



Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Document
D-91-17

**Analyse der Planungsverfahren der KI
im Hinblick auf ihre Eignung für
die Arbeitsplanung**

Andreas Becker

Dezember 1991

**Deutsches Forschungszentrum für Künstliche Intelligenz
GmbH**

Postfach 20 80
D-6750 Kaiserslautern, FRG
Tel.: (+49 631) 205-3211/13
Fax: (+49 631) 205-3210

Stuhlsatzenhausweg 3
D-6600 Saarbrücken 11, FRG
Tel.: (+49 681) 302-5252
Fax: (+49 681) 302-5341

Deutsches Forschungszentrum für Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern und Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Daimler Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Krupp-Atlas, Mannesmann-Kienzle, Philips, Sema Group Systems, Siemens and Siemens-Nixdorf. Research projects conducted at the DFKI are funded by the German Ministry for Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct *systems with technical knowledge and common sense* which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- Intelligent Engineering Systems
- Intelligent User Interfaces
- Intelligent Communication Networks
- Intelligent Cooperative Systems.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Prof. Dr. Gerhard Barth
Director

**Analyse der Planungsverfahren der KI im Hinblick
auf ihre Eignung für die Arbeitsplanung**

Andreas Becker

DFKI-D-91-17

Dieser Bericht ist unter Anleitung von Herrn Prof. Michael M. Richter, Herrn Dipl.-Inform Christoph Klauck und Herrn Dipl.-Ing. Ralf Legleitner angefertigt worden.

Die Anfertigung wurde finanziell unterstützt durch das Bundesministerium für Forschung und Technologie (FKZ ITW-8902 C4)

© Deutsches Forschungszentrum für Künstliche Intelligenz 1991

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Deutsches Forschungszentrum für Künstliche Intelligenz, Kaiserslautern, Federal Republic of Germany; an acknowledgement of the authors and individual contributors to the work; all applicable portions of this copyright notice. Copying, reproducing, or republishing for any other purpose shall require a licence with payment of fee to Deutsches Forschungszentrum für Künstliche Intelligenz.

**Analyse der Planungsverfahren der KI
im Hinblick auf ihre Eignung
für die Arbeitsplanung**

Andreas Becker

Dezember 1991

Inhaltsverzeichnis

1	Einleitung	5
2	Planungsverfahren in der Künstlichen Intelligenz	7
2.1	Planung, Diagnose und Konfiguration	7
2.2	Sichtweisen der Planungswelt	8
2.3	Chronologischer Rückblick	9
2.3.1	Erste Ansätze	9
2.3.2	STRIPS	9
2.3.3	NOAH	12
2.3.4	MOLGEN	14
2.3.5	Opportunistisches Planen	17
2.3.6	SIPE	17
2.3.7	Planen und Zeit	18
2.3.8	Distributed Planning	18
2.3.9	TWEAK	19
2.3.10	Lernende Systeme	19
3	Planungsverfahren in der Arbeitsplanung	21
3.1	Aufgaben der Arbeitsplanung	21
3.2	Verschiedene Ansätze	23
3.3	Der Varianten-Ansatz	23
3.4	Der generative Ansatz	23
3.4.1	Werkstückbeschreibung	24
3.4.2	Entscheidungslogik und Algorithmen	25
3.4.3	Herstellungswissen	32
3.5	Überblick: CAPP-Systeme	32
3.5.1	Die ersten Systeme	34
3.5.2	Von APPAS bis QTC	34

3.5.3	Weitere KI-basierte Systeme	36
3.5.4	Fazit	38
4	Diskussion	40
4.1	Beurteilungskriterien von Planungsverfahren	40
4.2	Merkmale des Planungsbereiches AP	41
4.3	Vernachlässigbare Eigenschaften der AP	43
4.4	Eignung der KI-Verfahren für die AP	43
4.4.1	Planen mittels Inferenz	43
4.4.2	lineares Planen (nach STRIPS)	44
4.4.3	nichtlineares Planen (nach NOAH, NONLIN)	44
4.4.4	hierarchisches Planen	45
4.4.5	Planen mit constraints	46
4.4.6	Meta-Planen, opportunistisches Planen	47
4.4.7	Weitere Techniken	47
4.5	Skelettplantchnik (PIM)	47
A	Skippy: Skelettplanen in der AP	52
A.1	Überblick	52
A.2	Der Aufbau eines Skelettplanes	54
A.2.1	<i>special-tree</i>	54
A.2.2	<i>op-mode</i>	56
A.2.3	<i>constraints</i>	56
A.2.4	<i>actions</i>	56
A.3	Skippys Kontrollstruktur	61
A.3.1	Skelettplanauswahl	61
A.3.2	Skelettplanverfeinerung	62
A.4	Skippys Moduln	62
B	Tabellarische Übersicht: CAPP-Systeme	65

Kapitel 1

Einleitung

Der erste und wichtigste Schritt bei der Fertigung eines Produktes ist die Erstellung eines Arbeitsplanes [49]. Der Arbeitsplan beeinflusst die Qualität des Produktes und die Kosten für die Herstellung. Wurde der Arbeitsplan früher manuell erstellt, so werden heute zunehmend sogenannte "CAPP-Systeme"¹ eingesetzt. Sie bilden die Schnittstelle zwischen dem Entwurf (CAD²) und der Fertigung (CAM³) in einem CIM-System⁴.

Ziel dieser Diplomarbeit ist es, zu analysieren, welche Planungsverfahren aus dem Gebiet der KI⁵ für den Einsatz in CAPP-Systemen in Frage kommen. Da das angewandte Planungsverfahren die Komplexität und die Effizienz eines CAPP-Systemes beeinflusst, spielt die Auswahl einer geeigneten Planungsmethode eine wichtige Rolle bei der Konzeption des CAPP-Systemes. Außerdem ist die Diplomarbeit als Orientierungshilfe für denjenigen Leser gedacht, der sich in das komplexe Gebiet der KI-basierten Arbeitsplanung einarbeiten möchte.

Die Erkenntnisse aus der Analyse der Planungsverfahren wurden im Rahmen des ARC-TEC-Projektes⁶ am DFKI⁷ in Kaiserslautern verwertet und mündeten in der Implementierung der Planungskomponente "Skippy" des CAPP-Systemes "PIM"⁸. Im Projekt ARC-TEC werden innerhalb der anwendungsorientierten Grundlagenforschung KI-Methoden entwickelt oder weiterentwickelt und mit anderen Techniken der Informatik verzahnt. Ergebnis des ARC-TEC-Projektes soll eine domänenspezifische Shell für ein Teilgebiet der Fertigungstechnik und eine oder mehrere exemplarische Anwendungen dieser Shell sein, z.B. ein Expertensystem für die Arbeitsplanung von Drehteilen.

Gliederung der Arbeit

Die Diplomarbeit ist in vier Kapitel unterteilt. Nach der Einleitung werden im nächsten Kapitel die bekanntesten Planungsverfahren der KI aufgeführt und teilweise anhand von Beispielen aus der Arbeitsplanung erklärt. Im dritten Kapitel wird das Gebiet der Ar-

¹Computer Aided Process Planning

²Computer Aided Design

³Computer Aided Manufacturing

⁴Computer Integrated Manufacturing System

⁵Künstliche Intelligenz

⁶Akquisition, Repräsentation und Compilation von TECnischem wissen

⁷Deutsches Forschungszentrum für Künstliche Intelligenz

⁸Planen Im Maschinenbau

beitsplanung näher erläutert. Weiterhin werden verschiedene CAPP-Systeme im Hinblick auf die verwendeten Planungsverfahren untersucht. Im vierten Kapitel schließlich wird die Frage diskutiert, welche Planungsverfahren der KI für die Arbeitsplanung am besten geeignet sind. Im Anhang dieser Diplomarbeit wird die oben erwähnte Planungskomponente Skippy des CAPP-Systemes PIM, die im Rahmen dieser Arbeit implementiert wurde, erläutert. Schließlich findet sich dort eine Tabelle mit einer Kurzbeschreibung der untersuchten CAPP-Systeme.

Kapitel 2

Planungsverfahren in der Künstlichen Intelligenz

In diesem Kapitel werden zunächst einige Begriffe aus der KI diskutiert. Danach werden die wichtigsten Planungssysteme der KI und die damit verbundenen Methoden und Begriffe in der Reihenfolge ihrer Entstehung erläutert.

2.1 Planung, Diagnose und Konfiguration

Da das Planen zu den Fähigkeiten des Menschen zählt, die Intelligenz erfordern, wurde es schon früh ein Thema für den Forschungsbereich der Künstlichen Intelligenz. In der Künstlichen Intelligenz wird der Begriff *"Planen"* häufig in einen Gegensatz zum Begriff *"Diagnose"* gestellt. Während bei einer Diagnose aus einer mehr oder weniger genau beschriebenen Menge (z.B. mögliche Krankheiten oder Fehlerarten) ein Element ausgewählt wird, wird bei einem Planverfahren mit einer Menge von Elementarbausteinen nach bestimmten Regeln ein im Prinzip beliebig komplexes Objekt erstellt. In der Realität gibt es jedoch keine reinen Planungs- bzw. Diagnoseprobleme. So kann die Diagnose ein sehr komplexer Prozeß sein, der geplant werden will. Umgekehrt kann man ausgehend von der Analyse (Diagnose) der aktuellen Plansituation den nächsten Planschritt gewinnen [31].

Planen ist also ein synthetischer Prozeß, Diagnose hingegen von analytischer Natur. Ein weiterer konstruktiver Prozeß ist das *"Konfigurieren"*. Beim Konfigurieren wird ein Zuordnungsproblem gelöst, d.h., die Basiselemente stehen fest und müssen richtig verknüpft werden. Dagegen wird beim Planen ein Transformationsproblem gelöst: ein Objekt wird durch die Verkettung von Operatoren von einem Anfangszustand in einen Zielzustand transformiert. Im Bereich der Arbeitsplanung kann man beide Probleme wiederfinden: das Finden einer Sequenz von Bearbeitungsschritten, die den Rohling in den Zielzustand überführt, ist ein Transformationsproblem, während die Bestimmung eines (optimalen) Maschinenbelegungsplans unter Berücksichtigung der gegebenen Maschinen und der auszuführenden Bearbeitungsschritte ein Zuordnungsproblem ist [30].

Die Begriffe *"Planen"* und *"Konfigurieren"* werden jedoch häufig nicht streng unterschieden.

2.2 Sichtweisen der Planungswelt

Die Sichtweise der (Planungs-)Welt hat einen nicht unerheblichen Einfluß auf das jeweilige Planungsverfahren. Die ersten Planungsverfahren teilten die Welt in zwei Klassen ein: Operatoren und Situationen. Unter Planen versteht man dann das Suchen einer Folge von Operatoren, die das Modell des aktuellen Weltzustandes (aktuelle Situation) in ein Zielmodell mit bestimmten Eigenschaften überführt. Diese Sichtweise geht auf Arbeiten

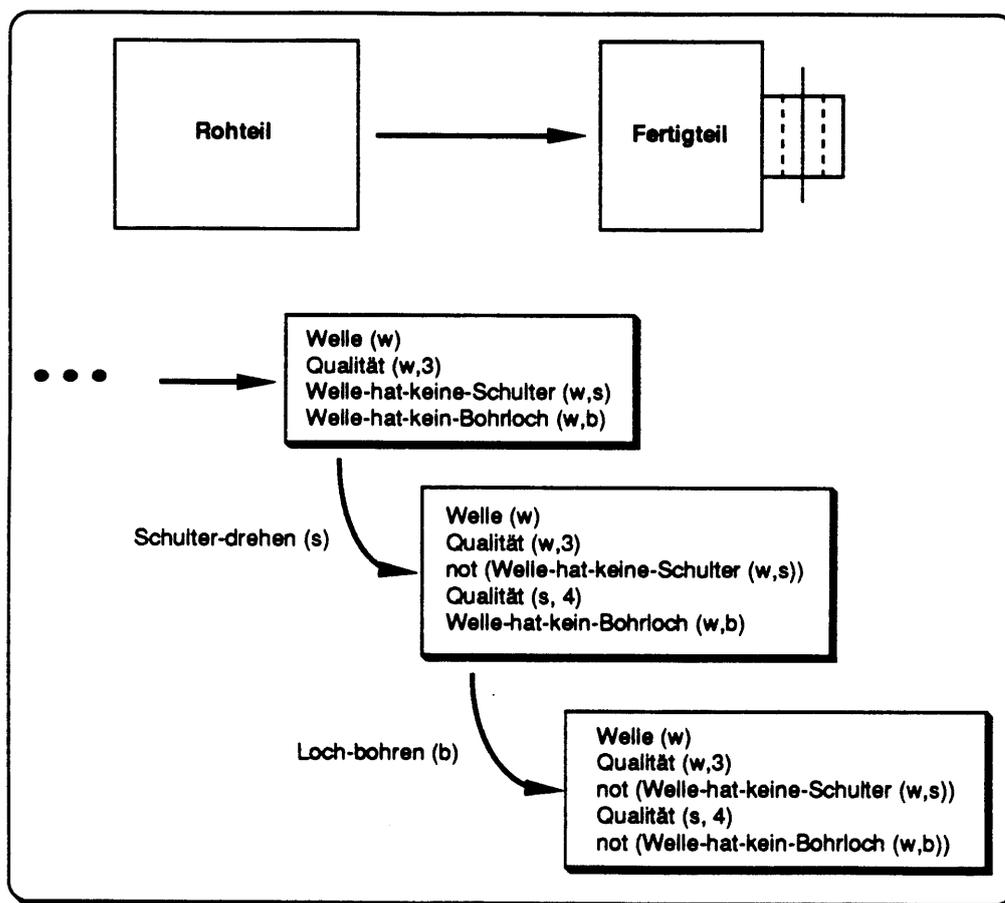


Abbildung 2.1: Sichtweise des "state-based approach"

von Newell und Simon Anfang der sechziger Jahre zurück. Hertzberg [17] bezeichnet diese Sicht verwirrenderweise als "Situationskalkül", womit er wohl auf eine der ersten Implementierungen dieser Sichtweise, den "situation calculus" von McCarthy und Hayes [24] anspielt. Ein passenderer Begriff für diese Sichtweise ist nach [16] "state space search"¹ oder auch

"state based approach" (vgl. Abbildung 2.1).

Ab 1975 etwa hat sich jedoch eine andere Sichtweise durchgesetzt: die Punkte im Zustandsraum stellen jetzt partiell entwickelte Pläne dar ("partial plan search space", [41]). Ausgehend von einem initialen (Skelett-) Plan wird eine Sequenz von Plantransformationen gesucht, die einen vollständig detaillierten Plan generieren. Die erste Implementierung

¹Zustandsraumsuche

dieser Sichtweise war E. Sacerdotis Planungssystem "NOAH" (vgl. Abschnitt 2.3.3, Seite 13).

2.3 Chronologischer Rückblick

Abbildung 2.2 zeigt eine Übersicht der wichtigsten Entwicklungen und Systeme im Planungsbereich der KI (vgl. [16]). Techniken, die speziell für den CAPP-Sektor entwickelt wurden, werden erst im nächsten Kapitel behandelt.

2.3.1 Erste Ansätze

Zu den ältesten Planungsverfahren zählt die "heuristische Suche" (vgl. A*-Algorithmus [12]). Eine ähnliche Klasse von Algorithmen ist unter dem Namen "branch and bound" im Gebiet der linearen Planungsrechnung² bekannt geworden [3]. Um die Zahl der Zwischenzustände, die während der Suche betrachtet werden, zu minimieren, führten Newell und Simon 1963 für ihr System GPS³ die "means-ends Analyse" ein [28]. Bei der means-ends Analyse werden nur die Aktionen betrachtet, die zur Erfüllung von ausstehenden Zielen in Frage kommen. Im Unterschied zur einfachen Rückwärtssuche wird zur Auswahl einer Aktion die Heuristik "wähle die Aktion, welche den Abstand (Unterschied) zur Zielsituation am meisten reduziert" verwendet. Diese Technik war die Grundlage für viele der suchraumbasierten Planer (z.B. STRIPS [6]).

Eine andere früh untersuchte Möglichkeit ist das "Planen mittels Inferenz", auch "deduktives Planen" genannt [19]. 1969 stellten McCarthy und Hayes [24] den bereits in Abschnitt 2.2 erwähnten Situationskalkül vor, in dem Situationen und Operatoren mittels prädikatenlogischer Formeln 1.Stufe beschrieben werden. Planen ist hier das Beweisen, daß aus der Problemsituation durch Anwenden der vorhandenen Operatoren eine Situation entsteht, die die Zielbedingungen erfüllt. Als Beweisverfahren kann z.B. Resolution verwendet werden, wobei allerdings die dabei durchgeführten Substitutionen aufzusammeln sind. Um auszudrücken, daß ein Merkmal in einer Situation gilt, führten McCarthy und Hayes den Begriff des "fluent" ein. Ein fluent ist eine Funktion auf Zuständen der Welt. Der Wert eines fluents für einen gegebenen Zustand ist der Wert eines Merkmals in diesem Zustand.

Ein Nachteil des Situationskalküls ist vor allem, daß für jeden Operator nicht nur das angegeben werden muß, was er verändert, sondern auch die Merkmale, die er nicht verändert; es ist also der konstante "Rahmen" zu spezifizieren, vor dem die Operatoren Änderungen bewirken. McCarthy und Hayes nannten dieses Problem das "frame problem" (Rahmenproblem).

2.3.2 STRIPS

Eine Lösung des frame problems erhält man, wenn man die Annahme macht, daß alle Formeln, die vor Anwendung eines Operators gültig waren und nicht explizit durch den

²Operation Research

³General Problem Solver

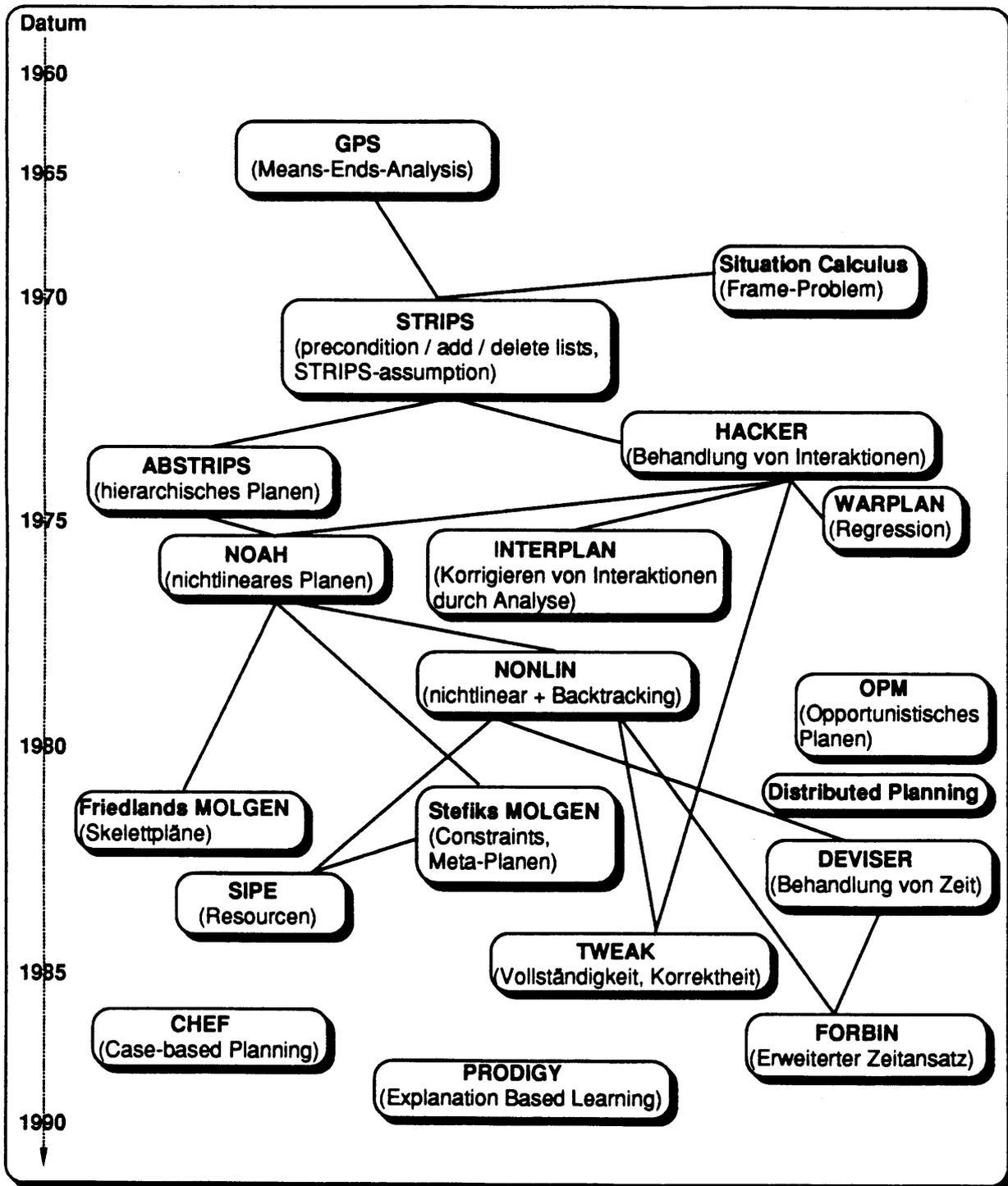


Abbildung 2.2: Chronologie der bekanntesten Planungssysteme

Operator ungültig gemacht werden, weiter gültig bleiben. Diese Bedingung ist bekannt unter dem Namen "*STRIPS-Assumption*", da sie in dieser Form zuerst 1971 in dem System *STRIPS*⁴ [6] verwendet wurde.

Bemerkenswert ist auch die Repräsentation der Operatoren in STRIPS. Ein "*STRIPS Operator*" (vgl. Abbildung 2.3) gliedert sich in drei Teile:

- die Vorbedingungsformel,
- einer *add-list* und
- einer *delete-list*.

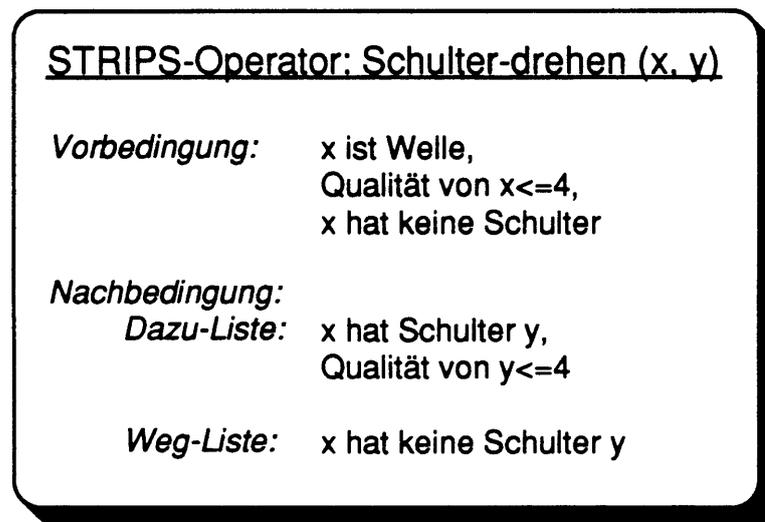


Abbildung 2.3: Typischer STRIPS-Operator

Die Vorbedingungsformel gibt an, wann der Operator angewandt werden kann. Beim Anwenden des Operators werden zunächst alle Formeln gelöscht, die in der *delete-list* stehen. Danach werden alle Formeln, die in der *add-list* stehen, zum neuen Zustand hinzugenommen. Dieses Schema zum Bestimmen des nachfolgenden Zustandes wird in [10] auch die "*STRIPS Rule*" genannt. Obwohl neuere Planungssysteme andere Ansätze verfolgen, wird die Terminologie der STRIPS-Operatoren noch häufig verwendet.

Ein weiterer Begriff, der durch STRIPS geprägt wurde, ist die "*Linearity Assumption*": In STRIPS geht man davon aus, daß es im allgemeinen nicht möglich ist, eine Konjunktion von Teilzielen durch einen einzigen Operator zu erfüllen. Deshalb versucht STRIPS eine Reihenfolge zu finden, in der es die Ziele nacheinander erfüllen kann, so daß sie am Ende alle noch gelten. Dieser Ansatz birgt die Gefahr, daß durch die Interaktion von Teilproblemen nicht optimale Pläne erzeugt werden können.

z.B. die "Sussman Anomalie". Sie ist ein Beispiel für eine "negative Teilzielinteraktion". Hertzberg bezeichnet in [18] damit den Fall, daß die Bearbeitung eines Zieles Teile der Arbeit, die man geleistet hat, um ein anderes Ziel zu erfüllen, zerstört. Ebenso ist eine "positive Teilzielinteraktion" möglich, wenn die Arbeit für ein Ziel die Arbeit für ein anderes Ziel "nebenbei" mit erfüllt. Außerdem kann es vorkommen, daß ein Unterproblem in mehreren Teilproblemen gleichzeitig vorkommt, so daß es genügen würde, das Unterproblem nur einmal zu lösen. Diese Art von Interaktion wird "hilfreiche Interaktion"⁵ genannt. Das Problem der Konflikterkennung und Auflösung wird von Hertzberg noch detaillierter beschrieben. So teilt er Konflikte in "elementare Konflikte" ("Linearkonflikt", "Gabelkonflikt", "Parallelkonflikt") und "zusammengesetzte Konflikte" (z.B. "Überkreuzkonflikt"⁶).

Andere Ansätze zur Behandlung von Interaktionen gibt es von Tate (*INTERPLAN*, [39]), Warren (*WARPLAN*, [44]) und von Waldinger (Regression von Teilzielen, [43]).

Mit dem System *ABSTRIPS*⁷ [33] erweiterte Earl D. Sacerdoti 1974 STRIPS um die Möglichkeit hierarchisch zu planen, in dem er verschiedene Wichtigkeitsstufen für die Vorbedingungen eines STRIPS-Operators einführt. Pläne werden mehrstufig erzeugt. Zunächst werden nur die wichtigsten Vorbedingungen der Operatoren erfüllt und danach stufenweise die weniger wichtigen Vorbedingungen. Der Effekt ist eine enorme Reduzierung des Suchaufwandes durch die frühe Eliminierung von Suchpfaden. Hertzberg bezeichnet in [18] diese Form der Abstraktion als "Situationsabstraktion".

2.3.3 NOAH

Das System *NOAH*⁸, das Sacerdoti [34] 1975 vorstellte, beeinflusste die weitere Entwicklung von Planungsverfahren nachhaltig. Sacerdoti war der erste, der von der Vorstellung eines

⁵ Planes als lineare Folge von Aktionen abrückte. In NOAH wird ein Plan bzgl. der Zeit als partiell geordnete Folge von Aktionen repräsentiert. Die Grundstrategie von solchen sogenannten "nichtlinearen" Planungssystemen ist es, bei mehreren Teilzielen diese unabhängig voneinander zu betrachten und — soweit keine Interaktionen auftreten — keine Einschränkung in der Reihenfolge der daraus resultierenden Teilpläne zu machen. Dieses Vorgehen wird häufig auch als "least commitment Strategie" bezeichnet. Es hat den Vorteil, daß möglicherweise die Entscheidung über die Reihenfolge der Aktionen bis zu dem Zeit-

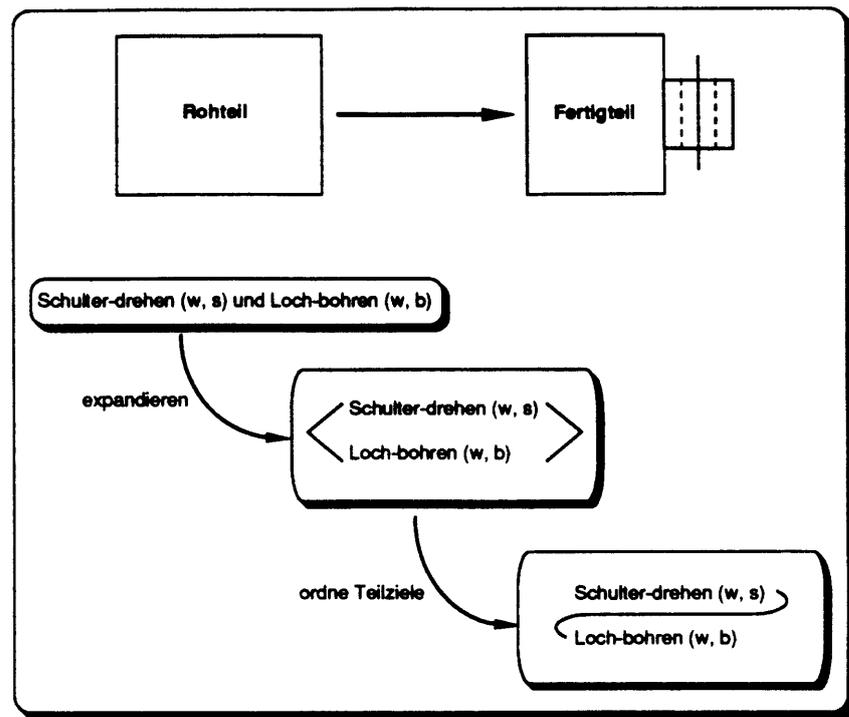


Abbildung 2.4: Procedural Net von NOAH: Stufen der Plandetaillierung

In NOAH dürfen nur ganze parallele Zweige eines Planes in beliebiger Reihenfolge ausgeführt werden. Andere mögliche Interpretationen der Nichtlinearität von Plänen sind nach Hertzberg [18]:

1. Bei parallelen Planzweigen dürfen die entsprechenden Operatoren in jeder vollständigen Ordnung ausgeführt werden, die mit der partiellen, nichtlinearen Ordnung verträglich ist.
2. Untereinander nicht geordnete Operatoren dürfen nebenläufig ausgeführt werden.

Ein Plan wird in NOAH durch ein "procedural net" dargestellt. Das procedural net ist eine Graphenstruktur, deren Knoten hierarchisch angeordnet sind. Jeder Knoten repräsentiert eine Aktion von unterschiedlichem Detaillierungsgrad und enthält Informationen über den Effekt der Aktion sowie über die Expandierung zu einem größeren Detaillierungsgrad. Die Knoten einer Hierarchieebene sind partiell bzgl. der Zeit geordnet. Die Beziehung zwischen den Aktionen wird also deklarativ, d.h. explizit und nicht implizit durch Zustände und darin gültigen Prädikaten dargestellt. Das bietet die Möglichkeit, zwei Aktionen zu ordnen, ohne daß man die Zwischenzustände kennt. Deshalb können komplexe kausale und temporale Zusammenhänge zwischen den Aktionen einfacher beschrieben werden als beim state-based Ansatz.

Ein weiterer Unterschied zu der ersten Version von STRIPS besteht darin, daß NOAH hierarchisch plant. Beginnend mit einem initialen Skelettplan wird Schritt für Schritt durch das Expandieren von Knoten ein detaillierter Plan erzeugt. Jeder so erzeugte Teilplan pro

Knoten ist für sich allein genommen korrekt. Das Planungssystem muß jedoch sicherstellen, daß Interaktionen zwischen den Teilplänen gelöst werden. In NOAH werden nach jeder Plandetaillierung sogenannte "critics" aufgerufen, die den neuen Plan optimieren. So gibt es z.B. einen "Resolve Conflicts Critic", der Interaktionen durch eine Linearisierung der beteiligten Planzweige zu lösen versucht (vgl. Abbildung 2.4). Ein anderer critic löscht redundante Planzweige, wieder andere berücksichtigen anwendungsspezifisches Wissen.

Ein Nachteil von NOAH ist, daß es weder beweisbar "korrekt" noch "vollständig" ist (vgl. [22]). Chapman bezeichnet in [4] ein Planungssystem als korrekt, wenn alle Pläne, die es erzeugt, korrekt sind. Ein Plan wird als korrekt bezeichnet, wenn der den Startzustand der Welt in den Zielzustand überführt. Ein Planungssystem heißt vollständig, wenn es eine korrekte Lösung findet, wann immer eine existiert.

Während der Generierung eines Planes gibt es i.a. drei verschiedene Arten von Wahlmöglichkeiten:

- Bestimmen der Reihenfolge von Operatoren
- Instanzieren von Variablen
- Wahl von Operatoren

Obwohl ein least commitment Planungssystem solange wie möglich eine Entscheidung hinausögern wird, gibt es jedoch häufig Fälle, in denen getroffene Entscheidungen revidiert werden müssen. Dann muß das System ein Backtracking-Feature besitzen, damit eine andere Wahlmöglichkeit ausprobiert werden kann und die Vollständigkeit nicht verloren geht.

1977 stellte Tate [40] mit dem System *NONLIN* eine Erweiterung von NOAH vor, welche über einen Backtracking-Mechanismus verfügt.

2.3.4 MOLGEN

Im Rahmen des MOLGEN-Projektes der Stanford University entstanden um 1980 zwei Planungssysteme, die beide unter dem Namen *MOLGEN*⁹ bekannt geworden sind. Im Rahmen dieses Projektes werden mögliche Anwendungen von Methoden der künstlichen Intelligenz im Bereich der Molekular-Biologie untersucht. Insbesondere wurde versucht, das Design eines Experimentes zu unterstützen.

Stefiks MOLGEN [37, 36], steht in der Tradition von NOAH. Wie NOAH plant es hierarchisch, verwendet den partial-plan search Ansatz und die least commitment Strategie. Es benutzt die least commitment Strategie jedoch nicht nur zum Linearisieren von Operatoren, sondern auch bei der Auswahl eines Objektes. Dies geschieht durch die Assoziation von Constraints mit Planschritten.

durch Constraints darstellen und diese dazu benutzen, die Lösung der Teilprobleme zu koordinieren.

Ein wichtiger Unterschied zu älteren Constraint-Systemen ist die Möglichkeit, während des Planungsprozesses dynamisch neue Constraints formulieren zu können, die auf einem abstrakteren Planungsniveau noch keine Rolle spielten.

Stefik unterscheidet drei Operationen auf Constraints:

1. "*constraint formulation*", d.h., das dynamische Hinzufügen neuer Constraints.
2. "*constraint propagation*". d.h., das Erzeugen von neuen Constraints aus alten Con-

3. "*constraint satisfaction*", d.h., das Finden von Werten für Variable, so daß eine Menge von Constraints für die Variable erfüllt wird.

Als weitere neue Planungsmethode führte Stefik das "*Meta-Planen*" ein. Die Grundidee des Meta-Planens ist es, die möglichen Aktionen des Planers, wie z.B. "Expandieren eines Knotens" oder "Instanzieren einer Variablen", genau wie die Aktionen des Planungsbereiches zu formalisieren und zu verarbeiten. Hayes fordert in [13] sogar, daß die Beschreibungssprache für diese "Meta-Operatoren" mit der der Anwendungsdomäne übereinstimmen soll. Diese Forderung wird jedoch von Wilkins [45] kritisiert, da die Ausdruckskraft der häufig modellbasierten Repräsentation der Welt für echtes Meta-Planen nicht ausreicht.

Meta-Planen ist sozusagen das Planen des Planens. Die Kontrollstruktur wird deklarativ spezifiziert, so daß das Programmverhalten durchschaubar und besser nachvollziehbar wird. Meta-Planen scheint sinnvoll zu sein, wenn viele verschiedene Strategien beim Planen miteinander kombiniert werden sollen. Dabei ist darauf zu achten, daß die Zeit, die

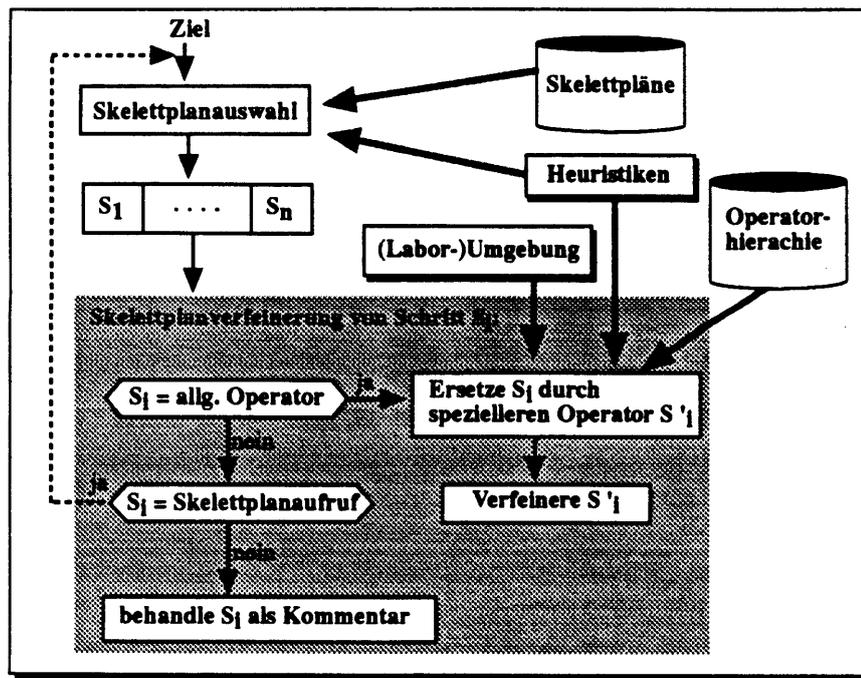


Abbildung 2.5: Friedlands MOLGEN System

über die Struktur des Planungsobjektes. So spricht Friedland von dem Bruder eines Planungsobjektes, wenn dieser zur gleichen Subklasse gehört.

2. Skelettplanverfeinerung

Ausgehend von dem ausgewählten Skelettplan wird eine Instanzierung der Schritte des Planes durch atomare Aktionen gesucht. Ein Planschritt kann eine der folgenden Bedeutungen haben:

- Der Planschritt steht für eine Klasse von Operatoren oder für einen speziellen Operator, oder er definiert ein Ziel, das in der Liste der Ziele vorkommt, die eine Klasse von Operatoren oder ein spezieller Operator erfüllt. Liegt einer dieser Fälle vor, dann wird der Planschritt durch eine Instanz aus einer in Frage kommenden Klasse ersetzt. Handelt es sich dabei um einen nicht atomaren Operator, dann wird der Schritt der Skelettplanverfeinerung wiederholt. Insofern liegt also auch dem Skelettplanen eine Operatorhierarchie zugrunde.
- Der Planschritt definiert das Ziel für einen neuen Skelettplan. Dies führt zu einem rekursiven Aufruf des Skelettplaners.
- Der Planschritt ist unbekannt. In diesem Fall wird er als Kommentar behandelt, und es bleibt dem Benutzer überlassen, wie er ihn instanziiert.

Die Bedeutung eines Planschrittes wird durch die Wissensbasis impliziert und vor dem Benutzer verborgen. Dies ermöglicht z.B. die einfache Änderung eines Planschrittes von einem Kommentar zu einem Operator wenn die Wissensbasis und die Sammlung der Skelettpläne sich vergrößert haben.

Anmerkungen:

1. Friedlands Ansatz geht von einer hierarchisch aufgebauten Miniwelt der Molekular-

Genetik, aus.

2. Das Skelettplanckonzept wurde ein zweites Mal mit dem System "SPEX"¹⁰ implementiert. Dieses System verwendet den Meta-Planungsansatz von Stefiks MOLGEN als Kontrollstruktur für das Skelettplanen.

2.3.5 Opportunistisches Planen

1979 wurde von B. und F. Hayes-Roth [15] das System *OPM* entwickelt, das das Planungsverhalten von Menschen beim Lösen einer Planungsaufgabe aus dem Alltag simulieren sollte. Bei der Analyse von Protokollen von Versuchspersonen stellten die Autoren fest, daß die Versuchspersonen natürlich nicht nach einer der klassischen KI-Planungsmethoden vorgehen, sondern wesentlich flexibler. Insbesondere wechselten die Versuchspersonen häufig die Ebene der Planabstraktion. Entscheidungen der logisch untersten Planungsebene wurden wechselweise zu strategischen Überlegungen berücksichtigt. Dieses sprunghafte Verhalten rührt anscheinend daher, daß gerade verfügbar gewordene Informationen, wie z.B. die Ausführbarkeit eines Problemlösungsschrittes, ausgenutzt werden. Dieses Vorgehen, das Mäntelchen der Problemlösungsentscheidungen unter anderem nach dem Wind der gerade aktuellen Information zu hängen, hat dem Kind seinen Namen gegeben: "opportunistisches" Planen [18].

Zur Realisierung eines flexiblen Planungssystems übernahmen B. und F. Hayes-Roth das "Blackboard-Konzept" des HEARSAY-II Systems [5]. Das Blackboard ist eine Datenstruktur, in der verschiedene Spezialisten Informationen über die Ergebnisse zu ihren Problemen ablegen und Informationen von anderen Spezialisten lesen können. Möchte man dieses Konzept auf einer sequentiellen Maschine implementieren, so müssen die Reihenfolge der Aufrufe der Spezialisten und der Zugriff auf das Blackboard verwaltet werden. Barbara Hayes-Roth nennt dieses Problem in [14] das "control problem". Seine Lösung bestimmt, welche Probleme gelöst, welches Wissen verwendet, welche Strategien angewandt werden und natürlich auch die Effizienz des Systems.

2.3.6 SIPE

Das System *SIPE*¹¹ von Wilkins [45] ist ein nichtlinearer, hierarchischer und Constraints verwendender Planer. Verglichen mit NOAH besitzt SIPE eine höhere Ausdruckskraft. So können z.B. deduktive Operatoren angegeben werden, die aus den momentan geltenden Fakten der Welt neue Fakten ableiten können. So muß der Benutzer sich weniger um die Vollständigkeit und Konsistenz seiner Eingaben kümmern und die Zuverlässigkeit der Eingaben steigt. Außerdem kann das Anwendungswissen in hybriden Formalismen repräsentiert werden. Wissen von statischer Natur kann durch Framehierarchien und dynamisches Wissen durch Formeln in Prädikatenlogik 1.Stufe dargestellt werden. Durch diese

¹⁰Skeletal Planner of EXperiments

¹¹System for Interactive Planing and Execution monitoring

Möglichkeiten wird jedoch die Repräsentationskapazität des Planers nicht gesteigert, sondern nur die Klarheit und Bequemlichkeit der Darstellungsweise von Problemen.

SIPE war das erste Planungssystem, in dem man mehrfach benötigte Objekte als "resource" definieren konnte. Eine resource ist für SIPE ein wiederverwendbares Objekt, das nicht von mehreren parallelen Operatoren gleichzeitig in Anspruch genommen werden kann.

Durch seine Benutzerschnittstelle bietet SIPE die Möglichkeit der interaktiven Steuerung der Planerzeugung und der Planausführung. Dieses Feature erlaubt ein elegantes "Replanning", wenn man durch die Ausführung des Planes in einen unerwarteten Zustand der Welt gelangt.

2.3.7 Planen und Zeit

Das Planungssystem *DEVISER* von Vere [42] ist eine Erweiterung von *NONLIN*. Es war das erste Planungssystem, in dem die Zeitdauer und die Startzeit von Aktionen spezifiziert werden konnte. Die Startzeit wird durch ein Zeitintervall ("time window") angegeben. Bei der Plangenerierung können die Zeitintervalle verkleinert werden, wenn Zeit-Constraints dies erfordern. Auch für die Planziele können Zeitbeschränkungen für ihre Erreichbarkeit definiert werden.

Allen und Koomen [2, 1] formulierten eine abgeschlossene Menge von sieben Zeitrelationen zwischen Aktionen.

Das System *FORBIN* [25] kombiniert die Ansätze von Vere und von Allen und Koomen.

2.3.8 Distributed Planning

Für viele Anwendungsprobleme kann man nicht voraussetzen, daß der fertige Plan von genau einem Akteur ("agent") ausgeführt wird, und daß es genau einen auszuführenden Plan gibt. Hertzberg [18] nennt drei Gründe, wieso verteiltes Planen¹² verteilter Problemlösungen Sinn machen kann:

- Verteiltheit: Das Problem kann durch voneinander unabhängige Spezialisten gelöst werden.
- Redundanz: Mehrere Planer für ein Problem, um Ausfallsicherheit zu gewährleisten.
- Effizienz: Manchmal können mehrere Planer zusammen einen Plan schneller konstruieren, als ein Planer allein.

Zur Koordination von mehreren Agenten schlägt Hertzberg eine Blackboard-Architektur vor.

Distributed Planning erfordert ein Überdenken der Annahmen, die für eine Welt mit nur einem Agenten noch galten. Insbesondere kann nicht mehr jede Aktion durch eine Relation auf Situationen beschrieben werden, da die Wirkung der Ausführung der Aktion davon abhängen kann, was während der Ausführung passiert.

¹²engl.: Distributed Planning, Multi-Agent Planning

Der im Abschnitt 2.3.7 schon erwähnte Planer von Allen und Koomen ist durch seine Intervallogik in der Lage verteilt zu planen. Die Gültigkeit von Merkmalen wird durch Funktionen auf Intervallen und nicht auf Zuständen definiert.

2.3.9 TWEAK

TWEAK von Chapman [4] ist das erste nichtlineare Planungssystem, das eine solide theoretische Grundlage hat. Chapman beweist, daß *TWEAK* sowohl vollständig als auch korrekt plant (vgl. Abschnitt 2.3.3, Seite 14).

Wie *SIPE* verwendet *TWEAK* constraints zum Planen. Wenn *TWEAK* ein Problem bearbeitet, dann bezieht es sich immer auf einen unvollständigen Plan, der eine partielle Spezifikation eines Planes zur Lösung des Problemes ist. Durch Hinzufügen von constraints kann der unvollständige Plan auf mehrere Arten vervollständigt werden. Der Planer terminiert, wenn alle Vervollständigungen des unvollständigen Planes das gegebene Problem lösen.

Die Ausdruckskraft von *TWEAK* ist nach Chapman allerdings so beschränkt, daß *TWEAK* für die meisten Anwendungsbereiche nicht geeignet ist. So fehlt z.B. die Möglichkeit, den Wertebereich von Variablen auf eine endliche Menge einzuschränken¹³. Weiterhin bemängeln Kartam und Wilkins in [22], daß keine Operatorpräferenzen eingeführt werden können.

2.3.10 Lernende Systeme

Die Fähigkeit, aus den gewonnenen Erfahrungen zu lernen, fehlt bei den meisten Planungssystemen. Eine Ausnahme bildet hier *STRIPS* [7], das 1972 durch das Konzept der "*Macrops*" erweitert wurde. Wird ein Teilproblem erfolgreich gelöst, dann produziert *STRIPS* eine parametrisierte Version der Lösung, einen sogenannten *Macrop*¹⁴, in dem es Konstante durch Variable unter Berücksichtigung der Abhängigkeiten zwischen den Planschritten ersetzt. Eine Sequenz von Operatoren wird also durch einen einzigen Operator substituiert, der in seiner Wirkung und in seiner Vorbedingung mit der Wirkung und der Vorbedingung der Operatorfolge übereinstimmt. Die Vorteile dieses Vorgehens bestehen darin, daß

- die Anwendbarkeit eines Makro-Operators i.a. schneller zu prüfen ist, als die Anwendbarkeit der korrespondierenden Operatorsequenz und
- durch die Benutzung von Makro-Operatoren häufig ein Backtracking umgangen wird.

S. Minton kritisiert jedoch in [26], daß sogar in kleineren Anwendungsgebieten der *STRIPS*-Ansatz zu einer Explosion von Makro-Operatoren führen kann. Dann hat sich der Effizienzgewinn ins Gegenteil verkehrt. Deshalb schlägt Minton vor, das Verfahren des "*Explanation-Based Learning*" ("*EBL*") zum selektiven Generalisieren von Teilplänen zu benutzen.

¹³Dann würde die Berechnung von constraints nämlich NP-vollständig sein; außerdem würde der Ansatz zum Beweis der Korrektheit von *TWEAK* scheitern.

¹⁴steht für Macro-Operator

2 Planungsverfahren in der Künstlichen Intelligenz

Eine Implementierung dieser Idee ist *PRODIGY* [27]. *PRODIGY* verwendet eine Means-Ends Analyse kombiniert mit anwendungsspezifischen Kontrollregeln, die die Suche steuern.

Kapitel 3

Planungsverfahren in der Arbeitsplanung

In diesem Kapitel werden zunächst einige Begriffe aus der Arbeitsplanung geklärt. Danach

3 Planungsverfahren in der Arbeitsplanung

Ausgangs- teil- bestim- mung	Ident-Nr.: 8604-7743		Datum: 30.02.91		ARBEITSPLAN		
			Bearbeiter: Klobner				
	Auftrags-Nr.: 1174-42		Benennung: Zapfen		Stückzahl: 8	Zeichnungs-Nr.: WE 1-19-1897	
Werkstoff: St 50-2		Rohform und -abmessungen: Rundstahl Ø 70		Rohgewicht: 9,01	Fertiggewicht: 7,83		
AVG-Nr.	Arbeitsvorgangs-anweisung	Kosten-stelle	Masch-gruppe	Fert-hilfsmittel	Lohn-gruppe	Rüst-zeit	Stück-zeit
10	Sägen auf L = 300	312	388		08	0,20	0,4
20	NC-Drehen <small>Paper Nr.: 42.1</small>	329	344	LS1033	10	0,20	1,4

Er-
stellung
des
NC-

3.2 Verschiedene Ansätze

CAPP-Systeme lassen sich anhand der Planungsmethode in zwei Klassen unterteilen: "variante" und "generative" Systeme¹. Variante Arbeitsplanungssysteme basieren auf der Annahme, daß es Teile gibt, die ähnliche Arbeitspläne haben. Durch die Analyse dieser Arbeitspläne versucht man zu erkennen, welche Geometrie- und Technologieeigenschaften für die Ähnlichkeit der Arbeitspläne verantwortlich sind. Diese Analyse bildet dann die Grundlage für die Klassifizierung der Werkstücke. Der Computer wird dazu benutzt, ähnliche Pläne zu suchen und diese zu editieren. Bei der generativen Planung, die auch Neuplanung genannt wird, wird der Arbeitsplan ohne Bezugnahme auf bereits existierende Pläne erstellt.

Vergleicht man das Erstellen eines Hauses nach der Fertigbauweise mit dem Bau eines Hauses durch Steine und Mörtel, so bekommt man ein analoges Verhältnis. Die Fertigbauweise entspricht dabei dem Varianten-Ansatz. Sie ermöglicht das Herstellen von begrenzt vielen verschiedenen Werkstücken bzw. Häusertypen. Dagegen ist der Neuplanungs-Ansatz bzw. das Bauen durch einzelne Steine zwar flexibler, erfordert aber auch mehr Zeit und Geschick [94].

3.3 Der Varianten-Ansatz

Der "Varianten-Ansatz" war der erste Ansatz [46] bei der Automatisierung der Arbeitsplanung (vgl. Abbildung 3.3). Die meisten varianten Systeme teilen die herzustellenden Teile in Gruppen² ein und kodieren die Gruppen. Die Einteilung wird gewöhnlich nach der geometrischen Form oder der Herstellungstechnologie der Teile vorgenommen. Für jede Teilefamilie wird ein Standardarbeitsplan abgespeichert. Während der Herstellungsphase wird ein Teil zuerst kodiert und dann anhand des Codes einer Teilefamilie zugeordnet. Schließlich wird der zugehörige Standardarbeitsplan modifiziert. Die Änderungen müssen leider oft noch manuell von einem erfahrenen Arbeitsplaner durchgeführt werden.

Neben dem geringen Automatisierungsgrad ist ein weiterer Nachteil das begrenzte Spektrum der Teile, die geplant werden können. Außerdem hängt die Qualität eines Arbeitsplanes immer noch vom Erfahrungswissen des Arbeitsplaners ab. Der Computer hat nur eine unterstützende Funktion. Trotzdem wird der Varianten-Ansatz noch häufig verwendet, da er verglichen mit dem generativen Ansatz einfacher zu implementieren, zu durchschauen, zu lernen und zu benutzen ist.

Bekannte Klassifizierungs- und Kodierungssysteme sind "CODE" [93] und "MICLASS" [100].

3.4 Der generative Ansatz

Beim generativen Ansatz wird der Arbeitsplan für jedes Teil ausgehend von der Teilbeschreibung auf der Basis von speziellem Wissen über die Fertigungsumgebung und allgemeinem

¹Manche Autoren [55, 48] unterteilen noch feiner in Neu-, Varianten-, Wiederhol-, Ähnlichkeits- und Alternativplanung.

²deshalb auch "Gruppentechnologie" genannt

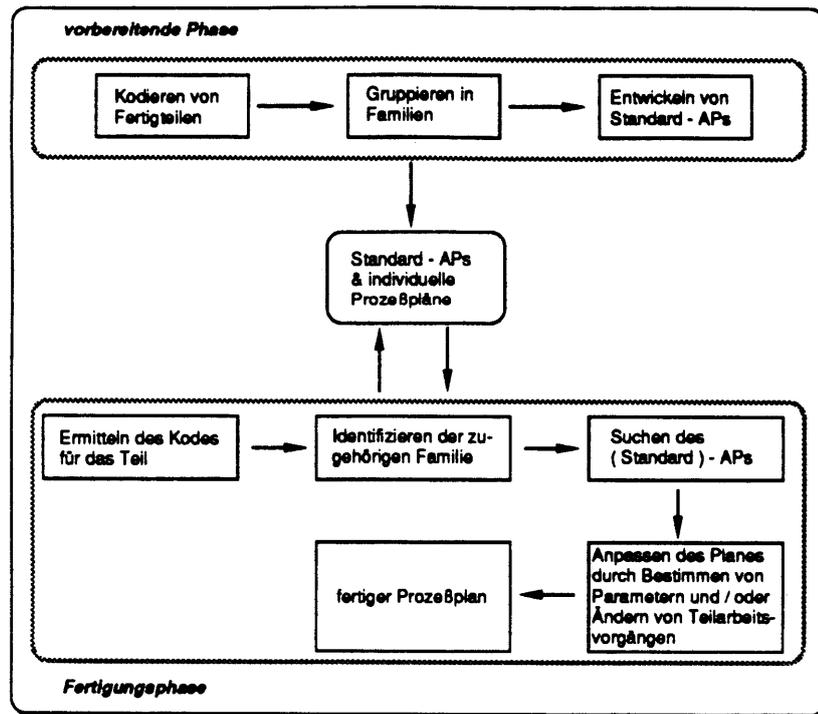


Abbildung 3.3: Schema des Varianten-Ansatzes nach [49]

Wissen über mögliche Herstellungsverfahren automatisch erstellt.

Nach T. C. Chang [49] besteht ein generatives Arbeitsplanungssystem aus drei Hauptkomponenten:

1. Die Werkstückbeschreibung
2. Entscheidungslogik und Algorithmen
3. Herstellungswissen

3.4.1 Werkstückbeschreibung

Nach T. C. Chang [49] und Cheung/Dowd [51] gibt es drei Möglichkeiten zur Werkstückbeschreibung:

1. Kodierungssysteme (z.B. durch Gruppentechnologie)
2. spezielle Beschreibungssprachen
3. CAD-Modelle

Die erste Generation von generativen AP³-Systemen benutzte den bereits von den varianten Systemen bekannten Ansatz der Gruppentechnologie zur Beschreibung des herzustellenden

³ArbeitsPlanung

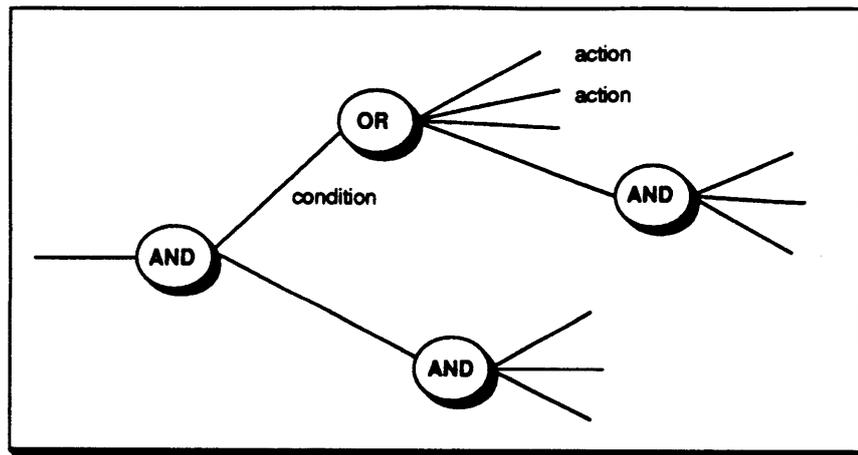


Abbildung 3.4: Beispiel für einen Entscheidungsbaum nach [49]

3.4.2.2 Entscheidungstabellen

Häufiger als Entscheidungsbäume werden Entscheidungstabellen (ET) (vgl. Abbildung 3.5) in AP-Systemen verwendet. Die Zeilen einer ET zerfallen in einen Bedingungsteil und in einen Aktionsteil. Somit wird durch eine Spalte eine Regel der Form

wenn <Bedingungsteil der ET> dann <Aktionsteil der ET>

ausgedrückt. Ist im Aktionsteil auch der Aufruf einer weiteren ET zulässig, so spricht man von einem "Entscheidungstabellen-Verbund" [55]. Dies ermöglicht eine Modularisierung der Entscheidungslogik. Ist zu jedem Zeitpunkt höchstens eine der vorhandenen Regeln anwendbar, spricht man von einer "eindeutigen Entscheidungstabelle"⁵, sonst von einer "mehrdeutigen Entscheidungstabelle"⁶. Ein großer Vorteil der ET-Technik ist die Einfachheit und Übersichtlichkeit der Darstellung. Problematisch dagegen ist vor allem die Garantie der Eindeutigkeit und Vollständigkeit der Bedingungen für eindeutige ET, sowie die Widerspruchsfreiheit von Aktionen und die Redundanzfreiheit der Regeln insgesamt. Insbesondere das Überprüfen der Widerspruchsfreiheit von mehrdeutigen ET hat eine Zeitkomplexität von $O(n!)^7$ [55].

3.4.2.3 KI-basierte Ansätze

T. C. Cheng [40] unterscheidet für den Einsatz von KI in CAPP Systemen zwei Ansätze:

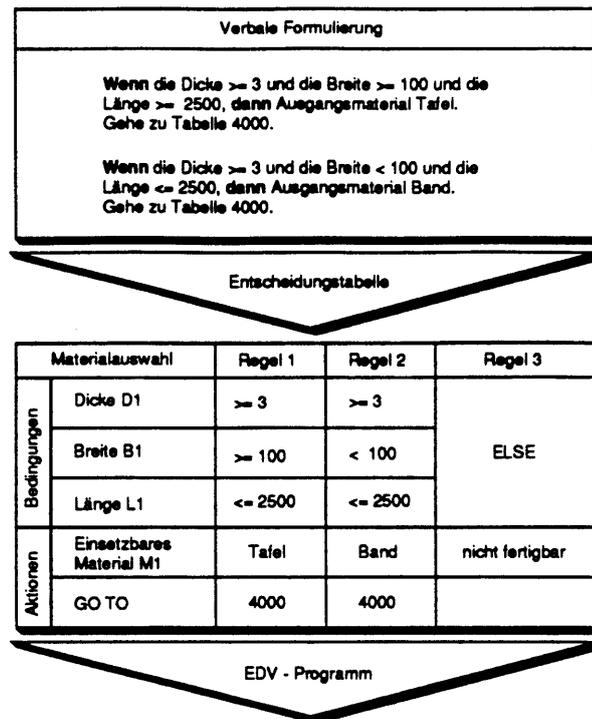


Abbildung 3.5: Beispiel für eine Entscheidungstabelle nach [48]

tation und die angewandte Kontrollstrategie entscheidend.

Wissensrepräsentation Zur Darstellung des Wissens in KI-basierten AP-Systemen werden demnach folgende Formalismen benutzt:

1. Prädikatenlogik
2. Regeln
3. Semantische Netze
4. Frames
5. Objektorientierte Programmierung

Diese Repräsentationsarten werden als dem Leser bekannt vorausgesetzt (vgl. [32]).

Arbeitsplanungsstrategie Zur Analyse der Arbeitsplanungsstrategie sollte man zwischen der Sicht des Arbeitsplaners und der Sicht eines KI-Experten unterscheiden. Die in Kapitel 2 vorgestellten Methoden verdeutlichen die Sichtweise des KI-Experten. Vom Standpunkt der Arbeitsplanung aber bietet sich nach [49] die folgende Untergliederung an:

1. Planungsstrategie in Abhängigkeit von der Planungsebene

(a) Lokaler Ansatz

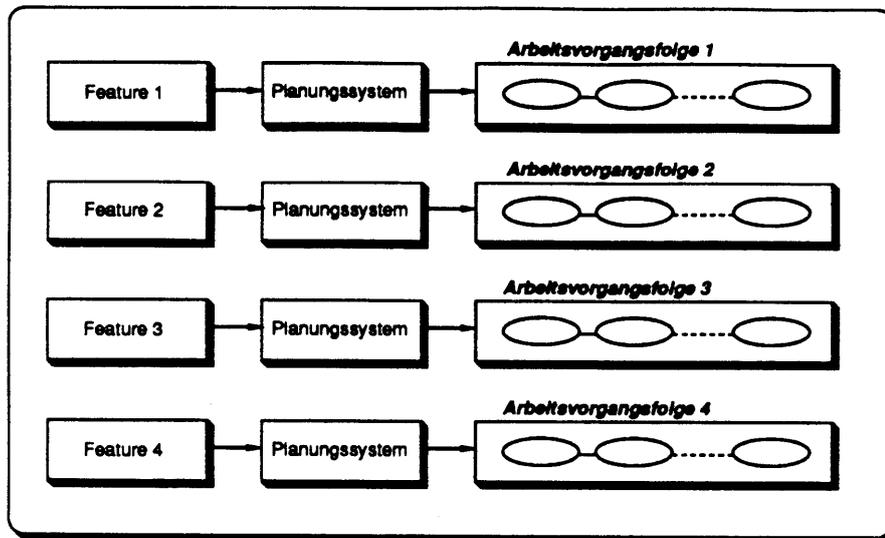


Abbildung 3.6: Planen aufgrund von lokaler Information nach [49]

Beim lokalen Ansatz wird zunächst jedes Feature des Werkstückes ohne Rücksicht auf das Verhältnis zu den anderen Features geplant (vgl. Abbildung 3.6). Der endgültige Arbeitsplan ist dann die Konkatenation aller Einzelpläne. Obwohl dieser Ansatz zu ineffizienten und falschen Ergebnissen führen kann, wird er wegen seiner Einfachheit von vielen existierenden Systemen bevorzugt [49]. Beispiele für Systeme nach diesem Ansatz sind u.a. "TIPPS" [50] und "TOM" [98].

(b) Globaler Ansatz

Der globale Ansatz berücksichtigt mögliche Interaktionen zwischen den Features bei der Fertigung. Häufig wird dazu eine Form von "geometrischem Schließen"⁸ verwendet. T. C. Chang nennt zwei Gründe für Interaktionen, die beim Planen zu beachten sind:

- i. Kollisionen zwischen dem nicht schneidenden Teil eines Schneidewerkzeugs und dem Werkstück
- ii. Technologische Gründe: z.B. kann ein Bohrer nicht an einer schrägen Fläche angesetzt werden.

Da das Nachprüfen von geometrischen Verhältnissen recht aufwendig ist, neigt dieser Ansatz zu ineffizienten Lösungen. Deshalb bietet sich der Einsatz von Heuristiken und anderen Methoden der künstlichen Intelligenz hier an. Als Beweis für die Realisierbarkeit dieses Konzeptes seien die Systeme "QTC" [49] und "Skippy"⁹ genannt.

⁸engl.: geometric reasoning

⁹siehe Anhang A

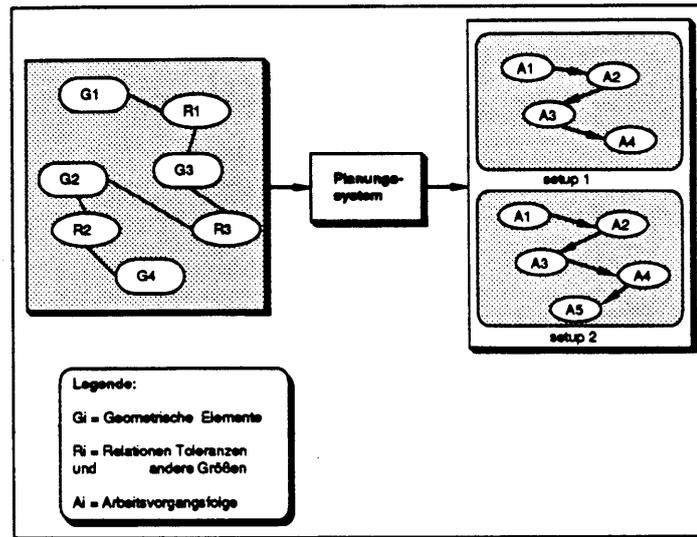


Abbildung 3.7: Planen aufgrund von globaler Information nach [49]

2. Planungsstrategie in Abhängigkeit von der Repräsentation der Eingabedaten

Auch die Darstellungsform des Werkstückes hat Einfluß auf das Planungsverfahren.

(a) Volumenorientierter Ansatz

Die meisten Systeme basieren auf einem volumenorientierten Ansatz, d.h. die geometrischen Möglichkeiten eines Fertigungsoperators werden durch volumenorientierte Features charakterisiert. So kann man z.B. den Arbeitsvorgang *Bohren* umschreiben als ein Prozeß, der Löcher produziert. Als Konsequenz wird also *Bohren* ausgesucht, wenn das Werkstück ein Loch enthält.

(b) Flächenorientierter Ansatz

Beim flächenorientierten Ansatz bildet die Fläche das geometrische Basiselement des Systems. Der Schwerpunkt dieses Konzeptes liegt auf der Darstellung von Flächeneigenschaften wie z.B. Oberflächentoleranzen.

3. Planungsstrategie in Abhängigkeit von der Beziehung zwischen Arbeitsvorgang und Feature

(a) Feature \Rightarrow Arbeitsvorgänge

Eine Möglichkeit für ein Arbeitsvorgangsmodell ist das Zusammenfassen von Features gleicher Herstellungsart in einer Teilefamilie. Für diese Familie wird dann ein parametrisierter Standardarbeitsplan, also eine Sequenz von Arbeitsvorgängen, definiert.

(b) Arbeitsvorgang \Rightarrow Features

Statt Features durch ihre Fertigungsweise darzustellen, kann auch umgekehrt ein Arbeitsvorgang durch die Menge der Features, die man damit herstellen kann, beschrieben werden. Abbildung 3.8 verdeutlicht diese Sichtweise.

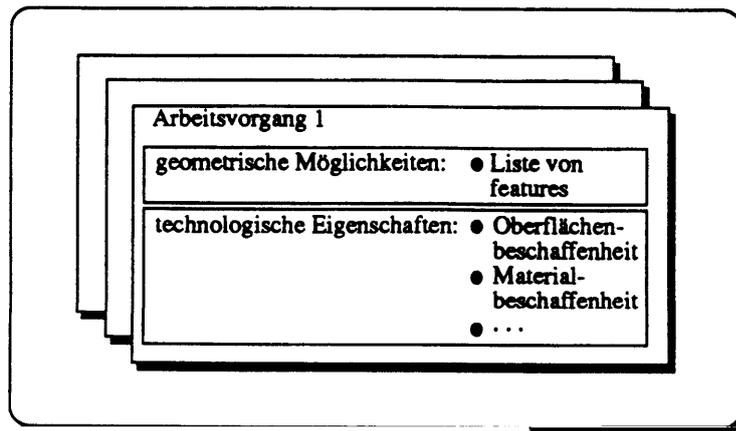


Abbildung 3.8: Modell eines Arbeitsvorgangs nach [49]

4. Planungsstrategie in Abhängigkeit von der Planungsrichtung

- (a) Die Planungsrichtung ist der Planungsprozeß so aufgebaut, daß die Suche beim Rohling beginnt und mit dem Erreichen der Werkstückbeschreibung endet, so bezeichnet man dies gewöhnlich als *"Forward Planning"*¹⁰ [49, 57], *"Top-Down"* Ansatz [46] oder auch als *"Forward Reasoning"*¹¹ [20]. Diese Richtung stimmt mit dem Verlauf der Fertigung überein. Analog wird das Planen vom Fertigteil zum Rohteil *"Backward Planning"*¹², *"Bottom-Up"* Ansatz oder auch *"Backward Reasoning"*¹³ genannt.
- (b) Die Inferenzrichtung
- Diese Termini sind jedoch nicht zu verwechseln¹⁴ mit der Richtung der Schlußfolgerung, die durch die aus der KI stammenden Begriffe *"Forward Chaining"*¹⁵ und *"Backward Chaining"*¹⁶ angegeben wird. Man spricht von einem datengetriebenen Inferenzprozeß, wenn durch den Vergleich des Bedingungssteiles einer Regel mit den geltenden Fakten versucht wird, den Aktionsteil der Regel als zusätz-

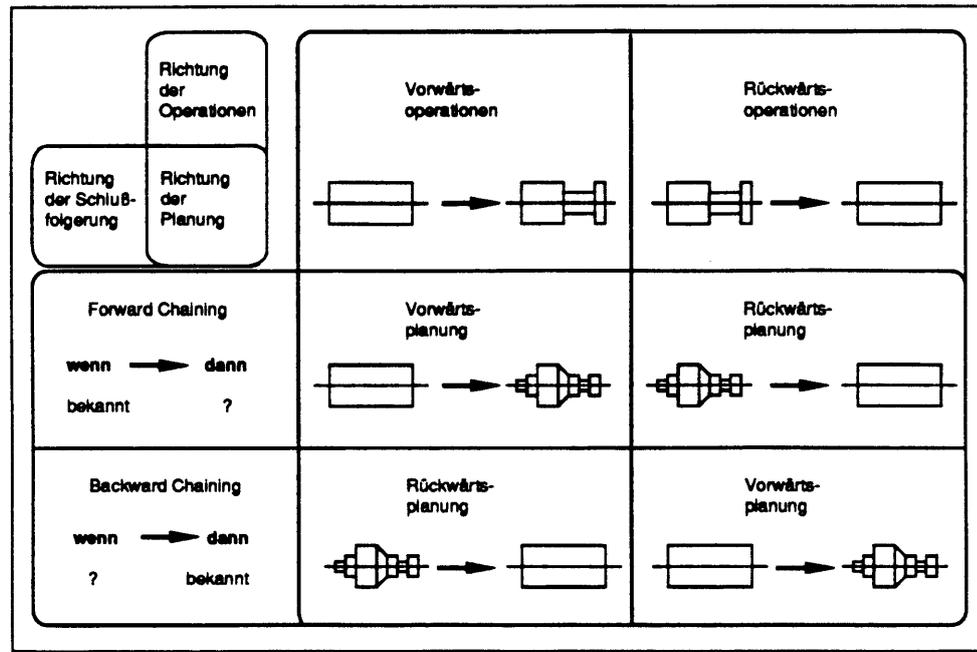


Abbildung 3.9: Vier Strategien der Arbeitsplangenerierung nach [57]

andererseits als *"materialhinzufügende"*¹⁸ Aktion spezifiziert werden. Deshalb ergeben sich in Abbildung 3.9 vier Möglichkeiten für die Planungsrichtung nach [57].

¹⁸additiver Operator

3.4.3 Herstellungswissen

Neben der Werkstückbeschreibung und der Entscheidungslogik benötigt ein generatives AP-System weitere Informationen zur Darstellung des Werkstückes. Häufig sind diese firmenspezifisch, wie z.B. Informationen über vorhandene Rohteile. Abbildung 3.10 zeigt, welche Daten häufig noch gebraucht werden.

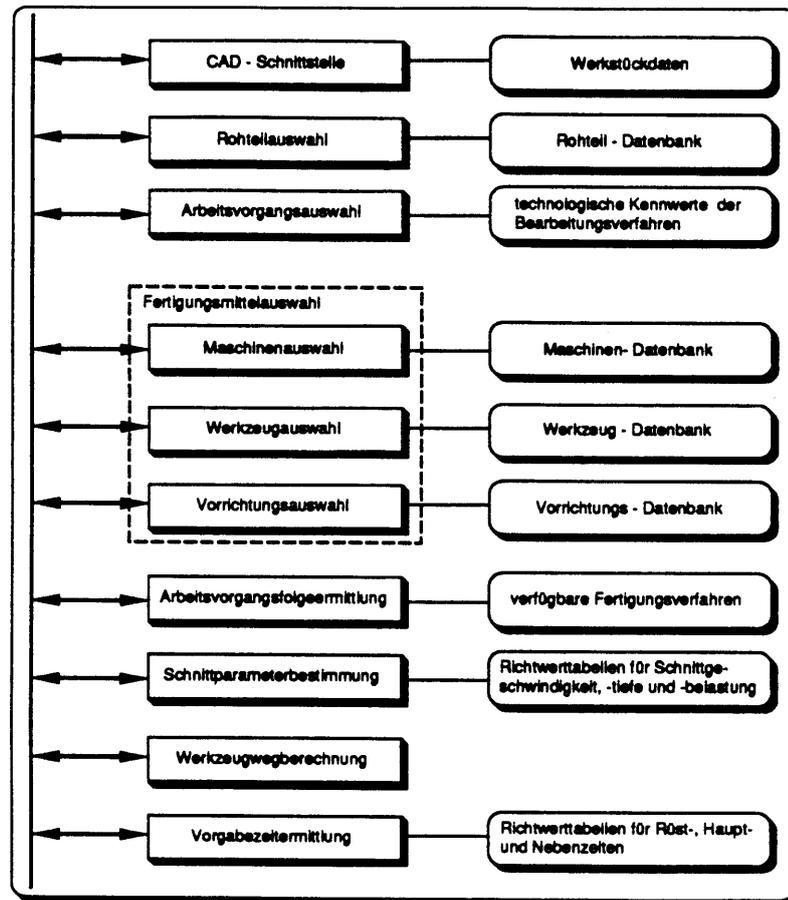


Abbildung 3.10: Arbeitsplanungsfunktionen und unterstützende Dateien nach [49] und [52]

3.5 Überblick: CAPP-Systeme

In diesem Abschnitt werden einige CAPP-Systeme beschrieben. Dabei werden vor allem die CAPP-Systeme herausgegriffen, die KI-Methoden verwenden. Abbildung 3.11 zeigt eine Übersicht (siehe auch Anhang B).

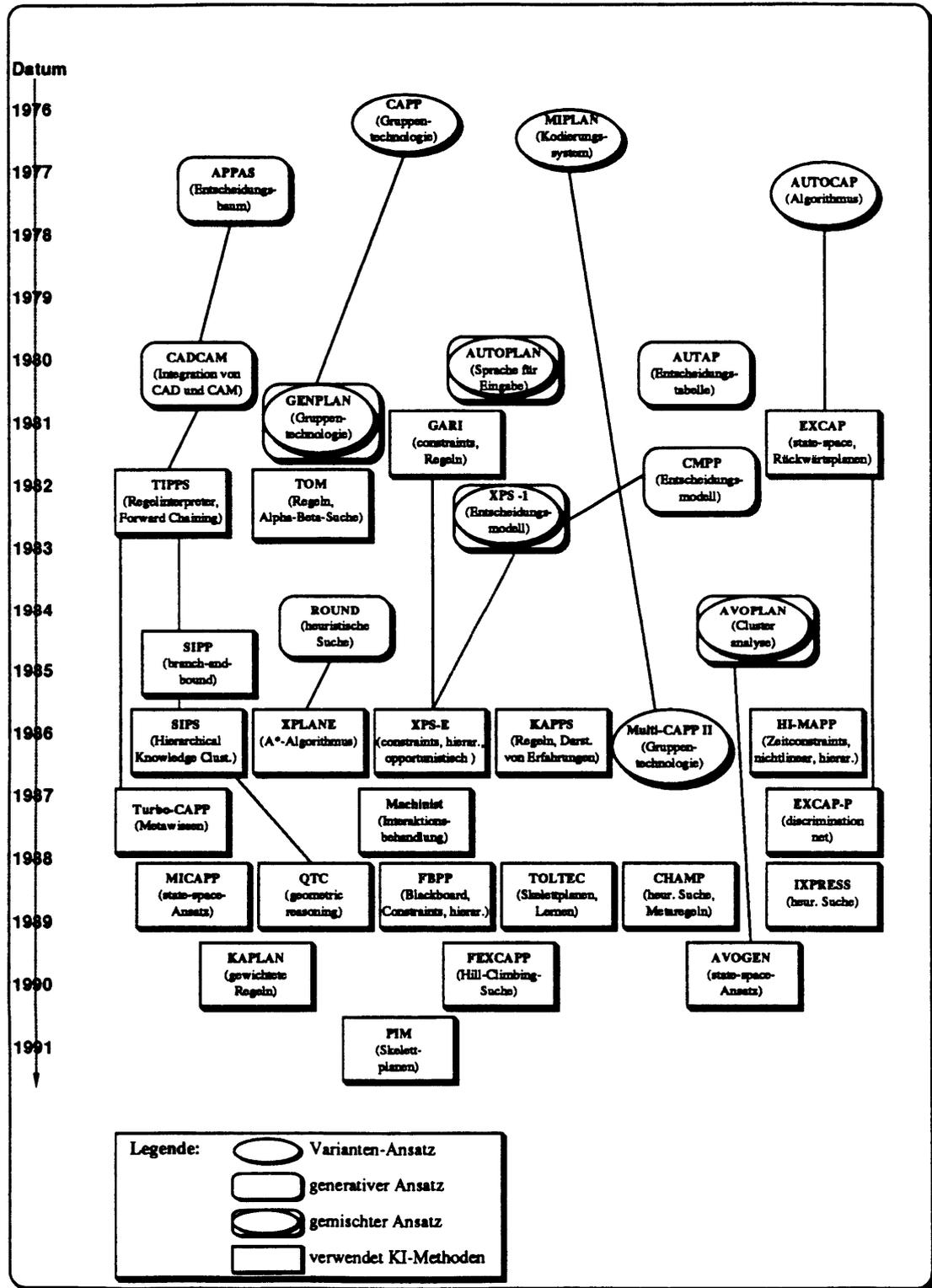


Abbildung 3.11: Chronologie der bekanntesten CAPP-Systeme, wobei der Schwerpunkt auf den KI-basierten Systemen liegt

3.5.1 Die ersten Systeme

Das nach [46] wahrscheinlich erste und auch am weitesten verbreitete AP-System ist das 1976 entwickelte System "CAPP"¹⁹ [96]. CAPP ist ein Datenbanksystem, welches das Wiederfinden und interaktive Bearbeiten von Standardarbeitsplänen mittels Gruppentechnologie unterstützt, also ein typisches Varianten-System. Ein weiteres frühes CAPP-System nach dem Varianten-Ansatz ist "MIPLAN" [81]. Das Nachfolgesystem von MIPLAN heißt "MULTI-CAPP II". Es zeichnet sich durch sein breites Spektrum der bearbeitbaren Teile aus. MULTI-CAPP II besteht aus mehreren Modulen, die alle eine einzige Datenbank benutzen [46].

3.5.2 Von APPAS bis QTC

Die Entwicklung im Bereich der CAPP-Systeme läßt sich am Beispiel der von Wysk, Chang und Nau beschriebenen Systeme verdeutlichen.

1977 wurde an der Purdue University von R. A. Wysk "APPAS"²⁰, das erste generative CAPP-System präsentiert [120]. APPAS verwendet implizit einen Entscheidungsbaumansatz und ist in FORTRAN implementiert. Ausgehend von der Beschreibung des Werkstückes durch den Anwender in einer speziellen Beschreibungssprache plant das System die Reihenfolge der Arbeitsvorgänge, sucht geeignete Werkzeuge aus und bestimmt Prozeßparameter, wie z.B. die Schnittgeschwindigkeit.

Das System "TIPPS"²¹ ist ein Nachfolgesystem von APPAS und wurde von T. C. Chang und R. A. Wysk 1982 am Virginia Polytechnic Institute and State University entwickelt [50] (vgl. Abbildung 3.12). In TIPPS ist ein CAD-System integriert, in dem das Werkstück durch dem Planungssystem bekannte Features beschrieben wird. Das Planungssystem besteht aus einer Wissensbasis, in der die Regeln zur Auswahl eines Arbeitsvorganges sowie eine Menge von Design-Features abgelegt sind, und einem Interpreter, der aufgrund der Werkstückbeschreibung und den Auswahlregeln einen Arbeitsplan generiert.

Ein weiterer Schritt in Richtung wissensbasierte Planung wurde dann mit dem System "SIPP"²² getan. SIPP wurde 1985 von D. S. Nau an der University of Maryland vorgestellt [101] (vgl. Abbildung 3.13). Es zeichnet sich durch eine Feature-Hierarchie aus, die durch ein Framesystem repräsentiert wird. Die Reihenfolge der Arbeitsvorgänge wird in SIPP durch die Design-Folge bestimmt. Als Planungsstrategie wird ein Suchverfahren basierend auf dem branch and bound Verfahren verwendet.

Während SIPP in Prolog implementiert wurde, setzt die Weiterentwicklung "SIPS"²³ auf LISP auf [102]. SIPS unterscheidet sich von SIPP durch die Repräsentation des Arbeitsplanungswissens in Form von "Hierarchical Knowledge Clustering" (vgl. Abbildung 3.14). Diese Operatorhierarchie steuert den Prozeß der Planverfeinerung. Allerdings wird jedes Feature einzeln und unabhängig von der Gesamtstruktur geplant. Deshalb werden mögliche

¹⁹CAM-I's Automated Process Planning System

²⁰Automated Process Planning And Selection

²¹Totally Integrated Process Planning System

²²Semi-Intelligent Process Planner

²³Semi-Intelligent Process Selector

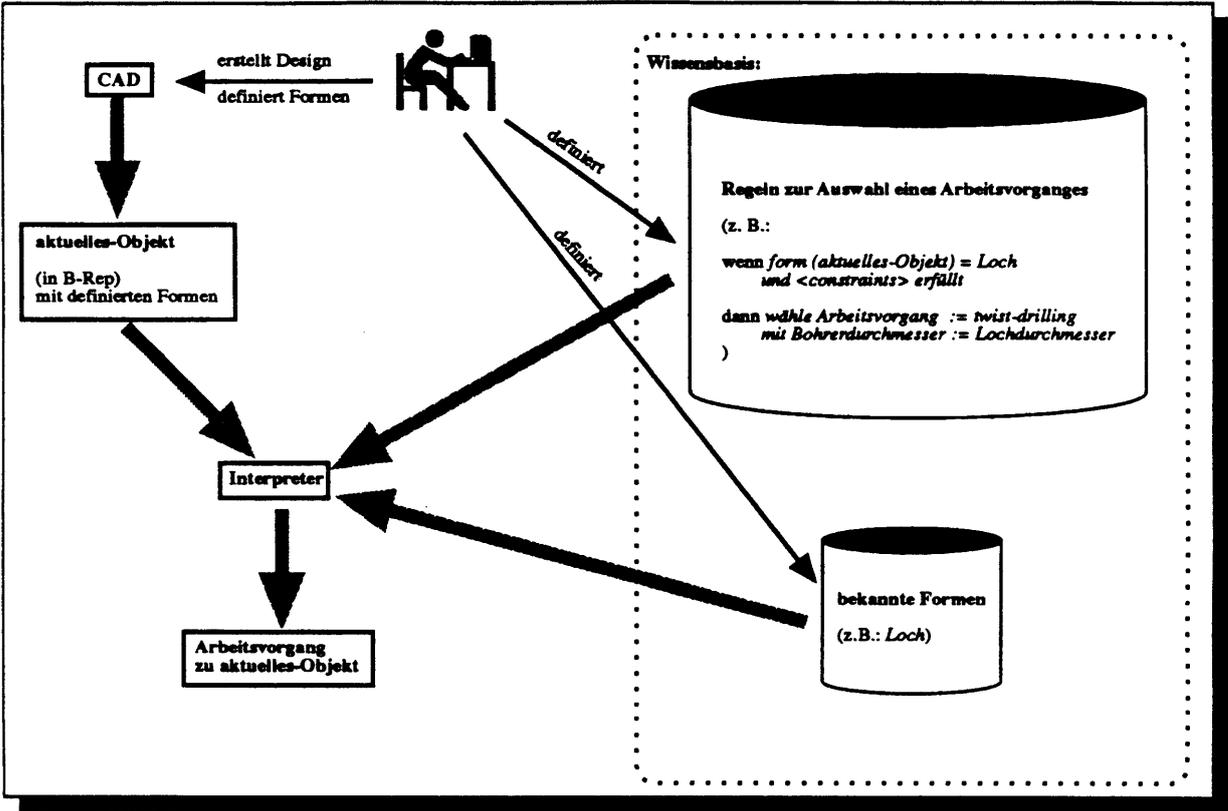


Abbildung 3.12: TIPPS: Auswahl eines Arbeitsvorganges

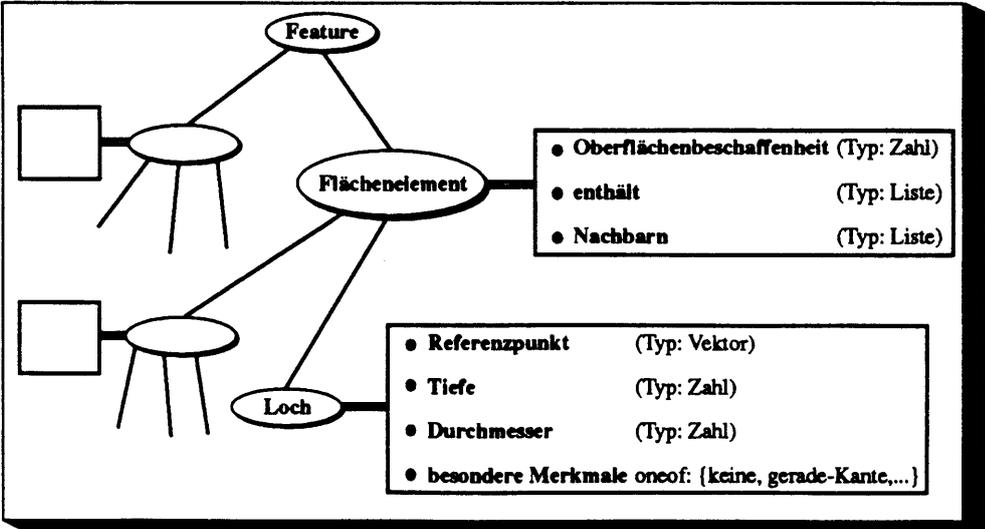


Abbildung 3.13: SIPP: Framesystem für Feature-Hierarchie

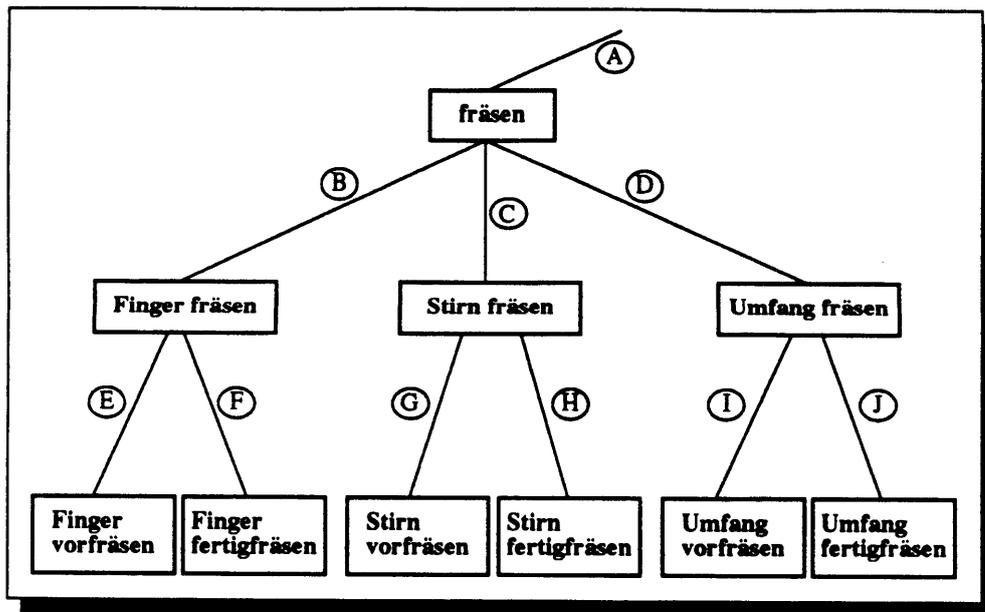


Abbildung 3.14: SIPS: Hierarchical Knowledge Clustering

Interaktionen zwischen den Features nicht berücksichtigt²⁴.

In [49] stellt T. C. Chang das System "QTC"²⁵ vor, das durch seinen hohen Automatisierungsgrad besticht (vgl. Abbildung 3.15). Deshalb ist das System besonders geeignet für Forschungs- und Entwicklungsabteilungen von großen Fertigungsunternehmen, da dort fast jedes Teil neu geplant werden muß. Nach dem Konstruieren des Werkstückes mittels eines Feature-basierten CAD-Systemes verläuft die weitere Fertigung automatisch. Die

wichtigsten Kennzeichen dieses Systemes sind:

- die Erkennung von fertigungsspezifischen Features aus Design-Features mittels geometric reasoning²⁶
- das Erstellen des Gesamtplanes unter Berücksichtigung von Interaktionen zwischen den Teilplänen
- Frames zur Darstellung der Prozeßhierarchie (Hierarchical Knowledge Clustering) und des Werkstückes

3.5.3 Weitere KI-basierte Systeme

Das erste KI-basierte System war "GARI", das von Descotte und Latombe 1981 an der Universität Grenoble entwickelt wurde [71]. GARI startet mit einem initialen "initiali-

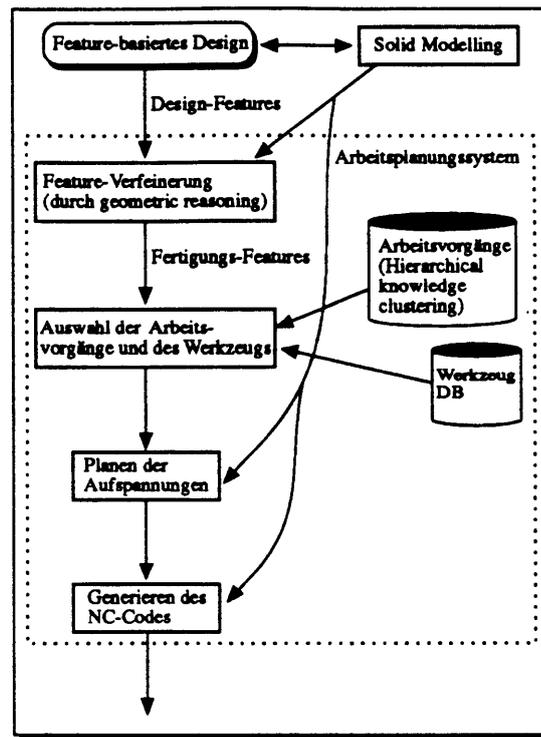


Abbildung 3.15: Architektur von QTC

Backtracking gelöst. Der wesentliche Nachteil von GARI ist das Fehlen von hierarchischen Planungsmethoden. Eine Weiterentwicklung von GARI ist das System "XPS-E" [112], das von J. P. Tsang 1986 beschrieben wird. Es verwendet eine "General Purpose/Specific Purpose Kernel"-Architektur, d.h., der innere Systemaufbau wird explizit von der Repräsentationsstruktur des Anwendungsbereiches getrennt. Dieser Ansatz unterstützt die Portabilität des Systemes auf verschiedene Planungsbereiche. XPS-E plant hierarchisch und opportunistisch.

Das System "TOM"²⁷ wurde 1982 an der Universität Tokio erstellt [98]. Das System besteht aus einer Menge von Rückwärtsregeln, die die Arbeitsvorgänge als Operatoren durch Vor- und Nachbedingung beschreiben. Als Kontrollstrategie verwendet TOM eine heuristisch gesteuerte Suche, die beim Zielzustand des Werkstückes beginnt (Backward Planning).

Auch das CAPP-System "EXCAP-P", das an der Universität Manchester 1986 vorgestellt wurde, benutzt Backward Planning mit Rückwärtsoperationen [87]. Der Zustand des Werkstückes wird durch ein internes Modell dargestellt, das nach jeder Operation durch Feature Recognition angepaßt wird.

Als letztes soll an dieser Stelle das System "TOLTEC" erwähnt werden, da es das bisher einzige System ist, das eine gewisse Lernfähigkeit besitzt [58]. Ein Planschritt besteht in TOLTEC aus drei Phasen:

1. Auswahl von geeigneten Skelettplänen

²⁷Technostructure Of Machining

2. Überprüfung der Anwendbarkeit und Güte der Skelettpläne; zuerst wird getestet, ob ein Plan constraints verletzt; danach wird der Plan, der am besten geeignet erscheint, ausgesucht; die anderen Pläne werden gespeichert, da sie für ein Backtracking in Betracht kommen.
3. Skelettplanverfeinerung

Der Lernmechanismus von TOLTEC basiert auf den Ideen von R. C. Schank [35] über Lernen aus dem Scheitern von Vorhersagen. Es werden drei verschiedene Fehler berücksichtigt:

1. constraint Widerspruch; als Reaktion identifiziert das System den Teilplan, der den Widerspruch produzierte, und fügt nach Rückfrage an den Benutzer den constraint für diesen Teilplan in die Wissensbasis ein.
2. Ein Plan ist nicht geeignet; dies kann die Folge aus einer schlechten Skelettplanauswahl sein.
3. Design Fehler; in diesem Fall generiert das System eine Warnung für den Benutzer

3.5.4 Fazit

Die untersuchten KI-basierten CAPP-Systeme (vgl. Anhang B) verwenden überwiegend die folgenden Methoden und Ansätze:

- Suchverfahren: PROPLAN, X-PLANE
- constraints: GARI, HI-MAPP
- hierarchisches Planen: PIM, XPS-E
- Skelettplantechnik: PIM, TOLTEC
- Feature-Recognition: FEXCAPP, PIM
- spezielles CAD-System (vgl. Seite 25): TIPPS, XCUT
- Blackboard-Architektur, Meta-Regeln: FBPP, XPS-E
- Backward Planning: alle
- Forward-Chain-Back-Chain-Technik: EXCAP, PIM

- hierarchisch (z.B. Frames): PROPEL, SIPP
- Beschreibungssprache: GARI, TOM
- Arbeitsplanungswissen:
 - hierarchisch (z.B. Frames): KAPPS, SIPS
 - Regeln: GARI, TOM
- Programmiersprachen
 - LISP: GARI, XMAPP
 - PROLOG: EXCAP-P, SIPP
 - OPS5: FEXCAPP
 - SMALLTALK: KAPPS
 - FORTRAN: TIPPS, X-PLANE
 - PASCAL: TOM

Kapitel 4

Diskussion

In diesem Kapitel wird eine Bewertung der Planungsverfahren der KI im Hinblick auf ihre Eignung für die Arbeitsplanung versucht.

4.1 Beurteilungskriterien von Planungsverfahren

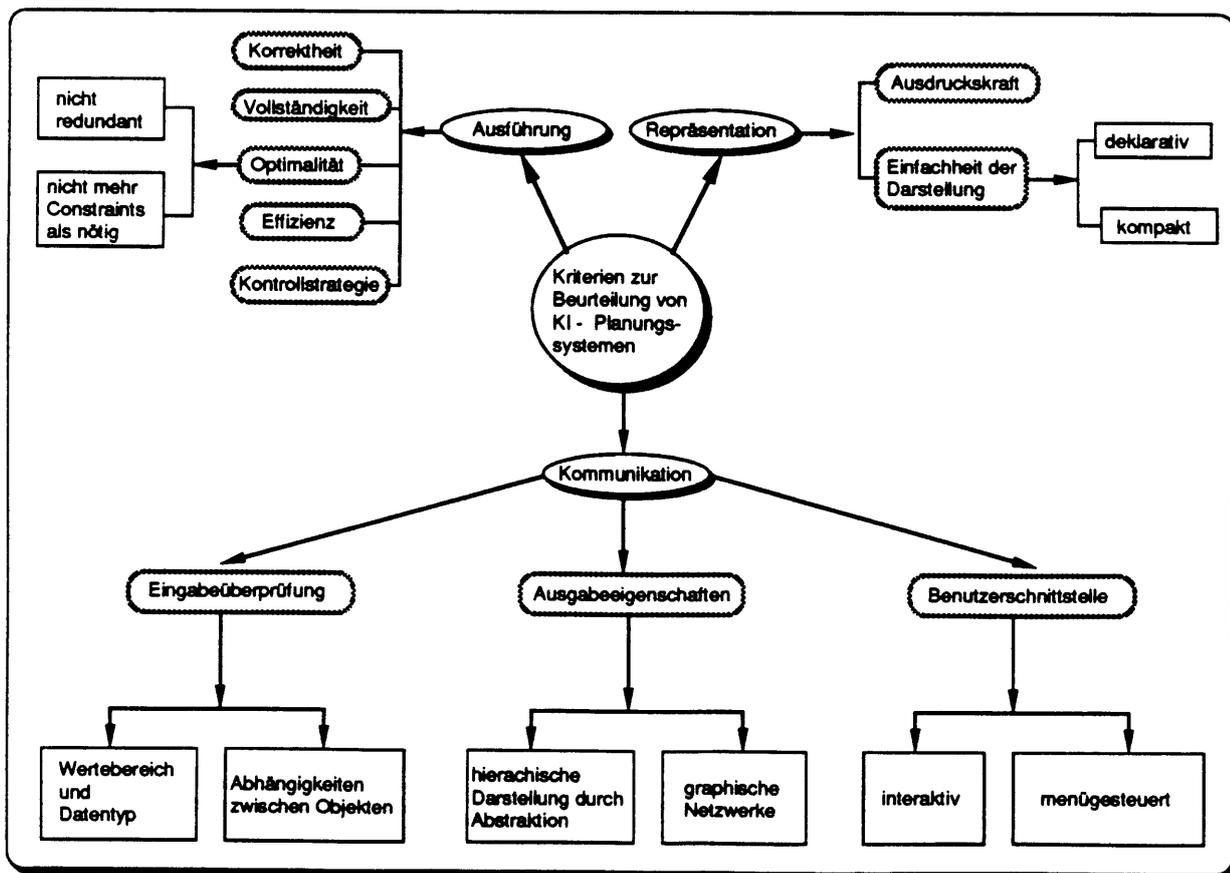


Abbildung 4.1: CAP-Bewertungskriterien nach [22]

N. A. Kartam und D. E. Wilkins nennen in [22] die folgenden Kriterien zur Beurteilung von Planungsverfahren (vgl. Abbildung 4.1):

1. Korrektheit
2. Vollständigkeit
3. Optimalität
4. Effizienz
5. Kontrollstrategie

Da manche Planungsverfahren eine bestimmte Repräsentationsform implizieren (so basiert z.B. STRIPS darauf, daß keine universell quantifizierten Operatoren erlaubt sind), spielt auch die Ausdruckskraft und die Einfachheit der Darstellung des Planungswissens eine Rolle bei der Beurteilung der Planungsverfahren. Die Unterstützung der Eingabe, die Ausgabeigenschaften des Planungssystemes und die Benutzerschnittstelle sind zur Beurteilung der Leistungsfähigkeit eines Planungsverfahrens zweitrangig und werden deshalb im folgenden nicht berücksichtigt.

4.2 Merkmale des Planungsbereiches AP

Zunächst stellt sich die Frage, welche Anforderungen an ein Planungsverfahren in der Arbeitsplanung gestellt werden. Der Planungsbereich Arbeitsplanung hat die folgenden Eigenschaften:

1. viele Interaktionen

Die meisten Arbeitsvorgänge sind nicht unabhängig voneinander. Diese "Interaktionen" muß das Planungssystem berücksichtigen. C. C. Hayes identifiziert in [54, 53] analog zu Hertzberg (vgl. Seite 12) negative und positive Interaktionen. Mit "Feature interaction" bezeichnet sie den Fall, wenn die Fertigung einer Gruppe von Features die Art und Weise der Fertigung von anderen Features beeinflusst. C. C. Hayes führt aus, daß die meisten Feature Interaktionen aus Einspannproblemen resultieren; so zerstört die Fertigung eines Features häufig die Spannflächen, die nötig sind, um ein anderes Feature zu fertigen. Zwei weitere Gründe für Interaktionen sind Kollisionen zwischen dem nicht schneidenden Teil eines Schneidewerkzeugs und dem Werkstück, sowie technologische Gründe (vgl. Seite 28).

Eine positive Interaktion, die zur Optimierung eines Arbeitsplanes ausgenutzt werden sollte, ist das "Überlappen von Arbeitsvorgangfolgen"¹. Die wichtigste und häufigste Art der Überlappung von Arbeitsvorgangfolgen ist die Anzahl und Art der Einspannungen, da sie die meiste Zeit in Anspruch nehmen (bis zu 90% der gesamten Fertigungszeit). Deshalb ist es eine gute Heuristik, die Anzahl der Einspannungen zu reduzieren. Eine andere wichtige Überlappungsart ist die Verwendung des gleichen Werkzeuges in mehreren Arbeitsvorgangfolgen. Die Anzahl der Werkzeugwechsel

¹von C. C. Hayes als "operator overlap" bezeichnet

sollte möglichst minimiert werden, z.B. durch das Gruppieren von Arbeitsvorgängen, die das gleiche Werkzeug benötigen.

2. viele ähnliche, parametrisierte Operatoren mit großem Wertebereich

Führt man das Arbeitsplanungsproblem auf das klassische Planungsproblem zurück, dann werden gewöhnlich die Fertigungsfeatures als Planziele und die Menge der verfügbaren Arbeitsvorgänge als Planoperatoren definiert. Da die Anwendbarkeit eines Fertigungsoperators von vielen Faktoren abhängt, benutzt ein Arbeitsplaner eine Operatorhierarchie, um zunächst von unwichtigen Details abstrahieren zu können. Ein Arbeitsvorgang hat meistens viele Parameter, wie z.B. die Ausgangsgeometrie des Werkstückes, das zu benutzende Werkzeug und die Schnittrichtung und -geschwindigkeit. Hier stellt sich das Problem, unter den möglichen Instanzen des Operators eine optimale Belegung zu finden.

3. viele, unterschiedlich optimale Lösungen

Zum Erreichen eines Zieles (Features) können oft viele verschiedene Operatoren (Arbeitsvorgänge) in Frage kommen. Auch hier taucht das Problem der Optimierung der Lösung auf. Die Optimalität einer Lösung hängt im wesentlichen von der Fertigungszeit und den Fertigungskosten ab. Die Relevanz der Fertigungskosten wird durch die Stückzahl der zu fertigenden Teile bestimmt.

4. die Korrektheit der Lösung ist wichtig

Die vom Planungssystem generierte Lösung (Arbeitsplan) muß korrekt sein, da die Wirkung der Operatoren (Arbeitsvorgänge) im Gegensatz zu anderen Domänen nicht rückgängig gemacht werden kann, außer wenn das Planungssystem über eine Komponente zur Simulation des Planes verfügt.

5. die Problemlösung ist vom Umfeld (Betrieb) abhängig

Die generierte Lösung (Arbeitsplan) hängt stark von den zur Verfügung stehenden Mitteln (Maschinen, Werkzeuge, Rohteile, usw.) ab. Deshalb ist eine Trennung in des Planungssystemes in einen betriebsabhängigen Teil und einen allgemeingültigen Teil sinnvoll. Außerdem ist es von Vorteil, mehrere Alternativpläne zur Herstellung eines Werkstückes zu verwalten.

6. das Umfeld kann sich mittelfristig ändern

Das Umfeld, also Maschinen, Werkzeuge und Rohteile, in einem Fertigungsbetrieb unterliegt einem ständigen Wandel. Ein Arbeitsplanungssystem muß in der Lage sein, diesen Wandel nachzuvollziehen. Deshalb bietet sich eine wissensbasierte, deklarative Programmierung an.

7. Entscheidungen basieren auf Erfahrung

Viele Entscheidungen bei der Arbeitsplanerstellung werden vom Arbeitsplaner aufgrund seiner Erfahrung getroffen. Es wäre wünschenswert, wenn auch das Planungssystem diese Fähigkeit zum Lernen besitzen würde. Außerdem sollte es über eine geeignete Repräsentationsform für den Erfahrungsschatz des Arbeitsplaners verfügen. Auch hier wäre eine wissensbasierte, deklarative Darstellung, wie sie z.B. durch Skelettpläne gegeben ist, von Vorteil.

- Die Lösung des Frame-Problems bei Verwendung der Prädikatenlogik 1. Stufe ist mit sehr viel Aufwand (Rahmenaxiome) verbunden.

Kommentar:

Da in der AP sehr viele Operatoren vorkommen, ist die Lösung des Frame-Problems praktisch unmöglich (vgl. [18]). Deshalb ist Planen mittels Inferenz als Planungsme-

4.4.2 lineares Planen (nach STRIPS)

Kurzbeschreibung:

Für einen linearen Planer besteht die Welt aus Start- und Zielzustand sowie zustandsverändernden Operatoren. Durch Rückwärtssuche vom Zielzustand zum Startzustand wird versucht, eine Operatorfolge zu finden, die den Startzustand in den Zielzustand überführt.

Vorteile:

- Das Frame-Problem wird durch die STRIPS-Assumption (vgl. Seite 11) gelöst.
- Die Kontrollstrategie ist einfach und durchschaubar.

Nachteile:

- Durch die Linearity-Assumption (vgl. Seite 11) werden nicht alle Lösungen gefunden.

Vorteile:

- Der resultierende Plan kann zu mehreren verschiedenen Plänen linearisiert werden.

Nachteile:

- Nichtlineares Planen löst die Interaktion von Teilplänen nicht effizienter als lineares Planen; man kann dort Konflikte lediglich rationaler behandeln (vgl. [18]).
- Das Formulieren von Regeln zum Erkennen von Interaktionen zwischen Arbeitsvorgangfolgen ist schwierig.
- Arbeitsplanungsspezifisches Wissen zur Steuerung der Auswahl von Operatoren kann nicht in geeigneter Form repräsentiert werden.

Kommentar:

Nichtlineares Planen allein ist ungeeignet für die Arbeitsplanung. In [60] wird gezeigt, daß das Finden des besten Planes beim Kombinieren von Arbeitsplänen von Features zu einem Gesamtplan für das ganze Werkstück ein NP-hartes Problem ist. Beschränkt man sich jedoch auf die Auflösung der *schlimmsten* Interaktionen, dann kann das Problem nach [56] in $O(n^2)$ gelöst werden.

4.4.4 hierarchisches Planen

Kurzbeschreibung:

Die Grundidee des hierarchischen Planens besteht darin, die Komplexität des Planens durch die Abstraktion von Details zu reduzieren.

Vorteile:

- Man bekommt bessere Lösungen, wenn man das Expertenwissen durch geeignete Abstraktionsebenen ausdrückt.
- Der Planvorgang wird durchschaubarer und schneller.
- Hierarchisches Planen entspricht am ehesten der menschlichen Arbeitsweise [55].
- Für einen abstrakten Operator sind weniger Vorbedingungen anzugeben.

Nachteile:

- Durch Abstraktion wird Planen übersichtlicher; andererseits kann sie zur Folge haben, daß man Details übersieht, weil man nicht jedes Detail berücksichtigt.
- Es kann passieren, daß man nach der Expansion eines abstrakten Operators einen nicht optimalen Plan erhält. In diesem Fall müßte also die Entscheidung über die Auflösung einer Interaktion bis nach der Konkretisierung des Planes zurückgestellt werden.
- Nicht jedes Problem läßt sich hierarchisch nach dem "Teile-und-Herrsche-Prinzip"² lösen.

²"divide et impera"-Prinzip

- Es sind mehr Operatoren zu spezifizieren. Diese haben allerdings dann weniger Vorbedingungen.

Kommentar:

Entscheidend für das hierarchische Planen ist es, daß man auf den höchsten Stufen die "richtigen" Pläne findet. Dies gilt besonders dann, wenn die Elemente einer Abstraktionsebene unabhängig voneinander sein sollen. Zwei Ansätze für hierarchisches Arbeitsplanen sind Feature-Hierarchie (vgl. Seite 35) und Arbeitsvorgangshierarchie (vgl. "Hierarchical Knowledge Clustering", Seite 36).

4.4.5 Planen mit constraints

Kurzbeschreibung:

Durch constraints lassen sich Abhängigkeiten zwischen Werten von Variablen ausdrücken. Die Grundidee des Planens mit constraints ist das "least commitment"-Prinzip, d.h., man macht nur soviele Beschränkungen für Tatsachen im Plan (z.B. Linearisierung von Operatoren), wie dies nötig ist, um die Vorbedingungen aller Operatoren zu erfüllen und Konflikte aufzulösen.

Vorteile:

- Die Auswahl von Operatoren und Planobjekten kann solange hinausgezögert werden, bis genug Information dazu bereit steht. Deshalb wird weniger Backtracking benötigt.
- Constraints eignen sich auch zur Darstellung von Relationen zwischen Planobjekten, die nicht hierarchisch organisiert sind.
- Interaktionen zwischen Planschritten (Arbeitsvorgängen) können explizit spezifiziert werden.
- Die Schwierigkeit, eine effiziente Repräsentation für die sich ständig ändernden Werkstückzustände zu finden, entfällt, da keine Zustände dargestellt werden [55].

Nachteile:

- Man sollte eine Vermischung von Merkmalen, für die constraints formuliert wurden, und Merkmalen, die durch Operatoren verändert werden können, vermeiden (vgl. [18]).
- Die Plangenerierung ist schwer nachvollziehbar.

Kommentar:

Da in der Arbeitsplanung häufig Auswahlprobleme (z.B. Werkzeugauswahl) zu lösen sind, sind constraints hier eine geeignete Technik, um den Wertebereich einzuschränken (vgl. z.B. GARI [71], HI-MAPP (Zeitconstraints) [47] und XPS-E [112]).

4.4.6 Meta-Planen, opportunistisches Planen

Kurzbeschreibung:

Meta-Planen ist sozusagen das Planen des Planens. Die Kontrollstruktur wird deklarativ spezifiziert, so daß das Programmverhalten durchschaubar und besser nachvollziehbar wird.

Beim opportunistischem Planen arbeiten mehrere Spezialisten gleichzeitig an der Lösung eines Zieles. Dem erfolgversprechendsten Ziel wird dann nachgegangen.

Gemeinsam ist beiden Ansätzen die deklarative Spezifikation der Kontrollstruktur.

Vorteile: Die wesentlichen Vorteile dieser Ansätze ergeben sich aus der deklarativen Spezifikation der Kontrollstruktur (vgl. [18]):

- Ausdrücklichkeit von Strategien der Programmkontrolle
- Nachvollziehbarkeit des Programmverhaltens
- flexible Kontrollstruktur

Nachteile:

- Die Darstellung des Kontrollwissens durch Operatoren beim Meta-Planen ist unnatürlich. Opportunistisches Planen hingegen vermeidet diesen Nachteil.

Kommentar:

Da opportunistisches Planen nicht den Nachteil des Meta-Planens besitzt, wird es in neueren CAPP-Systemen häufig verwendet (vgl. FBPP [68], und XPS-E [112]). Insbesondere eignet sich der Ansatz gut zur Trennung von anwendungsspezifischem und anwendungsunabhängigen Problemlösungswissen.

4.4.7 Weitere Techniken

Fallbasiertes Planen und lernende Systeme sind zwei weitere interessante Konzepte für die Arbeitsplanung.

Fallbasiertes Planen könnte eine Alternative zur Variantenplanung auf Basis von Gruppentechnologie sein, da nicht nur ein geeigneter Standardarbeitsplan gesucht wird, sondern auch die werkstückspezifische Anpassung des Planes versucht wird. Allerdings steckt diese Technik noch in den Kinderschuhen (vgl. System WAP [82]).

Die Fähigkeit zu lernen sollte in jedem KI-System vorhanden sein. Dies fordern die Entwickler des CAPP-Systemes TOLTEC [58], das sich durch dieses Feature auszeichnet. Was das System jedoch vor allem lernt, ist die Vermeidung von Fehlern zur Verbesserung seines Laufzeitverhaltens. Neue Teile können dadurch jedoch nicht geplant werden.

4.5 Skelettplantchnik (PIM)

Kurzbeschreibung:

In Anhang A wird die Planungskomponente Skippy des PIM-Systemes vorgestellt.

Skippy benutzt die auf Seite 15 beschriebene Skelettplantechnik zur Plangenerierung.

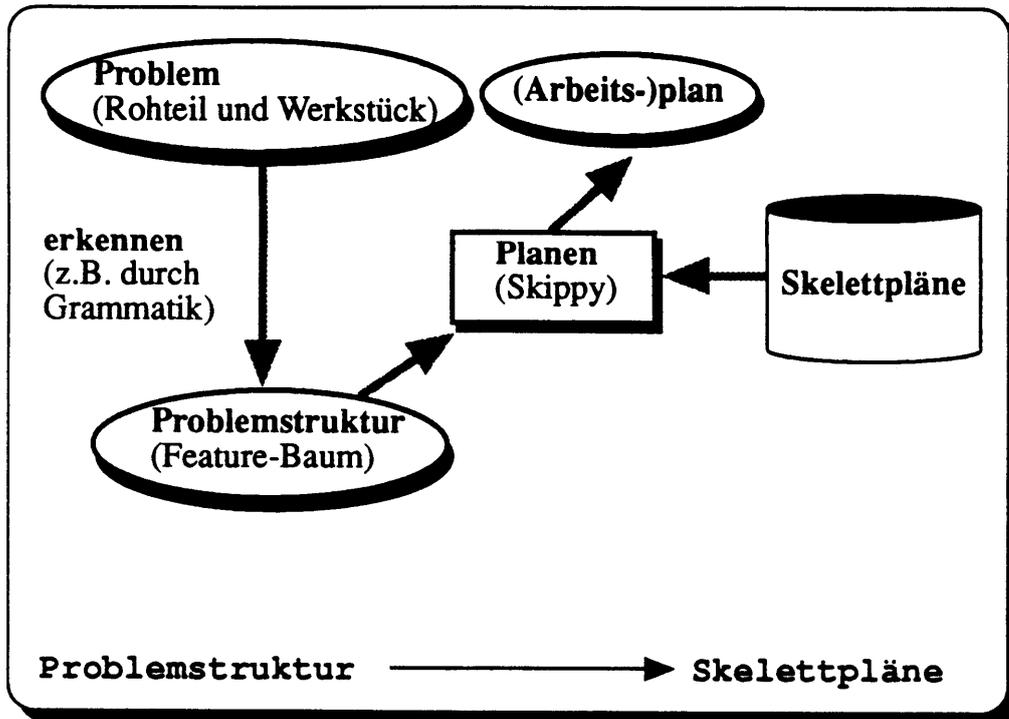


Abbildung 4.2: Skelettplanen im PIM-System

Im Unterschied zum nichtlinearen, hierarchischen Planen geht dieser Ansatz davon

ist möglich, da der Verfeinerungsprozess nicht losgelöst von der Problemstruktur (der

- die Plangenerierung wird übersichtlicher und schneller
- es sind weniger Vorbedingungen zu berücksichtigen (für die Anwendbarkeit eines Skelettplanes)
- Neben dem generativen Ansatz wird auch die Variantenplanung in natürlicher Weise unterstützt (vgl. Seite 55).

Nachteile:

- Das Erkennen der Problemstruktur ist ein komplexer Prozeß. Im Prinzip wird hier ein Teil der Planungsaufgabe schon mitgelöst. Andererseits sinkt durch die Aufteilung die Komplexität der Planungsaufgabe insgesamt. Deshalb könnte man diesen Nachteil auch als Vorteil interpretieren.
- Die starre Kontrollstruktur behindert die Formulierung allgemeingültiger Regeln. So kann z.B. der Ausdruck "Maschine x steht montags nicht zur Verfügung" momentan nicht geeignet dargestellt werden.

Kommentar:

Die aktuelle Version von Skippy geht davon aus, daß sich das Werkstück durch einen Feature-Baum darstellen läßt, da die Auswahl eines Skelettplanes sich aus dem Matching mit einem Teil des Feature-Baumes ergibt (vgl. Seite 61). Ein Feature kann jedoch Teil von zwei voneinander unabhängigen Features sein.

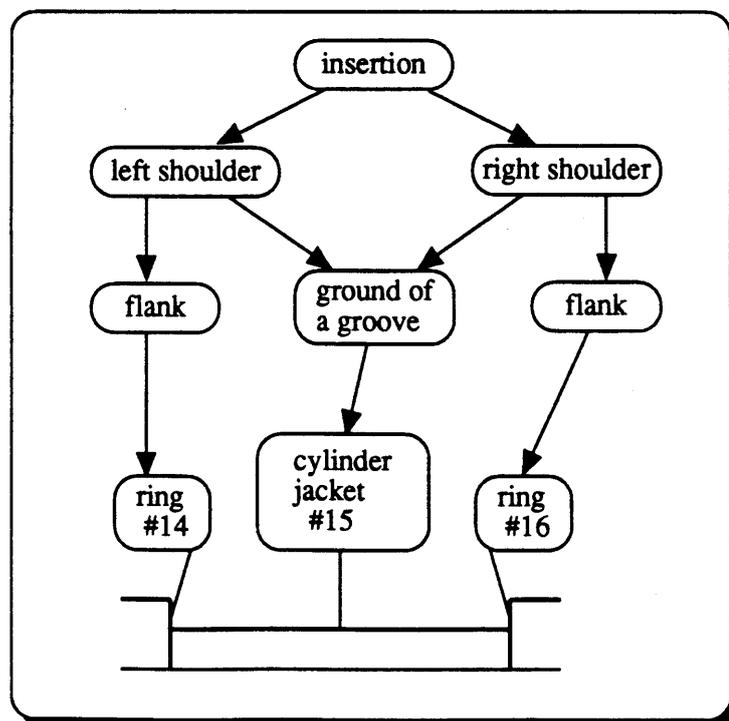


Abbildung 4.3: Ausschnitt aus einem Feature-Graph

```
(make-skp
  :special-tree
    ('("insertion" insertion
      (("lshoulder" nil
        (("flank" nil nil)
         ("groundofagroove" nil nil) ))
       ("rshoulder" nil
        (("groundofagroove" nil nil)
         ("flank" nil nil) )) ))
  :op-mode nil
  :constraints t
  :actions
  ...
)
```

Abbildung 4.4: Teil eines Skelettplans für den Feature-Graph von Abbildung 4.3

Beispiel:

Das Feature "ground of a groove" in Abbildung 4.3 ist sowohl ein Teil von "left shoulder" wie auch von "right shoulder". Dieses Problem läßt sich durch den Skelettplan in Abbildung 4.4 lösen (siehe auch Anhang A).

Das gleiche Problem auch für andere Features: "flank", "ground of a groove", "lshoulder", "rshoulder".

besteht, entstehen durch Kombinationen von Features im Skelettplan.

```
(make-skp
  :special-tree
    '("insertion" insertion
      ("lshoulder" lsh nil)
      ("rshoulder" rsh nil) ))
  :op-mode nil
  :constraints '(and (has-groundofagroove lsh) (has-groundofagroove rsh))
  :actions
  ...
)
```

Abbildung 4.5: Verbesserte Version des Skelettplanes von Abbildung 4.4

Anhang A

Skippy: Skelettplanen in der AP

Skippy ist eine prototypische Implementierung der Planungskomponente des Arbeitplanungssystemes PIM [65, 92, 88], das im Rahmen des ARC-TEC-Projektes am DFKI in Kaiserslautern entwickelt wird. Es ist integriert in das "*Feat-Rep-Graphics*"-System von J. Schwagereit, das zur Visualisierung von Werkstücken dient, die in der Werkstückbeschreibungssprache "*TEC-REP*" [64] repräsentiert werden.

Das System wurde in drei verschiedenen Common Lisp Umgebungen implementiert:

1. Symbolics Common Lisp + CLOS¹ + X Window Tool Kit ([62])
2. AKCL² + PCL³ + X Window Tool Kit
3. MacIntosh Allegro Common Lisp + PCL + Allegro Window Tool Kit ([62])

Das Vorbild von Skippy ist Friedlands MOLGEN-System [9, 8], das auf Seite 15 erläutert wurde.

Im folgenden Teil wird zunächst ein Überblick über Skippy gegeben. Dann werden der Aufbau eines Skelettplanes, die Kontrollstruktur von Skippy und weitere Details der Implementierung näher beschrieben.

A.1 Überblick

Abbildung A.1 zeigt die Integration von Skippy in PIM. Ein Werkstück⁴ hat aus der Sicht von Skippy zwei Darstellungen:

1. Die TEC-REP-Darstellung [64]
2. Die Feature-Tree-Darstellung
Der "*Feature-Tree*" eines Werkstückes ist ein Baum, der die Zerlegung des Werkstückes in geometrische und technologische Beschreibungselemente — auch Features genannt — widerspiegelt (vgl. Abbildung A.2).

¹Common Lisp Object System

²Austin Kyoto Common Lisp

³Public CommonLoops

⁴hier sind sowohl das herzustellende Teil wie auch das Rohteil gemeint

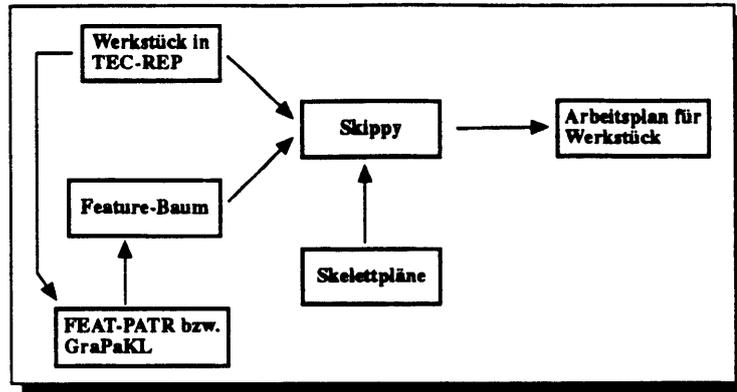


Abbildung A.1: Integration von Skippy in PIM

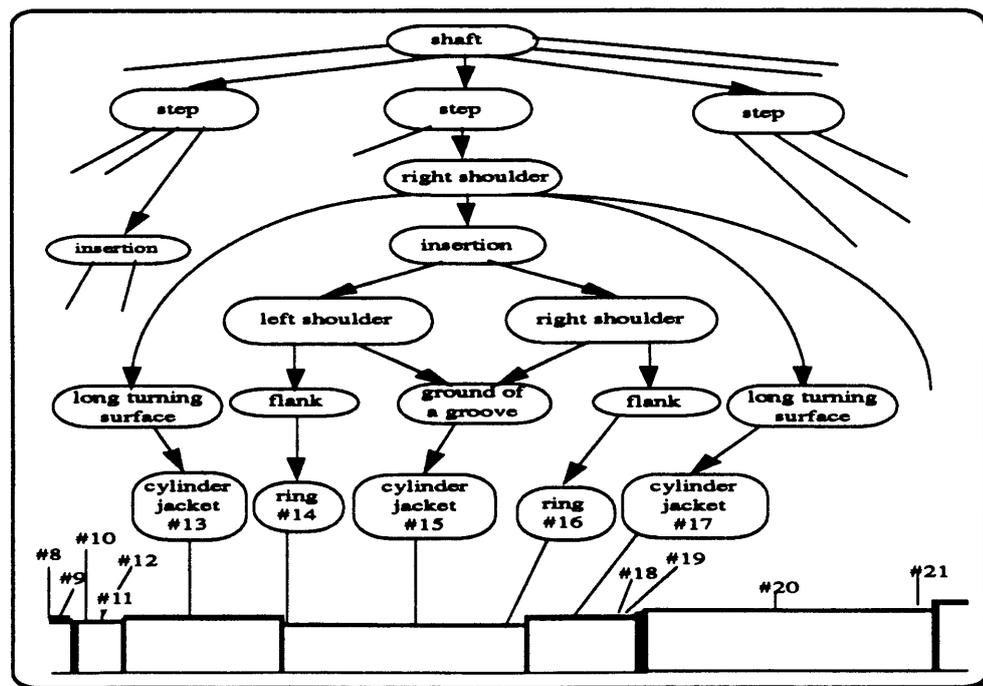


Abbildung A.2: Beschreibung einer Welle als Feature-Tree nach [88]

Momentan wird die Feature-Tree-Darstellung mittels des Parsing-Tools "FEAT-PATR" aus der TEC-REP-Darstellung des Werkstückes gewonnen (vgl. [63]). FEAT-PATR kann nur Drehteile erkennen. In naher Zukunft wird FEAT-PATR jedoch durch ein anderes Parsing-Tool ersetzt⁵, das auch für die Erkennung von Frästeilen geeignet ist.

Die Skelettpläne bilden die Wissensbasis von Skippy. Sie ermöglichen es dem Experten, sein Wissen über die Fertigung von Features in unterschiedlichen Detaillierungsstufen und in einer deklarativen Form zu spezifizieren. Der Aufbau eines Skelettplanes wird in Abschnitt A.2 noch näher erläutert.

Aufgrund der Information über das Werkstück und der Skelettplanwissensbasis generiert Skippy den Arbeitsplan für das Werkstück. Der Arbeitsplan besteht aus einer Folge von Schritten, die aus der Sicht von Skippy atomare Operationen zur Ansteuerung einer CNC-Maschine sind.

Skippy ist nur als prototypisches System anzusehen. Seine Implementierung hat aber gezeigt, daß die Konzepte "Feature Recognition" und "Skeletal Planning" als Grundlage für das Arbeitsplanungssystem PIM geeignet sind.

A.2 Der Aufbau eines Skelettplanes

Ein Skelettplan wird in Skippy repräsentiert durch ein Objekt, das aus vier Komponenten besteht:

1. *special-tree*
2. *op-mode*
3. *constraints*
4. *actions*

Die ersten drei Attribute dienen zur Auswahl eines Skelettplanes, während das Attribut *actions* die Skelettplanverfeinerung spezifiziert. Die ersten drei Attribute werden in diesem

Reihenfolge bei der Auswahl eines Skelettplanes berücksichtigt. Dies ist wichtig, da als Nebeneffekt bei der Auswertung des *special-tree* Variable gebunden werden können, auf die in den *constraints* Bezug genommen werden kann. Abbildung A.3 zeigt ein Beispiel für

```
(make-skp  
:special-tree  
  ("lshoulder" lshoulder  
   ("band" band nil)  
   ("longturninasurface" lts nil) ))
```

A.2.1.2 Semantik eines *special-tree*

Die Knoten eines *special-tree* bestehen — wie aus der obigen Syntaxbeschreibung leicht zu erkennen ist — aus einer Liste mit drei Elementen. Das erste Element bezeichnet den Typ eines Features. Der Wert *nil* dient hier als "Wildcard", d.h., *nil* matched jeden Feature-Typ. Das zweite Element der Liste wird als der Name einer Variablen interpretiert. Während des Matching-Prozesses wird der matchende Feature-Teilbaum an die Variable gebunden. Auf diese Weise hat der Experte die Möglichkeit in den *constraints* oder in den *actions*

mittels der Variablen den Feature-Teilbaum zu referenzieren. Der Wert *nil* bedeutet hier, daß keine Variable definiert wird, da der Feature-Teilbaum für diesen Skelettplan nicht von Interesse ist.

Durch das letzte Element der Liste werden die Unterkomponenten des Features beschrieben. Wird hier *nil* angegeben, dann gilt der Skelettplan für jeden Feature-Baum, dessen Wurzel den gleichen Feature-Typ wie *special-tree* hat (unter Berücksichtigung des *op-mode* und der *constraints*). Ansonsten ist hier ein Unterbaum anzugeben, auf den der Skelettplan beschränkt sein soll.

A.2.2 *op-mode*

Dieses Attribut bestimmt den Kontext des Skelettplanes. Die Vorbedingungen eines Skelettplanes können sehr komplex sein, so daß es sich anbietet, diese in mehreren Stufen zu überprüfen. In diesem Fall kann man mehrere Skelettpläne für den gleichen *special-tree* definieren und den jeweiligen Kontext mittels *op-mode* unterscheiden. Hat *op-mode* den Wert *nil*, dann spielt dieses Attribut bei der Auswahl des Skelettplanes keine Rolle.

A.2.3 *constraints*

Durch das *constraints*-Attribut kann der Experte Vorbedingungen für die Anwendung des Skelettplanes angeben. Dabei kann er sich auf die globalen Variablen **workpiece** und **stock** sowie auf die lokalen Variablen, die in *special-tree* definiert sind, beziehen. Als Bedingung kann jeder gültige Common Lisp Ausdruck verwendet werden.

Beispiel: Der Ausdruck

'(> (object-length *lts*) (object-height *band*))

aus Abbildung A.3 besagt, daß die Länge des Features "longturningsurface" größer als die Höhe des Features "band" sein muß, wenn dieser Skelettplan angewendet werden soll. Dabei sind *lts* und *band* Variable, deren Wert sich aus *special-tree* ableitet.

A.2.4 *actions*

Während die Attribute *special-tree*, *op-mode* und *constraints* Kriterien zur Auswahl eines Skelettplanes darstellen, wird durch *actions* die Skelettplanverfeinerung beschrieben. Unter *actions* ist eine Liste von Aktionen anzugeben. Man kann drei verschiedene Aktionstypen unterscheiden:

1. *atomic-actions*

Eine *atomic-action* kann nicht weiter verfeinert werden. Der Rückgabewert einer *atomic-action* ist eine Instanz der Klasse *atomic-operation*. Diese enthält die vollständige Information zum Ausführen der Aktion, also z.B. die betroffenen Flächen bzw. Features, Werkzeuggruppen usw. Momentan sind folgende *atomic-actions* definiert:

- *chuck*
- *center-hole*
- *roughing*
- *finishing*
- *recessing*
- *tapping*
- *message*

2. *non-atomic-actions*

Eine *non-atomic-action* ist eine Funktion, die eine Instanz der Klasse *atomic-plan* zurückgibt. Ein *atomic-plan* ist eine Folge von *atomic-operations*. Folgende *non-atomic-actions* sind definiert:

- (*plan* <*feature-tree*> :*op-mode* <*op-mode*>)
plan ist eine Funktion, die als erstes Argument einen Feature-Tree erwartet. Sie führt zu einem rekursiven Aufruf des Skelettplaners, d.h., es wird ein Skelettplan zum Feature-Tree gesucht, der dann verfeinert wird. Außerdem kann der Parameter *op-mode* angegeben werden, um den Kontext des gesuchten Skelettplanes näher zu bestimmen.
- (*plan-inverse* <*feature-tree*> :*op-mode* <*op-mode*>)
Manche Features besitzen den gleichen Skelettplan, da sie symmetrisch zueinander sind. Durch die Aktion *plan-inverse* kann der Experte dieses Wissen ausdrücken. *plan-inverse* hat dieselben Argumente wie die Aktion *plan*. Vor dem rekursiven Aufruf des Skelettplaners wird jedoch noch ein Flag, die Variable **inverse-mode**, negiert. Das Flag beeinflusst den Matchingprozeß bei der Skelettplanauswahl. Ist es gesetzt, dann wird durch die Assoziationsliste **inverse-features** definiert, welche Feature-Typen miteinander matchen. In diesem Fall wird z.B. "lshoulder" durch "rshoulder" und "rshoulder" durch "lshoulder" ersetzt. Ist das Feature, für das ein Skelettplan gesucht wird, ein Baum (Feature-Tree), dann werden die Kinder in umgekehrter Reihenfolge mit den Kindern des Feature-Trees (*special-tree*) des Skelettplanes verglichen. Auch hier wird das Matching durch das Flag **inverse-mode** beeinflusst. Ebenso werden die Kinder der Kinder usw. behandelt.

Anzumerken bleibt, daß nur das Verhalten des Matchingprozesses beeinflusst wird. Die Struktur des Feature-Trees, für den ein Skelettplan gesucht wird, bleibt davon unberührt.

• (*message* <*message-description*> :*message-description* <*message-description*>)

Teilpläne zu einen Plan zusammengefaßt. Als Argument von *merge-plans* ist eine Folge von *merge-descriptions* anzugeben, für die die folgende **Syntax** gilt:

```

<merge-description> ::= (<plan-list>
                        [:op <combinable-action> |
                         (<combinable-action>+)]
                        [:kind <kind> | (<kind>+)]
                        [:tool-groups (<tool-group>+)]
                        )
<plan-list>          ::= (<atomic-plan 1> ... <atomic-plan N>)
<combinable-action> ::= 'roughing | 'finishing | 'recessing
                        (vgl. Variable *combinable-actions*)
<kind>              ::= 'lengthwise | 'contour | 'radial
<tool-group>        ::= "eine Werkzeuggruppe"
    
```

Semantik von *merge-plans*

Um unnötige Werkzeugwechsel zu vermeiden und zur Steigerung der Effizienz und der Qualität eines Arbeitsvorganges faßt ein Arbeitsplaner die Bearbeitungsschritte soweit wie möglich zusammen. Beispielsweise lassen sich mehrere Schruppschritte zu einem Schruppschritt und mehrere Schlichtschritte zu einem Schlichtschritt zusammenfassen.

```

(make-skp
 :special-tree
  ("crest" nil
   ("planeface" pl nil)
   ("bezel" bz nil)
   ("longturningsurface" lts nil) ))
 :op-mode nil
 :constraints t
 :actions
 '(
  (bind-local s1
   (make-plan
    (roughing pl :kind 'radial :tool-groups '(w3))
    (finishing pl) ))
  (bind-local s2 (plan bz))
  (bind-local s3 (plan lts))
  (merge-plans
   (list (list s2 s3) :op 'roughing)
   (list (list s1 s2 s3) :op 'finishing) )))
    
```

Abbildung A.4: Skelettplan mit Aktion *merge-plans*

Es gelten die folgenden Regeln für das Zusammenfassen von Bearbeitungsschritten:

- (a) Es können nur Bearbeitungsschritte (*atomic-actions*) gleichen Typs zusammengefaßt werden.
- (b) Bezüglich des Zusammenfassens kann man für einen Bearbeitungsschritt drei verschiedene Merkmale unterscheiden:

- i. prinzipiell zusammenfaßbar

- Durch die Variable **combinable-actions** werden in Skippy die prinzipiell zusammenfaßbaren Bearbeitungsschritte aufgezählt. Momentan sind das die Schritte *roughing*, *finishing* und *recessing*.

- ii. nicht zusammenfaßbar

- Zu diesen gehören z.B. *tapping* und *chuck*.

- iii. ordnend

- Manche Bearbeitungsschritte verhindern das Zusammenfassen von anderen Schritten, da sie eine Partialordnung einführen. Ein Beispiel ist die Ein-, Aus-, bzw. Umspannaktion *chuck*: Bearbeitungsschritte von verschiedenen Aufspannungen können nicht zusammengefaßt werden. In Skippy wird dieses Problem nicht behandelt. Deshalb muß der Experte bei der Spezifikation einer Mischoperation darauf Rücksicht nehmen, ob die zu mischenden Teilpläne Aktionen enthalten, die eine Partialordnung einführen.

- (c) Die Arten (<kind>) der Bearbeitungsschritte müssen miteinander verträglich sein. Die Verträglichkeit der Bearbeitungsschrittarten ist durch das Prädikat *kind-equal* definiert. Es gibt z.B. drei verschiedene Schrupparten: Längsschruppen, Planschruppen und konturfolgend Schruppen. Diese Schrupparten sind verträglich mit sich selbst; außerdem lassen sich Längsschruppen und konturfolgend Schruppen zu einen konturfolgenden Schruppschritt zusammenfassen.
- (d) Es können nur Bearbeitungsschritte zusammengefaßt werden, für die der Schnitt ihrer Werkzeugmengen nicht leer ist.

Diese Regeln werden auch von *merge-plans* beachtet. Die Folge der <merge-description> wird hintereinander abgearbeitet. Jede <merge-description> hat das Mischen der in ihrer <plan-list> aufgeführten Pläne zur Folge. Durch die optionalen Elemente *:op*, *:kind* und *:tool-groups* kann der Experte zusätzliche Einschränkungen für das Mischen angeben.

Zwei *atomic-actions* werden zu einer *atomic-action* zusammengefaßt, in dem der Schnitt ihrer Werkzeuggruppen gebildet wird und die von ihnen bearbeiteten Flächen zusammengefaßt werden.

Nach dem Abarbeiten aller <merge-description> werden die beteiligten Teilpläne in der Reihenfolge ihres Auftretens durch Mischen zu einen Plan zusammengefaßt. Das Mischen ist ordnungserhaltend, d.h. die Ordnung der Operationen

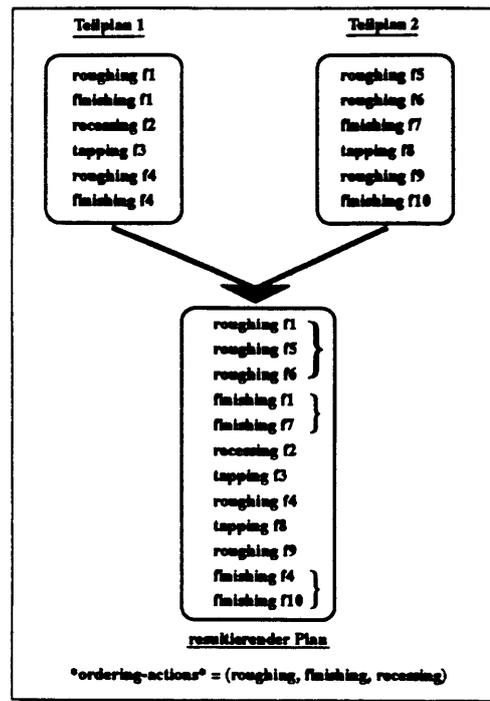


Abbildung A.5: Zusammenfassen von zwei Teilplänen in Skippy

zusammengefaßt werden dürfen.

- (*make-plan* <*atomic-action* 1> ... <*atomic-action* N>)
make-plan erzeugt aus einer Folge von *atomic-actions* einen *atomic-plan*. Dieser kann dann z.B. in *merge-plans* verwendet werden.
- (*dummy-plan* <*message*>)
 Diese Funktion ist als Platzhalter für einen durch <*message*> beschriebenen Plan gedacht. Sie erzeugt einen *atomic-plan*, der den Text <*message*> als einzige Aktion enthält.

3. *bindings*

(*bind-local* <*variablen-name*> <*common lisp ausdrück*>)

(*bind-global* <*variablen-name*> <*common lisp ausdrück*>)

Häufig möchte man das Ergebnis von mehreren *non-atomic-actions* miteinander kombinieren, z.B. beim Mischen von Plänen. Zu diesem Zweck kann man mittels *bind-local* lokale Variable definieren. Allerdings kann eine so definierte Variable nur innerhalb des Skelettplanes referenziert werden. Wenn diese Variable auch in anderen Skelettplänen referenziert werden soll, dann sollte dazu *bind-global* verwendet werden. Allerdings führt man so künstliche Abhängigkeiten zwischen den Skelettplänen ein.

Weitere Erläuterungen zur Semantik des *action* Attributes eines Skelettplanes folgen in Abschnitt A.3.2.

A.3 Skippys Kontrollstruktur

Abbildung A.6 zeigt ein Flußdiagramm des Planungsalgorithmus von Skippy. Der Algorithmus wird aufgerufen mit den Parametern *feature-tree* und *op-mode*. *feature-tree* ist dabei ein Baum, der die Feature-Struktur des Werkstückes widerspiegelt. Der Algorithmus läßt sich wie Friedlands MOLGEN-System [9, 8] in die zwei Schritte **Skelettplanauswahl** und **Skelettplanverfeinerung** unterteilen.

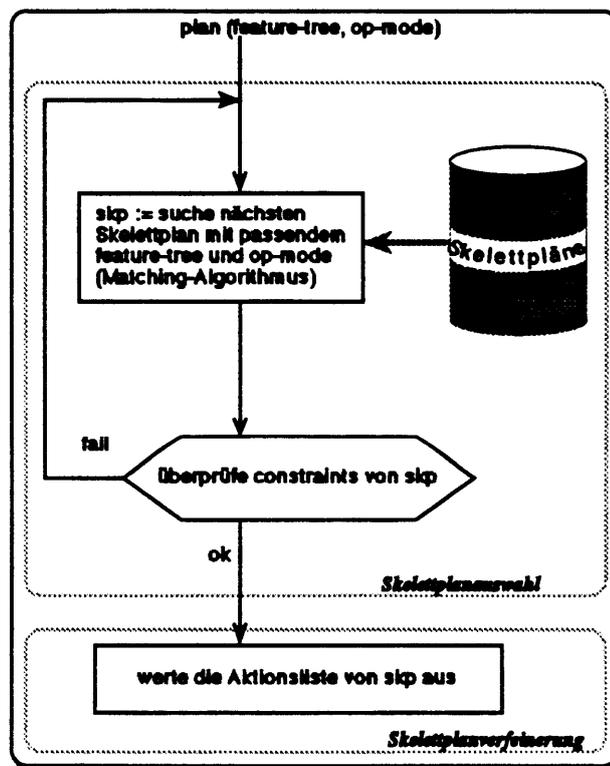


Abbildung A.6: Planungsalgorithmus von Skippy

A.3.1 Skelettplanauswahl

Bei der Suche nach einem geeigneten Skelettplan wird *feature-tree* mit dem *special-tree* der Skelettpläne verglichen (*Matching*). Die Suche verläuft entgegengesetzt zur Spezifikationsfolge der Skelettpläne. Wie schon in Abschnitt A.2.1 beschrieben, liegt ein erfolgreiches *Matching* vor, wenn *special-tree* ein Teilbaum von *feature-tree* ist und die Feature-Typen der Wurzeln der Bäume übereinstimmen. Danach wird überprüft, ob der Parameter *op-mode* mit dem *op-mode* des ausgesuchten Skelettplanes übereinstimmt. Schließlich werden die *constraints* des Skelettplanes ausgewertet. Besteht der Skelettplan alle drei Tests, dann wird die Aktionsliste *actions* des Skelettplans ausgewertet. Andernfalls wird der nächste Skelettplan getestet. Findet Skippy keinen anwendbaren Skelettplan, dann wird eine Warnung ausgegeben und ein Dummyplan generiert.

A.3.2 Skelettplanverfeinerung

Eine Aktion A , die in der Aktionsliste eines Skelettplanes spezifiziert wird, kann eine der folgenden Bedeutungen haben:

$A = (\text{bind-local } \langle \text{variable} \rangle \langle \text{common lisp ausdruck} \rangle)$

⇒ Es wird eine lokale Variable mit dem Namen $\langle \text{variable} \rangle$ definiert. Der Wert des $\langle \text{common lisp ausdruck} \rangle$ wird der Variablen als Wert zugewiesen. Der Geltungsbereich der Variablen beschränkt sich auf den Skelettplan, d.h. sie kann nur innerhalb des Skelettplanes referenziert werden.

$A = (\text{bind-global } \langle \text{variable} \rangle \langle \text{common lisp ausdruck} \rangle)$

⇒ Es wird eine globale Variable mit dem Namen $\langle \text{variable} \rangle$ definiert. Die Variable bekommt den Wert des $\langle \text{common lisp ausdruck} \rangle$ zugewiesen. Die Variable ist während des Auswertens der nachfolgenden Aktionen als globale Variable bekannt und kann deshalb auch in anderen Skelettplänen referenziert werden. Dieses Konstrukt sollte jedoch möglichst vermieden werden, da es eine Abhängigkeit zwischen der Definition der Variablen und ihrer Referenz in anderen Skelettplänen einführt. Skelettpläne sollten möglichst unabhängig voneinander sein, damit ein hoher Grad von Modularität erreicht wird. Andererseits bietet dieser Ausdruck eine einfache Möglichkeit Daten zu verwalten, die in allen Skelettplänen benötigt werden, wie z.B. die aktuelle Schnitttrichtung.

$A = (\langle \text{atomic-action} \rangle \langle \text{parameter} \rangle)$

⇒ Der bisher für den Skelettplan berechnete Arbeitsplan wird um die Aktion A erweitert. ~~A ist also eine $\langle \text{atomic-action} \rangle$ die alle Information zum Ausführen der~~

Aktion wie z.B. betroffene Flächen (Features), Werkzeuggruppen, usw. enthält.

$A = (\langle \text{non-atomic-action} \rangle \langle \text{parameter} \rangle)$

⇒ Der bisher für den Skelettplan berechnete Arbeitsplan wird erweitert um den Plan, der durch das Auswerten von $(\langle \text{non-atomic-action} \rangle \langle \text{parameter} \rangle)$ erzeugt wird.

A.4 Skippys Moduln

Dieser Abschnitt ist als Erläuterung zur Implementierung von Skippy zu verstehen. Skippy besteht aus den folgenden Dateien (in alphabetischer Reihenfolge):

- **actions.lisp**

enthält die Funktionen zum Erzeugen der *atomic-actions* und *non-atomic-actions*.

- *skeletal-plan*
- *feature-tree*
- *atomic-plan*
- *atomic-operation*

Diese Klasse hat die folgenden Unterklassen:

- * *chuck*
- * *center-hole*
- * *roughing*
- * *finishing*
- * *recessing*
- * *tapping*
- * *message*

- **classes-tec-rep.lisp**

enthält die Definitionen der TEC-REP-Klassen (vgl. [64]).

- **export.lisp**

definiert, welche Funktionen von Skippy exportiert werden. Dies sind im wesentlichen Testfunktionen; die wichtigste Funktion ist (*start-planning*).

- **frg-interface.lisp**

bildet die Schnittstelle zum "Feat-Rep-Graphics"-System von J. Schwagereit.

- **globals.lisp**

enthält Definitionen von globalen Variablen.

- **graphics.lisp**

definiert den *Plan Button* als Erweiterung des "Feat-Rep-Graphics"-Systemes.

- **init-skp.lisp**

enthält Funktionen zum Laden der Werkstückinformation (sowohl TEC-REP-Darstellung wie auch Feature-Tree-Darstellung) und zum Initialisieren von Skippy.

- **load.lisp**

lädt die Dateien von Skippy.

- **main.lisp**

enthält

1. den Hauptalgorithmus von Skippy, der zuständig für die Skelettplanauswahl und die Skelettplanverfeinerung ist.
2. den Algorithmus zum Mischen und anschließendem Zusammenfassen von Plänen.

- **methods-skp.lisp**

definiert Methoden zu den Skippy-Klassen. Wird eine neue *atomic-operation*-Klasse in

- **methods-tec-rep.lisp**
definiert Methoden zu den TEC-REP-Klassen. Insbesondere sind hier weitere Methoden zum Berechnen von Geometrieigenschaften, die in den Skelettplänen benötigt werden, abzulegen.
- **plans.lisp**
enthält die Wissensbasis von Skippy, insbesondere also die Skelettpläne. Sie repräsentieren das anwendungsspezifische Wissen. Die Reihenfolge der Skelettpläne beeinflusst die Skelettplanauswahl: Die Suche nach einem Skelettplan, der zum aktuellen *Feature-Tree* und *Op-Mode* paßt, verläuft entgegengesetzt zur Spezifikationsfolge der Skelettpläne. Deshalb sollte ein Plan von hohem Detaillierungsgrad nach einem allgemeineren Plan definiert werden, so daß bei der Skelettplanauswahl der detailliertere Plan dem allgemeineren vorgezogen wird.
- **utils.lisp**
enthält Hilfsfunktionen

Anhang B

Tabellarische Übersicht: CAPP-Systeme

Die folgende Tabelle ist ein Nebenprodukt dieser Studie. Sie enthält eine Übersicht über die im Verlauf dieser Arbeit betrachteten CAPP-Systeme. Eine weitere, umfangreiche Übersicht findet sich in [46], von dem ein Teil der Angaben stammt.

Tabelle B.1: Übersicht CAPP-Systeme

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
APPAS	77	Wysk, Purdue University, USA	1. generatives CAPP-System, Entscheidungsbaum	[120]
ASUPLAN	87	Lin, L./Bedworth, D .D., Arizona State University, Tempe, USA	Gruppentechnologie	[95]
AUTAP	80	Eversheim, Werkzeugmaschinen und Betriebslehre (WZL), RWTH Aachen	Entscheidungstabelle, Beschreibungssprache für das Werkstück, enthält geometrische und technologische Information	[75]
AUTOCAP	77	El-Midany/Darbyshire, University of Manchester Institute of Science and Technology (UMIST), Manchester, UK	Algorithmus zum Schätzen der Fertigungskosten und der Fertigungszeit, Drehteile, BASIC (später: PASCAL)	[119]
AUTOPLAN	80	MRA, USA	variant/generativ, eigene Eingabesprache, Drehteile, FORTRAN	[118]

weiter nächste Seite

Fortsetzung

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
AVOPLAN	84	Tönshoff, IFW der Uni Hannover	variant/generativ, Clusteranalyse	[105]
AVOGEN	89	Anders/Schaele/Prack, IFW der Uni Hannover	generativ, state-space Ansatz, Rückwärtsplanen, Werkstückmodell, Werkstattmodell, Planermodell (Steuerungsstruktur), LISP	[61]
BYUPLAN	85	Brigham Young University, Provo, Utah	Gruppentechnologie, Entscheidungsbaum	[70]
CADCAM	81	Chang, T. C./Wysk, Virginia Polytechnic Institute and State University	Erweiterung von APPAS, generativ	[69]
CAMEX	87	Uni Tel Aviv/IET	geometric reasoning, natürlichsprachliche Erfassung von Regeln, Erklärungskomponente, Lisp	[72]
CAPP	76	Computer Aided Manufacturing- International (CAM-I), Arlington, Texas, USA	1. CAPP-System, variant, Gruppentechnologie	[96]
CAPSY	78	Spur, IPK der TU Berlin	generativ, hierarchische Struktur, Dialog-orientiert	[109]
CHAMP	88	Mielke, TU Braunschweig	Interpretation von IGES-Dateien, Metaregeln, Heuristische Suche	[66]
CMPP	82	Bertelsen, United Technologies Research Center (UTRC), USA	generativ, Entscheidungsmodell, zylindrische Teile, FORTRAN, Schnittstellen zu vielen CAD und CAM Systemen von amerikanischen Luftfahrtgesellschaften	[106]

weiter nächste Seite

Fortsetzung

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
COATS	86	Giusti, University of Pisa	generativ, Werkzeugauswahlkomponente von PICAP für rotationssymmetrische Teile, PROLOG	[77]
CUTTECH/ CUTPLAN	85	Barkocy/Zdeblick/ Tulkoff, MRAT (USA)	variant/generativ, wissensbasierte Planung der Arbeitsschritte	[121]
DISAP	81	Weill/Eversheim, Werkzeugmaschinen und Betriebslehre (WZL),RWTH Aachen	Erweiterung von AUTAP, generativ, alle Werkstückformen	[59]
EXCAP-P	86	Wright/Davies/Joseph, University of Manchester Institute of Science and Technology (UMIST),Manchester, UK	Nachfolgesystem von AUTOCAP, EXCAP-A und EXCAP-Y, generativ, syntaktisch organisierte Feature Recognition ausgehend von einer Beschreibung des Werkstückes in Prolog, discrimination net, Forward Chaining, additive Operatoren, Backward Planning	[87]
FBPP	85	Bond/Chang, K. J., University of California, Los Angeles	Feature Recognition aus 3D-CAD-System, Feature \Rightarrow Fertigungsverfahren, Constraints zur Clustersuche, Blackboard-Architektur	[68]
FEXCAPP	89	Lee/Lee/Lee, Seoul University	generativ, Feature Recognition auf Basis von attributierten Nachbarschaftsgraphen mittels ROMULUS (geometrisches Modell: B-REP) und OPS5 (Forward Reasoning), Heuristiken zum Bestimmen der Arbeitsvorgangsfolge mittels Hill-Climbing-Suchverfahren	[91]

weiter nächste Seite

B Tabellarische Übersicht: CAPP-Systeme

Fortsetzung

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
FREXPP	84	Kung, H. K., Oklahoma State University	Feature Recognition	[89]
GAPPS	84	Kung, J. S., Arizona State University	spezielles CAD-System eingebettet in Prozeßplaner	[90]
GARI	81	Descotte/Latombe, Uni. Grenoble	1. KI-basiertes System, generativ, sukzessive Verfeinerung durch	[71]

Fortsetzung

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
KAPLAN	89	Giusti/Santochi/Dini, University of Pisa	Komponente von PICAP zur Generierung von Drehbearbeitungssequenzen, gewichtete Regeln, generiert alle möglichen Pläne	[78]
KAPPS	86	Iwata/Fukuda, Kobe University, Japan	generativ, wissensbasiert, Know-how repräsentiert durch Frames, Wissensakquisitionskomponente, CAD-Schnittstelle, Drehteile und prismatische Teile, LISP, SMALLTALK	[86]
Machinist	87	Hayes, C. C., Carnegie Mellon University, Pittsburgh	erkennt und behandelt positive und negative Interaktionen zwischen Features, Vergleich der Arbeitspläne mit Plänen, die von menschlichen Arbeitsplanern erstellt wurden	[54, 79]
MICAPP	88	Bok/Nee, University of Singapore	geometric reasoning, state-space Ansatz, PROLOG	[67]
MIPLAN	76	Organization for Industrial Research (OIR) und General Electric, USA	variant, basiert auf Kodierungssystem MICLASS	[81]
MULTI- CAPP II	86	Organization for Industrial Research (OIR), USA	neue Version von MIPLAN, variant, mehrere Moduln, die alle eine einzigste Datenbank benutzen, alle Teileklassen	[103]
OPEX	86	FME Ljubljana	generativ, Reihenfolgeplanung für Drehteile, PROLOG	[76]
PICAP	86	Santochi/Giusti, University Pisa	generativ, produktionsorientierte Konstruktion durch eigenes CAD-Modul, nur rotationssymmetrische Teile	[108]

weiter nächste Seite

Fortsetzung

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
PIM	91	Bernardi/Klauck/- Legleitner, DFKI, Kaiserslautern	syntaxbasierte Feature Recognition, Skelettplanen (Skippy), LISP, CLOS	[65, 92, 88]
PROPEL	87	Tsang, J. P.	Weiterentwicklung von GARI, Werkstückdarstellung durch Feature-Hierarchie und Verhältnishierarchie	[113]
PROPLAN	86	Mouleeswaran, PERA, GB	Integration von CAD-System und Expertensystem, Graphensuche: Knoten = Werkstückzustand, Kante = Arbeitsvorgang, natürlichsprachliche Erklärungskomponente, nur geeignet für einfache Werkstücke (z.B. rotationssymmetrische Teile)	[104]
QTC	88	Chang, T. C.	Feature-basiertes Design, Erkennen von Fertigungsfeatures durch geometric reasoning, Hierarchical Knowledge Clustering, Backward Planning, prismatische Teile, KEE	[49]
ROUND	84	van Houten, Twente University, NL	generative Planung von Drehoperationen, heuristische Suche	[80]
SEQUENCE	89	Ssemakula/Rangachar, University of Maryland, USA	Erweiterung von ICAPP, Optimierung der Fertigungszeit, LISP	[111]
SIPP	85	Nau, University of Maryland	lokaler Ansatz, Kontrollstrategie: branch-and-bound, Feature-Hierarchie durch Frames dargestellt, PROLOG	[101]

weiter nächste Seite

Fortsetzung

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
SIPPS	86	Liu, Y. S./Allen, R., Southampton, UK	generativ, wissensbasiert, interaktive Spezifikation von Features, alle Werkstückklassen, FORTRAN	[97]
SIPS	86	Nau/Gray, University of Maryland	Nachfolgesystem von SIPP, Hierarchical Knowledge Clustering (Kontrollwissen als Framehierarchie + globale Kontrollstrategie), Backward Planning, LISP	[102]
TIPPS	82	Chang, T. C./Wysk, Virginia Polytechnic Institute and State University	Nachfolgesystem von APPAS und CADCAM, Form-basiertes Design erzeugt Werkstück in B-Rep-Darstellung, lokaler Ansatz, volumenorientiert, prozedurales Wissen: Regeln (PKI-Sprache), prismatische Teile, FORTRAN	[50]
TOLTEC	88	Tsatsoulis/Kashyap, Purdue University, West Lafayette	speicherbasierte Inferenz, lernt aus dem Scheitern einer Vorhersage, Skelettplantechnik, hierarchisch, dynamisch generierte constraints steuern Planung	[58]
TOM	82	Matsushima, University of Tokyo	generativ, lokaler Ansatz, regelbasiert, Alpha-Beta-Suchverfahren nach [29], Backward Reasoning (additive Operationen, Forward Chaining), nur Löcher, PASCAL	[98]

weiter nächste Seite

Fortsetzung

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
TURBO-CAPP	87	Wang/Wysk, Penn. State University	generativ, Feature-Erkennung aus CAD-System (AUTOCAD), Wissensakquisitionskomponente ⇒ lernt dazu, Kombination von regel- und framebasiertem Ansatz für Wissensbasis, 3 Ebenen: Faktenebene, Ebene der Inferenzregeln und Metawissensebene, PROLOG	[117]
WAP	90	Schulz/Humm, CIM-Zentrum, Kaiserslautern	generative Planung und fallbasierte Planung, ein Bauteilmodell für alle CIM-Moduln, formelementbasierte Werkstückrepräsentation, KEE (LISP)	[82]
XCUT	86	Hummel/Brooks, Allied Corporation, BKC, Kansas City	Repräsentation des Planungswissen durch Regeln, objektorientierte Darstellung (FLAVORS) des Werkstückes, Feature-Taxonomie	[83]
XMAP	87	Inui, University of Tokyo	generativ, Prädikatenlogik, Expertensystem mit CAD-Modell-Schnittstelle, prismatische Teile, LISP	[84]
XPLANE	88	van't Erve/Kals, Twente University, NL	generativ, Feature-Extraktion aus einem Volumenmodell, A*-Algorithmus: Knoten = Werkstückzustand, Kante = Arbeitsvorgang, Heuristik: Kosten = Zahl der Operationen bis zu einem Knoten, additive Operationen, Backward Planning, Forward Chaining, Produktionsregeln, prismatische Teile, FORTRAN	[73]

weiter nächste Seite

Fortsetzung

CAPP-Systeme				
Name	J.	Entwickler (Autor), Institut (Firma, Ort)	Planungsmethode und an- dere Merkmale	Literatur
XPS-1	83	United Technologies Research Center (UTRC), und Computer Aided Manufacturing- International (CAM-I), USA	Nachfolgesystem zu CMPP, variant/generativ, Entscheidungsmodell, Datenlexikon, Relationale Datenbank, alle Werkstückklassen, FORTRAN	[107]
XPS-E	84	Tsang, J. P., LIFIA Laboratory, Frankreich	Weiterentwicklung von GARI, Feature-basierte Werkstückbeschreibung durch den Benutzer, constraint propagation, Opportunistic Combination of Tentative Plans (OCTP), hierarchisch, General Purpose/Specific Purpose Architektur, Erklärungskomponente, LISP	[112]

Literaturverzeichnis

[A] KI-Literatur

- [1] J. F. Allen. Towards a general theory of action and time. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 464–479. Morgan Kaufmann, Palo Alto, CA, 1984.
- [2] J. F. Allen and J. A. Koomen. Planning using a temporal world model. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 559–565. Morgan Kaufmann, Palo Alto, CA, 1983.
- [3] A. Barr and E. A. Feigenbaum. *The Handbook of Artificial Intelligence, Volume I*. Pitman Books, London, 1981.
- [4] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32:333–377, 1987.
- [5] L. D. Erman, F. Hayes-Roth, V. R. Lesser, and D. R. Reddy. The HEARSAY-II speech-understanding system: integrating knowledge to resolve uncertainty. *ACM Computing Surveys*, 12(2), 1980.
- [6] R. E. Fikes, P. E. Hart, and N. J. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3:251–288, 1972.
- [7] R. E. Fikes, P. E. Hart, and N. J. Nilsson. Some new directions in robot problem solving. *Machine Intelligence*, 7, 1972.
- [8] P. E. Friedland. Knowledge-based experiment design in molecular genetics. Technical Report STAN-CS-79-771, Stanford University, 1979.
- [9] P. E. Friedland and Y. Iwasaki. The concept and implementation of skeletal plans. *Journal of Automated Reasoning*, 1:161–208, 1985.
- [10] M. P. Georgeff. Planning. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 5–25. Morgan Kaufmann, Palo Alto, CA, 1987.
- [11] K. J. Hammond. *Case-Based Planning, Viewing Planning as a Memory Task*. Academic Press, San Diego, CA, 1989.

-
- [12] P. Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on System Science and Cybernetics*, 4(2):100–107, 1968.
- [13] P. J. Hayes. In defense of logic. In *Proc. of the 5th IJCAI*, pages 559–565, Cambridge, MA, 1977.
- [14] B. Hayes-Roth. A blackboard architecture for control. *Artificial Intelligence*, 26:251–321, 1985.
- [15] B. Hayes-Roth and F. Hayes-Roth. A cognitive model of planning. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 245–262. Morgan Kaufmann, Palo Alto, CA, 1979.
- [16] J. Hendler, A. Tate, and M. Drummond. AI planning: systems and techniques. *AI Magazine*, Summer 1990:61–77, 1990.
- [17] J. Hertzberg. Planerstellung-Methoden der Künstlichen Intelligenz. *Informatik-Spektrum*, 9:149–161, 1986.
- [18] J. Hertzberg. *Planen, Einführung in die Planerstellungsmethoden der künstlichen Intelligenz*. BI-Wissenschaftsverlag, 1989.
- [19] S. Hölldobler and J. Schneeberger. A new deductive approach to planning. *New Generation Computing*, 8:225–244, 1990.
- [20] P. Jackson. *Expertensysteme: eine Einführung*. Addison-Wesley Verlag, Bonn, 1987.
- [21] S. Kambhampati and J. Hendler. Flexible reuse of plans via annotation and verification. In *Proc. of the 5th IEEE Conference on Applications of Artificial Intelligence*, Miami, Florida, 1989.
- [22] N. A. Kartam and D. E. Wilkins. Towards a foundation for evaluating AI planners. *AI EDAM*, 4, 1990.
- [23] J. Kolodner. Case-based problem solving. In *Proc. of the 4th International Workshop on Machine Learning*, UC Irvine, 1987.
- [24] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence, Vol. 4*, pages 463–502. Edinburgh University Press, 1969.
- [25] D. Miller, R. J. Firby, and T. Dean. Deadlines, travel time, and robot problem solving. In *Proc. of the 9th IJCAI*, pages 1052–1054, Los Angeles, CA, 1985.
- [26] S. Minton. Interacting goals and their use. In *Proc. of the 9th IJCAI*, pages 596–599, Los Angeles, CA, 1985.
- [27] S. Minton, J. Carbonell, C. Knoblock, D. R. Kuokka, C. Etzioni, and Y. Gil. Explanation-based learning: A problem-solving perspective. *Artificial Intelligence*, 40:63–118, 1989.

- [28] A. Newell and H. A. Simon. GPS, a program that simulates human thought. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 59–66. Morgan Kaufmann, Palo Alto, CA, 1963.
- [29] N. J. Nilsson. *Principles of Artificial Intelligence*, chapter 7,8. Tioga Publishing Co., P.O. Box 98, Palo Alto, CA 94302, 1980.
- [30] F. Puppe. *Einführung in Expertensysteme*, chapter 11, pages 92–98. Springer Verlag, Berlin, Heidelberg, 1988.
- [31] M. M. Richter. Planung in Wissensbasierten Systemen. In W. Brauer and W. Wahlster, editors, *Wissensbasierte Systeme*, pages 224–232. Springer Verlag, Berlin, Heidelberg, 1987.
- [32] M. M. Richter. *Prinzipien der künstlichen Intelligenz*. B. G. Teubner, Stuttgart, 1989.
- [33] E. D. Sacerdoti. Planning in a hierarchy of abstraction spaces. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 98–108. Morgan Kaufmann, Palo Alto, CA, 1974.
- [34] E. D. Sacerdoti. The nonlinear nature of plans. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 162–170. Morgan Kaufmann, Palo Alto, CA, 1975.
- [35] R. C. Schank. *A theory of reminding and learning in computers and people*. Cambridge University Press, Cambridge, 1982.
- [36] M. Stefik. Planning and meta-planning (MOLGEN: Part 2). *Artificial Intelligence*, 16:141–170, 1981.
- [37] M. Stefik. Planning with constraints (MOLGEN: Part 1). *Artificial Intelligence*, 16:111–140, 1981.
- [38] G. J. Sussman. A computational model of skill acquisition. Technical Report AI TR-297, Artificial Intelligence Lab, Cambridge, Massachusetts, 1973.
- [39] A. Tate. Interacting goals and their use. In *Proc. of the 4th IJCAI*, pages 215–218, Tbilisi, USSR, 1975.
- [40] A. Tate. Generating project networks. In *Proc. of the 5th IJCAI*, pages 888–893, Cambridge, MA, 1977.
- [41] A. Tate, J. Hendler, and M. Drummond. A review of AI planning techniques. In J. F. Allen, J. Hendler, and A. Tate, editors, *Readings in Planning*, pages 26–49. Morgan Kaufmann, Palo Alto, CA, 1990.
- [42] S. Vere. Planning in time: windows and durations for activities and goals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(3):246–267, 1983.
- [43] R. Waldinger. Achieving several goals simultaneously. *Machine Intelligence*, 8:94, 1977.

- [44] D. H. D. Warren. WARPLAN: a system for generating plans. Technical report, Dept. of Computational Logic, Edinburgh Univ., 1974.
- [45] D. E. Wilkins. Domain-independent planning: Representation and plan generation. *Artificial Intelligence*, 22:269–301, 1984.

[B] Allgemeine Arbeitsplanungsliteratur

- [46] L. Alting and H. Zhang. Computer aided process planning: the state-of-art survey. *International Journal of Production Research*, 27(4):553–585, 1989.
- [47] H. R. Berenji and B. Khoshnevis. Use of artificial intelligence in automated process planning. *Computers in Mechanical Engineering*, September:47–55, 1986.
- [48] H. J. Bullinger, J. Warschat, and K. Lay. *Künstliche Intelligenz in Konstruktion und Arbeitsplanung*. Verlag Moderne Industrie, Landsberg/Lech, Germany, 1989.
- [49] T. C. Chang. *Expert Process Planning for Manufacturing*. Addison-Wesley, 1990.
- [50] T. C. Chang and R. A. Wysk. *An introduction to automated process planning systems*. Prentice Hall, Englewood, N.J., 1985.
- [51] Y.P. Cheung and A. L. Dowd. Artificial intelligence in process planning. *Computer Aided Engineering*, 5(4):153–156, 1988.
- [52] W. Eversheim. *Organisation in der Produktionstechnik*. VDI Verlag, Düsseldorf, 1989.
- [53] C. C. Hayes. Observing machinists' planning methods: Using goal interactions to guide search. In *Proceedings of the Ninth Annual Conference on Cognitive Science*, pages 952–958, Seattle, 1987.
- [54] C. C. Hayes. A model of planning for plan efficiency: Taking advantage of operator overlap. In *Proc. of the 11th IJCAI*, pages 949–953, Detroit, MI, 1989.
- [55] G. Jüttner and H. Feller. *Entscheidungstabellen und wissensbasierte Systeme*. Oldenbourg, Germany, 1989.
- [56] D. S. Nau, Q. Yang, and J. Hendler. Planning for multiple goals with limited interactions. In *Proc. of the 5th IEEE Conference on Applications of Artificial Intelligence*, Miami, Florida, 1989.
- [57] M. Schaele and K. Hellberg. Wissensbasierte Generierung von Arbeitsgangfolgen. *VDI-Z*, September:57–62, 1987.
- [58] C. Tsatsoulis and R. L. Kashyap. A system for knowledge-based process planning. *Artificial Intelligence in Engineering*, 3(2):61–75, 1988.
- [59] R. Weill. Survey of computer-aided process planning systems. *Annals of the CIRP*, 31(2), 1982.
- [60] Q. Yang, D. S. Nau, and J. Hendler. Optimizing multi-goal plans with limited interactions. Computer Science Department, University of Maryland, College Park, MD 20742, 1988.

[C] Literatur zu den CAPP-Systemen

- [61] N. Anders, M. Schaele, and K. W. Prack. AVOGEN - wissensbasierte Generierung von Arbeitsvorgangsfolgen. *VDI-Z*, 4:49–52, 1990.
- [62] A. Becker. The window tool kit. Dokument, D-90-06, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Postfach 20 80, D-6750 Kaiserslautern, 1990.
- [63] K. Becker, C. Klauck, and J. Schwagereit. Eine Erweiterung des D-PATR zur Feature-Erkennung in CAD/CAM. Technical Memo TM-91-12, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Postfach 20 80, D-6750 Kaiserslautern, October 1991.
- [64] A. Bernardi, C. Klauck, and R. Legleitner. TEC-REP: Repräsentation von Geometrie- und Technologieinformationen. Document, D-91-07, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Postfach 20 80, D-6750 Kaiserslautern, june 1991.
- [65] A. Bernardi, C. Klauck, and R. Legleitner. PIM: Skeletal Plan based CAPP. In *International Conference on Manufacturing Automation*, Hongkong, 1992.
- [66] U. Berr and T. Mielke. Entwicklung eines Expertensystems zur automatischen Arbeitsplanung. *Werkstattstechnik*, 78:243–250, 1988.
- [67] S. H. Bok and A. Y. C. Nee. MICAPP: A microcomputer-based process planning system. *Journal of Mechanical Working Technology*, 17:21–31, 1988.
- [68] A. H. Bond and K. J. Chang. Feature-based process planning for machined parts. In M. D. Patton, editor, *Proc. of the 1988 ASME International Computers in Engineering Conference and Exhibition*, volume 1, pages 571–576, San Francisco, California, 1988. Manufacturing Engineering Program, University of California Los Angeles.
- [69] T. C. Chang and R. A. Wysk. Interfacing CAD/automated process planning. *AIIE Transactions*, 13:223–233, 1981.
- [70] Computer Aided Manufacturing Laboratory, Brigham Young University, Provo, Utah. *BYUPLAN Generative Process Planning Systems User Manual*, 1985.
- [71] Y. Descotte and J. C. Latombe. Making compromises among antagonist constraints in a planner. *Artificial Intelligence*, 27:183–217, 1985.
- [72] O. Eliyahu, L. Zaidenberg, and M. Ben-Bassat. CAMEX: An expert system for process planning on CNC machines. In *Proc. of AAAI-87*, pages 794–798, Seattle, WA, 1987.
- [73] A. H. van't Erve and H. J. J. Kals. XPLANE, a generative computer aided process planning system for part manufacturing. *Annals of the CIRP*, 35/1:325–329, 1986.
- [74] W. Eversheim, H. Fuchs, and K. H. Zons. Anwendung des Systems AUTAP zur Arbeitsplanerstellung. *Industrie-Anzeiger*, 55:29–33, 1980.

- [75] W. Eversheim, B. Holz, and K. H. Zons. Application of automatic process planning and NC programming. In *Proc. of AUTOFACT WEST, Society of Manufacturing Engineers*, pages 779–800, Anaheim, Calif., 1980.
- [76] M. Gams. OPEX — an expert system for CAPP. In *The 6th International Workshop on Expert Systems and their Applications*, Avignon, France, April 1986.
- [77] F. Giusti. COATS: an expert module for optimal tool selection. *Annals of the CIRP*, 35(1), 1986.
- [78] F. Giusti, M. Santochi, and G. Dini. KAPLAN: a knowledge-based approach to process planning of rotational parts. *Annals of the CIRP*, 38/1:481–484, 1989.
- [79] C. C. Hayes. Using goal interactions to guide planning. In *Proc. of AAAI-87*, pages 224–228, Seattle, WA, 1987.
- [80] M. van Houten. Strategy in generative planning of turning processes. *Annals of the CIRP*, 35/1:331–335, 1986.
- [81] A. Houtzeel. The MICLASS system. In *Proc. of CAM-I's Executive Seminar Coding, Classification, and Group Technology for Automated Planning*, Arlington, Texas, 1976.
- [82] B. Humm, C. Schulz, M. Radtke, and G. Warnecke. A system for case-based process planning. Technical report, Universität Kaiserslautern, Postfach 3049, D-6750 Kaiserslautern, 1990.
- [83] K. E. Hummel and S. L. Brooks. Symbolic representation of manufacturing features for an automated process planning system. In *Bound Volume of the Symposium on Knowledge Based Expert Systems for Manufacturing*, pages 233–243, Anaheim, Calif., 1986. Winter Annual Meeting of the ASME.
- [84] M. Inui, H. Suzuki, F. Kimura, and T. Sata. Extending process planning capabilities with dynamic manipulation of product models. In *Proceedings of the 19th CIRP International Seminar on Manufacturing Systems*, pages 273–280, Pennsylvania State University, June 1987.
- [85] N. R. Iudica. GUMMEX - ein Expertensystem zur Generierung von Arbeitsplänen für die Fertigung. *Nachrichten für Dokumentation*, 36(1):22–27, 1985.
- [86] K. Iwata and Y. Fukuda. KAPPS: Know-how and knowledge assisted production planning system in the machining shop. In *Proceedings of the 19th CIRP International Seminar on Manufacturing Systems*, Penn. State, USA, June 1987.
- [87] A. T. Joseph and B. J. Davies. Knowledge based process planning system for turned components. *International Journal of Adv. Manufacturing Technology*, 5:52–65, 1990.
- [88] C. Klauck, A. Bernardi, and R. Legleitner. FEAT-REP: representing features in CAD/CAM. Research Report RR-91-20, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, Postfach 20 80, D-6750 Kaiserslautern, June 1991.

- [89] H. K. Kung. *An investigation into the development of process plans from solid geometric Modeling representation*. PhD thesis, School of Industrial Engineering, Oklahoma State University, Stillwater, Oklahoma, 1984.
- [90] J. S. Kung. Integrated CAD and CAM — a study of machined cylindrical parts design. Unpublished MS Engineering Report, Department of Industrial and Management Systems Engineering, Arizona State University, Tempe, Arizona, August 1984.
- [91] K. L. Lee, J. W. Lee, and J. M. Lee. Pattern recognition and process planning prismatic workpieces by knowledge based approach. *Annals of the CIRP*, 38/1:485–488, 1989.
- [92] R. Legleitner, A. Bernardi, and C. Klauck. Akquisition und Repräsentation von technischem Wissen für Planungsaufgaben im Bereich der Fertigungstechnik. *VDI Berichte*, 903, 1991.
- [93] S. Lewandowski and G. Stuckmann. *CAD-Systeme in den USA. Bericht von der IMTS '78 in Chicago und einer CAD/CAM-Konferenz in Los Angeles*, volume 4, pages 170–173. *ZwF* 74, 1979.
- [94] R. K. Li and D. D. Bedworth. A semi-generative approach to computer-aided process planning using group technology. *Computers and Industrial Engineering*, 14-2:127–137, 1988.
- [95] L. Lin. A classification and coding scheme and computer-aided process planning system for rotational and gear parts using DCLASS. Unpublished Master of Science in Engineering Research Report, Department of Industrial and Management Systems Engineering, Arizona State University, Tempe, Arizona, 1986.
- [96] C. H. Link. CAPP, CAM-I automated process planning system. In *Proc. of the 1976 NC Conference, CAM-I*, Arlington, Texas, 1976.
- [97] Y. S. Liu and R. Allen. A proposed synthetic, interactive process planning system. In *Proceedings of the International Conference on Computer Aided Production Engineering*, pages 201–208, Edinburgh, 1986.
- [98] K. Matsushima, N. Okada, and T. Sata. The integration of CAD and CAM by application of artificial-intelligence techniques. *Annals of the CIRP*, 31/1:329–332, 1982.
- [99] J. Mauss. *Entwicklung und Implementierung eines heuristisch gesteuerten, chart-basierten Parsers für attributierte Graphsprachen*. Diplomarbeit, Universität Kaiserslautern, Postfach 3049, D-6750 Kaiserslautern, 1992.
- [100] Metalinstitut TNO, Postfach 541, Apeldoorn, Niederlande. *MICLASS, Systembeschreibung*.
- [101] D. S. Nau. SIPP reference manual. Technical report 1515, Computer Science Department, University of Maryland, June 1985.

- [102] D. S. Nau and M. Gray. Hierarchical knowledge clustering: a way to represent and use problem-solving knowledge. In J. A. Hendler, editor, *EXPERT SYSTEMS: THE USER INTERFACE*, pages 81–98. Ablex Publishing Corporation, Norwood, New Jersey, 1987.
- [103] OIR (Organization for Industrial Research) Product News. *Multi-II group technology system*, 1987.
- [104] R. H. Phillips, X. D. Zhou, and C. Mouleeswaran. An artificial intelligence approach to integrating CAD and CAM through generative process planning. In *Proceedings, ASME International Computers in Engineering Conference*, Las Vegas, 1984.
- [105] K. W. Prack. AVOPLAN - Ein Gesamtkonzept zur rechnerunterstützten Arbeitsplanung. *Industrie*, 3/4:20–21, 1986.
- [106] C. F. Sack, Jr. Computer managed process planning - a bridge between CAD and CAM. In *The CASA/SME Autofact 4 Conference*, November 1982.
- [107] C. F. Sack, Jr. CAM-I's experimental planning system, XPS-1. In *Autofact 5 Conference Proceedings*, Detroit, Michigan, USA, November 1983.
- [108] M. Santochi and F. Giusti. PICAP: a fully integrated package for process planning of rotational parts. In *18th CIRP Seminar on Manufacturing Systems*, Stuttgart, 1986.
- [109] G. Spur, W. Kunzendorf, and W. Stuckmann. Automatisierte Arbeitsplanung für die Einzelteil- und Kleinserienfertigung. In *Proceedings, CIRP Seminars on Manufacturing Systems*, volume 5, 1976.
- [110] M. E. Ssemakula. Evaluation of an interactive computer aided process planning system (ICAPP) for non-rotational parts. Master's thesis, UMIST, Manchester, UK, 1981.
- [111] M. E. Ssemakula and R. M. Rangachar. The prospects of process sequence optimization in CAPP systems. *Computers and Industrial Engineering*, 16, No.1:161–170, 1989.
- [112] J. P. Tsang. Genericity in expert process planning systems. In D. Sriram and R. Adey, editors, *Applications of Artificial Intelligence in Engineering Problems, Proceedings of the 1st International Conference*, volume I, pages 621–637, Southampton University, U.K., April 1986. Springer Verlag, Berlin, Heidelberg, New York.
- [113] J. P. Tsang. The PROPEL process planner. In *Proceedings of the 19th CIRP International Seminar on Manufacturing Systems*, pages 71–77, Pennsylvania State University, 1987.
- [114] J. Tulkoff. Lockheed's GENPLAN. In *Proc. of 18th Numerical Control Society Annual Meeting and Technical Conference*, pages 417–421, Dallas, Texas, 1981.
- [115] M. B. Unger and S. R. Ray. Feature-based process planning in the AMRF. *Computers in Engineering, American Society of Mechanical Engineers*, pages 563–569, 1988.

- [116] N. Vissa. A frame-based generative process planning system for machining prismatic parts. Unpublished MS Thesis, Purdue University, 1987.
- [117] H.-P. Wang and R. A. Wysk. A knowledge-based computer aided process planning system. In *Proceedings of the 19th CIRP International Seminar on Manufacturing Systems*, Penn. State, USA, 1987.
- [118] P. M. Wolfe. Computer-aided process planning is link between CAD and CAM. *Industrial Engineering*, 17(8), 1985.
- [119] A. J. Wright. EXCAP und ICAPP: Integrated knowledge based systems for process-planning components. In *Proceedings of the 19th CIRP International Seminar on Manufacturing Systems*, Pennsylvania State, USA, June 1987.
- [120] R. A. Wysk. *An automated process planning and selection program*. PhD thesis, Purdue University, West Lafayette, Ind., 1977.
- [121] W. J. Zdeblick. CUTPLAN/CUTTECH a hybrid computer aided process planning and operation planning system. In *Proc. of the 1st CIRP Seminar on CAPP*, Paris, France, January 1985.

Index

- Überkreuzkonflikt, 12
- A*-Algorithmus, 9
- ABSTRIPS, 12
- agent, 18
- Allen, J. F., 18
- APPAS, 25, 34, 65
- ARC-TEC-Projekt, 5, 52
- ASUPLAN, 65
- AUTAP, 25, 65
- AUTOCAP, 65
- AUTOPLAN, 65
- AVOGEN, 66
- AVOPLAN, 66
- Backward
 - Chaining, 30
 - Planning, 30
 - Reasoning, 30
- Blackboard, 17
- Bottom-Up Ansatz, 30
- branch and bound, 9, 34
- BYUPLAN, 66
- CADCAM, 66
- CAM-I, 66, 73
- CAMEX, 66
- CAPP, 34, 66
- CAPSY, 66
- Case-Based Planning, 20
- Chaining
 - Backward, 30
 - Forward, 30
- CHAMP, 66
- Chang, T. C., 24, 34, 36
- Chapman, D., 19
- CHEF, 20
- CMPP, 66
- COATS, 67
- CODE, 23
- constraint, 14, 19
 - formulation, 15
 - propagation, 15
 - satisfaction, 15
- control problem, 17
- critics, 14
- CUTTECH/CUTPLAN, 67
- datengetrieben, 30
- deduktives Planen, 9
- DEVISER, 18
- DFKI, 5, 52
- DISAP, 67
- distributed planning, 18
- divide et impera, 45
- double cross conflict, 12
- EBL, 19
- Entscheidungsbaum, 25
- Entscheidungstabelle, 26
 - eindeutige, 26
 - Eintreffer-, 26
 - mehrdeutige, 26
 - Mehrtreffer-, 26
 - Verbund, 26
- Eversheim, W., 21
- EXCAP-P, 37, 48, 67
- Explanation-Based Learning, 19
- fallbasiertes Planen, 20, 47
- FBPP, 67
- FEAT-PATR, 54
- Feat-Rep-Graphics-System, 52
- Feature, 25
 - Extraction, 25
 - Hierarchie, 34
 - interaction, 41
- feature
 - refinement, 36
- Feature-Tree, 52

- FEXCAPP, 67
 FirstCut, 28
 fluent, 9
 FORBIN, 18
 Forward
 Chaining, 30
 Planning, 30
 Reasoning, 30
 frame problem, 9
 FREXPP, 68
 Friedland, P. E., 15

 Gabelkonflikt, 12
 GAPPS, 68
 GARI, 25, 36, 68
 GEFPOS, 68
 General Purpose/Specific Purpose Kernel
 Architektur, 37
 generative Arbeitsplanung, 23
 GENPLAN, 25, 68
 geometric reasoning, 28, 36
 geometrisches Schließen, 28
 globaler Ansatz, 28
 GPS, 9
 Gruppentechnologie, 23
 GUMMEX, 68

 HACKER, 11
 Hayes-Roth, B. and F., 17
 heuristische Suche, 9
 HI-MAPP, 68
 Hierarchical Knowledge Clustering, 34
 hierarchisches Planen, 12, 45
 Hill-Climbing-Suchverfahren, 67

 ICAPP, 68
 interaction
 helpful, 12
 Interaktion
 hilfreiche, 12
 negative, 12, 28, 41
 positive, 12, 41
 von Arbeitsvorgängen, 41
 von Teilproblemen, 11, 12, 14, 15
 IXPRESS, 68

 JULIA, 20
 KAPLAN, 69

 KAPPS, 69
 Konflikt
 Überkreuz-, 12
 elementarer, 12
 Gabel-, 12
 Linear-, 12
 Parallel-, 12
 zusammengesetzter, 12
 Koomen, J. A., 18
 korrektes Planungssystem, 14

 least commitment Strategie, 12
 Linearity Assumption, 11
 Linearitätsannahme, 11
 Linearkonflikt, 12
 lokaler Ansatz, 28

 Machinist, 69
 means-ends analysis, 9, 44
 Meta-Planen, 15, 47
 MICAPP, 69
 MICLASS, 23
 Minton, S., 19
 MIPLAN, 34, 69
 MOLGEN
 von Friedland, 15
 von Stefik, 14
 multi-agent planning, 18
 MULTI-CAPP II, 34, 69

 Nau, D. S., 34
 nichtlineares Planen, 12
 NOAH, 12
 NONLIN, 14

 OIR, 69
 operator overlap, 41
 OPEX, 69
 OPM, 17
 opportunistisches Planen, 17

 Parallelkonflikt, 12
 partial plan search space, 8
 PICAP, 69
 PIM, 5, 48, 52, 70
 Planen, 7
 deduktives, 9
 fallbasiertes, 20, 47
 hierarchisches, 12, 45

- lineares, 44
- Meta-, 15, 47
- mit constraints, 14, 46
- mittels Inferenz, 9, 43
- nichtlineares, 12, 44
- opportunistisches, 17
- Skelett-, 15
- verteiltes, 18
- Planungssystem
 - korrektes, 14
 - vollständiges, 14
- PRIAR, 20
- procedural net, 13
- PRODIGY, 20
- PROPEL, 70
- PROPLAN, 70

- QTC, 28, 36, 70

- Replanning, 18, 43
- resource, 18
- ROUND, 70
- Rückwärts
 - planen, 30
 - schließen, 30
 - verkettung, 30

- Sacerdoti, E. D., 12
- SEQUENCE, 70
- SIPE, 17
- SIPP, 34, 48, 70
- SIPPS, 71
- SIPS, 34, 48, 71
- Situationsabstraktion, 12
- Situationskalkül, 8
- skeletal plan, 15
 - refinement, 15
- Skelettplan, 9, 15
 - auswahl, 15
 - technik, 15, 47
 - verfeinerung, 15
- Skippy, 5, 28, 48, 52, 70
- SPEX, 17

- Operator, 11
 - Rule, 11
- STRIPS-Operator, 44
- Sussman Anomalie, 12
- Sussman, G. J., 11

- Tate, A., 14
- TEC-REP, 52
- Teile-und-Herrsche-Prinzip, 45
- time window, 18
- TIPPS, 25, 28, 34, 71
- TOLTEC, 37, 47, 48, 71
- TOM, 28, 48, 71
- Top-Down Ansatz, 30
- Tsang, J. P., 37
- TURBO-CAPP, 72
- TWEAK, 19

- UMIST, 65, 67, 68
- UTRC, 66, 73

- variante Arbeitsplanung, 23
- Vere, S., 18
- verteiltes Planen, 18
- vollständiges Planungssystem, 14
- volumenorientierter Ansatz, 29
- Vorwärts
 - planen, 30
 - schließen, 30
 - verkettung, 30

- WAP, 47, 72
- Wilkins, D. E., 17
- Wysk, R. A., 34
- WZL, 65, 67

- XCUT, 25, 72
- XMAPP, 72
- XPLANE, 72
- XPS-1, 73
- XPS-E, 37, 73

- Zeitkomplexität
 - der Arbeitsplankombination, 45



**Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH**

**DFKI
-Bibliothek-
PF 2080
D-6750 Kaiserslautern
FRG**

DFKI Publikationen

Die folgenden DFKI Veröffentlichungen sowie die aktuelle Liste von allen bisher erschienenen Publikationen können von der oben angegebenen Adresse bezogen werden.
Die Berichte werden, wenn nicht anders gekennzeichnet, kostenlos abgegeben.

DFKI Publications

The following DFKI publications or the list of all published papers so far can be ordered from the above address.
The reports are distributed free of charge except if otherwise indicated.

DFKI Research Reports

RR-90-07

Elisabeth André, Thomas Rist:

Wissensbasierte Informationspräsentation:

Zwei Beiträge zum Fachgespräch Graphik und KI

1. Ein planbasierter Ansatz zur Synthese illustrierter Dokumente
2. Wissensbasierte Perspektivenwahl für die automatische Erzeugung von 3D-Objektdarstellungen

24 Seiten

RR-90-08

Andreas Dengel: A Step Towards Understanding Paper Documents

25 pages

RR-90-09

Susanne Biundo: Plan Generation Using a Method of Deductive Program Synthesis

17 pages

RR-90-10

Franz Baader, Hans-Jürgen Bürckert, Bernhard Hollunder, Werner Nutt, Jörg H. Siekmann:

Concept Logics

26 pages

RR-90-11

Elisabeth André, Thomas Rist: Towards a Plan-Based Synthesis of Illustrated Documents

14 pages

RR-90-12

Harold Boley: Declarative Operations on Nets

43 pages

RR-90-13

Franz Baader: Augmenting Concept Languages by Transitive Closure of Roles: An Alternative to Terminological Cycles

40 pages

RR-90-14

Franz Schmalhofer, Otto Kühn, Gabriele Schmidt:

Integrated Knowledge Acquisition from Text, Previously Solved Cases, and Expert Memories

20 pages

RR-90-15

Harald Trost: The Application of Two-level Morphology to Non-concatenative German Morphology

13 pages

RR-90-16

Franz Baader, Werner Nutt: Adding Homomorphisms to Commutative/Monoidal Theories, or: How Algebra Can Help in Equational Unification

25 pages

RR-90-17

Stephan Busemann:

Generalisierte Phasenstrukturgrammatiken und ihre Verwendung zur maschinellen Sprachverarbeitung

114 Seiten

RR-91-01

Franz Baader, Hans-Jürgen Bürckert, Bernhard

Nebel, Werner Nutt, Gert Smolka: On the Expressivity of Feature Logics with Negation, Functional Uncertainty, and Sort Equations

20 pages

RR-91-02

Francesco Donini, Bernhard Hollunder, Maurizio Lenzerini, Alberto Marchetti Spaccamela, Daniele Nardi, Werner Nutt: The Complexity of Existential Quantification in Concept Languages

22 pages

RR-91-03

B.Hollunder, Franz Baader: Qualifying Number Restrictions in Concept Languages

34 pages

- RR-91-04**
Harald Trost: X2MORF: A Morphological Component Based on Augmented Two-Level Morphology
 19 pages
- RR-91-05**
Wolfgang Wahlster, Elisabeth André, Winfried Graf, Thomas Rist: Designing Illustrated Texts: How Language Production is Influenced by Graphics Generation.
 17 pages
- RR-91-06**
Elisabeth André, Thomas Rist: Synthesizing Illustrated Documents: A Plan-Based Approach
 11 pages
- RR-91-07**
Günter Neumann, Wolfgang Finkler: A Head-Driven Approach to Incremental and Parallel Generation of Syntactic Structures
 13 pages
- RR-91-08**
Wolfgang Wahlster, Elisabeth André, Som Bandyopadhyay, Winfried Graf, Thomas Rist: WIP: The Coordinated Generation of Multimodal Presentations from a Common Representation
 23 pages
- RR-91-09**
Hans-Jürgen Bürckert, Jürgen Müller, Achim Schupeta: RATMAN and its Relation to Other Multi-Agent Testbeds
 31 pages
- RR-91-10**
Franz Baader, Philipp Hanschke: A Scheme for Integrating Concrete Domains into Concept Languages
 31 pages
- RR-91-11**
Bernhard Nebel: Belief Revision and Default Reasoning: Syntax-Based Approaches
 37 pages
- RR-91-12**
J.Mark Gawron, John Nerbonne, Stanley Peters: The Absorption Principle and E-Type Anaphora
 33 pages
- RR-91-13**
Gert Smolka: Residuation and Guarded Rules for Constraint Logic Programming
 17 pages
- RR-91-14**
Peter Breuer, Jürgen Müller: A Two Level Representation for Spatial Relations, Part I
 27 pages
- RR-91-15**
Bernhard Nebel, Gert Smolka: Attributive Description Formalisms ... and the Rest of the World
 20 pages
- RR-91-16**
Stephan Busemann: Using Pattern-Action Rules for the Generation of GPSG Structures from Separate Semantic Representations
 18 pages
- RR-91-17**
Andreas Dengel, Nelson M. Mattos: The Use of Abstraction Concepts for Representing and Structuring Documents
 17 pages
- RR-91-18**
John Nerbonne, Klaus Netter, Abdel Kader Diagne, Ludwig Dickmann, Judith Klein: A Diagnostic Tool for German Syntax
 20 pages
- RR-91-19**
Munindar P. Singh: On the Commitments and Precommitments of Limited Agents
 15 pages
- RR-91-20**
Christoph Klauck, Ansgar Bernardi, Ralf Legleitner FEAT-Rep: Representing Features in CAD/CAM
 48 pages
- RR-91-21**
Klaus Netter: Clause Union and Verb Raising Phenomena in German
 38 pages
- RR-91-22**
Andreas Dengel: Self-Adapting Structuring and Representation of Space
 27 pages
- RR-91-23**
Michael Richter, Ansgar Bernardi, Christoph Klauck, Ralf Legleitner: Akquisition und Repräsentation von technischem Wissen für Planungsaufgaben im Bereich der Fertigungstechnik
 24 Seiten
- RR-91-24**
Jochen Heinsohn: A Hybrid Approach for Modeling Uncertainty in Terminological Logics
 22 pages
- RR-91-25**
Karin Harbusch, Wolfgang Finkler, Anne Schauder: Incremental Syntax Generation with Tree Adjoining Grammars
 16 pages

RR-91-26

*M. Bauer, S. Biundo, D. Dengler, M. Hecking,
J. Koehler, G. Merziger:*
Integrated Plan Generation and Recognition
- A Logic-Based Approach -
17 pages

RR-91-27

*A. Bernardi, H. Boley, Ph. Hanschke,
K. Hinkelmann, Ch. Klauck, O. Kühn,
R. Legleitner, M. Meyer, M. M. Richter,
F. Schmalhofer, G. Schmidt, W. Sommer:*
ARC-TEC: Acquisition, Representation and
Compilation of Technical Knowledge
18 pages

RR-91-28

Rolf Backofen, Harald Trost, Hans Uszkoreit:
Linking Typed Feature Formalisms and
Terminological Knowledge Representation
Languages in Natural Language Front-Ends
11 pages

RR-91-29

Hans Uszkoreit: Strategies for Adding Control
Information to Declarative Grammars
17 pages

RR-91-30

Dan Flickinger, John Nerbonne:
Inheritance and Complementation: A Case Study of
Easy Adjectives and Related Nouns
39 pages

RR-91-31

H.-U. Krieger, J. Nerbonne:
Feature-Based Inheritance Networks for
Computational Lexicons
11 pages

RR-91-32

Rolf Backofen, Lutz Euler, Günther Görz:
Towards the Integration of Functions, Relations and
Types in an AI Programming Language
14 pages

RR-91-33

Franz Baader, Klaus Schulz:
Unification in the Union of Disjoint Equational
Theories: Combining Decision Procedures
33 pages

RR-91-34

Bernhard Nebel, Christer Bäckström:
On the Computational Complexity of Temporal
Projection and some related Problems
35 pages

RR-91-35

Winfried Graf, Wolfgang Maaß: Constraint-basierte
Verarbeitung graphischen Wissens
14 Seiten

DFKI Technical Memos**TM-91-01**

Jana Köhler: Approaches to the Reuse of Plan
Schemata in Planning Formalisms
52 pages

TM-91-02

Knut Hinkelmann: Bidirectional Reasoning of Horn
Clause Programs: Transformation and Compilation
20 pages

TM-91-03

Otto Kühn, Marc Linster, Gabriele Schmidt:
Clamping, COKAM, KADS, and OMOS:
The Construction and Operationalization
of a KADS Conceptual Model
20 pages

TM-91-04

Harold Boley (Ed.):
A sampler of Relational/Functional Definitions
12 pages

TM-91-05

Jay C. Weber, Andreas Dengel, Rainer Bleisinger:
Theoretical Consideration of Goal Recognition
Aspects for Understanding Information in Business
Letters
10 pages

TM-91-06

Johannes Stein: Aspects of Cooperating Agents
22 pages

TM-91-08

Munindar P. Singh: Social and Psychological
Commitments in Multiagent Systems
11 pages

TM-91-09

Munindar P. Singh: On the Semantics of Protocols
Among Distributed Intelligent Agents
18 pages

TM-91-10

*Béla Buschauer, Peter Poller, Anne Schauder, Karin
Harbusch:* Tree Adjoining Grammars mit
Unifikation
149 pages

TM-91-11

Peter Wazinski: Generating Spatial Descriptions for
Cross-modal References
21 pages

TM-91-12

*Klaus Becker, Christoph Klauck, Johannes
Schwagereit:* FEAT-PATR: Eine Erweiterung des
D-PATR zur Feature-Erkennung in CAD/CAM
33 Seiten

TM-91-13*Knut Hinkelmann:*

Forward Logic Evaluation: Developing a Compiler from a Partially Evaluated Meta Interpreter

16 pages

TM-91-14*Rainer Bleisinger, Rainer Hoch, Andreas Dengel:*

ODA-based modeling for document analysis

14 pages

DFKI Documents**D-91-03***Harold Boley, Klaus Elsbernd, Hans-Günther Hein,**Thomas Krause: RFM Manual: Compiling*

RELFUN into the Relational/Functional Machine

43 pages

D-91-04

DFKI Wissenschaftlich-Technischer Jahresbericht 1990

93 Seiten

D-91-06*Gerd Kamp: Entwurf, vergleichende Beschreibung und Integration eines Arbeitsplanerstellungssystems für Drehteile*

130 Seiten

D-91-07*Ansgar Bernardi, Christoph Klauck, Ralf Legleitner*

TEC-REP: Representation von Geometrie und

D-91-12*Bernd Bachmann:*

HieraCon - a Knowledge Representation System with Typed Hierarchies and Constraints

75 pages

D-91-13

International Workshop on Terminological Logics

*Organizers: Bernhard Nebel, Christof Peltason,**Kai von Luck*

131 pages

D-91-14*Erich Achilles, Bernhard Hollunder, Armin Laux,**Jörg-Peter Mohren: KRIS: Knowledge*

Representation and Inference System

- Benutzerhandbuch -

28 Seiten

D-91-15*Harold Boley, Philipp Hanschke, Martin Harm,**Knut Hinkelmann, Thomas Labisch, Manfred**Meyer, Jörg Müller, Thomas Oltzen, Michael**Sintek, Werner Stein, Frank Steinle:* μ CAD2NC: A Declarative Lathe-Workplanning

Model Transforming CAD-like Geometries into

Abstract NC Programs

100 pages

D-91-16*Jörg Thoben, Franz Schmalhofer, Thomas Reinartz:*

Wiederholungs-, Varianten- und Neuplanung bei der

Fertigung rotations-symmetrischer Drehteile

Technologieinformationen

70 Seiten

D-91-08*Thomas Krause: Globale Datenflußanalyse und horizontale Compilation der relational-funktionalen Sprache RELFUN*

137 Seiten

D-91-09*David Powers, Lary Reeker (Eds.):*

Proceedings MLNLO'91 - Machine Learning of Natural Language and Ontology

211 pages

Note: This document is available only for a nominal charge of 25 DM (or 15 US-\$).**D-91-10***Donald R. Steiner, Jürgen Müller (Eds.):*

MAAMAW'91: Pre-Proceedings of the 3rd

134 Seiten

D-91-17*Andreas Becker:*

Analyse der Planungsverfahren der KI im Hinblick auf ihre Eignung für die Arbeitsplanung

86 Seiten

D-91-18*Thomas Reinartz: Definition von Problemklassen im Maschinenbau als eine Begriffsbildungsaufgabe*

107 Seiten

D-91-19*Peter Wazinski: Objektlokalisierung in graphischen Darstellungen*

110 Seiten

TM-91-13*Knut Hinkelmann:*

Forward Logic Evaluation: Developing a Compiler
 from a Partially Evaluated Meta Interpreter
 16 pages

TM-91-14

Rainer Bleisinger, Rainer Hoch, Andreas Dengel:
 ODA-based modeling for document analysis
 14 pages

DFKI Documents**D-91-03**

*Harold Boley, Klaus Elsbernd, Hans-Günther Hein,
 Thomas Krause:* RFM Manual: Compiling
 RELFUN into the Relational/Functional Machine
 43 pages

D-91-04

DFKI Wissenschaftlich-Technischer Jahresbericht
 1990
 93 Seiten

D-91-06

Gerd Kamp: Entwurf, vergleichende Beschreibung
 und Integration eines Arbeitsplanerstellungssystems
 für Drehteile
 130 Seiten

D-91-07

Ansgar Bernardi, Christoh Klauck, Ralf Leolaitner

D-91-12*Bernd Bachmann:*

HieraCon - a Knowledge Representation System
 with Typed Hierarchies and Constraints
 75 pages

D-91-13

International Workshop on Terminological Logics
*Organizers: Bernhard Nebel, Christof Peltason,
 Kai von Luck*

131 pages

D-91-14

*Erich Achilles, Bernhard Hollunder, Armin Laux,
 Jörg-Peter Mohren:* KRIS: Knowledge
 Representation and Inference System
 - Benutzerhandbuch -
 28 Seiten

D-91-15

*Harold Boley, Philipp Hanschke, Martin Harm,
 Knut Hinkelmann, Thomas Labisch, Manfred
 Meyer, Jörg Müller, Thomas Oltzen, Michael
 Sintek, Werner Stein, Frank Steinle:*
 μ CAD2NC: A Declarative Lathe-Worplanning
 Model Transforming CAD-like Geometries into
 Abstract NC Programs
 100 pages

D-91-16

Jörg Thoben, Franz Schmalhofer, Thomas Reinartz:
 Wiederholungs-, Varianten- und Neuplanung bei der





**Analyse der Planungsverfahren der KI
im Hinblick auf ihre Eignung für die Arbeitsplanung**

Andreas Becker

D-91-17
Document