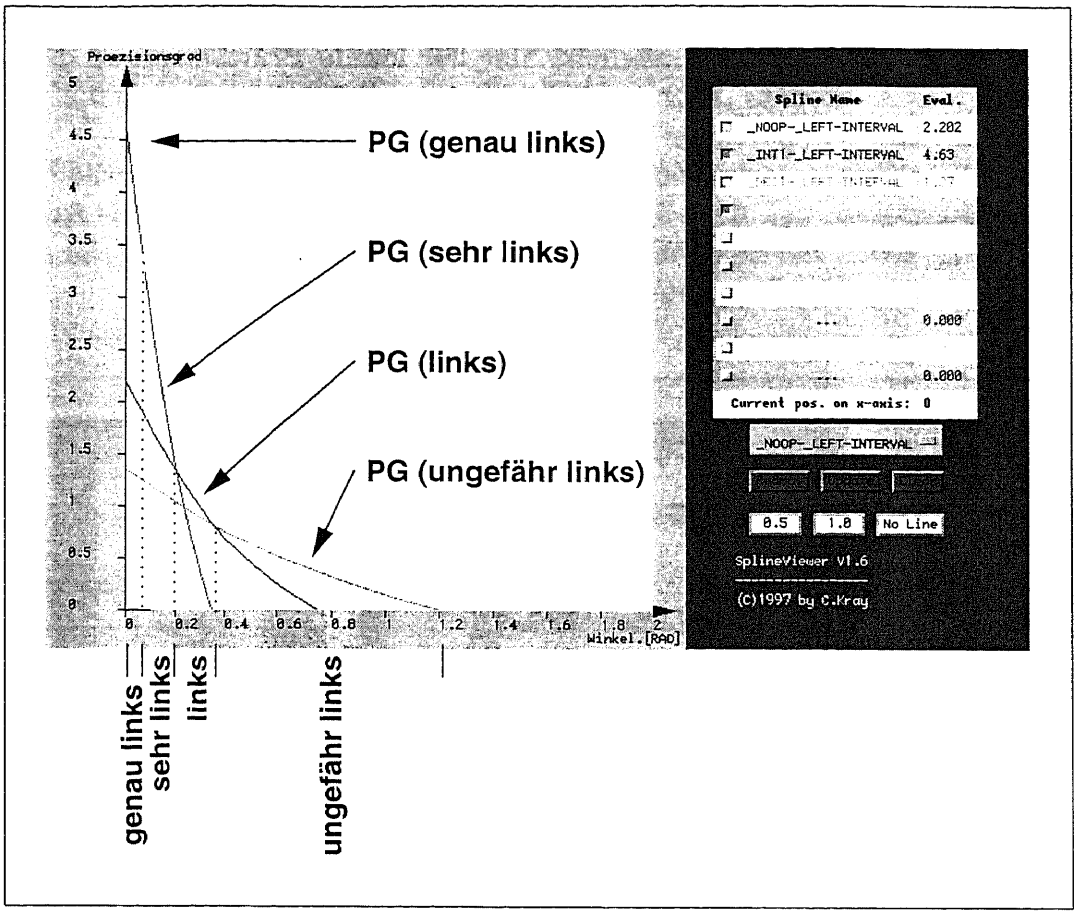

Abbildung 1.22: Lokaler Intervall-Präzisionsgrad

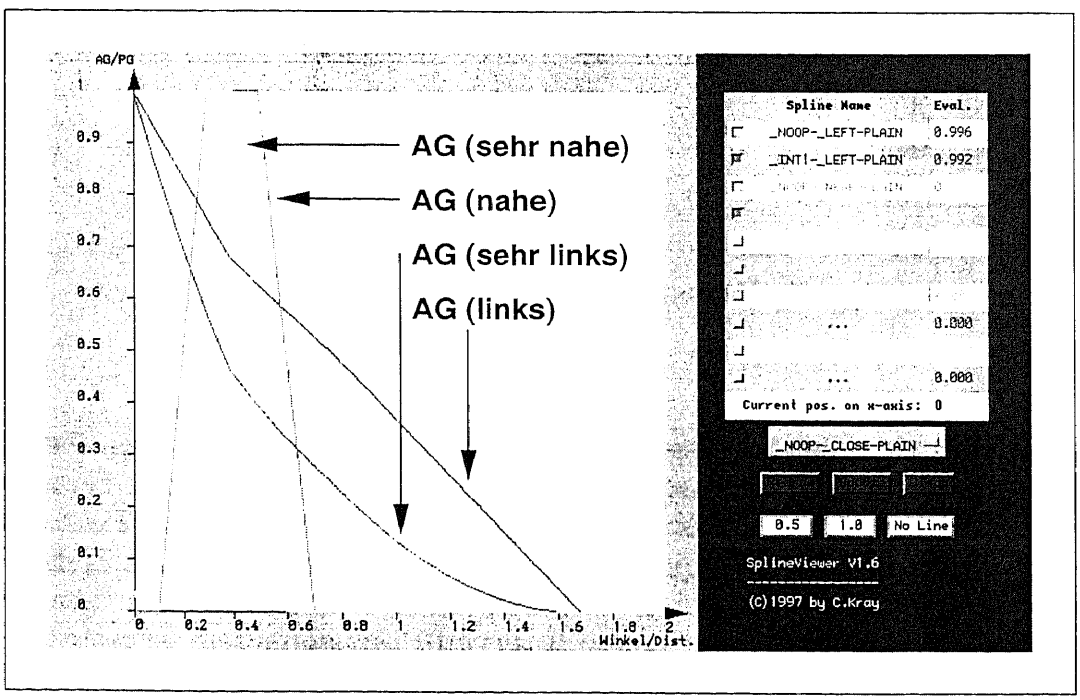

Abbildung 1.23: Auswirkungen einer linguistischen Hecke

Unterschiedliche Auswirkungen derselben linguistischen Hecke auf konzeptuell unterschiedliche räumliche Relationen (hier das distanzabhängige `nahe` und das winkelabhängige `links`) werden von Abb. 1.23 illustriert. Hier wird der Unterschied erkenntlich zwischen der in dieser Arbeit entwickelten mehrstufigen Modellierung unscharfer Konzepte und den bisherigen Ansätzen, die 1:1-Beziehungen von Relation und Hecke präferierten (vgl. z.B. (Lakoff, 1973)).

Zuletzt zeigt Abb. 1.24 die Auswirkung von Mehrfachanwendungen derselben linguistischen Hecke auf eine räumliche Relation. Dabei wird erstmals bei der Verarbeitung unscharfer Konzepte der Tatsache Rechnung getragen, daß jede weitere Anwendung eines Heckenausdrucks eine schwächere Wirkung hat als die vorangegangene.

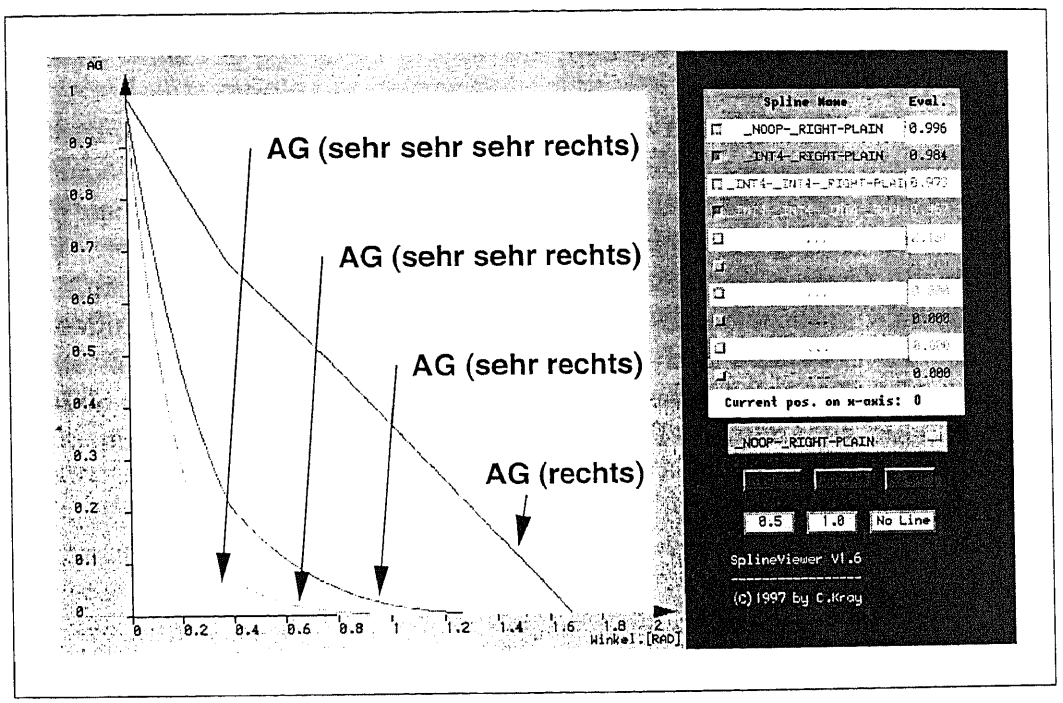

Abbildung 1.24: Mehrfachanwendung einer linguistischen Hecke

1.6 Zusammenfassung

In diesem Kapitel wurden zunächst die Grundlagen für das Sprechen über Raum gelegt, indem Begriffe wie Raumreferenz, Referenzrahmen und -objekt erläutert wurden. Von zentraler Bedeutung war dabei die räumliche Relation als einzelsprachunabhängiges Konzept. Nach einer Darstellung des bisherigen Ansatzes von Gapps Semantikmodell für statische räumliche Relationen, wurde argumentiert, daß dieses hinsichtlich des anstehenden Übergangs von Raumbeschreibungen zu ressourcenadaptierenden Wegbeschreibungen erweitert werden muß.

Dazu wurden zum einen die bestehenden Konzeptualisierungen um die Formalisierung und Modellierung sogenannter Pfad- oder n-Punkt-Relationen

ergänzt, die auf der sprachlichen Seite mit wegbezogenen Ausdrücken korrespondieren. N-Punkt-Relationen basieren dabei auf denselben Parametern Distanz und Winkelabweichung wie die bereits vorhandenen statischen Relationen. Dies ist für eine allgemeine Vergleichbarkeit erforderlich, ohne die die Bestimmung bester Relationen nicht möglich ist.

Zum anderen wurden Verfahren zur Formalisierung und Modellierung von Vagheit durch unscharfe Konzepte entwickelt, die es ermöglichen, räumliche Relationen durch linguistische Heckenausdrücke zu modifizieren, um so die Verständlichkeit einer Äußerung zu erhöhen. Ferner wurde das Konzept des Präzisionsgrades eingeführt, der die Vergleichbarkeit aller beteiligten Konzepte – seien sie binär oder unscharf – bezüglich ihrer Genauigkeit ermöglicht. Sowohl die Modellierung unscharfer Konzepte als auch des Präzisionsgrades sind zwar im Rahmen des Themas *Raumbeschreibung* entstanden, doch konnten sie so allgemein gehalten werden, daß sie für beliebige gradierte Konzepte verwendet werden können.

Zusammenfassend konnte der Gapp'sche Ansatz insofern erweitert werden, daß erstmals wegbezogene Relationen und die Verarbeitung vager Konzepte zur ressourcenadaptierenden Präzisierung für ein KI-System zur Raum- und Wegbeschreibung entwickelt und in dieses integriert wurden. Damit konnte für ein System der Künstlichen Intelligenz wie den beschränkt-optimalen Lokalisationsagenten BOLA das Spektrum sprachlicher Raumbeschreibung erheblich erweitert und im Sinne von Grice kommunikativer werden, allerdings ohne die sprachlichen Möglichkeiten bereits vollständig abdecken zu können. Hier sind weitere Forschungsanstrengungen erforderlich.

Kapitel 2

Ressourcenbeschränkungen

In diesem Kapitel wird als weiterer Teil der Grundlagen eines *beschränkt-optimalen Lokalisationsagenten* das Gebiet der *ressourcenbeschränkten Berechnung* behandelt. Die Notwendigkeit einer aktiven Verarbeitung von Ressourcenbeschränkungen zeigt sich insbesondere bei der Problematik dialogbasierter, inkrementeller Wegauskunftssysteme oder in der Robotik. In beiden Fällen kann z.B. die Ressource *Zeit* eine bedeutende Rolle spielen, etwa insofern, daß die Erledigung einer Aufgabe nach einer bestimmten Zeitspanne nutzlos wird: Hat der Autofahrer die Kreuzung schon passiert, ist die Aufforderung *„Bitte biegen Sie an der nächsten Kreuzung rechts ab"* sinnlos geworden.

Die Integration des Begriffs der *Ressource* in neuen Forschungsansätze erfordert zunächst eine Definition derselben im Kontext der Künstlichen Intelligenz, da hier Input-Faktoren wie verfügbare Rechenzeit, Speicherkapazität oder Prozessorleistung bislang meistens idealisierend als unbeschränkt angenommen wurden. Ausgehend von diesem Ressourcenbegriff werden Konzepte wie *beschränkte Optimalität* und schließlich eine Klasse von Berechnungsverfahren, die sogenannten *Anytime-Algorithmen*, abgeleitet, die im weiteren von zentraler Bedeutung sind. Begleitend werden vorhandene Forschungsergebnisse vorgestellt und diskutiert.

2.1 Ressourcenbegriff

Das Konzept der *Ressourcenbeschränkung* wird erst seit kurzer Zeit im Bereich der KI betrachtet. Daher erscheint es notwendig, vorweg eine Begriffsklärung vorzunehmen, da der Terminus *Ressource* in unterschiedlichen Forschungsgebieten auch teilweise unterschiedliche Bedeutungen besitzt. Dies leistet für die interdisziplinäre Forschung im SFB 378 eine terminologische Festlegung von Jameson und Buchholz (1998) aus kognitionswissenschaftlicher Sicht, die im folgenden paraphrasiert wird. Dabei soll nicht der aktuelle Sprachgebrauch beschrieben sondern ein Vorschlag zur Verwendung gemacht werden, der eine Analyse der Verarbeitung von Ressourcenbeschränkungen bei natürlichen und künstlichen Agenten erlaubt. Als *Agent* wird nach Russell und Norvig (1995) *„eine Entität, die ihre Umgebung durch Sensoren wahr-*

nimmt und in ihr durch Effektoren agiert" bezeichnet. Sensoren und Effektoren des Menschen sind z.B. Augen bzw. Hände, bei einem Roboter etwa Kameras respective Manipulatoren.

Jameson und Buchholz definieren den Begriff *Ressource* weit gefaßt:

Definition 2.1 (Ressource) *Eine* Ressource *ist ein Hilfsmittel zur Lösung bestimmter Aufgaben.*

Typische Ressourcen des täglichen Handelns sind Gegenstände, Fähigkeiten, Information, Energie und Zeit. Jameson und Buchholz unterscheiden *Ressourcentypen* hinsichtlich ihrer Eigenschaften: So gibt es Ressourcen, die beliebig portionierbar sind, wie die Zeit, und solche die nur als Ganzes verwendet werden können, wie eine Telefonnummer. Einige werden verbraucht (Strom), andere nicht (Fähigkeit des Lesens). Ressourcen können ferner bezüglich der *Kosten* und der Beschränkungen ihrer Verwendung analysiert werden. Über den Kostenfaktor kann möglicherweise eine Ressourcensubstitution durchgeführt werden. Eine Ressourcenbeschränkungen bei einer Verwendung kann ein Zeitlimit zum Erfüllen einer Aufgabe sein, wie beim komplexen Problem des Prozeßscheduling in Betriebssystemen, dem sogenannten *Kontextwechsel* (vgl. (Tanenbaum, 1992)). In den Abschnitten 2.5.2 und 2.6.2 wird gezeigt, wie Ressourcen (und ihre Verwendung) auf der Outputseite z.B. mittels *Erfolgsmetriken* und *Performanzprofilen* untersucht und bewertet werden können (vgl. (Dean & Boddy, 1988) und Abschnitt 2.5.2). Abbildung 2.1 stellt den entwickelten Begriffsrahmen graphisch dar (nach (Jameson & Buchholz, 1998)).

In Bezug auf das Thema des SFB 378, ressourcenadaptive kognitive Prozesse, leiten Jameson und Buchholz einige Fragetypen ab, von denen die für die vorliegende Arbeit interessanten die folgenden sind:

- Wie können die verfügbaren Ressourcen effektiv an die zu erledigenden Aufgaben zugeteilt werden?
- Wie können auch diejenigen Ressourcen effektiv eingesetzt werden, deren Verfügbarkeit nicht von vornherein bekannt ist?
- Bei inhaltlich heterogenen Informationsressourcen: Wie können die Ressourcen zusammen ausgenützt werden?
- Welche Form haben die relevanten Erfolgsprofile, und wie können sie theoretisch erklärt werden?
- In welchem Sinne ist der analysierte kognitive Prozeß *ressourcenadaptiv*?

Der nächste Abschnitt befaßt sich ausführlich mit der letzten Frage.

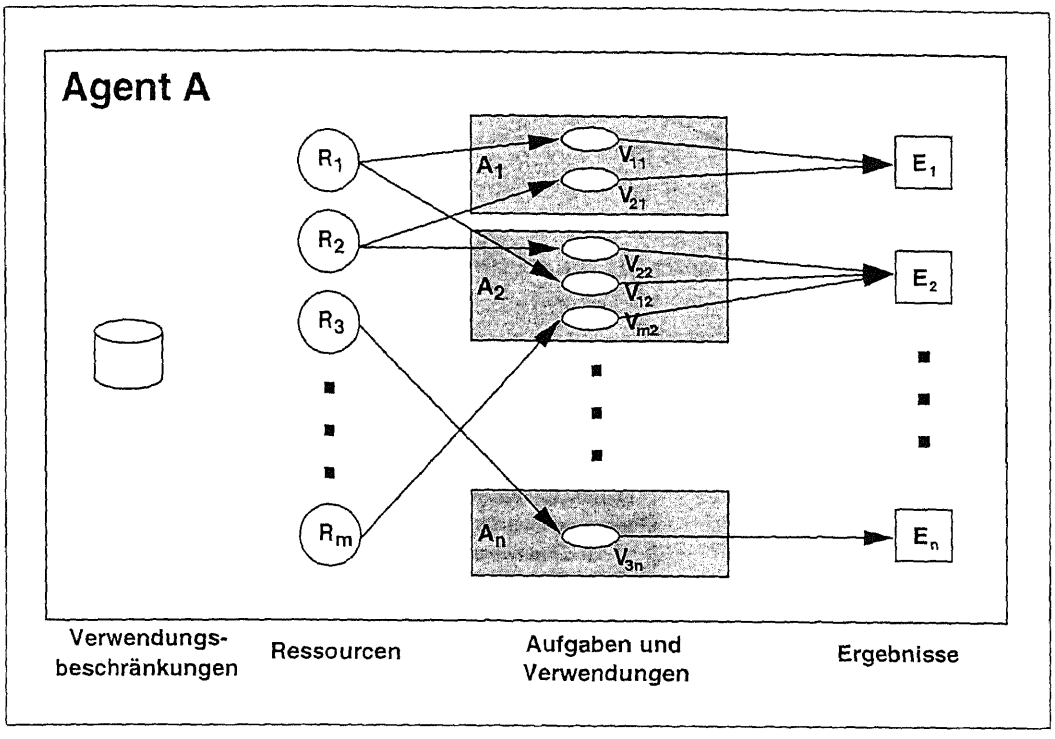

Abbildung 2.1: Kognitionswissenschaftliches Ressourcen-Konzept: Verwendung durch einen Agenten

2.2 Ressourcensensitivität

Wahlster und Tack (1997) sprechen bei der Berücksichtigung von Ressourcenbeschränkungen, wie sie insbesondere die beiden ersten Fragestellungen von Jameson und Buchholz an Ende des vorigen Abschnitts nahelegen, von *Ressourcensensitivität* oder *ressourcensensitivem Verhalten*, wobei sie zuerst nach der Art der Ressourcenbeschränkung selbst und letztlich gemäß der Verarbeitungsstrategie insgesamt drei Typen unterscheiden (vgl. Abb. 2.2).

Auf der untersten Stufe findet sich *ressourcenadaptiertes Verhalten*, bei dem aus informatischer Sicht ein Algorithmus auf bekannte und feste Ressourcenbeschränkungen hin optimiert ist. Eine konstante Eingabequalität wird ungeachtet von Ressourcenveränderungen immer dieselbe Resultatsqualität liefern. Die zweite und dritte Klasse ressourcensensitiver Prozesse kann variable Ressourcenbeschränkungen adäquat verarbeiten, d.h. die Ausgabequalität hängt neben dem Input auch von den vorhandenen Ressourcen ab. Unterschieden werden diese Typen anhand der Verarbeitungsstrategie: Von *ressourcenadaptiven Verhalten* spricht man, wenn eine feste, aber von den akuten Ressourcenbeschränkungen abhängige Strategie bei der Bearbeitung eines Problems verfolgt wird. Mittels *Meta-Wissen* kann z.B. *ex-ante* ein bestimmter Algorithmus, der die vorhandenen Ressourcen optimal ausnutzt, vom Agenten gewählt werden; ist das geschehen, so können eventuelle spätere Veränderungen der Ressourcen nicht mehr berücksichtigt werden. Genau dies leistet jedoch *ressourcenadaptierendes Verhalten*. In diesem

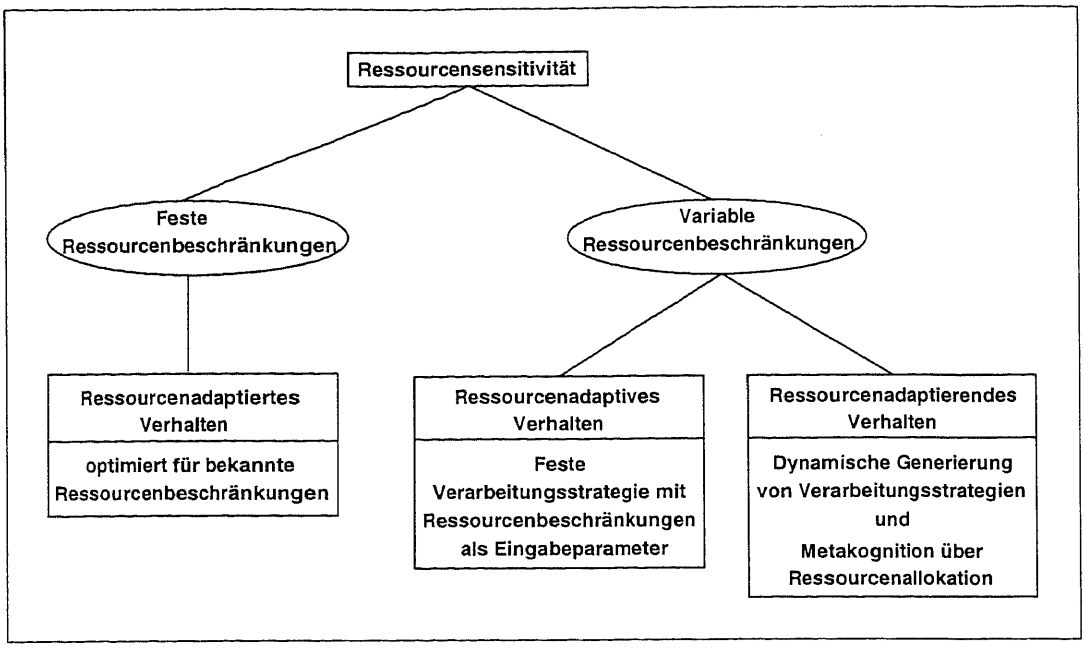

Abbildung 2.2: Ressourcensensitivität: Klassifikation nach Freiheitsgraden

Fall können durch Meta-Wissen Entscheidungen über Strategien und Strategiewechsel dynamisch, abhängig von aktuellen Ressourcenbeschränkungen getroffen werden.

Der Agent wählt zwischen Verfahren, die unterschiedlichen Ressourcenbedarf haben und deren Ergebnisse qualitativ differieren. Stehen viele Ressourcen zur Verfügung, so können diese voll ausgenutzt werden, was zu einem guten, verläßlichen Resultat führt. Umgekehrt liefert eventuell eine andere Strategie unter starken Restriktionen ein schlechteres Ergebnis. Notwendig hierbei ist allerdings, daß überhaupt verschiedene Lösungsstrategien mit unterschiedlichem Ressourcenverbrauch existieren, und daß es einen Zusammenhang zwischen Ressourceneinsatz und Ergebnisqualität gibt. Damit ressourcenadaptierende Verfahren Antworten zu den von Jameson und Buchholz genannten Fragestellungen liefern können, muß nun noch das Problem der Effizienz betrachtet werden: Wie kann eine möglichst optimale *Ressourcenverteilung* erreicht werden?

2.3 Rationalität und beschränkte Optimalität

Der Begriff der *Rationalität*, wie ihn Neumann und Morgenstern (1947) definiert haben, beschreibt einen Agenten, der aus einer Menge möglicher Aktionen diejenige ausführt, die bezüglich des zu erwartenden Resultats den größten Nutzen zeitigt. Dies setzt voraus, daß probabilistische Information über Handlungen und ihre Ergebnisse vorliegen und deren Qualität mittels sogenannter *Nutzenfunktionen* bestimmt werden kann. Diese *perfekte Rationalität* kann ein (künstlicher) Agent aber nicht erreichen: Insbesondere müßte jede Entscheidung darüber, welche Aktion auszuführen sei, in Null-

Zeit getroffen werden, da sich die Welt während des *Nachdenkens* verändert. Good (1971) unterscheidet zwischen *Typ I Rationalität*, wie er perfekte Rationalität nennt, und *Typ II Rationalität* (oder *Meta-Rationalität*), die berücksichtigt, daß der Agent nachdenken muß, bevor er handelt. Bei der Maximierung des Nutzen sind die dabei entstehenden Kosten in Rechnung zu stellen. Dies ist aber nur eine scheinbare Vereinfachung des Problems, da es einfach auf eine höhere Ebene verlagert wird; auch Typ II Rationalität kann ein Agent in der realen Welt nicht erreichen.

Schließlich wurde das Konzept der *beschränkten Rationalität* eingeführt, um dem Problem der durch den *Reasoning-Prozeß* entstehenden Kosten zu Leibe zu rücken. Dabei wird die tatsächliche Berechnungsfähigkeit der Agenten als Grundlage genommen und aus allen möglichen Implementationen diejenige mit der besten Nutzenfunktion gewählt. Doch auch dieser Ansatz ist in der Praxis unbrauchbar, da eine direkte Konstruktion dieser besten Implementation nicht möglich ist.

In einem weiteren Ansatz trennt Zilberstein (1993) zwischen Kontrollinstanz und Lösungsverfahren. Bei dieser *operationalen Rationalität* optimiert der Agent auf einer Meta-Ebene die Zuteilung der Ressourcen an Berechnungskomponenten, über die selbst kein weiteres Reasoning betrieben wird.

Eine in der Praxis anwendbare Eigenschaft von Agenten ist die von Russell und Subramanian (1995) ausgearbeitete *beschränkte Optimalität*. Dabei wird Optimalität nur bezogen auf die tatsächlichen Fähigkeiten des Agenten, d.h. ein Programm auf einem mit bestimmten Ressourcen ausgestatteten Computer muß mindestens so gut wie alle anderen implementierbaren Alternativen sein.

Ein entscheidender Faktor ist allen Ansätzen gemein: Zur – wie auch immer gearteten – optimalen Ressourcenverteilung muß der Agent über *Meta-Wissen* verfügen, d.h. er verwendet Wissen über die zur Verfügung stehenden Ressourcen und Lösungsstrategien, um über ihren jeweiligen Einsatz bei der Problemlösung zu entscheiden. Die Ausübung einer solchen Meta-Kontrolle verbraucht *Zeit*, selbst eine in der Regel beschränkte Ressource. Dies macht deutlich, daß (auch) beschränkte Optimalität nicht ohne Konzepte zur Behandlung dieser zentralen Ressource auskommen kann.

2.4 Zeit als Ressource

In ihrem Ansatz *Value of Computation* entwickeln Russell und Wefald (1991) eine Meta-Kontrolle, die es einem Agenten ermöglicht, die Eigenschaft der beschränkten Optimalität zu erfüllen. Dabei gehen sie von einer *Nutzenfunktion* aus, die jedem Zustand, in dem sich ein Agent befinden kann, eine reelle Zahl zuordnet, die die Güte dieses Zustandes bewertet. Der Agent muß zu jedem Zeitpunkt entscheiden, ob er eine (weitere) Aktion auf der Meta-Ebene (*Reasoning*) oder die bis jetzt als optimal bewertete Handlung auf der Performanz-Ebene (also eine eigentliche Berechnungen) vornehmen soll, um so einen Zustand mit höherer Qualität zu erreichen. Da die Nutzenfunktion – wie bereits oben erwähnt – sinnvollerweise auch die Größe Zeit, die z.B.

auf der Meta-Ebene verbraucht wird, berücksichtigen sollte, kann diese als Kostenfaktor abgetrennt werden, so daß man eine Abhängigkeit des Nutzen von der Zeit erhält. In Abb. 2.3 wird dies für drei unterschiedliche Ansätze dargestellt.

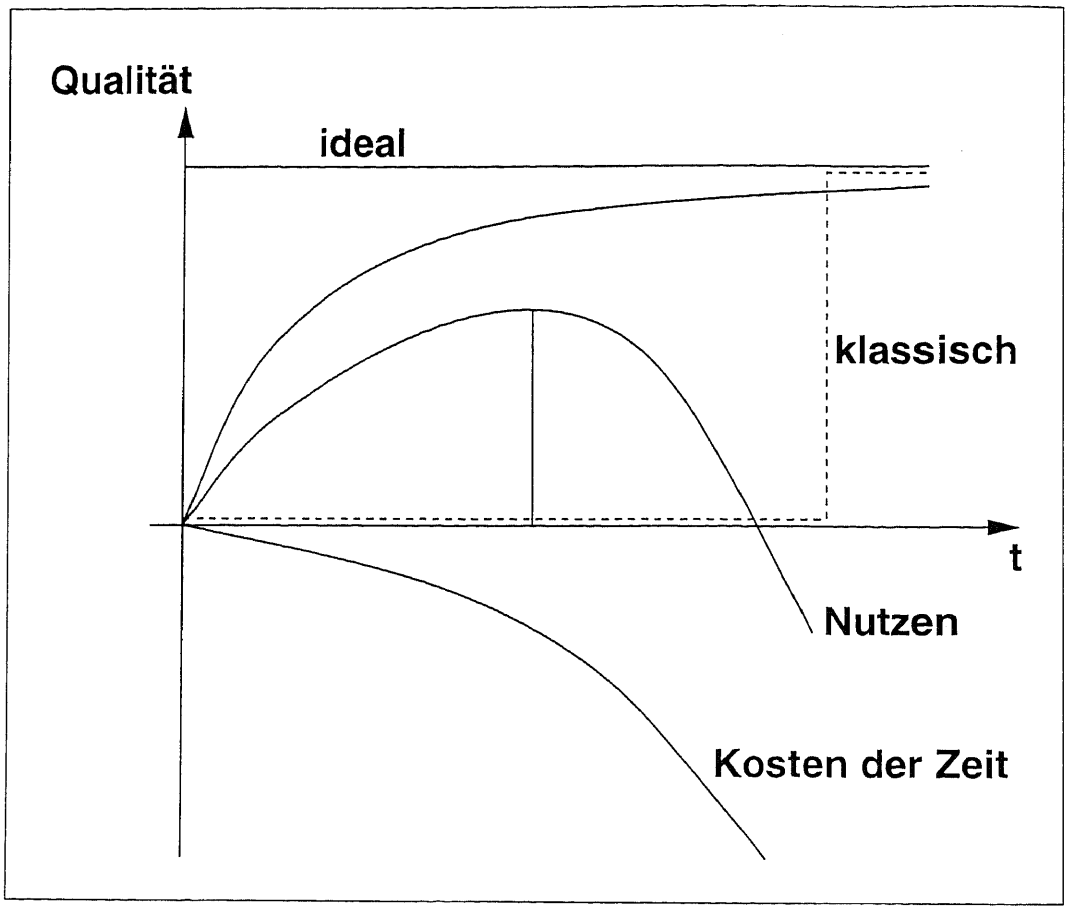

Abbildung 2.3: Qualitätsentwicklungen sowie Kosten und Nutzen in Abhängigkeit der Zeit

Ein Agent, der über die Eigenschaft der perfekten Rationalität verfügte, könnte das Ergebnis ohne jeden Zeitverlust berechnen. Dies entspricht dem Verlauf der *idealen* Kurve. Der klassische Ansatz benötigt eine gewisse Zeit zur Berechnung. Bis das Resultat vorliegt, ist der Nutzen prinzipiell negativ (er entspricht den Kosten des Zeitverbrauchs ohne *Gegenleistung*). Betrachtet man die Idee der beschränkten Optimalität, so erlaubt die durchgeführte Meta-Kognition, eine jederzeitige Unterbrechung der Berechnung und die Ausgabe des bislang als bestes erkannten Resultats. Falls die Ressource Zeit ausreicht, wird natürlich die optimale Qualität erreicht, wobei zu berücksichtigen ist, daß der Zeitverbrauch der Meta-Ebene die Gesamtrechenzeit verlängert. Abbildung 2.3 zeigt deutlich, daß sowohl beim klassischen als auch beim Meta-Ebenen-Ansatz der Nutzen eventuell negativ, das Ergebnis also unbrauchbar werden kann. Während dies allerdings im ersten Fall nicht mehr beeinflußbar ist, kann im zweiten zumindest ein suboptimales Resultat geliefert werden. Dabei lassen sich drei Arten von Be-

schränkungen der Ressource *Zeit* unterscheiden, deren Vorliegen von der jeweiligen Problemstellung abhängen:

- Es existiert eine *fixe Zeitschranke* nach deren Ablauf etwaige Resultate nutzlos sind.
- Es existiert eine *definierte Kostenfunktion* mit dem Ziel, diese zu minimieren; alternativ bietet sich die Maximierung der Nutzenfunktion (unter Berücksichtigung der Kosten) an.
- Es existiert eine *stochastische Zeitschranke*. So könnte z.B. eine Wahrscheinlichkeitsverteilung angeben, ab wann ein erzieltes Ergebnis nutzlos ist.

Der folgende Abschnitt behandelt einige praktische Forschungsansätze, die sich mit Aspekten der beschränkten Optimalität befassen.

2.5 Ansätze zur ressourcenbeschränkten Berechnung

2.5.1 Flexible Berechnungen (Horvitz)

Die Idee der Maximierung des Nutzen von Russell und Wefald (1991) aus Abschnitt 2.4 schlägt sich in der Verwendung *flexibler Berechnungen* zur Lösung von Problemen in Abhängigkeit vorhandener Ressourcen nieder, ein Ansatz den Horvitz (1987) initiierte. Sein medizinisches Diagnosesystem PROTOS verwendet flexible Strategien zum automatisierten Schließen unter Unsicherheit, bei dem mittels Teilergebnissen Näherungslösungen unterschiedlicher Qualität berechnet werden. Dabei bezeichnet er eine Verbesserung eines schon erzielten Resultats unter Erhöhung der aufgewendeten Zeit als *Verfeinerung*. Horvitz stellt an die verwendeten Algorithmen folgende Anforderungen:

- *Kontinuität des Nutzen und der Qualität:* Nutzen und Qualität, die im Verlauf einer *Verfeinerung* des Ergebnisses berechnet werden, stellen stetige Funktionen dar.
- *Monotonie der Qualität:* Die Qualitätswerte müssen sich monoton steigend zum Ressourceneinsatz verhalten.
- *Konvergenz der Qualität:* Die Qualität konvergiert gegen die des optimalen Ergebnisses, das berechnet würde, wenn genügend Ressourcen zur Verfügung stehen würden.
- *Dominanz des Nutzen:* Es existieren Zeitintervalle, während deren die Nutzenfunktion der Algorithmen bezüglich des Ressourcenverbrauchs monoton steigt.

2.5.2 Anytime-Algorithmen (Dean und Boddy)

Der von Dean und Boddy (1988) geprägte Begriff *Anytime-Algorithmus* baut auf Horvitz auf und ist eine Formalisierung seines Ansatzes der flexiblen Berechnungen (vgl. auch (Boddy, 1991b, 1991a)). Dazu verwendet der dort vorgestellte Agent eine Meta-Kontrolle analog zu Russell und Wefald, die einzelnen Teilaufgaben Zeit(-Ressourcen) zuordnet. Die Autoren illustrieren dies am Beispiel eines stationären Roboters, der Objekte auf einem Fließband erkennen und sortieren muß. Dabei hat er keinen Einfluß auf die Dauer, die er zur Erledigung beider Aufgaben hat. Die konkreten Aufgaben werden von Anytime-Algorithmen ausgeführt, die folgende Eigenschaften besitzen:

1. Anytime-Algorithmen können jederzeit mit minimalem administrativem Aufwand unterbrochen und wieder fortgesetzt werden.

2. Anytime-Algorithmen stellen zu jedem Zeitpunkt ein Ergebnis bereit, dessen Qualität als Funktion der investierten Arbeitszeit monoton ansteigt.

3. Anytime-Algorithmen können sogenannte *Performanzprofile* zugeordnet werden, die die Relation von aufgewendeter Zeit zu erzielter Qualität kodieren. Performanzprofile sind *per definitionem* monoton steigend (vgl. Punkt 2).

Diese Eigenschaften lassen sich wie folgt in Definitionen fassen:

Definition 2.2 (Anytime-Algorithmus) *Ein* Anytime-Algorithmus *kann zu jedem Zeitpunkt unterbrochen werden, wobei die Qualität des jeweils erreichten Zwischenresultats mit der Laufzeit monoton steigt.*

Definition 2.3 (Performanzprofil) *Ein* Performanzprofil *beschreibt die Qualitätsentwicklung eines Anytime-Algorithmus' im Verhältnis zur eingesetzten Zeit.*

Im Gegensatz zu einem Anytime-Algorithmus entspricht ein *Standard-Algorithmus* einer Turing-Maschine. Er ist nicht unterbrechbar, auf Zwischenergebnisse kann nicht zugegriffen werden. Der Begriff des *schwachen Anytime-Algorithmus'* wird von Menzel (1994) eingeführt. Seine Verfahren besitzen zwar ebenfalls monotone Performanzprofile, sind aber nicht für alle Anwendungen identisch. Als Beispiel dient ihm ein constraint-basierter Parsing-Algorithmus.

Anytime-Algorithmen wurden bislang vorwiegend in iterativen Näherungsverfahren angewendet, in denen eine Unterbrechung problemlos möglich ist. Die Verwendung in komplexen symbolischen Aufgaben bzw. in komplexen kognitiven Prozessen, wie Analyse oder Generierung natürlicher Sprache, Planung, Deduktion und visuelle Perzeption, wurde noch kaum erforscht. Auf diesem Gebiet wird die Unterbrechbarkeit von Anytime-Algorithmen durch das Transaktionskonzept von Görz und Kesseler (1994) beschränkt.

Definition 2.4 (Transaktion) *Eine Folge von Berechnungsschritten, die nicht unterbrochen werden kann, heißt* Transaktion.

Mit diesem Konzept, das ursprünglich aus den Bereich der Datenbank- und Betriebssysteme (vgl. beispielsweise (Silberschatz & Galvin, 1994)) stammt, sollen Synchronisationsprobleme, die bei der Verwendung nebenläufiger Prozesse auftreten können, vermieden werden. Durch das Sicherstellen, daß das System immer von einem korrekten Zustand zu einen nächstem korrekten Zustand übergeht, können inkonsistente Zugriffe ausgeschlossen werden[1].

2.5.3 Compilierung von Anytime-Algorithmen (Zilberstein)

Aufbauend auf dem Konzept der jederzeit unterbrechbaren Anytime-Algorithmen schlägt Zilberstein (1993) zur Konstruktion größerer Systeme eine *Offline-Compilierung* vor (vgl. auch (Zilberstein, 1996)). Sie nimmt die Verteilung der Ressourcen auf die einzelnen Komponenten vor. Zilberstein führt den Begriff des *bedingten Performanzprofils* ein, das die Relation zwischen eingesetzter Ressource und Ergebnisqualität unter Berücksichtigung einer bestimmten Eingabequalität beschreibt. Dies ermöglicht die sequentielle Abarbeitung von Anytime-Algorithmen und die Verwendung von Teilergebnissen als Input beim Compilierungsvorgang. Das letztlich konstruierte Gesamtsystem kann mittels einer Meta-Kontrolle, bei Zilberstein als *Monitoring* bezeichnet, auch zur Laufzeit überwacht und in begrenzter Weise optimiert werden. So ist zwar eine Umverteilung von Ressourcen möglich, eine flexible Anpassung an sich verändernde Umweltzustände aber nicht. Insbesondere muß die Gesamtlaufzeit des Systems zu Beginn bekannt sein.

Definition 2.5 (Monitoring) *Kontrolle und Optimierung der Ressourcenzuteilung eines Systems wird* Monitoring *genannt.*

Anytime-Algorithmen und verwandte Konzepte spielen eine tragende Rolle in der vorliegenden Arbeit; sie werden deshalb in Abschnitt 2.6 eingehender vorgestellt.

2.5.4 Design-to-Time-Scheduling (Garvey und Lesser)

Die Idee des *Design-to-Time-Scheduling* von Garvey und Lesser (1993) geht von der Annahme aus, daß bei der Lösung einer komplexen Aufgabe häufig

[1] Wittig (1998) illustriert den Nutzen von Transaktionen wie folgt: *„Als einfaches Beispiel zum Transaktionskonzept und der damit lösbaren Probleme kann die Überweisung eines Geldbetrags von einem Konto auf ein anderes dienen. Der Betrag wird vom Kontostand des ersten abgezogen und zu dem des zweiten hinzuaddiert. Bevor nicht beide Aktionen komplett ausgeführt sind, darf der Vorgang nicht unterbrochen werden, denn sonst könnten Abfragen über den Kontostand Inkonsistenzen liefern. Tritt eine Unterbrechung zum Beispiel nach dem Abziehen des Betrags vom ersten Konto auf und werden dann die beiden Kontostände betrachtet, so verschwindet das zu überweisende Geld. Da aber die triviale Nebenbedingung gilt, daß die Gesamtgeldmenge der beiden an dem Vorgang beteiligten Konten sich zu keinem Zeitpunkt verändern darf, muß die Überweisung als Transaktion im obigen Sinne aufgefaßt werden und darf nur in ihrer Gesamtheit und ohne Unterbrechung durchgeführt werden.“*

mehrere Verfahren zur Bearbeitung einzelner Teilprobleme existieren, die sich bezüglich ihres Ressourcenverbrauchs, aber auch hinsichtlich der Ergebnisqualität unterscheiden (vgl. auch (Garvey & Lesser, 1994; Garvey, Humphrey & Lesser, 1993)).

Die Struktur eines Problems wird dabei als Baum kodiert. Während die Blätter konkrete Berechnungen repräsentieren, stehen innere Knoten für (eventuell abstrakte) Teilprobleme. Zeitliche Abhängigkeiten zwischen Knoten werden durch zusätzliche gerichtete Kanten repräsentiert: Die zu einem Ausgangsknoten gehörende Berechnung muß beendet sein, bevor die des Eingangsknoten begonnen werden darf. Diese Abhängigkeiten können auch zwischen ansonsten voneinander unabhängigen Teilproblemen (Subbäumen) vorkommen. Das Gesamtsystem verfügt ebenso wie bei Zilberstein über eine Monitoring-Komponente, die Abweichungen zwischen den der Ressourcenzuteilung zugrunde liegenden Annahmen und der Realität erkennen und verarbeiten kann.

Das Konzept wurde hinsichtlich einer Integration von Anytime-Algorithmen erweitert (vgl. (Garvey & Lesser, 1996)), so daß deren spezifischen Vorteile genutzt werden können: Insbesondere ist die Verwendung desselben Anytime-Algorithmus' mit unterschiedlichen Laufzeiten möglich, um alternative Gesamtsysteme zu konstruieren.

2.5.5 Controlled Concurrency (Wang)

Die interne Struktur des Interferenzsystems NARS (Non-Axiomatic Reasoning System) basiert auf der von Wang (1996) entworfenen *Controlled Concurrency* zur nebenläufigen Verarbeitung einzelner Aufgaben, sogenannter *Tasks*. Die Gesamtheit aller Tasks bildet ein System, das ähnliche Eigenschaften wie ein Anytime-Algorithmus aufweist: Unterbrechbarkeit und monoton steigende Ergebnisqualität.

Alle Tasks sind (z.B. bezüglich der Inferenztiefe) einzeln parametrisierbar und erhalten zusätzlich jeweils zwei Werte zugeordnet: *Dringlichkeit* und *Verfall*. Ersteres erlaubt die Induktion einer Ordnung auf die Menge der Tasks. So erhalten zwei Tasks A und B mit den Dringlichkeiten d_A und d_B die vorhandenen Ressourcen im Verhältnis d_A/d_B. Der zweite Wert, der sogenannte Verfall, gibt an, wie sich die Dringlichkeit relativ zur Zeit verhält. Damit kann sozusagen der Alterungsprozeß eines Tasks modelliert werden.

Auch in diesem Ansatz können die Tasks als Anytime-Algorithmen implementiert werden; Dringlichkeit und Verfall lassen sich dann aus den entsprechenden Performanzprofile ableiten.

2.5.6 Die Ansätze im Vergleich

Die vergleichende Gegenüberstellung der oben abgehandelten Forschungsansätze zur ressourcenbeschränkten Berechnung umschließt nur solche, deren Ziel die Konstruktion von Meta-Kontrollen für komplexe Gesamtsysteme aus einzelnen Komponenten ist. Somit bleiben *flexible Berechnungen* und *Anytime-Algorithmen*, deren Eigenschaften sich auf einzelne Berechnungen

Eigenschaft	Forschungsansatz			Forschungs-Ziel
	Compilierung von Anytime-Algorithmen	Design-to-Time-Scheduling	Controlled Concurrency	
Standard-Algorithmen	*nicht* verwendbar	verwendbar	*nicht* verwendbar	*verwendbar*
Anytime-Algorithmen	verwendbar	verwendbar	verwendbar	*verwendbar*
Lösungs-strategie(n)	eine	mehrere	eine	*mehrere*
Abhängige Teilaufgaben	zulässig	zulässig	*nicht* zulässig	*zulässig*
Unterbrechbarkeit	Komponenten	Komponenten	Gesamt-System	*Gesamt-System*
Monitoring-Komponente	passiv (aktiv)	passiv aktiv	passiv (aktiv)	*passiv aktiv*
Gesamt-ressourcen	vorgegeben	vorgegeben	variabel	*variabel*
Bewertung (Punkte)	2,75	5,00	3,75	*7,00*

Tabelle 2.1: Vergleich der vorgestellten Forschungsansätze zur ressourcenbeschränkten Berechnung

beziehen, hier unberücksichtigt. Tatsächlich können (teilweise müssen) sie in jedem der Ansätze, quasi als Dienstleister der eigentlichen Arbeit eingesetzt werden.

Der nachfolgende Vergleich der verbleibenden drei Konzepte *Compilierung von Anytime-Algorithmen*, *Design-to-Time-Scheduling* und *Controlled Concurrency* wird zur besseren Übersichtlichkeit auch in Tab. 2.1 dargestellt[2].

Die Möglichkeit der Verwendung von *Standard-Algorithmen* bei der Konstruktion des Gesamtsystems ist nur bei Design-to-Time-Scheduling gegeben. Der Vorteil einer solchen Fähigkeit besteht darin, eventuell schon vorhandene, erprobte Methoden weiter einsetzen und auf die Entwicklung neuer Anytime-Algorithmen verzichten zu können.

Umgekehrt erlaubt die Möglichkeit der Verwendung von *Anytime-Algorithmen* die Nutzung ihrer Vorteile, insbesondere jederzeitige Unterbrechbarkeit und die in den Performanzprofilen abgelegte Information. Alle drei betrachteten Ansätze leisten dies.

Die Unterstützung unterschiedlicher *Lösungsstrategien* hingegen wird wieder nur vom Design-to-Time-Scheduling gewährleistet. Ihre Verwendung bietet einer Meta-Kontrolle eine erhöhte Zahl signifikanter Variationsmöglichkeiten zur Ressourcenallokation, vor allem dann, wenn sich die entsprechenden Verfahren deutlich hinsichtlich Ressourcenverbrauch und Ergebnisqua-

[2]Die Tabelle lehnt sich an Wittig (1998) an.

lität unterscheiden, im Gegensatz etwa zu einem einfachen iterativen Algorithmus.

Die Bearbeitung *abhängiger Teilaufgaben* gewährleistet eine flexible Entwicklung der Systemstruktur und einen modularen Systemaufbau. Wangs Controlled Concurrency läßt dies nicht zu.

Die jederzeitige *Unterbrechbarkeit* des konstruierten Systems ist eine stärkere Forderung als die Unterbrechbarkeit einzelner Komponenten. Insbesondere impliziert sie, daß die zur Verfügung stehende Gesamtzeit nicht vor dem Start bekannt sein muß. Dieses a priori-Wissen steht in der Realität oft nicht oder zumindest nicht ausreichend zur Verfügung. Einzig das Konzept von Wang besitzt diese Eigenschaft.

Eine *Monitoring-Komponente* sollte eine eventuelle Adaption der Ressourcenverteilung auch zur Laufzeit des Gesamtsystems erlauben und somit eine Verbesserung der Performanz ermöglichen. Dies ist zwar bei allen Ansätzen vorgesehen, unterliegt aber teilweise starken Einschränkungen; so können z.B. bei Controlled Concurrency nur Parameter einzelner Prozesse verändert werden.

Ganz allgemein sollte eine Vorgabe der *Gesamtressourcen* nicht notwendig sein. Dies leistet nur Wangs Ansatz.

Die einzelnen Punkte dieses Vergleichs sind nicht notwendigerweise disjunkt, sondern können sich durchaus überschneiden. Die unterste Zeile stellt den Versuch einer Bewertung der einzelnen Verfahren dar: Wichtig ist die Tatsache, daß bisher kein Ansatz völlig überzeugen kann, da alle die eine oder andere Eigenschaft nicht besitzen, die als Forschungsziel wünschenswert ist. Dem trägt die letzte Spalte in Tab. 2.1 Rechnung. Die in dieser Arbeit in Kapitel 3 vorgestellte Anytime-System-Shell integriert in sich alle genannten positiven Aspekte: Sie ist in der Lage, sowohl Standard- als auch Anytime-Algorithmen zu verarbeiten, wobei sie Abhängigkeiten zwischen Teilaufgaben berücksichtigt und aus unterschiedlichen Lösungsstrategien selbständig auswählen kann. Das Gesamtsystem ist im Rahmen des Transaktionskonzeptes jederzeit unterbrechbar, ohne daß die verfügbaren Ressourcen zu Beginn bekannt sein müssen.

Die zugehörige Monitoring-Komponente kann somit zur Laufzeit durchgehend die Ressourcenverteilung kontrollieren und gegebenenfalls an sich verändernde Umweltbedingungen anpassen.

2.6 Anytime-Berechnung

Nachdem in den voranstehenden Abschnitten dieses Kapitels ein definitorischer Rahmen für das Thema *ressourcenbeschränkte Berechnung* gegeben und einige Forschungsansätze vorgestellt wurden, der Schwerpunkt aber eher auf der *Ressourcenbeschränkung* lag, soll nun die *Berechnung* in den Vordergrund treten. Zunächst erfolgt eine ausführlichere Beschreibung des Begriffs *Anytime-Algorithmus* und der dazugehörenden Konzepte. Daran schließt sich eine detaillierte Erläuterung von Zilbersteins Compilierungs-Verfahren (vgl. 2.5.3) an, das eine große Bedeutung für die Entwicklung der in Kapitel

3 vorgestellten Anytime-System-Shell hat.

War der Ausdruck *Ressource* bis hierher in der Regel nicht auf ein bestimmtes Hilfsmittel eingeschränkt, so wird nun – any *time* legt es nahe – nur die spezielle Ressource *Zeit* betrachtet.

2.6.1 Begriffe, Eigenschaften und Konstruktion

Zuerst soll die bereits in Abschnitt 2.5.2 erwähnte Unterscheidung in Standard- und Anytime-Algorithmen vertieft werden. Erstere entsprechen wie gesagt in ihrer sequentiellen Vorgehensweise einer Turing-Maschine (vgl. etwa (Lewis & Papdimitriou, 1981)), die eine Berechnung auf einer Eingabe ausführt und nach Abarbeitung aller Teilschritte ein im booleschen Sinne korrektes Ergebnis liefert, d.h. falsche Resultate entstehen durch fehlerhafte Algorithmen. Weder ist eine Unterbrechung möglich, noch kann auf etwaige Zwischenergebnisse zugegriffen werden. Abbildung 2.4(a) illustriert diese Arbeitsweise.

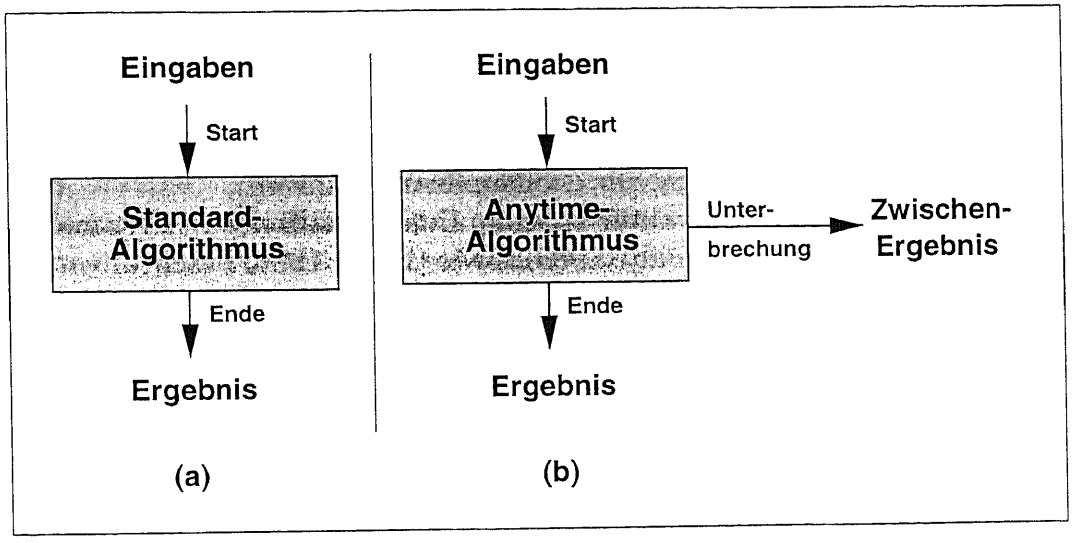

Abbildung 2.4: Algorithmentypen: Vergleich der prinzipiellen Arbeitsweisen

Im Gegensatz dazu *können* Anytime-Algorithmen jederzeit ohne nennenswerten Aufwand unterbrochen und – wenn gewünscht – weiter fortgesetzt werden. Ebenso stehen Zwischenergebnisse problemlos zur Verfügung (vgl. Abb. 2.4(b)). Eine *Korrektheit* der Ergebnisse ist allerdings nicht mehr im booleschen Sinne zu interpretieren: Neben dem eigentlichen Resultat muß noch die *Qualität* der bisher durchlaufenen Berechnungsverfahren angegeben werden. Dies dient quasi als Maß dafür, wie groß der Ressourcenverbrauch war und welche Güte diese Ressourcen hatten.

Die Qualität wird üblicherweise mit Werten aus dem Intervall $[0.0, 1.0]$ beschrieben. Sie darf nicht mit dem *Nutzen* verwechselt werden, der auch negativ werden kann, etwa wenn die Laufzeit einen kritischen Punkt überschritten hat und ein Resultat nicht mehr verwendet werden kann. Der Wert 0.0 steht für die Anfangs-Qualität einer Berechnung, die noch keinerlei Teilergebnis

generieren konnte. Der Wert 1.0, der einem optimalen Ergebnis zugeordnet ist, sollte erreicht werden, wenn die Ressourcenbeschränkungen nicht (mehr) relevant sind, also z.B. genug Zeit vorhandenden ist, um eine iterative Berechnung bis zum gewünschten Ende zu führen, und wenn insbesondere überhaupt ein Algorithmus implementiert ist, der ein optimales Resultat erzielen kann. (Umgekehrt kann bei einer Iteration *ohne* Beschränkung *das* optimale Ergebnis natürlich nie erreicht werden, da immer noch eine Verbesserung, so klein sie auch sein mag, möglich ist. Wie ein Wert für einen optimalen Nutzen zu bestimmen ist, hängt letztlich von der Problemstellung ab.)

Ein einfaches Beispiel für einen Anytime-Algorithmus ist das Newton-Verfahren zur Bestimmung einer Näherungslösung der Nullstelle einer differenzierbaren Funktion (vgl. (Königsberger, 1990)). Während bei einem Standard-Algorithmus als Abbruchkriterium eine Schranke für die zuletzt erreichte Verbesserung des Ergebnisses dienen würde, kann einem Anytime-Algorithmus eine bestimmte Zeitdauer zur Berechnung zugeteilt werden, nach deren Ablauf das bislang vorliegende beste Resultat abgefragt wird. Ein weiterer Vorteil besteht darin, daß die Auswertung an derselben Stelle wiederaufgenommen werden kann, falls etwa mehr Zeit als erwartet zur Verfügung steht. (Wird dies von der Meta-Kontrolle rechtzeitig erkannt, könnte die Unterbrechung einfach verschoben werden.) Die genannten Punkte können gegebenenfalls zu einer Verbesserung der Problemlösung führen.

Im folgenden sind die charakteristischen Eigenschaften von Anytime-Algorithmen, wie sie Dean und Boddy (1988) sowie Grass (1996) beschreiben, noch einmal zusammengestellt (vgl. auch (Boddy, 1991b, 1991a)).

- *Unterbrechbarkeit und Fortsetzbarkeit:* Ein Anytime-Algorithmus kann jederzeit mit vernachlässigbarem administrativen Aufwand unterbrochen und gegebenfalls wieder fortgesetzt werden. Ferner kann jederzeit das aktuell beste Resultat abgefragt werden.

- *Qualitätsmaß:* Das dem Ergebnis eines Anytime-Algorithmus' zugeordnete Qualitätsmaß spiegelt die Verläßlichkeit wieder, die das Resultat bietet (sei es einem menschlichen Anwender oder einem aufrufenden Programm) und ist u.a. abhängig von den eingesetzten Ressourcen, der Güte der Berechnungsverfahren und eventuell der Güte des Resultats selbst. Die Qualität des jeweils aktuellen Ergebnisses eines Anytime-Algorithmus' verhält sich monoton steigend und wird in Performanzprofilen gespeichert. Formal ist ein Performanzprofil eine Abbildung der Form

$$PP : \mathbb{R}_0^+ \longrightarrow [0,1];\ t \longmapsto PP(t), \tag{2.1}$$

  die einer Zeit t eine Qualität $Q = PP(t) \in [0,1]$ zuordnet.

- *Prognostizierbarkeit:* Die durchgängige Speicherung von Zwischenergebnissen mit den ihnen zugeordneten Qualitäten in Performanzprofilen hat eine Prognostizierbarkeit zur Folge, die Aussagen über die Qualität eines zukünftigen Resultats erlaubt. Darauf wird in einem gesonderten Abschnitt 2.6.2 näher eingegangen.

Diese Eigenschaften sind gleichzeitig Anforderungen, die ein Entwickler beim Entwurf von Anytime-Algorithmen berücksichtigen muß. Um diesen teilweise sehr komplexen Vorgang zu vereinfachen, führt Zilberstein (1993) das Konzept des *Contract-Anytime-Algorithmus* als Spezialfall ein. Dieser schränkt das Prinzip der jederzeitigen Unterbrechbarkeit insofern ein, als eine gewisse Laufzeit, während der *keine* Unterbrechung möglich und *kein* (Zwischen-)Ergebnis verfügbar ist, quasi *vertraglich* zugesichert wird. Zilberstein kann jedoch mit seinem Reduktionssatz zeigen, daß die eigentlich gewünschte Eigenschaft der jederzeitigen Unterbrechbarkeit durch eine Erhöhung der Laufzeit erkauft werden kann und somit Contract-Anytime-Algorithmen gleichmächtig wie *normale* Anytime-Algorithmen sind.

Satz 2.6 (Reduktionssatz von Zilberstein) *Zu jedem Contract-Anytime-Algorithmus CAA, dessen Qualität durch q_{CAA} beschrieben wird, kann ein (jederzeit) unterbrechbarer Anytime-Algorithmus AA mit q_{AA} konstruiert werden, so daß für jede feste Eingabe gilt:*

$$q_{AA}(4t) \geq q_{CAA}(t). \tag{2.2}$$

Zilberstein (1993, S. 38)

Der Satz basiert auf der Idee, einen jederzeit unterbrechbaren Anytime-Algorithmus A aus nacheinander stattfindenden Läufen des Contract-Anytime-Algorithmus' B mit exponentiell ansteigender Laufzeit zu konstruieren.

Die Notwendigkeit der Generierung von Zwischenergebnissen unterscheidet die Entwicklung von Standard- und Anytime-Algorithmen. Während bei einem *normalen* Berechnungsverfahren den Benutzer nur das Endergebnis interessiert, zwingt die Eigenschaft der Unterbrechbarkeit den Programmierer dazu, auch eine Anzahl von (sinnvollen) Teilresultaten zu erzeugen. Dies geschieht meist am besten durch eine Zerlegung des Gesamtproblems in Teilaufgaben. Zilberstein (1993) bezeichnet diese Art der Entwicklung als *indirekte Programmiertechnik der Anytime-Algorithmen*, die nicht nur das Endresultat zum Ziel hat. Zusammen mit Grass schlägt er vor, innerhalb des jeweiligen Algorithmus' einen *Verbesserungsschritt* hervorzuheben, d.h. jenen Teil, der die eigentliche Arbeit erledigt (vgl. (Grass & Zilberstein, 1995)). Abbildung 2.5 zeigt den Pseudocode eines solchen Anytime-Algorithmus'.

```
    Anytime-Prototyp(Eingaben)
(1) Ergebnis ← initiale_Lösung(Eingaben)
(2) lege_ab(Ergebnis)
(3) while not Unterbrechung and not Abbruchkriterium do
(4)     Ergebnis ← Verbesserungsschritt(Ergebnis)
(5)     lege_ab(Ergebnis)
(6) od
```

Abbildung 2.5: Konstruktion eines Anytime-Algorithmus'

Die Bereitstellung einer Initiallösung stellt sicher, daß jederzeit ein Resultat, hier natürlich von geringer Qualität, zurückgeliefert werden kann. Danach beginnt die Iteration, die außer dem Verbesserungsschritt noch die Aktualisierung des Ergebnisses enthält. Diese Schleife wird entweder extern durch eine Unterbrechung oder durch ein *normales* Abbruchkriterium beendet. Für die Implementation des Verbesserungsschrittes können beliebige Verfahren verwendet werden, wie z.B. Tiefen- oder Breitensuche. Aber auch frei parametrisierbare Subalgorithmen, die völlig unterschiedlich arbeiten, sind möglich. Iterative Verfahren haben den Vorteil, daß sie kaum verändert werden müssen, da sie schon unterbrechbar sind.

Nach diesen grundlegenden Anmerkungen zu Anytime-Algorithmen wird im folgenden Abschnitt das Thema der Zuordnung von Qualitätsbewertungen zu (Zwischen-)Ergebnissen vertieft.

2.6.2 Qualitätsmaß und Performanzprofil

Ein entscheidender Aspekt bei der Verwendung von Anytime-Algorithmen liegt in der Verknüpfung von erreichten (Teil-)Resultaten mit Qualitäten. Nachdem die Idee an sich bereits im vorigen Abschnitt erläutert wurde, stehen nun die Faktoren, die ein solches Qualitätsmaß beeinflussen, und die Verarbeitung von Performanzprofilen im Mittelpunkt.

Für die Kombination eines Ergebnisses mit einem die Qualität beschreibenden Wert aus dem Intervall $[0, 1]$ gibt es keine allgemeingültige Regel. Vielmehr richtet sich allein schon die Wahl der zu verwendenden *Qualitätsmetrik(en)* in besonderen Maße nach der jeweiligen Anwendung. Gleiches gilt für Abstufungen innerhalb einer Metrik oder bei der Gewichtung von Kombinationen mehrerer Metriken. Insgesamt besitzt der jeweilige Entwickler hier einen großen Freiraum, der die Möglichkeit einer exakten Modellierung der Bewertungen bietet.

Prinzipiell sind beliebige Qualitätsmetriken denkbar; in der Praxis sind solche gebräuchlich, die konkrete Eigenschaften des Problems oder des Algorithmus' als Basis haben. Grundvoraussetzung ist in jedem Fall aber eine monotone Zunahme der Qualität bei erhöhtem Ressourceneinsatz. Zilberstein und Russell (1996) nennen folgende drei Typen von Qualitätsmetriken:

- *Sicherheit:* Diese Metrik bewertet ein Ergebnis bezüglich der Wahrscheinlichkeit, daß es korrekt ist.
- *Genauigkeit:* Hier wird die Abweichung des bislang approximierten zum exakten Resultat berücksichtigt. Beispielsweise könnte eine bestimmte Rechengenauigkeit verlangt sein.
- *Spezifität:* Bei dieser Metrik wird der Detaillierungsgrad des Ergebnisses herangezogen. So steigt etwa beim hierarchischen Planen die Qualität, je mehr ein abstrakter Plan konkretisiert wird.

Diese Liste kann nach Wahlster noch erweitert werden[3]:

[3] Als Quelle diente ein Vortrag von Prof. W. Wahlster.

- *Allgemeinheit:* Bei Aufgaben wie dem Problemlösen, ist eine verallgemeinerbare Lösung einer Speziallösung vorzuziehen.
- *Vertrauen:* Das Vertrauen in das erzielte Resultat nimmt mit den eingesetzten Ressourcen zu. Ein klassisches Beispiel hierzu sind experimentelle Versuchsreihen.

Stellt man die Qualitätsentwicklung eines Anytime-Algorithmus' in ihrer zeitlichen Abhängigkeit dar, so erhält man ein Performanzprofil nach (Dean & Boddy, 1988). In Abb. 2.6 ist neben dem Performanzprofil eines Standard-Algorithmus' (a) und eines idealisierten Anytime-Algorithmus' (b), mit einer starken Steigung zu Beginn und geringerer Verbesserung gegen Ende, auch der realistische Verlauf in Treppenform abgebildet (c), mit einem stufenweisen Anstieg der Qualität, jedesmal wenn eine einzelne Teilaufgabe gelöst wurde. Die Speicherung von Performanzprofilen kann in diesem Fall am einfachsten durch die Angabe der relevanten Stützstellen erfolgen. Ist deren Anzahl sehr groß – und somit auch die entsprechende Datenmenge – werden häufig werden Näherungslösungen verwendet, z.B. durch lineare (abschnittsweise) Approximation.

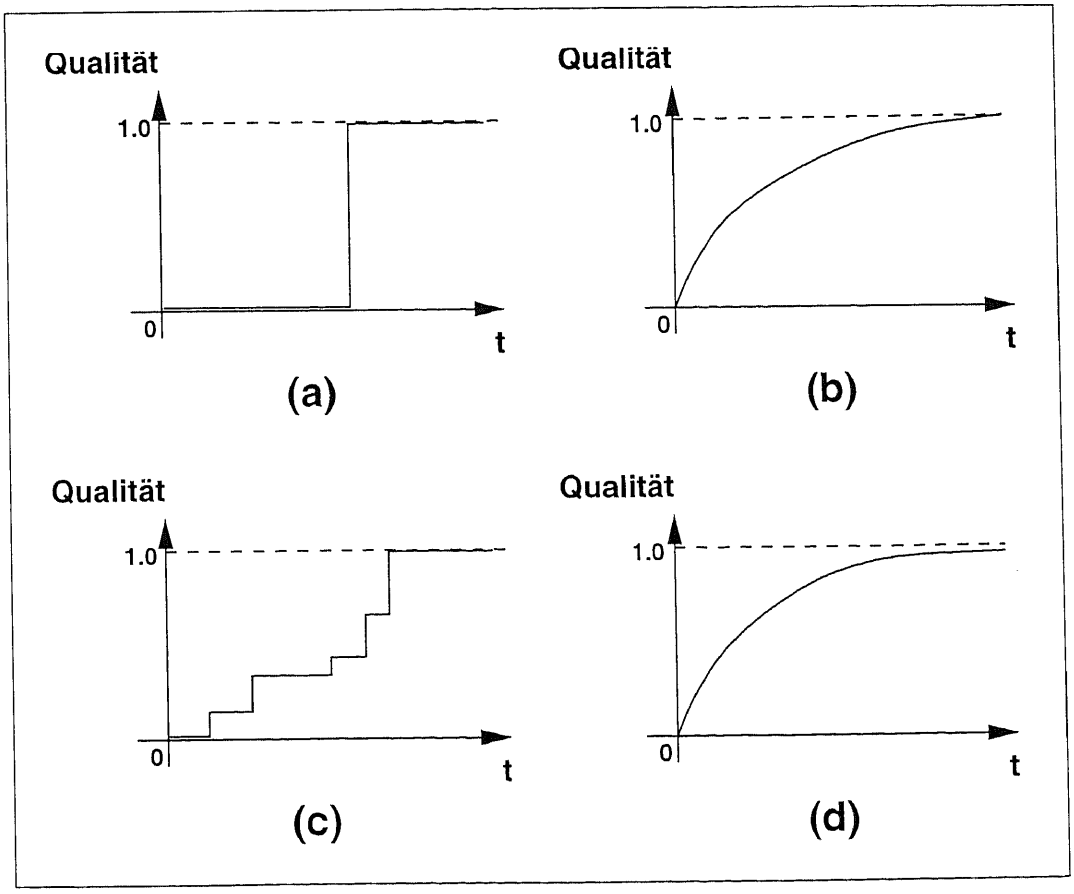

Abbildung 2.6: Performanzprofile unterschiedlicher Algorithmen(-typen)

Dean und Boddy schlagen die Repräsentation von Performanzprofilen gegebenenfalls durch Annäherung an eine Exponentialfunktion der Art $f(x) =$

$1 - e^{-\lambda x}$ vor (vgl. Abb. 2.6d). Diese Funktionsklasse ähnelt dem idealisierten Verlauf und ist monoton steigend, hat aber den Nachteil, den maximalen Wert 1.0 nicht zu erreichen. Gerade deshalb lassen sich jedoch Näherungsverfahren wie das von Newton, die selbst die optimale Lösung nicht erreichen können, durch diese Funktionsklasse sehr gut approximieren.

Die beschriebenen Näherungsverfahren zur Repräsentation von Performanzprofilen spielen insbesondere dann eine gewichtige Rolle, wenn mehrere Durchläufe desselben Algorithmus' verarbeitet werden sollen, um beispielsweise bei der Bewertung des Berechnungsverfahrens als Ganzes von unterschiedlichen Eingaben oder Umweltbedingungen abstrahieren zu können.

2.6.3 Konstruktion von Anytime-Systemen

Zur Lösung der komplexen Probleme auf dem Gebiet der Künstlichen Intelligenz, insbesondere unter dem kognitionswissenschaftlichen Gesichtspunkt und der Postulierung von ressourcenbeschränkter Berechnung, ist die Kombination von Anytime-Algorithmen zu ganzen Anytime-Systemen sehr interessant, erlaubt sie doch eine parallele Verarbeitung von Teilaufgaben bei gleichzeitiger inkrementeller Verbesserung des Endresultats. Dies kommt beispielsweise den Annahmen über das menschliche Gehirn recht nahe. Die Kombination von Anytime-Algorithmen führt direkt zur Forderung nach beschränkter Optimalität (vgl. Abschnitt 2.3): Eine zur Verfügung stehende Gesamtzeit soll so eingesetzt – also auf die beteiligten Algorithmen verteilt – werden, daß das Endresultat optimal ist, ja es wird sogar eine *jederzeitige* Optimalität angestrebt, denn die Anytime-Eigenschaft der Unterbrechbarkeit soll selbstverständlich auch für das zu konstruierende Gesamtsystem gelten.

Die *Compilierung von Anytime-Algorithmen nach Zilberstein* (vgl. (Zilberstein, 1993)) stellt einen ersten Ansatz zur Konstruktion solcher Systeme dar. Nachdem er in Abschnitt 2.5.3 kurz vorgestellt wurde, soll seine Arbeitsweise nun detaillierter erläutert werden, da die im Rahmen der vorliegenden Arbeit entwickelte Anytime-System-Shell (vgl. Abschnitt 3) eine Reihe von Zilbersteins Ideen verwendet.

Die von Zilberstein vorgeschlagene *Offline-Compilierung* verknüpft einzelne Komponenten zu einem Gesamtsystem. Nachteilig ist, daß zu Beginn die zur Verfügung stehende Zeit bekannt sein muß. Dadurch kann Zilberstein zwar beschränkte Optimalität erreichen, verzichtet aber auf eine jederzeitige Unterbrechbarkeit, die er nur bei der Kombination der Einzelalgorithmen nutzt.

Abbildung 2.7 (nach (Zilberstein, 1993, S. 56)) zeigt, wie der *Compiler* aus einem *zusammengesetzten Anytime-Modul*, das noch keine Informationen über die Ressourcenzuteilung enthält, und den zu diesen Algorithmen gehörenden Performanzprofilen, die in einer Art Bibliothek gespeichert sind, ein aus drei Teilen bestehendes *ausführbares Anytime-Modul* erzeugt. Das eigentliche Programm, das sogenannte *compilierte zusammengesetzte Anytime-Modul* wird von einer *Monitoring-Komponente* kontrolliert. Dazu wurde aus den Einzelperformanzprofilen und der Ressourcenverteilung ein *Gesamtper-*

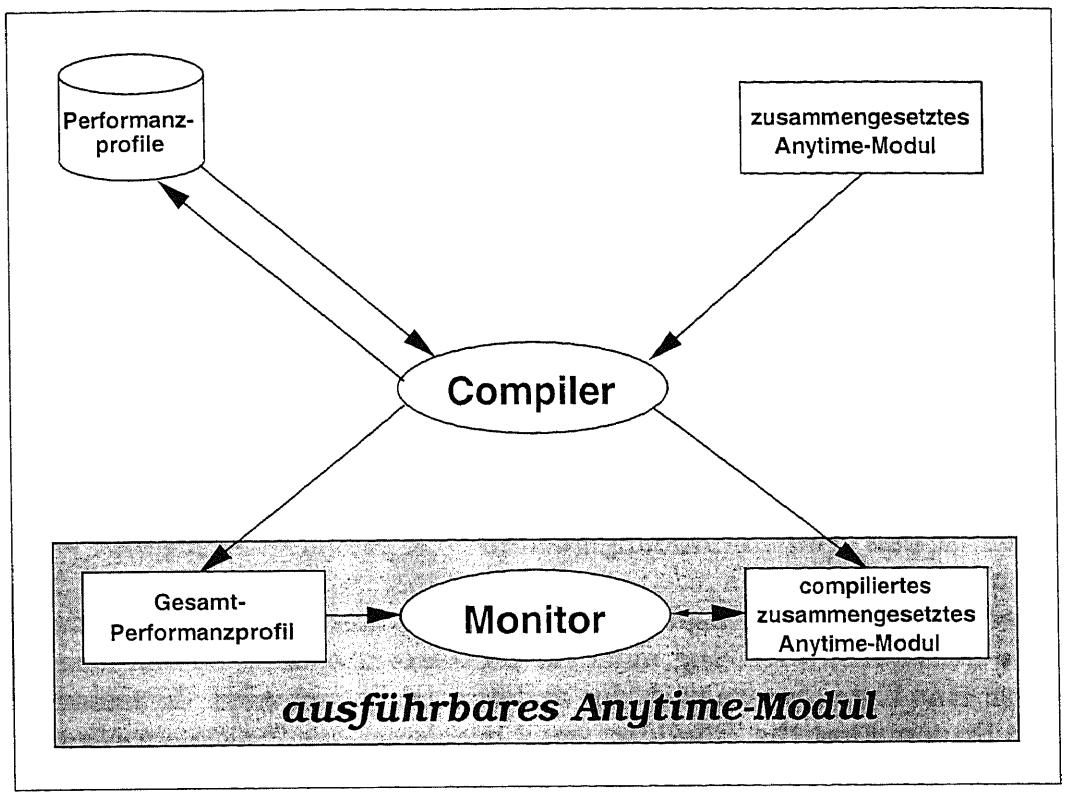

Abbildung 2.7: Compilierungs- und Monitoring-Architektur von Anytime-Algorithmen nach Zilberstein

formanzprofil bestimmt, anhand dessen der Monitor über Aktivierung und Deaktivierung der einzelnen Subalgorithmen entscheidet, um ein optimales Verhalten des Gesamtsystems zu erzielen.

Um überhaupt eine Kombination von Anytime-Algorithmen vornehmen zu können, muß der Compiler wissen, welche Abhängigkeiten die einzelnen Berechnungsverfahren voneinander haben. Prinzipiell werden drei Varianten unterschieden: sequentielles, paralleles oder rekursives Verhältnis der Algorithmen zueinander. Im ersten Fall nutzt ein Algorithmus das Ergebnis eines anderen, ist also abhängig davon, insbesondere wird sein Performanzprofil zu einen bedingten Performanzprofil (vgl. Abschnitt 2.5.3). Die Verbindung zweier parallel verlaufender Anytime-Algorithmen ist ein Spezialfall der sequentiellen Verknüpfung bei der keine Resultatabhängigkeit vorliegt; die Reihenfolge ist somit beliebig. Die rekursive Variante ist unproblematisch, das Berechnungsverfahren ruft sich selbst wieder auf.

Bei der sequentiellen Kombination muß nun eine Verknüpfungsart festgelegt werden, die angibt, in welchem Verhältnis die Qualitäten der beteiligten Algorithmen zueinander stehen. Dies ist im wesentlichen anwendungsspezifisch. Neben Addition und Multiplikation sind beliebige Operatoren anwendbar, solange sie die Monotonie der Performanzprofile bewahren. Das Resultat der Verknüpfung von n Anytime-Algorithmen ist dann eine Funktion mit n Variablen (für jeden Algorithmus eine Zeitzuteilung). Die optimale Ressourcenverteilung kann jetzt durch die Bestimmung des globalen Maximums der

Gesamtqualität, d.h. durch Ableiten besagter Funktion, berechnet werden. Dazu muß ein System mit $n-1$ Gleichungen gelöst werden[4].

In der Regel ist die Annäherung von Performanzprofile durch stetige Funktionen aber kritisch, da eine genauere Repräsentation der Qualitätsentwicklung erforderlich ist. Für die präzisere Speicherung in Tabellenform zeigt Zilberstein (1993) jedoch, daß das zugehörige Suchproblem NP-vollständig ist. Gleiches gilt insbesondere für eine Darstellung der Performanz durch eine Treppenstufenfunktion, die sich bei der Generierung der Profile häufig ergibt. Ein weiterer Nachteil dieser Art der Ressourcenverteilung besteht darin, daß sie sozusagen *global* zuteilt, d.h. alle Verfahren erhalten ihre Quoten von einer zentralen Stelle, alle Quoten sind aber auch von allen Verfahren abhängig. In vielen Fällen, z.B. bei den hier betrachteten hochkomplexen Systemen, erscheint es aber angebrachter, eine dezentrale Verteilung vorzunehmen, so daß etwa Un-/Abhängigkeiten sehr genau widergegeben werden können.

Dies kann durch den Ansatz der *lokalen Compilierung* erreicht werden, bei der die Ressourcenverteilung nicht für das Gesamtsystem *en bloc* ermittelt, sondern eine Komponente nach der anderen analysiert wird. In einem komplexen System kann die Zuteilung somit problemnäher für die einzelnen Teilaufgaben gelöst werden. Erkauft wird dieser Vorteil allerdings dadurch, daß die Optimalität der Gesamtverteilung nicht mehr garantiert werden kann wie bei der globalen Compilierung. Ein Beispiel hierfür ist das wiederholte Auftreten von Subkomponenten, wie es in Abb. 2.8(a) dargestellt wird. Dabei benötigen zwei oder mehr Berechnungsverfahren das Ergebnis ein und desselben Subalgorithmus', das aber doppelt berechnet wird. In diesem Fall würde die lokale Compilierung zu einer stark überhöhten, eventuell doppelten Ressourcenzuteilung führen, was nicht ohne weiteres gerechtfertigt ist.

Abhilfe schaffen könnte eine Identifizierung solcher Subkomponenten (vgl. Abb. 2.8(b)): Dabei existierte der betroffene Subalgorithmus tatsächlich nur einmal, alle übergeordneten Verfahren erhielten das Resultat von ihm. Die Abbildung macht aber auch deutlich, daß nach einer Identifizierung keine Unterteilung in Subkomponenten, wie sie eigentlich für die lokale Compilierung erforderlich ist, mehr möglich ist. Diese erfordert somit, um effizient zu sein, eine Problemstruktur, die sich als Baum darstellen läßt, insbesondere dürfen keine Zyklen auftreten. Ist diese Hauptvoraussetzung erfüllt, zeigt Zilberstein neben der Optimalität des Verfahrens der lokalen Compilierung auch daß die Ressourcenallokation in pseudo-polynomialer Zeit erfolgen kann[5].

Das Problem der mehrfach auftretenden Subkomponenten ist allerdings nicht grundsätzlich auszuschließen, bzw. der zu betreibende Aufwand wäre zu groß. Daher beschreibt Zilberstein drei approximative Verfahren, die sowohl die angeführten Nachteile der globalen als auch die Probleme der lokalen Compilierung umgehen.

[4]Auf alternative Verfahren wird in Abschnitt 3.4.2 näher eingegangen.

[5]Weitere Bedingungen sind die Beschränkung der Eingaben in die Komponenten auf eine kleine Konstante und die Monotonie der Eingabequalität, d.h. eine Inputverbesserung bedingt eine Outputverbesserung.

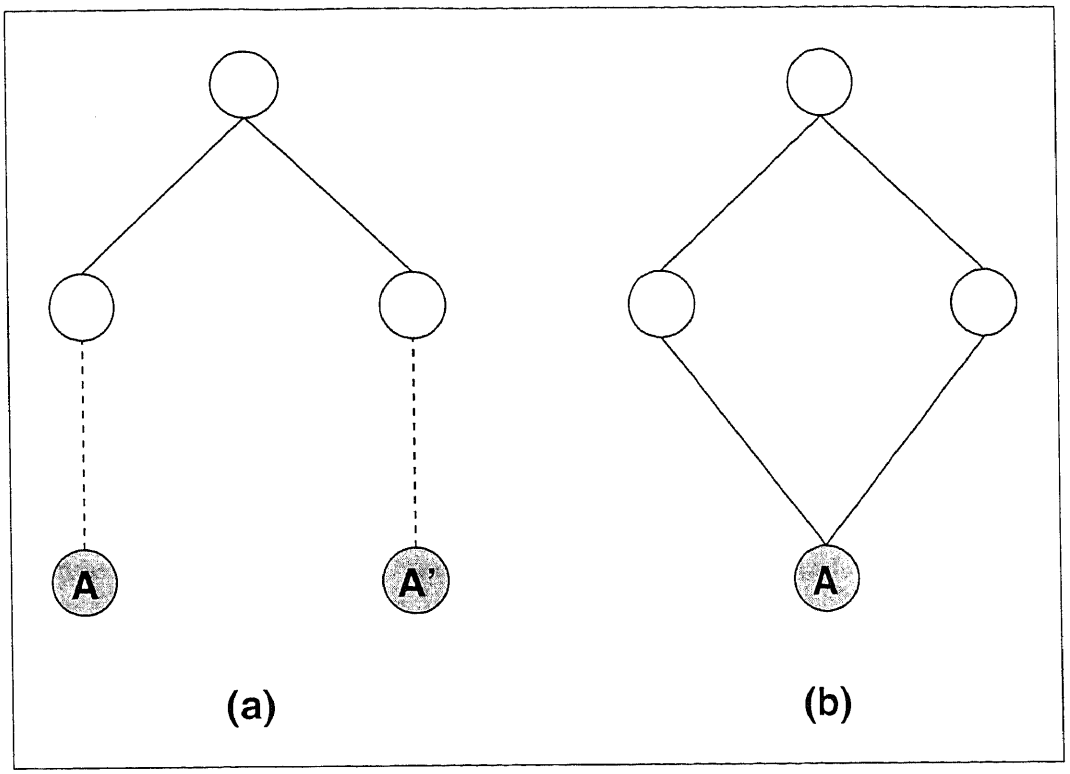

Abbildung 2.8: Mehrfach auftretende Subkomponenten

Der erste Ansatz verwendet einen Hill-climbing-Algorithmus, der ausgehend von einer Gleichverteilung auf die beteiligten Komponenten solange Ressourcen zwischen jeweils zwei Modulen austauscht, wie eine Verbesserung des Endresultats erreicht werden kann. Hierbei handelt es sich also um ein approximatives globales Verfahren. Wie bei allen Hill-climbing-Algorithmen kann ein Erreichen der optimalen Lösung nicht garantiert werden.

Das zweite Verfahren modifiziert die lokale Compilierung dahingehend, daß die Ressourcen für mehrfach auftretende Subkomponenten zuerst gesondert festgelegt werden. Danach läuft die normale lokale Compilierung ab, die die verbliebenen Ressourcen auf die restlichen Module verteilt.

Schließlich schlägt Zilberstein ein Verfahren vor, das die Ressourcenverteilung auf mehrfach auftretende Subkomponenten durch Anpassung der Performanzprofile über eine Anzahl von Testdurchläufen *lernt*.

Zum Abschluß der Vorstellung von Zilbersteins Compilierung von Anytime-Algorithmen muß noch die Monitoring-Komponente diskutiert werden, die in dem vom Compiler erzeugten ausführbaren Anytime-Modul die Funktion einer Kontrollinstanz ausübt. Zilberstein unterscheidet grundsätzlich *passives Monitoring*, bei dem alle Ressourcenzuteilungen vor Beginn des Systemlaufs erfolgt sind, von *aktivem Monitoring*; hier können auch zur Laufzeit noch Veränderungen an der Ressourcenallokation vorgenommen werden. Dies ist hinsichtlich der Unsicherheit bezüglich Faktoren wie Umwelt- oder auch interner Systemzustand in der Regel erforderlich. Insbesondere repräsentieren

die Performanzprofile kein absolutes Verhalten, sondern Meßwerte, die, da nie exakt gleiche Voraussetzungen vorliegen, variieren können. Aktives Monitoring verwendet aus diesen Gründen eine *zeitabhängige Nutzenfunktion* (vgl. (Russell & Wefald, 1991)), wie sie in Abschnitt 2.4 erläutert wurde. Diese Nutzenfunktion ist anwendungsabhängig und muß für jede Problemstellung erarbeitet werden.

Definition 2.7 (Passives Monitoring) *Wenn die gesamte Ressourcenallokation vor dem Start eines problemlösenden Systems erfolgt, spricht man von* passivem Monitoring.

Definition 2.8 (Aktives Monitoring) *Wenn Ressourcenallokation auch zur Laufzeit eines problemlösenden Systems erfolgen kann, spricht man von* aktivem Monitoring.

Ein Beispiel für passives Monitoring ist die *fixed-contract* Strategie, die sich besonders für Systeme eignet, die aus Contract-Anytime-Algorithmen bestehen. Sie besitzen Performanzprofile, die jeweils nur geringfügig variieren, so daß Korrekturen eher unnötig sind. Im Falle des aktiven Monitoring könnte bei Contract-Anytime-Algorithmen nach Ende der *Vertragslaufzeit* – und nur dann – eingegriffen und gegebenenfalls anhand neuer Informationen, wie z.B. veränderter Umgebungsvariablen, abweichender aktueller Performanzprofile etc. die Ressourcenallokation angepaßt werden, indem z.B. noch nicht beendete Komponenten Restkapazitäten erhalten. Alternativ ist es denkbar, das Verhältnis der Ressourcenzuteilungen zu modifizieren. Im Gegensatz zu Contract-Anytime-Algorithmen kann bei jederzeit unterbrechbaren Anytime-Systemen die Ressourcenverteilung zu jedem Zeitpunkt optimiert werden. Hier kann eine Qualitäts- oder eine Nutzenfunktion als Indikator für die Verteilung dienen.

Die in diesem Abschnitt vorgestellten Vor- und Nachteile der einzelnen Verfahren erheben keinen Anspruch auf Vollständigkeit, sondern sollen den Blick schärfen für die Problematik einer optimalen Ressourcenallokation. Die einzelnen Algorithmen sind in der angegebenen Literatur ausführlich dargestellt.

2.7 Zusammenfassung

Im Mittelpunkt dieses Kapitel stand die ressourcenbeschränkte Berechnung: nach einer ausführlichen Definition von Begriffen und Konzepten erfolgte die Beschreibung des Problems, eine optimale Verteilung von Ressourcen zu erzielen.

Dazu wurde eine Reihe von Ansätzen vorgestellt, die – teilweise aufeinander aufbauend – verschiedene Strategien zur Lösung dieser Problematik anbieten. Die Idee der Compilierung von Anytime-Algorithmen von Shlomo

Zilberstein erlaubt erstmals die Konstruktion ganzer unterbrechbarer Systeme. Eine Reihe der dort vorgestellten Ideen konnte in dem dieser Arbeit zugrunde liegenden System verwendet werden. Dennoch wurde deutlich gemacht, daß insbesondere für komplexe Aufgaben aus den Gebieten der Künstlichen Intelligenz und der Kognitionswissenschaft noch Entwicklungsbedarf besteht. Problematisch ist beispielsweise die teilweise notwendige Kenntnis der verfügbaren Zeit – etwas, das in der Realität nur selten möglich ist.

Anhand der vorhandenen Ansätze wurde erörtert, welche Fähigkeiten ein zu entwickelndes System haben muß, um in der Lage zu sein, ressourcenadaptierend zu agieren. Hier sind insbesondere die freie Wahl des Algorithmentyps (Standard oder Anytime), die – eventuell parallel durchgeführte – Erprobung unterschiedlicher Lösungsstrategien sowie eine jederzeitige Unterbrechbarkeit des Gesamtsystems (bei Unkenntnis über die zur Verfügung stehende Zeit) zu nennen.

Der nun folgende Teil III dieser Arbeit ist der praktischen Anwendung und der Vorstellung des beschränkt-optimalen Lokalisationsagenten BOLA gewidmet. Die diesem System zugrundeliegende ressourcensensitive Architektur wird in Kap. 3 präsentiert. Sie vermeidet die Defizite der bestehenden Ansätze indem sie die Forderung nach den oben genannten Fähigkeiten erfüllt und so einem eingebetteten System die Eigenschaft der beschränkten Optimalität ermöglicht. Das darauffolgenden Kapitel 4 behandelt die konkrete Konstruktion eines ressourcenadaptierenden Lokalisationsagenten ausgehend von dieser Architektur.

Teil III

Ein beschränkt-optimaler Lokalisationsagent

Kapitel 3

Eine ressourcensensitive Architektur durch eine Anytime-System-Shell

Aufbauend auf den im vorangegangenen Kapitel 2 vorgestellten Grundlagen und Anforderungen der ressourcenbeschränkten Berechnung wird nun eine ressourcensensitive Architektur entwickelt, die im folgenden Kapitel 4 die Basis des beschränkt-optimalen Lokalisationsagenten BOLA bildet. Sie ermöglicht entsprechend konzipierten Systemen die Eigenschaft der beschränkten Optimalität in Verbindung mit jederzeitiger Unterbrechbarkeit.

Als direkte Folge des streng modularen Aufbaus des späteren Gesamtsystems zeichnet sich diese Architektur ferner durch ihren *rekursiven Shell-Charakter* aus, d.h. sie ist offen, prinzipiell beliebige *Gastsysteme* oder miteinander kommunizierende modulare Subsysteme aufzunehmen und zu vollwertigen Realzeit-Systemen zu erweitern[1].

3.1 Grundlagen

3.1.1 Eigenschaften

In diesem Abschnitt werden die dem Konzept der Anytime-System-Shell zugrunde liegenden Ideen sowie ihre charakteristischen Eigenschaften vorgestellt und im Vergleich mit den Ansätzen aus Kapitel 2.5 diskutiert.

3.1.1.1 Unabhängiges Ressourcenmanagement

Bei der Entwicklung der Anytime-System-Shell stand neben der jederzeitigen Unterbrechbarkeit insbesondere der Gedanke im Vordergrund, die Ressourcenverteilung mit allen daran beteiligten Komponenten völlig von der jeweiligen Problemstellung und -lösung zu trennen. Wir werden im folgenden

[1]Diese Architektur wurde in Zusammenarbeit mit Frank Wittig entwickelt und von ihm im Rahmen einer Diplomarbeit innerhalb des Systems JAMES (Java Anytime Management & Editor System) implementiert (vgl. (Wittig, 1998)).

sehen, daß – und wie – dies gelungen ist. Diese Trennung ermöglicht eine *universelle Einsetzbarkeit* der Shell.

Definition 3.1 (Anytime-System-Shell) *Eine* Anytime-System-Shell, *stellt Werkzeuge zum Ressourcenmanagement beliebiger Systeme zur Verfügung.*

Das die eigentliche Aufgabenstellung lösende Programmpaket wird im folgenden als *Gastsystem* bezeichnet. Es umfaßt (alle) Lösungsstrategien und Daten, die zu einer erfolgreichen Bearbeitung einer entsprechenden Anfrage erforderlich sind.

Definition 3.2 (Gastsystem) *Ein Programmpaket, das Strategien und Daten zur Abarbeitung einer bestimmten Aufgabe enthält, heißt* Gastsystem.

Um in die Anytime-System-Shell, die das komplette Ressourcenmanagement leistet, eingebettet werden zu können, muß das Gastsystem – abgesehen von der Integration einer entsprechenden Schnittstelle – lediglich bestimmte Anforderungen erfüllen, wie z.B. die Bereitstellung von Zwischenergebnissen (vgl. Abschnitt 3.1.3). Eine grundlegende, in der Regel sehr (zeit-)aufwendige Neuentwicklung ist jedoch nicht notwendig. Als Ergebnis erhält der Benutzer ein Realzeit-System, das *Gesamtsystem*, das sich aus Anytime-Management und Gastsystem zusammensetzt und über die gewünschten Anytime-Eigenschaften verfügt.

Definition 3.3 (Gesamtsystem) *Als* Gesamtsystem *wird das resultierende System nach der Einbettung eines Gastsystem in eine Anytime-System-Shell bezeichnet. Das Verhalten des entstandenen Realzeit-Systems ist ressourcenadaptierend.*

Die Anytime-System-Shell stellt für ein Gesamtsystem eine *flexible Architektur* zur Verfügung, die es z.B. erlaubt, ein bereits eingebundenes Gastsystem um weitere Lösungsstrategien zu erweitern. Durch dieses Baukastenprinzip, bei dem auch ganze Subsysteme integriert werden können (vgl. Abb. 3.1), ist sowohl die Konstruktion als auch das Austesten großer Realzeitsysteme stark vereinfacht.

Ein *Subsystem* ist eine in sich abgeschlossene Komponente eines Gastsystems, die – analog zu diesem – auf alle zur Lösung einer bestimmten Teilaufgabe benötigten Strategien und Daten Zugriff hat. Es sind ganze Bibliotheken von Subsystemen denkbar, aus denen sich nach Bedarf komplexe Anytime-Systeme zusammensetzen lassen.

Definition 3.4 (Subsystem) *Eine in sich abgeschlossene Komponente eines Gastsystems, die in der Regel einen eigenen Aufgabenbereich bearbeitet, heißt* Subsystem. *Das Verhalten des entstandenen Realzeit-Systems ist ressourcenadaptierend.*

In der vorliegenden Arbeit wurde z.B. die Komponente zur Berechnung räumlicher Relationen, selbst ein komplettes Anytime-System, in einen jederzeit unterbrechbaren Lokalisationsagenten eingebaut (vgl. Abschnitt 4.2).

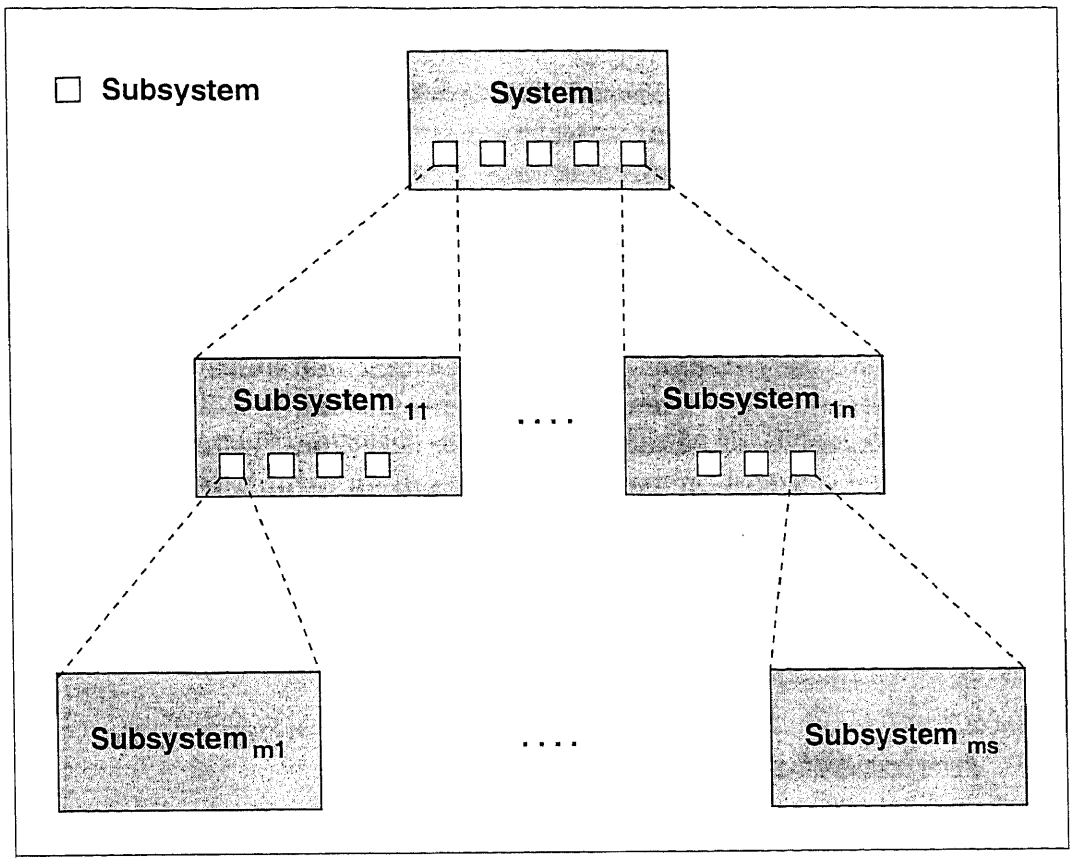

Abbildung 3.1: Flexible Architektur für erweiterbare Gastsysteme

3.1.1.2 Ressourcenadaptierendes Verhalten

Die oben beschriebene Möglichkeit, eine beliebige Anzahl unterschiedlicher Lösungsstrategien für ein und dieselbe (Teil-)Aufgabe in ein Gesamtsystem zu integrieren, ermöglicht diesem ein *ressourcenadaptierendes Verhalten*, wie es in Abschnitt 2.2 erläutert wurde. Dazu verwendet die Meta-Kontrolle die in den einzelnen Durchläufen gesammelten Performanzdaten zur Ressourcenverteilung auf die verschiedenen Lösungsvarianten. Je nach vorhandener Zeit ist diese Verteilung nicht nur absolut sondern auch relativ unterschiedlich, d.h. die eine Strategie wird bei viel vorhandener Zeit präferiert, bei weniger Ressourcen eine andere.

Dies entspricht exakt dem *ressourcenadaptierenden Verhalten*, wie es in Abschnitt 2.2 beschrieben wurde: Entscheidungen über Strategien und Strategiewechsel werden zur Laufzeit aufgrund der aktuellen Ressourcenbeschränkungen getroffen. Das Ergebnis ist ein Anytime-System, das die vorhandenen Ressourcen beschränkt-optimal nutzt. Neben den bereits vorgestellten Verfahren zur Ressourcenverteilung anhand der Optimierung über Performanzprofile (vgl. Abschnitt 2.6.3) werden in Abschnitt 3.4.2 eine Reihe neu entwickelter Allokationsmöglichkeiten beschrieben, die auf diesen Daten aufbauen.

3.1.1.3 Unterbrechbarkeit und Transaktionen

Gesamtsysteme, also problemlösende Gastsysteme, die in die Anytime-Shell integriert wurden, sind jederzeit unterbrechbar. Insbesondere müssen vor Beginn des Systemlaufs keine Informationen über die zur Verfügung stehende Zeit o.ä. vorliegen. Dies gereicht dem vorliegenden Ansatz zum Vorteil gegenüber Garvey und Lessers Design-to-time-Scheduling (vgl. Abschnitt 2.5.4) oder Zilbersteins Compilierung von Anytime-Algorithmen (vgl. Abschnitt 2.5.3), die auch bei aktivem Monitoring hier ihre Schwäche hat.

Die Eigenschaft der jederzeitigen Unterbrechbarkeit sollte jedoch bei komplexen (symbolischen) Systemen durch das in Abschnitt 2.5.2 beschriebene Transaktionskonzept beschränkt werden. Die Anytime-System-Shell ermöglicht deshalb die Verwendung von Transaktionen innerhalb des Gastsystems. Gerade bei den angesprochenen komplexen Systemen, die kognitive Fähigkeiten modellieren, wie z.B. gleichzeitiges Erkennen einer Objektkonstellation und Beschreiben derselben, ist dies relevant. Man könnte sogar so weit gehen, die Ununterbrechbarkeit von Transaktionen mit der Reaktionszeit des kognitiven Systems *Mensch* zu vergleichen[2].

3.1.2 Architektur

Die Architektur eines beliebigen Gesamtsystems, bestehend aus Anytime-System-Shell und Gastsystem, ist in in Abb. 3.2 dargestellt.

Hauptbestandteil ist die *Ressourcenadaptierende Modul-Instanz (RAMI)*, die das Gastsystem und die zugehörige Meta-Kontrolle, hier Anytime-Kontrolle genannt, enthält. Letztere ist verantwortlich für die Ressourcenverteilung und hat Zugriff auf die Bibliothek der zur Problemlösungskomponente gehörenden Performanzprofile.

Definition 3.5 (RAMI) *Ein Modul eines Gesamtsystems, das ein komplexes Subsystem repräsentiert, heißt* Ressourcenadaptierende Modul-Instanz (RAMI). *Jede RAMI besteht aus einem Gastsystem und einer Anytime-Kontrolle. (Die Anytime-System-Shell ist selbst auch eine RAMI.)*

Definition 3.6 (Anytime-Kontrolle) *Eine* Anytime-Kontrolle *steuert das Ressourcenmanagement innerhalb einer RAMI für das eingebettete Gastsystem.*

Das Gastsystem selbst zerfällt in eine Anzahl von *Modulen*, die bestimmte Teilaufgaben bearbeiten, und eine *Modul-Kontrolle*, die die Koordinierung zwischen diesen Modulen übernimmt; ferner ist eine adäquate Wissensbasis angegliedert.

[2]Transaktionen werden in diesem beispielhaft betrachteten System zur Objektlokalisation zur Kapselung von kritischen Manipulationen an den Objekten genutzt. Beispielsweise werden im Rahmen der Relationenbestimmung Drehungen der beteiligten Objekte vorgenommen. Diese müssen so in Transaktionen gekapselt werden, daß nach deren Beendigung der ursprüngliche Zustand wiederhergestellt ist. Damit wird vermieden, daß andere nebenläufig stattfindende Berechnungen ebenfalls die manipulierten Objekte benutzten und falsche Ergebnisse liefern.

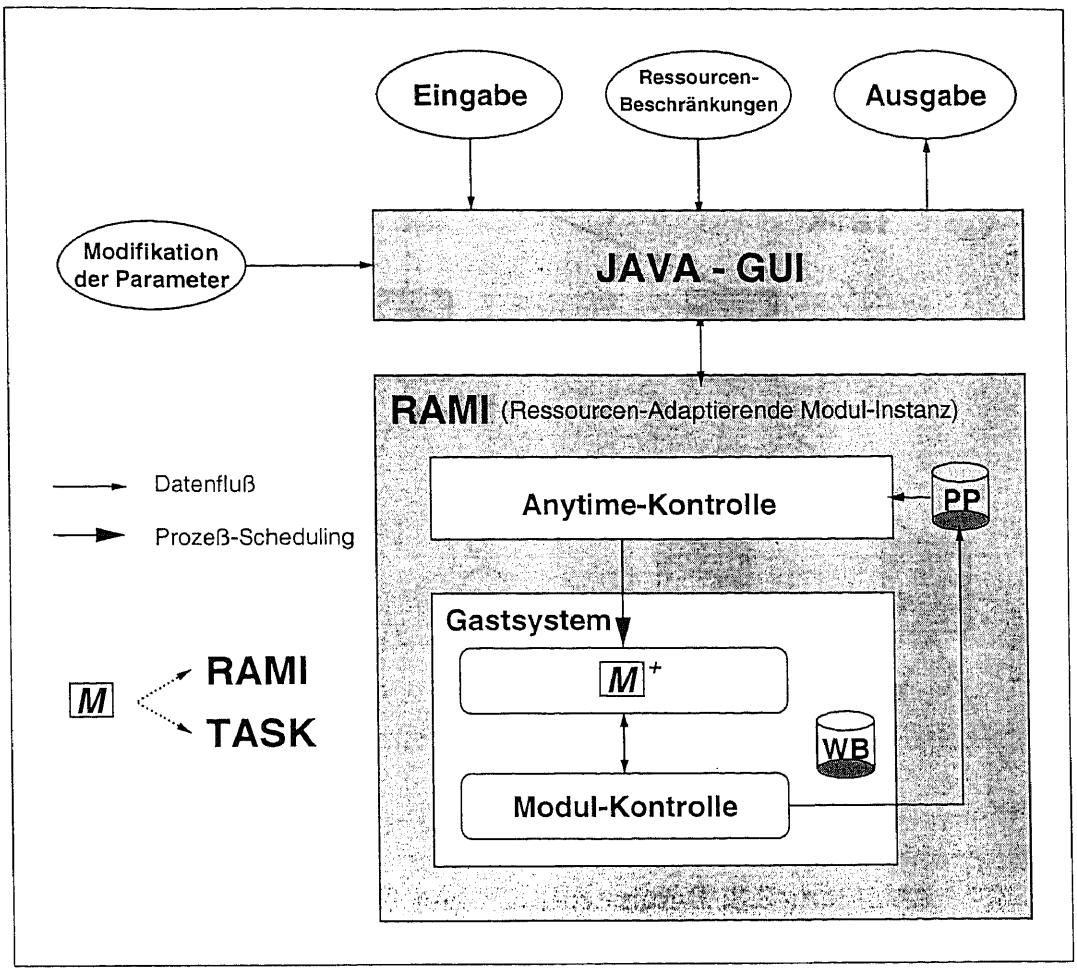

Abbildung 3.2: Architektur eines Gesamtsystems (mit Java-Werkbank)

Definition 3.7 (Modul-Kontrolle) *Eine* Modul-Kontrolle *ist Teil eines Gastsystems und steuert den Datenfluß zwischen seinen Modulen und anderen RAMIs; desweiteren protokolliert sie sämtliche Ergebnisqualitäten der Module im Zeitverlauf.*

Während einer Initialisierungsphase ist die Modul-Kontrolle ebenfalls zuständig für die Übergabe von Konfigurationsdaten an die Anytime-Kontrolle. Diese ist, wie in Abschnitt 3.1.1.1 beschrieben, völlig unabhängig vom Gastsystem und erhält auf diese Weise die Informationen über den internen Aufbau des Gastsystems, die sie für ihre Arbeit benötigt. Desweiteren ist die Modul-Kontrolle zur Laufzeit für die Bereitstellung der Performanzdaten verantwortlich, die die Anytime-Kontrolle zur Ressourcenverteilung verwendet.

Ein Gesamtsystem weist eine *rekursive Struktur* auf: Jedes Modul kann entweder ein sogenannter *Task*, also eine Art elementares Berechnungsverfahren, oder selbst wieder eine RAMI, d.h. ein eventuell komplexes Subsystem, sein.

Definition 3.8 (Task) *Ein Modul eines Gesamtsystems, das eine Lösungsstrategie repräsentiert, heißt* Task.

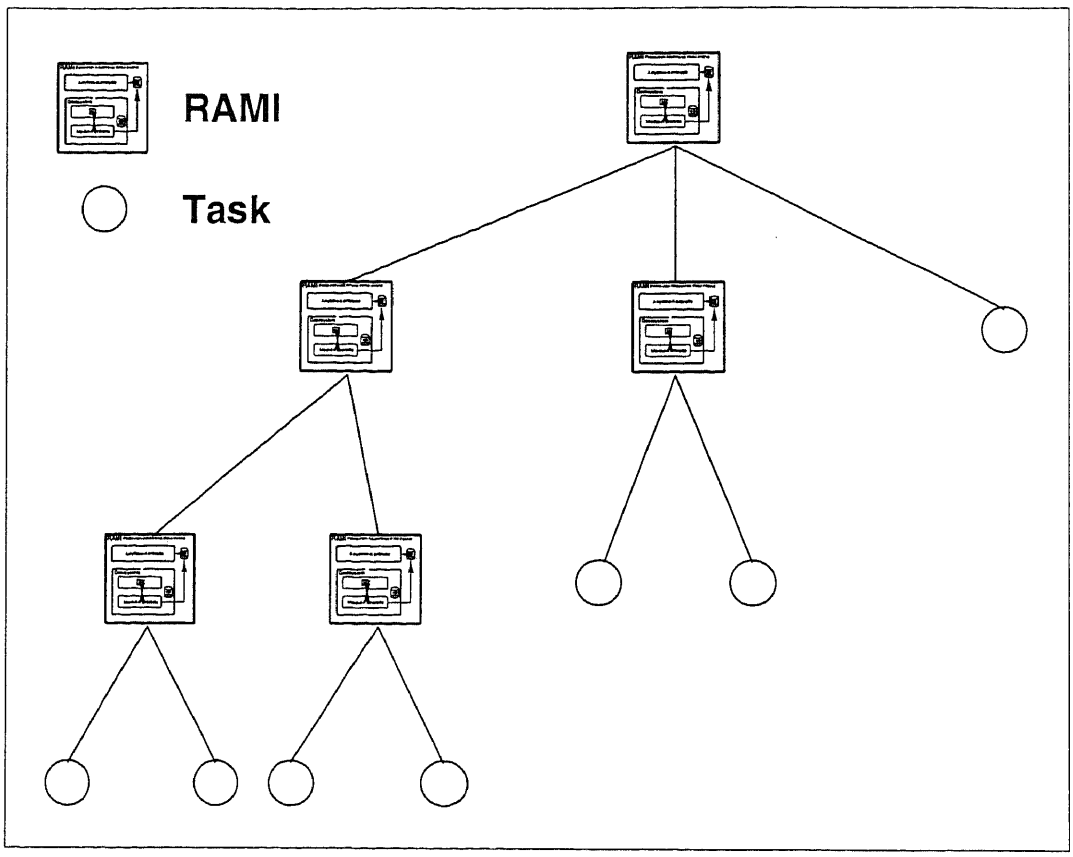

Abbildung 3.3: Baumstruktur eines Gesamtsystems

Hier gilt es hervorzuheben, daß also mehrere, voneinander unabhängige Anytime-Kontrollen – für jede RAMI eine – existieren. In Abschnitt 3.2.2 wird dies näher erläutert. Die Blätter des durch die RAMIs aufgespannten Baumes (vgl. Abb. 3.3) sind Tasks, d.h. hier erfolgt die eigentliche Berechnung.

Das entwickelte Gesamtpaket verfügt auch über eine plattformunabhängige Benutzerschnittstelle, die nicht nur Ein- und Ausgaben steuert, sondern auch Modifikationen an Systemeinstellungen zuläßt sowie interne Zustände der Anytime-System-Shell visualisiert. Diese Java-Werkbank wird von Wittig (1998) ausführlich beschrieben.

3.1.3 Struktur von Gastsystemen

Um in die Anytime-System-Shell integriert werden zu können, muß ein Gastsystem in einzelne Teilaufgaben gegliedert werden, die eine homogene hierarchische Systemstruktur erzeugen. Diese Module beinhalten alle konkreten Berechnungsverfahren. Insbesondere sollten jeweils mehrere unterschiedliche Lösungsverfahren, deren Ressourcenverbrauch differiert, implementiert werden. Um die angestrebte Eigenschaft der jederzeitigen Unterbrechbarkeit des Gesamtsystem zu optimieren, ist der Einsatz von Anytime-Algorithmen vorteilhaft; nichtsdestoweniger können aber auch (eventuell bereits existierende) Standard-Algorithmen Verwendung finden.

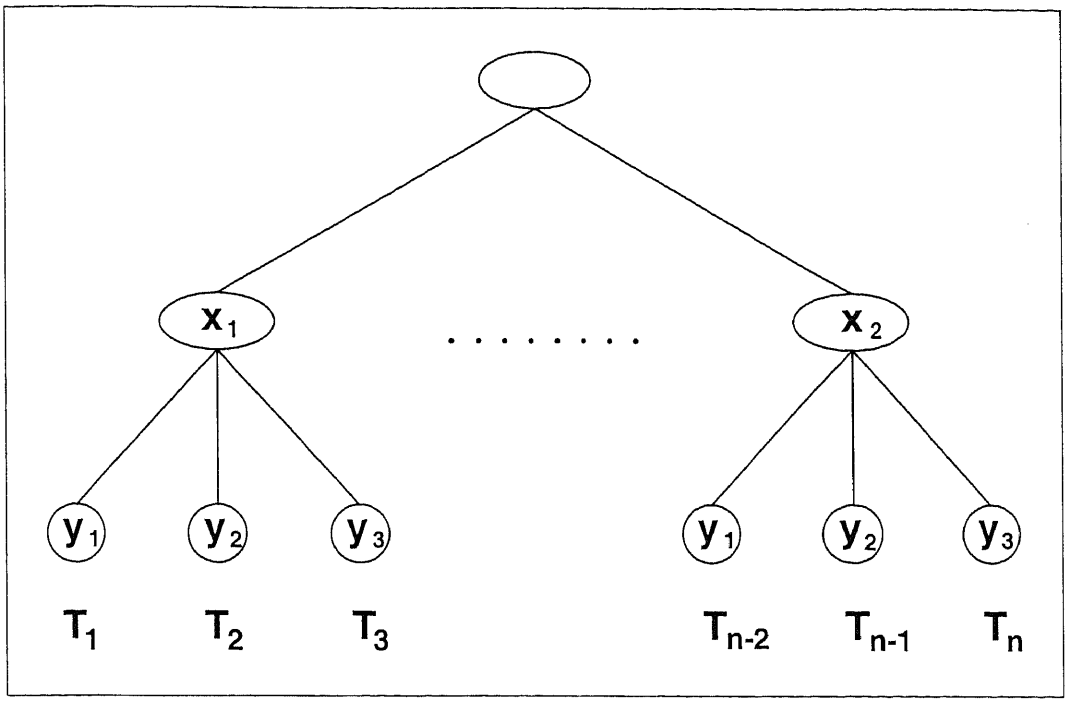

Abbildung 3.4: Homogene hierarchische Struktur eines Gastsystems

Die beispielhaft in Abbildung 3.4 dargestellte homogene hierarchische Systemstruktur spannt einen Baum auf, dessen Blätter alle die gleiche Tiefe haben. Dazu wird das Gesamtproblem über *Diskriminatoren*, die den inneren Knoten des Baumes entsprechen, immer weiter in Teilprobleme zergliedert, bis auf der Blattebene entweder elementare Lösungsstrategien (Tasks) oder weiterführende Subsysteme (RAMIs) erreicht sind. Anders gesagt wird durch eine Folge von Diskriminatoren – einem Pfad zu einem Blatt im Baum – ein bestimmter Algorithmus determiniert. Diskriminatoren können z.B. Parameter oder Entscheidungskriterien sein.

3.2 Zweierlei Kontrolle

Eine RAMI besteht im wesentlichen aus zwei Teilen: einer Anytime-Kontrolle und einem Gastsystem, das sich wie oben beschrieben aus Modulen und einer Modul-Kontrolle zusammensetzt. Die Aufspaltung in zwei unterschiedliche Kontrollinstanzen ermöglicht eine saubere Trennung zwischen problemspezifischen Aktivitäten auf Seite des Gastsystems einerseits und dem allgemeinen Ressourcenmanagement, das für alle RAMIs gleich strukturiert ist.

3.2.1 Modul-Kontrolle

Die Modul-Kontrolle beinhaltet den Programmcode eines Gastsystems, der nicht die eigentlichen Berechnungen umfaßt.

In einer Initialisierungsphase übergibt die Modul-Kontrolle Konfigurationsdaten an die zur RAMI des Gastsystems gehörende Anytime-Kontrolle.

Dies ist notwendig, da letztere vorher keine Informationen über die Programmstruktur besitzt, die für das Ressourcenmanagement erforderlich sind. Wichtigster Bestandteil ist die homogene hierarchische Systemstruktur (vgl. Abschnitt 3.1.3). Dazu gehören auch Angaben darüber, welche Subsysteme integriert sind. Wie die Anytime-Kontrolle mit diesem Wissen ihre internen Strukturen für die spätere Ressourcenverteilung aufbaut, wird in Abschnitt 3.2.2 dargestellt.

Im Rahmen der Konfigurationsdaten wird der Anytime-Kontrolle auch übermittelt, welche Berechnungen als Transaktionen, also als ununterbrechbare Programmteile registriert werden müssen. Eine geschachtelte Transaktion, die, wie in Abb. 3.5 gezeigt, innerhalb einer anderen Transaktion beginnt, ist nicht zulässig.

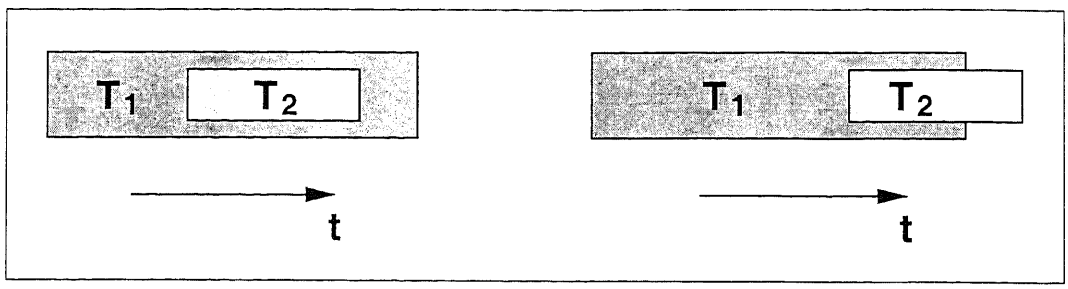

Abbildung 3.5: Geschachtelte Transaktionen

Andernfalls könnte das Ende der ersten oder zweiten Transaktion nicht berücksichtigt werden, da die andere noch nicht abgeschlossen wäre. Damit könnte eine eventuelle Unterbrechung nicht durchgeführt werden, wie es bei zwei aufeinanderfolgenden Transaktionen möglich ist (vgl. Abb. 3.6).

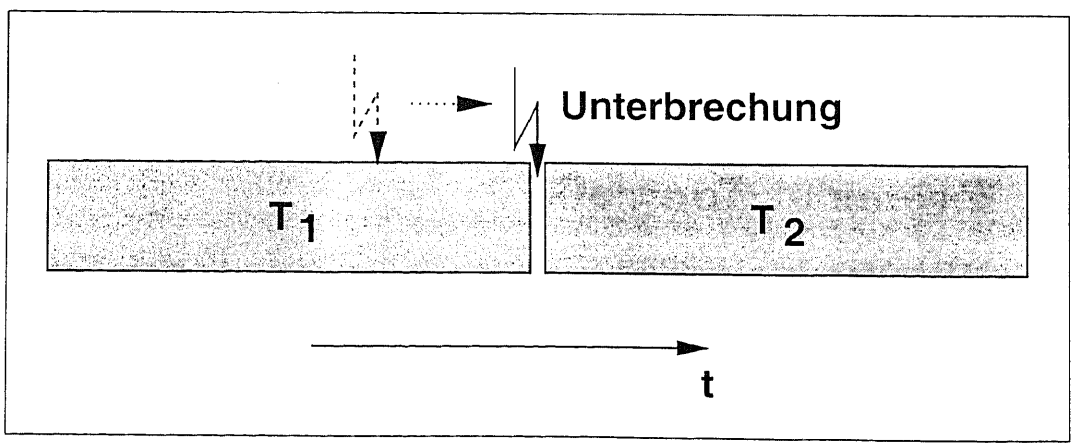

Abbildung 3.6: Direkt aufeinanderfolgende Transaktionen

Nach der Initialisierungsphase steuert die Modul-Kontrolle im weiteren Verlauf den Datenfluß zwischen den einzelnen Modulen des Gastsystems. Hat eine Komponente eine Berechnung abgeschlossen, wird das Resultat von der Kontrollinstanz bewertet und gegebenenfalls in das Gesamtergebnis integriert, welches für den Fall einer Unterbrechung immer aktualisiert wird.

Damit ist gewährleistet, daß jederzeit das bislang beste Resultat vorliegt, und das Gesamtsystem anytime-fähig ist (vgl. Abschnitt 2.6.1). Zusätzlich zum reinen Ergebnis der Berechnungen können auch dessen Ressourcenverbrauch und seine Qualität abgefragt werden. Während dies z.B. bei der Weitergabe von Zwischenergebnissen aus Subsystemen für die Qualitätsermittlung auf der nächsthöheren Ebene relevant ist, interessiert den Endbenutzer in der Regel nur das Ergebnis der Wurzel-RAMI, an die auch die entsprechende Anfrage gerichtet wurde.

Die Modul-Kontrolle beachtet auch Abhängigkeiten zwischen den Komponenten und kann so, unabhängig von der Ressourcenzuteilung, eigene Priorisierungen vornehmen.

Desweiteren protokolliert eine Modul-Kontrolle sämtliche Änderungen der Ergebnisqualitäten des entsprechenden Gastsystems und seiner Module im Zeitverlauf. Aus den so erhaltenen Zeit-Qualität-Wertepaaren werden stützpunktorientierte Performanzprofile generiert. Ihre Weiterverarbeitung, bevor sie von der Anytime-Kontrolle zur Ressourcenzuteilung herangezogen werden können (vgl. Abschnitt 3.4), wird in Abschnitt 3.4.2 beschrieben.

3.2.2 Anytime-Kontrolle

Die vorrangigste Aufgabe bei einem ressourcenbeschränkten System ist, neben der Problemlösung selbst, die Kontrolle über die verfügbaren Ressourcen. In der vorliegenden Arbeit leistet dies die Anytime-Kontrolle. Sie ist Bestandteil jeder RAMI und verwaltet neben der Ressource *Zeit* auch eine vom Benutzer vorgegebene Anzahl von Prozessen, die nebenläufig zur Problemlösung genutzt werden können.

Nachdem wie oben beschrieben die Modul-Kontrolle die Konfigurationsdaten eines Gastsystems an die zugehörige Anytime-Kontrolle übergeben und damit den internen Aufbau des Systems bekannt gemacht hat, wird ein *lokaler Ressourcenbaum* aufgebaut, der der homogenen hierarchischen Systemstruktur aus Abschnitt 3.1.3 entspricht. Der Ausdruck *lokal* deutet an, daß es sich um die Struktur eines Moduls handelt; betrachtet man das Gesamtsystem bestehend aus allen RAMIs, so erhält man einen *globalen Ressourcenbaum* (vgl. Abschnitt 3.3.2).

Innere Knoten eines lokalen Ressourcenbaumes repräsentieren *virtuelle Tasks*, die nicht mit konkreten Berechnungsverfahren assoziiert sind; sie stellen abstrakte Teilaufgaben dar. Im Gegensatz dazu können *realen Tasks* – den Blättern des Ressourcenbaumes – Prozesse zugeordnet werden, um eine oder mehrere Transaktionen durchzuführen.

Definition 3.9 (Realer Task) *Ein Task, der eine konkrete Berechnung durchführt, heißt* realer Task. *Er ist mit den Blättern eines lokalen Ressourcenbaumes assoziiert.*

Definition 3.10 (Virtueller Task) *Ein Task, der keine konkrete Berechnung durchführt, sondern nur eine Lösungsstrategie repräsentiert, heißt* virtueller

Task. *Er ist mit den inneren Knoten eines lokalen Ressourcenbaumes assoziiert.*

Reale Tasks, im folgenden einfach als Tasks bezeichnet, können folgende Zustände annehmen (vgl. dazu auch Abb. 3.7):

Definition 3.11 (Lauffähiger Task) *Ein realer Task, dem ein Prozeß zugeordnet ist, heißt* lauffähiger Task.

Definition 3.12 (Nichtlauffähiger Task) *Ein realer Task, dem kein Prozeß zugeordnet ist, heißt* nichtlauffähiger Task.

Definition 3.13 (Laufender Task) *Ein lauffähiger Task, der für die ihm zugeteilte Zeit CPU-Zugriff hat, heißt* laufender Task.

Definition 3.14 (Beendeter Task) *Ein realer Task, der die assoziierte Berechnung durchgeführt hat, heißt* beendeter Task.

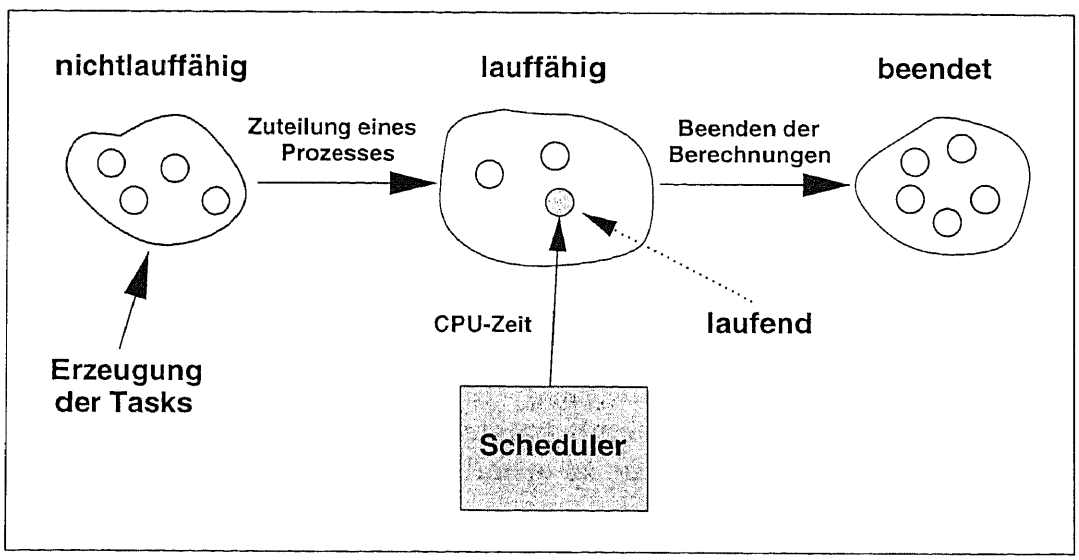

Abbildung 3.7: Zustände der Tasks

Zu Beginn ist jeder Task nichtlauffähig, bis ihm ein Prozeß zugeteilt werden kann und er lauffähig wird. Nur Prozessen in diesem Zustand kann dann gegebenenfalls Zeit für einen CPU-Zugriff zugeteilt werden. Geschieht dies, wird aus dem lauffähigen Task ein laufender, der seine zugehörigen Berechnungen abarbeiten kann. Nach Beendigung der in der zugeteilten Zeit ausführbaren Transaktionen gibt es zwei Möglichkeiten: Entweder ist die gesamte Berechnung abgeschlossen und der Task geht in den Zustand *beendet* über oder er wird wieder zu einem lauffähigen Task und muß auf eine weitere Zeit-Zuteilung für den CPU-Zugriff warten.

Das Scheduling der Tasks erfolgt zyklusweise, ähnlich dem *Round-Robin-Verfahren*, das bei Betriebssystemen Anwendung findet (vgl. (Silberschatz & Galvin, 1994)). Innerhalb eines Zyklus' wird lauffähigen Tasks CPU-Zeit zugeteilt. Ebenso wie die Zuordnung von Prozessen zu lauffähigen Tasks, also die Entscheidung darüber, welche Berechnung ausgeführt werden soll, richten sich sowohl Zyklenlänge als auch CPU-Zeit-Portionen nach den Ergebnissen der Ressourcenverteilung und können von unterschiedlicher Dauer sein. Wie das Ressourcenmanagement vor sich geht wird in den Abschnitten 3.3 und 3.4.2 beschrieben. Nach Beendigung eines Zyklus' beginnt dieser Allokationsvorgang erneut, bis alle Berechnungen durchgeführt sind oder eine externe Unterbrechung erfolgte.

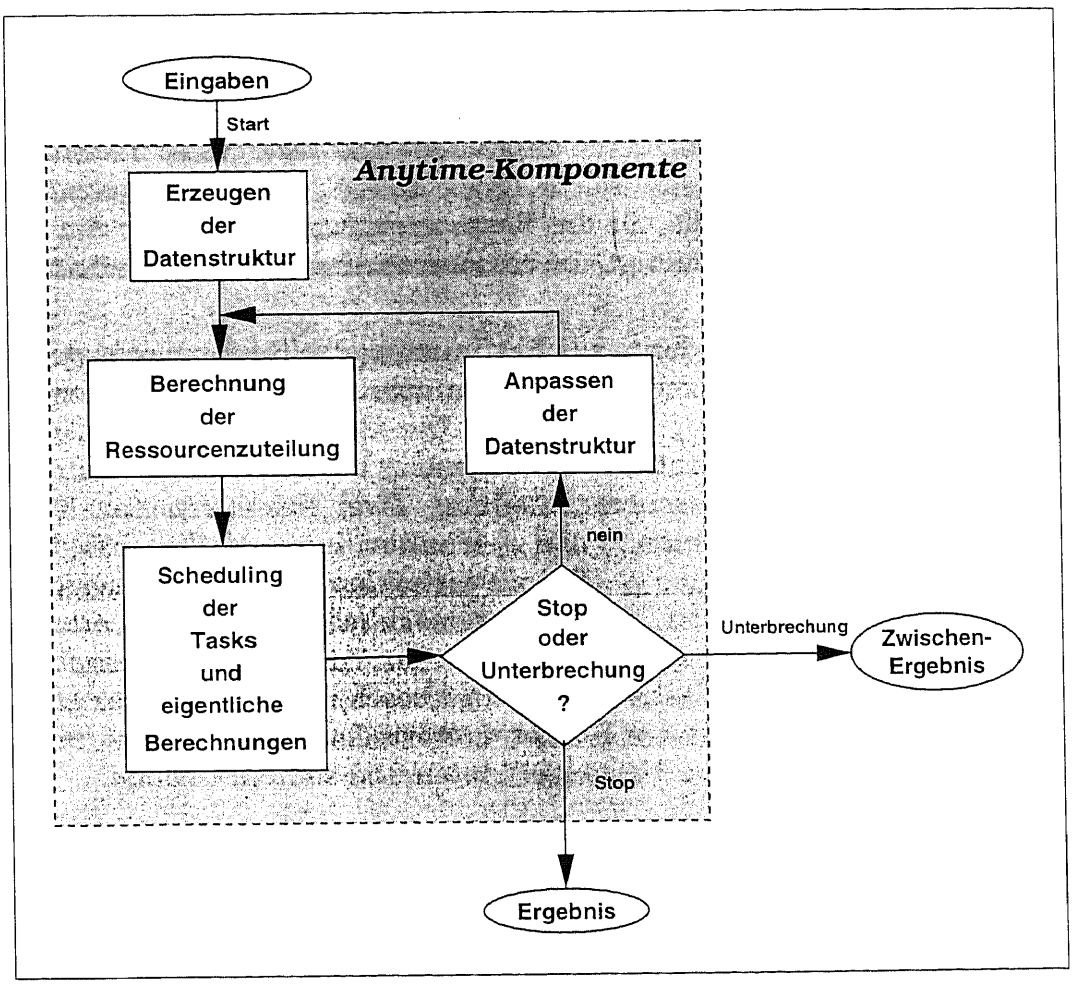

Abbildung 3.8: Prinzipieller Ablauf der Anytime-Kontrolle

Abbildung 3.8 stellt den gesamten Ablauf von der Initialisierungsphase bis zum Berechnungsende nochmals graphisch dar.

Das Ressourcenmanagement setzt sich aus der Bestimmung der Ressourcenverteilung und der Verarbeitung dieser Zuteilung zusammen; dies wird in den folgenden Abschnitten dargestellt.

3.3 Prozeßallokation und Scheduling

In diesem ersten Ansatz zur Bestimmung einer Ressourcenverteilung wird von einer durch den Systementwickler vorgegebenen Priorisierung der Teilaufgaben ausgegangen. Dadurch kann die prinzipielle Vorgehensweise bei der Verwaltung und Verteilung der Ressourcen unabhängig von der Errechnung ihrer jeweiligen Qualität dargestellt werden. Letzteres ist Thema von Abschnitt 3.4.

3.3.1 Lokale Ressourcenbäume: Struktur

Eine bei der Entwicklung des Systems vorgenommene Gewichtung der einzelnen Berechnungsverfahren ist z.B. für eine Testphase von großer Bedeutung, während der erste Performanzdaten gesammelt und verarbeitet werden (vgl. Abschnitt 3.4.2). Je länger eine solche Phase dauert, desto aussagekräftiger werden die Performanzprofile, die das Ressourcenmanagement für den zweiten, dynamischen Ansatz benötigt, um ressourcenadaptierendes Verhalten zu zeigen.

Bei dieser statischen Variante zur Bestimmung der Ressourcenverteilung wird jeder Kante eines lokalen Ressourcenbaumes – ausgehend von der Wurzel – ein *relatives Gewicht* g zugeordnet. Die Gesamtheit dieser Gewichte bestimmt eine ebenenorientierte Priorisierung aller virtuellen und realen Tasks, als Basis für die Ressourcenzuteilung. Zwei Nachbarknoten K_1 und K_2 mit den relativen Gewichten g_1 und g_2 erhalten demnach die verfügbaren Ressourcen im Verhältnis $g_1 : g_2$. Da mit virtuellen Tasks keine konkreten Berechnungen assoziiert sind, findet die tatsächliche Ressourcenverteilung nur auf Blattebene für die realen Tasks statt. Der Anteil an den Ressourcen, die ein bestimmter realer Tasks erhält, wird beschrieben durch das *globale Gewicht* g_{glob} des entsprechenden Blattes und errechnet sich als das Produkt der relativen Gewichte seiner Vorgängerknoten und seinem eigenen:

$$g_{glob}(T) = g(T) * \prod_{i=1}^{h-1} g(p_i). \tag{3.1}$$

Dabei sind $p_1, \ldots, p_{h-1}$ die Vorgängerknoten eines Tasks T im Wurzelpfad eines lokalen Ressourcenbaumes der Höhe h. Abbildung 3.9 illustriert das Verfahren an einem Beispiel.

Damit ist festgelegt, in welchem Verhältnis Zeit und Prozesse anhand der globalen Gewichte in einem lokalem Ressourcenbaum auf die realen Tasks verteilt werden. Wie diese Verteilung (bei der z.B. beachtet werden muß, daß Prozesse nicht beliebig teilbar sind) in der Praxis vor sich geht, ist Inhalt der Abschnitte 3.3.2 (Prozeßzuteilung) und 3.3.4 (Scheduling).

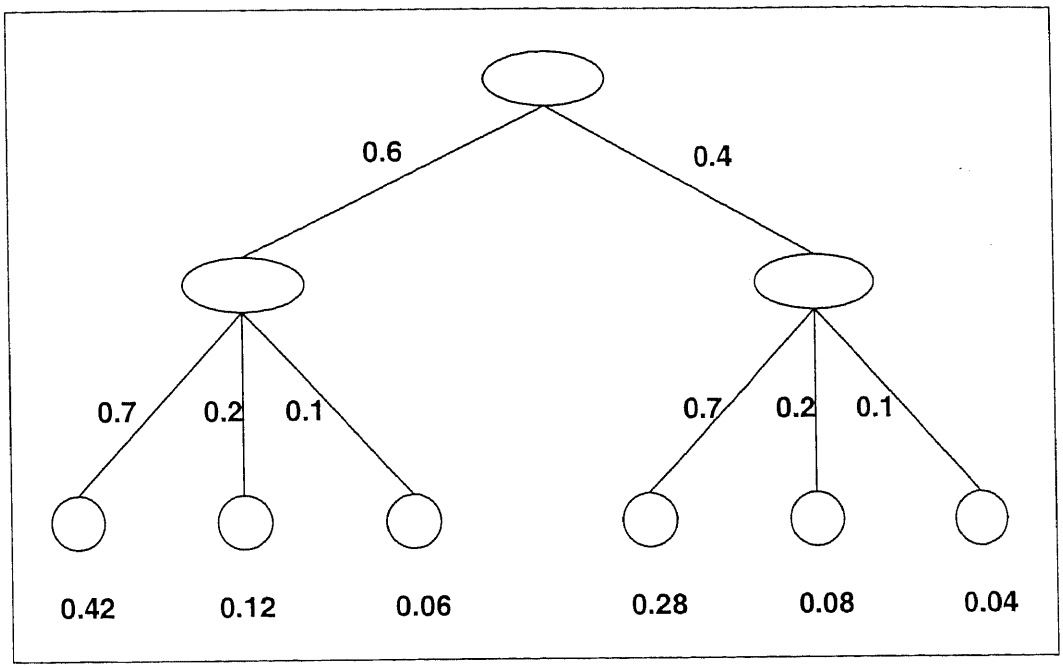

Abbildung 3.9: Beispiel für relative und globale Gewichte von Tasks

3.3.2 Globaler Ressourcenbaum: Aufbau und Prozeßallokation

Lokale Ressourcenbäume beschreiben wie oben erläutert die Ressourcenverteilung innerhalb *einer* RAMI. Um die vollständige Ressourcenallokation für ein Gesamtsystem zu erhalten, muß aus *allen* zum Gesamtsystem gehörenden *lokalen* Ressourcenbäumen ein *globaler Ressourcenbaum* aufgebaut werden. Auch hier können die Nachfolgeknoten virtueller Tasks sowohl reale als auch weitere virtuelle Tasks sein; allerdings mit dem Unterschied, daß in einem globalen – im Gegensatz zu einem lokalen – Ressourcenbaum ein realer Task sehr wohl Nachfolgeknoten besitzen kann, wenn der betreffende Task eine RAMI ist, also ein Subsystem repräsentiert. Für den globalen Ressourcenbaum bedeutet das, daß nicht alle Blätter die gleiche Tiefe besitzen. Ein erneuter Blick auf Abb. 3.3 verdeutlicht diese Tatsache: Die (lokalen) Wurzelpfade aller RAMIs sind jeweils gleich lang, während dies beim Gesamtbaum nicht der Fall ist.

Die Zuteilung der vorhandenen Prozesse auf die RAMIs erfolgt unter Verwendung des globalen Ressourcenbaumes in einem rekursiven Verfahren: Die oberste RAMI, also das Gesamtsystem, erhält alle Prozesse zugewiesen. Einen davon benötigt sie für sich selbst, die restlichen verteilt sie an ihre Sub-RAMIs entsprechend deren globalen Gewichten. Jede Sub-RAMI verfährt analog. Da *jede* RAMI zur Anytime-Kontrolle einen Prozeß *verbraucht*, muß sichergestellt werden, daß genügend Prozesse vorhanden sind, um überhaupt konkrete Berechnungen ausführen zu können und nicht nur Verwaltungsaufgaben zu erfüllen. Deshalb sollte die initiale Prozeßzahl entsprechend gewählt werden. Falls sich dennoch einmal dieses Problem ergibt, so werden

automatisch zusätzliche Prozesse zur Laufzeit erzeugt, was sich natürlich negativ auf dieselbe auswirkt und das Realzeitverhalten beeinträchtigt.

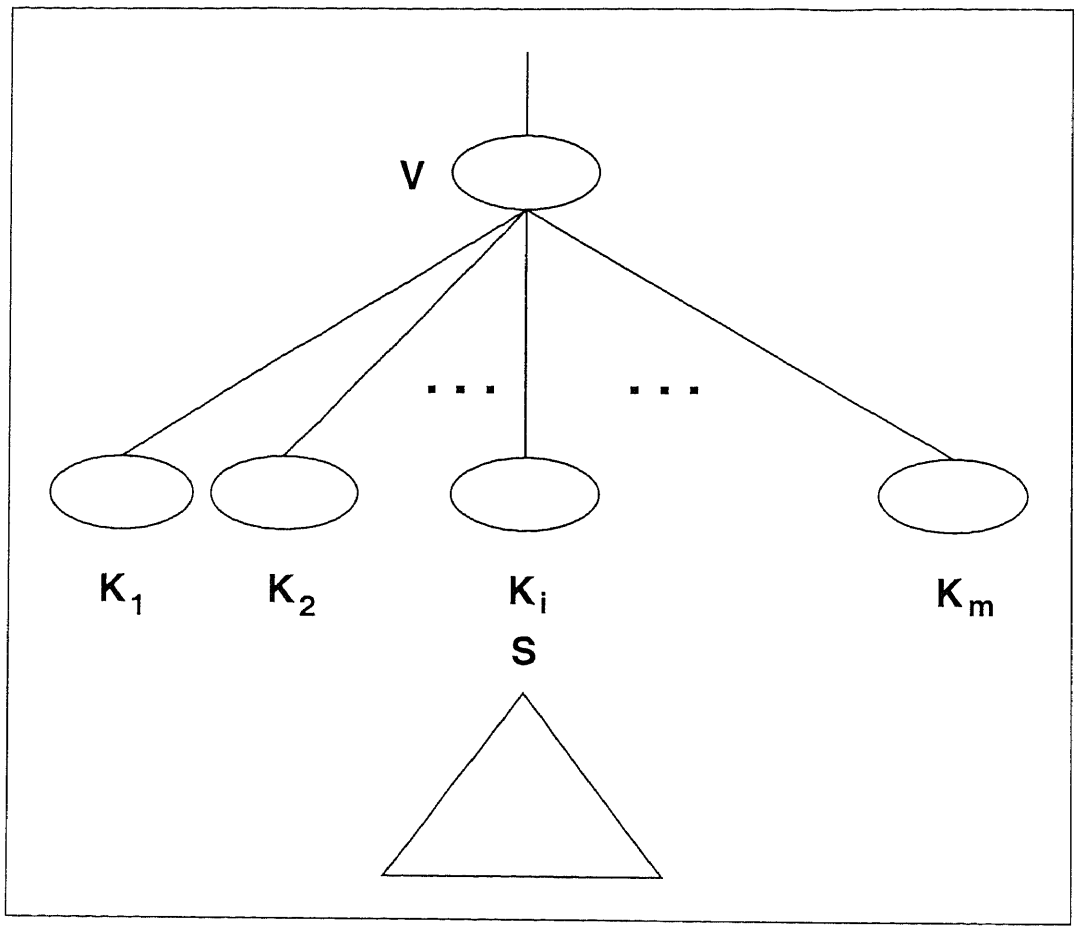

Abbildung 3.10: Verteilung der Prozesse an Subsysteme

Abbildung 3.10 illustriert noch einmal die Prozeßzuteilung. Wenn ein Vorgängerknoten V die Nachfolgeknoten $K_1, \ldots, K_m$ besitzt, dann errechnet sich die Anzahl der Prozesse P_{RAMI}, die einem Subsystemknoten $S \in K_i$ zugeteilt werden, wie folgt:

$$P_{RAMI}(S) = \max\{\lfloor \frac{g_{glob}(S)}{\sum_{j=1}^{m} g_{glob}(K_j)} * (P_{RAMI}(V) - Tasks(V)) \rfloor, 1\}. \qquad (3.2)$$

Dieser Wert setzt sich demnach aus dem Produkt des Anteils des globalen Gewichtes $g_{glob}(S)$ von S am Gesamtgewicht der $K_1, \ldots, K_m$ sowie der Differenz der Anzahl der zur übergeordneten RAMI V gehörenden Prozesse $P_{RAMI}(V)$ und der lauffähigen Tasks $Tasks(V)$ zusammen. Zu diesen Tasks zählt auch V selbst. Wie oben dargelegt, muß mindestens *ein* Prozeß zugeteilt werden, um sicherzustellen, daß konkrete Berechnungen durchgeführt werden können.

Für die Berechnung der Prozeßverteilung in lokalen Ressourcenbäumen werden im folgenden zwei Verfahren vorgestellt.

3.3.3 Lokaler Ressourcenbaum: Aufbau und Prozeßallokation

3.3.3.1 A priori-Aufbau und horizontale Prozeßallokation

Bei der horizontalen Prozeßallokation handelt es sich um ein der Breitensuche ähnelndes Verfahren, das die mittels des globalen Ressourcenbaumes den einzelnen RAMIs zugewiesenen Prozesse nun Ebene für Ebene anhand der relativen Gewichte $g(K_i)$ der Nachfolgeknoten $K_1, \ldots, K_m$ im jeweiligen lokalen Ressourcenbaum intern weiterverteilt. Die Zahl $P_{Task}(T)$ der Prozesse, die dabei einem virtuellen Task $T \in K_i$ aus dem Vorrat $P_{Task}(V)$ seines Vorgängers V zugeordnet werden, wird ähnlich wie oben berechnet. Zusätzlich wird sie begrenzt durch die maximal im Subbaum des betrachteten virtuellen Tasks benötigte Anzahl M, die derjenigen der untergeordneten Tasks entspricht und die über die Konfigurationsdaten bekannt ist. Zusammengefaßt erhält man:

$$P_{Task}(T) = \min\{\lceil \frac{g(T)}{\sum_{j=1}^{m} g(K_j)} * P_{Task}(V) \rceil, M\}. \tag{3.3}$$

Auf der Ebene der lokalen Ressourcenbäume werden *keine* eventuell benötigten Prozesse zusätzlich generiert; die Anytime-Kontrolle muß also mit den vorhandenen Ressourcen haushalten. Problematisch ist der Fall, in dem sehr viele Geschwisterknoten auftreten, die jeweils nur den Bruchteil eines Prozesses erhalten würden. Deshalb wird der entsprechende Term in Formel 3.3 aufgerundet. Die Prozesse werden – solange sie ausreichen – in absteigender Reihenfolge der relativen Gewichte an die einzelnen Tasks vergeben. Handelt es sich dabei um reale Tasks, so werden diese lauffähig; ansonsten wird das Verteilungsverfahren rekursiv fortgesetzt.

Ein konkretes Beispiel soll das Verfahren illustrieren (vgl. Abb. 3.11). Bei jedem Knoten eines virtuellen Tasks wurden dazu vier Werte annotiert: Oberhalb kann der Wert des ersten Arguments der Minimumbildung aus Formel 3.3 abgelesen werden, links die für die untergeordneten Tasks benötigte maximale Prozeßanzahl und rechts die Anzahl der tatsächlich zugeteilten Prozesse. Tasks, denen ein Prozeß zugeteilt wurde, die also lauffähig sind, werden schraffiert dargestellt. Insgesamt stehen fünf Prozesse zur Verfügung, die dem (Wurzel-)Knoten K_1 zugeteilt wurden (ersichtlich an der bei einem Wurzelknoten normalerweise unbesetzten oberen Annotationsposition). Nach Formel 3.3 werden den Nachfolgerknoten K_2 und K_3 dann drei bzw. zwei Prozesse zugeordnet:

$$P_{Task}(K_2) = \min\{\lceil \frac{0.75}{0.75 + 0.25} * 5 \rceil, 3\} = \min\{4, 3\} = 3; \tag{3.4}$$

$$P_{Task}(K_3) = \min\{\lceil \frac{0.25}{0.75 + 0.25} * 5 \rceil, 3\} = \min\{2, 3\} = 2. \tag{3.5}$$

Obwohl K_3 drei Prozesse benötigte, konnte er nur noch zwei erhalten. Diese werden entsprechend der relativen Gewichte auf die realen Tasks verteilt, so daß T_6 nicht lauffähig werden kann.

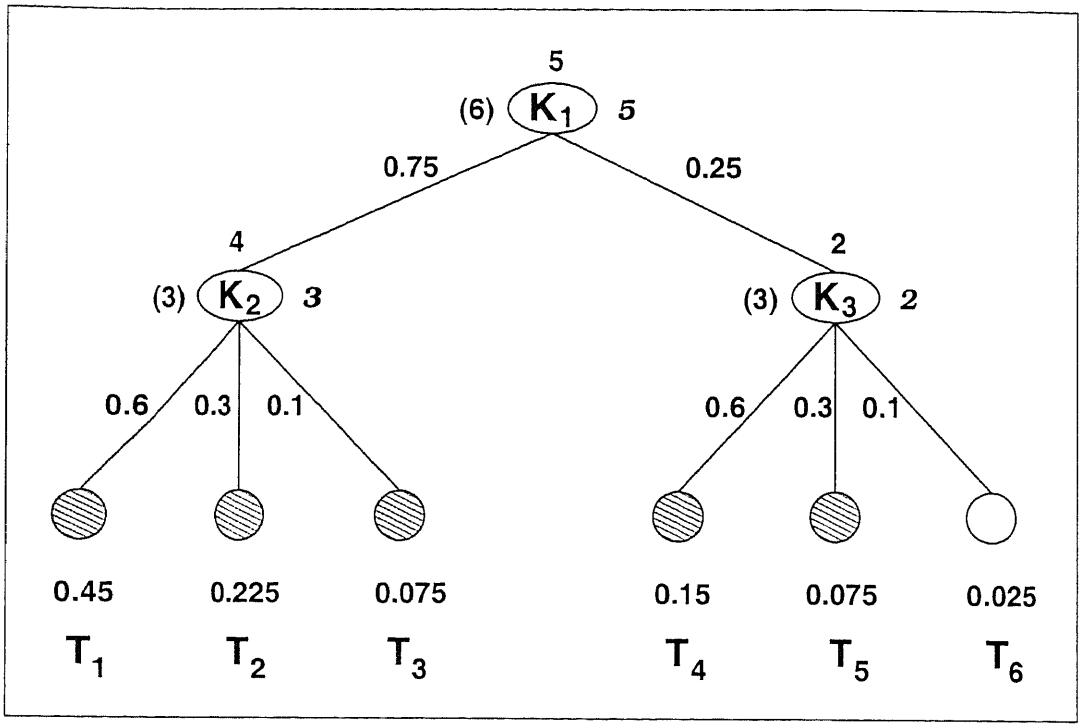

Abbildung 3.11: Beispiel zur Verteilung der Prozesse

3.3.3.2 Sukzessiver Aufbau und vertikale Prozeßallokation

Alternativ zu dem im vorigen Abschnitt erläuterten Verfahren soll nun die vertikale Prozeßallokation vorgestellt werden, die Analogien zur Tiefensuche aufweist. Während bei dem vorangegangenen Verfahren der lokale Ressourcenbaum zu Beginn eines Systemlaufs komplett konstruiert wurde, so wird er nun sukzessive vor jedem Zyklus nur soweit aufgebaut, wie freie Prozesse zur Verfügung stehen. Dadurch können keine nichtlauffähigen Tasks existieren. Die Prozesse werden einfach *von oben nach unten* – vertikal – entlang der Pfade mit den jeweils größten Gewichten auf die Tasks verteilt. Die globalen Gewichte spielen hier keine Rolle, die Reihenfolge richtet sich Ebene für Ebene nur nach den relativen Gewichten.

Abbildung 3.12 zeigt den lokalen Ressourcenbaum für das bekannte Beispiel, wie er sich bei diesem Verfahren ergibt.

Der wesentliche Unterschied besteht darin, daß überhaupt nur fünf Tasks berücksichtigt werden und der in Abb. 3.11 zu sehende nichtlauffähigen Task T_6 fehlt. Er kann erst in einem späteren Zyklus, wenn ein Prozeß frei geworden ist, etwa weil T_2 terminierte, verarbeitet werden (vgl. Abb. 3.13).

3.3.3.3 Vergleich der Alternativen zur Prozeßallokation

Horizontale und vertikale Prozeßallokation unterscheiden sich insbesondere in der Interpretation der Gewichtungen der Tasks. Erstere betont den nebenläufigen Charakter indem explizit die globalen Gewichte in die Verteilung Eingang finden. Dadurch sollte das Gesamtsystem in der Regel ein

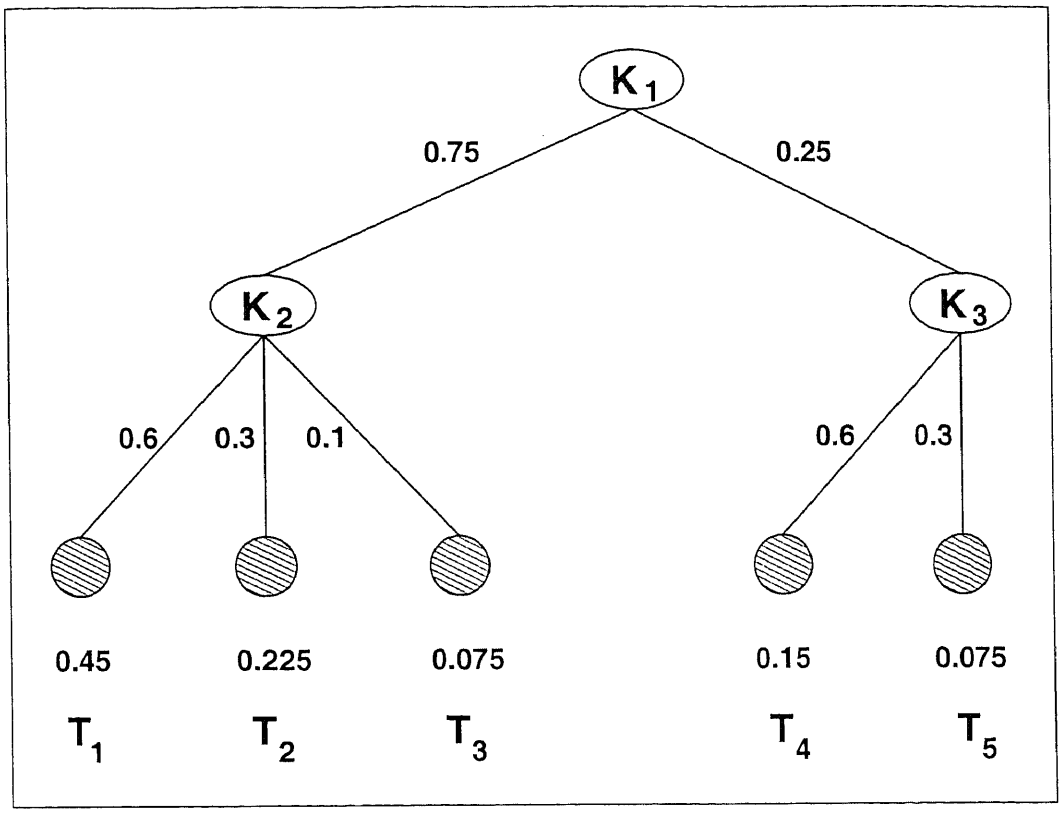

Abbildung 3.12: Beispiel des alternativen Aufbaus eines lokalen Ressourcenbaums

gleichmäßiges Anytime-Verhalten an den Tag legen. Im Gegensatz dazu ist die Sichtweise beim sukzessiven Aufbau des lokalen Ressourcenbaums gröber, da die Zuteilungsentscheidungen nur jeweils die relativen Gewichte auf einer einzelnen Ebene berücksichtigen. Dies führt dazu, daß bei einer *gelungenen* Bestimmung dieser Gewichte damit zu rechnen ist, daß – eventuell nach einer gewissen Anlaufzeit mit schlechteren Resultaten als oben – recht schnell mit einer guten Qualität der (Zwischen-)Ergebnisse gerechnet werden darf.

Der Aufwand für beide Strategien zur Prozeßallokation, also die Zeit, die zur Konstruktion des lokalen Ressourcenbaums benötigt wird, ist ähnlich, tritt aber zu unterschiedlichen Zeiten auf: Während beim a priori-Aufbau die Arbeit, wie der Name schon sagt, komplett zu Beginn anfällt, verteilt sie sich bei der schrittweisen Variante auf die Gesamtdauer der Berechnung. Dies hat den Vorteil, daß hier bei vorzeitiger Unterbrechung keine möglicherweise unnötigen Zuteilung berechnet werden müssen und somit Zeit gespart werden kann. Die horizontale Zuteilung ist zu Beginn langsamer, es dauert länger bis ein erstes Resultat vorliegt; hier hat die vertikale Variante einen Vorteil, der verfahrensbedingt mit zunehmender Laufzeit immer geringer wird, weil der Ressourcenbaum vor jedem Zyklus aktualisiert werden muß.

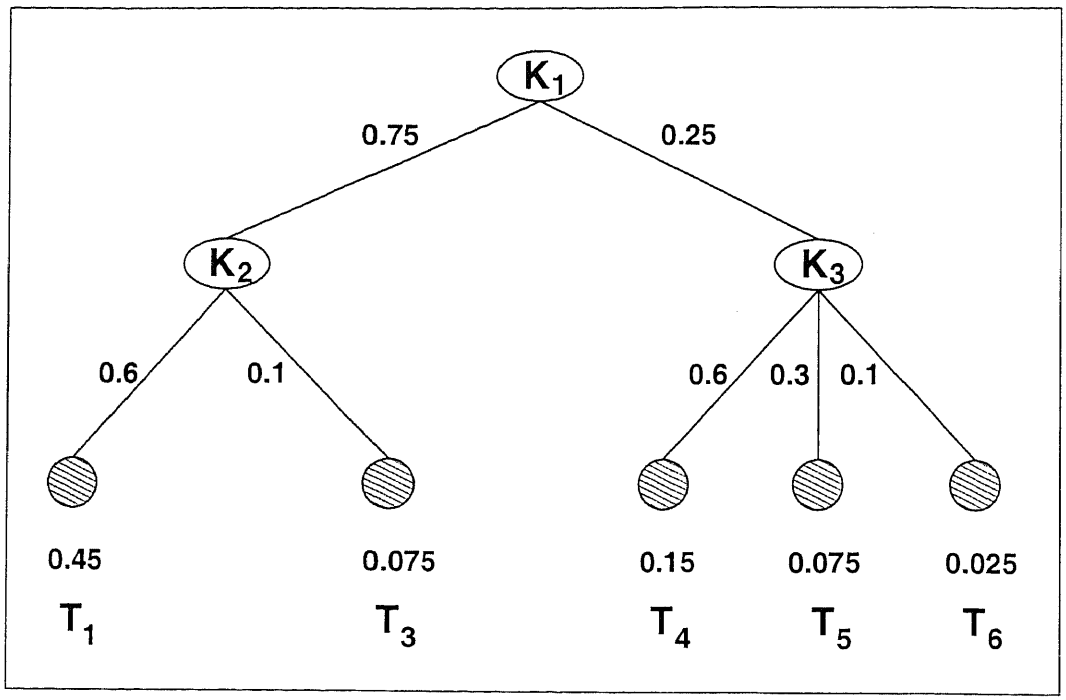

Abbildung 3.13: Beispiel des Updates eines lokalen Ressourcenbaums bei alternativem Aufbau

3.3.4 Scheduling

In den vorangegangenen Abschnitten wurde zunächst beschrieben, wie ein Gesamtsystem in einzelne Teilaufgaben gegliedert und diesen dann mittels bestimmter Verfahren Prozesse zugeordnet werden. Wie in Abschnitt 3.2.2 erläutert, wird zyklusweise aus der so erzeugten Menge lauffähiger Tasks einem nach dem anderen CPU-Zeit zugeteilt, so daß er zeitweilig in den Zustand *laufend* übergeht und seine Berechnungen ausführen kann. Nach welchen Kriterien dieser Scheduling-Vorgang abläuft, wird nachfolgend dargelegt.

Die *Dauer der einzelnen Zyklen* ist ein Systemparameter, der entweder vom Entwickler vorgegeben wird oder anhand interner Daten vor dem Beginn eines jeden neuen Zyklus' optimiert werden kann. Bei der Größe dieses Wertes spielt z.B. eine Rolle, wie lang durchschnittliche Tasks dauern, aber auch besonders langwierige sind zu berücksichtigen. Allgemein ist festzuhalten, daß eine geringe Varianz in der Dauer der Tasks das Anytime-Verhalten verbessert. Die Zyklusdauer hat somit einen beträchtlichen Einfluß auf die Performanz des Gesamtsystems. Ein zweiter Systemparameter, der die Ressourcenzuteilung auf die Tasks beeinflußt, ist das *minimale Prozeßquantum*: Dieser Wert ist plattformabhängig und gibt an, wie viel Zeit mindestens pro Tasks aufgewendet werden muß, um anfallende administrative Aufgaben wie z.B. den erforderlichen Kontextwechsel überhaupt vornehmen zu können. Ist das minimale Prozeßquantum zu klein, können eventuell *keine* eigentlichen Berechnungen durchgeführt werden.

Der Scheduler einer Anytime-Kontrolle verteilt die Zyklusdauer, also die

vorhandene Zeit, gemäß der globalen Gewichte in absteigender Reihenfolge auf die lauffähigen Tasks:

$$t(T) = \max\{\frac{g_{glob}(T)}{\sum_{j=1}^{m} g_{glob}(T_j)} * t_{Rest}, PQ\}. \tag{3.6}$$

Hierbei ist beachtenswert, daß davon ausgegangen werden kann, daß die Ressource *Zeit* im Gegensatz zur Ressource *Prozeßanzahl* beliebig teilbar ist. Die CPU-Zeit $t(T)$, die einem Task T aus der Menge aller noch nicht bearbeiteten lauffähigen Tasks $T_1, \ldots, T_m$ in einen Zyklus zugeteilt wird, ist dann das Maximum von minimalem Prozeßquantums PQ oder dem Produkt aus dem Verhältnis des globalen Gewichtes $g_{glob}(T)$ von T zur Summe aller globalen Gewichte $g_{glob}(T_1), \ldots, g_{glob}(T_m)$ und der verbliebenen Zykluszeit t_{Rest}.

Nachdem die CPU-Zeit für einen Task nach Formel 3.6 berechnet ist, wird er aktiviert, d.h. sein Zustand ändert sich von *lauffähig* zu *laufend* und er führt seine Berechnungen aus. Dabei gibt es zwei Möglichkeiten: Entweder terminiert der Task in der ihm zugeteilten Zeit und gibt die Kontrolle von sich aus zurück oder er wird unterbrochen, sei es weil seine CPU-Zeit abgelaufen ist oder weil ein externes Unterbrechungssignal gegeben wurde. In aller Regel wird die einem Task zugewiesene Zeit nicht exakt mit seiner darauffolgenden Laufzeit übereinstimmen: Zum einen beruht die Berechnung der Zuteilung auf Schätzungen, zum anderen können systeminterne Verzögerungen die Rechenzeit beeinflussen. Der Task beendet eventuell seine Berechnungen schneller als erwartet oder er befindet sich in einer Transaktion, die ja nicht unterbrochen werden kann, und kehrt somit erst verspätet zurück. Um diese Ungenauigkeiten möglichst zu minimieren – insbesondere sollen sie nicht durch den ganzen Zyklus propagiert werden – wird die reale Laufzeit $t_{real}(T^i)$ des Tasks T^i gemessen und zur Aktualisierung der verbliebenen Restzeit t_{Rest}^{i+1} herangezogen:

$$t_{Rest}^{i+1} = t_{Rest}^{i} - t_{real}^{i}(T^i). \tag{3.7}$$

Danach kann die CPU-Zeit für den nächsten Tasks T^{i+1} berechnet werden, ohne daß Ressourcen verschwendet wurden.

Ein Zyklus ist beendet, wenn einer der folgenden Fälle eintritt:

- Allen lauffähigen Tasks wurde nacheinander einmal CPU-Zeit zugeteilt und sie haben die entsprechenden Berechnungen durchgeführt.
- Die gesamte Zykluszeit ist verbraucht. Möglicherweise kann nicht allen Tasks Rechenzeit zugewiesen werden.
- Ein externes Unterbrechungssignal erfolgt.

Der Algorithmus dieses Verfahrens wird in Abb. 3.14 nochmals als Pseudocode dargestellt:

Die Funktion `pop` holt dabei immer den Task aus der Menge der lauffähigen mit dem größten globalen Gewicht. `lasse_laufen` aktiviert den Task und kontrolliert seine korrekte Abarbeitung bezüglich Laufzeit, Transaktionen und Unterbrechungssignale.

```
    Scheduler(Tasks_lauffähig, t_Zyklus)
(1) t_Rest ← t_Zyklus
(2) while not empty Tasks_lauffähig and t_Rest > 0.0 do
(3)     T ← pop(Tasks_lauffähig)
(4)     t(T) ← max{ g_glob(T) / Σ_{j=1}^{m} g_glob(T_j) * t_Rest, PQ }
(5)     initialisiere_Uhr
(6)     lasse_laufen(T, t(T))
(7)     t_real(T) ← stoppe_Uhr
(8)     t_Rest ← t_Rest − t_real(T)
(9) od
```

Abbildung 3.14: Vorgehensweise des Schedulers

In Beispiel aus Abschnitt 3.3.3.1, Abb. 3.11, konnte den Tasks $T_1, \ldots, T_5$ ein Prozeß zugeordnet werden, sie sind also lauffähig und können jetzt vom Scheduler verwaltet werden. Die Dauer des im folgenden betrachteten Zyklus betrage zehn Zeiteinheiten, das minimale Prozeßquantum eine Zeiteinheit. Task T_1 hat das größte globale Gewicht, ihm wird somit zuerst CPU-Zeit zugeteilt:

$$t(T_1) = \max\{\frac{0.45}{0.45 + 0.225 + 0.075 + 0.15 + 0.075} * 10ZE, 1ZE\} \quad (3.8)$$

$$= \max\{4.62ZE, 1ZE\} \quad (3.9)$$

$$= 4.62ZE. \quad (3.10)$$

Der Scheduler weist also dem Task T_1 $4.62ZE$ zu, in denen dieser Berechnungen ausführen kann. In diesem Beispiel wird davon ausgegangen, daß keine externe Unterbrechung erfolgt; ferner decke sich die tatsächlich gemessene Laufzeit mit der zugewiesenen CPU-Zeit. Nun kann die noch verbleibende Rest-Zykluszeit berechnet werden:

$$t_{Rest} = 10ZE - 4.62ZE = 5.38ZE. \quad (3.11)$$

Der nächste Task, der vom Scheduler abgearbeitet werden muß, ist T_2. Er erhält $2.31ZE$:

$$t(T_2) = \max\{\frac{0.225}{0.225 + 0.075 + 0.15 + 0.075} * 5.38ZE, 1ZE\} \quad (3.12)$$

$$= \max\{2.31ZE, 1ZE\} \quad (3.13)$$

$$= 2.31ZE. \quad (3.14)$$

Nach Ablauf der ihm zugeteilten $2.27ZE$ befinde sich T_2 in einer Transaktion, der Scheduler kann also nicht unterbrechen bevor diese beendet wurde. Damit erhöhe sich die reale Laufzeit auf $3.11ZE$ und die jetzt verbleibende Restzeit in diesem Zyklus sind $2.27ZE$:

$$t_{Rest} = 5.38ZE - 3.11ZE = 2.27ZE. \tag{3.15}$$

Bei der Zuteilung für T_3 greift das minimale Prozeßquantum:

$$\begin{aligned} t(T_3) &= \max\{\frac{0.075}{0.075 + 0.15 + 0.075} * 2.27ZE, 1ZE\} && (3.16)\\ &= \max\{0.57ZE, 1ZE\} && (3.17)\\ &= 1ZE. && (3.18) \end{aligned}$$

Die weitere Restzeitbestimmung und Zeitverteilung erfolgt analog auf die Tasks T_4 und T_5.

Für den nächsten Zyklus muß nun der lokale Ressourcenbaum angepaßt werden. Dies ist notwendig, da eventuell Tasks vollständig beendet werden konnten, oder aber weil sich Gewichte verändern können, wie Abschnitt 3.4.2 zeigen wird. Im ersten Fall entfernt der Scheduler den beendeten Task aus dem lokalen Ressourcenbaum und gibt den zugeordneten Prozeß frei, der dann weiterverwendet werden kann. Abbildung 3.15 zeigt den Algorithmus, der ausgehend von dem betreffenden Task auch dessen Vorgängerknoten ohne weitere Kinder aus dem Baum entfernt:

```
    Anpassen_des_Ressourcenbaumes(T)
(1) while not Nachfolger(T) do
(2)     Entferne_Task(T)
(3)     T ← Vorgänger(T)
(4) od
```

Abbildung 3.15: Anpassen des lokalen Ressourcenbaumes

Auf die entsprechende RAMI übertragen, bedeutet das, daß bestimmte Teilaufgaben, die durch reale und eventuell auch virtuelle Tasks beschrieben wurden, gelöst sind.

Angenommen, in obigem Beispiel konnte nach den Tasks T_1, T_3 auch T_2 beendet werden, so wird der gesamte linke Teilbaum entfernt (vgl. Abb. 3.16). Nachdem auf diese Weise der lokale Ressourcenbaum adaptiert wurde, kann die Zuteilung der Rechenzeit auf die verbliebenen lauffähigen Tasks im nächsten Zyklus wie beschrieben erfolgen.

Als letzter Punkt in diesem Abschnitt wird die rekursive Verarbeitung von Unterbrechungssignalen erläutert. Dabei handelt es sich um den Fall, daß ein laufender Task eine RAMI darstellt. Er wird unterbrochen, damit dem nächsten Task Rechenzeit zugeteilt werden kann. Die Besonderheit besteht nun darin, daß dieses Stopsignal *nach unten* durchgereicht werden muß, da besagte RAMI ja selbst wieder über ihre Anytime-Kontrolle CPU-Zeit verteilt hat. Der hier aktuell laufende Task muß ebenso unterbrochen werden, damit er nicht unkontrolliert Ressourcen verbraucht.

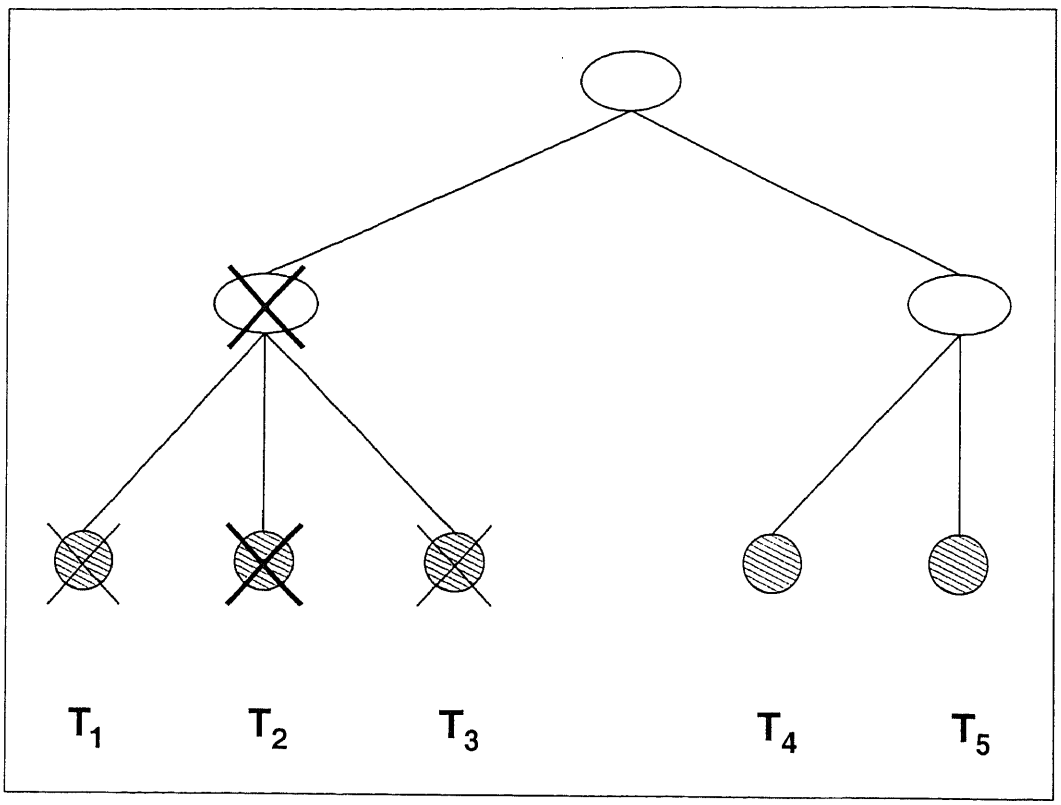

Abbildung 3.16: Rekursives Entfernen beendeter Tasks aus dem lokalen Ressourcenbaum

Abbildung 3.17 zeigt, wie das Signal zur untersten RAMI weitergeleitet wird und danach die Tasks von den Blättern zur (relativen) Wurzel unterbrochen werden.

Der folgende Abschnitt, der die Optimierung der Ressourcenverteilung zur Laufzeit unter Verwendung von Performanzprofilen (PP) behandelt, zeigt, daß eine Anpassung des lokalen Ressourcenbaumes eventuell auch erfolgen muß, weil sich – wie oben schon erwähnt – die relativen und globalen Gewichte der Tasks während eines Systemlaufs ändern können. Dies verbessert das Anytime-Verhalten und die Ergebnisqualitäten, beeinflußt jedoch nicht den hier beschriebenen Ablauf der Prozeßzuteilung und des Scheduling, sondern wird durch eine dynamische Verwaltung der Gewichte berücksichtigt, die sich bei der zyklusweisen Adaption des Baumes auswirkt.

3.4 Dynamische Optimierung der Ressourcenverteilung

Bisher fußte die Ressourcenverteilung immer auf vorgegeben Gewichten. Wie am Ende des vorigen Abschnitts angedeutet, kann sowohl das Anytime-Verhalten als auch die Ergebnisqualitäten verbessert werden, wenn die Gewichtungen dynamisch adaptiert werden.

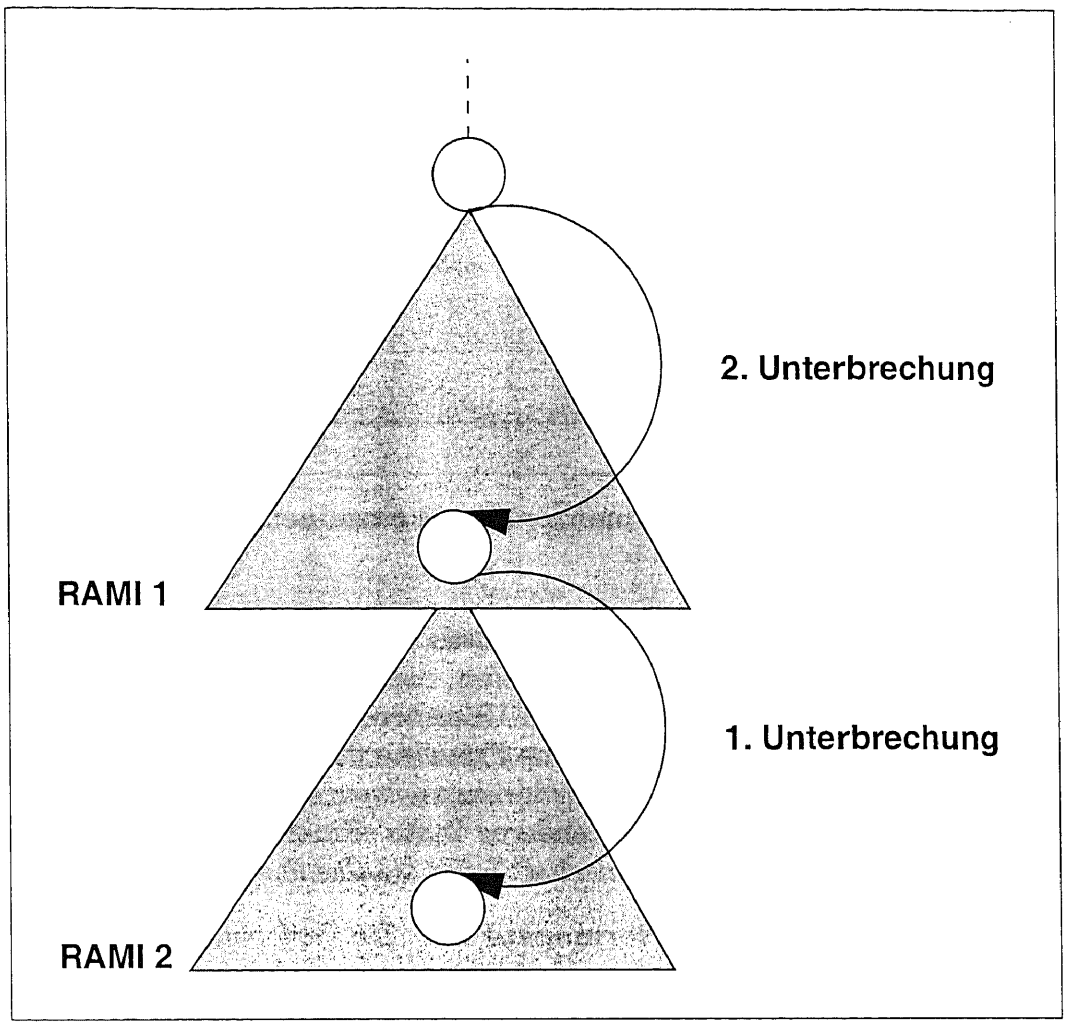

Abbildung 3.17: Rekursives Unterbrechen von Tasks

3.4.1 Voraussetzungen

Grundlage einer Optimierung des Verhaltens von Gastsystemen in der Anytime-System-Shell sind die Performanzprofile (vgl. u.a. Abschnitt 2.6.2): Diese werden zur Laufzeit protokolliert und in einer Datenbasis verwaltet. Sie enthält sozusagen das Wissen über den Systemverlauf vorangegangener Anfragen und kann zur Optimierung der laufenden herangezogen werden. Wie das geschieht, wird im weiteren erläutert. Die dazu benötigten Begriffe werden nachfolgend eingeführt.

Performanzprofile werden in der zugrundeliegenden Arbeit *stützpunktweise* repräsentiert (vgl. Abb. 3.18):

$$PP = \{(t_i, q_i) \mid t_i \in \mathbb{R}_0^+; q_i \in [0,1]; i = 1, \ldots, n\}. \tag{3.19}$$

Zu bestimmten Zeitpunkten t_i, etwa nach Beendigung jeder Transaktion, werden Zeit-Qualität-Paare (t_i, q_i) hinzugefügt.

Zweck der Performanzprofile ist die Repräsentation der Qualitätsentwicklung *eines* Moduls, d.h. selbst wenn das Modul von Systemstart bis -Ende

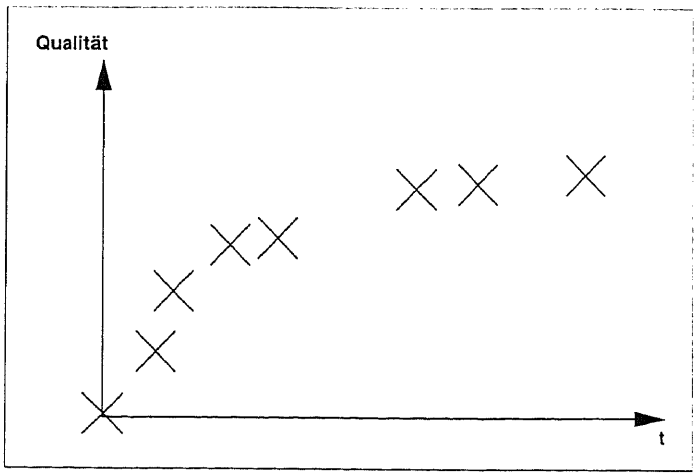

Abbildung 3.18: Ein Performanzprofil aus einzelnen Stützpunkten

immer wieder einmal Zeit zugeteilt erhalten hat, so interessieren nur exakt die Zeiträume in denen die Tasks liefen, nicht etwa der erste und letzte Aktivierungszeitpunkt. Ebenso müssen Unterbrechungen u.ä. berücksichigt werden, dergestalt, daß das Profil keine Fremdeinflüsse mehr enthält. Dazu werden die Aktivierungs- und Deaktivierungszeitpunkte jedes Tasks mitprotokolliert, um so die Rechenphasen anderer Tasks eliminieren zu können. Ein Profil mit der beschriebenen Eigenschaft heißt *lokales Performanzprofil*:

Definition 3.15 (Lokales Performanzprofil) *Ein Performanzprofil, das die Qualitätsentwicklung eines Moduls unabhängig von anderen Modulen abbildet, heißt* lokales Performanzprofil.

Lokale Performanzprofile bilden den Ausgangspunkt für die in der Performanzdatenbasis enthaltenen Profile:

Definition 3.16 (Akquiriertes Performanzprofil) *Ein Performanzprofil eines virtuellen oder realen Tasks oder einer RAMI, das die Menge der Performanzprofile, die bei realen Systemläufen unter Verwendung der Anytime-System-Shell gewonnen wurden auf eine geeignete Weise repräsentiert, heißt* akquiriertes Performanzprofile (aPP)*; es wird in einer entsprechenden Datenbasis zusammen mit seiner* Wertigkeit *gespeichert.*

Alle vollständigen Performanzprofile des selben Tasks haben, da es sich um lokale Profile handelt, gleich viele Stützstellen. Dies wird für ein gewichtetes Normalisierungsverfahren genutzt, das jedes neu erstellte Profil in das bereits akquirierte der Datenbasis integriert. Abbildung 3.19 skizziert das Vorgehen.

Das vorhandene akquirierte Profil repräsentiert zwei Durchläufe, weshalb ihm eine Wertigkeit von 2 zugeordnet ist, während das aktuelle Performanzprofil einfach gewertet wird. Beide Profile gehen entsprechend dieser Gewichte bei der Kombination zu einem aktualisierten akquirierten Profil, das dann eine Wertigkeit von 3 hat, ein. Bei der Erweiterung eines akquirierten Performanzprofils muß berücksichtigt werden, daß die Reihenfolge der Abarbeitung

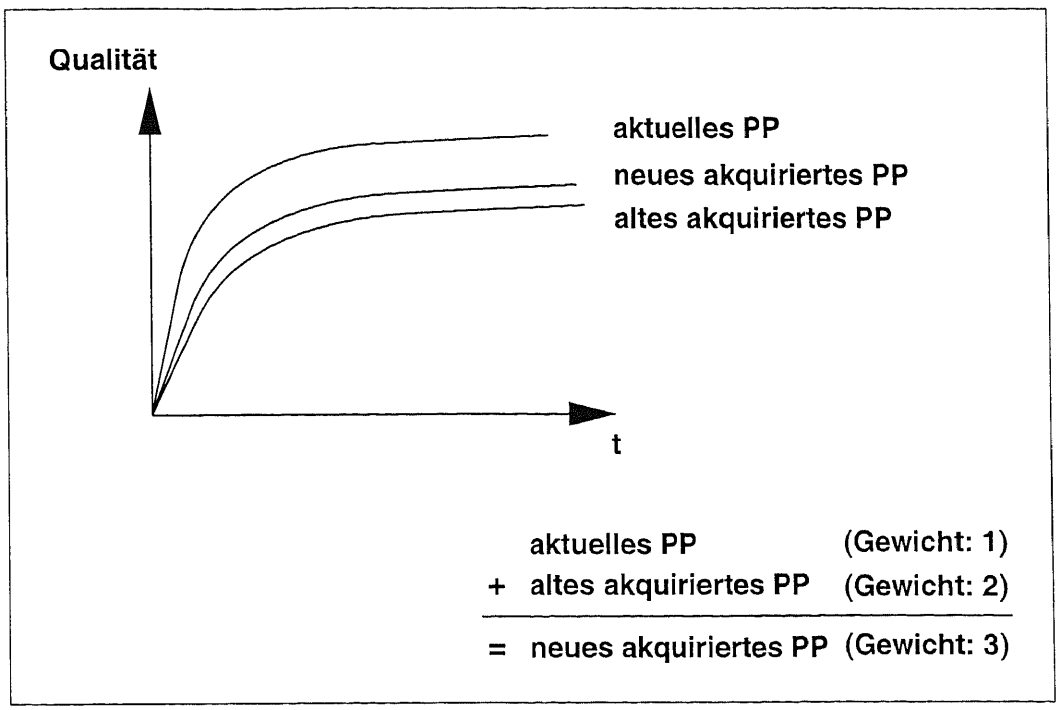

Abbildung 3.19: Erweiterung eines akquirierten Performanzprofils

der einzelnen Transaktionen variieren kann, was sich in unterschiedlichen Zeit- und/oder Qualitätsintervallen wiederspiegelt.

Auf der Zeitachse werden die Stützpunkte des akquirierten PP deshalb mittels des folgenden Faktors skaliert:

$$f = \frac{t_{max}^{aPP} + (count + 1) * (t_{max}^{nPP} - t_{max}^{aPP})}{t_{max}^{aPP}}. \tag{3.20}$$

Dabei bezeichnen t_{max}^{aPP} und t_{max}^{nPP} die maximalen Zeitwerte des akquirierten bzw. des neu zu integrierenden lokalen Performanzprofils, während *count* die Wertigkeit angibt. Analog wird für die Qualitätswerte vorgegangen, so daß mit wachsender Anzahl der Systemläufe eine stabile Performanzdatenbasis entsteht. In diesem Beispiel handelt es sich also um eine Gleichgewichtung; natürlich sind auch andere Relationen denkbar, etwa wenn jüngere Ergebnisse stärker berücksichtigt werden sollen als ältere.

Um eine Ressourcenverteilung mittels Performanzprofilen durchführen zu können, müssen diese für alle Tasks des Systems in der Datenbasis existieren, da es sonst zu Inkonsistenzen kommen kann, wenn etwa vorgegebene Gewichte als Default verwendet werden. Dies wird verhindert, indem in einem solchen Fall ganz auf die statische Variante zurückgegriffen wird.

Eine dynamische Ressourcenverteilung, setzt – wie erwähnt – eine Anpassung des lokalen Ressourcenbaumes in jedem Zyklus voraus, um die Gewichte zu aktualisieren. Dazu reicht aber die Kenntnis vergangener Performanzen nicht aus. Vielmehr muß ein akquiriertes Performanzprofil stets in Relation zum gegenwärtigen Stand des aktuellen Laufs betrachtet werden, der durch das gerade entstehende Profil repräsentiert wird. Um eine optimale Vertei-

lung der Ressource *Zeit* zu gewährleisten, muß aber die *zukünftige* Entwicklung der beteiligten Module so genau wie möglich vorhergesagt werden. Dazu werden *projezierte Performanzprofile* verwendet:

Definition 3.17 (Projeziertes Performanzprofil) *Ein Performanzprofil, das, ausgehend von aktuellem Zeitpunkt und erreichter Qualität eine Abschätzung der zukünftigen Entwicklung dieses Moduls unter Zuhilfenahme des akquirierten Performanzprofils repräsentiert, heißt* projeziertes Performanzprofil (pPP).

Die Konstruktion eines projezierten Performanzprofils illustriert Abb. 3.20.

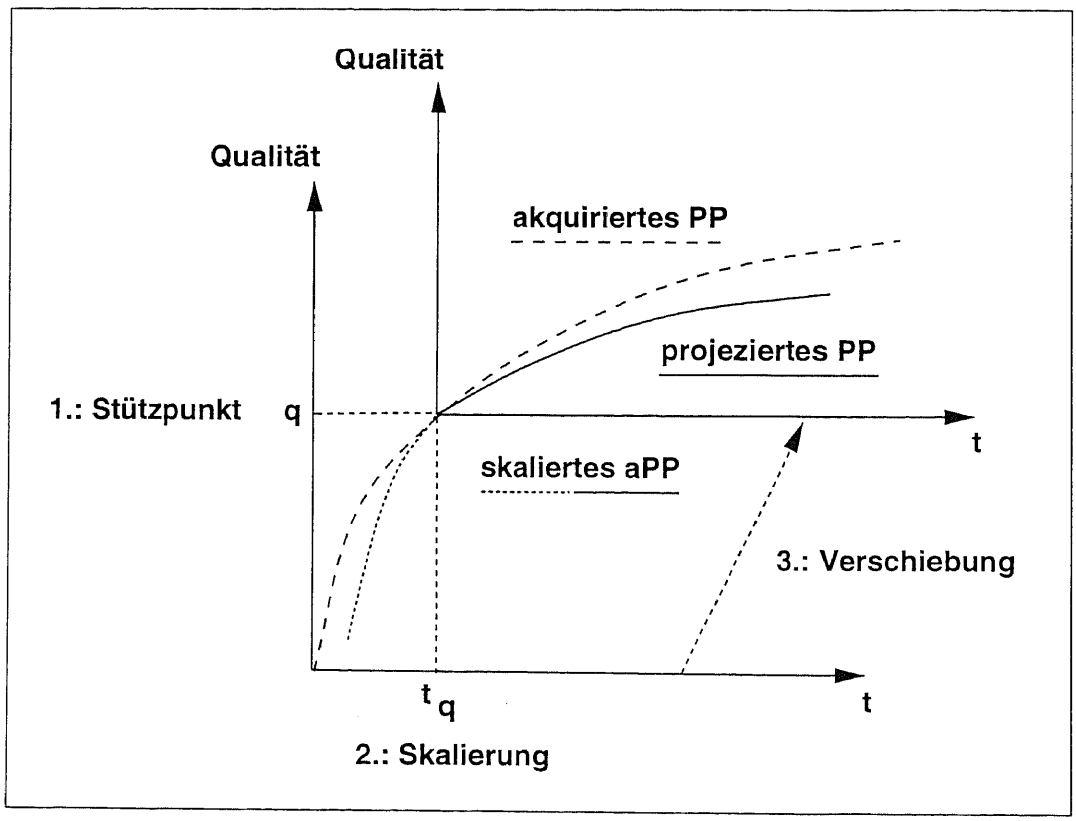

Abbildung 3.20: Konstruktion eines projezierten Performanzprofils

Wie in Definition 3.17 dargelegt, benötigt der Algorithmus aus Abb. 3.21 zur Bestimmung des projezierten Performanzprofils als Eingabe den aktuellen Zeitpunkt t_{akt} und die bisher durch das Modul erreichte Qualität q sowie das zugehörige akquirierte Performanzprofil.

Im ersten Schritt wird an der Stelle t_q, an der das akquirierte Performanzprofil die aktuelle Qualität q erreicht, ein neuer Stützpunkt mit den Koordinaten (t_q, q) in eine Kopie des aPP eingefügt. Danach erfolgt eine Skalierung bezüglich der Zeitachse mit dem Faktor $\frac{t_q}{t_{akt}}$. Damit wird der gegebenenfalls unterschiedlichen Zeitdauer zur Erreichung der Qualität q im aktuellen Lauf einerseits und im akquirierten Performanzprofil andererseits Rechnung getragen. Schließlich wird die skalierte Kopie des aPP so verschoben, daß der

```
        projeziertes_PP(PP, t_akt, q)
(1)     (t_q, q) ← neuer_Stützpunkt()
(2)     skaliere(PP, t_q / t_akt)
(3)     verschiebe(PP)
(4)     return (PP)
```

Abbildung 3.21: Algorithmus zur Bestimmung des projezierten Performanzprofils

neue Stützpunkt im Ursprung zu liegen kommt; das projezierte PP ist dann der Teil des Profils, der innerhalb des ersten Quadranten verläuft. Das so konstruierte projezierte Performanzprofil gibt nun die zu erwartende Qualitätsentwicklung wieder und kann mit anderen pPP verglichen werden (vgl. Abschnitt 3.4.2).

Abbildung 3.22 skizziert noch einmal den Algorithmus, der vor jedem neuen Zyklus den aktuellen Zustand mit dem erlernten Wissen aus vorangegengenen Systemläufen vergleicht und daraus globale Gewichte für die zu bearbeitenden Tasks bestimmt.

```
        Ressourcenverteilung()
(1)     Überprüfung, ob alle benötigten akquirierten Performanzprofile in der
        Datenbasis enthalten sind
(2)     Erstellen der projezierten Performanzprofile für die Tasks
(3)     Algorithmus zur Bestimmung der globalen Gewichte der lauffähigen
        Tasks
```

Abbildung 3.22: Algorithmus zur Bestimmung der globalen Gewichte

Für die konkrete Berechnung der Gewichte stehen verschiedene Verfahren zur Verfügung; sie werden in folgenden Abschnitt detailliert vorgestellt.

3.4.2 Verfahren zur Optimierung der Zeitverteilung

Für die Kombination von Anytime-Algorithmen hat Zilberstein bereits einige Methoden vorgeschlagen (vgl. Abschnitt 2.6.3). Ein Schwachpunkt seiner Offline-Compilierung war allerdings, daß die zur Verfügung stehende Zeit vor dem Systemstart bekannt sein mußte. Dies ist bei der hier vorgestellten Anytime-Kontrolle nicht notwendig. Nichtsdestoweniger können Zilbersteins Verfahren – zwar abgewandelt – durchaus angewendet werden, um Module zu verknüpfen, d.h. um einerseits eine Abarbeitungsreihenfolge und andererseits eine Ressourcenzuteilung vorzunehmen. Zusätzlich wurden alternativ neue Algorithmen entwickelt, die die Besonderheiten des vorliegenden Systems optimal ausnützen und somit zu einer Performanzsteigerung des Ge-

samtsystems beitragen können[3]. Grundlage für alle Methoden sind die im vorigen Abschnitt eingeführten projezierten Performanzprofile, die einen direkten Vergleich untereinander erlauben.

3.4.2.1 Die Regressions-Methode

Wie in Abschnitt 2.6.3 angedeutet, schlägt Zilberstein für die Kombination von Anytime-Algorithmen eine Maximumbestimmung nach vorangegangener monotonie-erhaltender Verknüpfung vor; speziell beschreibt er ein additives und ein multiplikatives Verfahren für als stetige Funktionen vorliegende Performanzprofile. Diese Funktionen sollten jeweils alle aus derselben Klasse sein, z.B. lineare oder Exponentialfunktionen.

Die von der Anytime-System-Shell generierten Profile sind allerdings stützpunktweise gegeben, so daß sie erst über ein Regressions-Verfahren zu einer stetigen Funktion approximiert werden müssen.

Eine erste Analyse soll die optimale Verknüpfung zweier durch Performanzprofile repräsentierten Anytime-Algorithmen AA_1 und AA_2 behandeln, die sich mittels linearer Regression gut an Funktionen des Typs $PP(t) = q_0 + at$ annähern lassen (vgl. Abb. 3.23). Für Algorithmen, deren Performanzprofile sich besser exponentiell als linear annähern lassen, kann – wie ebenfalls in der Abbildung dargestellt – analog vorgegangen werden (vgl. (Baus & Beckert, 1998)). Die Parameter für ein Profil mit n Stützstellen (t_i, q_i) bestimmen sich dann jeweils wie folgt, wobei q_0 eine eventuelle Inputqualität darstellt:

$$q_0 = \frac{\sum_i t_i^2 \sum_i q_i - \sum_i t_i q_i \sum_i t_i}{n \sum_i t_i^2 - \left(\sum_i t_i\right)^2} \tag{3.21}$$

und

$$a = \frac{n \sum_i t_i q_i - \sum_i t_i \sum_i q_i}{n \sum_i t_i^2 - \left(\sum_i t_i\right)^2}. \tag{3.22}$$

Um die bestmögliche Verteilung einer Zeit t_{ges} auf zwei Profile

$$PP_1(t_1) = q_1 + a_1 t_1 \tag{3.23}$$

und

$$PP_2(t_2) = q_2 + a_2 t_2 \tag{3.24}$$

zu erreichen, muß die Gesamtqualität berechnet werden; mit $t_2 = t_{ges} - t_1$ und multiplikativer Verknüpfung ergibt sich:

$$\begin{aligned} Q_{ges}(t_{ges}, t_1) &= PP_1(t_1) PP_2(t_{ges} - t_1) && (3.25)\\ &= (q_1 + a_1 t_1)\,(q_2 + a_2(t_{ges} - t_1)) && (3.26)\\ &= -a_1 a_2 t_1^2 + (a_1 a_2 t_{ges} + a_1 q_2 - a_2 q_1) t_1 + q_1 a_2 t_{ges} + q_1 q_2. && (3.27) \end{aligned}$$

[3]Die konkreten Implementationen aller im weiteren besprochenen Verfahren werden in der Diplomarbeit von Beckert (2000) beschrieben (vgl. auch (Baus & Beckert, 1998)).

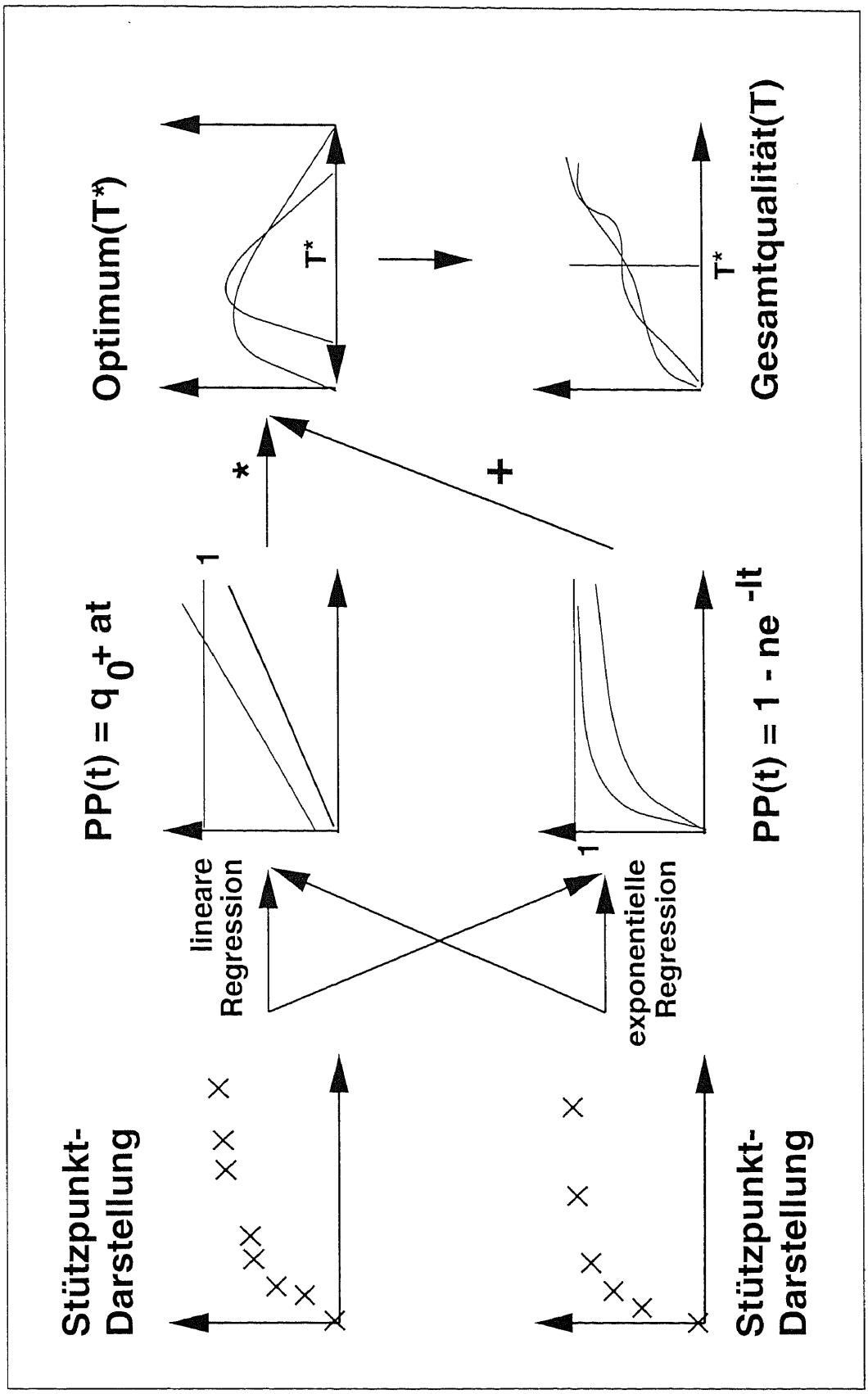

Abbildung 3.23: Optimale Ressourcenverteilung auf zwei Anytime-Algorithmen durch lineare bzw. exponentielle Regression

Durch Nullsetzen der Ableitung wird die in Abb. 3.23 rechts oben veranschaulichte Gesamtqualität maximal:

$$\frac{\delta Q_{ges}}{\delta t_1} = 0 \tag{3.28}$$

$$-2a_1a_2t_1 + a_1a_2t_{ges} + a_1q_2 - a_2q_1 = 0 \tag{3.29}$$

$$\Leftrightarrow t_1 = \frac{a_1q_2 - a_2q_1 + a_1a_2t_{ges}}{2a_1a_2} \tag{3.30}$$

$$= \frac{t_{ges}}{2} + \frac{a_1q_2 - a_2q_1}{2a_1a_2}. \tag{3.31}$$

Die Zeitverteilung auf die beiden Algorithmen wird also optimal für

$$t_1 = \frac{t_{ges}}{2} + \frac{a_1q_2 - a_2q_1}{2a_1a_2} \tag{3.32}$$

und

$$t_2 = t_{ges} - t_1 = \frac{t_{ges}}{2} + \frac{a_2q_1 - a_1q_2}{2a_1a_2}. \tag{3.33}$$

Die jeweils zugeteilten Zeitspannen t_1 und $t_2 = t_{ges} - t_1$ sind direkt von der Gesamtzeit t_{ges} abhängig, so daß sich t_1 (und damit auch t_2) als Funktion $t_1(t_{ges})$ darstellen lassen. Die optimale Qualität des Gesamtergebnisses der Kombination von AA_1 und AA_2 kann demnach allgemein zu jedem beliebigen t_{ges} berechnet werden als:

$$Q_{ges}(t_{ges}, t_1(t_{ges}))$$

$$= PP_1(t_1(t_{ges}))PP_2(t_{ges} - t_1(t_{ges})) \tag{3.34}$$

$$= \left(q_1 + a_1\left(\frac{t_{ges}}{2} + \frac{a_1q_2 - a_2q_1}{2a_1a_2}\right)\right)\left(q_2 + a_2\left(\frac{t_{ges}}{2} + \frac{a_2q_1 - a_1q_2}{2a_1a_2}\right)\right) \tag{3.35}$$

$$= \frac{(a_1a_2t_{ges} + a_1q_2 + a_2q_1)^2}{4a_1a_2} \tag{3.36}$$

$$=: Q_{ges}(t_{ges}). \tag{3.37}$$

Abbildung 3.23 zeigt rechts unten den Verlauf der Funktion für die Gesamtqualität $Q_{ges}(t_{ges})$, die nur noch von der Gesamtzeit t_{ges} abhängig ist und im folgenden als *Gesamtprofil* (der Verknüpfung der beiden Algorithmen) bezeichnet wird.

Die bisher vorgestellten Methoden zur Gewinnung von Funktionen aus Stützpunktmengen haben den Nachteil, daß sie nur in sehr speziellen Fällen eine gute Annäherung darstellen. Besonders problematisch ist die Tatsache, daß nicht immer im voraus bekannt sein dürfte, welche Funktion(-sklasse) in gegebenen Fall zu wählen ist; ist diese Entscheidung allerdings getroffen, ist die Weiterverarbeitung einfach.

Eine Alternative zur Repräsentation von Stützstellenmengen als *eine* Funktion ist eine abschnittsweise Annäherung, wie die lineare Interpolation zwischen den einzelnen Punkten aus Abb. 3.24:

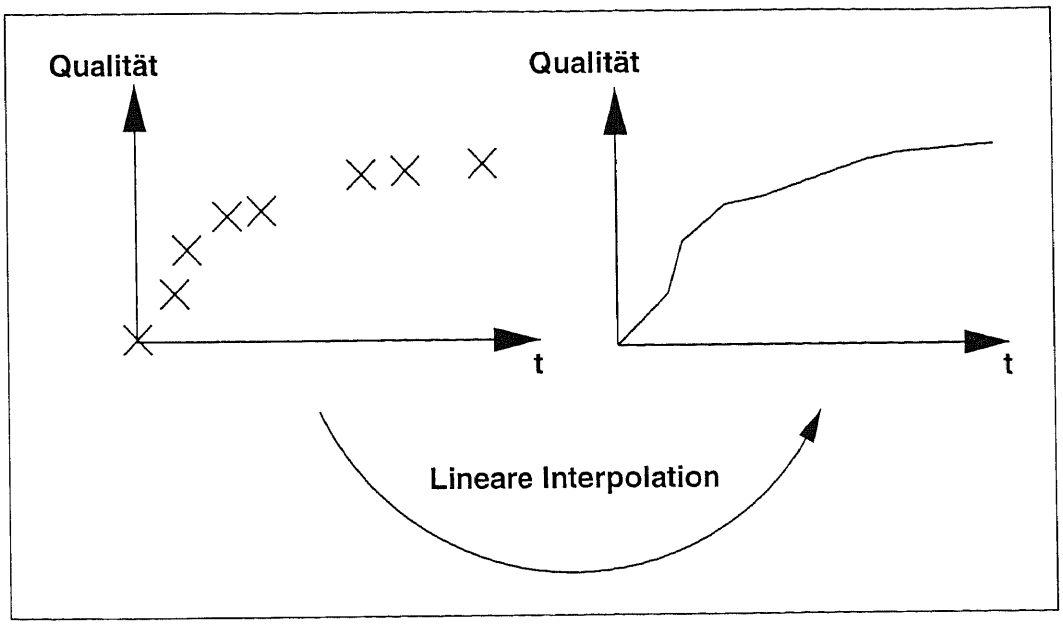

Abbildung 3.24: Annäherung durch lineare Interpolation

$$PP(t) := q_i + (t - t_i)\frac{q_{i+1} - q_i}{t_{i+1} - t_i} \quad \text{für} \quad i \quad \text{mit} \quad t_i \leq t < t_{i+1}. \tag{3.38}$$

Die optimale Kombination zweier Algorithmen kann bei abschnittsweise definierten Funktionen nicht (wie z.B. oben bei der vollständigen linearen Regression) als geschlossene Lösung angegeben werden; d.h. es muß also ein zweiter Approximationsschritt stattfinden, der die Rechenzeit erhöht und die vorher gewonnene Genauigkeit zumindest teilweise wieder einbüßt. Dieser Nachteil manifestiert sich auch bei der Bestimmung des Gesamtprofils, das sich in der Regel nur über Stichproben annähern läßt. Ein Beispiel wie die Verarbeitung erfolgen kann, findet sich in (Baus & Beckert, 1998). Obwohl bei diesem Verfahren also durchaus Probleme auftreten können, werden die Resultate der Zeitverteilung dennoch im allgemeinen besser ausfallen als bei den vorher vorgestellten Regressionsmethoden.

Bisher wurde nur der Fall der Kombination von *zwei* Anytime-Algorithmen betrachtet; normalerweise sind jedoch mehr, allgemein n Algorithmen zu verknüpfen. Zilberstein schlägt die Lösung durch ein System mit $n-1$ Gleichungen von (vgl. Abschnitt 2.6.3) vor. Die Verwendung von Gesamtprofilen in einem von den zu kombinierenden Performanzprofilen aufgespannten binären Baum wird in (Baus & Beckert, 1998) als Alternative zu Zilbersteins Ansatz zur Zeitverteilung auf n Algorithmen vorgestellt.

3.4.2.2 Die Hill-climbing-Methode

Schon Zilberstein hatte als eine Möglichkeit zur Abarbeitung mehrfachauftretender Subkomponenten ein Hill-climbing-Verfahren verwendet (vgl. Abschnitt 2.6.3). Dieser Algorithmustyp läßt sich, wie im folgenden beschrieben wird, aber ganz allgemein sehr gut für die Optimierung der Zeitvertei-

lung bei mehr als zwei Performanzprofilen einsetzen. Sowohl vollständige als auch insbesondere die genauere abschnittsweise Regression können verarbeitet werden. Ein Hill-climbing-Algorithmus startet dabei mit einer initialen Verteilung und versucht, diese durch schrittweise Änderung der Zusammensetzung zu verbessern bis festgelegte Abbruchkriterien erreicht werden. Es kann grundsätzlich nicht ausgeschlossen werden, daß das Verfahren nicht in einem globalen Optimum, sondern in einem lokalen Extremum terminiert. Wie die schrittweise Modifikation schon vermuten läßt, kann Hill-climbing leicht als Anytime-Algorithmus implementiert werden.

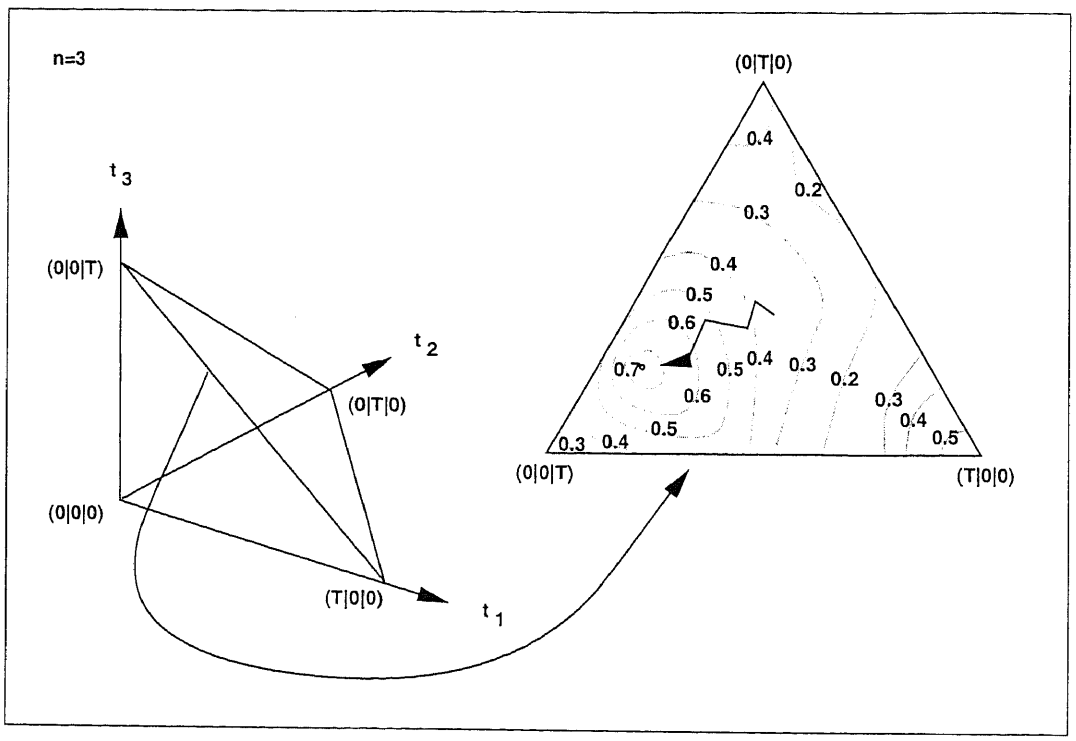

Abbildung 3.25: Drei-dimensionales Hill-climbing

Die Fläche in Abb. 3.25 stellt für ein drei-dimensionales Beispiel den Bereich dar, in dem bei der – vollständigen – Verteilung einer Gesamtzeit t_{ges} die optimale Lösung liegt. Auf der rechten Seite der Abbildung ist dieselbe *Zeitscheibe* mit Qualitätsisobaren zu sehen. Diese Linien geben also Orte gleicher Gesamtqualität an, der eine monotonie-erhaltende Verknüpfung $\circ$ der Profile zugrunde liegt:

$$Q_{ges} = PP_1(t_1) \circ PP_2(t_2) \circ \cdots \circ PP_n(t_n) \quad \text{mit} \quad t_{ges} = \sum t_i. \tag{3.39}$$

Als Startpunkt wird eine Gleichverteilung der Zeit angenommen, die im obigen Beispiel dem Mittelpunkt der Fläche entspricht; für n Algorithmen erhält man:

$$t_i = \frac{t_{ges}}{n} \quad \forall i \in \{1, \cdots, n\}. \tag{3.40}$$

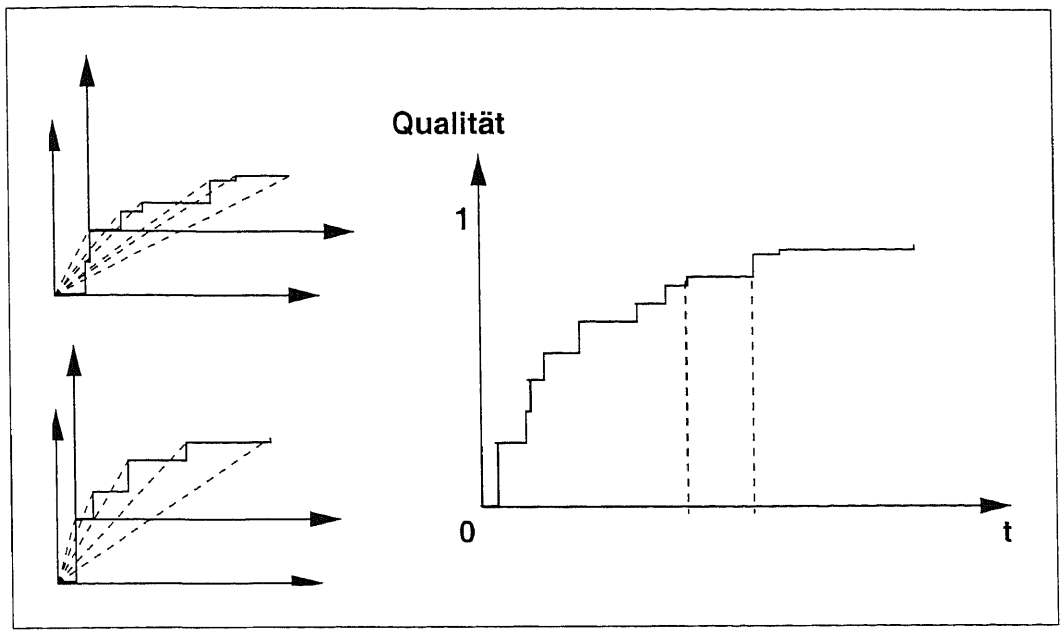

Abbildung 3.26: Das Treppenstufen-Verfahren

Von dieser Initiallösung ausgehend, wird der Verbesserungsschritt in die Richtung getan, die den größten Zuwachs an Qualität verspricht; d.h. der Algorithmus, der durch das entsprechende Performanzprofil repräsentiert wird, erhält mehr Zeit zugeteilt, alle anderen Algorithmen weniger. Dieser Schritt wird solange in dieselbe Richtung ausgeführt, bis keine Verbesserung mehr eintritt und eine neue beste Richtung bestimmt wird. Ist dies nicht möglich, wird die Schrittweite modifiziert. Die Abbruchkriterien sind wie folgt festgelegt:

- Einem der Performanzprofile wurde die gesamte Zeit zugewiesen.
- Schrittweite und/oder Qualitätsänderung haben ihre Toleranzgrenze erreicht.

Um zu berücksichtigen, daß keinem der Module negative Zeit zugeordnet werden kann – dies entspräche im drei-dimensionalen Fall einem Verlassen der Fläche über deren Rand –, wird die der besten Richtung zuzuschlagende Zeitspanne allen anderen Algorithmen entsprechend deren prozentualen Anteile an der Verteilung abgezogen. Wenn t_{alt_i} die dem Modul i im vorangegangenen Zyklus zugeteilte Zeit ist, ergibt sich somit für die Umverteilung von *step* Zeiteinheiten auf das Modul *dir*:

$$step_{aktuell} = min\left(step, \sum_{i \neq dir} t_{alt_i}\right), \tag{3.41}$$

$$s_i = \begin{cases} -\frac{t_{alt_i}}{\sum_{i \neq dir} t_{alt_i}} \cdot step_{aktuell} & \text{, falls } i \neq dir, \\ step_{aktuell} & \text{, falls } i = dir \end{cases} \tag{3.42}$$

$$t_{neu_i} = t_{alt_i} + s_i. \tag{3.43}$$

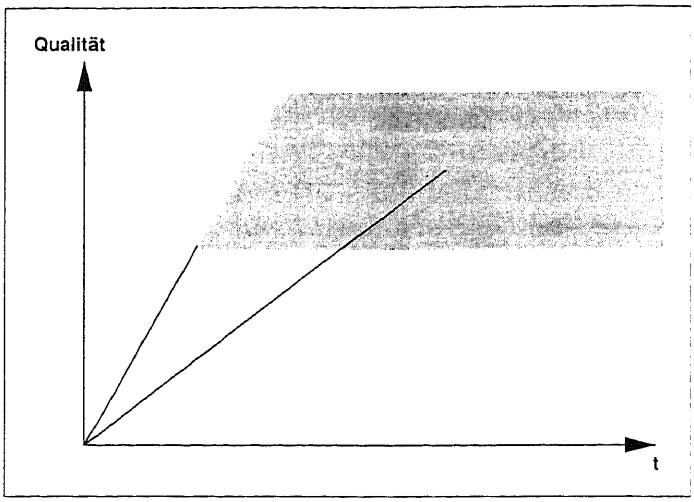

Abbildung 3.27: Problemfall *kritischer Bereich*

Der hier vorgestellte Hill-climbing-Algorithmus hat gegenüber dem von Zilberstein vorgeschlagenem – der Zeiteinheiten immer nur zwischen den zwei Modulen umschichtet, bei denen der Qualitätsgewinn maximal ist – besonders bei großen n eine bessere Laufzeit, da an Stelle von $2(n-1)!$ (jedes Modul mit jedem anderen in beide Richtungen) probeweisen Umverteilungen pro Verbesserungsschritt, hier nur n vorgenommen werden müssen.

3.4.2.3 Die Treppenstufen-Methode

Als letztes Verfahren zur Optimierung der Zeitverteilung bei Anytime-Systemen soll die Treppenstufen-Methode vorgestellt werden. Sie greift direkt auf die vorhandenen Profildaten zu, ohne daß (gegebenenfalls aufwendige) Umformungen notwendig sind. Dabei geht sie korrekterweise davon aus, daß zwischen zwei Stützpunkten *keine* Veränderung der Qualität erfolgt. Dies entspricht der Konstruktion der Performanzprofile in der Anytime-System-Shell. Als einziges hier beschriebenes Verfahren ist die Treppenstufen-Methode nicht profil- sondern stützstellen-orientiert, was bei akquirierten PP eventuell zu erhöhtem Rechenaufwand führen kann.

Die prinzipielle Vorgehensweise ist problemlos anytime-kompatibel: Anhand eines Kriteriums, wie z.B. des relativen Qualitätszuwachses mit der Zeit, ausdrückbar durch die Steigung der Treppenstufenfunktion, werden *alle* Stützstellen *aller* n Profile bewertet, die innerhalb der vorhandenen Zeit liegen, also alle Elemente der Menge

$$M = (t_0, q_0)^{PP_i} \ldots . (t_j, q_j)^{PP_i} \quad \text{mit} \quad t_j^{PP_i} \leq t_{rest} \quad \text{und} \quad i = 1, \ldots, n. \tag{3.44}$$

Es kann nun ein optimales Intervall $[t_0, t_j]^{PP_i}$ der ersten j+1-Stützpunkte des i-ten-Performanzprofils bestimmt werden. Während PP_i um diesen optimalen Initialbereich verkürzt in den Nullpunkt verschoben wird, werden die Stützstellen $(t_0, q_0)^{PP_i}, \cdots, (t_j, q_j)^{PP_i}$ an das zu konstruierende Gesamtprofil *angehängt* (vgl. Abb. 3.26). Dazu wird $(t_0, q_0)^{PP_i}$ mit dem jeweils letzten

Punkt des Gesamtprofils identifiziert. Nach einer Aktualisierung der verbliebenen Restzeit $t_{rest}^{neu} \leftarrow t_{rest}^{alt} - t_j^{PP_i}$, beginnt das Verfahren von vorne, wobei außer PP_i nur diejenigen Profile neu bewertet werden müssen, bei denen das lokale optimale Intervall länger als die noch zu verteilende Zeit ist, also $t_j^{PP_{k \neq i}} > t_{rest}$ gilt. Bei jedem Durchlauf wird demnach ein Teil eines Performanzprofils abgeschnitten und somit das Ausgangsproblem reduziert.

Problematisch bei dieser Methode kann der Vergleich des Kriteriums sein, wie Abb. 3.27 am Beispiel der Steigung illustriert. Was ist besser: Eine stärkere Steigung oder eine höhere Endqualität?

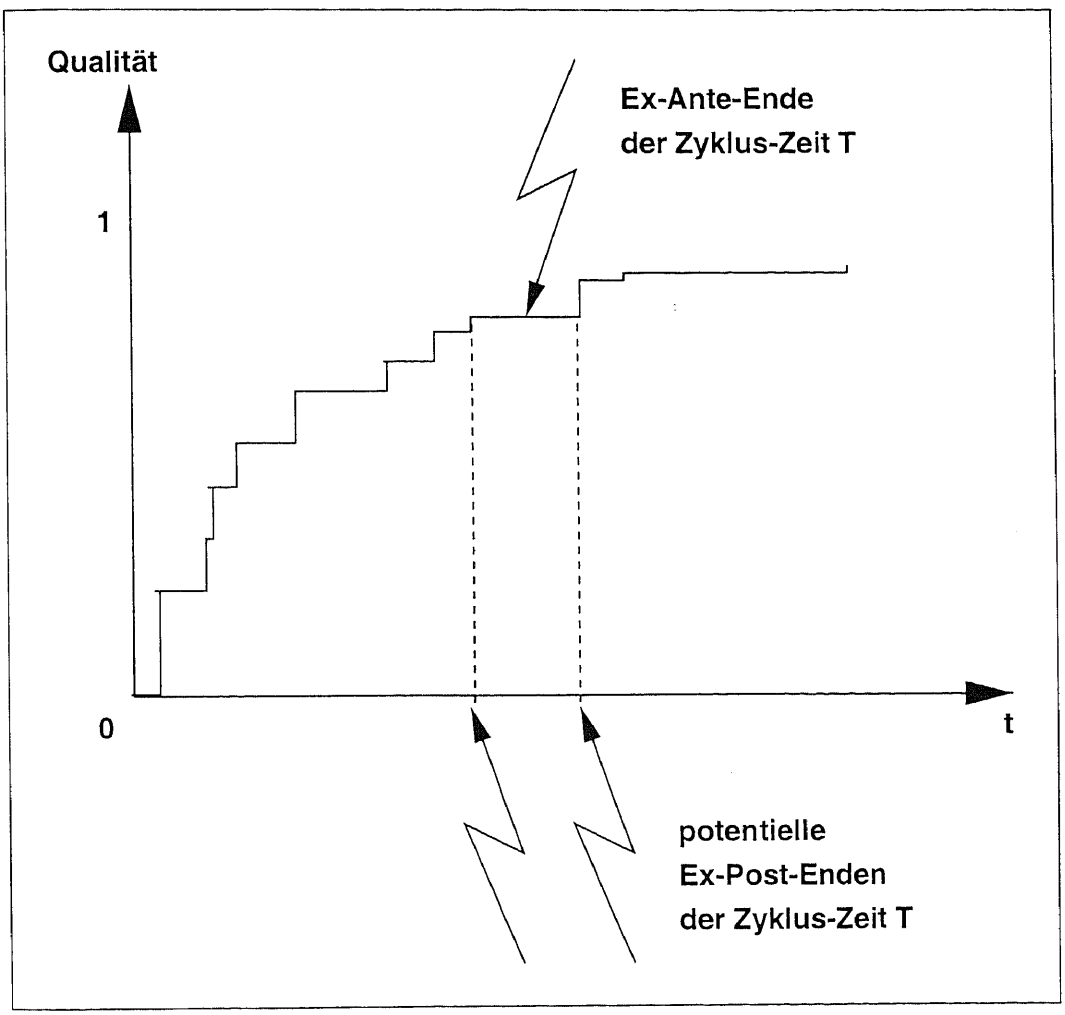

Abbildung 3.28: Problemfall *ununterbrechbare Transaktion*

Ebenso schwierig scheint es zu sein, wenn die Zykluszeit nicht vollständig verteilt werden kann, da ihr Ende in eine ununterbrechbare Transaktion fallen würde (vgl. Abb. 3.28). In diesem Fall existiert jedoch eine pragmatische Lösung: Da die Zykluszeit ja keine zwingend festgelegte Größe sein muß, sondern optimalerweise dynamisch vor jedem Zyklus bestimmt wird, kann sie – im Rahmen einer gewissen Schwankungsbreite variiert werden, so daß das resultierende Gesamtprofil etwas kürzer oder länger als ursprünglich geplant wird.

Methode	Vorteile	Nachteile
Regression (vollständig)	• genau für bestimmte Profilklassen • schnell für kleine Profilzahl	• aufwendig für große Profilzahl • eventuell grobe Näherung
Regression (abschnittsweise)	• schnell bei wenigen Stützstellen • relativ genau	• langsam bei vielen Stützstellen • aufwendig für große Profilzahl
Hill-climbing	• für große Profilzahlen geeignet • verschiedene Varianten möglich • anytime-fähig	• erhöhte Laufzeit bei wenigen Profilen • Standardprobleme (lokale Extrema etc.)
Treppenstufen	• genaueste Repräsentation • gute Annäherung an optimale Lösung • Laufzeit *nicht* abhängig von Profilzahl • anytime-fähig	• Laufzeit abhängig von Stützstellenzahl • vorgegebene Zykluszeit kann verändert werden

Tabelle 3.1: Vergleich der verschiedenen Methoden zur optimalen Zeitverteilung auf Anytime-Algorithmen

Die geschilderten Probleme zeigen, daß natürlich auch mit dem Treppenstufen-Verfahren nur Näherungslösungen erreicht werden, da das Grundproblem der optimalen Zeitverteilung auf n Anytime-Algorithmen NP-vollständig ist.

3.4.2.4 Der Vergleich der Methoden

Zum Abschluß dieses Themas sollen die vorgestellten Methoden noch einmal mit ihren jeweiligen Vor- und Nachteilen, die in Tab. 3.1 zusammengefaßt sind, besprochen werden.

Die vollständige Regression ist nur dann sinnvoll anzuwenden, wenn die Performanzprofile der beteiligten Algorithmen gut durch Funktionen angenähert werden können. Ist dies der Fall, sind für eine kleine Modulzahl schnelle und genaue Verteilungen erreichbar. Präziser ist zwar die abschnittsweise Variante, doch ist hier der Rechenaufwand erheblich, da er von der Zahl der Profile *und* der Stützstellen abhängig ist. Hill-climbing-Algorithmen sind erprobte Verfahren, die viele Variationsmöglichkeiten bieten und auch mit großen Datenmengen schnell und gut zurechtkommen. Ihr Nachteil besteht im Risiko des Auftretens von Standardproblemen, wie dem *„Hängenbleiben"* in lokalen Extrema. Ebenso wie Hill-climbing ist die Treppenstufen-Methode anytime-fähig. Sie ist von den vorgestellten und implementierten Methoden

die beste, sowohl hinsichtlich ihrer eigenen wie der Systemperformanz: Das Verfahren nutzt die Gegebenheiten der Anytime-System-Shell optimal aus und kann trotz bzw. gerade wegen der Stützpunkt-Orientierung sowohl kleine als auch große Anzahlen von Performanzprofilen schnell verarbeiten. Dabei ist das Resultat sehr häufig optimal.

3.5 Zusammenfassung

In diesem Kapitel wurde eine ressourcensensitive Architektur vorgestellt, auf der die Konstruktion des beschränkt-optimalen Lokalisationsagenten BOLA im folgende Kapitel fußt.

Um die Verarbeitung modularer Systemstrukturen zu ermöglichen, wurde die Architektur als rekursives Shell-System konzipiert. Sie erfüllt die Anforderungen der beschränkten Optimalität und ist jederzeit unterbrechbar. Somit können prinzipiell beliebige Gastsysteme ohne großen Aufwand zu ressourcenadaptierenden Anytime-Systemen erweitert werden.

Die durch das Shell-Konzept verwirklichte strikte Trennung von Modul- und Anytime-Kontrolle ist die Basis für eine Reihe positiver Eigenschaften:

- *Struktur der Anytime-Kontrolle:* Neben dem kompletten Aufbau des lokalen Ressourcenbaumes bietet sich die sukzessive Konstruktion an.
- *Gewichtung der beteiligten Algorithmen:* Aus einer statischen Initialgewichtung kann eine optimale *erlernt* werden.
- *Optimierung der Zeitverteilung:* Mehrere Methoden können angewandt werden, um die Kombination von Anytime-Algorithmen so effizient wie möglich zu gestalten.

Die Fähigkeiten dieses innovativen Konzepts einer ressourcensensitiven Architektur in Form einer Anytime-System-Shell werden im nächsten Kapitel 4 zur Konstruktion des beschränkt-optimalen Lokalisationsagent BOLA verwendet, der *Wo-Fragen* eines menschlichen Benutzers bei variierender Ressourcenlage adäquat beantworten und dabei jederzeit ein approximatives Resultat liefern kann.

Kapitel 4

Ein Lokalisationsagent als Gastsystem

Am Beispiel des *beschränkt-optimalen Lokalisationsagenten* BOLA wird in diesem Kapitel die Integration der zuvor entwickelten Verfahren zur Generierung natürlichsprachlicher Raumbeschreibungen unter Ressourcenbeschränkungen gezeigt.

Dazu wird zuerst die Konstruktion anytime-fähiger Subsysteme zur Ermittlung eines besten Referenzobjektes bzw. zur Berechnung räumlicher Relationen (einschließlich linguistischer Hecken) erläutert und danach deren Verknüpfung innerhalb einer Anytime-System-Shell zu einem Gesamtsystem[1].

Die Verarbeitung der (Sub-)Systeme erfolgt in nebenläufigen Prozessen, die jederzeit unterbrechbar sind. Ferner können zu jedem Zeitpunkt inkrementell aufgebaute (Zwischen-)Ergebnisse angefordert werden, deren Qualität mit den investierten Ressourcen (hier im wesentlichen *Zeit*) monoton wächst. Zur Optimierung der Ressourcenverteilung zur Laufzeit werden dabei sowohl die Ergebnisse früherer Systemdurchläufe als auch der aktuellen Zustand des Systems berücksichtigt, was sich in unterschiedlichen Berechnungsverfahren – je nach vorhandenen Ressourcen – niederschlägt. Somit kann die Gesamtperformanz des Systems optimiert und von einem *ressourcenadaptierenden Verhalten* gesprochen werden.

Das Gesamtsystem zeichnet sich also durch eine komplexe Interaktion verschiedener Prozesse aus und orientiert sich damit an neueren Erkenntnissen zur Steuerung kognitiver Prozesse (Kluwe, 1997), die für das flexible Verhalten des Menschen in unterschiedlichen Situationen eine *einheitliche zentrale Komponente* für wenig wahrscheinlich erachten. Die bislang erreichte partielle Adäquatheit des vorliegenden Ansatzes konnte in ersten kognitionspsychologischen Experimenten gezeigt werden.

Wie erwähnt konzentriert sich die vorliegende Arbeit auf die Bewertung von Referenzobjekten sowie die Generierung räumlicher Relationen und linguistischer Hecken. Daraus ergibt sich für die Systemstruktur von BOLA auf

[1]Weitere wichtige Teilaufgaben wie eine Kontrolle des Hörerverständnisses mittels einer Antizipationsrückkopplungsschleife (Schirra, 1994; Blocher & Schirra, 1995) oder eine Verbalisierungskomponente werden hier nicht betrachtet, so daß die im folgenden dargestellten Resultate sämtlich vorsprachlichen Charakter haben.

oberster Ebene unmittelbar eine Aufteilung in zwei Subsysteme, wie in Abb. 4.1 dargestellt.

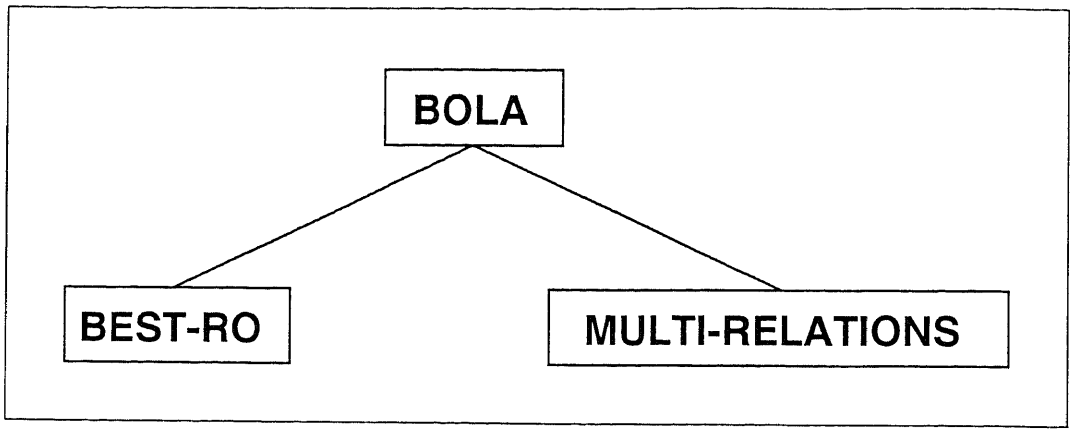

Abbildung 4.1: Homogene hierarchische Systemstruktur von BOLA

Das Subsystem BEST-RO dient der Ermittlung potentieller Referenzobjekte sowie der Berechnung ihrer Qualität, während parallel dazu das zweite Subsystem MULTI-RELATIONS für die Generierung räumlicher Relationen zwischen einem zu lokalisierenden Objekt und diesen Referenzobjekten zuständig ist.

Übertragen auf die Konstruktion der Anytime-System-Shell des Gesamtsystems BOLA bedeutet dies, daß die oberste RAMI zwei reale Tasks – in einer homogenen hierarchischen Systemstruktur – besitzt, die ihrerseits wieder RAMIs sind. Es werden hier also noch keine elementaren Lösungsstrategien verwendet. Konkrete Berechnungen finden erst auf untergeordneten Ebenen statt und werden später erläutert.

Die beiden Subsysteme BEST-RO und MULTI-RELATIONS werden in den nun folgenden Abschnitten näher beschrieben.

4.1 Ein Gastsystem zur Bewertung von Referenzobjekten

Die Vorgehensweise zur Ermittlung und Qualitätsbewertung von potentiellen Referenzobjekten lehnt sich stark an Gapp (1997) an (vgl. Abschnitt 1.3). Insbesondere wurden seine Merkmale für Referenzobjekte und die Methoden ihrer Verrechnung zur Generierung eines Qualitätsmaßes übernommen. Die wesentlichen Unterschiede bestehen einerseits in der konsequenten Berücksichtigung unterschiedlicher Idealisierungsniveaus und andererseits in der Möglichkeit, die Verfahren jederzeit zu unterbrechen und Zwischenergebnisse abfragen zu können. Beides ist für die Verwendung in der Anytime-System-Shell erforderlich, so daß die ursprünglichen Verfahren dahingehend ergänzt werden mußten.

4.1.1 Differenzierung über Eigenschaften potentieller Referenzobjekte

In einer Initialisierungsphase werden zunächst die Objekte bestimmt, die überhaupt als Referenzen in Frage kommen. Die in Abschnitt 1.3.1 thematisierte Nähe-Umgebung wird durch einen von der Größe des zu lokalisierenden Objektes abhängigen Zugriff auf einen Range-Tree realisiert, in dem alle Objekte bezüglich ihrer Koordinaten abgelegt sind[2]. Die so gefundenen Objekte werden als *potentielle Referenzobjekte* bezeichnet[3].

Folgende Eigenschaften der potentiellen Referenzobjekte werden berücksichtigt:

- *Größe* `size` (0,2)
- *Distanz* `distance` (0,5)
- *Farbe* `color` (0,2)
- *Vorerwähntheit* `mentioned` (0,1)
- *Intrinsische Front* `intrinsic-front` (0,2)
- *Mobilität* `movability` (0,3)
- *Funktionale Abhängigkeit* `functional-dependency` (0,25)
- *Störobjekte* `intervening-objects` (0,25)
- *Identifizierbarkeit* `referentiality` (0,35)

Neben den Bezeichnern der einzelnen Tasks ist in Klammern jeweils eine initiale Gewichtung angegeben, die für erste Systemläufe zur Ermittlung der Ressourcenverteilung verwendet werden kann. Später sollte sie durch erlernte, wenn möglich durch experimentelle Befunde abgesicherte Werte ersetzt werden, um den kognitiven Ansprüchen des Systems gerecht zu werden[4].

4.1.2 Differenzierung über unterschiedliche Idealisierungen

Prinzipiell werden alle Eigenschaften hinsichtlich der in Abschnitt 1.3.2 beschriebenen Idealisierungsklassen berechnet.

Dabei wird insbesondere auch zwischen zwei- und dreidimensionalen Domänen unterschieden wie z.B. einer Deutschlandkarte zur Lokalisation von Bundesländern und Städten (vgl. Abb. 4.2) oder einem Büro als Indoor-Szenario

[2]Diese Vorgehensweise könnte in realen Anwendungen z.B. durch ein geeignetes Bildanalyseverfahren ersetzt werden.

[3]Sollte die beschriebene Nähe-Umgebung keine Objekte enthalten, wird sie entsprechend ausgeweitet.

[4]Dies ist – wie in Abschnitt 4.4 dargelegt wird, für einzelne Aspekte der Berechnung räumlicher Relationen bereits geschehen.

mit Computer, Monitor, Maus, Tastatur etc. (vgl. Abb. 4.3). Wieder andere Domänen können in beiden Dimensionen sinnvoll dargestellt und untersucht werden: Eine Wegbeschreibung kann sich ebenso gut auf eine reale, dreidimensionale Stadt wie auf ihre zweidimensionale Projektion in einem Plan beziehen.

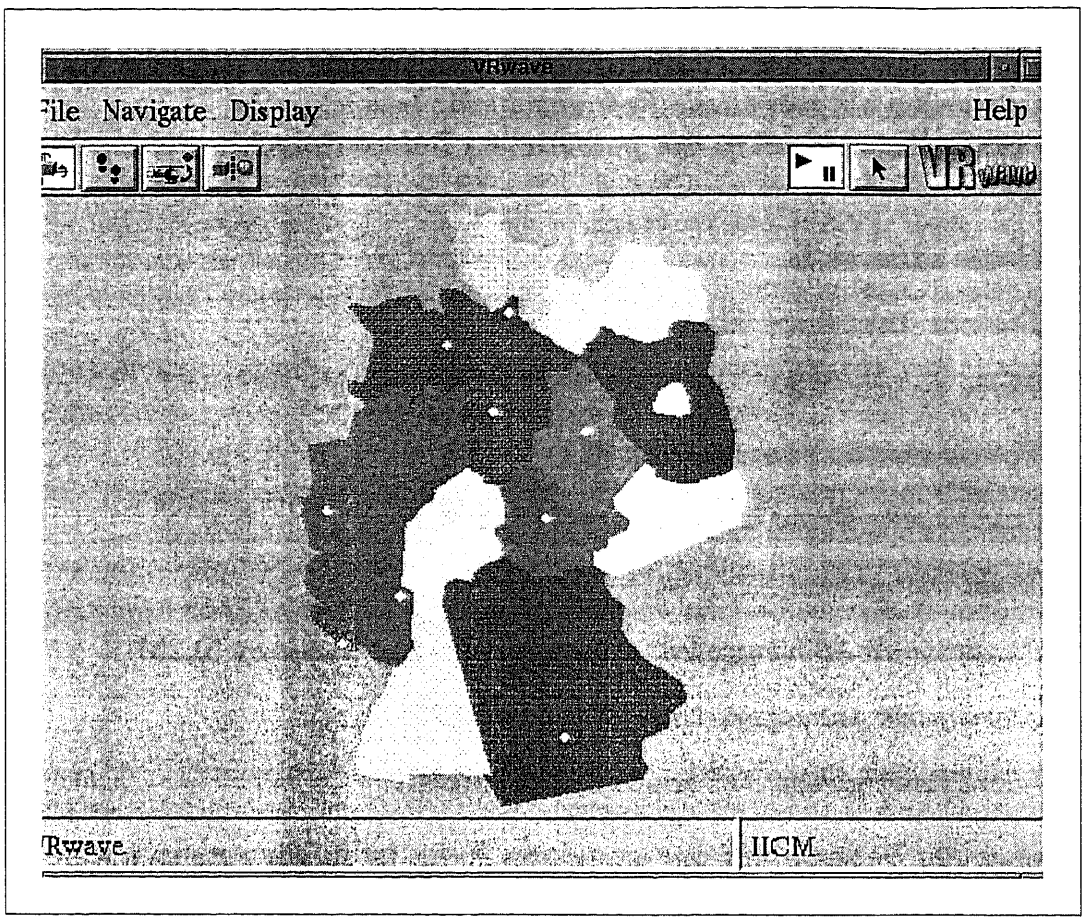

Abbildung 4.2: 2-dimensionale Beispieldomäne: Deutschlandkarte

Genauer gesagt werden potentielle Referenzobjekte in zwei Dimensionen dargestellt:

- *2-dimensional* `2d` (0,5)
- *3-dimensional* `3d` (1,0)

Dabei werden jeweils vier Idealisierungen realisiert:

- *Schwerpunkt* `pt` (0,25)
- *Achsenparalleler umschreibender Quader* `b-axis` (0,5)
- *Minimaler umschreibender Quader* `b-mini` (0,75)
- *Vollständiges geometrisches Modell* `full` (1,0)

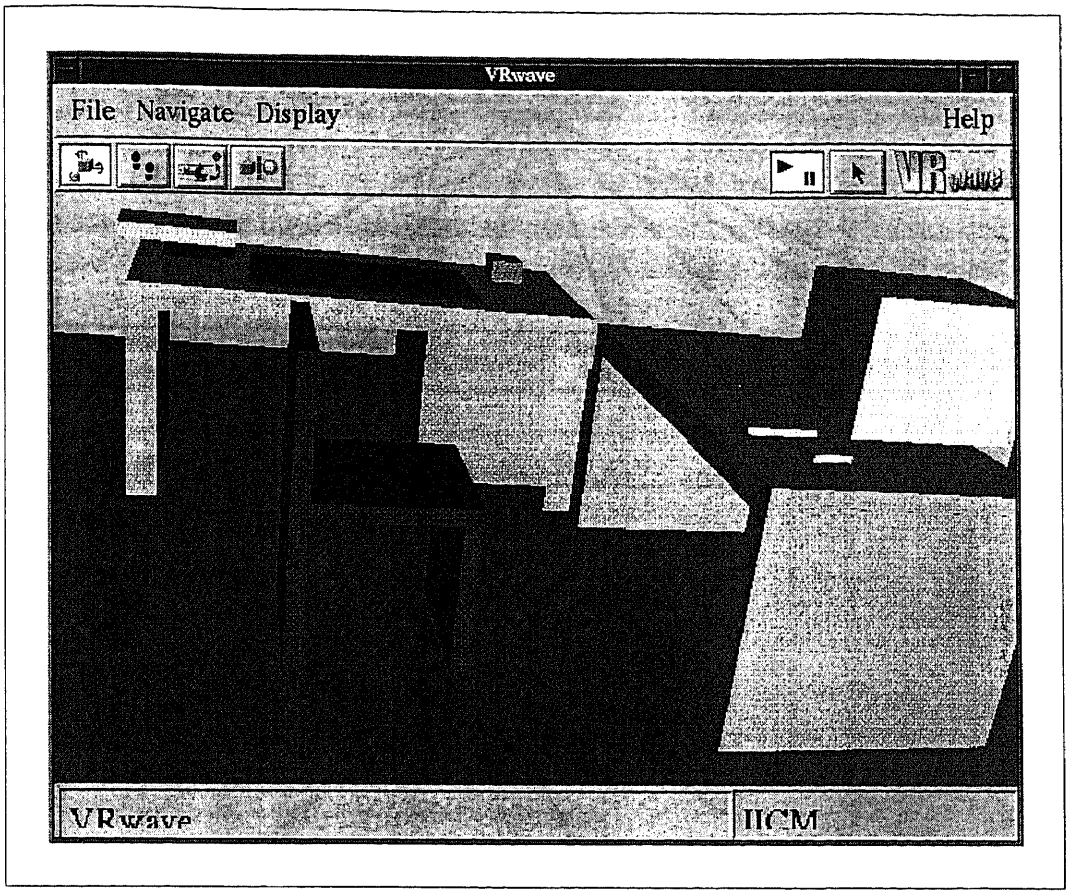

Abbildung 4.3: 3-dimensionale Beispieldomäne: Büro

Selbstverständlich müssen hier einige Einschränkungen gemacht werden: So ist z.B. zu berücksichtigen, daß bei Aufgaben, die etwa auf Kartendarstellungen basieren, schon im zweidimensionalen Raum das Optimum erreicht wird und eine dreidimensionale Bearbeitung keinen Sinn macht. Ferner können nicht alle Eigenschaften in allen Ausprägungen untersucht werden: Das einfachste Beispiel ist hier die Größe, die bei einer Schwerpunktdarstellung irrelevant ist. Dies wird natürlich bei Berechnung der entsprechenden Gesamtqualität beachtet.

Letztlich besteht die Möglichkeit, Abhängigkeiten oder die Weiterverwendung von Teil- oder Zwischenergebnissen in der Modulkontrolle zu verankern. So kann beispielsweise die im dreidimensionalen Fall für die Idealisierungsklasse *achsenparalleler umschreibender Quader* gefundene Menge potentieller Störobjekte als Ausgangsbasis für die Analyse höherwertiger Idealisierungen dienen.

Für die in Abb. 4.4 dargestellte homogene hierarchische Systemstruktur von BEST-RO gibt es also jeweils bis zu acht unterschiedliche Berechnungsverfahren, deren Ressourcenbedarf und Komplexität sowie Ergebnisqualität und -entwicklung sehr stark variieren. Dadurch können je nach Bedarf sehr schnelle oder sehr gute Resultate generiert werden.

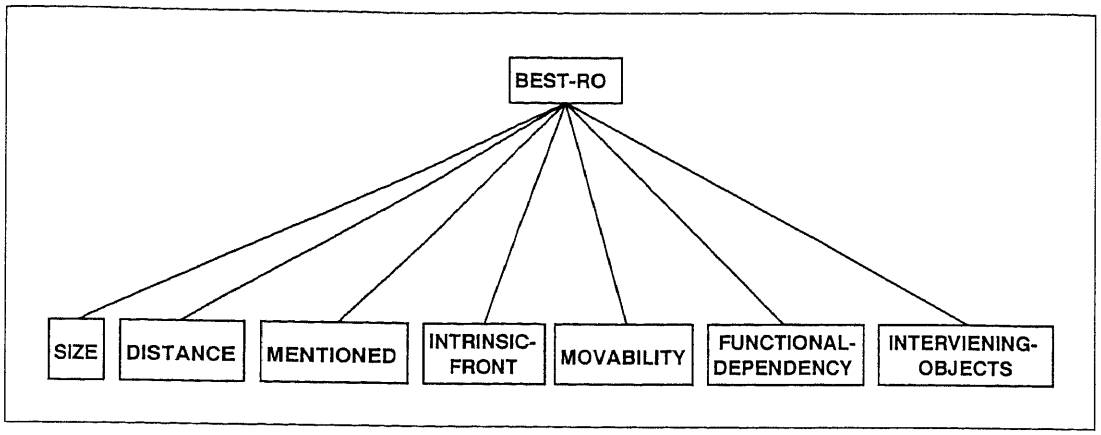

Abbildung 4.4: Homogene hierarchische Systemstruktur von BEST-RO

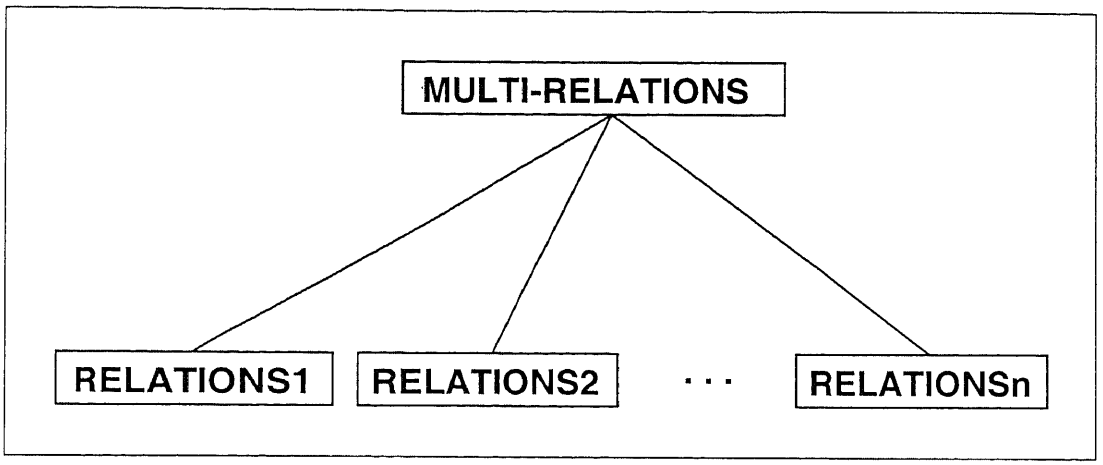

Abbildung 4.5: Homogene hierarchische Systemstruktur von MULTI-RELATIONS

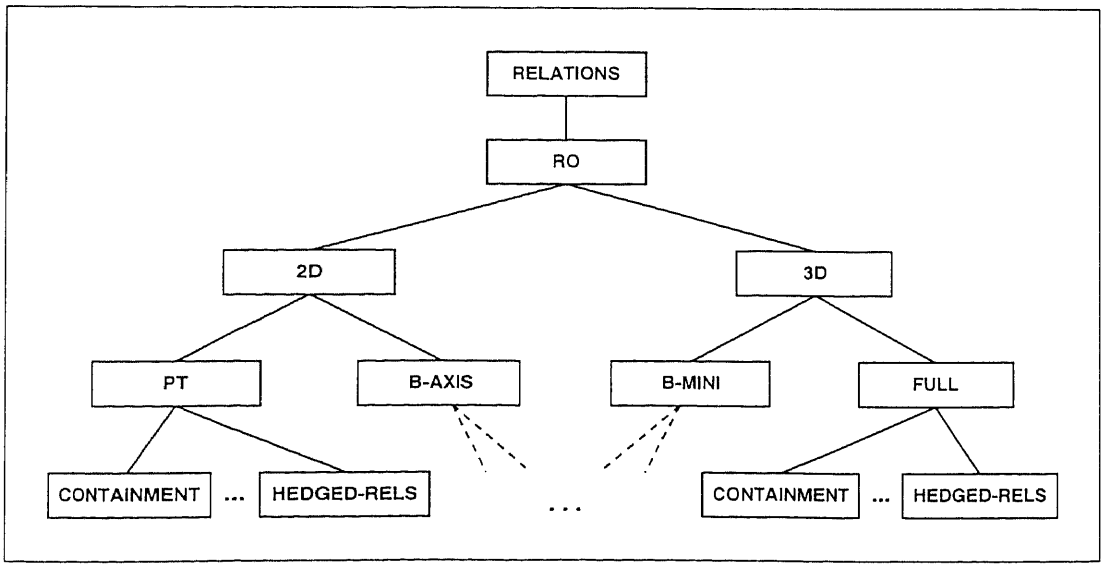

Abbildung 4.6: Homogene hierarchische Systemstruktur von RELATIONS

Jedem Merkmal eines potentiellen Referenzobjektes ist dabei ein realer Task zugeordnet, dessen Struktur sich über die Dimensionen und die Idealisierungen aufgliedert. Dabei hängt die konkrete Ausprägung dieser Struktur letztlich von den relativen Gewichten der einzelnen Tasks ab.

Parallel zur Bewertung potentieller Referenzobjekte kann die Berechnung räumlicher Relationen ausgeführt werden.

4.2 Gastsysteme zur Berechnung räumlicher Relationen

Das zweite Subsystem der obersten RAMI berechnet die räumlichen Relationen zwischen dem zu lokalisierenden und *mehreren* potentiellen Referenzobjekten. Die Qualität der jeweils besten Relation ergibt zusammen mit der Bewertung der im vorigen Abschnitt beschriebenen Objektmerkmalen die Gesamtqualität der entsprechenden Raumbeschreibung. Das beste bisher erzielte Resultat steht jederzeit zur Verfügung.

Die RAMI dieses Subsystem MULTI-RELATIONS führt selbst keine konkrete Berechnung aus, sondern gliedert sich in einzelne RAMIs für RELATIONS-Subsysteme, eines für jedes potentielle Referenzobjekt. Abbildung 4.5 zeigt die so entstehende homogene hierarchische Systemstruktur. Betrachtet man die innerhalb eines dieser RELATIONS-Subsysteme stattfindende eigentliche Ermittlung der Lagebeziehung zwischen zu lokalisierendem und einem bestimmten Referenzobjekt genauer, so besitzt sie die komplexeste homogene hierarchische Systemstruktur aller verwendeten Subsysteme (vgl. Abb. 4.6).

Auch hier gilt, daß die tatsächliche Ausprägung in einem konkreten Systemlauf nicht unbedingt diese Reihenfolge der Aufgliederung besitzen muß.

4.2.1 Ontologie räumlicher Relationen

Neben der Identifikation desjenigen Referenzobjektes zu dem räumliche Relationen bestimmt werden sollen und den bereits weiter oben erwähnten Parametern *Dimension* und *Idealisierung* tritt nun mit dem *Relationen-Level* noch ein weiteres Merkmal. Damit wird die Qualitätsstufe definiert, auf der die Lagebeziehung zwischen den Objekten beschrieben werden soll. Ebenso wie die Verarbeitung des vollständigen geometrischen Modells eine qualitative Verbesserung gegenüber einer Punktrepräsentation darstellt, liefert eine komplexe Relation wie `links_oben_auf` eine erheblich genauere Angabe als etwa `links` allein. Wie schon in Abschnitt 1.4.2 beschrieben, überlappen sich die einzelnen Anwendbarkeitsräume nur zu einem Teil, so daß der resultierende Suchraum wesentlich kleiner wird. Dieser positive Effekt wird aber nur durch einen größeren Ressourceneinsatz erreicht.

Umgekehrt kann also über eine stufenweise Generierung von immer komplexeren räumlichen Relationen eine Grundlage für ressourcenadaptives Verhalten gelegt werden. Dabei darf natürlich die sprachliche Seite nicht außer acht gelassen werden: Die Kombination einzelner Relationen zu neuen, komplexeren muß sich (zumindest als Klasse) auch auf der linguistischen

Seite wiederfinden lassen. Die Problematik dieser Forderung wird deutlich, wenn man bedenkt, daß z.B. im Englischen zusammengesetzte Ausdrücke wie *„links oben"* im Gegensatz zum Deutschen ungebräuchlich sind.

Basierend auf den in Abschnitt 1.4.1 vorgestellten Kriterien wie essentielle Parameter (Distanz und Winkel), Komplexität der Relation (elementar oder zusammengesetzt) oder des zu lokalisierenden Objektes (2- oder n-Punkt-Relationen) kann eine Klassifikation der in der vorliegenden Arbeit betrachteten räumlichen Relationen vorgenommen werden. Sie berücksichtigt neben der Komplexität (die sich im Ressourcenbedarf wiederspiegelt) auch feste Abhängigkeiten bzw. Abarbeitungsfolgen, die in der Modulkontrolle kodiert sind: So kann beispielsweise eine zusammengesetzte gemischte Relation nur dann berechnet werden, wenn zuvor bereits mindestens je eine elementare winkel- und distanzabhängige Relation bestimmt wurde. Ähnliches gilt für linguistische Hecken, die schon bewertete Relationen modifizieren und die infolgedessen erst zu einem späteren Zeitpunkt betrachtet werden können. Abbildung 4.7 zeigt die Klassifikation inclusive aller notwendigen Abhängigkeiten.

Insgesamt gliedert sich das Kriterium *Relationen-Level* also wie folgt (die als Schwerpunkt der vorliegenden Arbeit originär entwickelten Berechnungsmethoden sind **hervorgehoben**):

- *Anwendbarkeit*
  - *2-Punkt-Relationen*
    - *binär*
      1. *Enthaltensein* `containment`
      2. *Halbraummodell* `hsm-model`
      3. *Winkelkombination im Halbraummodell* `combined-angle-hsm`
    - *gradiert*
      4. *Distanzabhängigkeit* `distance-rels`
      5. *Winkelabhängigkeit* `angle-rels`
      6. *Winkelkombination* `combined-angle-rels`
      7. *Geographische Relationen* `geographic3-rels`
      8. *Distanz- und Winkelkombination* `combined-mixed-rels`
      9. *Sonderfälle* `special-rels`
      10. *Kombinationen mit Sonderfällen* `combined-special-rels`
  - ***N-Punkt-Relationen***
    11. ***Endpunktorientierung*** `raw-path-rels`
    12. ***Stützpunktorientierung*** `base-path-rels`
    13. ***Kombination von n-Punkt-Relationen*** `combined-path-rels`
    14. ***Kombination mit 2-Punkt-Relationen*** `mixed-path-rels`
- ***Präzision***
  15. ***Applikation linguistischer Hecken*** `hedged-rels`

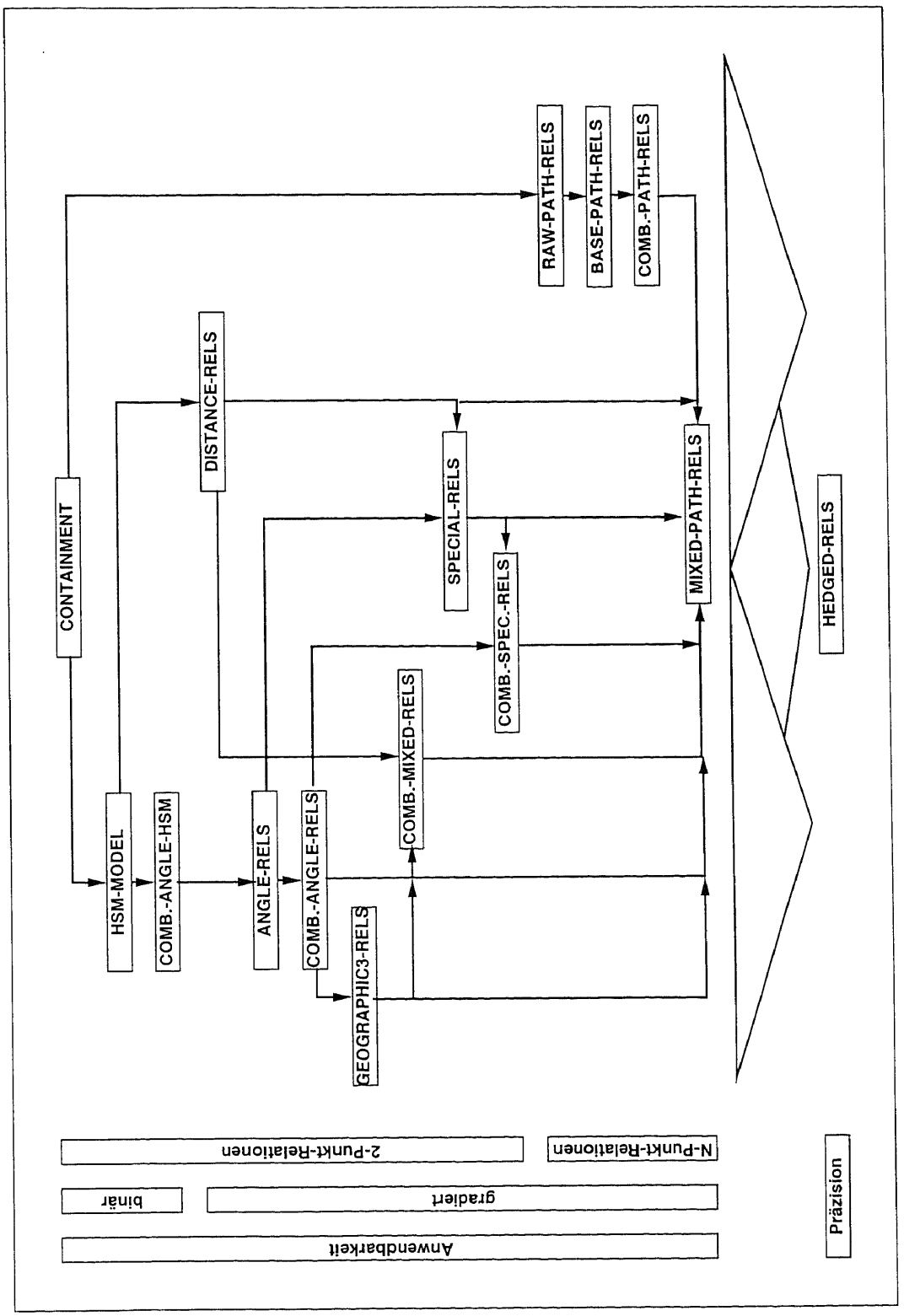

Abbildung 4.7: Klassifikation räumlicher Relationen nach Komplexität

Insgesamt können also (bis zu) fünfzehn Relationenklassen (einschließlich der linguistischen Hecken) für die unterschiedlichen Idealisierungen berechnet werden. Die Algorithmen auf den einzelnen Ebenen werden in den nächsten Abschnitten erläutert.

4.2.2 Berechnungsverfahren für räumliche Relationen und linguistische Hecken

Zu den im folgenden beschriebenen konkreten Berechnungsverfahren für räumliche Relationen und linguistische Hecken müssen zwei Bemerkungen gemacht werden:

Zum einem basieren alle Algorithmen –wie in Abschnitt 1.2 angedeutet – auf einem festen Referenzsystem, es wird also nicht nach der Gebrauchsart der Relationen unterschieden. Dieser Referenzrahmen wird durch die geometrische Modellierung vorgegeben. Um andere Referenzsysteme zu integrieren, muß lediglich eine Transformationsmatrix berechnet werden, die den Übergang von vorgegebenen zu einem beliebigen anderen Rahmen kodiert. Ein entsprechender zusätzlicher Parameter würde analog zu Dimension und Idealisierung integriert und erhöht die Komplexität der Gesamtberechnung.

Zum anderen liegt allen Verfahren die Idee zugrunde, daß zur Berechnung räumlicher Relationen zwischen zwei Objekten in der Regel nur die jeweils nächsten Punkte berücksichtigt werden müssen. Dies trifft insbesondere für konvexe Objekte zu. Eine abschließende Analyse nicht-konvexer Objekte wurde noch nicht erstellt. Jedoch bieten sich die in Abschnitt 1.4.4 beschriebenen n-Punkt-Relationen hier für eine Lösung an. So kann man in zweidimensionalen Fall den Umriß eines solchen Referenzobjektes als geschlossene Trajektorie ansehen und die entsprechenden Berechnungsmethoden anwenden. Erste Versuche hierzu zeigten ermutigende Resultate. Eine Übertragung auf den dreidimensionalen Fall steht noch aus.

Für alle Relationen, die als gradierte Konzepte realisiert wurden, wird als Resultat der Präzisionsgrad zurückgeliefert. Dies garantiert eine Vergleichbarkeit bis hin zu Modifikationen durch linguistische Hecken. Allerdings mußten aus diesem Grund einige der Berechnungsverfahren von Gapp abgeändert werden. Der jeweiligen Anwendbarkeitsgrad wird als Zwischenergebnis für eine eventuelle spätere Verarbeitungen gespeichert.

Die einzelnen Berechnungsverfahren sind jeweils als ununterbrechbare Transaktionen realisiert, um eine problemlose Übergabe eventuell vorhandener Zwischenresultate sicherzustellen. Dies hat allerdings auf die jederzeitige Unterbrechbarkeit des Gesamtsystems – nach Beendigung einer Transaktion – keine negative Auswirkung, da die Algorithmen keine komplexen Berechnung durchführen und daher sehr schnell sind.

4.2.3 Differenzierung über die Komplexität räumlicher Relationen

4.2.3.1 Enthaltensein

Aus dem oben Gesagten geht hervor, daß zunächst die Lagebeziehung zwischen dem zu lokalisierenden Objekt und dem Referenzobjekt berechnet werden muß. Dazu werden die beiden nächsten Punkten np_{RO} und np_{LO} bestimmt (vgl. Abb. 4.8).

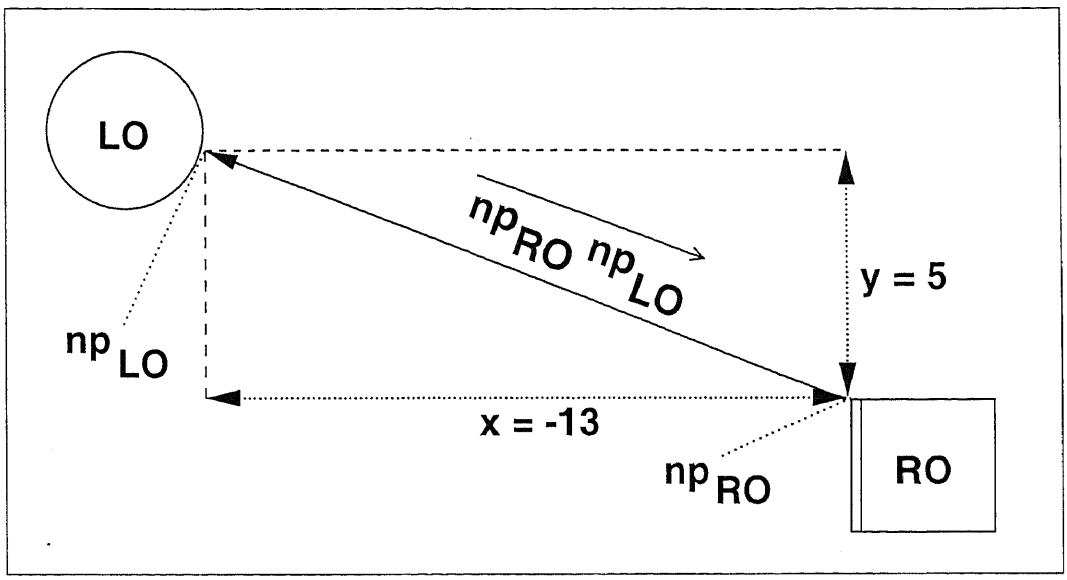

Abbildung 4.8: Die zueinander nächsten Punkten von RO und LO und andere Grundlagen der Relationenberechnung

Als erstes Teilergebnis kann hierbei festgestellt werden, ob das zu lokalisierende Objekt innerhalb oder außerhalb des Referenzobjektes liegt oder sich beide Objekte berühren. Dies korrespondiert mit den elementaren topologischen Relationen `inside`, `disjoint` und `meet` von Egenhofer (1991) und ermöglicht eine erste, noch sehr einfache Lokalisierung des LO falls nur eine äußerst kurze Zeitressource zur Verfügung steht.

Hieraus können jedoch nur rein qualitative Aussagen der Art *„Das LO ist im RO"* generiert werden. Eine Vergleichbarkeit ist weder bezüglich der Relationen untereinander noch für verschiedene Referenzobjekte möglich. Dies gilt auch für das Ergebnis des folgenden Verfahrens.

4.2.3.2 Halbraummodell

Auf der zweiten Stufe der Relationenberechnung wird der Vektor $\overrightarrow{np_{RO}np_{LO}}$ vom nächsten Punkt des RO np_{RO} zu dem des LO np_{LO}, wie in Abb. 4.8 zu sehen, bestimmt und näher betrachtet: Dieser wird im dreidimensionalen Raum durch drei kanonische Richtungen x, y, z ausgedrückt, die (im gegebenen oder transformierten Referenzsystem) mit den Relationenpaaren `links-rechts`, `vorne-hinten` und `unten-oben` assoziiert werden können. Damit

wird eine dreifache Zerlegung induziert, wobei die resultierenden Halbraum-Paare jeweils binären Charakter haben. Das Vorzeichen einer Vektor-Koordinate bestimmt dabei, welcher Halbraum relevant ist, während ihr Absolutbetrag die Stärke der Ausprägung angibt. Somit können auf diesem Level bereits erste Vergleiche bei *einem* Referenzobjekt erfolgen. Im zweidimensionalen Fall von Abb. 4.8 beträgt der x-Wert -13, der y-Wert 5. Damit sind die Halbräume `links` und `oben` bestimmt, wobei ersterer stärker ausgeprägt ist und als Zwischenresultat bereitgehalten wird. Bildet man den Quotienten aus der Länge pro Dimension zur absoluten Länge des Vektors, so ist auch eine Vergleichbarkeit zwischen *mehreren* Referenzobjekten gegeben, die jedoch nichts über die Qualität der Relation an sich aussagt. Somit können also unterschiedliche Relationenklassen nicht verglichen werden.

Die entsprechenden Werte werden für eine eventuelle Weiterverarbeitung gespeichert.

4.2.3.3 Winkelkombination im Halbraummodell

Falls das zu lokalisierende Objekt nicht *exakt* auf einer der Achsen des Referenzsystems liegt und also nur *eine* Richtung existiert, können die beiden besten Werte des Halbraummodells in einer ersten Variante zur Berechnung komplexerer Relationen kombiniert werden. (Dies entspricht etwa der Darstellung in Abb. 1.13.) Dazu wird das geometrische Mittel der bereits vorliegenden Werte gebildet. Das Resultat ist im Sinne des Halbraummodells vergleichbar.

4.2.3.4 Distanzabhängigkeit

Betrachtet man die Länge des Vektors $\overrightarrow{np_{RO}np_{LO}}$ bezüglich der Größe des Referenzobjektes, so können Anwendbarkeitsgrade für distanzabhängige Relationen berechnet werden (vgl. (Gapp, 1997)). Damit werden die zu Beginn erzielten Ergebnisse verfeinert, da sie nun beliebig vergleichbar sind.

Die folgende Relationen, deren Anwendbarkeitsfunktionen in Abb. 4.9 schematisch dargestellt sind, können berechnet werden:

- `in`: Wenn diese Relation erfüllt ist, so wird ihr Anwendbarkeitsgrad immer als optimal angesehen, d.h. eine Differenzierung wie sie im seltenen Fall einer teilweisen Durchdringung sinnvoll wäre, ist nicht vorgesehen. Die Berechnungsverfahren können aber problemlos in diese Richtung erweitert werden.

- `zentral`: Hiermit wird ein eingeschränkter Innenbereich definiert, der ein Drittel des gesamten Innenraumes einnimmt. Bezüglich des Anwendbarkeitsgrades gilt das zuvor Gesagte.

- `kontakt`: Das zu lokalisierende und das Referenzobjekt berühren sich: Der Anwendbarkeitsgrad ist optimal.

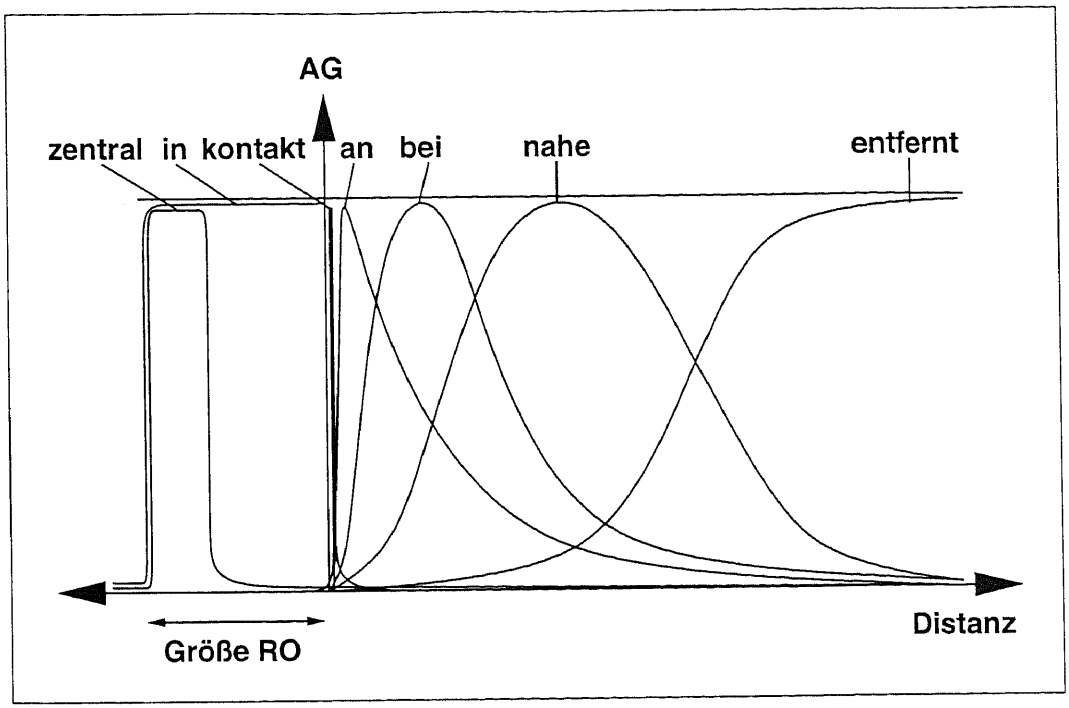

Abbildung 4.9: Verschiedene distanzabhängige Relationen

- an: Dies ist die erste echte gradierte Relation, nachdem den vorangegangenen ein Anwendbarkeitsgrad sozusagen zu Vergleichszwecken zugewiesen wurde. Die Relation drückt dabei eine unmittelbare Nähe zum RO aus, ohne daß ein Kontakt auftritt.

- bei: Im Gegensatz zu an ist eine deutliche Trennung der Objekte sichtbar, die Distanz nimmt zu.

- nahe: Die Distanz ist noch größer, aber das LO liegt immer noch in der Umgebung des Referenzobjektes.

- entfernt: Hierbei handelt es sich um eine Art *negative* Relation, die sich daraus ergibt, daß das zu lokalisierende Objekt sich eben nicht in der Nähe des RO befindet.

Wünschenswert wäre eine Rückführung der jeweiligen Anwendbarkeitsfunktionen auf eine gradierte Modifikation von Egenhofers topologischen Relationen, wie von Rupp (1996) dargelegt. Problematisch ist allerdings die übermäßig hohe Komplexität der dort entwickelten Algorithmen im Vergleich zur Berechnung winkelabhängiger Relationen, die Gapp einführt und die im folgenden erläutert werden.

4.2.3.5 Winkelabhängigkeit

Der Anwendbarkeits- und darauf aufbauend der Präzisionsgrad winkelabhängiger Relationen errechnet sich aus der Winkelabweichung zwischen dem

Vektor $\overrightarrow{np_{RO}np_{LO}}$ und den kanonischen Richtungen. Dazu kann auf Vorarbeiten aus dem Halbraummodell zurückgegriffen werden.

4.2.3.6 Winkelkombination

Im Gegensatz zum weiter oben geschilderten Berechnungsverfahren zur Kombination winkelabhängiger Relationen im Halbraummodell (vgl. 4.2.3.3) beruht dieses Verfahren auf bereits gradierten Anwendbarkeiten. Insofern ist der hier ermittelte Wert komplexer, aber dafür exakter.

Gapp (1997) diskutiert vier Möglichkeiten, ausgehend von den Anwendbarkeitsgraden zweier kanonischer Richtungen den einer zusammengesetzten zu berechnen:

- Bei der *Differenzbildung* errechnet sich der neue AG als die Negation (*eins minus*) der absolut genommenen Differenz der kanonischen Werte.
- Die *Rotation* dreht zuerst das Referenzsystem um 45 Grad, danach wird ein kanonischer Anwendbarkeitsgrad bestimmt, der der zusammengesetzten Relation zugeordnet wird. Hier findet also keine direkte Kombination bereits berechneter Werte statt.
- Die Bildung eines einfachen *Minimums* der vorhandenen AG zur Ermittlung des neuen führt in der Regel zu sehr kleinen Anwendbarkeitsbereichen.
- Deshalb schlägt Gapp letztlich ein *skaliertes Minimum* vor, das den oben genannten Fehler vermeidet. Die Skalierung ist frei und somit ist dieses Verfahren auch geeignet, als Basis für die Integration experimenteller Befunde[5].

Wie bereits weiter oben ausgeführt, wird bei allen Verfahren, die gradierte Konzepte behandeln, der Präzisionsgrad als Ergebnis geliefert. So soll es auch hier sein. Da der Präzisionsgrad aber nur *einen* Anwendbarkeitsgrad als Eingabe erwartet (zusammen mit dem entsprechenden x-Wert) kann das von Gapp präferierte Verfahren des skalierten Minimums nicht verwendet werden, da hierbei je *zwei* AG und x-Werte Berücksichtigung finden. Daher kombiniert der in der vorliegende Arbeit verwendete Algorithmus das Rotationsverfahren mit einer Maximumauswahl, bei der sich der gesuchte Präzisionsgrad an dem der dominanten Teil-Relation orientiert (bei `links_oben` also an `links`). Dies berücksichtigt indirekt das von Gapp ermittelte Phänomen, daß die Anwendbarkeitsfunktionen auf den Achsen des Koordinatensystems nicht identisch sind. Auch dieses Verfahren kann aber, obwohl es überzeugende Resultate liefert, aus den oben genannten Gründen nicht als gesichert angesehen werden, da eine experimentelle Validierung, die alle Faktoren einschließt noch aussteht.

[5]Insbesondere haben die in Abschnitt 4.4 beschriebenen Experimente gezeigt, daß eventuell ein noch größerer Anwendbarkeitsraum, als er mit dem Skalierungsfaktor von Gapp erreicht wurde, kognitiv adäquat ist. Dies stünde im Widerspruch zu Gapps Ergebnissen. Insgesamt ist hier noch Untersuchungbedarf, so daß eine variable Berechnung sicher nützlich sein kann.

4.2.3.7 Geographische Relationen

Die in den vorangegangenen Abschnitten behandelten winkelabhängigen Lagebeziehungen wurden bisher Relationen wie `links` oder `hinter` zugeordnet. Insbesondere bei (Land-)Kartenmaterial, das z.B. bei Wegbeschreibungen Verwendung findet, sind stattdessen geographische Relationen wie `westlich` oder `nördlich` relevant. Diese Unterscheidung erfolgt durch die Berücksichtigung des Kontextes. Auf kanonischer Ebene und bei den eben beschriebenen zusammengesetzten winkelabhängigen Relationen sind die Berechnungsverfahren identisch.

Zusätzlich ist eine weitere Klasse zusammengesetzter geographischer Relationen gebräuchlich, die den Suchraum noch einmal unterteilt, wie etwa `südsüdwestlich`. Ihre Berechnung erfolgt analog zu den Verfahren für *normale* kombinierte winkelabhängige Relationen.

4.2.3.8 Distanz- und Winkelkombination

Das Berechnungsverfahren auf dieser Ebene greift auf schon bestimmte winkelabhängige und distanzabhängige Relationen zurück und kombiniert sie – falls möglich – zu komplexeren Konstrukten wie z.B. `links_bei`. Der Anwendbarkeitsgrad der zusammengesetzten Relation errechnet sich bei Gapp durch Minimumbildung. Dies ist nach Einführung des Präzisionsgrades nicht mehr möglich, da dazu nur *eine* gradierte Relation verrechnet werden kann. Aus diesem Grund wurde direkt auf die PG der bereits berechneten Relationen zurückgegriffen.

4.2.3.9 Sonderfälle

Diese Gruppe von Relationen umfaßt solche, die sich nicht ohne weiteres klassifizieren lassen. Exemplarisch wurde die Relation `auf` zusammengesetzt aus `über` und `kontakt` integriert. Die Berechnung erfolgt auf analoge Weise wie die Kombination von distanzabhängigen und winkelabhängigen Relationen.

Während `neben` als simple Vergröberung von `links` *oder* `rechts` angesehen werden kann, ist die Problematik bei `zwischen` komplexer: Bei dieser Relation muß zumindest ein *zweites* Referenzobjekt einbezogen werden. Auch kann sich die Lagebeziehung auf eine Gruppe von Referenten beziehen.

4.2.3.10 Kombinationen mit Sonderfällen

Diese Klasse beinhaltet Relationen wie `links_auf`. Dazu müssen zuvor die zu kombinierenden Teile schon berechnet sein. Problematik und Algorithmus gleichen den zuletzt beschriebenen.

4.2.3.11 Endpunktorientierung

Dieses erste Verfahren zur Berechnung von n-Punkt-Relationen betrachtet nur die beiden Endpunkte der Trajektorie und interpretiert diese dadurch

quasi als 2-Punkt-Trajektorie. Somit können wie in Abschnitt 1.4.4.4 beschrieben Anwendbarkeits- und Präzisionsgrad der Relationen `hin_zu`, `weg_von`, `längs` sowie `rechtsrum_um` und `linksrum_um` in einer groben Näherung bestimmt werden.

4.2.3.12 Stützpunktorientierung

Auf dieser zweiten Stufe der Berechnung von pfadbezogenen Relationen werden *alle* Stützpunkte einbezogen (vgl. Abschnitt 1.4.4.5). Dabei baut das Verfahren direkt auf den bei der Endpunktorientierung erzielten Zwischenresultaten auf und kann nach der Einbeziehung eines jeden Punktes unterbrochen werden. Dies ist inbesondere bei Trajektorien mit vielen Stützpunkten von Vorteil. So kann z.B. auch lediglich ein Teilverlauf berechnet werden.

Zusätzlich zu den oben aufgezählten Pfadrelationen können `vorbei`, das abstrakte `trip` und `durch` berechnet werden. Letzteres liefert allerdings noch keine Information darüber, welcher Teil – Start- oder Endpunkt oder ein Mittelstück – der Trajektorie das RO durchdringt.

4.2.3.13 Kombination von n-Punkt-Relationen

Das klassische Beispiel für eine Relation dieser Gruppe ist die Realisierung der Präposition *„um“* durch `rechtsherum_um` (`linksherum_um`) als Kombination von `rechtsrum_um` (`linksrum_um`) und `längs`[6]. Die Relation `entlang` kann aus dem abstrakten `nicht_drehend` und `längs` zusammengesetzt werden, im Sinne eines *um*, das nur eine Seite des Referenzobjektes (oder wenig mehr) überstreicht. Das eine Stufe vorher generierte `durch` kann nun mit `hin_zu` bzw. `weg_von` zu `hinein_in` respective `heraus_aus` erweitert werden. Wie bei den schon beschriebenen Kombinationen von 2-Punkt-Relationen errechnet sich der Präzisionswert auch hier direkt aus den Teilwerten.

4.2.3.14 Kombination mit 2-Punkt-Relationen

Diese letzte Klasse der Berechnungsverfahren für räumliche Relationen berechnet den Präzisionsgrad von Kombinationen zwischen n- und 2-Punkt-Relationen. Dabei lassen sich zwei Gruppen unterscheiden: Zum einen kann der Start- oder Endpunkt der Trajektorie bezüglich 2-Punkt-Relationen analysiert und mit einer Pfadrelation kombiniert werden. Daraus entstehen Konstrukte, die auf den ersten Blick etwas seltsam anmuten, wie z.B. `hin_zu_hinter`, aber durchaus sinnvoll auf natürlichsprachliche Entitäten – hier vielleicht *„Er geht hinter das Haus“* – abgebildet werden können. Die andere Untergruppe verfeinert eine n-Punkt-Relation durch eine 2-Punkt-Relation, wobei sich beide auf die gesamte Trajektorie beziehen. Das Resultat sind zusammengesetzte Relationen wie `links_entlang` oder `nahe_vorbei`.

[6] Damit wird natürlich nur *einer* von mehreren möglichen Ausprägungen des sprachlichen Konzeptes *„um“* entsprochen.

4.2.3.15 Applikation linguistischer Hecken

Die Verarbeitung linguistischer Hecken erfolgt wegen ihrer Komplexität in einem eigenen anytime-fähigen Modul, das als Eingabe eine räumliche Proposition aus abstrakter Relation (aR), LO und RO erwartet, auf die eine oder mehrere potentielle abstrakte Hecken (aH) angewendet werden sollen (vgl. Abb. 4.10). Die Frage, ob und gegebenenfalls wie, die Menge dieser potentiellen Hecken – z.B. unter Einbeziehung des Kontextes – vorab eingeschränkt werden kann, ist noch unbeantwortet. Allerdings spielt dies bei der relativ kleinen Anzahl unterschiedlicher Heckenausdrücke und ihrer Kombination (bis maximal drei) eine untergeordnete Rolle, zumal die Verfahren in einem selbstorganisierenden System verarbeitet werden, das die vorhandenen Ressourcen beschränkt-optimal alloziert.

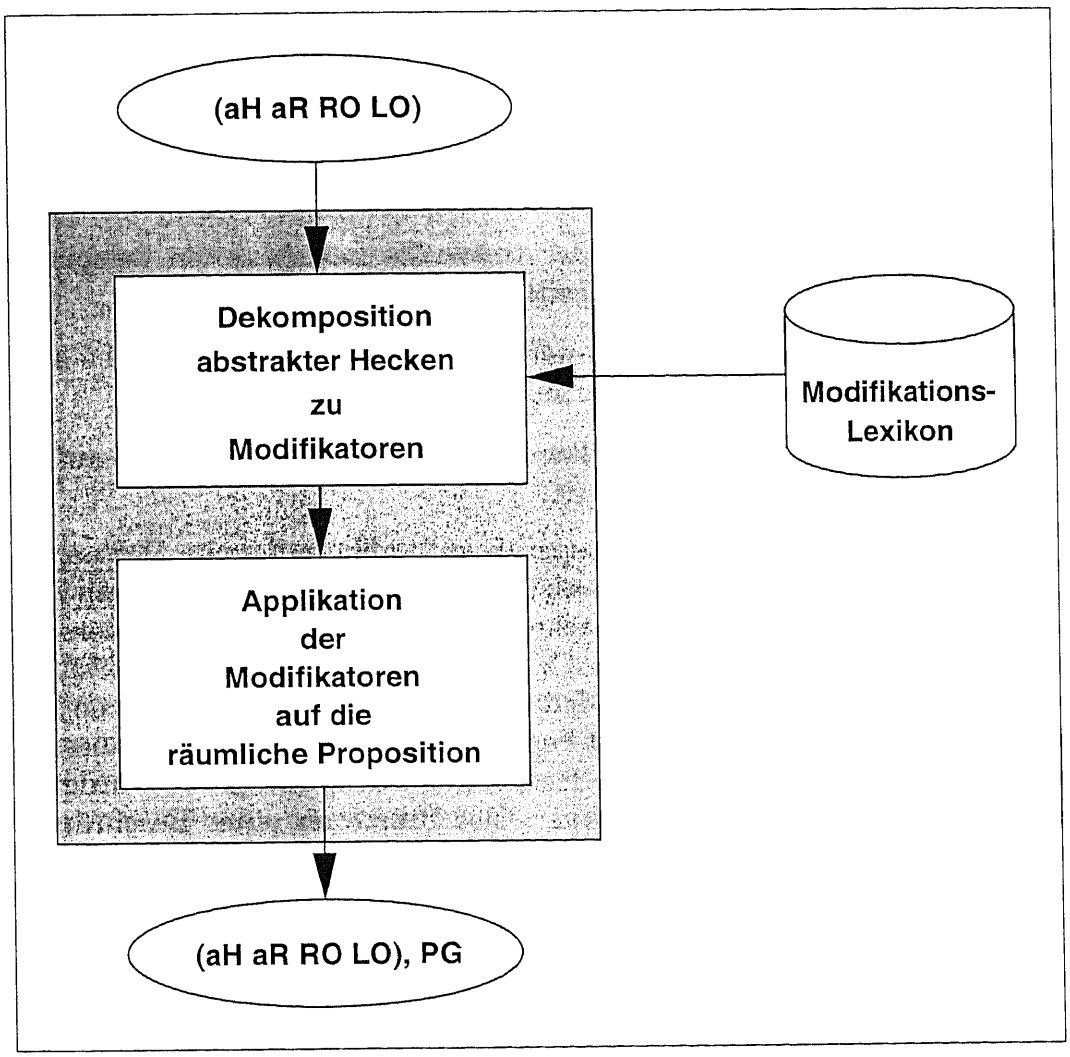

Abbildung 4.10: Algorithmus zur Berechnung linguistischer Hecken

Die Heckenkandidaten werden in einem Dekompositionsschritt mittels eines Lexikons in ihre konstituierenden Prä- oder Post-Modifikationsoperatoren aufgespalten. Tabelle 4.1 stellt dies dar und ordnet auch exemplarisch

einige Lexeme abstrakten Hecken zu. Dadurch wird deutlich, daß *keine* 1:1-Beziehung zwischen beiden Gruppen vorliegt. Die vorletzte Spalte gibt an, mit welchen Hecken eine weitergehende Kombination *nicht* möglich ist. (Wie bei allen Verfahren können auch hier Zwischenergebnisse weiterverwendet werden.)

Folgende elementare Modifikationsoperatoren wurden in Anlehnung an Zadeh (1972) definiert:

- *Konzentration* (`con`) verkleinert den Bereich höherer Anwendbarkeit und macht dadurch eine Relation schärfer.
- *Dilation* (`dil`) vergrößert den Bereich höherer Anwendbarkeit und macht dadurch eine Relation unschärfer.
- *Kontrastintensifikation* (`conint`) führt zu einer größeren Akzentuierung der Anwendbarkeit: Damit wirken sich Änderungen stärker aus.
- *Negation* (`nega`) kehrt die Anwendbarkeit einer Relation um.
- *Linksverschiebung* (`ltrans`) der Anwendbarkeitsfunktion, beispielsweise zur Präzisierung bei linksseitigen Maxima.
- *Rechtsverschiebung* (`rtrans`) der Anwendbarkeitsfunktion, beispielsweise zur Präzisierung bei rechtsseitigen Maxima.
- *Flip* (`flip`) spiegelt alle Anwendbarkeitsgrade oberhalb von 0,5 an der 0,5-Gerade, etwa zur Modellierung von `kaum`.
- *Cuttop* (`cuttop`) setzt Anwendbarkeitsgrade oberhalb von 0,95 auf Null, etwa zur Modellierung von `fast`.

Dieses Modell erhebt keinen Anspruch auf Vollständigkeit. Doch erlaubt sein modularer Aufbau und die Kombination atomarer Modifikatoren zu beliebig komplexen einerseits entsprechende Erweiterungen und andererseits Anpassungen an neue Erkenntnisse, z.B. aus experimentellen Untersuchungen, die die bisher introspektiven Modellierungen eventuell validieren könnten. Der Schwerpunkt dieser Arbeit liegt auf dem Bereich der Raumbeschreibung, weshalb insbesondere solche Heckenausdrücke integriert wurden, die in diesem Kontext gebräuchlich sind. Dies führte zu einer Beschränkung auf ungefähr zwanzig verschiedene Modifikatorenmengen und circa zehn Lexeme, die in wesentlichen den Grad- oder Steigerungspartikeln zuzuordnen sind. Auch hierzu liegen noch keine Untersuchungen vor. Die Arbeit von Kipper (1995) behandelt Modalworte und Abtönungspartikeln, die für hier weniger relevant sind. Das Phänomen der Negation wurde wegen der großen Komplexität nur durch eine einfache Invertierung interpretiert.

Trotz dieser Einschränkungen verfügt der vorgeschlagene Ansatz über einige Vorteile, die ihn über die in Abschnitt 1.5.1.3 beschriebenen herausheben: Zuerst ist hier die Modularität sowie die konsequente Trennung in abstrakte und konkrete Hecken zu nennen. Beides ermöglicht eine einfache Erweiterung des Moduls und ermöglicht letztlich die Vermeidung einer starren 1:1-Beziehung von Modifikationsfunktionen zu Lexemen.

Abstrakte Hecke	Relation	Modifikator(en)	nicht komb.	Lexem
Intensifikatoren				
int1	an, bei, nahe, entfernt	con	des1, vag1	sehr
int2	nahe	rtrans	int3	sehr
int3	entfernt	ltrans	int2	sehr
int4	an, bei, nahe, entfernt	con con	des1, vag1	äußerst
int41	links, rechts, vor, hinter, über, unter	con con	des1, vag1	ganz
int5	links, rechts, vor, hinter, über, unter, nahe	con rtrans	int6, int61	ganz
int51	an, bei, nahe, entfernt	con rtrans	int6, int61	äußerst
int6	links, rechts, vor, hinter, über, unter, entfernt	con ltrans	int5, int51	ganz
int61	an, bei, nahe, entfernt	con ltrans	int5, int51	äußerst
Desintensifikatoren				
des1	(alle)	dil	int1, des1	ziemlich
des2	(alle)	flip	des2	kaum
des3	entfernt	cuttop dil nega	(alle)	etwas
Negation				
neg1	(alle)	nega	des2	nicht
Gradpartikeln: ungefähr-Gruppe				
vag1	(alle)	flip conint dil	int1, int4	ungefähr
vag2	(alle)	flip conint conint dil	int1, int4	fast
Gradpartikeln: gerade-Gruppe				
exa1	entfernt	conint conint con con	(alle)	genau
exa2	entfernt	conint con con con	(alle)	genau

Tabelle 4.1: Anwendung von Hecken-Modifikatoren auf abstrakte Relationen

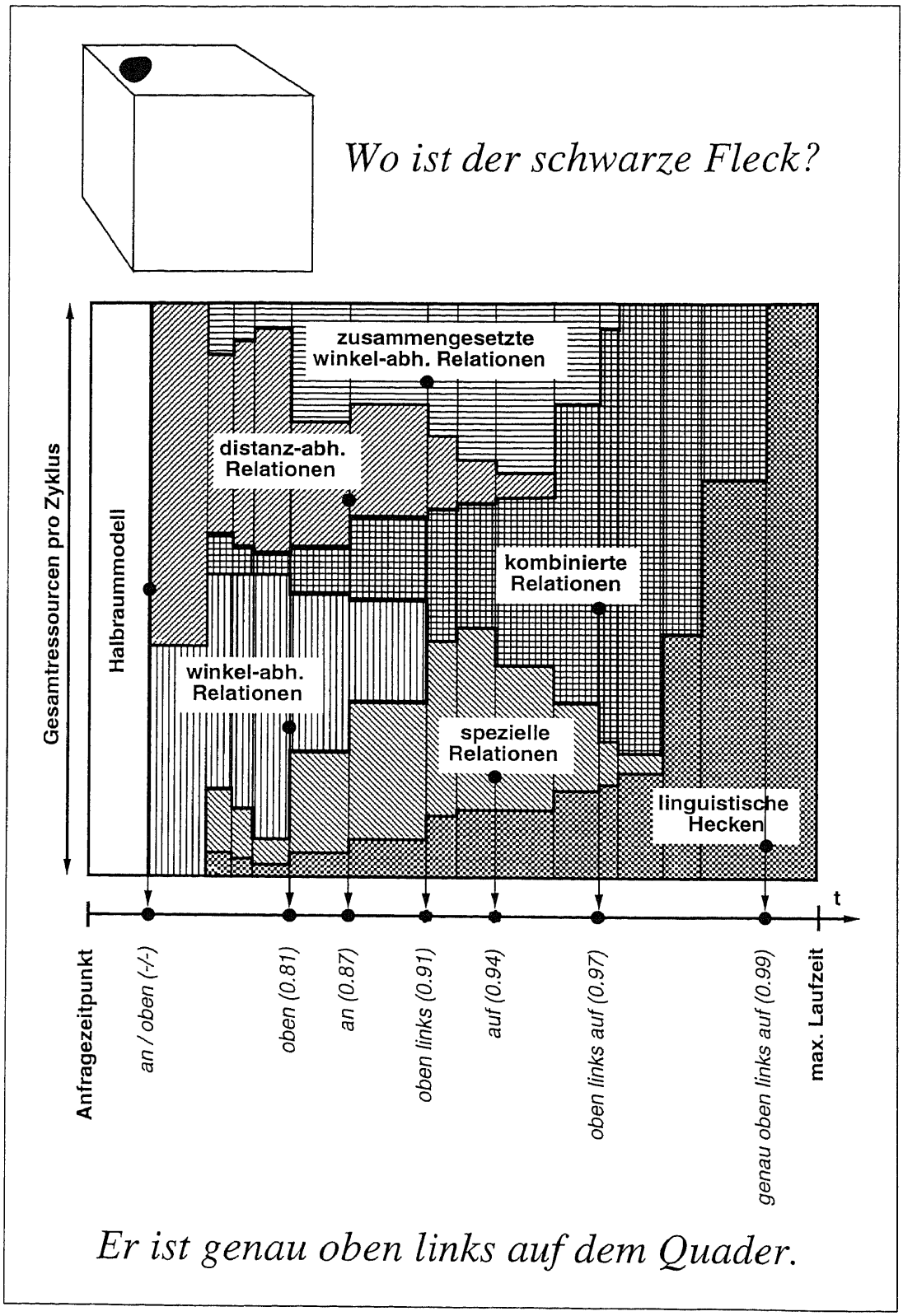

Abbildung 4.11: Die Ressourcenverteilung der Anytime-Prozesse

Zweitens erlaubt der Präzisionsgrad die Vergleichbarkeit verheckter und nicht-verheckter Relationen, so daß hier wie bei der Generierung der Heckenausdrücken volle Ressourcenadaptivität gewährleistet ist: Es müssen nicht immer und nicht immer alle Hecken berechnet werden, um aussagekräftige (Zwischen-)Ergebnisse zu erhalten. Auch können gegebenenfalls nicht nur verschiedene Hecken kombiniert sondern auch dieselbe Hecke mehrfach angewendet werden, was jeweils zu einem Abschwächungseffekt führt: Je mehr Hecken appliziert werden, umso geringer wird ihr Einfluß. Dies entspricht dem Sprachgebrauch, in dem nach Pinkal (1985) eine übermäßige Kombination vermieden wird. Im vorliegenden Ansatz werden maximal drei Heckenausdrücke miteinander verknüpft.

4.2.4 Beschreibung eines prototypischen Systemlaufs

In diesem Abschnitt werden die Erläuterungen, die zuvor zu den einzelnen Berechnungsverfahren gegeben wurden, in einem Pseudo-Systemdurchlauf, wie er auch realiter vorkommen kann, zusammengefaßt, um Zwischenergebnisse, Abhängigkeiten und Interaktionen zu illustrieren. Eine schematische Darstellung des Anytime-Moduls von BOLA zur Auswahl einer besten Relation ist in Abb. 4.11 wiedergegeben. Dabei sind Zyklen als vertikale Balken variabler Breite (d.h. Zeitdauer) visualisiert und entsprechend der prozentualen Ressourcenverteilung auf die beteiligten Prozesse gegliedert.

In einem ersten Schritt wird ein Referenzsystem aufgebaut, das seinen Ursprung im RO hat und durch seine drei kanonischen Achsen ein Halbraummodell induziert. Dieses erlaubt zwar lediglich eine binäre Unterteilung des Raumes, kann aber sehr rasch ausgewertet werden.

Als Ergebnis wird eine Präferenzliste der Halbräume sowie eine qualitative Distanzinformation geliefert. Sie unterscheidet zwischen den topologischen Relationen `inside`, `meet` (in Abb. 4.11 auf `an` abgebildet) und `disjoint`.[7]

Es könnte nun als eine erste grobe Lagebeschreibung eine Distanz- und die zum besten Halbraum gehörende winkelabhängige Relation generiert werden. Allerdings können noch keine Aussagen über die Qualität von Lokationsbeschreibungen gemacht werden, denen diese Relationen zugrunde liegen, so daß eine Vergleichbarkeit mit anderen Referenzobjekten nicht gegeben ist.

Aufbau und Auswertung des Halbraummodells entsprechen *einer* Transaktion, ohne deren vollständige Durchführung keine Antwort generiert werden kann.

Falls noch Zeit zur Verfügung steht, wird eine Optimierung des Resultats angestrebt. Im Falle von Zeitrestriktion (vgl. Experiment in Abschnitt 4.4) sollte beim Erreichen einer bestimmten Qualitätsstufe die Berechnung abgebrochen und die Raumbeschreibung verbalisiert werden, auch wenn spätere Verbesserungen nicht auszuschließen sind.

Auf den Ergebnissen des Halbraummodells bauen zunächst zwei weitere Prozesse auf: Die bisher gefundenen Relationen werden hinsichtlich der Qua-

[7] Diese vereinfachte Darstellung gilt nur bei der Repräsentation des LO als Punkt.

lität bezogen auf ihre Anwendbarkeit und Präzision untersucht. Dadurch wird einerseits die Vergleichbarkeit untereinander sowie mit anderen ROs ermöglicht, andererseits kann sich herausstellen, daß die Qualität einer Relation den Ansprüchen einer Lokationsbeschreibung nicht genügt. In diesem Fall sollte eine Verfeinerung der Beschreibung angestrebt werden. Dies geschieht in weiteren, teilweise parallelen Prozessen, die basierend auf den erzielten Zwischenergebnissen komplexere räumliche Strukturen wie zusammengesetzte sowie spezielle räumliche Relationen und letztlich die mögliche Anwendung linguistischer Hecken analysieren.

Bei einer vollständigen Abarbeitung aller Berechnungsebenen ist das Resultat *der* räumliche Ausdruck, der die gegebene Objektkonstellation am präzisesten beschreibt. Dies muß nicht immer eine komplexe räumliche Relation sein: In manchen Fällen wird auch nach größtem Aufwand noch die allererste Lösung valide sein – allerdings ist die Verläßlichkeit ihrer Aussagekraft gesichert.

4.3 Beispiellauf des Lokalisationsagenten

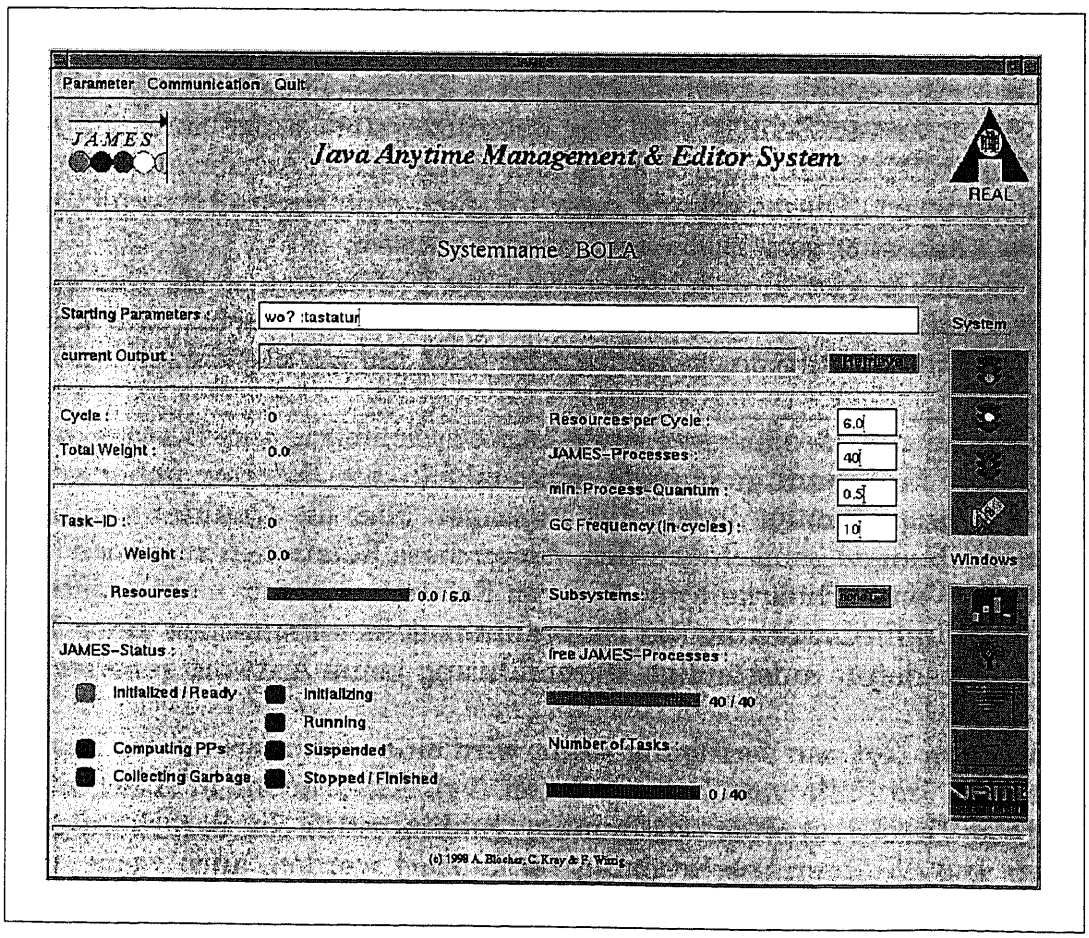

Abbildung 4.12: Beispiellauf - Anfragestellung

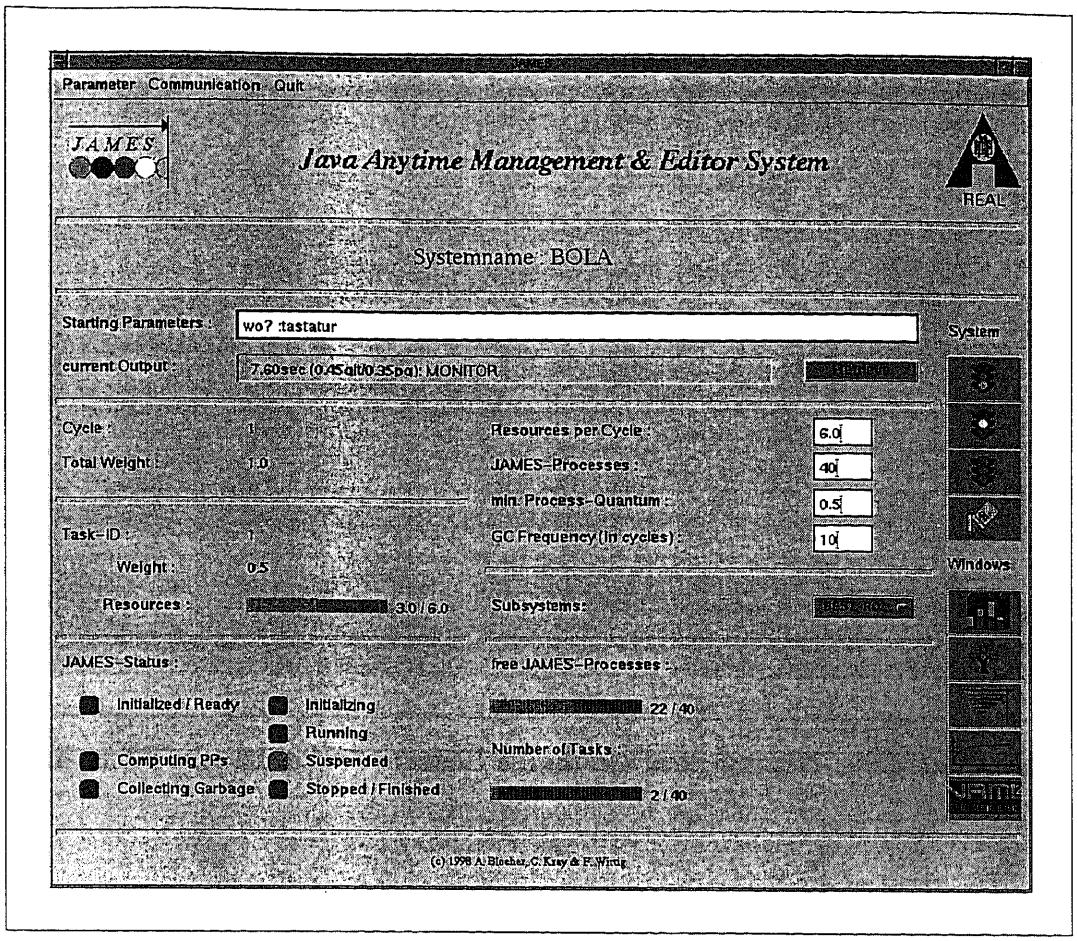

Abbildung 4.13: Beispiellauf - Unterbrechung

Nachdem nun alle Komponenten des beschränkt-optimalen Lokalisationsagenten beschrieben wurden, kann anhand eines Beispiellaufs des Gesamtsystems die Vorgehensweise demonstriert werden. Dazu wird in der Bürodomäne aus Abb. 4.3 die Frage *„Wo ist die Tastatur?“* bearbeitet.

Abbildung 4.12 zeigt das graphische Interface der Anytime-System-Shell, in die BOLA als Gastsystem integriert wurde, vor dem Start (vgl. (Wittig, 1998)).

Nach Eingabe der Anfrage wird das System durch Drücken des Start-Buttons (grüne Ampel) mit den gewählten Parametern (die Zykluslänge beträgt 6.0 Zeiteinheiten (ZE), die Anzahl der verfügbaren Prozesse 40) aktiviert. Da für dieses Beispiel eine Vielzahl von Trace-Funktionen die Laufzeit erheblich verlängern, sind Zeitangaben irrelevant. Für Performanzbestimmungen müssen diese Funktionalitäten dementsprechend deaktiviert werden.

Eine erste durch den Benutzer initiierte Unterbrechung (gelbe Ampel) liefert das bis zu diesem Moment berechnete Zwischenergebnis aus Abb. 4.13. Bislang konnte nur ein Referenzobjekt (*„Monitor“* mit der Qualität $0,45$), aber noch keine Relation geliefert werden.

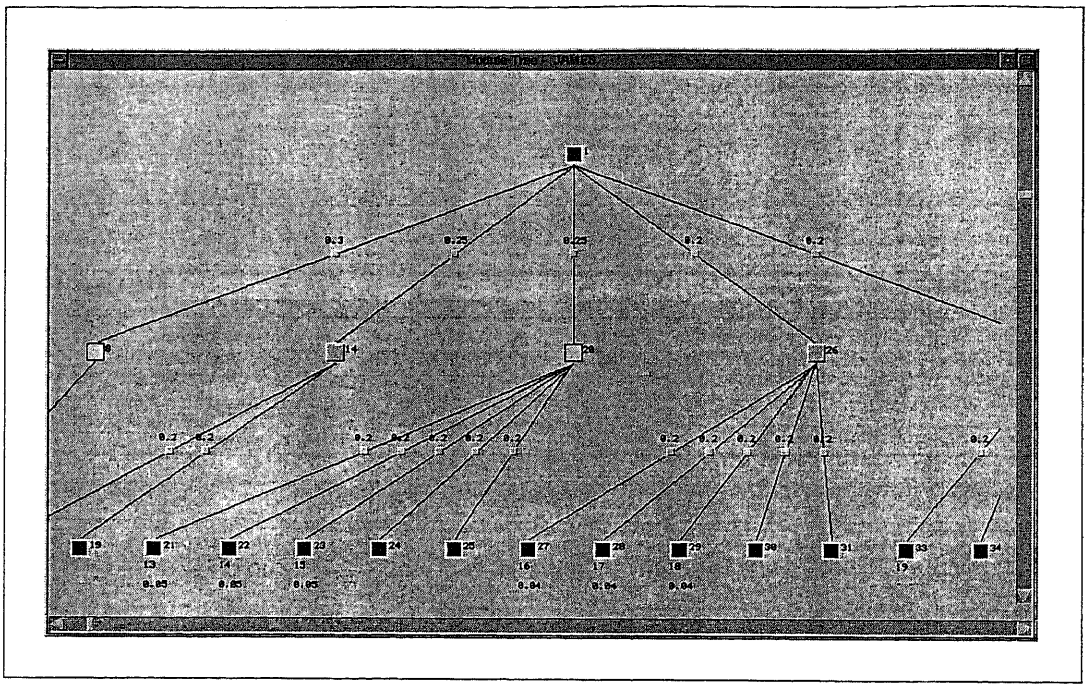

Abbildung 4.14: Beispiellauf - Lokaler Ressourcenbaum von BEST-RO

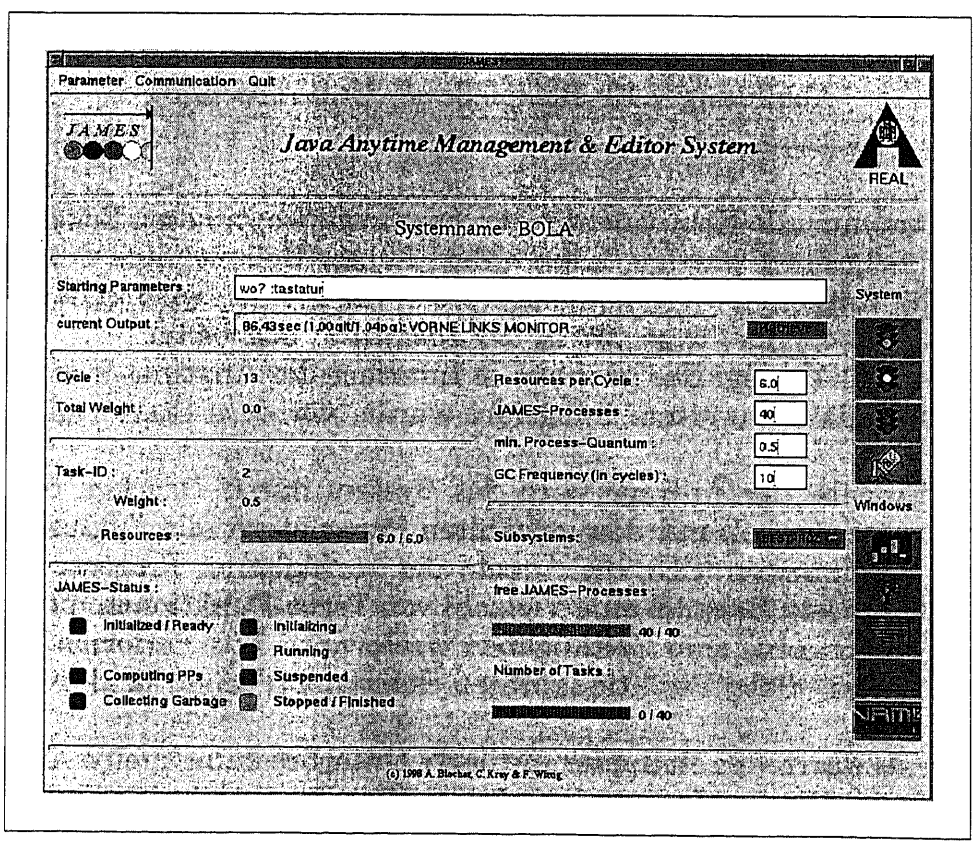

Abbildung 4.15: Beispiellauf - Endergebnis

Abbildung 4.14 zeigt einen Ausschnitt aus dem lokalen Ressourcenbaums der RAMI des Subsystems BEST-RO. Deutlich läßt sich an der unterschiedlichen Anzahl der Nachfolger z.B. des zweiten inneren Knoten erkennen, daß in dieser ursprünglich homogenen hierarchischen Struktur bereits einige Tasks vollständig abgearbeitet wurden – vermutlich hauptsächlich solche, die Bewertungen für das potentielle Referenzobjekt *„Monitor"* beitragen.

Der unterbrochene Systemlauf wird fortgesetzt und als Endresultat ergibt sich *„Die Tastatur befindet sich links vor dem Monitor"* mit einer Qualität von 1,0 (vgl. Abb. 4.15).

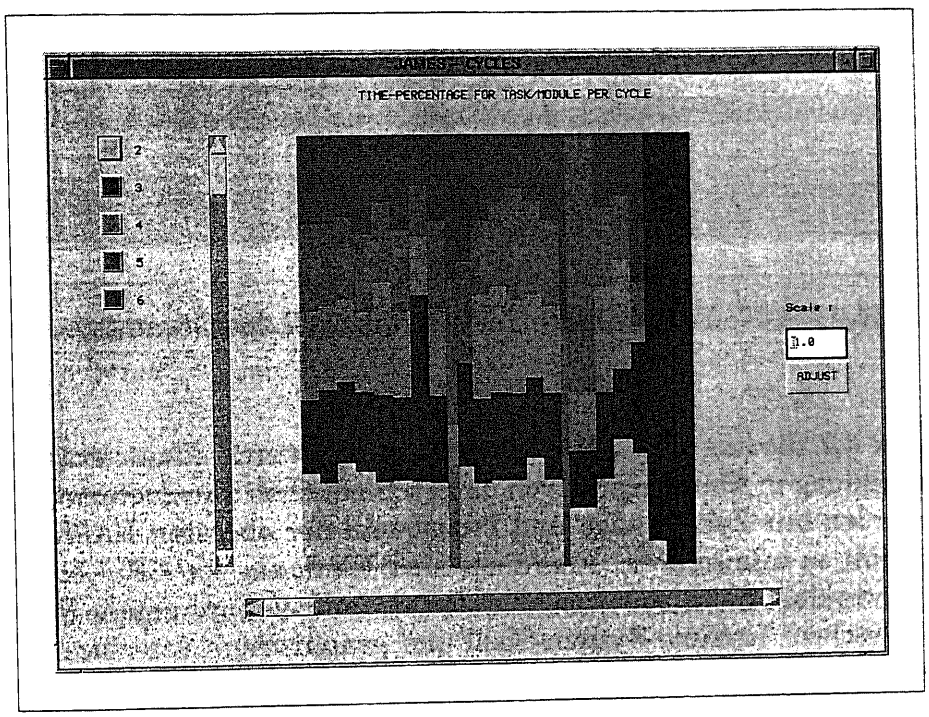

Abbildung 4.16: Beispiellauf - Ressourcenverteilung von MULTI-RELATIONS

Als beispielhafte Darstellung einer in diesem realen Ablauf erzeugten Ressourcenverteilung ist in Abb. 4.16 das entsprechende Balkendiagramm für die RAMI des Subsystems MULTI-RELATIONS, die fünf RELATIONS-Subsysteme – für jedes potentielle Referenzobjekt – verwaltet, zu sehen. Die Unregelmäßigkeiten in der Verteilung (für die hier zu Beginn gleich gewichteten ROs) entwickeln sich anfangs z.B. aus unterschiedlich komplexen Objekten und summieren sich dann von Schritt zu Schritt bei der Berechnung der jeweils besten Relation. Es treten auch Zyklen auf, in denen nicht alle Tasks (oder sogar nur ein Task) bearbeitet wird, sei es, weil eine *lange* Transaktion alle verfügbaren Ressourcen beansprucht oder weil das eine oder andere RELATIONS-Subsystem schon beendet werden konnte.

Abbildung 4.17 zeigt das Performanzprofil des Gesamtsystems, wobei jede Stufe den Qualitätszuwachs durch Beendigung einer Teilaufgabe darstellt.

Verzichtet man auf die erwähnte Trace-Möglichkeit, so lassen sich die Auswirkungen von Parameteränderungen auf des Systemverhalten analysieren.

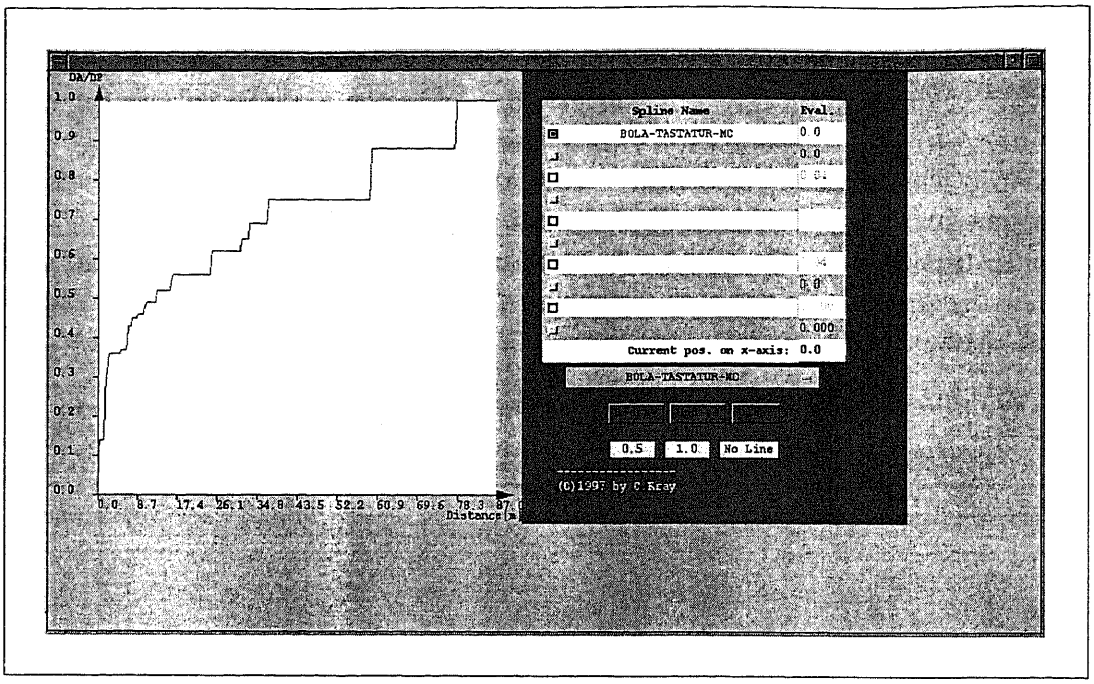

Abbildung 4.17: Beispiellauf - Performanzprofil des Gesamtsystems

Tabelle 4.2, die aus Wittig (1998) übernommen wurde, illustriert die Laufzeitentwicklung in Abhängigkeit von unterschiedlichen Parameterwerten. Variiert wurden nur Zykluslänge und Prozeßzahl der obersten RAMI, um die Komplexität zu minimieren. Allerdings sind Einflüsse wie CPU-Auslastung und Netzwerklast nicht zu vermeiden, ferner muß eine minimale Kommunikation zwischen Anytime-System-Shell und graphischem Interface erhalten bleiben. Die absoluten Laufzeiten variieren auf dem verwendeten Rechner zwischen 13 und 26 Sekunden.

Zur Interpretation der Tabelle ist anzumerken, daß sich eine Erhöhung der Zykluszeit insbesondere bei einer geringen Prozeßzahl positiv auswirkt, sonst aber kein Effekt auftritt, da die lauffähigen Tasks schon ihre Berechnungen durchführen können, wenn auch im wesentlichen sequentiell. Betrachtet man auf der anderen Seite die Veränderung der Zahl der verfügbaren Prozesse, so zeigt sich, daß hier ein Mehr zwar zu Beginn eine starke, danach aber kaum noch Beschleunigung bringt. Der Grund liegt in den verwendeten Transaktionen, die die Anzahl der parallelen Berechnungen beschränken.

Ein letzter Blick soll auf eine Modifikation der relativen Gewichte in den Ressourcenbäumen gerichtet werden, wie sie beispielsweise durch eine Ressourcenadaptierung anhand der Performanzdaten bedingt sein kann. In diesem Fall erfolgt – wenn überhaupt – keine große Änderung der Laufzeit, sondern der Performanzprofile, da auf diesem Weg eine gezielte Veränderung der Abarbeitungsreihenfolge vorgenommen wird. Dies kann z.B. genutzt werden, um zu Anfang der Berechnung schnelle *oder* gute Resultate zu generieren.

Für eine abschließende Bewertung von Parametereinflüssen ist es noch zu früh. Hierfür wären intensive Testreihen vonnöten, die über den Rahmen dieser Arbeit, die Verfahren zur Integration ressourcenadaptierenden Verhal-

Prozeßanzahl	Zykluslänge (in ZE)	Laufzeit (in Sekunden)
5	3	ca. 26
5	6	ca. 21
5	9	ca. 21
7	3	ca. 16
7	6	ca. 16
7	9	ca. 16
12	3	ca. 14
12	6	ca. 13
12	9	ca. 13
20	3	ca. 14
20	6	ca. 13
20	9	ca. 13

Tabelle 4.2: Laufzeiten des Gesamtsystems bei variierten Parametern

tens entwickeln sollte, hinausgehen würde. Im nächsten Abschnitt wird ein erster Ansatz zur Validierung von Teilaspekten – auf dem Gebiet der Berechnung räumlicher Relationen – vorgenommen.

4.4 Validierung des Ansatzes anhand psychologischer Experimente

Im Rahmen der engen Zusammenarbeit des SFB 378 zwischen dem Projekt REAL, in dem diese Arbeit angesiedelt ist, und dem Projekt VEVIAG (Verbales und visuelles Arbeitsgedächtnis unter der Leitung von Prof. Dr. H. Engelkamp und Dr. H. Zimmer) wurde gemeinsam eine computergestützte Experimentalserie entwickelt und durchgeführt (vgl. (Wahlster, Blocher, Baus, Stopp & Speiser, 1998)). Sie sollte die Auswirkung einer Beschränkung der Ressource Zeit auf das Erkennen einer räumlichen Situation und ihrer natürlichsprachlichen Beschreibung in einer Dialogsituation untersuchen. Anders gesagt: Wie schnell können Versuchspersonen unter Zeitdruck eine Lagebeziehung kognitiv verarbeiten, zum einen, um die Situation zu beschreiben, zum anderen, um eine solche Beschreibung zu verstehen? In den ersten Versuchsreihen wurde der Gebrauch räumlicher Lokalisationsausdrücke zur Verbalisierung einfacher winkelabhängiger Relationen analysiert (Zimmer, Speiser, Baus, Blocher & Stopp, 1998).

Das Experimentalsetting gestaltete sich wie folgt: Zwei durch eine Sichtwand getrennte Versuchspersonen befinden sich in einer Kommunikationssituation. Eine von ihnen, der Produzent, muß eine auf seinem Monitor dargebotene räumliche Konfiguration, bestehend aus einem Referenzobjekt und einem später eingeblendeten zu lokalisierenden Objekt (ein roter und ein blauer Punkt) möglichst schnell seinem Kommunikationspartner beschreiben. Diese Beschreibung darf allerdings nur durch präpositionale Verwendung von *„links“*, *„rechts“*, *„oben“*, *„unten“* oder deren (sinnvolle) Kombination

„links oben" erfolgen. Die zweite Person, der Rezipient, versucht, ausgehend von dem auf seinem Monitor sichtbaren RO, anhand der gehörten Information das für ihn nicht sichtbare LO mit einem Suchfenster, das per Maus bewegt werden kann, zu finden. Im Erfolgsfall erscheint das zu lokalisierende Objekt auf dem Bildschirm und ein neuer Trial kann beginnen.

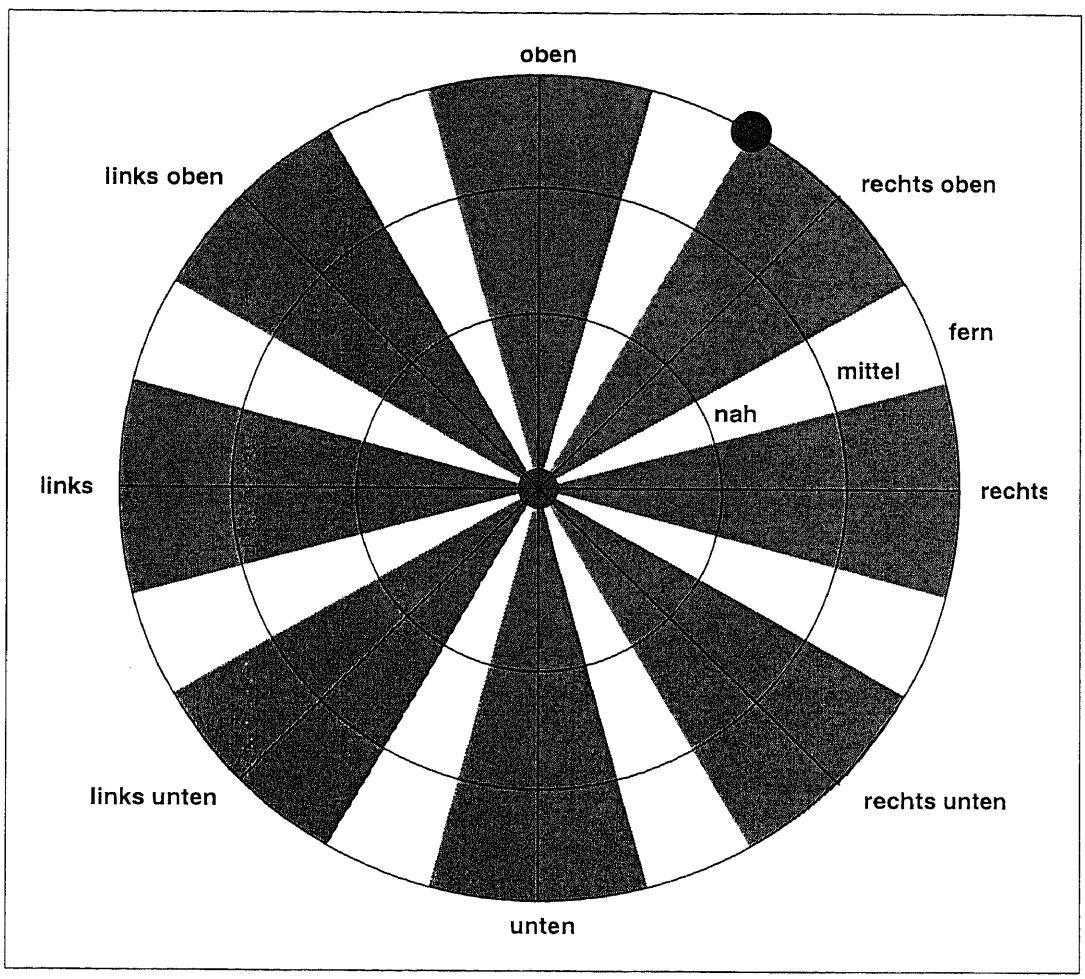

Abbildung 4.18: Experimentallayout

Das Referenzobjekt wird immer im Zentrum des lokalen Referenzsystems positioniert, während bei dem LO Winkel und Distanz variieren. Die Veränderung des Winkels erfolgt in Schritten zu 15 Grad, die Variation der Distanz in drei konzentrischen Kreisen (nah, mittel, fern) um das RO, so daß insgesamt 72 unterschiedliche Positionen von dem zu lokalisierenden Objekt eingenommen werden können (vgl. Abb. 4.18). Diese Konfigurationen werden pseudozufällig generiert und zusätzlich über den gesamten Bildschirm gestreut, so daß ein Verteilungsmuster für die Versuchspersonen nicht zu erkennen ist. Eine detailliertere Beschreibung von Aufbau und Durchführung der Experimentalreihe sowie der zu messenden Latenzzeiten findet sich in (Zimmer et al., 1998).

Die Äußerungen des Produzenten bezüglich der räumlichen Konstellation wurden auf Tonband aufgenommen und später transkribiert. Die Produkti-

onslatenz – der Zeitraum zwischen dem Erscheinen des LO und dem Beginn der Verbalisierung der Konfiguration – wurde mittels eines Voicekey gemessen.

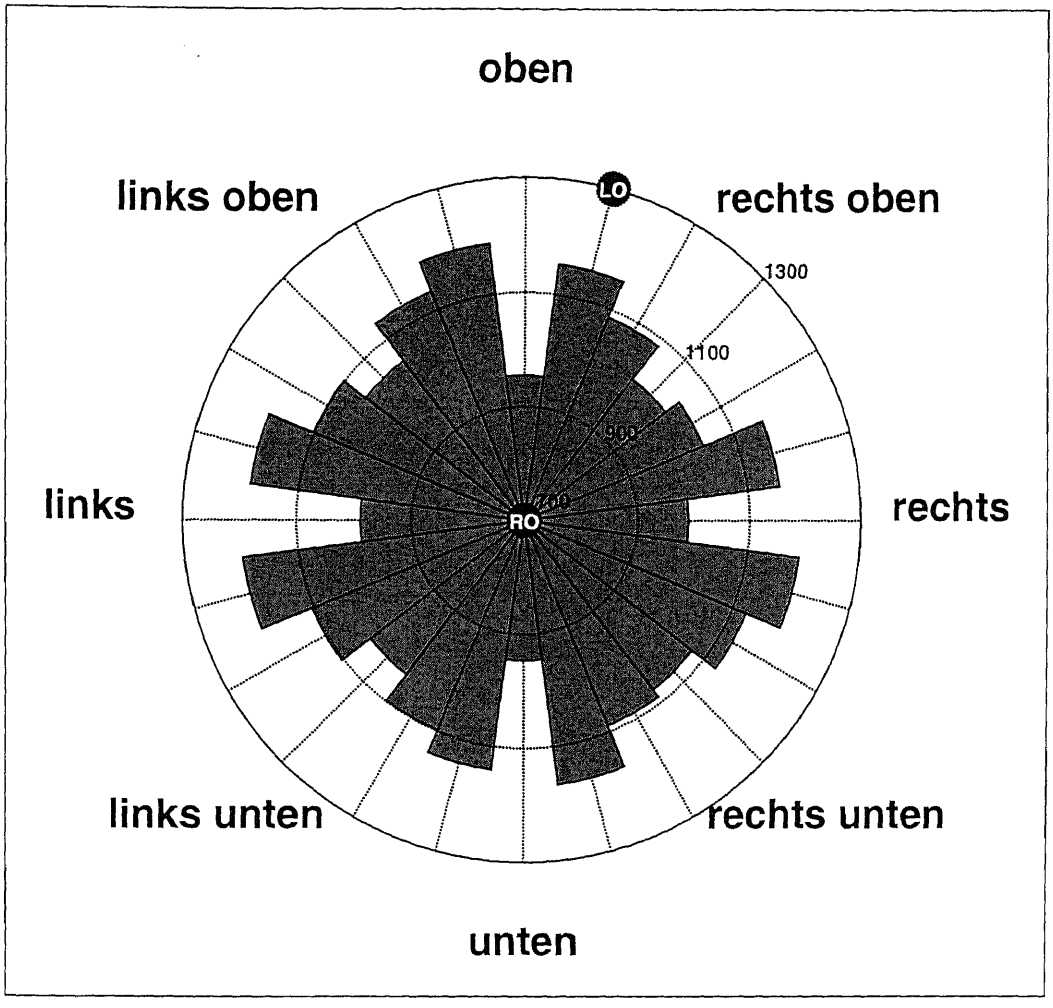

Abbildung 4.19: Die Produktionslatenzen (in ms) der dargebotenen räumlichen Konfigurationen

Abbildung 4.19 zeigt das Ergebnis der Analyse der so gewonnenen Daten hinsichtlich der Beschreibung der räumlichen Lagebeziehungen: Die entsprechenden räumlichen Relationen sind in dem lokalen Referenzsystem mit den jeweiligen Latenzzeiten dargestellt. Augenfällig ist die Symmetrie in den vier Quadranten. Die Latenzen der Beschreibungen von LO-Positionen auf den kanonischen Achsen *links*, *rechts*, *oben*, *unten* sind kürzer als auf den Diagonalen (*links oben*) $[F(2,58) = 7,5; p < 0,01; MSE = 9748]$. Ein Konfliktbereich befindet sich bei einer Abweichung von +/- 15 Grad von den kanonischen Achsen bzw. von +/- 30 Grad von den Diagonalen, da dort sowohl elementare als auch zusammengesetzte winkelabhängige Präpositionen auftreten (z.B. *„rechts“* und *„rechts über“*). Der anstehende Entscheidungsprozeß scheint der Grund für die bemerkenswerte Erhöhung der Latenzzeit zu sein $[F(1,29) = 104,9; p < 0,001; MSE = 8276]$. Eine abschließende Wertung

ist hier allerdings noch nicht möglich, vielmehr müssen weitere Experimente durchgeführt werden.

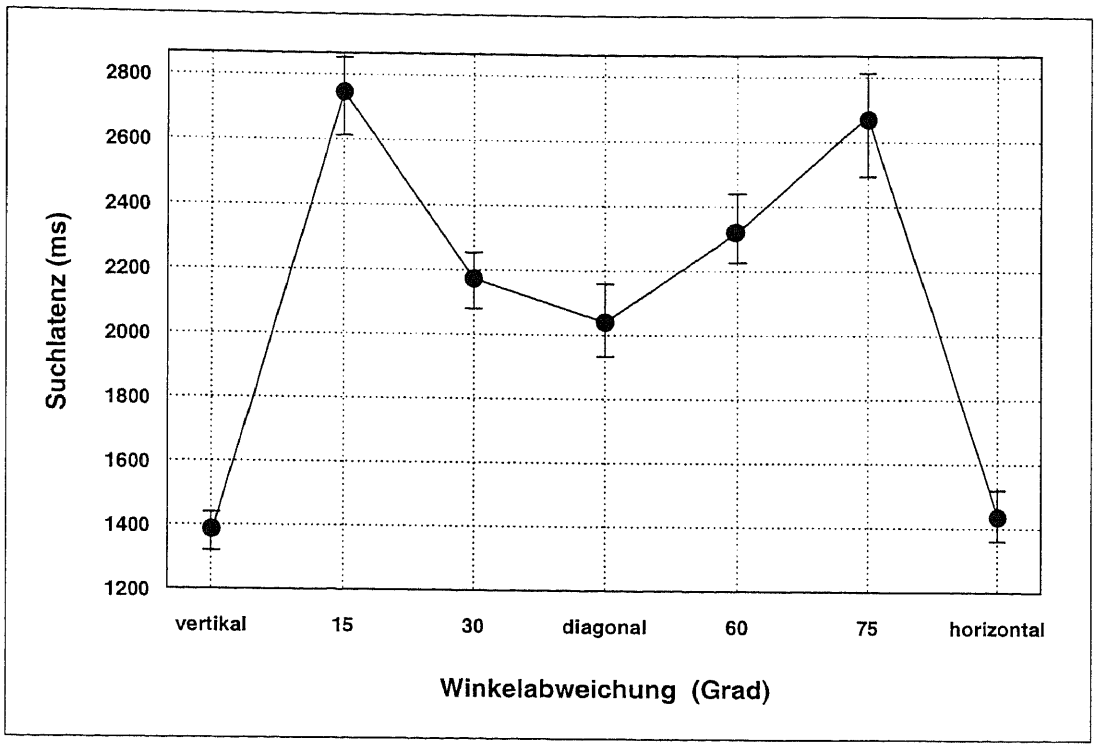

Abbildung 4.20: Die Suchlatenzen (in ms) in Abhängigkeit von der Winkelabweichung des LO

Abbildung 4.20 illustriert das Ergebnis der Analyse der bei der Suche des Rezipienten nach dem LO auftretenden Latenzzeiten mit ihrem jeweiligen Standardfehler. Bei der Darstellung wurden die Verteilungen der Quadranten so zusammengefaßt, daß *links* und *rechts* der horizontalen, *oben* und *unten* der vertikalen Achse und Kombinationen den Diagonalen zugeordnet wurden. Die Suchlatenzen ähneln stark den Produktionslatenzen. Die Zeiten steigen von vertikal (1400 ms) über horizontal (1448 ms) nach diagonal (2050 ms $[p < 0,001]$) und erreichen nach 2177 ms bzw. 2352 ms für Abweichungen von +/- 15 Grad von der Hauptdiagonalen ihr Maximum bei +/- 30 Grad mit 2739 ms bzw. 2712 ms $[p < 0,01]$.

Auch hier zeigt sich, daß ein Konfliktbereich existiert, der eine Lokalisierung verlangsamt. Interessant ist nun die Tatsache, daß anhand der Daten ein schnelleres Auffinden mittels elementarer Präpositionen ermittelt wurde, der Produzent hier aber kombinierte bevorzugt. Scheinbar folgt dieser räumliche Dialog nicht dem oben erwähnten Grice'schen Kooperationsprinzip.

Abschließend sind in Abb. 4.21(b) die Ergebnisse einer Simulation der eben beschriebenen Experimentalserie durch BOLA zu sehen.

Es zeigt sich, daß qualitativ ähnliche Resultate geliefert werden: Die kanonischen Relationen werden am schnellsten generiert, gefolgt von den prototypischen Diagonalen, während der Konfliktbereich auch hier die längsten

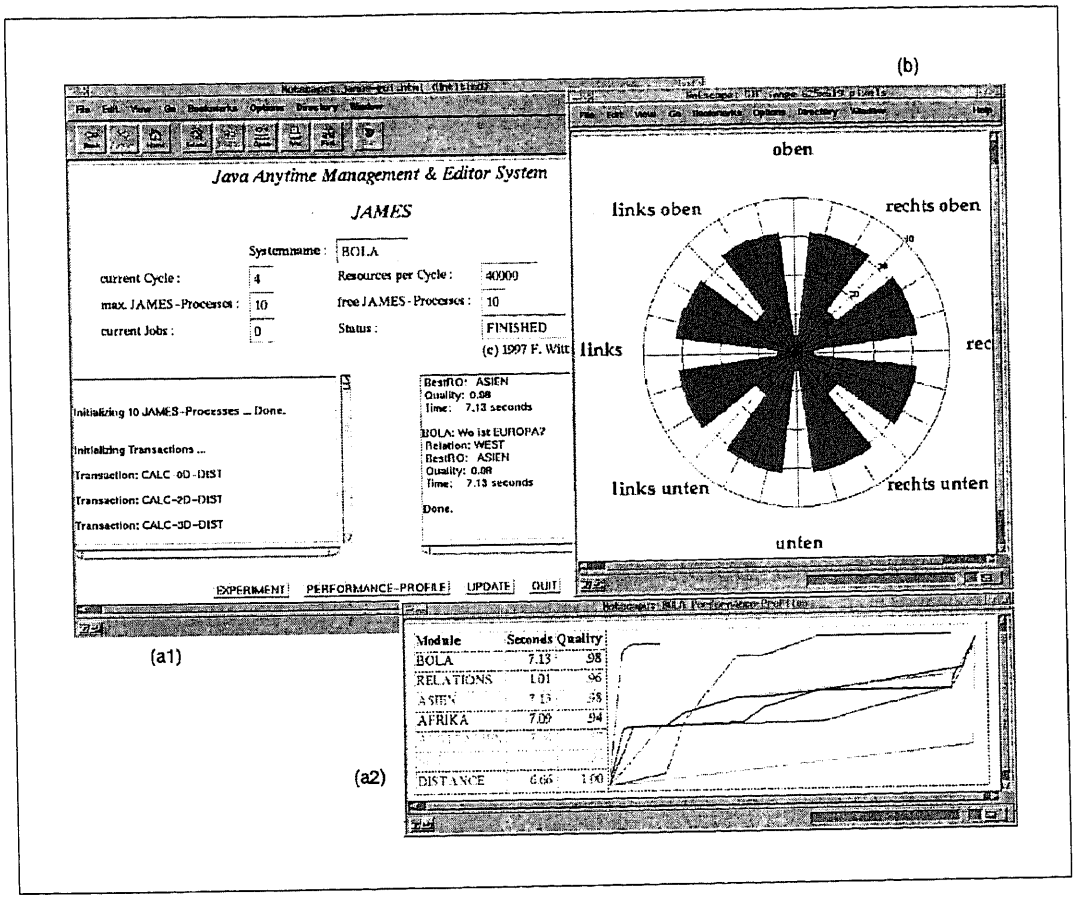

Abbildung 4.21: BOLA: (a1) graphische Oberfläche mit Beispielanfrage, (a2) ausgewählte Performanzprofile zu (a1), (b) Ergebnisse eines simulierten Experimentaldurchlaufs

Latenzzeiten aufweist. Er erstreckt sich allerdings über eine größere Winkelabweichung.

Für die Rekonstruktion der experimentellen Ergebnisse ergibt sich folgender Ablauf: Die Anfrage wird so parametrisiert, daß nur elementare und kombinierte winkelabhängige Relationen betrachtet werden, die auf dem Ergebnis des Halbraummodells basieren. Bei einer Anfrage unter Zeitdruck spielt die absolute Qualität einer Relation eine untergeordnete Rolle: Relevant ist ihre relative Qualität gegenüber einer oder mehrerer Konkurrenzrelationen. Für den konkreten vorliegenden Fall bedeutet dies, daß die Berechnung abgebrochen und das erzielte Ergebnis – auch wenn es eventuell nicht das optimale sein sollte – ausgegeben werden kann, wenn die entsprechende Relation signifikant anwendbarer ist als alle anderen bis jetzt berechneten. Liefert also das Halbraummodell ein Resultat, nach dem eine kanonische Relation deutlich besser anwendbar ist als andere, so sollte sie geäußert werden und das System terminieren. Kann durch die Modulkontrolle keine eindeutige Entscheidung getroffen werden, folgt die nächste Relationen-Ebene, in diesem Fall also, da keine distanzabhängigen Relationen betrachtet werden, die winkelabhängige Kombination im Halbraummodell. Führt auch dieser Schritt

zu keinem abschließenden Ergebnis, d.h. liegt das zu lokalisierende Objekt nicht ziemlich genau auf einer Diagonalen, so müssen Anwendbarkeits- und Präzisionsgrade berechnet und verglichen werden, was zu einer wesentlich längeren Laufzeit führt: Wir befinden uns im Konfliktbereich.

4.5 Zusammenfassung

Diese Kapitel behandelte am Beispiel des *beschränkt-optimalen Lokalisationsagenten* BOLA die Integration der zuvor entwickelten Verfahren zur ressourcenadaptierenden Generierung natürlichsprachlicher Raumbeschreibungen.

Den Schwerpunkt der Darstellung bildete die Beschreibung der beteiligten Module, die als in eigene Anytime-Shells gekapselte Subsysteme in nebenläufigen Prozessen unabhängige Teilaufgaben lösen, wie die Bewertung der Eigenschaften potentieller Referenzobjekte und die Bestimmung der am besten anwendbaren räumlichen Relation. Die für die Klasse der räumlichen Relationen erstellte Ontologie ist die Grundlage für die Modellierung der entsprechenden Berechnungsverfahren und die Verknüpfungen bezüglich der jeweiligen Abhängigkeiten, auf die detailliert eingegangen wurde.

Eine Optimierung der Ressourcenverteilung wird durch die Anwendung unterschiedlicher, der jeweiligen Ressourcenlage angepaßten Problemlösungsstrategien erreicht. Dazu werden während eines Systemlaufs alle relevanten Performanzdaten zur Laufzeit mit Erkenntnissen aus früheren Durchgängen korreliert und eine immer bessere Ressourcenverteilung erlernt. Das Ergebnis ist ein ressourcenadaptierendes System, dessen Gesamtperformanz optimiert ist.

Damit ist die Beschreibung von Aufbau und Verarbeitungsweise des beschränkt-optimalen Lokalisationsagenten, seiner relevanten Subsysteme und seiner Fähigkeiten abgeschlossen. Der folgende, letzte Teil IV befaßt sich mit einer Diskussion des bisher Erreichten und beleuchtet zukünftig noch zu bearbeitende Aspekte.

Teil IV

Zusammenfassung und Ausblick

Zusammenfassung

Ziel dieser Arbeit war die Entwicklung eines beschränkt-optimalen Lokalisationsagenten zur ressourcenadaptierenden Raumbeschreibung. Ausgangspunkt war die Absicht, an diesem Beispiel aufzuzeigen, wie Systeme der Künstlichen Intelligenz situationsbedingte, veränderliche Ressourcenbeschränkungen – wie sie insbesondere bei Wegbeschreibungen auftreten – zur Laufzeit berücksichtigen und dadurch dem menschlichen Benutzer zu jedem Zeitpunkt adäquate Resultate auf seine Anfragen liefern können. Den Hintergrund stellte die enge Verknüpfung mit Ergebnissen aus psychologischen Untersuchungen zu ressourcenadaptiven kognitiven Prozessen auf dem Gebiet der natürlichsprachlichen Raum- und Wegbeschreibung dar.

Die Integration der beiden Bereiche Raumbeschreibung und Ressourcenadaptivität machte es erforderlich, bereits in den einzelnen Gebieten existierende Modelle an die erhöhten Anforderungen anzupassen und neue Berechnungsverfahren zu erarbeiten. Hierbei wurde auf den modularen Charakter der einzelnen Bausteine größten Wert gelegt. Zu den bereits vorhandenen Vorarbeiten, auf die teilweise Bezug genommen werden konnte, gehören Ansätze zur Formalisierung und Modellierung räumlicher Relationen sowie unscharfer Konzepte ebenso wie zur Konstruktion unterbrechbarer Systeme. Diese Faktoren stellen elementare Bestandteile eines beschränkt-optimalen Lokalisationsagenten dar. Allerdings konnten sie nicht direkt und ohne weiteres zu einem entsprechenden Gesamtsystem zusammengefügt werden. Hierzu waren vielfältige Neu- und Weiterentwicklungen erforderlich.

Das System stützt sich in mehrfacher Hinsicht auf experimentelle Ergebnisse: Neben älteren Untersuchungen zu winkelabhängigen Relationen ist hier ganz besonders die Validierung des Ansatzes hinsichtlich der Produktionslatenzen anhand der in Zusammenarbeit mit der kognitiven Psychologie durchgeführten Versuchsreihen hervorzuheben.

Im ersten Teil dieser Arbeit wurden vier wissenschaftliche Fragestellungen hervorgehoben, denen in diesem Kontext nachgegangen werden sollte.

Die erste Frage behandelte dynamische Raumkonzepte, ihre Definition und Formalisierung sowie ihre Integration in ein entsprechendes KI-System.

Hervorzuheben ist hier als Ergebnis dieser Arbeit die Bestimmung einer Klasse von dynamischen räumlichen Relationen, die erstmals die Menge der (bislang im wesentlichen betrachteten statischen) räumlichen Relationen um solche Konzeptualisierungen, die wegbezogenen Charakter haben, die sogenannten n-Punkt- bzw. Pfadrelationen, ergänzt. Dabei ist es gelungen, deren Formalisierung an denselben essentiellen Parametern (Distanz und Winkelanweichung) zu verankern, wie im Basismodell. Damit konnte die erforderliche durchgängige Vergleichbarkeit über alle Relationenklassen auf dem Gebiet der Raumbeschreibung gewährleistet werden.

Die zweite Frage beschäftigte sich mit Definition, Formalisierung und Integration von linguistischen Heckenausdrücken vor dem Hintergrund eines Modells räumlicher Relationen.

In dieser Arbeit ist es durch die mehrstufige Abbildung einer für die Raumbeschreibung ausreichenden Teilmenge unscharfer Konzepte gelungen, ein funktionsfähiges System zu erzeugen, das beliebige gradierte Konzepte mit linguistischen Heckenausdrücken verknüpft. In diesem Zusammenhang wurde das erprobte Konzept des Anwendbarkeitsgrades um den Präzisionsgrad ergänzt. Damit ist es möglich geworden, beispielsweise räumliche Konzepte nicht nur bezüglich ihrer inhärenten Qualität zu bewerten, sondern auch bezüglich ihres Nutzens, in diesem Falle die möglichst starke Einschränkung eines Suchraumes. Auch in diesem Fall stellt das entwickelte Modell zur Behandlung unscharfer Konzepte eine Neuheit dar, da es bisher lediglich einzelne, eher unvollständige Ansätze gab, die mehrheitlich das Manko hatten, unflexibel zu sein.

Die dritte Frage betraf die Auswirkungen von Ressourcenbeschränkungen auf räumliche Beschreibungen und ihre Berücksichtigung in einem KI-System.

Raumkognitionsexperimente in der Psychologie, wie sie z.B. auch im Rahmen des Sonderforschungsbereichs 378 durchgeführt wurden, zeigen, daß Menschen bei der Kommunikation über Raum- und Wegbeschreibungen unterschiedliche, der jeweiligen Ressourcenlage angepaßte Strategien verwenden. Um dieser Erkenntnis in der vorliegenden Arbeit zu entsprechen, wurden die verwendeten Methoden zur Bestimmung eines besten Referenzobjektes und der zugehörigen besten räumlichen Relation sowie einer eventuellen linguistischen Hecke konsequent als Anytime-Algorithmen realisiert. Dabei wurde neben der jederzeitigen Unterbrechbarkeit das Hauptaugenmerk auf das Vorhandensein unterschiedlicher Problemlösungsstrategien gelegt, die sich etwa in der Basierung der Berechnungsverfahren auf binären oder gradierten Raumkonzepten manifestieren. Dies erfolgte im Rahmen der Integration von Raumbeschreibung und Ressourcenadaptivitätder durch die Erarbeitung einer funktionalen Ontologie räumlicher Relationen.

Die vierte und letzte Frage befaßte sich mit den notwendigen Eigenschaften und Fähigkeiten für eine dynamische Optimierung der Auswirkungen von Ressourcenbeschränkungen und mit konkreten Modellen zur Ressourcenallokation.

Die Anforderungen an ein System zur Generierung räumlicher Beschreibungen, wie es in dieser Arbeit vorgestellt wurde, liegen vor allem in einer jederzeitigen Unterbrechbarkeit der Beantwortung einer Anfrage (in Sinne

einer jederzeitigen *Abrufbarkeit* des Resultats), einhergehend mit einer Verbesserung der Qualität des Ergebnisses bei steigendem Ressourceneinsatz. Davon ausgehend wurde in dieser Arbeit zum ersten Mal eine ressourcensensitive Architektur zur Konstruktion symbolverarbeitender Systeme entwickelt. Diese Architektur verfügt über einen rekursiven Shell-Charakter, der es prinzipiell erlaubt, beliebige Systeme aufzunehmen. Auch im Bereich der Ressourcensensitivität gab es etwa bei den unterbrechbaren Algorithmen zahlreiche Vorarbeiten – hier ist insbesondere Zilberstein zu nennen, dessen Ideen diese Arbeit stark beeinflußt haben – ohne daß *konkrete* Übertragungen dieser Konzepte auf das Gebiet der KI gelungen wären. Durch die ressourcensensitive Architektur verfügen die entsprechend erzeugten Gesamtsysteme über die Eigenschaft der beschränkten Optimalität, sind jederzeit unterbrechbar und liefern ein sich mit wachsendem Ressourceneinsatz verbesserndes Resultat. Beschränkte Optimalität wird dabei durch die Möglichkeit paralleler Verarbeitung und insbesondere – wie oben beschrieben – durch die ressourcenadaptierende, d.h. zur Laufzeit an die Ressourcenlage angepaßt erfolgende Bearbeitung unterschiedlicher Problemlösungsstrategien erreicht. Zur Optimierung der Ressourcenverteilung wurden eine Reihe von Verfahren entwickelt und implementiert, die selbst wieder ressourcensensitiv sind. Die zugrundeliegenden Modelle basieren auf unterschiedlichen Strategien zur Analyse früherer Systemläufe, so daß ein sich stetig verbesserndes Verhalten erlernt wird.

Das Endergebnis bildet ein System der Künstlichen Intelligenz, das dem Ziel einer ressourcenadaptierenden Raumbeschreibung sehr nahe kommt. Dennoch gibt es eine Reihe von offenen Fragen, die Möglichkeiten der Weiterentwicklung in künftiger Forschungsarbeit bieten.

Ausblick

An erster Stelle soll hier auf die weitere Validierung des Ansatzes eingegangen werden, die natürlich für alle Aspekte erfolgen sollte. So werden gegenwärtig Experimente zu pfadbezogen Relationen durchgeführt, deren Auswertung in die entsprechenden Berechnungsverfahren einfließen sollte. Erste kursorische Analysen lassen auf eine große Übereinstimmung zwischen den Ergebnissen der Versuchsreihen und der gewählten Formalisierung schließen. Ein interessanter Punkt bei der Verarbeitung dieser Relationenklasse ist die Frage der Segmentierung komplexer Trajektorien, die sich nur unzureichend mit einer einzelnen Relation beschreiben lassen, um eine feinere Granularität und damit präzisere Ergebnisse zu erhalten. Denkbar ist die Verwendung des vorliegenden Ansatzes in diesem Sinne bis hin zu einer rekursiven Vorgehensweise, die den ressourcensensitiven Charakter des Gesamtsystems weiter unterstützen würde.

Problematisch für ein Realzeit-System ist gegenwärtig noch die teilweise recht große Transaktionslänge. Eine Verkürzung wäre erstrebenswert, da sie die *jederzeitige* Unterbrechbarkeit durch ein variableres Scheduling fördern und gegebenenfalls zu verkürzten Antwortzeiten führen würde.

Noch nicht vollzogen wurde der Übergang von fixen Idealisierungsklassen zu einer zur Laufzeit generierten, ressourcenadaptiven Abstraktionsgenerierung, die gerade im vorliegenden Kontext von Vorteil wäre. Hier sind die Vorarbeiten erfolgreich abgeschlossen, so daß einer baldigen Integration nichts im Wege steht, da die Berechnungsverfahren etwa der räumlichen Relationen selbst nicht verändert werden müssen.

Auf dem Gebiet der Ressourcensensitivität bietet sich als Erweiterungsmöglichkeit insbesondere die Berücksichtigung zusätzlicher Ressourcen – neben der Zeit und der Prozeßanzahl – an. Hier könnte man beispielsweise an die Arbeitsspeicherkapazität denken und der Frage nachgehen, ob und wie eine Korrelation auf kognitiver Seite mit dem menschlichen Arbeitsgedächtnis vorliegt. In diesem Zusammenhang interessiert auch die Möglichkeit, die jeweilige Belastung durch eine geplante Äußerung des Arbeitsgedächtnisses des Rezipienten zu antizipieren und diese adäquat zu gestalten, so daß sie ein Hörer beispielsweise auch in Streßsituationen memorieren kann. Das Teilprojekt READY des Sonderforschungsbereichs 378 modelliert u.a. die Belastung des Arbeitsgedächtnisses in Dialogsituationen und kann zu diesem Thema wertvolle Ergebnisse liefern (vgl. (Jameson, Schäfer, Weis, Berthold & Weyrath, 1999)).

Auch auf dem besonders komplexen Gebiet der unscharfen Konzepte, das im Sinne dieser Arbeit nur geringe Vorleistungen mitbringt, bieten sich einige Erweiterungen an: Zunächst ist an einen Ausbau des Korpus im allgemeinen zu denken, um die Menge möglicher linguistischer Hecken auch für andere Anwendungen besser abzudecken. Ferner muß neben einer Verbesserung der schwierigen Negationsbehandlung auch die Wirkungsweise einzelner Hecken, die Frage ihrer Anwendbarkeit auf einzelne räumliche oder andere Relationen oder Kombinationen von diesen untersucht werden.

Weitere Verbesserungsmöglichkeiten in diesem Bereich liegen in einer intelligenteren Auswahl der Hecken, die eventuell auch auf Performanzprofile zurückgreifen könnte. Ähnliches gilt für die Wahl des in der jeweiligen Situation am besten geeigneten Präzisionsgrades. Auch hier könnte das Ausnutzen von Erkenntnissen aus früheren Systemläufen zu einer Verbesserung führen.

Zu allen diesen Punkten sollten unterstützend experimentelle Versuchreihen durchgeführt und deren Resultate bei Formalisierung und Modellierung berücksichtigt werden.

Bislang nur angedacht wurden zwei Aspekte, die über die reine Verknüpfung von Relationen mit Heckenausdrücken weit hinausgehen: Zum einen bietet es sich an, den Einfluß von Objektcharakteristika wie Größe oder Farbe auf die Anwendung linguistischer Hecken zu untersuchen. Zum anderen wäre – gerade im Kontext einer ressourcenadaptierenden Generierung sprachlicher Raumbeschreibungen eine bewertende Kommentierung der gesamten Äußerung wünschenswert, um so dem Hörer in ihr enthaltene Zusatzinformation mitzuteilen. Dies korrespondiert mit dem Gesichtspunkt des Vertrauens als Qualitätsmetrik.

Letztlich ist eine Erprobung der Anwendung aller allgemein verwendbaren Teile des Gesamtsystems und seiner Module in anderen Domänen von großem Interesse, um die Robustheit der gewählten Verfahren zu validieren.

Abschließend kann gesagt werden, daß das in dieser Arbeit vorgestellte System eines beschränkt-optimalen Lokalisationsagenten zur ressourcenadaptierenden Raumbeschreibung einen erfolgreichen Schritt in Richtung auf eine verbesserte Mensch-Maschine-Kommunikation und Mensch-Technik-Interaktion darstellt.

Appendix

Literaturverzeichnis

André, E., Bosch, G., Herzog, G. & Rist, T. (1986). Characterizing Trajectories of Moving Objects Using Natural Language Path Descriptions. In *ECAI86P2* (Vol. 2, S. 1-8). Brighton, UK.

André, E., Bosch, G., Herzog, G. & Rist, T. (1987). Coping with the Intrinsic and the Deictic Uses of Spatial Prepositions. In K. Jorrand & L. Sgurev (Hrsg.), *Artificial Intelligence II: Methodology, Systems, Applications* (S. 375-382). Amsterdam: North-Holland.

André, E., Herzog, G. & Rist, T. (1988). On the Simultaneous Interpretation of Real World Image Sequences and their Natural Language Description: The System SOCCER. In *Proc. of the 8th ECAI* (S. 449-454). München.

André, E., Herzog, G. & Rist, T. (1989). Natural Language Access to Visual Data: Dealing with Space and Movement. In *Proc. of the 1st Workshop on Logical Semantics of Time, Space and Movement in Natural Language.* Toulouse, France.

Austin, J. (1962). *How to do Things with Words.* New York: Oxford University Press.

Barkowsky, T., Berendt, B., Freksa, C. & Kelter, S. (1997). Aspektkarten - Integriert räumlich-symbolische Repräsentationsstrukturen. In C. Umbach, M. Grabski & R. Hörnig (Hrsg.), *Perspektive in Sprache und Raum.* Wiesbaden: Deutscher Universitätsverlag.

Baus, A. & Beckert, A. (1998). *ORCAN - Implementation von Methoden zur Compilierung von Anytime-Algorithmen* (Abschlußbericht Fortgeschrittenen Praktikum). Saarbrücken: Projekt REAL, Fachbereich für Informatik, Univ. des Saarlandes.

Beckert, A. (2000). *Analyse von Verfahren zur Ressourcen-Verteilung bei Anytime-Algorithmen.* Diplomarbeit, Fachbereich Informatik, Univ. des Saarlandes. (In Vorbereitung.)

Blocher, A. (1994). *KOREF: Zum Vergleich intendierter und imaginierter Äußerungsgehalte.* Diplomarbeit, Fachbereich Informatik, Univ. des Saarlandes.

Blocher, A., Essig, F., Krüger, A. & Maaß, W. (1999). Towards a Computational Semantics of Path Relations. In P. Olivier (Hrsg.), *Language and Space: cognitive and computational views.* Kluwer Academic Publisher.

Blocher, A. & Schirra, J. (1995). Optional Deep Case Filling and Focus Control with Mental Images: ANTLIMA-KOREF. In *Proc. of the 14th IJCAI* (S. 417-423). Montréal, Canada.

Blocher, A. & Stopp, E. (1998). Time-Dependent Generation of Minimal Sets of Spatial Descriptions. In P.Olivier & K.-P. Gapp (Hrsg.), *Representation and Processing of Spatial Expressions* (S. 57-72). Mahwah, NJ, USA: Lawrence Erlbaum Associates.

Boddy, M. (1991a). Anytime Problem Solving Using Dynamic Programming. In *Proc. of AAAI-91* (S. 738-743). Anaheim, CA.

Boddy, M. (1991b). Anytime Problem Solving using Dynamic Programming. In *Proc. of the Ninth National Conf. on Artificial Intelligence* (S. 738-743). Anaheim, CA.

Bolinger, D. (1972). *Degree words.* The Hague: Janua linguarum.

Bouchon-Meunier, B. (1992). Fuzzy logic and knowledge representation using linguistic modifiers. In L. A. Zadeh & J. Kacprzyk (Hrsg.), *Fuzzy Logic for the Management of Uncertainty* (S. 399-414). New York: Wiley.

Brockhaus. (1997). *Die Enzyklopädie* (20. Aufl.). Leipzig: Brockhaus. (24 Bde.)

Bußmann, H. (1990). *Lexikon der Sprachwissenschaft* (Vol. 452). Stuttgart: Kröner.

Butz, A. & Krüger, A. (1996). Lean Modeling - The intelligent use of geometrical abstraction in 3D animations. In W. Wahlster (Hrsg.), *ECAI 96 12th European Conf. on Artificial Intelligence.* John Wiley & Sons, Ltd.

Carsten, I. & Janson, T. (1985). *Verfahren zur Evaluierung räumlicher Präpositionen anhand geometrischer Szenenbeschreibungen.* Diplomarbeit, Fachbereich für Informatik, Univ. Hamburg.

Carter, R. C. & Carter, E. C. (1981). Color and Conspicousness. *Optical Society of America, 71,* 723-729.

Cleeren, R., Vandenberghe, R., Gyseghem, N. V. & Caluwe, R. D. (1993). The Modelling of Vague Predicates Used in Linguistic Expressions by Means of Fuzzy Set Theory. In *Proc. of the Fifth Int. Fuzzy Systems Association World Congress* (S. 54-57).

Dean, T. & Boddy, M. (1988). An Analysis of Time-Dependent Planning. In *Proc. of the Seventh National Conf. on Artificial Intelligence* (S. 49-54). Minneapolis, Minnesota.

Deep Map. (1999). *Intelligent next-generation Geo-Information Systems.* http://www.eml.org/englisch/projekte/deepmap/deepmap.html.

Egenhofer, M. J. (1991). Reasoning about Binary Topological Relations. In O. Günther & H.-J. Schek (Hrsg.), *Advances in Spatial Databases* (S. 144-160). Berlin, Heidelberg: Springer.

Ehrich, V. (1985). Zur Linguistik und Psycholinguistik der sekundären Raumdeixis. In H. Schweizer (Hrsg.), *Sprache und Raum* (S. 130-161). Stuttgart: Metzler.

Eschenbach, C. & Habel, C. (1995). Abstrakte Räumlichkeit in der Kognition. *Kognitionswissenschaft*(4), 171-176.

Fürnsinn, M., Khenkhar, M. & Ruschkowski, B. (1984). GEOSYS - Ein Frage-Antwort-System mit räumlichem Vorstellungsvermögen. In C.-R. Rollinger (Hrsg.), *Probleme des (Text-) Verstehens, Ansätze der Künstlichen Intelligenz* (S. 172-184). Tübingen: Niemeyer.

Gapp, K.-P. (1995a). An Empirically Validated Model for Computing Spatial Relations. In I. Wachsmuth, C.-R. Rollinger & W. Brauer (Hrsg.), *KI-95: Advances in Artificial Intelligence. 19th Annual German Conf. on Artificial Intelligence* (S. 245-256). Berlin, Heidelberg: Springer.

Gapp, K.-P. (1995b). Angle, Distance, Shape, and their Relationship to Projective Relations. In J. D. Moore & J. F. Lehman (Hrsg.), *Proc. of the 17th Annual Conf. of the Cognitive Science Society* (S. 112-117). Mahwah, NJ: Lawrence Erlbaum.

Gapp, K.-P. (1996). Selection of Best Reference Objects in Object Localizations. In *Proc. of the AAAI Spring Symposium on Cognitive and Computational Models of Spatial Representations* (S. 23-34). Stanford, CA.

Gapp, K.-P. (1997). *Objektlokalisation: Ein System zur sprachlichen Raumbeschreibung.* Wiesbaden: Deutscher Universitätsverlag.

Garvey, A., Humphrey, M. & Lesser, V. (1993). Task Interdependencies in Design-to-time Real-time Scheduling. In *Procceedings of the Eleventh National Conf. on Artificial Intelligence* (S. 580-585). Washington D.C.

Garvey, A. & Lesser, V. (1993). *A Survey of Research in Deliberative Real-Time Artificial Intelligence* (UMass Computer Science Technical Report No. 93-84). Department of Computer Science, Univ. of Massachusetts.

Garvey, A. & Lesser, V. (1994). Design-to-time Real-Time Scheduling. *IEEE Transactions on Systems, Man and Cybernetics, 23*(6), 1491-1502.

Garvey, A. & Lesser, V. (1996). Design-to-time Scheduling and Anytime-Algorithms. *SIGART Bulletin, 7*(3).

Good, I. J. (1971). Twenty-seven principles of rationality. In V. P. Godambe & D. A. Sprott (Hrsg.), *Foundation of Statistical Inference.* Toronto: Holt, Rinehart & Winston.

Görz, G. & Kesseler, M. (1994). Anytime Algorithms for Speech Parsing? In *Proc. COLING-94* (S. 997-1001). Kyoto.

Grass, J. (1996). Reasoning about computational resource allocation - An introduction to anytime algorithms. *ACM Crossroads.*

Grass, J. & Zilberstein, S. (1995). *Anytime Algorithm Development Tools* (CMOSCI No. 95-94). Univ. of Massachusetts at Amherst: Computer Science Department.

Grice, H. P. (1975). Logic and Conversation. In P. Cole & J. L. Morgan (Hrsg.), *Syntax and Semantics: Vol. 3: Speech Acts* (S. 41-58). London: Academic Press.

Habel, C. & Pribbenow, S. (1988). *Gebietskonstituierende Prozesse* (LILOG-Report No. 18). Stuttgart: IBM.

Hanßmann, K.-J. (1980). *Sprachliche Bildinterpretation für ein Frage-Antwort-System* (Bericht No. 74). Fachbereich Informatik, Univ. Hamburg.

Helbig, G. (1990). *Lexikon deutscher Partikeln.* Leipzig: Verlag Enzyklopädie.

Helbig, G. & Helbig, A. (1990). *Lexikon deutscher Modalwörter.* Leipzig: Verlag Enzyklopädie.

Herrmann, T. & Grabowski, J. (1994). *Sprechen: Psychologie der Sprachproduktion.* Berlin: Spektrum Akademischer Verlag.

Herskovits, A. (1986). *Language and Spatial Cognition. An Interdisciplinary Study of the Prepositions in English.* Cambridge, London: Cambridge University Press.

Herzog, G., Blocher, A., Gapp, K.-P., Stopp, E. & Wahlster, W. (1996). VITRA: Verbalisierung visueller Information. *Informatik - Forschung und Entwicklung, 11*(1), 12-19.

Herzog, G., Maaß, W. & Wazinski, P. (1993). VITRA GUIDE: Utilisation du Langage Naturel et de Représentation Graphiques pour la Description d'Itinéraires. In *Colloque Interdisciplinaire du Comité National „Images et Langages: Multimodalité et Modélisation Cognitive"* (S. 243-251). Paris.

Herzog, G., Rist, T. & André, E. (1990). Sprache und Raum: Natürlichsprachlicher Zugang zu visuellen Daten. In C. Freksa & C. Habel (Hrsg.), *Repräsentation und Verarbeitung räumlichen Wissens* (S. 207-220). Berlin, Heidelberg: Springer.

Herzog, G. & Rohr, K. (1995). Integrating Vision and Language: Towards Automatic Description of Human Movements. In I. Wachsmuth & C.-R. Rollinger (Hrsg.), *KI-95: Advances in Artificial Intelligence. 19th Annual German Conf. on Artificial Intelligence* (S. 257-268). Berlin, Heidelberg: Springer.

Herzog, G., Sung, C.-K., André, E., Enkelmann, W., Nagel, H.-H., Rist, T., Wahlster, W. & Zimmermann, G. (1989). Incremental Natural Language Description of Dynamic Imagery. In W. Brauer & C. Freksa (Hrsg.), *Wissensbasierte Systeme. 3. Int. GI-Kongreß* (S. 153-162). Berlin, Heidelberg: Springer.

Hirtle, S. C. & Heidorn, P. B. (1993). The Structure of Cognitive Maps: Representation and Processes. In T. Gärling & R. G. Golledge (Hrsg.), *Behaviour and Environment: Psychological and Geographical Approaches* (S. 177-189). Elsevier Science Publishers.

Horvitz, E. J. (1987). Reasoning about Beliefs and Actions under Computational Resource Constraints. In *Proc. of the Third Workshop on Uncertainty in Artificial Intelligence, Seattle WA* (S. 429-444). Mountain View, CA.

Hußmann, M. & Schefe, P. (1984). The Design of SWYSS, a Dialogue System for Scene Analysis. In L. Bolc (Hrsg.), *Natural Language Communication with Pictorial Information Systems* (S. 143-201). München: Hanser/McMillan.

Jameson, A. & Buchholz, K. (1998). Einleitung zum Themenheft „Ressourcenadaptive kognitive Prozesse". *Kognitionswissenschaft*, 7(3), 95-100.

Jameson, A., Schäfer, R., Weis, T., Berthold, A. & Weyrath, T. (1999). Making systems sensitive to the user's time and working memory constraints. In M. T. Maybury (Hrsg.), *IUI99: International Conference on Intelligent User Interfaces* (S. 79-86). New York: ACM.

Kipper, B. (1995). *Repräsentation und Verarbeitung propositionaler Einstellungen in natürlichsprachlichen Systemen.* Doktorarbeit, Univ. des Saarlandes, Saarbrücken.

Kluwe, R. (1997). Intentionale Steuerung kognitiver Prozesse. *Kognitionswissenschaft*(6), 53-69.

Kolde, G. (1986). Zur Lexikographie sogenannter Hecken-Ausdrücke. In M. Reiss (Hrsg.), *Kontroversen, alte und neue: Akten des VII. Internationalen Germanisten Kongress, Göttingen 1985* (Vol. Band 3: Textlinguistik contra Stilistik? Wortschatz und Wörterbuch). Tübingen: Niemeyer.

Königsberger, K. (1990). *Analysis 1.* Berlin, Heidelberg: Springer.

Kray, C. (1998). *Ressourcenadaptierende Verfahren zur Präzisionsbewertung von Lokalisationsausdrücken und zur Generierung von linguistischen Hecken.* Diplomarbeit, Fachbereich Informatik, Univ. des Saarlandes.

Kray, C. & Blocher, A. (1999). Modeling the Basic Meanings of Path Relations. In *Proc. of the 16th IJCAI* (S. 384-389). Stockholm, Schweden.

Krüger, A. (1999). *Automatische Abstraktion in 3D-Graphiken.* Doktorarbeit, Technische Fakultät, Univ. des Saarlandes, Saarbrücken.

Lakoff, G. (1973). Hedges: A Study in Meaning Criteria and the Logic of Fuzzy Concepts. *Journal of Philosophical Logic, 2*, 458-508.

Landau, B. & Jackendoff, R. (1993). „What" and „Where" in Spatial Language and Spatial Cognition. *Behavioral and Brain Sciences, 16*, 217-265.

Längle, T., Lüth, T., Herzog, G., Stopp, E. & Kamstrup, G. (1995). KANTRA - A Natural Language Interface for Intelligent Robots. In U. Rembold, R. Dillman, L. O. Hertzberger & T. Kanade (Hrsg.), *Proc. of the 4th Int. Conf. on Intelligent Autonomous Systems.*

Lewis, H. R. & Papdimitriou, C. H. (1981). *Elements of the Theory of Computation.* Prentice-Hall International, Inc.

Maaß, W. (1994). From Visual Perception to Multimodal Communication: Incremental Route Descriptions. *Artificial Intelligence Review Journal, 8*(5/6), 159-174. (Special Volume on Integration of Natural Language and Vision Processing.)

Maaß, W. (1996). *Von visuellen Daten zu inkrementellen Wegbeschreibungen in dreidimensionalen Umgebungen: Das Modell eines kognitiven Agenten.* Doktorarbeit, Technische Fakultät, Univ. des Saarlandes, Saarbrücken.

Menzel, W. (1994). Parsing Natural Language under Time Constraints. In A. G. Cohn (Hrsg.), *Proc. of the 11th European Conf. of Artificial Intelligence* (S. 560-564). Amsterdam: Chichester: Wiley.

Miller, G. A. & Johnson-Laird, P. N. (1976). *Language and Perception.* Cambridge, London: Cambridge University Press.

Nagel, H.-H. (1985). Analyse und Interpretation von Bildfolgen. Teil I und II. *Informatik Spektrum, 8*, 178-200, 312-327.

Nagel, H.-H. (1988). From Image Sequences Towards Conceptual Descriptions. *Image and Vision Computing, 6*(2), 59-74.

Neumann, B. & Novak, H.-J. (1986). NAOS: Ein System zur natürlichsprachlichen Beschreibung zeitveränderlicher Szenen. *Informatik Forschung und Entwicklung, 1*, 83-92.

Neumann, J. von & Morgenstern, O. (1947). *Theory of games and economic behavior.* Princeton, New Jersey: Princeton University Press.

Pinkal, M. (1985). Kontextabhängigkeit, Vagheit, Mehrdeutigkeit. In C. Schwarze & D. Wunderlich (Hrsg.), *Handbuch der Lexikologie* (S. 27-63). Königstein/Ts.: Athendum.

Powitz, B. (1993). *Zur Automatisierung der Kartographischen Generalisierung topographischer Daten in Geo-Informationssystemen.* Doktorarbeit, Univ. Hannover.

Pribbenow, S. (1991). *Zur Verarbeitung von Lokalisierungsausdrücken in einem hybriden System.* Doktorarbeit, Fachbereich Informatik, Univ. Hamburg.

REAL. (1996). *Ressourcen-adaptive Lokalisation: Interaktion von Objektlokalisation und Sprachproduktion.* URL: http://w5.cs.uni-sb.de/real/.

Retz-Schmidt, G. (1988). Various Views on Spatial Prepositions. *AI Magazine, 9*(2), 95-105.

Retz-Schmidt, G. (1992). *Die Interpretation des Verhaltens mehrerer Akteure in Szenenfolgen.* Berlin, Heidelberg: Springer.

Rupp, U. (1996). *GRATOR - Räumliches Schließen mit GRAdierten TOpologischen Relationen über Punktmengen.* Diplomarbeit, Univ. des Saarlandes, Saarbrücken.

Russell, S. & Norvig, P. (1995). *Artificial Intelligence - A Modern Approach.* Prentice Hall, Inc.

Russell, S. J. & Subramanian, D. (1995). Provably bounded-optimal agents. *Journal of Artificial Intelligence Research, 1*(1), 1-36.

Russell, S. J. & Wefald, E. (1991). *Do the right thing: studies in limited rationality.* Cambridge, MA: MIT Press.

Sadalla, E., Burroughs, W. J. & Staplin, L. (1980). Reference Points in Spatial Cognition. *Journal of Experimental Psychology: Human Learning and Memory, 6*(5), 516-528.

Schirra, J. R. J. (1994). *Bildbeschreibung als Verbindung von visuellem und sprachlichem Raum: Eine interdisziplinäre Untersuchung von Bildvorstellungen in einem Hörermodell.* St. Augustin: infix.

Schirra, J. R. J., Bosch, G., Sung, C.-K. & Zimmermann, G. (1987). From Image Sequences to Natural Language: A First Step Towards Automatic Perception and Description of Motions. *Applied Artificial Intelligence, 1*, 287-305.

Schober, M. F. (1995). Speakers, Adressees, and Frames of Reference: Whose Effort is Minimized in Conversations About Locations. *Discourse Processes*(20), 219-247.

Schweizer, H. (Hrsg.). (1985). *Sprache und Raum.* Stuttgart: Metzler.

Searle, J. R. (1969). *Speech Acts: An Essay in the Philosophy of Language.* Cambridge, London: Cambridge University Press.

Searle, J. R. (1994). *Sprechakte: Ein sprachphilosophischer Essay* (Vol. 458, 6 ed.). Frankfurt/Main: Suhrkamp Taschenbuch.

SFB 378. (1997). *Sonderforschungsbereich 378 „Ressourcenadaptive kognitive Prozesse".* URL http://www.coli.uni-sb.de/sfb378/.

Silberschatz, A. & Galvin, P. B. (1994). *Operating System Concepts* (4. ed.). Addison Wesley.

Socher, G., Fink, G., Kummert, F. & Sagerer, G. (1996). A Hybrid Approach to Identifying Objects from Verbal Descriptions. In *Proc. of the Workshop on Multi-Lingual Spontaneous Speech Recognition in Real Environments.* Nancy, France.

Stopp, E. (1998). *Natürlichsprachliche Dialoge mit mobilen Robotern auf der Basis einer ressourcenadaptiven Referenzsemantik räumlicher Umgebungen.* Doktorarbeit, Univ. des Saarlandes, Saarbrücken.

Stopp, E. & Blocher, A. (1996). Construction of Mental Images and their Use in a Listener Model. In R. Meyer-Klabunde & C. von Stutterheim (Hrsg.), *Proc. of the Workshop on Conceptual and Semantic Knowledge in Language Production, 15.-17. November 1995* (Vol. 2, S. 270-280). Heidelberg, Germany: Special Collaboration Program 245 (Language and Situation), Univ. of Heidelberg.

Stopp, E., Gapp, K.-P., Herzog, G., Längle, T. & Lüth, T. C. (1994). Utilizing Spatial Relations for Natural Language Access to an Autonomous Mobile Robot. In B. Nebel & L. Dreschler-Fischer (Hrsg.), *KI-94: Advances in Artificial Intelligence* (S. 39-50). Berlin, Heidelberg: Springer.

Talmy, L. (1983). How Language Structures Space. In H. Pick & L. Acredolo (Hrsg.), *Spatial Orientation: Theory, Research and Application* (S. 225-282). New York, London: Plenum.

Tanenbaum, A. S. (1992). *Modern Operating Systems.* Prentice-Hall International, Inc.

Treisman, A. (1988). Features and Objects: The Fourteenth Bartlett Memorial Lecture. *The Quarterly Journal of Experimental Psychology, 40a*(2), 201-237.

Vorwerg, C., Socher, G., Fuhr, T., Sagerer, G. & Rickheit, G. (1997). Projective Relations for 3D Space: Computational Model, Application, and Psychological Evaluation. In *Proc. of AAAI-97, Providence, Rhode Island, USA* (S. 159-164). Cambridge, MA: MIT Press.

Wahlster, W. (1977). *Die Repräsentation von vagem Wissen in natürlichsprachlichen Systemen der Künstlichen Intelligenz* (Bericht No. 38). Fachbereich Informatik, Univ. Hamburg.

Wahlster, W. (1989). One Word Says More Than a Thousand Pictures. On the Automatic Verbalization of the Results of Image Sequence Analysis Systems. *Computers and Artifial Intelligence*, *8*, 479-492.

Wahlster, W., Blocher, A., Baus, J., Stopp, E. & Speiser, H. (1998). Ressourcenadaptierende Objektlokalisation: Sprachliche Raumbeschreibung unter Zeitdruck. *Kognitionswissenschaft, Sonderheft zum Sonderforschungsbereich 378*, *7*(3), 111-117.

Wahlster, W., Marburger, H., Jameson, A. & Busemann, S. (1983). Overanswering Yes-No Questions: Extended Responses in a NL Interface to a Vision System. In *Proc. of the 8th IJCAI* (S. 643-646). Karlsruhe, FRG.

Wahlster, W. & Tack, W. (1997). SFB 378: Ressourcenadaptive Kognitive Prozesse. In M. Jarke, K. Pasedach & K. Pohl (Hrsg.), *Informatik'97 - Informatik als Innovationsmotor, 27. Jahresta gung der Gesellschaft für Informatik, Aachen, 24.-26. September 1997* (S. 51-57). Berlin, Heidelberg: Springer.

Wang, P. (1996). Problem-Solving under Insufficient Resources. In *Working Notes of the AAAI Fall Symposium on Flexible Computation.* Cambridge, Massachusetts.

Werner, S., Krieg-Brückner, B., Mallot, H., Schweizer, K. & Freksa, C. (1997). Spatial Cognition: The Role of Landmark, Route, and Survey Knowledge in Human and Robot Navigation. In *Informatik '97.* Berlin, Heidelberg: Springer.

Wittgenstein, L. (1953). *Philosophische Untersuchungen.* Oxford: Blackwell.

Wittig, F. (1998). *Ein Java-basiertes System zum Ressourcenmanagement in Anytime-Systemen.* Diplomarbeit, Fachbereich Informatik, Univ. des Saarlandes.

Wunderlich, D. (1982). Sprache und Raum. *Studium Linguistik, 12/13*, 1-19, 37-59.

Zadeh, L. A. (1965). Fuzzy Sets. *Information and Control*, *8*, 338-353.

Zadeh, L. A. (1972). A Fuzzy-Set Theoretic Interpretation of Linguistic Hedges. *Journal of Cybernetics*, *2*(3), 4-34.

Zadeh, L. A. (1975). Fuzzy Logic and Approximate Reasoning. *Synthese*, *30*, 407-428.

Zadeh, L. A. (1993). Fuzzy Sets. In D. Dubois, H. Prade & R. R. Yager (Hrsg.), *Readings in Fuzzy Sets for Intelligent Systems* (S. 27-64). San Mateo, CA: Morgan Kaufmann.

Zilberstein, S. (1993). *Operational Rationality through Compilation of Anytime Algorithms.* Doktorarbeit, Univ. of California at Berkeley, Berkeley.

Zilberstein, S. (1996). Using Anytime Algorithms in Intelligent Systems. *AI Magazine, 17*(3), 73-83.

Zilberstein, S. & Russell, S. (1996). Optimal composition of real-time systems. *Artificial Intelligence, 1-2*(82), 181-213.

Zimmer, H., Speiser, H., Baus, J., Blocher, A. & Stopp, E. (1998). The Use of Locative Expressions in Dependence of the Spatial Relation between Target and Reference Object in Two-Dimensional Layouts. In C. Freksa, C. Habel & K. F. Wender (Hrsg.), *Spatial Cognition - An interdisciplinary approach to representation and processing of spatial knowledge.* Berlin, Heidelberg: Springer.

Zimmermann, H.-J. (1987). *Fuzzy sets, decision making and expert systems.* Boston: Kluwer Academic Publishers.

Abbildungsverzeichnis

Theoremverzeichnis